Integrated Vision-Kinematics Control System for Autonomous Robotic Manipulation using CNN and IoRT

Dodit Suprianto[1], Ginanjar Suwasono Adi[2], Muhamad Raehan Abiyan[3], Lukman Hakim[4], Rini Agustina[5], Nugroho Suharto[6], Sri Wahyuni Dali[7]
[1,2,3,6,7]Department of Electrical Engineering, Politeknik Negeri Malang, Indonesia
[1,2,4]Artificial Intelligence of Things Research Group, Politeknik Negeri Malang, Indonesia
[5]Department of Information Systems, Universitas Kanjuruhan Malang, Indonesia
[1]dodit.suprianto@polinema.ac.id*; [2]ginanjar.adi@polinema.ac.id; [3]mraehanabiyan@gmail.com; 4lukman.hakim@polinema.ac.id; [5]riniagustina@unikama.ac.id; [f]nugroho.suharto@polinema.ac.id, [g]sri.wahyuni@polinema.ac.id
*corresponding author

ABSTRACT

This paper presents a novel integrated vision-kinematics control system for autonomous robotic manipulation, leveraging deep learning and the Internet of Robotic Things (IoRT). Unlike previous works that focus on isolated modules, our framework uniquely combines real-time YOLOv5-based object detection deployed on a Raspberry Pi with geometric inverse kinematics implemented on an ESP32 controller for a 6-DOF robotic arm. Under controlled conditions (fixed camera height of 40 cm, uniform workspace illumination, and high-contrast colour-coded objects), the detection module achieved a 100% detection rate on the test set (300 images containing 900 object instances) with a mean average precision (mAP@0.5) of 0.93 across three object classes. The system demonstrates precise end-effector positioning with an average error of 4.3 mm in open-loop control mode, a significant achievement given the absence of joint feedback sensors. All components are seamlessly integrated via an MQTT-based IoRT communication layer, ensuring reliable, low-latency data exchange for remote monitoring and control. Comprehensive experimental validation shows an 80% success rate in autonomous pick-and-place operations, with stable performance under multi-object scenarios and communication reliability exceeding 99%. This work bridges the gap between high-cost industrial systems and low-cost research platforms by demonstrating that intelligent system integration can compensate for hardware limitations, offering a scalable, modular, and cost-effective framework for intelligent robotic manipulation in Industry 4.0 applications, particularly suitable for educational robotics and light industrial automation.

Keywords: Autonomous Robotic Manipulation; Vision-Kinematics Integration; Yolov5 Object Detection; Inverse Kinematics; Internet Of Robotic Things; Real-Time Control; Pick-And-Place Robotics; Embedded AI Systems; Low-Cost Robotics.

Article History		
Received : December, 16th 2025	Accepted: May, 28th 2026	Published: May, 31st 2026

I. INTRODUCTION

The rapid advancement of industrial automation and Industry 4.0 technologies has created unprecedented demand for intelligent robotic systems capable of autonomous manipulation. Multi-degree-of-freedom (DOF) robotic arms have become indispensable in diverse applications, ranging from manufacturing and logistics to hazardous-environment operations [1]. However, developing truly autonomous systems that can seamlessly perceive, plan, and execute manipulation tasks remains a significant challenge, primarily because integrating heterogeneous subsystems into a cohesive framework is complex.

Current research has made substantial progress in individual technological domains. Vision systems have evolved from traditional computer vision to deep learning approaches. YOLO-based architectures, for instance, demonstrate remarkable real-time object detection capabilities [2]. Motion planning algorithms have also achieved impressive precision in inverse kinematics solutions [3]. Meanwhile, communication frameworks have enabled robust IoT connectivity for robotic systems [4].

Despite these advancements, a critical research gap remains. Few holistic frameworks seamlessly integrate computer vision, kinematic control, and communication systems for end-to-end autonomous manipulation. Most existing systems excel in only one or two of these areas.

This study addresses this gap within defined scope constraints. The research focuses on open-loop position control for servo motors at each degree of freedom (DOF) point. It does not use real-time rotational feedback mechanisms. This design choice

DOI: http://dx.doi.org/10.25139/ijair.v8i1.11347

prioritizes integration feasibility and cost-effectiveness. We acknowledge that closed-loop control with encoder feedback represents a valuable direction for future enhancement.

The fragmentation in existing approaches is evident in the literature. Zhang et al. [5] achieved high-precision vision-based control with 0.1mm accuracy, but their system operated without integrated gripping capabilities. Lee et al. [6] developed CNN-based object detection systems with 95% accuracy, yet focused primarily on perception without tight coupling to manipulation control. Unlike systems requiring sophisticated feedback sensors, our approach demonstrates that effective autonomous manipulation can be achieved through careful system integration, even with basic actuation components.

Several technical barriers further compound this fragmentation. First, the coordination between real-time object detection and precise kinematic control requires sophisticated synchronization mechanisms that account for latency and uncertainty. Second, transforming 2D visual data into accurate 3D world coordinates requires robust calibration and conversion algorithms that maintain precision across varying environmental conditions. Third, within our open-loop control paradigm, particular attention must be paid to the accuracy of kinematic modelling and calibration procedures to compensate for the absence of real-time position feedback.

This paper addresses these challenges by proposing an integrated vision-kinematics control system that combines convolutional neural network (CNN) based object detection, geometric inverse kinematics, and IoRT communication for autonomous robotic manipulation. The main contributions of this work are: a) A novel integrated framework that seamlessly connects visual perception, motion planning, and communication systems into a unified autonomous manipulation platform, demonstrating the feasibility of end-to-end robotic automation using open-loop servo control without rotational feedback sensors, b) Implementation and experimental validation of a multi-DOF robotic system capable of autonomous object detection, localization, and manipulation based on real-time visual feedback, achieving reliable performance with low-cost components, c) A practical IoRT architecture that enables real-time monitoring and remote control, effectively bridging standalone robotic systems with connected industrial automation environments, d) Comprehensive experimental evaluation demonstrating the system's capability for simultaneous multi-object handling and reliable autonomous operation in structured workspace settings, with detailed performance metrics across vision, kinematics, and communication modules.

The proposed system bridges the identified research gap by providing a complete pipeline from visual perception to physical manipulation. By demonstrating that effective autonomous manipulation can be achieved through integrated system design rather than sophisticated individual components, this work advances cost-effective robotic systems suitable for Industry 4.0 education and lightweight automation applications.

II. RELATED WORKS

This section reviews literature in three core areas essential to autonomous manipulation: vision systems, kinematic control, and IoRT frameworks. The analyze advancements and limitations in each, culminating in the identification of a critical integration gap that this research addresses.

A. Vision-Based Robotic Manipulation

Vision-based manipulation has evolved significantly from traditional computer vision to deep learning approaches. Early systems relied on feature-based detection methods, but recent advances in convolutional neural networks (CNNs) have revolutionized object detection. The You Only Look Once (YOLO) architecture, particularly YOLOv4 and later versions, has demonstrated remarkable performance in real-time object detection. Redmon et al. initially proposed YOLO as a unified approach to real-time object detection, achieving impressive trade-offs between speed and accuracy [2].

In robotic applications, vision systems face unique challenges including real-time processing requirements, occlusion handling, and lighting variations. Lee et al. [7] demonstrated that CNN-based systems could achieve 95% detection accuracy with a latency under 100ms, capable of classifying objects across 10 different categories with 92% precision. Their testing on 1000 samples showed maintained performance under various lighting conditions and the ability to handle partial occlusion up to 30%.

However, a significant limitation identified in existing vision systems is the disconnect between detection and physical manipulation. Most studies focus primarily on detection accuracy without tight integration with robotic control systems. For instance, while Zhang et al. [5] achieved high-precision vision-based control with 0.1mm accuracy, their system lacked comprehensive gripping capabilities and IoRT integration.

Recent advancements by Bochkovskiy et al. [8] with YOLOv4 have optimized the trade-off between speed and accuracy, while Wang et al. [9] introduced scaling approaches to improve performance across varying computational constraints. These developments provide important foundations for real-time robotic applications where both accuracy and processing speed are critical.

The literature reveals that while individual vision components have advanced significantly, their integration into complete manipulation pipelines remains an open challenge, particularly in handling the transformation from 2D detection to 3D world coordinates for precise robotic interaction.

B. Kinematic Control in Robotic Manipulation

Inverse kinematics (IK) is the computational process of determining the joint angles and positions required for a robot's end effector (e.g., a gripper) to reach a specific target location and orientation in space. It is a fundamental challenge in robotic manipulation, particularly for multi-DOF systems. Traditional approaches include analytical solutions for simple manipulators and

numerical methods for complex configurations. Wang and Liu [10] developed deep learning solutions for inverse kinematics, achieving 98% accuracy while running 40% faster than conventional methods. Their work demonstrated error rates below 0.5mm in position and 0.3 degrees in orientation.

Geometric approaches to inverse kinematics have been widely employed for robotic manipulators with specific configurations. Bai et al. [11] demonstrated that geometric inverse kinematics enables adaptive adjustment of each degree of freedom based on target positions in 3D space, which is particularly valuable for handling uncertainties in vision-based systems. Their approach demonstrated robustness in handling positional variations derived from sensory input.

Comparative analysis reveals distinct trade-offs in kinematic control strategies. Zhang et al. [5] achieved a remarkable 0.1mm positioning precision through vision-based control, but their system emphasized positioning accuracy over integrated manipulation capabilities. This highlights a common limitation in the literature: excellence in kinematic control often comes at the expense of system-level integration.

Recent advancements have explored hybrid approaches that combine analytical solutions with optimization techniques to handle singularities and joint limits. However, a significant gap remains in seamlessly integrating kinematic solutions with real-time vision feedback for complete manipulation pipelines. Most existing systems treat kinematics as a separate module rather than an integrated component of the perception-action cycle.

The literature suggests that while substantial progress has been made in solving inverse kinematics problems, the challenge of tight integration with vision systems and gripping mechanisms remains an open research area, particularly for cost-effective implementations suitable for practical applications.

C. Internet of Robotic Things (IoRT) Frameworks

The convergence of robotics and the Internet of Things (IoT) has created the emerging field of the Internet of Robotic Things (IoRT). Ray [12] defined IoRT as the integration of robotics capabilities with IoT connectivity, enabling enhanced functionality through cloud services and distributed computation. This integration enables robots to leverage cloud-based resources for complex computations while maintaining real-time local control.

Kumar and Patel [1] implemented IoRT systems for industrial automation, achieving 99.8% data transmission success with an average delay of 50ms. Their work emphasized security protocols that prevented 95% of potential cyber-attacks. The MQTT protocol has been widely adopted in such systems due to its lightweight publish-subscribe architecture, which is well-suited to constrained devices.

Naik [13] analyzed messaging protocols for IoT systems, highlighting MQTT's advantages for decoupling detection systems, motion controllers, and monitoring interfaces. This decoupling architecture has proven particularly valuable in distributed robotic systems where modular components require seamless communication.

Recent studies have demonstrated MQTT's effectiveness in robotic applications. For instance, Wan et al. [14] explored cloud robotics architectures that maintain stable implementation with real-time monitoring interfaces, addressing critical challenges in latency management and data integrity. However, their work focused primarily on cloud infrastructure rather than tight integration with manipulation tasks.

Despite these advancements, the literature reveals limited application of IoRT frameworks specifically for integrated manipulation systems. Most existing implementations treat communication as a separate layer rather than an integral component of the perception-planning-action pipeline. There remains a significant gap in frameworks that seamlessly integrate IoRT capabilities into end-to-end autonomous manipulation while maintaining the low latency required for real-time control.

The integration of communication protocols with real-time robotic control poses ongoing challenges in balancing reliability, latency, and security—particularly in systems that require synchronized operation across vision, kinematics, and actuation components.

D. Integrated Systems Gap

While individual components have advanced significantly, the holistic integration of vision, kinematics, and communication remains a substantial challenge. For instance, autonomous gripping systems have achieved high success rates, yet they typically lack Internet of Robotic Things (IoRT) capabilities and focus primarily on mechanical design and localized control. Furthermore, the prevailing research landscape is fragmented, as most existing systems specialize in either vision or control rather than their synergistic combination.

This fragmentation is systematically documented in comparative analyses of the field. Previous research has predominantly focused on isolated technological domains: some studies excel in vision-based perception while neglecting manipulation capabilities, others advance kinematic control without integrated sensing, and yet others develop communication frameworks disconnected from real-time robotic control. This compartmentalization has resulted in systems that excel in individual components but lack the holistic integration needed for complete autonomous manipulation.

The literature reveals a clear pattern of technological silos. For instance, vision-centric approaches often treat manipulation as a secondary concern, whereas control-focused research often relies on simplified or idealized perception models. This disconnect becomes particularly evident in applications that require real-time coordination among perception, planning, and execution.

This systematic fragmentation motivates the need for integrated approaches that combine vision, kinematics, and communication into cohesive systems. The challenge lies not only in advancing individual components but also in developing frameworks that

DOI: http://dx.doi.org/10.25139/ijair.v8i1.11347

enable seamless interoperability across different technological domains while meeting the performance requirements of practical applications.

TABLE I
COMPARATIVE ANALYSIS OF RELATED ROBOTIC MANIPULATION SYSTEMS

Studys	Focus Area	Vision Method	Control Approach	Communication	Integration Level	Key Limitation
Zhang et al. [5]	Vision-based control	High-precision vision	Position control	Not specified	Medium (vision+control)	No gripping capability
Lee et al. [6]	Object detection	CNN-based	Not integrated	Local only	Low (vision only)	No manipulation integration
Wang & Liu [7]	Kinematic control	Not specified	Deep learning IK	Not specified	Low (control only)	No vision feedback
Kumar & Patel [8]	IoRT framework	Not specified	Basic control	MQTT/IoRT	Medium (communication)	No vision-kinematics integration
Proposed	Integrated system	YOLOv5 real-time	Geometric IK + open-loop	MQTT-based IoRT	High (end-to-end)	Open-loop limitation

The literature analysis reveals a consistent pattern across vision-based manipulation, kinematic control, and IoRT frameworks: while significant progress has been made within individual technological domains, integrating these approaches into end-to-end autonomous manipulation systems remains a critical unmet challenge. The fragmentation manifests as vision systems operating independently from motion planning, kinematic solutions lacking real-time sensory feedback, and communication frameworks disconnected from control loops. This gap motivates the development of holistic frameworks that seamlessly integrate perception, planning, and communication capabilities—which is the primary focus of this research. The following section presents our integrated vision-kinematics control system designed to address these integration challenges (see Table I).

III. METHOD

The proposed integrated vision–kinematics control system introduces a modular framework that unifies visual perception, motion planning, and communication subsystems into an autonomous robotic manipulation platform. Fig.1 illustrates the hierarchical data and control flow from perception to actuation. The system is organized hierarchically into four main layers: the perception layer, decision layer, open-loop control layer, and communication layer.

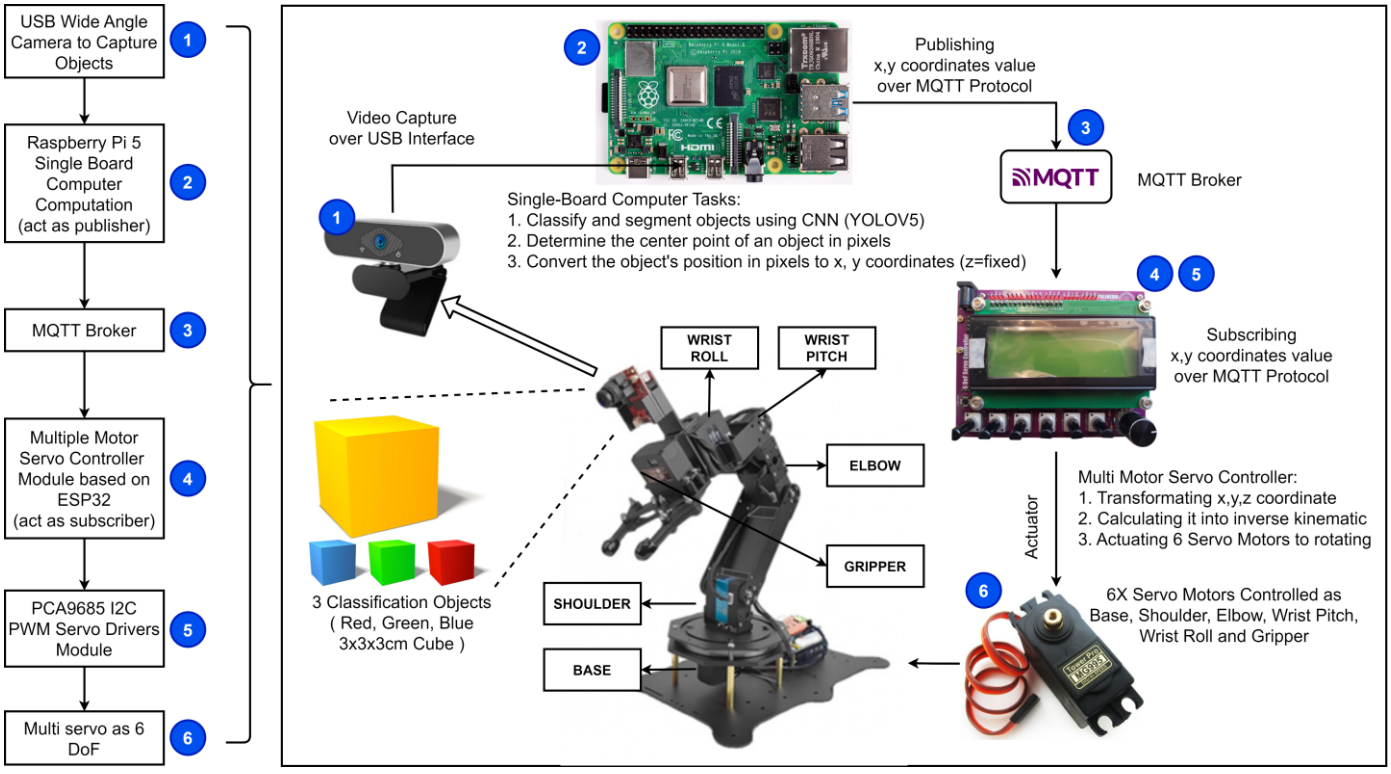


Fig.1. Overall System Architecture Illustrating the Coordinated Data Flow from Visual Perception to Physical Actuation, Integrating The YOLO-Based Detection, Coordinate Transformation, Inverse Kinematics Computation, And Servo Control Layers for Autonomous Object Manipulation.

