

Forward and optimised IK utilising the Levenberg-Marquardt algorithm 6-DOF robot manipulator for sorting applications

Muhammad Hafidz Hasnor Faiz¹, