



ISSN: 0067-2904

Non-Calibrated Camera-Based Robot Arm for Pick and Place Tasks

Asma T. Saadoon¹, Abdulrazzaq M. Kamil², Ahmed A. Jasim¹, Basheera M. Mahmmod¹, Sadiq H. Abdulhussain^{1*}, Muntadher Alsabah³, Abir Hussain⁴

¹Department of Computer Engineering, College of Engineering, University of Baghdad, Baghdad, Iraq

²College of Engineering, Al-Naji University

³Medical Technical College, Al-Farahidi University, Baghdad, Iraq

⁴Department of Electrical Engineering, University of Sharjah, Sharjah, United Arab Emirates

Received: 31/12/2024

Accepted: 11/2/2025

Published: 30/7/2026

Abstract

The integration of robotic systems with machine learning and computer vision has become critical in automating systems. Accurate object localization and manipulation pose challenges for robotic systems relying on non-calibrated cameras due to positional inaccuracies. This paper introduces a new non-calibrated camera mapping technique to enhance precision in robotic pick-and-place operations. To this end, a robotic arm controller is implemented using machine learning and computer vision techniques. The proposed approach is used for different objects that are identified and labeled by certain classes and sorted accordingly. The location of the objects is found using a proposed non-calibrated camera approach. This approach eliminates the need for expensive calibration procedures while maintaining high accuracy. The methodology starts by processing the images to compute the object location with respect to the robot arm and sends it to the Arduino, which in turn moves the arm using inverse kinematics. The proposed method is evaluated by comparing actual object locations with those found using the camera mapping, resulting in a 2.458% error rate. In addition, a comparison with other non-calibrated camera approaches demonstrated that the proposed method achieved higher accuracy and lower errors. These results show promise for applications in industrial automation and robotics systems.

Keywords: Camera mapping, Non-calibrated camera, Robot arm, Object recognition, computer vision, pick and place tasks.

ذراع روبوتية تعتمد على كاميرا غير معايرة لمهام الالتقاط والوضع

اسماء طه سعدون¹, عبدالرزاق مصطفى كامل², احمد عبدالهادي جاسم¹, بشيرة محمدرضا محمود¹

صادق حبيب عبدالحسين¹, منتظر الصباح³, عبيد جعفر حسين⁴

¹قسم هندسة الحاسبات, جامعة بغداد, بغداد, العراق

²كلية الهندسة, جامعة الناجي

³الكلية التقنية الطبية, جامعة الفراهيدي, بغداد, العراق

⁴قسم الهندسة الكهربائية - جامعة الشارقة - الشارقة, الإمارات العربية المتحدة

*Email: sadiqhabeeb@coeng.uobaghdad.edu.iq



الخلاصة

أصبح دمج الأنظمة الروبوتية مع التعلم الآلي والرؤية الحاسوبية أمراً بالغ الأهمية في أتمتة الأنظمة. إن تحديد موقع الأشياء بشكل دقيق والتلاعب بها يشكلان تحدياً للأنظمة الروبوتية التي تعتمد على كاميرات غير معايرة بسبب عدم دقة الموضع. تقدم هذه الورقة تقنية جديدة لتحويل احداثيات الكاميرا غير المعايرة الى احداثيات الروبوت (تحويل الكاميرا) لتعزيز الدقة في عمليات الالتقاط والموضع الآلية. ولتحقيق هذه الغاية، تم تنفيذ وحدة تحكم ذراع آلية لالتقاط الأشياء ووضعها باستخدام تقنيات التعلم الآلي والرؤية الحاسوبية. يتم استخدام النهج المقترح لالتقاط ووضع اشياء مختلفة يتم تحديدها وتصنيفها بواسطة فئات معينة وفرزها وفقاً لذلك. يتم العثور على موقع الاشياء باستخدام نهج الكاميرا غير المعايرة المقترح. يؤدي هذا النهج إلى التخلص من الحاجة إلى إجراءات المعايرة المكلفة مع الحفاظ على الدقة العالية. تم تقييم الطريقة المقترحة من خلال مقارنة مواقع الأشياء الفعلية بتلك التي تم العثور عليها باستخدام طريقة تحويل احداثيات الكاميرا المقترحة، ونتج عن التقييم معدل خطأ بنسبة 2.458٪ بينهم. بالإضافة إلى ذلك، أظهرت المقارنة مع طرق الكاميرا الأخرى غير المعايرة أن الطريقة المقترحة حققت دقة أعلى وأخطاء أقل. وتظهر هذه النتائج إمكانات واعدة للتطبيقات في أنظمة الأتمتة الصناعية والروبوتات.

1. Introduction

In recent years, the world's focus has shifted to simplifying and improving automation technologies that can perform tasks automatically instead of human workers. Automation can be defined as a process that includes software, machines, and other technologies. This will help to improve the efficiency and effectiveness of work in a wide range of applications without human intervention. In this field, various automated systems are considered, from simple systems to complex industrial systems. Most of these systems are powered by artificial intelligence (AI) and machine learning (ML). Robots are a part of automation systems, which are defined as programmable machines that are capable of performing various tasks autonomously or semi-autonomously. Engineering and computer science cooperate together in the field of robotics, which is often used to assist people in performing their tasks easily [1]. In particular, robots can perform various functions and tasks in human life in a simpler and more productive way, such as controlling position, trajectory, or performing tasks of picking and dropping objects. Typically, robots can be divided into three broad categories. The first category relies entirely on human input for operation. While the second type is known as semi-autonomous robots that can do other tasks based on their own, but require some human intervention and monitoring. The last category, known as autonomous robots, can perform tasks completely on their own and respond to real-world environments with minimal human intervention. Manipulative robots are classified based on various criteria to determine which robot is suitable for a particular task. The most general classification is based on the robot's power source, geometry (or kinematic structure), desired application, or control method [2].

To perform tasks, robots interact with the physical world via sensory inputs. Besides, robots are reprogrammable; therefore, they offer high flexibility over single-function machines [3]. Different robot types are used in various ways to assist people in their daily work. Examples of robots are agricultural robots, welding robots, and drones. In general, robotic manipulations can also be categorized into two categories: prehensile and non-prehensile manipulation. Prehensile manipulation includes grasping or capturing, while non-prehensile involves propelling or pushing [4].

A robot arm is considered a significant type of robot that performs tasks like a human arm. Robot arms can be either autonomous or controlled manually to perform a variety of tasks with high accuracy. Besides, it is similar to a human hand and designed for industrial

applications or home applications, and can be fixed or mobile (i.e., wheeled) [5]. A robot arm can perform repetitive tasks at high speeds and with greater accuracy than human operators. A robotic arm is typically a programmable control device that has a function like human arms that can seamlessly integrate into automated workflows. The joints of this arm are connected to allow either translational displacement, i.e., linear displacement, or rotational motion (like a jointed robot) [5]. Robotic arms are flexible and suitable for hazardous places or unsafe environments such as aerospace, automotive, bomb disposal, and fireworks manufacturing industries [6]. Moreover, the applications of robot arms have expanded to living spaces and outer space.

