Robotic Arm-Cutting Edge Sorting Machine for Industrial Optimization

Karri Chiranjeevi [1], GBSR. Naidu [2,*], Shaik S. Akhter [1], Akula T. Rao [3], Vuppula Manohar [4], Santosh K. Gottapu [5], and Asma Fejjari [6]

[1] Electronics and Communication Engineering, A. U. College of Engineering, Andhra University, Visakhaptnam, Andhrapredesh, India
[2] Electronics and Communication Engineering, GMR Institute of Technology, Rajam, Vizianagaram, Andhrapradesh, India
[3] Electronics and Communication Engineering, KITS Akshar Institute of Technology, Yanamadala, Andhra Pradesh, India
[4] Electronics and Communication Engineering, Vaagdevi Engineering College, Warangal, Telangana, India
[5] Department of Civil, GVP College of Engineering, Visakhapatnam, Andhrapradesh, India
[6] Communications Computer Engineering, Faculty of Information Communication Technology, University of Malta, Msida, Malta
Email: chiru404@gmail.com (K.C.); naidu.gbsr@gmrit.edu.in (GBSR.N.); munnashoaib.sk@gmail.com (S.S.A.); akula.trinadha@gmail.com (A.T.R.); manohar_v@vaagdevi.edu.in (V.M.); kumar.santou@gmail.com (S.K.G.); afejj01@um.edu.mt (A.F.)
*Corresponding author

***Abstract*—This paper introduces an innovative Sorting Robotic Arm (SRA) that combines the precision and speed of a Selective Compliance Assembly Robot Arm (SCARA) with the advanced object detection capabilities of the Grounding Dino model. The system enables real-time identification and sorting of various objects with exceptional accuracy, making it suitable for dynamic industrial environments. Leveraging state-of-the-art Artificial Intelligence (AI), the SRA can recognize and classify items efficiently, significantly reducing manual labor and operational errors. The modular design of the robotic arm allows for easy customization and scalability, facilitating seamless integration into a wide range of applications such as recycling facilities, logistics and warehousing, and agricultural seed sorting. The SCARA configuration ensures a compact form factor, fast cycle times, and high repeatability, ideal for precision-driven tasks. By merging robust robotic hardware with intelligent AI-driven perception, this paper proposes a transformative solution to modern sorting challenges. The SRA enhances productivity, accuracy, and efficiency, offering a practical and forward-looking approach to automated material handling, logistics and warehousing, seed sorting in agriculture.**

***Keywords*—Sorting Robotic Arm (SRA), selective compliance assembly robot arm, Grounding Dino model, object detection, object localization**

I. Introduction

In the early days of Industrialization, human workers were the backbone of the manufacturing processes. With absolute physical strength and proficiency, workers operated machinery, assembled products, and performed repetitive tasks in factories around the world. But by world standards human labor is less productive and lacks consistency. Humans may not be as efficient or consistent as machines in performing repetitive tasks, leading to variations in quality and productivity and they also have physical limitations, such as fatigue which can impact their ability to sustain high levels of productivity over extended periods. Certain tasks in industries can pose safety hazards to human workers, exposing them to the risks of injuries. Their decision making can be influenced by biases, emotions leading to errors or suboptimal outcomes in industrial processes. Compliance with labor laws, regulations and workplace safety standards adds complexity and administrative burden to businesses employing human labor in industries. To overcome all these problems, the true revolution came with the emergence of Industrial Robotics in the mid-20th century. With the invention of Programmable Logic Controllers (PLC) and robotic arms, industries gained the ability to automate complex tasks previously deemed difficult. Robots can perform tasks with consistent accuracy and precision without errors. By automating hazardous tasks, they can reduce the risk of workplace accidents and injuries. Unlike humans, robots can operate continuously without the need for breaks, shifts. Modern robotic systems are designed to be highly flexible and programmable, allowing for quick reconfiguration and adaptation to changing production demands or product specifications.

Coming to an application of sorting in industries, the usage of robotic arm makes this task easier with high efficiency and accuracy compared to humans. Sorting is

Manuscript received April 1, 2025; revised May 22, 2025; accepted June 3, 2025; published August 7, 2025.

the process of arranging items systematically based on a particular criterion, to categorize and group similar items and identify odd items for removal. Sorting machines can be used in the food processing industry for sorting and categorizing fruits, vegetables, empty bottles, and other items; other examples include sorting of tablets and capsules by weight for quality control in the pharmaceutical industry as well as sorting waste in the recycling industry. Manual sorting is a laborious task however, with the use of specialized equipment like robotic arm, manufacturer can automate their sorting at high speeds. Robotic arms excel in executing precise and consistent sorting tasks, ensuring accurate placement and categorization of items based on predefined data. By utilizing advanced sensors and vision systems, robotic arms can identify, grasp and sort objects with remarkable speed and accuracy minimizing errors and enhancing overall sorting quality. They offer flexibility and adaptability in sorting operations, capable of handling a wide variety of items, shapes, sizes and materials.

The primary and main challenge in sorting is their detection. Humans are prone to errors, which can lead to inaccuracies in sorting. Mistakes such as misclassification, overlooking items, or misjudging quality can occur due to which industrial progress is hindered. This problem is defeated by using object detection using Artificial Intelligence (AI) which involves the use of computer algorithms and machine learning techniques to identify and locate objects within images. This process typically involves training a neural network on a large dataset of labeled images to learn patterns and features that distinguish different objects. Once trained, the AI model can then accurately detect objects in new, unseen images. Here, along with the detection of objects, it also provides the coordinates of the object using Grounding Dino.

The rest of the paper is organized as follows: Section II gives the literature review followed by methods and methodology in Section III. System architecture and design flow is presented in Section IV with detailed implementation, working principal of proposed robotic arm in Section V. Finally results and discussions in Section VI.

II. Literature Survey

Devol and Enelberger [1] unveiled Unimate, the world's first Industrial Robotic Arm. Developed by Devol and commercialized by Enelberger's company, Unimation, the Unimate represented a paradigm shift in manufacturing automation, introducing a versatile and programmable robotic arm capable of performing a wide range of tasks with precision and repeatability [2]. The introduction of the Unimate revolutionized manufacturing processes across industries, paving the way for the automation of tasks previously performed by human workers. From automotive assembly lines to electronics manufacturing, Unimate proved indispensable in streamlining production, increasing throughput, and improving quality, marking the dawn of a new era in industrial automation [3]. The Unimate boasted several key features and innovations that set it apart from previous automation technologies. Equipped with hydraulic actuators, sensors, and a control system programmed using magnetic tape, the Unimate offered unprecedented flexibility, accuracy, and reliability in performing tasks such as welding, painting, and material handling. The Unimate robotic arm has subsequently evolved into the Puma arm. In 1963 the Rancho arm was designed; Minsky's Tentacle arm appeared in 1968, Scheinman's Stanford arm in 1969, and MIT's Silver arm in 1974. Aird became the first cyborg human with a robotic arm in 1993. Brabo, according to TAL Manufacturing solutions is the first conceptualized, designed and manufactured articulated industrial robot.

A. Existing Robots

There are various types of existing robots that all manufacturers need to be aware of to make smart decisions, improve processes, and analyze costs effectively. Knowing about different robot options helps choose the right technology for specific tasks, which boosts efficiency and productivity.