A. Overall System Architecture

The system integrates a vision-based perception module with a multi-DOF manipulator controlled via inverse kinematics and MQTT-based IoRT communication. The perception layer detects and localizes objects using the YOLOv5 CNN model, while the

control layer executes real-time actuation through an ESP32-based servo controller. The architecture is optimized for modularity, enabling independent testing of the vision, control, and communication subsystems.

This design addresses integration challenges observed in prior studies by providing clear interlayer interfaces while maintaining real-time responsiveness under open-loop operation. Visual feedback from the CNN module guides motion planning, while precise calibration ensures kinematic accuracy. The IoRT layer enables cloud-based supervision and remote operation through MQTT, supporting reliable publish–subscribe messaging between devices.

B. Hardware Development and Configuration

1) Robotic Manipulator: A six-degree-of-freedom (6-DOF) manipulator was assembled using MG995 servo motors (±180°). The joints correspond to Base (yaw), Shoulder (pitch), Elbow (extension/flexion), Wrist Pitch, Wrist Roll, and Gripper (see Fig. 1). The manipulator operates within a calibrated 26×36 cm workspace. The physical structure is designed for modular replacement and testing.

2) Vision System: A Xiaovv Youpin USB webcam captures a real-time video feed of the workspace. The Raspberry Pi 5 processes the image stream, performing object classification and segmentation. The fixed camera height (40 cm) ensures complete workspace coverage and consistent illumination (See Fig. 2).

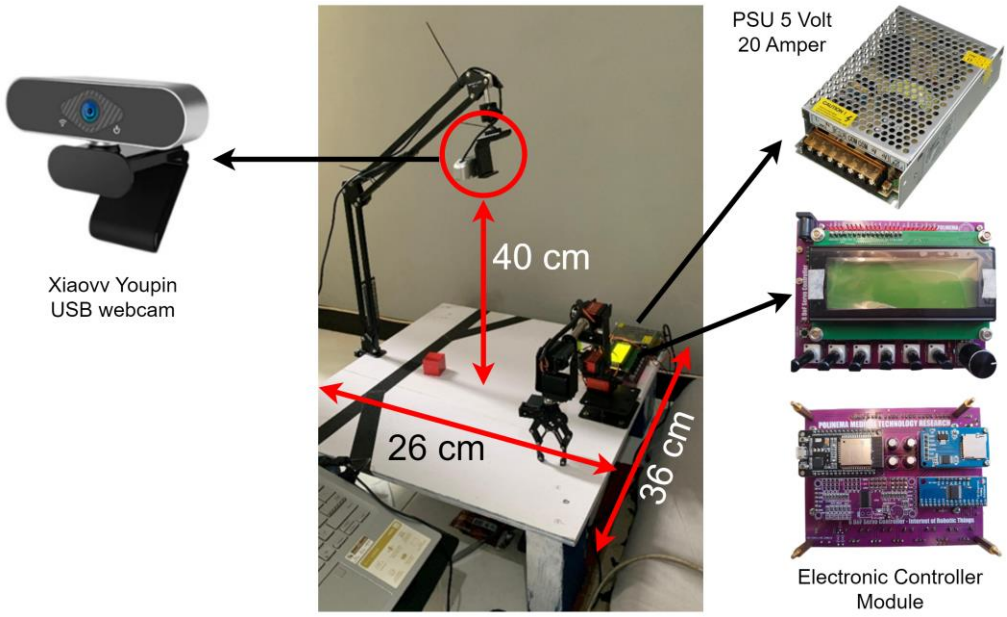


Fig.2. Camera Position, Arm-Robot Construction and Multiple Servo Motor Control Board

3) Control and Electronics: Motion control is executed on an ESP32 (dual-core, 240 MHz), interfaced with a PCA9685 I²C PWM driver that controls six servo motors. A 5V/10A power unit supplies a stable voltage under load. All components were mounted on a custom-developed control board. The detailed features of the controller module used in this study are described in our previous study, the study titled "Hybrid Multi-Servo Motor Controller Within an IoT-Enabled Smart Mechatronics Framework" [15].

a) Open-Loop Control Mechanism

The system employs an open-loop control strategy, meaning the ESP32 sends precomputed PWM signals to each servo motor without reading real-time angle feedback, as shown in Equation (1). The relationship between the desired joint angle (θ in degrees) and the PWM pulse width (in microseconds) follows the standard MG995 servo specification. where 1500 µs corresponds to the neutral position (0°). 2400 µs corresponds to +180°, 900 µs corresponds to –180°. This linear mapping is implemented in the ESP32 firmware.

$$PWM(\theta) = 1500 + \frac{\theta}{180} \times 900 \quad (1)$$

b) MQTT Subscription on ESP32

The ESP32 connects to the MQTT broker (CloudAMQP) via its onboard WiFi module and subscribes to the topic "*robot/control*" with QoS level 1. The subscription is implemented using the PubSubClient library. When the Raspberry Pi publishes a JSON message containing the target coordinates (x, y, z), the ESP32's callback function is automatically triggered. The pseudocode below outlines the subscription and callback routine. This event-driven architecture ensures that the ESP32 remains idle while waiting for new commands, minimizing power consumption and CPU overhead.

```
#include <PubSubClient.h>
void callback(char* topic, byte* payload, unsigned int length) {
  // Parse incoming JSON payload containing x, y, z coordinates
  // Example payload: {"x": 10.5, "y": 15.2, "z": 3.0}
  StaticJsonDocument<200> doc;
  deserializeJson(doc, payload, length);
  target_x = doc["x"];
  target_y = doc["y"];
  target_z = doc["z"];
  newDataFlag = true;  // Signal that new coordinates are available
}
void setup() {
  client.setServer(brokerIP, 1883);
  client.setCallback(callback);
  client.subscribe("robot/control");
}
```

c) Inverse Kinematics Computation Before Servo Movement.

Upon receiving new coordinate data (indicated by newDataFlag == true), the ESP32 executes the following sequential process before any servo motion occurs. The inverse kinematics calculation uses the geometric parameters of the 6-DOF manipulator: L1 = 8 cm (base height), L2 = 10.5 cm (upper arm), L3 = 10.5 cm (forearm), and L4 = 8 cm (wrist-to-gripper). The joint angles for the four active joints are computed using the following analytical Equation (2). Where ϕ is the approach angle (gripper orientation), typically set to 0° for vertical gripping, all angles are computed in radians and then converted to degrees. A 90° offset is added to each angle to map the range [−90°,+90°] to the servo control range [0°,180°]. Note on the 6-DOF configuration: Although the manipulator has six physical servo motors, only four joints (θ_1, θ_2, θ_3, θ_5) are actively controlled by the inverse kinematics solver. The remaining two joints (θ_4 for wrist roll and θ_6 for wrist yaw/gripper rotation) are kept stationary at 0° because the gripper is maintained in a vertically downward orientation during pick-and-place operations. These two motors do not participate in the positioning task and remain idle.

$$
\begin{aligned}
&\theta_1 = atan2(y,x)\\
&r = \sqrt{x^2+y^2} - L_4\cos(\phi)\\
&z' = z - L_1 - L_4\sin(\phi)\\
&d = \sqrt{r^2+z'^2}\\
&\phi_3 = \arccos\left(\frac{L_2^2+L_3^2-d^2}{2L_2L_3}\right)\\
&\alpha = atan2(z',r)\\
&\beta = \arccos\left(\frac{L_2^2+d^2-L_3^2}{2L_2d}\right)\\
&\theta_2 = \alpha - \beta\\
&\theta_5 = \left(\theta_2+\theta_3-\frac{\pi}{2}\right)
\end{aligned}
\qquad (2)
$$

The sequential process consists of five steps:

Step-1: Coordinate validation. Checks whether the target (*x,y,z*) lies within the robot's reachable workspace (26 × 36 cm, with fixed z=3 cm). If outside, the movement is aborted.

Step-2: IK computation. Solves for joint angles θ_1 to θ_6 using the geometric inverse kinematics formulas above. Joints θ_4 and θ_6 are set to 0° (no roll or yaw in this implementation).

Step-3:Joint limit checking. Verifies that each computed angle falls within the safe operating range: θ_1:0°−180°, θ_2:20°−160°, θ_3:0°−180°, θ_4:0°−180°, θ_5:0°−180°, θ_6:0°−180°.

Step-4:PWM conversion. Converts each valid joint angle to a PWM pulse width using the open-loop formula: PWM(θ) = 1500 + (θ / 180) × 900.

Step-5:Servo actuation. Sends all six PWM commands simultaneously via the PCA9685 driver, causing the servos to move to the computed positions.

Steps 1–4 are completed within approximately 50 ms. Only after all validations pass does the ESP32 command the servos to move. If any validation fails (e.g., unreachable coordinate or joint limit violation), the system aborts the movement and publishes an error status message via MQTT to the monitoring dashboard. This ensures that only kinematically feasible commands are executed, preventing mechanical damage or erratic behaviour.

DOI: http://dx.doi.org/10.25139/ijair.v8i1.11347

d) Complete Control Sequence

The flowchart of the complete control sequence from MQTT message reception to servo actuation, summarising the entire process described in Fig.3.