The field of robotics research has recently become very rich due to its control, kinematics, and dynamics. In particular, kinematics plays an important role in this field, especially in relation to the behavior of industrial manipulators. Therefore, the basic steps in a robotics system are to analyze and model the kinematics of the manipulator [7]. The links of the manipulator are represented as a kinematic form chain. The business end of the kinematic chain is called the end effectors, which are designed based on the application type to perform any specific task, such as spinning, welding, and gripping.

However, controlling the robot arm with higher degrees of freedom (DOF), which is defined as the number of independent motions an object can make, is a challenging issue. Therefore, trajectory planning and position analysis are considered as decisive steps in the design and control of the robot arm [8]. It is worth noting that in space, a free rigid body has six degrees of freedom - three for directions and three for positions [9]. Currently, most industrial manipulators have a maximum of six degrees of freedom. They are usually kinematically classified based on the first three joints of the arm, with the wrist described separately. The majority of these manipulators fall into one of the five geometric types as given in Figure (1) [2] below, where the letter P refers to Prismatic and the letter R refers to Revolute.

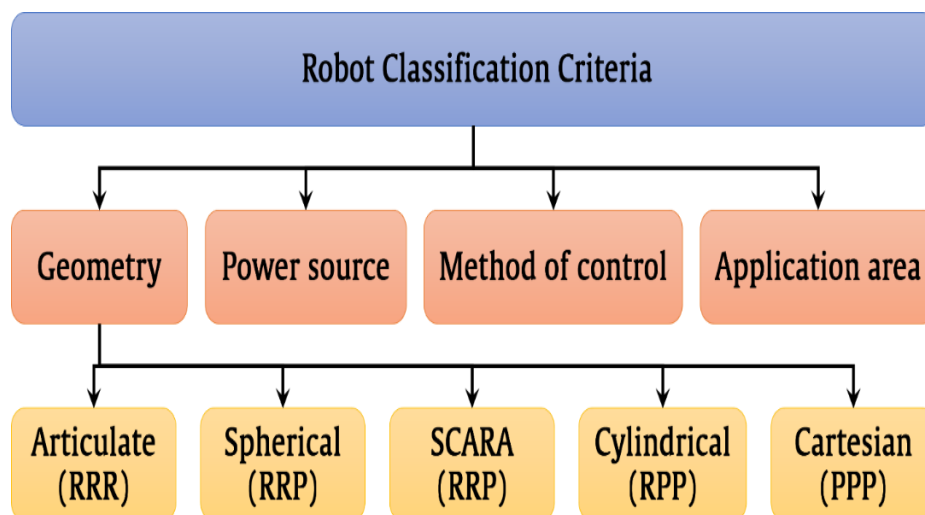


Figure 1: Classification of robots and an entrance into geometry-based robots.

Recently, numerous position and trajectory control systems have been developed and adopted. These include back stepping, robust control, Proportional-Integral-Derivative controller (PID), Intelligent control algorithms, adaptive control, and deep learning [7]. However, modern robots need to be more flexible and intelligent. In particular, the object and its location need to be identified carefully. Identifying the targeted (desired) object among many other objects is essential. Therefore, computer vision technology has been introduced as a viable solution that can be integrated with robot systems to meet the practical needs and hence achieve intelligent control systems [10]. Computer vision technology mimics human vision by

enabling computers to extract and interpret information from signals such as 2D or 3D images and videos, with a focus on image processing techniques. This allows machines to independently comprehend and analyze visual data in an automated manner.

Generally, computer vision technology can be efficiently implemented using AI and/or transformation techniques [11, 12]. Computer vision enhances the adaptability of robots using knowledge of the manipulated objects. Moreover, the techniques of ML enable the control system of a robot to learn and adapt to the unexpected variations in the surrounding environment [13]. Robotics provides an intelligent connection between perception and action, where sensors (with special attention to the vision system) handle perception while actuators perform actions. Cameras, interchangeably called vision sensors or image sensors, provide visual feedback in response to the controller's commands [14].

There are several important factors that are involved in the construction of robots and automation fields and could have a significant impact on their performance and efficiency. In general, sensors are one of the main factors that have great potential and impact on the performance of robots and automation fields. This is due to the fact that robots contain hundreds of types of sensors. Sensors can be sensitive (proprioceptive) to position, velocity, and acceleration/torque (on joints). Sensors can be sensitive (exteroceptive) to force, touch (e.g., grasping), vision, proximity, and range [14].

The other important factor that is involved in field is machine vision, which refers to both software and hardware technologies. As mentioned, computer vision is used to obtain the desired information to perform the required action, such as inspection in the industrial field, process control, and robot guidance. Computer vision-based robotics applications have recently been expanded to include various applications of robotic vision systems such as automatic sorting, dynamic grasping, vision-guided robots, visual servoing, and vision-based inspection [15]. For example, robot vision uses images to perform actual actions such as visual servoing (vision-based robot control). Visual servoing controls the movement of a machine via a feedback system based on visual feedback. Moreover, actuators are basically considered an essential part in modern engineering fields and automation systems. Specifically, they have a critical role in these fields by converting the stored energy into motion or force. They act as the muscles of robots, which allow the robots to perform different movements such as rotation of a joint, walking, and grabbing objects. Therefore, without the use of visual servoing, the robots will be static and might be incapable of performing actions properly. Besides, without actuators, robots would not be able to perform any movement or action. The versatility of actuators allows robots to be used in many applications. Recently, actuators have been integrated with sensors and control systems to enhance autonomous operation and increase the robot's ability to interact with the environment. The robotic arm must move precisely to pick up and place objects using high-quality actuators [16]. The actuator usually refers to the motor as one of the significant components that directly act on the joints and also involves other elements necessary for the operation of the entire system [14]. It is worth mentioning that one of the critical issues that arises during the design of a robot arm is the function and location of the used joints [17]. The design of the robot controller influences mechanical design decisions such as inverse kinematic calculations. These are performed analytically to enable the robot controller to make decisions and issue commands in real time [17].

Furthermore, a camera model is required to do triangulation computations that map points from a 3D space to a 2D image. The parameters of the camera model must be defined, where the computation of the parameters is called the camera calibration process [1]. Over the years, camera calibration has evolved through the use of computer to improve the robustness [18]. Typically, calibration methods use a calibration pattern to find the parameters of a camera

model, but recently, most developed methods do not use calibration patterns. Several developed methods are based on visual information to control robot manipulators. The 2D images from cameras are used to control robotic mechanisms which are based on a mapping among different coordinate systems such as Camera-Space Manipulation [19, 20].

The key contributions of this paper can be summarized as follows:

- This paper introduces an innovative non-calibrated camera mapping method, significantly enhancing the accuracy of robotic pick-and-place operations without requiring expensive or time-intensive camera calibration processes.
- A robotic arm controller was designed and implemented, leveraging machine learning and computer vision techniques to accurately identify, classify, and manipulate objects within an automated system.
- The proposed system demonstrates the capability to identify and label objects based on predefined classes and sort them, providing a robust solution for tasks that require precise object handling.
- The study provides a comparative analysis, showing that the proposed technique outperforms existing non-calibrated camera approaches by achieving higher accuracy and reduced errors, thereby establishing its superiority.

2. Related Works

Based on the literature, several methods of vision-based control have been developed so far, including stereo vision, visual servoing, and camera-space manipulation (CSM) [20-22]. These methods are primarily used for controlling serial manipulators and mobile robots. In particular, CSM is a calibration-free method that requires neither prior knowledge of the robot's Jacobian nor the calibration of the robot's geometric or camera parameters [23]. Several recent modifications have been proposed to improve the performance of CSM. For example, in [19], a modification to the classic CSM is suggested by establishing observation equations based on the matrix associated with the pin-hole camera model, instead of using the flattening procedure or combining orthographic cameras. The camera matrix, being a linear model, simplifies the estimation of view parameters and reduces computational complexity. The design of robotic arms varies based on the application and the criteria used in their implementation. In [24], a 4-DOF 3D robotic arm was proposed for object separation using a camera. The system is designed to move objects with random coordinates to predetermined locations based on camera input to a computer. The object coordinates are input into inverse kinematics to calculate the position of the robot's end effector to reach the target object. While back kinematic analysis was performed, errors remained due to inaccuracies in the servo base and arm servos.

Another proposed approach addresses misdetection caused by environmental inconsistencies such as lighting and background. This is achieved using a dual-arm 6-DoF collaborative robot, ABB YuMi, with an RGB-D camera and YOLOv5. As RGB images are processed by YOLOv5, data regarding the position and color of objects like bottles are extracted from both depth and color images, enabling the robot to adjust its movement accordingly. Additionally, a proposed method for UAV navigation employs a depth camera for obstacle avoidance [25].

A small-scale 3-DoF robot arm system is proposed for pick-and-place operations based on inverse kinematics, with the mechanical system designed in SolidWorks and controlled using an Arduino Mega 2560. The robot's movements are based on the coordinates of the pick-and-place destination [26]. Moreover, a multi-degree-of-freedom measurement platform is proposed in [27]. This enables the camera to move within the robot's motion space to capture optimal viewpoints for measuring both the robot's end effector and the assembly target. In

[15], a low-cost delta robot arm was implemented for grasping objects of unspecified size, utilizing computer vision to pick objects from conveyors. The system includes a 4-DOF delta robot with a gripper and a camera mounted above the conveyor to capture object images. While the system can pick up objects of various sizes, it would benefit from increased flexibility, such as incorporating an interface app to control joint angles or positions in 3D space.

Recent developments in digital control and advances in robotics and networked strategies are enabling greater Human-Robot Collaboration (HRC) and more collaborative environments [28]. A multi-robot arm system is proposed for object sorting based on a stereo vision system. Master and slave robots communicate via the Modbus Remote Terminal Unit (RTU) protocol, exchanging data and performing operations on demand. The stereo vision system calculates the 3D coordinates of objects by determining their centroids in two images and using the pinhole camera model formula [29].

Additionally, other work has trained a support vector machine (SVM) model to classify objects using a robot arm in an automatic sorting system. The robot arm grabs and moves objects based on predicted shapes. Object positions are identified using contouring techniques, with the object's centroid determined from the image moments of its contour. This system successfully classifies four types of objects, although it made misclassifications in 9 objects out of 1,440 total [30].

Despite advancements in robotic vision and control, several limitations remain. Camera-space manipulation methods often face precision issues due to simplified camera models and errors in kinematic analysis. Environmental factors like lighting inconsistencies continue to affect the performance of vision-based systems. While low-cost robotic arms improve flexibility, they still struggle with adapting to varying object sizes. Additionally, multi-robot sorting and machine learning approaches are limited by misclassifications, reducing accuracy and scalability. These challenges underscore the need for improved detection, control algorithms, and system adaptability.

3. Proposed Methodology

This section presents the methodology of the proposed non-calibrated camera mapping method for the robot arm system. The proposed method aims to enhance camera mapping by simplifying the essential process while improving accuracy. This is achieved by using inverse kinematics to calculate the required dimensions, rather than relying on manual measurements. The resulting camera mapping equations enable more precise determination of the object's x, y, and z coordinates relative to the robotic arm, thereby reducing errors and improving the overall system performance. In the proposed methodology, we aim to enable the robot arm to collect randomly placed objects and sort them according to their color or type. To the best of our knowledge, this represents the first approach that leverages cooperation between artificial intelligence and image processing within a robotic system.

This non-calibrated camera mapping offers a more efficient solution for vision-based robotic control. Figure (2) shows the flow diagram of the proposed method for robot arm control.

The proposed methodology for the robot arm system starts with capturing video frames from the camera. After the video frames are captured, the classification process is performed. The classification process includes several steps, which are converting the captured frames into gray scale, smoothing the images using a Gaussian filter, detecting edges to locate the object(s), and finally getting the RGB frames to classify the object type based on its color or class (AI- detected object). After detecting the object, the center of the object is located

regardless of the detecting algorithm. The object location is computed w.r.t the robot arm (in millimeters) using camera mapping, where the input is the object location (x, y, z) coordinates w.r.t camera coordinates. These locations are sent alongside the types of the object class to the controller. The robot arm is moved using inverse kinematics, where the controller transforms these locations (x, y, z) to angles to control the servo motors and grab the required object. The detection and grasping of objects are performed for a specified workspace that the robot arm can reach. In the following, the detail of the methodology is explained.