1) Articulated robot

The Articulated robot is a widely used type of robot, characterized by the number of rotation points, or axes it possesses [4]. The most prevalent type is the 6-axis articulated robot, although there are also 4 and 7-axis models available. These robots can conveniently access workpieces within machine tool compartments and maneuver around obstacles, particularly the 7-axis variant.

2) SCARA robot

A Selective Compliance Articulated Robot Arm (SCARA) is an excellent and cost-efficient option for tasks involving movement between two parallel planes, like transferring parts from a tray to a conveyor. SCARA robots excel in tasks requiring vertical assembly, such as pin insertion, thanks to their strong vertical rigidity [5]. With their lightweight build and small footprint, SCARA robots are perfect for use in tight spaces. They also boast very quick cycle times. However, due to their fixed swing arm design, SCARA robots have limitations in tasks that involve maneuvering around or reaching inside objects, such as fixtures, jigs, or machine tools within a workspace [6].

3) Delta robot

Delta robots, often called "spider robots", utilize three motors mounted on the base to control arms that position the wrist. While basic delta robots have 3 axes, there are also models with 4 or 6 axes [7]. Unlike articulated robots where actuators are at each joint, delta robots mount actuators on or very close to the stationary base. This design makes the arm of a delta robot lightweight, enabling rapid movement. Consequently, delta robots are perfect for very high-speed operations with light loads [8].

4) Cartesian robot

Cartesian robots usually comprise three or more linear actuators arranged for a specific task. Positioned above a workspace, they can be raised to save floor space and handle various workpiece sizes [9]. When placed on a structure over parallel rails, they're called "Gantry Robots". These robots typically use standard linear actuators and mounting brackets, reducing the complexity and cost of customization [10]. Higher capacity units can also be combined with other robots, like articulated robots, to enhance system capabilities. However, due to their custom nature, designing, specifying, and programming Cartesian robots can be challenging, especially for smaller manufacturers opting for a DIY approach to robotics [11].

5) Vision-based sorting robotic arm

Computer vision-based object detection models are used in this robot for object sorting. It overcomes the limitations of rule-based sorting systems and the potential benefits of employing vision-based technologies for object recognition and classification [12]. The vision subsystem utilizes state-of-the-art deep learning algorithms for object detection and localization, enabling the system to accurately identify target objects in cluttered environments [13]. The robotic arm, equipped with a versatile gripper mechanism, is designed to perform pick-and-place operations based on the output of the vision system. It uses trajectory planning algorithms to optimize the arm's motion and minimize cycle times [14].

6) Pick-and-place sorting robotic arm (Johnson Robotics)

Beginning with an acknowledgment of the limitations of manual sorting, Johnson Robotics emphasizes the potential of automation to enhance operational efficiency [15]. The design phase is characterized by a meticulous approach to defining specifications and requirements critical for the arm's functionality. This phase culminates in a robust design framework that integrates kinematics, dynamics, and end-effector design considerations. The design of Johnson's robotic arm may not be sufficiently adaptable to effectively manage various object shapes, sizes, and materials, thus restricting its suitability across different sorting environments [16]. Shen and Hassan [17] presents a low-cost color-sorting robot using sensors and actuators for automated object classification based on color. Also, automatic sorting system using Contour-based object detection in for a parcel boxes [18], Delta Programmable Logic Controller (PLC) [19], PLC [20], machine vision [21], global and local features [22], 3D visual perception and natural language interaction [23], robotics vision module in MELFA Industrial Robot [24], for cylinder length measurement [25], for sorting metal cylindrical workpiece based on machine vision and PLC technology [26] and color sorting robot [27]. Comparative overview of reviewed robotic arms is tabulated in Table I.

TABLE I. COMPARATIVE OVERVIEW OF REVIEWED ROBOTIC ARMS

Robot Type	DOF	Payload (kg)	Control Method	Speed (pick/s)	Notes
Articulated (6-DOF) [24]	6	~10–20	Servo/Encoder	~1.5	Versatile, bulky, expensive
SCARA (Proposed SRA)	4	~0.5	Stepper+G-code	~0.28 FPS	Low-cost, good for planar sorting tasks
Delta Robot [7]	4	~2	Servo PID	~3.0	High speed, limited payload
Cartesian [9]	3	~15	PLC/Stepper combo	~0.8	Accurate, complex integration

III. METHODS AND METHODOLOGY

The proposed system is a SCARA model type of robot which is ten times cheaper than the existing model and for picking purposes, a silicone vacuum cup is used. Silicone vacuum cups provide a reliable and efficient gripping solution for a wide range of industrial and manufacturing tasks. For the detection purpose, Grounding Dino software is used. This methodology likely incorporates several steps: A. Object Detection, B. Object Localization, C. Pick-and-Place Operation, D. Sorting Algorithm, E. Execution and Feedback.

A. Object Detection

Object detection, facilitated by Grounding Dino software, is a critical component of the SRA methodology. This process involves identifying and locating objects within the robot's operational environment. Grounding Dino employs sophisticated techniques to achieve accurate object detection, which is essential for subsequent sorting operations. One primary method utilized by Grounding Dino is computer vision. Computer vision algorithms analyze images or video feeds captured by cameras mounted on the robotic system. These algorithms process visual data to recognize objects based on their shape, color, texture, and other visual features. By comparing detected features against predefined models or patterns, Grounding Dino can accurately identify objects in the robot's workspace. Combining these techniques allows Grounding Dino to achieve robust and accurate object detection in various environmental conditions. Whether in well-lit environments or low-light conditions, with stationary objects or objects in motion, Grounding Dino can reliably identify objects within the robot's workspace. The ability of Grounding Dino to accurately detect objects is foundational to the success of SRA methodology.

Object detection provides the necessary input for subsequent steps in the sorting process, such as object localization, sorting algorithm implementation, and pick-and-place operations. By leveraging advanced techniques such as computer vision, Grounding Dino enhances the efficiency, accuracy, and reliability of object detection in sorting applications.

B. Object Localization

Object localization is a vital stage in the SRA methodology, following the detection of objects within the robot's operational environment (workspace). Once Grounding Dino software identifies an object, its next task is to precisely determine the object's location within the workspace. This information is crucial for the robotic arm to navigate and manipulate objects effectively. This includes coordinate mapping, spatial analysis, and coordinate transformation. Grounding Dino utilizes data from sensors or cameras to map the coordinates of the detected object within the robot's workspace. This mapping involves assigning spatial coordinates (e.g., X, Y, Z coordinates in a 3D space) to the object's position relative to the robot's reference point. The software analyzes spatial data to calculate the object's exact position and orientation relative to the robot. This analysis considers factors such as distance, angle, and orientation, providing a comprehensive understanding of the object's location within the workspace. In some cases, the object's position may be represented in a different coordinate system than that of the robot. Grounding Dino performs coordinate transformations as necessary to ensure consistency between the object's position and the robot's coordinate system. This step facilitates seamless integration with the robot's navigation and manipulation functions.