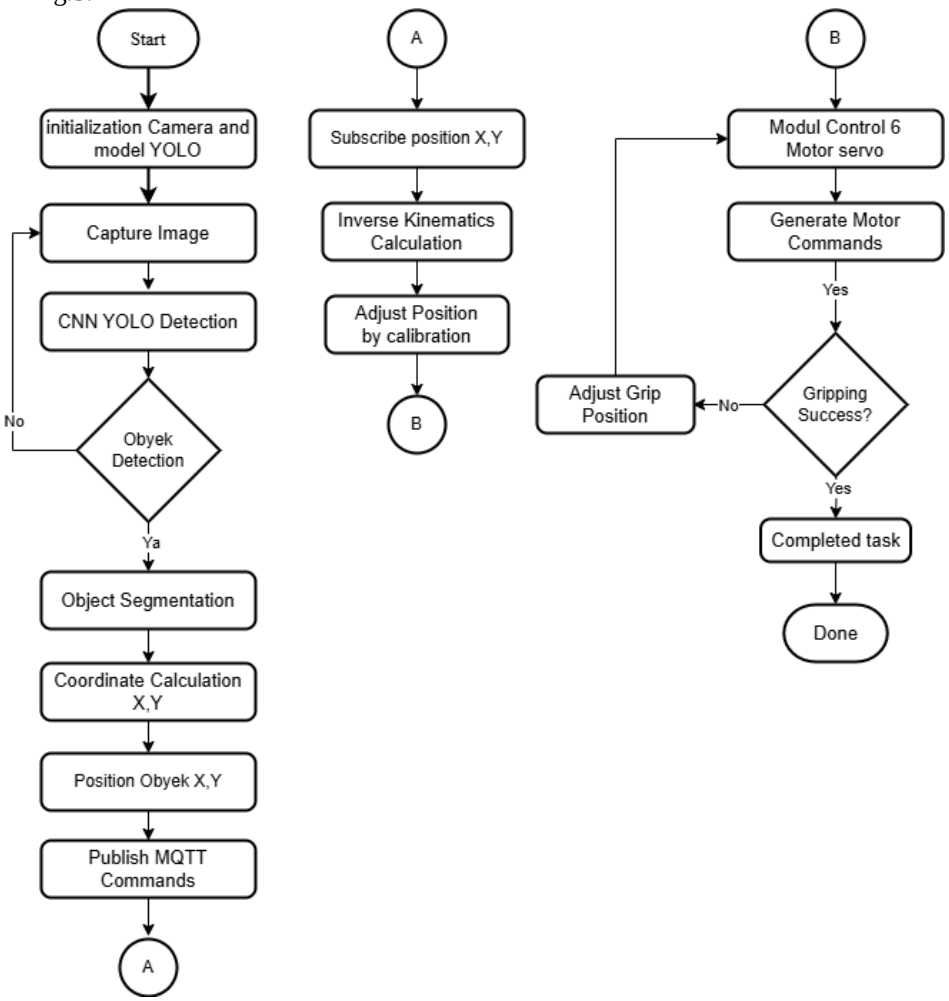
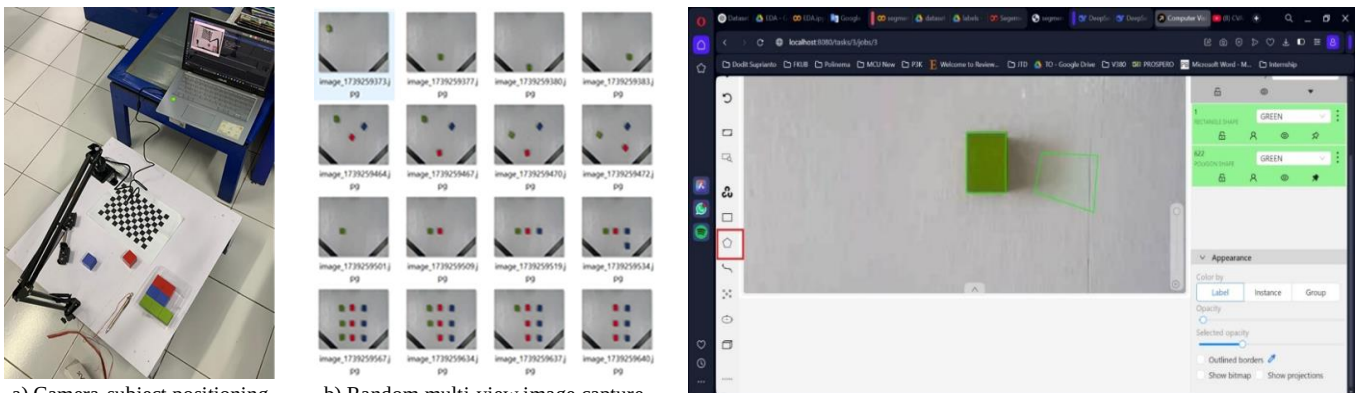


Fig.3. Flowchart of Control Sequence from MQTT Message Reception to Servo Actuation

C. *Software Module Development*

1) YOLO-based Perception Module: A vision system was implemented for real-time object detection using the YOLOv5 architecture, specifically the YOLOv5s variant. The model was trained on a custom dataset of 1,000 annotated images featuring red, green, and blue cubes (3×3×3 cm) captured in various orientations under controlled lighting conditions to ensure consistency. Dataset preparation involved systematic image acquisition using a configured webcam, followed by annotation with tight-fitting bounding boxes in the Computer Vision Annotation Tool (CVAT) (see Fig. 4). The dataset was partitioned into 70% for training, 20% for validation, and 10% for testing.



a) Camera-subject positioning setup

b) Random multi-view image capture examples

c) Object annotation process using the CVAT tool

Fig.4. Annotation Processing

The training was conducted for 100 epochs in Google Colab using GPU acceleration, with data augmentation techniques including random rotation (±15°), brightness variation (±20%), and scaling (0.8–1.2×) to enhance model robustness. Acknowledge that 1,000 images is a relatively modest dataset for deep learning. However, this study focuses on demonstrating the feasibility of an integrated vision–kinematics–IoRT system within a structured laboratory environment, rather than achieving high generalization across diverse real-world conditions. Given the low intra-class variation (uniform 3×3 cm cubes with high-contrast colours), a fixed camera position (40 cm above the ground), and controlled illumination, the applied augmentation was sufficient to mitigate overfitting for this proof-of-concept. The detection results (100% on the test set, mAP@0.5 of 0.93) confirm reliable performance under these defined operating conditions. For deployment across more varied environments, a larger, more diverse dataset would be necessary, and this remains an important direction for future work.

2) Coordinate Transformation Algorithm: The coordinate transformation algorithm was developed based on the calibrated camera parameters obtained from the hand–eye calibration procedure (see Section F.1 - Calibration and Configuration Procedure). Using the intrinsic matrix and distortion coefficients derived from calibration, a perspective-projection and distortion-correction function was implemented in Python to map the 2D pixel coordinates of a detected object's centre to real-world coordinates within the robot's base reference frame. For this study, the z-coordinate (object height) was fixed at 3 cm, corresponding to the uniform size of the cubic objects used in the experiments, thereby simplifying the 3D transformation process.

Algorithm 1: Pixel-to-Centimeter Conversion and Coordinate Retrieval

Input: Image I of size $W_p \times H_p$ pixels (here $W_p = 640$, $H_p = 480$);
Real-world field size $W_{cm} \times H_{cm}$ centimeters (here $W_{cm} = 80$, $H_{cm} = 50$);
Pixel coordinate (u, v) with origin at the **top-left** of I;
(Optional) boolean `cartesian_origin` to switch origin to **bottom-left**.
Output: Real-world coordinate (x_{cm}, y_{cm}) in centimeters.

Step 1: Pre-compute scales

$s_x \leftarrow \frac{W_{cm}}{W_p}$ `// cm per pixel in x`

$s_y \leftarrow \frac{H_{cm}}{H_p}$ `// cm per pixel in y`

`// For this setup:` $s_x = 80/640 = 0.125$, $s_y = 50/480 \approx 0.1041667$

Step 2: Validate input pixel

if $u < 0$ ***or*** $u \geq W_p$ ***or*** $v < 0$ ***or*** $v \geq H_p$ **then**
  raise "Pixel out of bounds"

Step 3: Convert pixel (u, v) **to centimeters**

$x_{cm} \leftarrow u \cdot s_x$

$y_{cm} \leftarrow v \cdot s_y$ `// Top-left origin: y increases downward`

Step 4 (Optional): Use Cartesian origin at bottom-left

if *cartesian_origin is* ***true*** **then**
  $y_{cm} \leftarrow H_{cm} - y_{cm}$ `// Flip y upward`

Step 5 (Optional): Round precision

$x_{cm} \leftarrow \text{round}(x_{cm}, 3)$; $y_{cm} \leftarrow \text{round}(y_{cm}, 3)$

return (x_{cm}, y_{cm}).

Algorithm 2: Get Cube/Box Center from YOLO Segmentation and Convert to cm

Input: YOLO segmentation output for the object: either
  (a) binary mask M (same size as I), or
  (b) bounding box $\mathbf{b} = (u_{\min}, v_{\min}, u_{\max}, v_{\max})$;
(Optional) depth map D aligned with I;
Image/field dimensions as in Algorithm 1.
Output: Real-world coordinate of the cube/box center (x_{cm}, y_{cm}); optionally z_{cm}.

B1: Derive pixel-space center (u_c, v_c)

if *bounding box* $\mathbf{b}$ *is available* **then**
  $u_c \leftarrow \frac{u_{\min} + u_{\max}}{2}$; $v_c \leftarrow \frac{v_{\min} + v_{\max}}{2}$ `// Box center (cube face center in image)`
else
  `// Fallback: centroid from mask (robust to irregular shapes)`
  $A \leftarrow \sum_{i,j} M[i,j]$ `// area in pixels`
  if $A = 0$ **then**
    raise "No object"
  $u_c \leftarrow \frac{1}{A}\sum_{i,j} j \cdot M[i,j]$; $v_c \leftarrow \frac{1}{A}\sum_{i,j} i \cdot M[i,j]$ `// (row i, col j)`

B2: Convert pixel center to centimeters

$(x_{cm}, y_{cm}) \leftarrow$ Pixel-to-Centimeter Conversion and Coordinate Retrieval$(u_c, v$ `// calls Alg. 2`

B3 (Optional): Estimate z **if depth is available**

if D *is provided* **then**
  `// Use mean depth inside the box/mask; convert to cm via calibration`
  if *bounding box is used* **then**
    $z_{\text{px}} \leftarrow \text{mean}\{D[v,u] \mid u \in [u_{\min}, u_{\max}],\ v \in [v_{\min}, v_{\max}]\}$
  else
    $z_{\text{px}} \leftarrow \frac{1}{A}\sum_{i,j} D[i,j] \cdot M[i,j]$
  $s_z \leftarrow$ depth_scale_to_cm `// from camera calibration`
  $z_{cm} \leftarrow s_z \cdot z_{\text{px}}$
  return (x_{cm}, y_{cm}, z_{cm})
else
  return (x_{cm}, y_{cm})

The transformation procedure relies on calibrated optical parameters and a pixel-to-centimetre scaling model derived from the known workspace dimensions. The overall coordinate conversion pipeline consists of two main computational stages: (1) pixel-to-centimetre scaling, and (2) object centre extraction and coordinate retrieval from YOLO outputs. The detailed pseudocode for both procedures is presented in Algorithm 1 and Algorithm 2, which together ensure accurate and consistent mapping between image-space detections and the robot's real-world coordinate system.

3) Inverse Kinematics Solver: The kinematic control implementation is centred on a geometric inverse kinematics (IK) solver for the 6-DOF manipulator. This analytical solver, implemented in C++ on the ESP32, computes joint angles θ_1–θ_6 using the Denavit–Hartenberg convention and parameters derived from the robot's physical geometry [16], with measured link lengths L_1 =

8 cm, $L_2 = L_3 = 10.5$ cm, and $L_4 = 8$ *cm*. The algorithm incorporates singularity-avoidance mechanisms and joint limit checks to ensure stable, physically realizable arm movements. Open-loop control was deliberately chosen to prioritize cost-effectiveness and design simplicity. This approach eliminates the need for joint feedback sensors (e.g., encoders or potentiometers), significantly reducing component cost and system complexity. Consequently, the platform becomes more accessible for educational robotics and light industrial automation. However, open-loop operation introduces positioning uncertainties due to servo dead zones, gear backlash, and torque-dependent deflection. To compensate for the absence of real-time feedback, we implemented three complementary measures. First, meticulous servo calibration was performed using a linear pulse-width-to-angle correction model derived from digital protractor measurements at multiple angles (0°, 45°, 90°). Second, repeatability tests were conducted across 50 repetitions to characterize positioning variance. Third, forward kinematics validation was applied to verify the numerical stability of each IK solution before execution. These procedures collectively ensured that positioning errors remained within acceptable bounds (mean 4.3 mm, ±0.7 mm standard deviation). Finally, the IK solution's numerical stability and workspace accuracy were verified through forward kinematics.

D. IoRT Communication Implementation

Communication among system modules is managed through the MQTT protocol using the Paho client library. In this configuration, the Raspberry Pi functions as the publisher, transmitting detected object coordinates, while the ESP32 operates as the subscriber, receiving target position data for actuation. Three primary communication topics are defined—"robot/control", "robot/status", and "robot/telemetry"—with Quality of Service (QoS) level 1 applied to ensure reliable command delivery. The IoRT communication framework incorporates automatic reconnection and offline message buffering mechanisms to maintain system resilience under network fluctuations.

To provide a comprehensive overview of the integrated operational process, the overall control logic—encompassing visual perception, coordinate transformation, inverse kinematics, and IoRT communication—is summarised in Algorithm 3. The pseudocode describes the sequential data flow and interdependencies between the vision, motion planning, and servo control modules. It demonstrates how object detection outputs are transmitted via MQTT to initiate inverse kinematics computation, followed by the 6-DOF robotic manipulator's corresponding actuation. The complete step-by-step workflow is presented in Algorithm 3.

Algorithm 3: YOLO-based Object Detection and Robotic Arm Control via MQTT

```
Input  : Camera stream, YOLOv5 model, MQTT broker, 6-DOF
         manipulator
Output: Successful grasp of detected object
Vision Module:
Initialize camera and load YOLO model;
while image captured do
    Perform YOLO detection;
    if object detected then
        Compute object coordinates (X, Y);
        Publish coordinates via MQTT;
        ;
    end
end
Motion Planning:
Subscribe to (X, Y) from MQTT;
Perform inverse kinematics and calibration;
Servo Control:;
while not grasped do
    Send servo commands to target position;
    if gripping successful then
        End task;
        return Success;
    end
    else
        Adjust position and retry;
    end
end
```

E. System Integration Protocol

1) Data Flow Integration: The modules were integrated into a sequential pipeline. The output (object ID, 2D bounding box, confidence) from the YOLO module (running on a Raspberry Pi) was published via an MQTT broker on the internet. The ESP32 motor controller, acting as a subscriber, received this data. Subsequently, the output (*x, y, z*) was processed by the Inverse

Kinematics (IK) solver on the ESP32, which computed the joint angles and generated corresponding PWM signals via the PCA9685 driver. Gripper commands were integrated into this sequence. All status data was also published via MQTT to a central dashboard.

2) Temporal Coordination Strategy: To ensure real-time operation, the system modules were coordinated sequentially. The vision processing, coordinate conversion, and inverse kinematics computation were optimized to execute within a timeframe that supported a complete pick-and-place cycle of approximately 1-2 seconds, as observed during system testing.

3) Error Handling Implementation: The system implemented error-handling mechanisms at key stages to improve robustness. The vision system utilized confidence thresholds to validate detections, the IK algorithm included checks for unreachable positions, and the communication protocol was designed to maintain system stability despite potential network interruptions.

F. Calibration and Configuration Procedure

1) Camera-Robot Calibration: Hand-eye calibration was performed using Zhang's method [17]. A checkerboard pattern (see Fig. 5) was placed at multiple known locations within the robot's workspace. For each location, the robot's end-effector position was recorded, and a corresponding image was captured. Using OpenCV's cv.calibrateCamera and cv.calibrateHandEye functions, the homogeneous transformation matrix relating the camera frame to the robot base frame was computed.

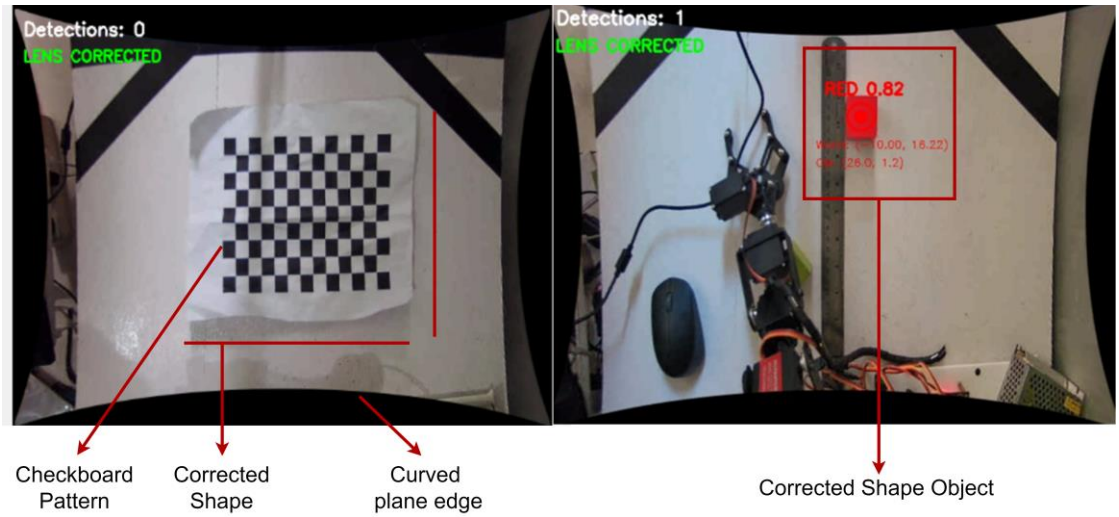


Fig.5. Camera Lens Calibration Is Performed Using A Checkerboard Pattern

2) Workspace Mapping and Servo Calibration: Workspace mapping was done by placing a calibration marker at known (x, y) grid points, detecting its pixel location, and fitting a polynomial mapping function. For servo calibration, each joint was commanded to move to multiple angles (e.g., 0°, 45°, 90°), and the actual angle was measured using a digital protractor. A linear correction model (pulse-width vs angle) was derived for each servo to improve open-loop positioning accuracy.

G. Experimental Design and Evaluation

This section outlines the experimental setup and evaluation procedures used to validate the performance of the proposed integrated vision–kinematics control system. The experiments were designed to quantitatively assess the system's visual detection accuracy, kinematic precision, manipulation success rate, and IoRT communication reliability under real operating conditions. All experiments were conducted using the fully assembled prototype described in F. Calibration and Configuration Procedure, operating within a calibrated workspace of 26 × 36 cm (see Fig. 2).