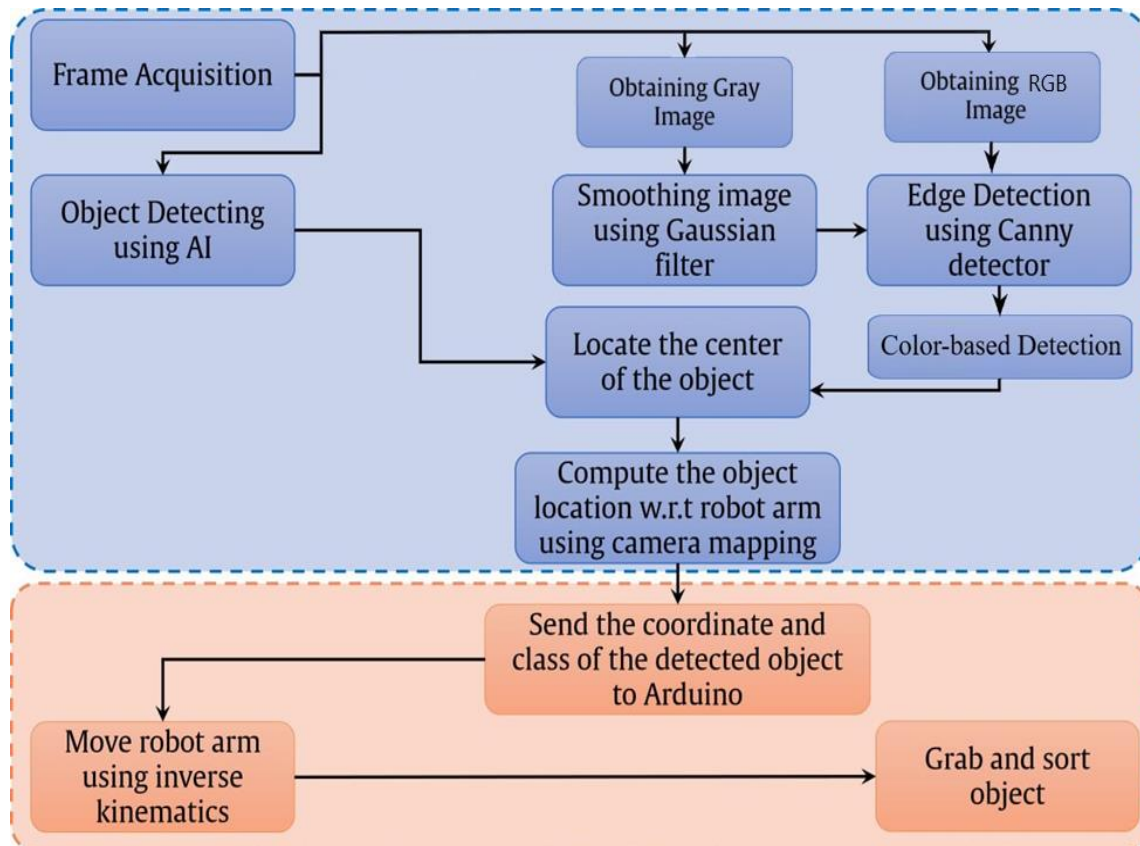


Figure 2: Flow diagram of the proposed methodology of robot arm system.

3.1 Hardware Design

The complete diagram is given in Figure (3), and each component in the hardware design is explained below. The hardware components can be classified as follows:

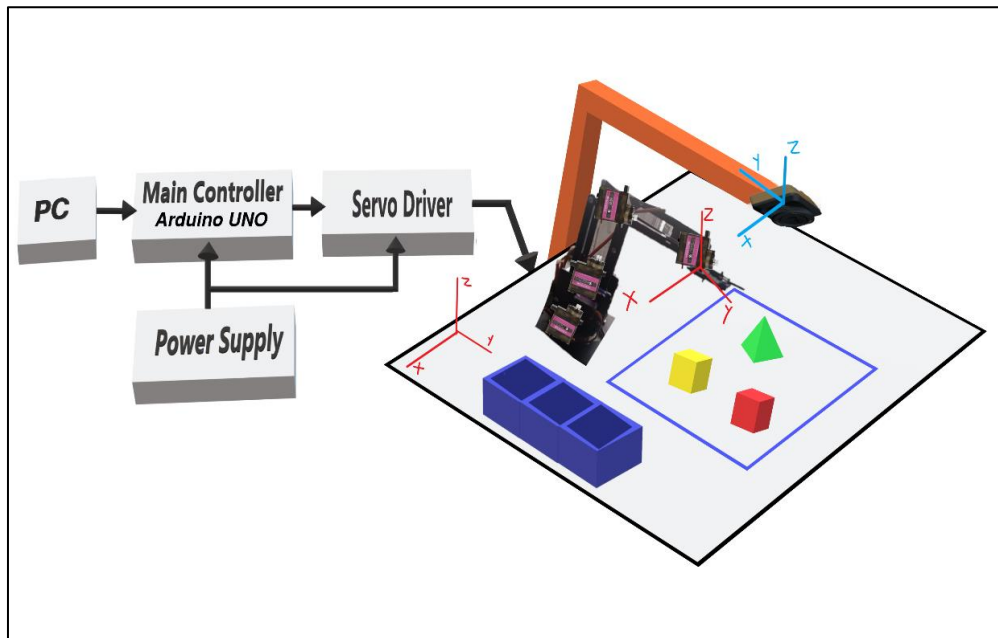


Figure 3: Circuit diagram and implementation of the system.

a) Main Controller

The main control unit consists of a computer and an Arduino Uno microcontroller. The Arduino is connected to the servo motor driver circuit through the Serial Clock (SCL) pin and Serial Data (SDA) pin for Inter-Integrated Circuit (I2C) communication. Furthermore, the Arduino is connected to the computer serially through a serial to USB cable.

b) Robot Arm

A four degrees-of-freedom (4-DOF) articulated robot arm is used in this study. This type comprises four joints (the base joint, the elbow joint, the shoulder joint, and the gripper joint). Each joint is actuated by a servo motor for precise movement and control.

c) Servo Motor Driver Circuit

The servo motors that drive the robot arm are interfaced with the Arduino via a PCA9685 servo motor driver. The +VCC and ground nodes of the driver are connected to the +VCC and Ground of the Arduino, respectively. In addition, communication between the Arduino and the driver is conducted using I2C communication by connecting the SCL and SDL ports of both components. The use of this driver minimizes signal noise and motor jitter, allowing the robot arm to move more smoothly and with greater stability than it would without the driver.

d) Camera

The camera is the most essential part of the hardware design. In this study, it has a resolution of 640×480. It connects directly to the computer, providing vision for the robot. The images are captured using the camera, and then they are processed by computer vision techniques to get information about the required objects, which is one of the main parts of our proposed methodology. As shown in Figure (3), the camera should be on top of the workspace area exactly with no tilting to ensure the success and desired accuracy of the system.

e) Power Supply

The system is powered by a 5V, 10A and 50 Watt power supply (other power supplies with similar ratings, especially the same voltage, may also be used).

3.2 Software Design

The software design can be divided mainly into two parts: the Python part and the C++ part. The Python part is conducted using PyCharm IDE, which is used to perform the following tasks:

1. Applying a 7x7 Gaussian filter (sigma = 1), a Canny edge detector (thresholds 1&2 = 0) and dilation with kernel size 5x5 to an HSV image for defining and detecting colored objects.
2. Object detection and classification into two types of classes (colored objects, and YOLOv8n custom training detection for cups' class).
3. Defining a bounding box for each object that provides information about the object location (x, y) in camera frame.
4. Benefitting from the information provided by the bounding box to do camera mapping to get the (x, y) coordinates of the object location (w.r.t the robot coordinates).
5. Communicating with Arduino serially to send the x, y and z of the object being detected (z-axis is the height of the object from the surface and it is fixed).

All of these tasks are performed and aided by a group of python libraries (OpenCV, PySerial, Ultralytics, Color Thief and Albumentations).

The C++ part is performed using Arduino IDE. The following tasks are accomplished using C++ part.

1. Receiving the (x, y and z) robot coordinates in which the z coordinate is fixed and converting them to angles by inverse kinematics using MeArm library, which is an open-source library with extension [35] for Adafruit servo driver's compatibility.
2. The coordinate's angles are then converted to PWM to control the robot arm's servo motors. The robot arm moves to the position of the object accordingly.
3. The movement of arm is done in two stages: the first stage is going to a location farther than the center of the object by some pixels, and then it moves horizontally toward the object and opens the gripper and grasps the object.
4. Finally, the Arduino communicates with the computer to confirm that the object is collected successfully.

3.2.1 Inverse Kinematics

The functionality of the Inverse Kinematics for robot arms is simply to convert cartesian location (x, y, z) of the end-effector to the joints' angles that lead to that location. The robot arm in this work is of model "phenoptics robot arm" that has base, shoulder, and elbow joints. After simplification, the final equations of Inverse Kinematics [35] of the robot arm can be given as

$$\text{Base angle: } \theta_1 = \cos^{-1}\left(\frac{x}{r}\right) \quad (1)$$

$$\text{Shoulder angle: } \theta_2 = \cos^{-1}\left(\frac{r}{R}\right) + \cos^{-1}\left(\frac{L1^2 + r^2 - L2^2}{2 * L1 * r}\right) \quad (2)$$

$$\text{Elbow angle: } \theta_3 = \theta_2 + \cos^{-1}\left(\frac{L1^2 + L2^2 - R^2}{2 * L1 * L2}\right) \quad (3)$$

where: $r = \sqrt{x^2 + y^2}$, $R = \sqrt{r^2 + z^2}$, L1 & L2 are Link lengths of robot arm.

3.2.2 System configuration

The implemented robot arm system has successfully grabbed and sorted different objects using computer vision and machine learning. The system is implemented as in the following:

- a) Firstly, the camera and arm limits are defined as given in table (1). Where cWidth = 640 and cHeight = 480 which are the image width and height in pixels respectively.

Table 1: The camera and robot arm limits for the workspace

positions	Camera limits (X,Y) coordinates in pixels	Robot limits (X,Y) coordinates in millimeters
Top left	(0,0)	(-60, 205)
Top right	(cWidth, 0)	(110, 205)
Bottom left	(0, cHeight)	(-60, 45)
Bottom right	(cWidth, cHeight)	(110, 45)

The detection and workspace area are shown in Figure (4)



Figure 4: The workspace area of the system (highlighted by a square box) the red square is exactly above the blue one.

The selected workspace area should be accessible by the robot arm including the four points as shown in Figure (5) which presents the camera limits or (x, y) points in each corner (top left, top right, bottom right, and bottom left) and its corresponding points in the robot limits (highlighted in red). Recall that cWidth is camera frame's width and cHeight is camera frame's height. The camera limits are defined depending on frame properties while the robot limits are found manually using inverse kinematics. The robot limits represent the workspace area limits in robot coordinates.

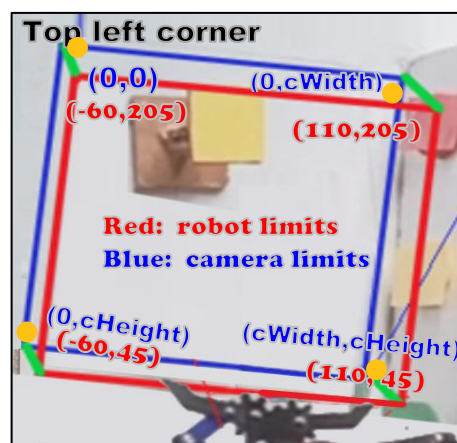


Figure 5: Robot and camera limits form in the selected workspace area of the system. Notice the orange dots that indicate the exact location of the four points.

b) The second step of system configuration is camera mapping and object identification using computer vision techniques to find the required object. The identification is according to its color or its AI-trained class. When an object is identified, it would be bounded by a box, labeled with its class and its center point (Cx, Cy) coordinates in pixels as shown in Figure (6a). The center of the object is the main point from which the system will calculate the exact x, y coordinates of the robot to collect the object and sort it. This operation of converting camera pixels coordinates to robot coordinates is called Camera Mapping.

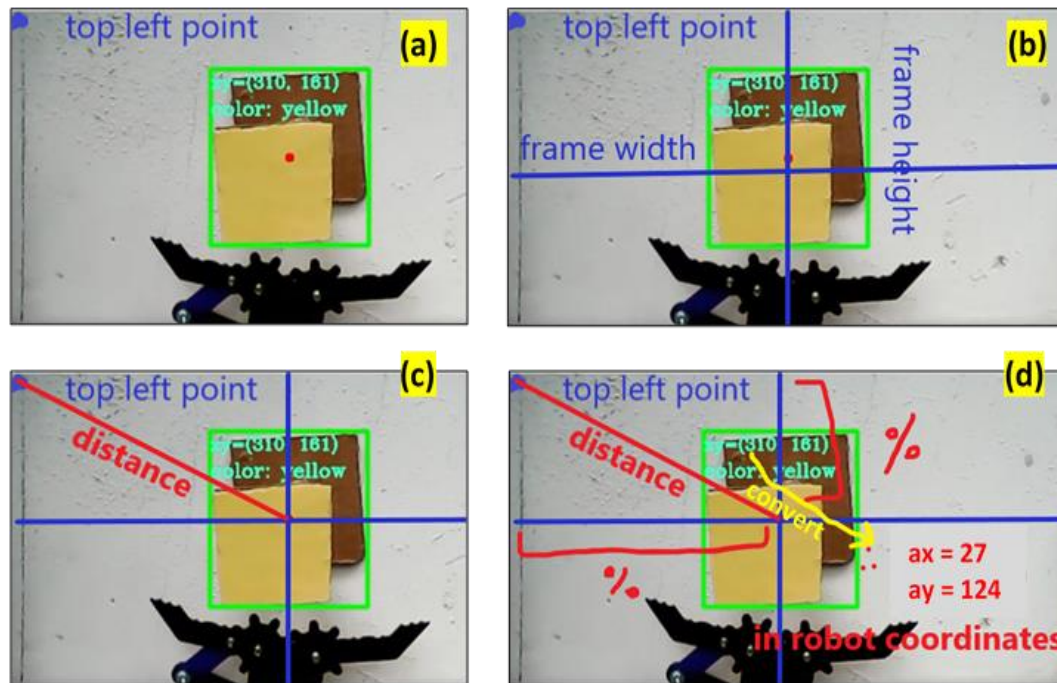


Figure 6: Camera mapping. (a) indicating top left point in reachable workspace (frame). (b) Defining frame width and height in camera coordinates. (c) Distance calculated between top left point and center position of object in camera coordinates. (d) Normalizing distance and converting it into robot coordinates to get ax, ay location of the object in robot coordinates.

A camera mapping method is introduced that is both easy to implement and capable of delivering satisfactory accuracy, even when using non-calibrated, low-cost cameras. The process involves the following steps: Finding the width and height of the workspace in robot coordinates in millimeters either using a ruler in real world or performing:

$$Rw = |R_{TopRightX}| - |R_{TopLeftX}| \quad (4)$$

$$Rh = |R_{BottomLeftY}| - |R_{TopLeftY}| \quad (5)$$

where Rw and Rh are width and height of workspace area in Robot coordinates, respectively. RTopRightX is X value of Top Right point in Robot coordinates, and the same concept for the others.

From Figure (6c), the parameter d is the distance between the center of the object and the top left point in camera coordinates (in pixels), which is calculated as in the following:

$$dstX = |Cx| - |CTopLeftX| \quad (6)$$

$$dstY = |Cy| - |CTopLeftY| \quad (7)$$

Where: CTopLeftX is X value of Top Left point in Camera coordinates. Cx and Cy are centered (x, y) location of object in Camera coordinates, respectively.

Normalize the distances (in order to see how far the object is from top left point in %): (see Figure 6d)

$$N_dstX = (dstX)/Fw \quad (8)$$

$$N_dstY = (dstY)/Fh \quad (9)$$

where: Fw and Fh are frame width and frame height of workspace in camera coordinates, respectively. N_dstX and N_dstY are the Normalized form of dstX and dstY respectively.

The last step is simply converting the normalized distance to distance in robot-coordinates with the top left point of robot coordinates as a reference:

$$Rx = RTopLeftX + Rw \times N_dstX \quad (10)$$

$$Ry = RTopLeftY + Rh \times N_dstY \quad (11)$$

where Rx and Ry are X and Y values of object in the Robot coordinates respectively. Rw and Rh are width and height of workspace area in Robot coordinates, respectively. RTopLeft is the Top Left point (X, Y) in Robot coordinates.

The resulting values of Rx, Ry are sent to the Arduino along with the object's class, then the inverse kinematics will convert Rx, Ry with a specified Z values to angles so that it goes to the detected object correctly.

Noting that the camera mapping equations are provided in general forms, which means that they can be used in any coordinated system due to the fact that some coordinated systems have X-axis that start from the left, Y-axis that starts from the center or have negative & positive values.

3.3 Robot arm grasping

The robot arm coordinates are sent to Arduino, where they are converted to angles using inverse kinematics in order to move the robot arm. Each object is equipped with a handle in order to grasp the object without letting it slip from the gripper. After the gripper approaches the location just before the object's location by 30mm, it moves forward in order to grasp the handle of the object and pick it up, then moves to the correct container.

3.4 YOLO model training

In this work, the YOLO model is trained (fine-tuned) to detect the second type of objects (AI detected objects) using the light-weighted model type (YOLOv8n) or the state-of-the-art model in real-time applications. The custom training is done on the new custom class "cup" which requires changing the model's weights to get the desired class detected (fine-tuning). The training phase is done by training the model with 1280 images with augmentations and preprocessing (blurring, brightness adjustment, hue channel), where all of the mentioned augmentations are used to improve the generalization of the model and reduce overfitting. The dataset's percentages are 88.88% for training, 5.56% for validation, and 5.56% for testing. Finally, the performance measurements will state the accuracy of the model after training. The figure below shows the block diagram of the general steps for training this model.

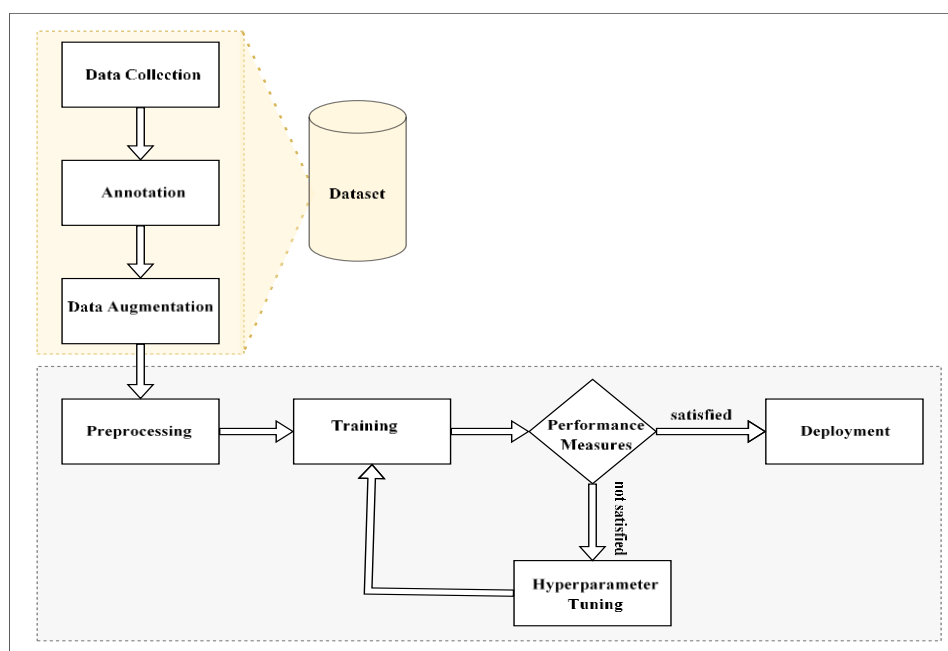


Figure 7: Block diagram showing the steps of training(fine-tuning) the YOLO model

4. Final stage of the methodology (Detection)

The following experiments show the steps of detecting the two types of objects: which are colored objects and AI detected objects

a) Colored objects

Colored objects are detected using computer vision techniques in order to define the presented color for the robot arm. It is done after smoothing the image using Gaussian filter 7x7 kernel with sigma equal to 1 and masking it to avoid noise. The edge of the object is detected using the Canny edge detector (thresholds 1&2 = 0). Finally, the Color Thief library is used to get the majority color within the bounded object. Figure (8) shows a sample object and with the masked view using OpenCV in python language.

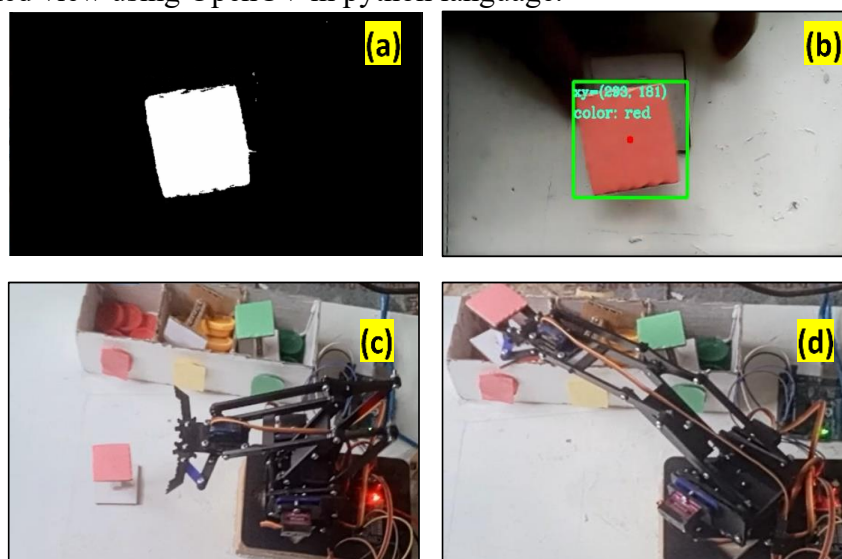


Figure 8: (a) illustrates the masked view after a series of image processing functions on colors. (b) Shows the detected color object in real-time (even in such environments of changed brightness). (c) Grabbing the object after the proposed camera mapping method. (d) Bringing the object to the suitable container after being grabbed.

b) AI detected objects

AI detected objects are the second type used in this work. The model used in this study is YOLOv8n. Note that the smallest type of YOLOv8 is used, which is the nano version "YOLOv8n". However, the detection is fast, accurate, and well suited for real-time detection. Figure (9) shows the predicted sample object in real-time detection.