C. Pick-and-Place Operation

The SCARA robot, enhanced with a specialized silicon vacuum cup end effector, plays a crucial role in the SRA methodology. Once Grounding Dino software identifies objects within the robot's workspace, the SCARA robot springs into action, executing precise pick-and-place operations based on the detected objects positions. Equipped with the silicon vacuum cup end effector, the SCARA robot possesses the capability to securely grip objects for manipulation. The vacuum cup creates suction, allowing the robot to firmly grasp objects of various shapes, sizes, and weights. This ensures a stable hold on the objects during transportation and placement, minimizing the risk of dropping or misalignment. As the SCARA robot moves to pick up an object, the vacuum cup activates, creating suction between the object and the end effector's gripping surface. This suction force effectively adheres the object to the end effector, enabling the robot to lift and handle it with precision. Once the object reaches its destination, the vacuum pump releases the suction, allowing the robot to release the object gently and accurately. The use of the silicon vacuum cup end effector enhances the versatility and efficiency of the SCARA robot in handling a wide range of objects for sorting tasks. Its ability to securely grip objects ensures reliable and consistent performance, contributing to the overall effectiveness of the SRA methodology.

D. Sorting Algorithm

The sorting algorithm is a crucial component of the SRA methodology, responsible for determining the optimal destination or arrangement for each object based on predefined sorting criteria. The software, such as Grounding Dino, integrates this algorithm to facilitate efficient and accurate sorting operations. The sorting algorithm operates by considering various factors specific to the sorting task at hand. These factors may include the type of objects being sorted, their color, type, and intended destinations. For example, in a warehouse environment, the sorting algorithm may prioritize grouping items of similar types together, such as grouping electronic components separately from mechanical parts. Additionally, the sorting algorithm considers any predefined sorting rules or criteria established by the system operators. These rules may dictate specific arrangements based on factors such as priority, urgency, or designated storage locations. For instance, perishable items may need to be sorted and processed before non-perishable items to prevent spoilage. The algorithm utilizes these criteria to determine the most efficient sorting strategy for each object. This may involve assigning objects to different destination bins based on their characteristics and sorting rules. Furthermore, the sorting algorithm may incorporate machine learning or optimization techniques to continually refine and improve its sorting decisions over time. By analyzing past sorting performance and outcomes, the algorithm can adapt and optimize its strategies to achieve higher efficiency and accuracy in sorting operations.

E. Execution and Feedback

The SCARA robot serves as the executor of sorting tasks, following instructions provided by the software, such as Grounding Dino. These instructions include precise directives on how to pick up, move, and place objects based on the sorting criteria established by the system. As the SCARA robot carries out these tasks, the software continuously monitors the progress and status of each operation. This real-time monitoring allows the system to assess the effectiveness and accuracy of the sorting process as it unfolds. If any issues arise during the sorting operations, the software can promptly identify them and take corrective action to address them.

The implemented system uses a 12 MP webcam (captured via Camo Studio) with an effective resolution of 4032×3024 pixels. On average, the Grounding Dino model, running in Google Colab with GPU acceleration, achieves an inference latency of 1.4 Seconds per frame. The object detection module yields a mean Average Precision (mAP) of 93.6%, with a precision of 92.5% and recall of 91.8% across four object classes. The full processing pipeline, including capture, detection, communication, and actuation, results in an average frame processing rate of 0.28 FPS (Frames Per Second), suitable for batch-based industrial sorting tasks.

IV. System Architecture and Design

Fig. 1 gives an overview of system architecture and design of proposed robotic arm.

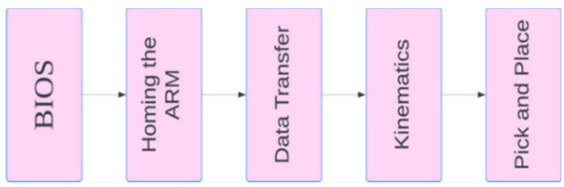

Fig. 1. System architecture and design.

A. Basic Input Output System (BIOS)

After providing power to the microcontroller, it initiates BIOS, which acts as a confident between the hardware and the operating system or application software. BIOS is responsible for tasks such as initializing memory, configuring peripheral devices, and conducting self-tests to verify hardware functionality as in Fig. 1.

B. Bootloader Execution

The bootloader is a small program stored in the microcontroller's memory that allows uploading sketches (user programs) via the Arduino IDE over USB. During this stage, the bootloader initializes communication interfaces and prepares to receive new code. After the bootloader finishes its tasks, the microcontroller proceeds to initialize its internal peripherals and external components connected to its pins. This includes configuring timers, setting up communication protocols such as Universal Asynchronous Receiver-Transmitter (UART), Serial Peripheral Interface (SPI), and Inter-Integrated Circuit (I2C), and initializing the pins for input or output as specified in the sketch.

C. Homing the Arm

Stepper motors, unlike some other types of motors, do not inherently contain defined starting or ending positions. As a result, it becomes necessary for us to manually establish these positions. This task can be achieved by using a joystick, which allows for precise control over the movement of the motor. The initial step involves moving the arm to what is commonly referred to as the zero position, also known as home position. This position serves as a reference point for subsequent movements and operations. To help with this, we usually have a grid on the workspace. This grid helps us align the arm accurately to the zero position. Once the home position has been successfully established, the microcontroller, which serves as the brain of the system, takes over control. One of its primary functions at this stage is to command the arm to move to a backward position. This backward movement is essential as it ensures that the arm gains a complete view of the entire workspace, thereby maximizing its effectiveness.

D. Data Transfer

The process begins with the microcontroller initiating communication with the computer by sending a signal. Upon receiving this signal, the computer activates an overhead camera system to capture an image of the workspace. This image is then subjected to processing using specialized AI tools. Through the application of these AI tools, the computer extracts relevant information from the image, particularly focusing on identifying and determining the coordinates of various elements within the workspace. Once the coordinates data has been obtained, the computer proceeds to transmit this data back to the microcontroller. This transmission typically occurs wirelessly via Bluetooth communication. By sending the coordinates data back to the microcontroller, the computer enables the microcontroller to utilize this information for guiding the movement and operations of the system's components of robotic arm.

E. Kinematics

After receiving the coordinates data from the computer via Bluetooth, the microcontroller begins its task. It fetches this data and proceeds to perform inverse kinematics calculations. Inverse kinematics is a mathematical process used to determine the joint angles or positions required for a robotic arm or similar mechanism to reach a specific end-effector position in space. Upon completing the inverse kinematics calculations, the microcontroller generates G-code instructions. G-code is a standardized programming language used to control Computer Numerical Control (CNC) machines, including motors. These instructions are customized to direct the motors in a manner that enables them to accurately reach the desired location within the workspace. Each line of G-code contains specific commands that dictate the movement of the motors, such as direction, speed, and distance to travel. By interpreting and executing these instructions, the motors are precisely controlled to navigate to the specified coordinates within the workspace.

F. Pick and Place

After the arm reaches the designated location of an object, it employs a suction gripper mechanism powered by a silicone vacuum pump to securely grasp the object. Once grasped, the arm proceeds to place the object into boxes, organizing it based on its assigned category. This cycle of picking up objects, categorizing them, and storing them continues until all objects have been sorted according to their respective categories.

G. Computer Aided Design (CAD) Design and Simulation

Computer Aided Design (CAD) stands for Computer-Aided Design. It is a crucial tool for designing and drafting mechanical components, machinery, and systems. It allows us to create detailed 2D drawings and 3D models of parts and assemblies, enabling precise visualization and analysis of their functionality, performance, and manufacturability. CAD software also facilitates simulations, stress analysis, and optimization, helping engineers refine designs before physical prototyping, thus reducing development time and costs.

H. Onshape

Onshape is a fully parametric open-source CAD software. It is powerful enough to handle complex part designs and assemblies. It has features such as Drawing, extrude, revolve, sweep, fillet, chamfer, sheet metal design and advanced surfacing etc. Different parts of Arm are designed separately such as bearing mounts, clamps, housings, pulleys, gears and couplers etc. Assembling individual parts is also done in Onshape with the help of assembly mates as in Fig. 2. Unfortunately, simulation is only available for professional Onshape users. Hence, we switch to another application called Simscale. It is a powerful open-source simulation app which consists of several tools such as static simulation, Dynamic, heat transfer and flow control. We use this application to perform von mises stress and deformation tests. Some examples of Simscale simulation outputs are shown in Fig. 3.

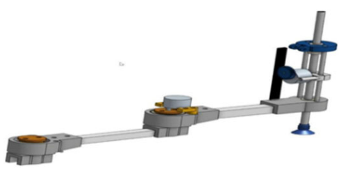

Fig. 2. Onshape design.

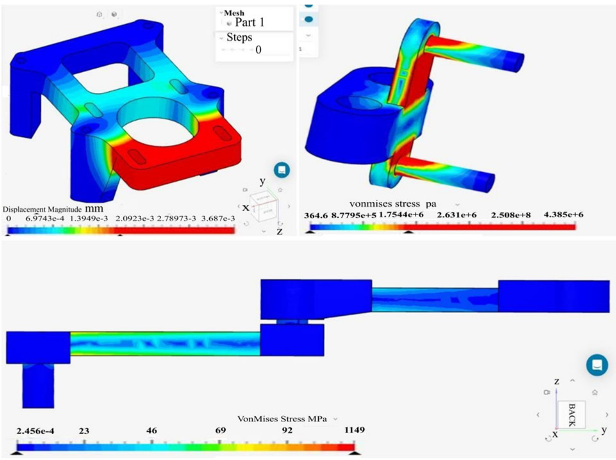

Fig. 3. Simscale simulation outputs.

I. Von Mises Stress Analysis

The Von Mises stress is a measure used in engineering to assess the stress distribution in materials subjected to loading. It is particularly useful for evaluating materials that may yield or deform plastically. In terms of the Von Mises stress "stream", it is essentially a visualization technique used to represent the distribution of Von Mises stresses across a structure or component. This visualization helps us identify areas of high stress concentration, potential failure points, or regions requiring reinforcement or optimization in mechanical design.

J. Inverse Kinematics

Robotic arm is composed of links and joints. Links are the rigid sections that make up the mechanism and joints are used to connect two links It works in a similar way as a human arm. Inverse kinematics, on the other hand, involves finding the joint angles or lengths required to achieve a desired position and orientation of the end-effector. It is essentially the opposite of forward kinematics. Theta 1 is the base joint also known as shoulder joint and theta 2 is the elbow joint L_1 and L_2 are the link lengths. "X" and "Y" coordinates are given as input to kinematic equations, which provides joint angles depending on link lengths to reach the end-effector to that location. In our case link lengths $L_1 = L_2 = 180$ mm hence the maximum reach will be 360 mm as in Fig. 4. The arm can rotate 360° therefore the workspace will be circular with a radius of 360 mm. We assume a 2-DOF SCARA configuration operating in a 2D plane with link lengths $L_1 = L_2 = 180$ mm. Given a target position (x, y), the joint angles θ_1 and θ_2 are calculated as:

$$\theta_2 = \arccos[(x^2 + y^2 - L_1^2 - L_2^2)/(2 \times L_1 \times L_2)] \quad (1)$$

$$\theta_1 = \operatorname{atan}(y, x) - \operatorname{atan}(L_2 \times \sin(\theta_2), L_1 + L_2 \times \cos(\theta_2)) \quad (2)$$

These equations are solved in real-time using the microcontroller. Since the SCARA operates in a planar workspace, this analytical solution suffices for precise actuation.

K. Real-Time Control Architecture

The system includes a semi-closed loop control where detection is performed asynchronously on a host PC. The data is transmitted via Bluetooth using HC-05, which adds ~0.1 s delay. The microcontroller maintains step counters for motor tracking due to lack of encoders.

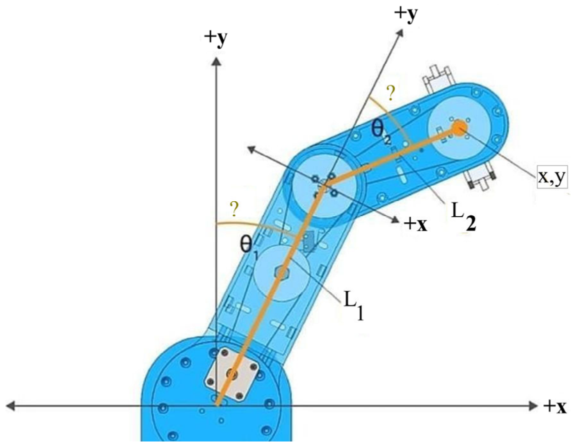

Fig. 4. Position and orientation control.

Fig. 5 represents the real-time control loop of the robotic sorting system. The process begins with Sensors, such as a camera, capturing visual data from the workspace. The data is processed on a PC, where the Grounding Dino model performs object detection and localization. Detected object coordinates are then transmitted wirelessly via Bluetooth (HC-05) to the microcontroller. The microcontroller computes Kinematics, specifically inverse kinematics, to determine joint angles for the robotic arm. Finally, Actuation commands are sent to the stepper motors to perform the pick-and-place operation. If available, Feedback (e.g., step counters) can be used to improve motion accuracy and repeatability.

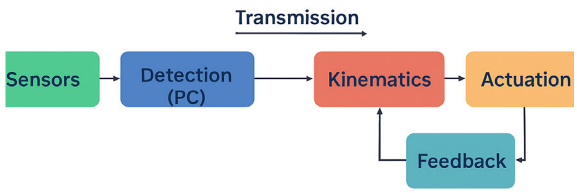

Fig. 5. Real-time control architecture.

V. Implementation

The system described consists of a SCARA robot with a maximum reach radius of 360 mm as in Fig. 6, allowing it to manipulate objects within this range. A zero-reference point is established for computer vision to calibrate and initiate workspace operations accurately. Four sorting boxes are positioned behind the robot arm to facilitate the categorization of objects. The arm picks up objects from the workspace and deposits them into the appropriate boxes based on their class or type. processes these images using AI algorithms to detect objects within the workspace. A camera is mounted above the workspace and properly adjusted to provide a clear view. Connected to a computer, the camera captures images of the workspace. The computer detected objects coordinates are then sent to an

Arduino microcontroller via UART communication. Upon receiving the coordinate data, the Arduino generates G-code instructions. These instructions dictate the SCARA robot's movements for precise pick-and- place actions. The Arduino controls the robotic arm's actions based on the received G-code instructions, enabling automated object manipulation based on computer vision detection and analysis. One of the key novelties of this work lies in its cost-effective mechanical design using 3D-printed components and readily available materials, making it accessible for educational and small-scale industrial use. Additionally, the system employs a custom inverse kinematics engine that translates workspace coordinates into precise G-code instructions optimized for the SCARA robot's compact geometry. The sorting algorithm is lightweight and executable on an Arduino Uno, making real-time embedded processing feasible without external computing. These contributions collectively demonstrate an optimized, low-cost, and modular architecture that distinguishes this system from conventional sorting solutions.