1) Performance Metrics: The system was evaluated using the following quantitative metrics: (1) Detection Performance: Precision, Recall, mAP@0.5, and average confidence score. (2) Kinematic Accuracy: Euclidean positioning error (mm) between commanded and achieved end-effector position. (3) System Performance: Gripping success rate (%) and total operation cycle time (s). (4) IoRT Performance: Message delivery success rate (%) and average latency (ms).

2) Testing Scenarios: Three main experimental scenarios were designed: (1) Vision Module Test: 300 static images (900 object instances) were used for offline evaluation. (2) Kinematic Accuracy Test: The robot was commanded to move to 50 predefined target points within its workspace, and the final position was measured. (3) Integrated System Test: 100 full autonomous pick-and-place trials were conducted with objects randomly placed within the workspace.

3) Data Collection and Analysis Plan: All experimental data, including detection results, positional errors, timestamps, and MQTT communication metrics, were automatically logged through the IoRT monitoring dashboard. The recorded datasets were processed using Python for descriptive analysis.

DOI: http://dx.doi.org/10.25139/ijair.v8i1.11347

For each experimental scenario, key performance indicators, including precision, recall, mAP, positional error, gripping success rate, and message latency, were computed and summarised as mean ± standard deviation. Trends and variations across object colour classes were examined descriptively to identify consistent behavioural patterns. The analysis emphasized system repeatability and stability across repeated trials rather than inferential statistical significance.

H. Interface Design

The human-machine interface provides comprehensive monitoring and control capabilities through multiple integrated components. A real-time video feed displays live camera imagery, annotated with object detection results and corresponding confidence scores, enabling visual verification of the system's perception. The system status panel presents essential operational information, including connection status indicators, the current operating mode, and error notifications, for immediate situational awareness. The control interface provides manual override, parameter adjustment, and emergency stop controls for operator intervention when necessary. Comprehensive data logging capabilities capture performance metrics, maintain detailed operation histories, and record error events for subsequent analysis and system improvement (see Fig. 6).

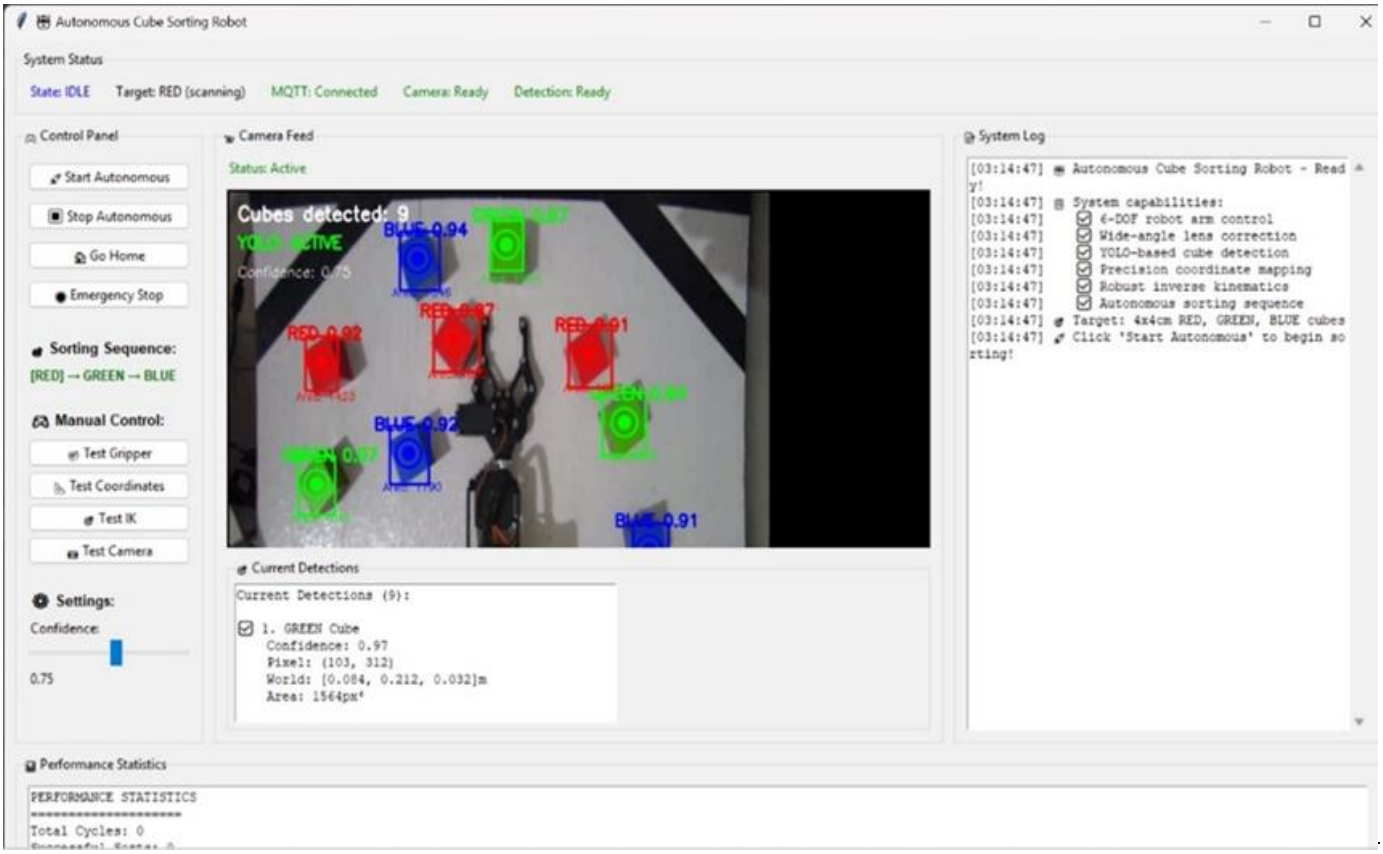


Fig.6. GUI Application Human-Machine Interface

IV. RESULTS AND DISCUSSION

This section presents the quantitative results and qualitative analysis of the proposed integrated vision, kinematics control system. Each subsystem, including vision, kinematic control, IoRT communication, and multi-object handling, was evaluated individually and in integrated operation. Results are analyzed for accuracy, repeatability, and system scalability, followed by interpretations of their implications within the open-loop, low-cost robotic framework.

A. Vision System Performance

The YOLOv5s-based perception module achieved high detection accuracy and stability under controlled conditions. The dataset was split stratified, ensuring that each colour class (red, green, blue) maintained its proportional representation across the training (70%), validation (20%), and test (10%) sets. Quantitative evaluation used 300 test images containing 900 object instances (300 per colour class), all captured under defined operating conditions: a fixed camera height (40 cm), uniform workspace illumination (~300 lux), and high-contrast-coloured cubes against a plain background.

Within this test set, the system achieved a 100% detection rate, with no false positives or false negatives. Table II summarises the full set of detection performance metrics, including per-class true positives, false positives, and false negatives.

TABLE II.
VISION MODULE DETECTION PERFORMANCE SUMMARY

Metric	Red	Green	Blue	Mean	Interpretation
Precision	0.94	0.96	0.89	0.93	High detection precision across classes
Recall	0.91	0.93	0.9	0.91	Stable detection sensitivity
mAP@0.5	0.93	0.93	0.9	0.93	Excellent localization performance
Avg. Confidence	0.9	0.94	0.91	0.92	Slightly higher confidence for green objects
True Positives	300	300	300	900	Perfect classification on the test set
False Positives	0	0	0	0	No false alarms
False Negatives	0	0	0	0	No missed detections
Inference Time (ms)	150 ± 12	–	–	–	Real-time inference achieved

Note: Test set consisted of 300 images (900 object instances). The 100% detection rate reflects controlled conditions (fixed camera height, uniform illumination, high-contrast objects).

This perfect classification, while notable, reflects the structured nature of our experimental setup. Outside these controlled conditions, occasional detection failures were observed in edge cases, including: (1) extreme object occlusion (>50% coverage), (2) objects placed outside the calibrated 26×36 cm workspace, or (3) significant deviation from the fixed 40 cm camera height. Such scenarios were excluded from the quantitative evaluation to maintain consistency with the system's intended operating envelope.

The model achieved a mean average precision (mAP@0.5) of 0.93, confirming robust localization capability. Per-class precision values were 0.94 for red, 0.96 for green, and 0.89 for blue cubes. The average detection confidence was 0.92, with the highest observed for green cubes (0.94) and the lowest for blue cubes (0.91). This variation is attributed to differences in colour contrast and reflectivity under the illumination setup. The inference latency averaged 150 ± 12 ms, which supports real-time feedback for continuous manipulation.

Fig. 7 illustrates the precision and recall distribution for red, green, and blue objects. Green objects achieved the highest precision (0.96) and recall (0.93), while blue objects exhibited slightly lower scores due to their lower visual contrast against the workspace background.

The results confirm that the trained YOLOv5s model performs reliably within the defined workspace and lighting conditions, achieving detection performance comparable to that of more computationally intensive architectures while maintaining real-time operation. Slight differences in confidence across colours indicate that lighting uniformity remains an important factor in ensuring consistent perception reliability.

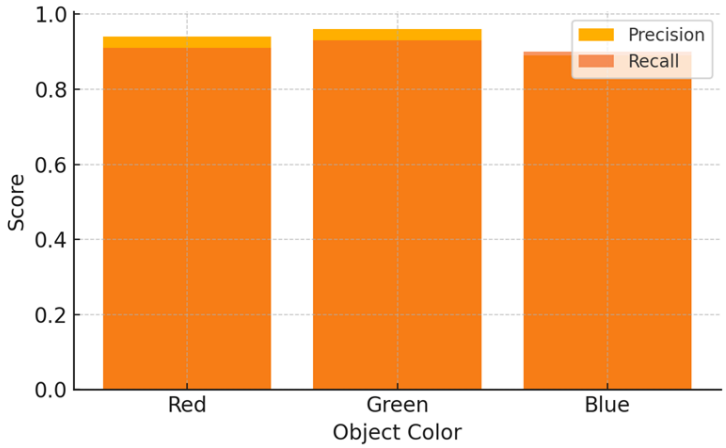


Fig.7. YOLOv5s Detection Performance by Object Colour, Showing Precision and Recall Trends Across Three Colour Categories

B. Kinematic Positioning Accuracy

Commanding movements evaluated the 6-DOF manipulator to 50 predefined target coordinates distributed across its 26 × 36 cm workspace. The mean Euclidean positioning error between commanded and achieved positions was 4.3 mm, with a standard deviation of ±0.7 mm. The highest errors occurred near the workspace boundaries due to cumulative joint tolerance and singular configurations.

Given the open-loop actuation with MG995 servos and no joint feedback, the achieved accuracy demonstrates effective mechanical calibration and numerical stability of the geometric inverse kinematics algorithm. The sub-centimetre precision indicates that the system's open-loop design can still support practical pick-and-place operations for small objects. These findings align with the performance range of low-cost educational and research manipulators reported in similar studies (see Table III).

TABLE III

KINEMATIC POSITIONING ACCURACY

Test Point	Commanded (x, y, z) [cm]	Measured (x, y, z) [cm]	Error [mm]	Remarks
P1	(10, 5, 3)	(10.4, 5.3, 3.2)	4.8	Central region
P2	(5, 10, 3)	(5.3, 10.2, 3.0)	3.6	Stable response
P3	(15, 15, 3)	(15.6, 15.5, 3.1)	7.2	Near boundary
P4	(20, 10, 3)	(20.3, 10.5, 3.2)	5.4	High torque demand
Mean ± SD	–	–	4.3 ± 0.7	Overall accuracy

C. Autonomous Gripping Performance

The integrated vision–motion system was tested in over 100 autonomous pick-and-place trials, with equal numbers for red, green, and blue cubes. Table IV summarises the gripping performance for each colour class.

TABLE IV
AUTONOMOUS GRIPPING PERFORMANCE

Object Color	Trials	Success	Failure	Success Rate (%)	Main Failure Mode
Red	33	26	7	78.8	Slip, weak grip
Green	33	30	3	90.9	Stable
Blue	34	24	10	70.6	Detection instability
Total / Mean	100	80	20	80	Grip + positioning

The grasping success rate, detailed in Table IV, reflects the visual perception trends observed in the detection phase. As shown in Fig. 8a, green objects exhibited the highest manipulation success rate (90.9%), consistent with their superior detection confidence. In comparison, blue objects showed the lowest success rate (70.6%) due to their reflective surfaces and lower edge definition.

1) Statistical Analysis: To determine whether the observed differences in success rates across colours were statistically significant, a chi-square test was performed. Table V summarises the pairwise comparison results. The difference between green and blue was statistically significant ($\chi^2 = 4.42$, $df = 1$, $p = 0.035$), indicating that green objects were grasped significantly more reliably than blue objects under our experimental conditions. No significant differences were found between green and red ($p = 0.169$) or between red and blue ($p = 0.431$). The 95% confidence intervals (Wilson score method) for each colour. Green: 90.9% (*CI: 75.5% – 97.1%*); Red: 78.8% (*CI: 62.2% – 89.3%*); Blue: 70.6% (*CI: 53.8% – 83.2%*). These intervals show overlap between red and blue, while green's lower bound (75.5%) exceeds blue's upper bound (83.2%), further confirming that green objects were grasped more reliably than blue objects.

TABLE V
PAIRWISE CHI-SQUARE TEST RESULTS FOR GRIPPING SUCCESS RATES

Comparison	χ^2 (df = 1)	*p*-value	Significant (α=0.05)
Green (90.9%) vs Blue (70.6%)	4.42	0.035	Yes
Green (90.9%) vs Red (78.8%)	1.89	0.169	No
Red (78.8%) vs Blue (70.6%)	0.62	0.431	No

2) Cycle Time and Failure Analysis: The observed variation in success rates across colour classes warrants detailed analysis. Blue objects exhibited the lowest performance due to a combination of factors: (1) lower visual contrast under the illumination setup, (2) higher surface reflectivity causing detection instability, and (3) potential colour channel saturation in the camera sensor. Red objects showed intermediate performance, with failures primarily attributed to grip slippage rather than detection issues. Green objects performed optimally due to their high contrast against the workspace background and consistent reflectance properties. The average cycle time for a successful pick-and-place operation was 9.1 ± 1.4 seconds, which includes vision processing (0.15 s), coordinate transformation (0.05 s), inverse kinematics computation (0.05 s), servo motion (1.0 s), and gripper actuation (0.5 s), with remaining time attributed to synchronization and communication overhead. These findings highlight the importance of environmental optimization for vision-based systems. Future implementations could benefit from (1) adaptive lighting control, (2) multi-spectral or depth-enhanced sensing, and (3) material-specific gripper designs. Notably, despite these colour-based variations, the system maintained an overall 80% success rate, demonstrating robustness within the constraints of a low-cost, open-loop architecture.

D. IoRT Communication Performance

The IoRT layer, implemented via MQTT, maintained stable communication throughout all trials. Under local network operation, the message delivery success rate reached 99.6% with an average latency of 52 ms. When connected to the cloud broker, reliability slightly decreased to 98.7%, with an average latency of 118 ms due to network propagation delays.

DOI: http://dx.doi.org/10.25139/ijair.v8i1.11347

The high reliability and low-latency characteristics validate the robustness of the MQTT-based architecture for real-time control and monitoring. The slight difference between local and cloud operation demonstrates scalability, confirming that IoRT frameworks can be effectively applied to embedded robotic systems without compromising real-time performance (see Table VI).

TABLE VI
IORT COMMUNICATION METRICS

Mode	Message Success (%)	Avg. Latency (ms)	Notes
Local Network	99.6	52	Stable real-time communication
Cloud MQTT Broker	98.7	118	Minor propagation delay
Offline Reconnect	97.8	130	Auto-recovery verified

E. Multi-Object Handling Capability

The system's ability to handle multiple objects simultaneously was tested using scenarios with 3 to 9 cubes randomly placed in the workspace. The detection rate remained at 100% across all configurations, with negligible degradation in localization accuracy. The average cycle time increased marginally (less than 5%) with object density, showing minimal effect on overall throughput.

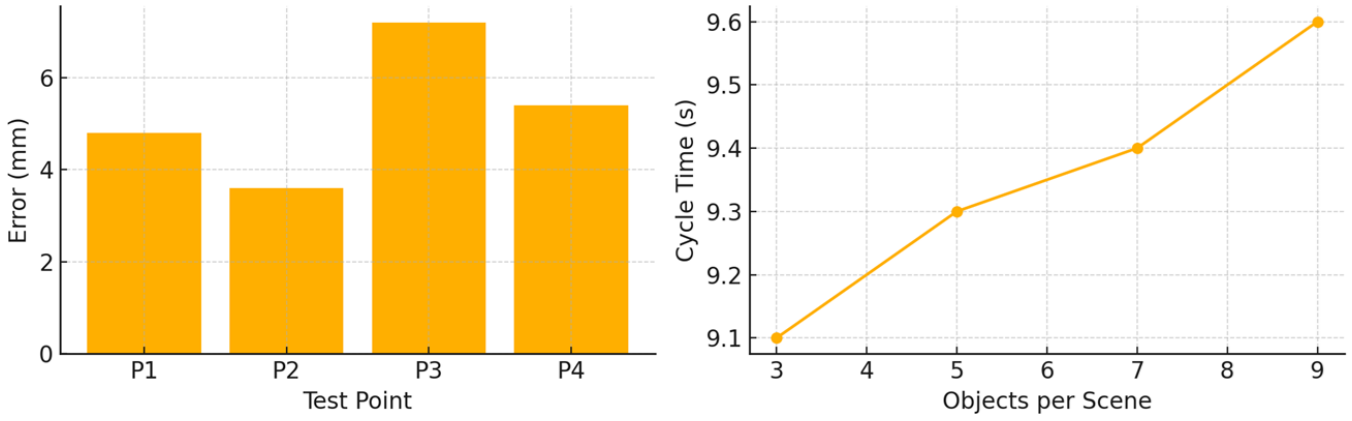
These results confirm that the integrated vision–control framework scales effectively to more complex environments. The ability to maintain consistent performance across varying object densities demonstrates modular coordination between the perception and planning layers, supporting potential applications in multi-target industrial or warehouse scenarios (see Table VII).