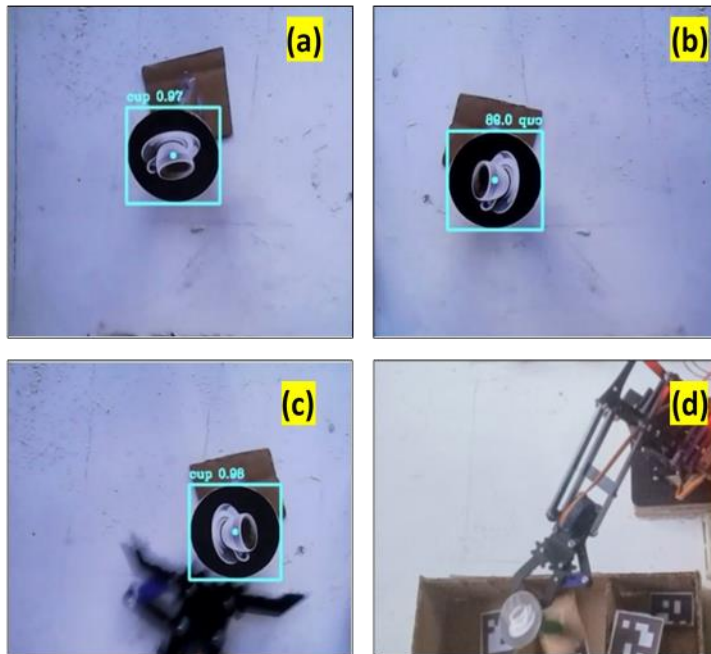


Figure 9: (a) AI-detected object using YOLOv8n custom training. (b & c) Same object with different orientations. (d) The robot arm brings the object to the suitable container

In this work, the dataset is collected manually regarding machine learning to detect objects of the custom class "cup". Figure (10) shows a subset of the custom dataset that is fed to the YOLO model.



Figure 10: A Subset of the custom dataset for "cup" detected objects.

5. Results and Discussion

The accuracy of the pick and place comes from finding the locations of the objects with fewer errors, so the proposed camera mapping approach is evaluated thoroughly. The results of the evaluation of the proposed method are provided in Table (3). The error function used in the table is calculated as follows:

$$\text{error}\% = \left| \frac{\text{actual value} - \text{reading value}}{\text{reading value}} \right| \times 100\% \quad (12)$$

The selected components used to make these results in this study are listed below

Table 2: The components used in this study

Component	Description
MG90	A 9g Servo Motor-Micro Metal Gear for Robots applications
PCA9685	Servo driver (I ² C-bus controller) has the ability to handle and control up to 16 servo motors.
Arduino UNO	One of the most popular development boards, programmable with c/c++, has multi-functional pins.
Camera	A 640x480 USB camera for computer vision support for machines.
Power Source	Either battery or power supply. In this work, it is a power supply of 5v, 10 Ampere and 50Watt.

The evaluation is carried out by choosing a number of random actual locations (x,y) in the real world and comparing them with the ones calculated using the proposed camera mapping approach.

Tables (3) and (4) show comparisons of the proposed method with two other camera approaches given in [31] and [24]. The comparison with [31] is shown in the last column in Table (3), which was implemented entirely in the same environment as the one proposed in this work. The results show that the method in [31] produced an error of 5.167% using the camera mapping, which is much higher than the error provided using the proposed camera mapping. This signifies the importance of our proposed method in providing much higher accuracy in comparison with the state-of-the-art method.

Table 3: Comparison of the proposed camera mapping versus the method proposed in [31].

Sample point	axis	actual coordinate in (mm)	Using proposed method		Using [31] method	
			Reading coordinate from Camera	error%	reading coordinate from Camera	error%
1	X	-40	-40.1667	0.4149%	-38	5.2631%
	Y	140	140.17	0.1212%	136	2.9411%
2	X	-28	-27.4166	2.1276%	-26.5	5.6603%
	Y	173	174.25	0.7173%	177	2.2598%
3	X	-22	-20.3334	8.1967%	-20	10%
	Y	155	157.16	1.3743%	149	4.0268%
4	X	38	36.6875	3.5775%	41	7.3170%
	Y	127	130.87	2.9571%	121	4.9586%
5	X	54	53.3333	1.2500%	55.5	2.7027%
	Y	148	144.82	2.1958%	145	2.0689%
6	X	67	67.5	0.7407%	65.5	2.2900%

The normalized confusion matrix results for training the YOLOv8n model on cup class is shown in Figure (11) which presents a 2x2 matrix due to one class plus background.

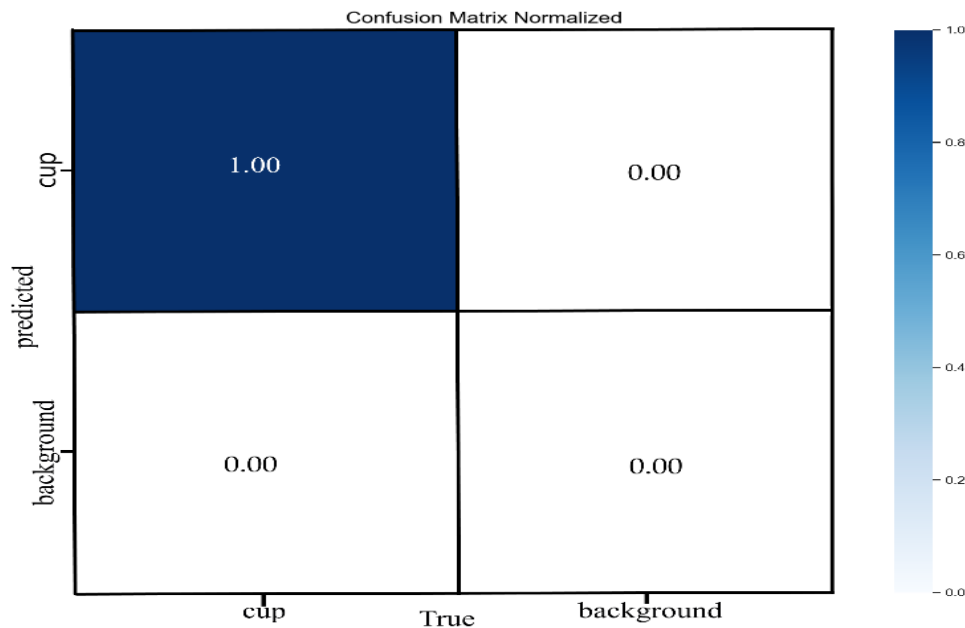


Figure 11: The normalized confusion matrix

6. Conclusion

This paper considered the integration of robotic systems with machine learning and computer vision. To this end, the applicability of the proposed non-calibrated camera mapping method in industrial automation and robotic systems was investigated, presenting a practical and cost-effective solution for enhancing precision in automated operations. The performance of the proposed approach was rigorously evaluated, achieving an impressively low error rate of 2.458% in object localization, demonstrating its reliability and potential for real-world applications. In addition, this paper provided a comparative analysis, showing that the proposed technique outperforms existing non-calibrated camera approaches by achieving higher accuracy and reduced errors, thereby establishing its superiority.

References

- [1] R. K. Kanna, L. Hema, V. S. Ramya, N. Kripa, R. Gomlavalli, and A. Ambikapathy, "Smart Electronic Arm Module using Arduino Applications," in *2022 IEEE International Conference on Current Development in Engineering and Technology (CCET)*, 2022: IEEE, pp. 1-5.
- [2] W. Mark, "Spong, Seth Hutchinson, and M," *Vidyasagar, Robot Modeling and Control*, 2008.
- [3] S. Team. "Automation and Robotics." spatial. <https://blog.spatial.com/automation-and-robotics> (accessed Tue Dec 07, 2021).
- [4] M. B. Imtiaz, Y. Qiao, and B. Lee, "Prehensile and non-prehensile robotic pick-and-place of objects in clutter using deep reinforcement learning," *Sensors*, vol. 23, no. 3, p. 1513, 2023.
- [5] R. Gautam, A. Gedam, A. Zade, and A. Mahawadiwar, "Review on development of industrial robotic arm," *International Research Journal of Engineering and Technology (IRJET)*, vol. 4, no. 03, p. 429, 2017.
- [6] S. Dheeban, D. Harish, A. H. Vignesh, and M. Prasanna, "Arduino controlled gesture robot," in *2018 IEEE 4th International Symposium in Robotics and Manufacturing Automation (ROMA)*, 2018: IEEE, pp. 1-6.
- [7] A. A. Mohammed and M. Sunar, "Kinematics modeling of a 4-DOF robotic arm," in *2015 International Conference on Control, Automation and Robotics*, 2015: IEEE, pp. 87-91.

- [8] H. Z. Ting, M. Hairi, M. Zaman, M. Ibrahim, and A. Moubark, "Kinematic analysis for trajectory planning of open-source 4-DoF robot arm," *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, pp. 769-777, 2021.
- [9] R. Mourya, A. Shelke, S. Satpute, S. Kakade, and M. Botre, "Design and implementation of pick and place robotic arm," *Int. J. Recent Res. Civ. Mech. Eng.*, vol. 2, no. 1, pp. 232-240, 2015.
- [10] B. X. Duong et al., "Deep Learning based 6DOF Robot Arm Control with Computer Vision," *Internasional Journal of Engineering Research & Technology (IJERT)*, vol. 10, no. 2, pp. 531-536, 2021.
- [11] B. M. Mahmmoud et al., "Performance enhancement of high degree Charlier polynomials using multithreaded algorithm," *Ain Shams Engineering Journal*, vol. 15, no. 5, p. 102657, 2024.
- [12] B. M. Mahmmoud, W. N. Flayyih, Z. H. Fakhri, S. H. Abdulhussain, W. Khan, and A. Hussain, "Performance enhancement of high order Hahn polynomials using multithreading," *Plos one*, vol. 18, no. 10, p. e0286878, 2023.
- [13] N. M. Gomes, F. N. Martins, J. Lima, and H. Wörtche, "Reinforcement learning for collaborative robots pick-and-place applications: A case study," *Automation*, vol. 3, no. 1, pp. 223-241, 2022.
- [14] A. Aiello, "Robotic arm pick-and-place tasks: Implementation and comparison of approaches with and without machine learning (deep reinforcement learning) techniques," Politecnico di Torino, 2020.
- [15] P. L. Hoai, V. D. Cong, and T. T. Hiep, "Design a low-cost delta robot arm for pick and place applications based on computer vision," *FME Transactions*, vol. 51, no. 1, pp. 99-108, 2023.
- [16] WEVOLVER. "Robot Actuators: A Comprehensive Guide to Types, Design, and Emerging Trends." <https://www.wevolver.com/article/robotic-actuators-the-muscle-power-of-industry-40> (accessed 04 Dec, 2023).
- [17] S. Dikmenli, "Forward & inverse kinematics solution of 6-dof robots those have offset & spherical wrists," *Eurasian Journal of Science Engineering and Technology*, vol. 3, no. 1, pp. 14-28, 2022.
- [18] J.-Y. Guillemaut, A. S. Aguado, and J. Illingworth, "Using points at infinity for parameter decoupling in camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 265-270, 2005.
- [19] J. M. Rendón-Mancha, A. Cárdenas, M. A. García, E. González-Galván, and B. Lara, "Robot positioning using camera-space manipulation with a linear camera model," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 726-733, 2010.
- [20] E. J. González-Galván and M. J. Seelinger, "Efficient camera-space target disposition in a matrix of moments structure using camera-space manipulation," *The International Journal of Robotics Research*, vol. 18, no. 8, pp. 809-818, 1999.
- [21] Y. Xie, W. Wang, C. Zhao, S. B. Skaar, and Q. Wang, "A Differential Evolution Approach for Camera Space Manipulation," in *2020 2nd World Symposium on Artificial Intelligence (WSAI)*, 2020: IEEE, pp. 103-107.
- [22] P.-J. Hwang, C.-C. Hsu, P.-Y. Chou, W.-Y. Wang, and C.-H. Lin, "Vision-Based Learning from Demonstration System for Robot Arms," *Sensors*, vol. 22, no. 7, p. 2678, 2022.
- [23] A. González, E. J. Gonzalez-Galvan, M. Maya, A. Cardenas, and D. Piovesan, "Estimation of camera-space manipulation parameters by means of an extended Kalman filter: Applications to parallel robots," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, p. 1729881419842987, 2019.
- [24] A. Fahruz, B. S. Agomo, and Y. A. Prabowo, "Design Of 4DOF 3D robotic arm to separate the objects using a camera," *International Journal of Artificial Intelligence & Robotics (IJAIR)*, vol. 3, no. 1, pp. 27-35, 2021.
- [25] J. Hu, Y. Niu, and Z. Wang, "Obstacle avoidance methods for rotor UAVs using RealSense camera," in *2017 Chinese Automation Congress (CAC)*, 2017: IEEE, pp. 7151-7155.
- [26] A. R. Al Tahtawi, M. Agni, and T. D. Hendrawati, "Small-scale robot arm design with pick and place mission based on inverse kinematics," *Journal of Robotics and Control (JRC)*, vol. 2, no. 6, pp. 469-475, 2021.
- [27] Y. Liu, J. Zhou, Y. Li, Y. Zhang, Y. He, and J. Wang, "A high-accuracy pose measurement system for robotic automated assembly in large-scale space," *Measurement*, vol. 188, p. 110426, 2022.

- [28] L. Caruana and E. Francalanza, "A safety 4.0 approach for collaborative robotics in the factories of the future," *Procedia Computer Science*, vol. 217, pp. 1784-1793, 2023.
- [29] C. D. Vo, D. A. Dang, and P. H. Le, "Development of Multi-Robotic Arm System for Sorting System Using Computer Vision," *Journal of Robotics and Control (JRC)*, vol. 3, no. 5, pp. 690-698, 2022.
- [30] V. D. Cong and T. T. Hiep, "Support vector machine-based object classification for robot arm system," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 13, no. 5, pp. 5047-5053, 2023.
- [31] D. Kijdech and S. Vongbunyon, "Pick-and-place application using a dual arm collaborative robot and an RGB-D camera with YOLOv5," *IAES International Journal of Robotics and Automation (IJRA)*, vol. 12, no. 2, p. 197, 2023.