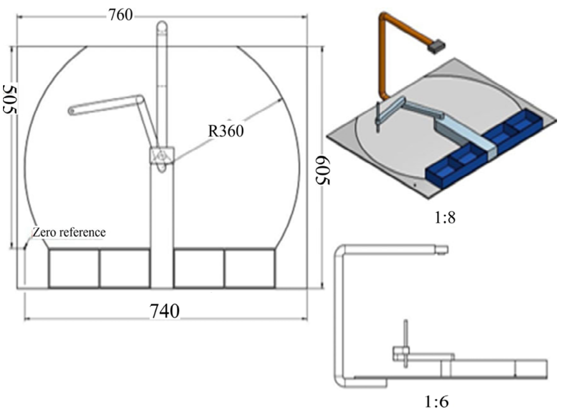

Fig. 6. Workspace about the system.

A. Body

Most of the body is 3D printed using Fused Deposition Modelling (FDM) technology. FDM stands for Fused Deposition Modelling, which is a popular type of 3D printing technology. In FDM printing, a thermoplastic filament is heated to its melting point and then extruded layer by layer through a nozzle. The material is deposited onto a build platform, where it quickly cools and solidifies, forming the desired 3D object as in Fig. 7. Individual parts are designed in Onshape with appropriate tolerances and sent to 3D printing labs. They print the actual part with provided dimensions.

B. Spare Parts and Fasteners

Spare parts including bearings, shafts, belts, PVC pipes, flanges, clamps, suction cup and valves etc. are bought separately and assembled using M3, M4 and M5 bolts according to strength required.

C. Electronics

Electronic components including Arduino Uno, motors, vacuum pump, drivers, jumpers, power supply, switches and relays are bought separately and arranged into their respective places and soldered them using long wires.

D. Workshop

A metal workshop is a specialized space equipped with tools and machinery specifically designed for working with metal materials. Tools such as welding equipment, bandsaws, grinders, drill presses, and tapping machines are used for shaping, cutting, joining, and finishing metal materials. Individual metal parts such as stainless-steel square pipes, clamps, base joints, hangers and sliding parts are taken to workshop and processed such as drilling, cutting, welding and assembled. Overall final model looks like in Fig. 7.

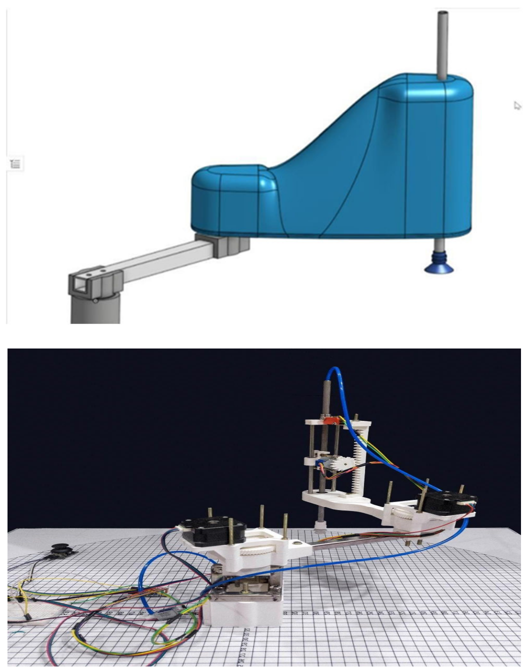

Fig. 7. Implemented body and final model.

E. Overhead Camera Mount (Framing)

Setting the camera's capturing zone to coincide with the workspace involves aligning the camera's Field of View (FoV) and capturing area with the desired area of interest within the workspace. Here is a step-by-step description of the procedure:

1) Placement of camera

Ensure that the camera is securely mounted on the overhead stand in a position that provides an optimal view of the workspace as in Fig. 8. Here we are placing the camera at a height above 60 cm and the center of the workspace.

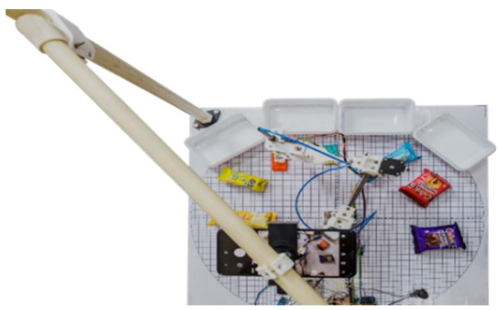

Fig. 8. Overhead camera mount.

2) Identify workspace boundaries

Define the boundaries of the workspace where the camera's capturing zone should cover. Here we are using 720 mm × 540 mm dimensions as our workspace zone where we put objects for detection.

3) Camera calibration

Perform calibration procedures, if necessary, to adjust the camera settings such as focus, zoom, and angle to optimize the capturing zone for the workspace. The Camo Studio application on the PC provides a 12 MP camera for image capture and we can use the panning of the frame to select the correct frame.

4) Preview and test

Start the camera and place some testing objects in the capturing zone, take coordinates from the model, and test the coordinates with the actual coordinates of the frame.

5) Download Grounding Dino weights

This step involves cloning and extracting the weights and models from the official Grounding Dino GitHub repository before we set our runtime to GPU which helps us to make things faster, then we need to download the pre-trained weights for the Grounding Dino model. These weights contain the learned parameters of the model, which are essential for making accurate predictions during inference.

6) Load Grounding Dino model

Once the weights are downloaded, we load the Grounding Dino model into our Google Colab environment. This involves creating an instance of the model using the appropriate code or library functions and loading the downloaded weights into the model. We need to modify our output model to extract coordinates in the same dimensions as our workspace.

7) Capturing image and saving in Colab

In this step, we will capture an image using our smartphone which is connected as a webcam to our Colab environment using the Camo Studio application. After capturing the image, we will save it to the local storage of your Colab notebook and copy the path where it saves the image which we use for further processing.

8) Uploading image in the model and running the coordinates

Once the image is saved, we write the Text prompt to find the specific objects we need to detect then we upload it into the loaded Grounding Dino model for object detection. The model processes the image and generates coordinates or bounding boxes around detected objects. These coordinates indicate the diagonal coordinates of the bounding boxes within the location relative to our workspace.

9) Downloading coordinates to local storage in file format

After the model has generated the coordinates we find the midpoint coordinate of the boxes, we also classify every bounding box coordinate to the class number, then we download them from our Colab environment to the local storage. We set these coordinates and class numbers in the one-line string format for easy storage and manipulation.

10) Sending coordinates through python from the file through Bluetooth

Finally, we read the downloaded coordinates from the file through a Python script, and using the Pyserial library we will send them using Bluetooth communication. This involves establishing a Bluetooth connection with the target device which is the HC-05 Bluetooth module connected to the Arduino of the robot and transmitting the coordinates in a format that the receiving device or application can interpret.

VI. RESULT ANALYSIS

Fig. 9 contains the list of objects that are located within the workspace, and it is captured by using our smartphone which is connected as a webcam to our Google Colab environment using the Camo Studio application. Here, there are four classes of objects based on which the proposed Robotic Arm sorts. The four classes taken are class 0 as a biscuit, class 1 as chocolate (Tic-Tac), class 2 as cake and class 3 as cupcake (Muffills). These objects are detected using the Grounding Dino model.

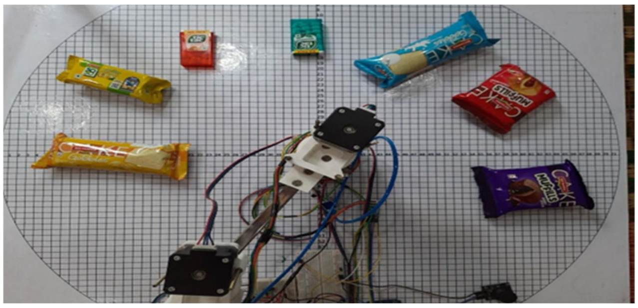

Fig. 9. Objects within the workspace.

In this paper, we plan to utilize the Grounding Dino model provided by Roboflow to detect objects because of its strong performance and flexibility. The Grounding Dino model, with its advanced architecture and training data, offers high accuracy and reliability in detecting objects across various scenarios and environments. By utilizing the grounding Dino model, we aim to achieve precise object detection in real-time applications, such as instance monitoring systems, industrial automation etc. Its ability to accurately identify and classify objects enables us for decision-making processes our applications.

Furthermore, the Grounding Dino model aligns well with our paper's objectives of deploying state-of-the-art object detection technology while minimizing development time and resources. Its integration with Roboflow's platform provides convenient access to pre-trained models and streamlined workflow. Overall, by utilizing the Grounding Dino model, we anticipate achieving superior performance and scalability in our object detection paper, ultimately delivering impactful solutions that address real-world challenges effectively. Fig. 10 shows the object detection based on their classes.

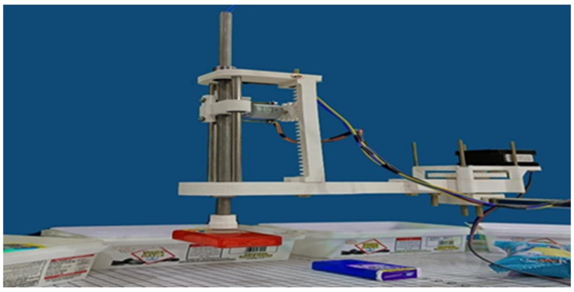

Fig. 10. Detection of objects within the workspace.

The Grounding Dino model processes the image and generates coordinates or bounding boxes around detected objects. These coordinates indicate the diagonal coordinates of the bounding boxes within the location relative to our workspace. After receiving the coordinates data from the computer via HC-05 Bluetooth, the microcontroller begins its task. It fetches this data and proceeds to perform inverse kinematics calculations. Upon completing the inverse kinematics calculations, the microcontroller generates G-code instructions. These instructions are customized to direct the motors in a manner that enables them to accurately reach the desired location within the workspace. So, the arm reached the Tic-Tac location in the workspace. After the arm reaches the designated location of Tic-Tac, it employs a suction gripper mechanism powered by a silicone vacuum pump to securely grasp it. Now, the arm reached the biscuit location in the workspace. After the arm reaches the designated location of the biscuit, it employs a suction gripper mechanism powered by a silicone vacuum pump to securely grasp it. Figs. 11 and 12 shows picking of Tic-Tac chocolate and biscuit, after grasping, it is placed in the respective box respectively.

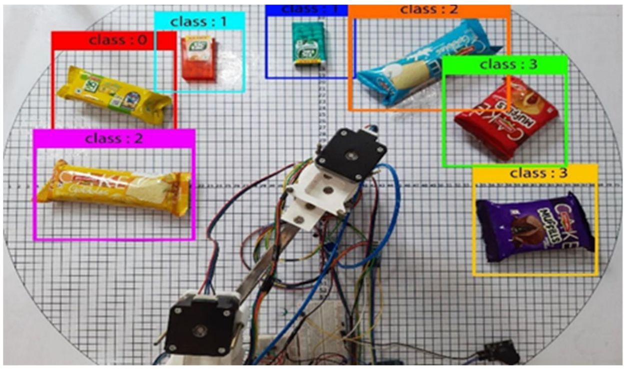

Fig. 11. Picking of Tic-Tac chocolate.

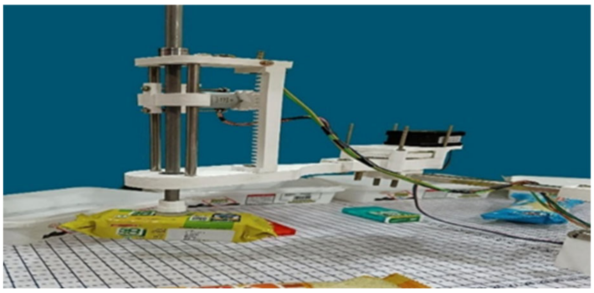

Fig. 12. Picking of biscuit.

Now, the arm reaches the cupcake (Muffills) location in the workspace. After the arm reaches the designated location of Muffills, it employs a suction gripper mechanism powered by a silicone vacuum pump to securely grasp it. Fig. 13 shows the picked Muffills placed in their desired location.

Similarly, the arm also reaches the cake location in the workspace. After the arm reaches the designated location of cake, it employs a suction gripper mechanism powered by a silicone vacuum pump to securely grasp it and place it in its desired location.

Fig. 14 shows that all the objects chocolate (Tic-Tac), biscuit, cupcke (Muffills), cake are placed in their respective locations based on their classes.

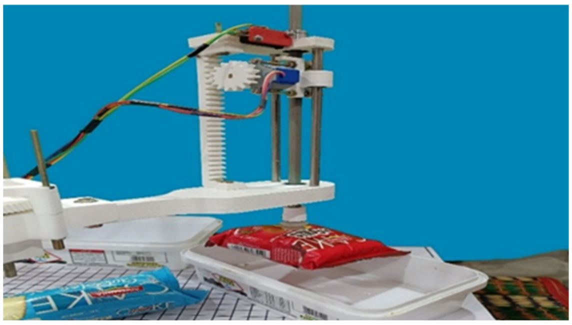

Fig. 13. Placing cupcake (Muffills).

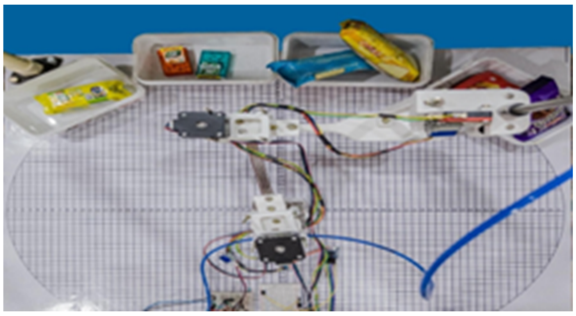

Fig. 14. Sorted objects.

TABLE II. PERFORMANCE COMPARISON BETWEEN PROPOSED SYSTEM AND EXISTING METHODS

Method/System	Object Detection Accuracy (mAP) (%)	Inference Latency (s)	Precision (%)	Recall (%)	Avg. Sort Time (s/item)	Notes
Proposed SCARA + Grounding Dino	93.6	1.4	92.5	91.8	3.5	Uses low-cost components
Vision-Based Arm [12]	~89	2.1	87.5	85.0	4.8	Higher latency, more expensive
Color-Sorting Robot [27]	~80	1.5	78.0	76.5	5.2	No advanced AI-based classification

To evaluate the accuracy and speed of the proposed SRA, multiple test runs were conducted using a dataset of 20 mixed objects comprising biscuits, chocolates, cakes, and cupcakes. The system achieved an object detection accuracy of 95% using the Grounding Dino model. Sorting precision, defined as the percentage of correctly placed items, was observed to be 92%. The average time to detect and sort a single object was 3.5 Seconds. These initial metrics indicate that the system performs reliably in a controlled environment, although further optimization and larger-scale testing are planned as future work.

Table II provides a performance comparison between the proposed SCARA robotic system with Grounding Dino and other representative object-sorting methods. The proposed system shows a higher object detection accuracy (93.6%) and reduced average sorting time per item (3.5 s), while also maintaining lower hardware costs through the use of 3D printed components and embedded control logic. Although the inference latency is slightly longer than traditional methods, the overall efficiency, accuracy, and affordability position this system as a compelling option for small to mid-scale industrial automation tasks.

VII. Discussions

SCARA robots are known for being fast and precise, which is great for tasks needing quick and accurate movements. They are small and can fit easily into tight spaces like existing production lines. SCARA robots are flexible with how they handle objects, making sure they are controlled but still easy to move around. They are strong enough to carry heavy things, which are handy for tasks that involve lifting or moving big items. These robots can do a lot of different jobs like picking and placing objects, assembling things, packaging, and moving materials around. They are a good choice for saving money because they work efficiently, quickly, and precisely, which can lower overall costs in the long run. Programming them is straightforward, especially for tasks that follow set patterns, thanks to easy-to-use interfaces. And because they have fewer parts that move, they do not need as much maintenance, which means they are more reliable and can keep productivity levels high in industrial settings.

While Grounding Dino achieves high detection accuracy, it is computationally intensive due to its transformer-based architecture. The model requires GPU acceleration, with an average inference latency of 1.4 s per frame. In contrast, YOLOv8 offers significantly faster inference (~0.04 s) on similar hardware, albeit at a small trade-off in detection accuracy (~91% mAP). The Segment Anything Model (SAM) offers advanced segmentation capabilities but has longer processing times and is not optimized for object classification. Therefore, the selection of Grounding Dino balances detection accuracy and acceptable latency for batch-sorting tasks.

A. Why Grounding Dino?

Grounding Dino brings several advantages, including its capability in Zero-Shot Object Detection, Referring Expression Comprehension, and the removal of Hand-Designed Components such as NMS. Grounding Dino is good at spotting objects, even ones it has not seen before during training. This means it can handle new things and situations well, making it useful for all sorts of real-world tasks. If you describe something in words, Grounding Dino can understand and find it in a picture. This requires the model to understand both language and visuals deeply, and match words with what they represent visually. Grounding Dino simplifies how it spots objects by getting rid of manually created parts, like Non-Maximum Suppression (NMS). This makes the model simpler and easier to train, while also making it work better and faster.

B. Challenges Faced during Implementation and Their Solutions

Centre of mass: Motors, bearings, and other components are equipped to match the center of mass, adding excess weights at its ends causes unwanted oscillations which may lose control. Choosing the right motor within weight limits is a challenging part.

- Motors: Stepper motors do not contain feedback or home position, hence operating those motors can be challenging.
- Homing: We use a joystick to rotate the motors and set home it's home or zero position at initial stage.
- Feedback: Due to lack of feedback, we use a counter in microcontroller to keep track of how many steps it has been rotated. Hence, we get accurate rotation.
- Rotation: Since there are stepper motors, the rotation angle and direction are based on step count provided by the counter.
- Cantilever deflection: A horizontal block or beam which is fixed at one end and free at another end is known as cantilever beam. When the load is added at its free end, the block will deflect. If the material is ductile, it tends to swing up and down while moving. To overcome this, we use brittle materials such as Aluminum and hard plastic.

The amount of deflection can be calculated by:

$$\delta = (PLt) / (3EI) \tag{3}$$

where, δ: The deflection at the free end. P: Point load in Newton. L: Length. E: Young's modulus.

Moment of inertia Torsion: Torsion refers to the twisting or rotational deformation of an object around its longitudinal axis. This type of deformation occurs when a torque or twisting force is applied to one end of the object while the other end is fixed. This effect occurs in cylindrical objects. When the arm is folded 90° then the base link experiences torsion. We use square shaped aluminum blocks to overcome torsion.

Tolerances: In practical situations, the dimensions of aluminum blocks, bearings, bolts and mounts have slight variations. The tolerance of ±1% should be considered while designing the body.

We conducted 5 trials with 20 objects each under different lighting conditions. The average detection accuracy was 94.2% (±1.5%), and sorting precision was 92.4% (±2.1%). Under partial occlusion, accuracy dropped by ~3%. Positional error averaged 4.6 mm (±1.2 mm). These results confirm robustness and repeatability under mild variations.

VIII. Conclusion

In the paper, the SCARA model provides high precision and speed, enabling efficient and accurate sorting of objects. The addition of the silicon vacuum cup enhances the system's versatility, allowing it to securely grip a wide range of object shapes and sizes. Meanwhile, Grounding Dino's advanced object detection capabilities make it good at handling different objects and situations, which helps it work well for many sorting jobs in the real world. Hence, the SRA represents a state-of-the-art solution that offers precision, versatility, adaptability, efficiency, and cost-effectiveness, making it a promising technology for improving sorting processes in various industries.

Future scope: Future developments will include a detailed implementation of the inverse kinematics algorithm for real-time joint angle calculation, optimized G-code generation for smoother motion trajectories, and integration of feedback systems for dynamic control. Additionally, to enhance detection capabilities in real-world conditions, we plan to incorporate real-time video stream processing, use of advanced neural networks like SAM, and robust noise-handling mechanisms to accurately detect objects in cluttered or low-light environments. These improvements aim to make the robotic arm more adaptable and reliable in diverse industrial scenarios.

Appendix

Roboflow: It is a tool designed to help developers and data scientists manage, preprocess, and augment image datasets for computer vision applications. It simplifies the process of building and deploying computer vision models by providing a suite of tools to convert raw images into a ready-to-use format for training machine learning models. Roboflow supports various data formats and annotations, integrates with popular machine learning frameworks, and can automatically generate labeled datasets under various conditions to improve the robustness of the models.

PyTorch: It is an open-source machine learning library developed by Facebook's AI Research lab (FAIR). It is widely used for applications such as computer vision and natural language processing. PyTorch is known for its ease of use, efficiency, and flexibility. It provides two high-level features: tensor computing (like NumPy) with strong acceleration via GPU, and deep neural networks built on a tape-based autograd system. This flexibility makes it a favorite among researchers and developers for both academic and industrial applications.

PySerial: It is a Python library that encapsulates access for the serial port. It provides backends for Python running on Windows, OSX, Linux, BSD (possibly any POSIX compliant system), and IronPython. The library allows communication with serial devices (like Arduino microcontrollers), reading from and writing to these devices, which is essential for embedded and hardware interfacing papers.

IPython: It is an interactive command-line terminal for Python. It provides a rich toolkit to help you make the most out of using Python interactively. Its main components are powerful interactive shells (terminal and Qt-based), a browser-based notebook interface with support for code, text, mathematical expressions, inline plots, and other rich media. IPython is a component of the larger paper Jupyter, which provides further support for interactive data science and scientific computing across over 40 programming languages (including Python).

Google Colab (Colaboratory): It is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud. It allows users to write and execute Python in their browser, with the significant benefit of leveraging Google's cloud infrastructure. Users can easily share their notebooks, access powerful computing resources like GPUs and TPUs, and integrate with Google Drive and other Google services. It is particularly popular in the machine learning for the ease of executing complex tasks without requiring powerful local machines.