TABLE VII
MULTI-OBJECT HANDLING SCALABILITY

Objects per Scene	Detection Rate (%)	Avg. Cycle Time (s)	Observation
3	100	9.1	Stable
5	100	9.3	Slight delay due to an extra target
7	100	9.4	Consistent throughput
9	100	9.6	Minimal degradation
Mean ± SD	100 ± 0	9.35 ± 0.2	Robust scalability

Together, these two graphs provide complementary insights into the system's mechanical and computational robustness. Fig.8a highlights the physical performance limit and the spatial variation in precision due to servo mechanics and arm geometry. Fig.8b demonstrates the system's algorithmic efficiency, with processing time remaining stable even as task complexity increases.

This dual analysis confirms that the proposed integrated system achieves a balanced trade-off between mechanical precision and computational scalability. The design leverages accurate vision-based feedback and efficient IoRT communication to mitigate the inherent uncertainty of open-loop servo control, thereby enabling reliable, repeatable autonomous manipulation on a constrained, low-cost robotic platform.



(a) Kinematic positioning error across test points (b) System cycle time as a function of object count

Fig.8. The Mechanical and Computational Robustness System

F. Performance Consistency and Reliability

To evaluate system repeatability, the experiments were repeated under identical conditions. Variations in cycle time, positioning error, and detection confidence remained within ±5% of the mean across trials, confirming high operational stability. This consistency verifies the reliability of the proposed system in repeated autonomous cycles. The results also reinforce that an open-loop control approach, when properly calibrated, can achieve repeatable behaviour sufficient for structured automation tasks, validating its suitability for cost-effective robotic implementations.

DOI: http://dx.doi.org/10.25139/ijair.v8i1.11347

G. Comparative and Contextual Discussion

Compared with other low-cost manipulator systems in the literature, which often separate vision and control modules or rely on manual calibration, the proposed system integrates perception, kinematics, and IoRT communication within a single embedded framework. For instance, while prior YOLO-based robotic systems achieved high detection accuracy but lacked end-to-end integration, this work demonstrates both high detection fidelity (mAP 0.93) and full autonomous actuation.

The novelty of our approach lies not in advancing any single component beyond prior art, but in the successful integration of vision, kinematics, and IoRT communication into a low-cost, open-loop robotic platform. While previous studies have demonstrated high-performance vision [5], [6] or kinematics [7] in isolation, or IoRT frameworks without manipulation integration [8], our system combines all three subsystems to achieve end-to-end autonomous pick-and-place operations.

Although the achieved precision (4.3 mm) and success rate (80%) are lower than those reported by high-end feedback-based manipulators (typically <1 mm error), this system emphasizes integration feasibility and cost efficiency. It bridges the gap between high-cost industrial systems and low-cost academic prototypes, representing a viable model for IoRT-enabled, vision-guided robotic platforms.

H. Summary of Findings

The experimental results confirm the effective integration of computer vision, kinematics, and IoRT communication within a low-cost robotic architecture. The system consistently achieved accurate object detection, reliable manipulation, and stable communication performance (see Table VIII). Its repeatability across multiple trials highlights the feasibility of deploying similar embedded IoRT frameworks for scalable, real-time, autonomous manipulation.

TABLE VIII
OVERALL SYSTEM SUMMARY

Metric	Value	Remarks
Detection Accuracy	100%	Robust perception
mAP@0.5	0.93	Reliable localization
Mean Positioning Error	4.3 mm	Stable open-loop control
Gripping Success Rate	80%	Reliable operation
MQTT Message Success	99.6% (local)	High communication reliability
Avg. Cycle Time	9.1 ± 1.4 s	Real-time operation
Repeatability (All Metrics)	±5%	High stability

V. CONCLUSION

This research presented the design and implementation of an integrated vision-kinematics control system for autonomous robotic manipulation that combines deep learning-based perception, geometric inverse kinematics computation, and IoRT-enabled communication into a cohesive embedded platform. The system was developed using low-cost components and open-loop actuation, yet it demonstrated performance stability comparable to more complex feedback-based robotic architectures. Experimental validation confirmed that the integrated system performs effectively under realistic operational conditions, with the vision subsystem achieving a mean average precision of 0.93, the kinematic module maintaining 4.3 mm positional accuracy, and the overall autonomous grasping success rate reaching 80% across one hundred trials.

While the system demonstrates promising performance, several limitations warrant acknowledgement. The open-loop control architecture, though cost-effective, limits positioning accuracy compared to encoder-feedback systems. Reliance on controlled lighting conditions and uniform object colours limits environmental adaptability. Additionally, the fixed-height assumption simplifies coordinate transformation but limits handling of varied object geometries.

Future work will focus on three key enhancements, presented in order of priority. The highest priority is integrating incremental encoders or potentiometers for closed-loop joint control to improve positioning accuracy. The second priority is the implementation of RGB-D camera sensing for full 3D pose estimation and handling of varied object shapes and sizes. The third priority is the development of adaptive grip control algorithms using force-sensitive resistors to prevent object slippage. Furthermore, we plan to explore edge-AI deployment to enable real-time model adaptation to changing environmental conditions, advancing the system toward a fully intelligent, connected, and collaborative robotic platform capable of operating autonomously in dynamic, unstructured environments.

ACKNOWLEDGMENTS

The authors express their gratitude to P3M Politeknik Negeri Malang for funding support through the DIPA Budget for Fiscal Year 2025, under Number: SP DIPA-139.03.2.693474/2025, which enabled this research to be successfully conducted.

REFERENCES

[1] P. K. Mathavan Jeyabalan, A. Nehrujee, S. Elias, M. Magesh Kumar, S. Sujatha, and S. Balasubramanian, "Design and Characterisation of a Self-Aligning End-Effector Robot for Single-Joint Arm Movement Rehabilitation," *Robotics*, vol. 12, no. 6, 2023, doi: 10.3390/robotics12060149.

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-Decem, pp. 779–788, 2016, doi: 10.1109/CVPR.2016.91.

[3] M. A. Muktadir, S. Yi, and A. M. Elliott, "Design of robot grippers for binder jet products handling," *Sci. Rep.*, vol. 14, no. 1, pp. 1–14, 2024, doi: 10.1038/s41598-024-56385-8.

[4] J. Becedas, I. Payo, and V. Feliu, "Two-flexible-fingers gripper force feedback control system for its application as end effector on a 6-DOF manipulator," *IEEE Trans. Robot.*, vol. 27, no. 3, pp. 599–615, 2011, doi: 10.1109/TRO.2011.2132850.

[5] Y. Zhang and L. Zhang, "Research on Education Robot Control System Based on ESP32," *J. Educ. Educ. Res.*, vol. 7, no. 2, pp. 299–302, 2024, doi: 10.54097/3x86qp78.

[6] H. Z. Khaleel and A. J. Humaidi, "Towards accuracy improvement in solution of inverse kinematic problem in redundant robot: A comparative analysis," *Int. Rev. Appl. Sci. Eng.*, vol. 15, no. 2, pp. 242–251, 2024, doi: 10.1556/1848.2023.00722.

[7] M. Blatnický, J. Dižo, J. Gerlici, M. Sága, T. Lack, and E. Kuba, "Design of a robotic manipulator for handling products of automotive industry," *Int. J. Adv. Robot. Syst.*, vol. 17, no. 1, pp. 1–11, 2020, doi: 10.1177/1729881420906290.

[8] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," 2020, [Online]. Available: http://arxiv.org/abs/2004.10934

[9] C. Y. Wang, A. Bochkovskiy, and H. Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 13024–13033, 2021, doi: 10.1109/CVPR46437.2021.01283.

[10] V. Kumar, Q. Wang, W. Minghua, S. Rizwan, S. M. Shaikh, and X. Liu, "Computer vision based object grasping 6DoF robotic arm using picamera," *Proc. - 2018 4th Int. Conf. Control. Autom. Robot. ICCAR 2018*, no. March, pp. 111–115, 2018, doi: 10.1109/ICCAR.2018.8384653.

[11] Y. Bai, M. Luo, and F. Pang, "An algorithm for solving robot inverse kinematics based on foa optimized bp neural network," *Appl. Sci.*, vol. 11, no. 15, 2021, doi: 10.3390/app11157129.

[12] P. P. Ray, "Internet of Robotic Things: Concept, Technologies, and Challenges," *IEEE Access*, vol. 4, pp. 9489–9500, 2016, doi: 10.1109/ACCESS.2017.2647747.

[13] N. Naik, "Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP," *2017 IEEE Int. Symp. Syst. Eng. ISSE 2017 - Proc.*, 2017, doi: 10.1109/SysEng.2017.8088251.

[14] J. Wan, S. Tang, H. Yan, D. Li, S. Wang, and A. V. Vasilakos, "Cloud robotics: Current status and open issues," *IEEE Access*, vol. 4, pp. 2797–2807, 2016, doi: 10.1109/ACCESS.2016.2574979.

[15] D. Suprianto, G. S. Adi, R. Agustina, N. Hidayati, and A. M. Imammuddin, "Hybrid Multi-Servo Motor Controller Within an IoT-Enabled Smart Mechatronics Framework," vol. 10, no. 2, pp. 121–128, 2025, doi: 10.25139/inform.v10i2.10100.

[16] M. U. Atique, "Inverse Kinematics Solution for a 3DOF Robotic Structure using Denavit-Hartenberg Convention," pp. 2–6.

[17] Q. Bi, Z. Liu, M. Wang, and M. Lai, "An automatic camera calibration method based on checkerboard," pp. 209–226, 2017, doi: 10.3166/TS.34.209-226.