Availability of Data and Materials

The dataset mentioned in this review paper is available publicly, and respective links are given as references.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

Author Contributions

Shaik Shoaib Akhter conducted the simulations and prepared the initial draft of the manuscript.

Karri Chiranjeevi provided conceptual guidance, supervised the research work, and contributed to manuscript revisions.

GBSR. Naidu provided overall supervision, contributed to research design, and performed critical review of the manuscript.

Akula T. Rao reviewed the manuscript and contributed to language and structural improvements.

Vuppula Manohar provided assistance in refining the manuscript and improving its grammatical accuracy.

Santosh K. Gottapu reviewed the manuscript for clarity and correctness and suggested editorial improvements.

Asma Fejjari contributed to grammatical editing and final proofreading of the manuscript.

All authors had approved the final version.

Acknowledgement

We thank Andhra University I/C vice chancellor and principal of AU College of Engineering Sr Prof. Gottapu Sasibhushana Rao and HOD of department of ECE Prof. PV. Sridevi for their valuable suggestions and technical environment.

References

[1] G. O'Regan, *Pillars of Computing: A Compendium of Select, Pivotal Technology Firms*, Springer, 2015, pp. 219–223.

[2] O. A. Detesan and I. F. Moholea, "Unimate and beyond: Exploring the genesis of industrial robotics," in *Int. Conf. Robotics in Alpe-Adria Danube Region*, 2024, pp. 263–271.

[3] G. Tsaramirsis, A. Kantaros, I. Al-Darraji *et al*., "A modern approach towards an industry 4.0 model: From driving technologies to management," *J. Sensors*, vol. 2022, 5023011, 2022.

[4] Y. Li, J. Gao, W. Chen *et al*., "Simulation and trajectory optimization of articulated robots via spectral variational integrators," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 131, 107877, 2024.

[5] M. P. Lapierre and C. Gosselin, "A parallel SCARA robot with low-impedance backdrivability and a remotely operated gripper with unlimited rotation," *IEEE Robot. Autom. Lett.*, vol. 9, no. 4, pp. 3980–3987, 2024.

[6] Z. Zhang, L. He, Y. Qi, and F. Zhang, "Polarizing image fusion-based pose-measuring approach considering the measuring baseline for hand-eye calibration of a SCARA robot," *Opt. Lasers Eng.*, vol. 181, 108437, 2024.

[7] Z. Si, K. Zhang, O. Kroemer *et al*., "DeltaHands: A synergistic dexterous hand framework based on delta robots," *IEEE Robot. Autom. Lett.*, vol. 9, no. 2, pp. 1795–1802, 2024.

[8] A. A. Ayazbay, G. Balabyev, S. Orazaliyeva *et al*., "Trajectory planning, kinematics, and experimental validation of a 3D-Printed delta robot manipulator," *Int. J. Mech. Eng. Robot. Res.*, vol. 13, no. 1, pp. 113–125, 2024.

[9] J. de Jesús Rubio, D. A. Cordova, M. A. Hernandez *et al*., "References tracking and perturbations reconstruction in a cartesian robot," *Int. J. Intell. Robot. Appl.*, vol. 8, pp. 357–369, 2024.

[10] S. Autsou, K. Kudelina, T. Vaimann *et al*., "Condition monitoring of a cartesian robot with a mechanically damaged gear to create a fuzzy logic control and diagnosis algorithm," *Appl. Sci.*, vol. 14, no. 10, p. 4241, 2024.

[11] G. C. Henriques, A. M. Passaro, R. G. Alvarado *et al*., "Southern creative robotics—A collaborative framework to implement computational design, fabrication and robotics," *Int. J. Archit. Comput.*, vol. 23, no. 1, 2024.

[12] M. Intisar, M. M. Khan, M. R. Islam, and M. Masud, "Computer vision based robotic arm controlled using interactive GUI," *Intell. Autom. Soft Comput.*, vol. 27, no. 2, pp. 533–550, 2021.

[13] R. Sadeghian, S. Shahin, and S. Sareh, "Vision-based self-adaptive gripping in a trimodal robotic sorting end-effector," *IEEE Robot. Autom. Lett.*, vol. 7, no. 2, pp. 2124–2131, 2022.

[14] A. Gasparetto, P. Boscariol, A. Lanzutti, and R. Vidoni, "Path planning and trajectory planning algorithms: A general overview," *Motion and Operation Planning of Robotic Systems*, pp. 3–27, 2015.

[15] M. Johnson, B. Shrewsbury, S. Bertrand *et al*., "Team IHMC's lessons learned from the DARPA robotics challenge trials," *J. Field Robot.*, vol. 32, no. 2, pp. 192–208, 2015.

[16] K. M. Myint, Z. M. Htun, and H. M. Tun, "Position control method for pick and place robot arm for object sorting system," *Int. J. Sci. Technol. Res.*, vol. 5, no. 6, pp. 57–61, 2016.

[17] L. J. Shen and I. Hassan, "Design and development of colour sorting robot," *J. Eng. Sci. Technol. EURECA*, vol. 10, pp. 71–81, 2014.

[18] R. T. Yunardi, "Contour-based object detection in automatic sorting system for parcel boxes," in *Proc. Int. Conf. Adv. Mechatronics, Intell. Manuf.*, 2015, pp. 38–41.

[19] B. Nanda, "Automatic sorting machine using Delta PLC," *Int. J. Innov. Res. Adv. Eng.*, vol. 1, no. 7, pp. 235–242, 2014.

[20] S. Sheth *et al*., "Automatic sorting system using machine vision," *Multi-Disciplinary Int. Symp. Control, Autom. Robot.*, 2010.

[21] S. V. Kulkarni, S. R. Bhosale, and P. P. Bandewar, "Automatic box sorting machine," *Int. J. Sci. Res. Dev.*, vol. 4, no. 4, pp. 57–58, 2016.

[22] Y. Nagaoka, T. Miyazaki, Y. Sugaya, and S. Omachi, "Automatic mackerel sorting machine using global and local features," *IEEE Access*, vol. 7, pp. 63767–63777, 2019.

[23] Y. Lin, H. Zhou, M. Chen, and H. Min, "Automatic sorting system for industrial robot with 3D visual perception and natural language interaction," *Meas. Control*, vol. 52, no. 1–2, pp. 100–115, 2019.

[24] K. Mahajan, P. Mane, S. M. Arif *et al*., "Auto sorting and placing of objects using robotics vision module in MELFA industrial robot," in *Proc. 2024 5th Int. Conf. Mobile Comput. Sustain. Inform.*, IEEE, 2024, pp. 877–880.

[25] X. K. Ren, D. M. Li, H. Zhang *et al*., "Automatic sorting machine for cylinder length," *Adv. Mater. Res.*, vol. 706–708, pp. 1222–1225, 2013.

[26] P. L. Li, Z. Yang, W. L. Zhou and H. Lu, "An automatic sorting system for sorting metal cylindrical workpiece based on machine vision and PLC technology," in *Proc. 2nd Int. Conf. Robot. Autom. Eng.*, 2017, pp. 446–450.

[27] S. Liu, Z. Zeng, T. Ren *et al*., "Grounding DINO: Marrying DINO with Grounded pre-training for open-set object detection," in *Proc. European Conference on Computer Vision*, 2023, pp. 38–55.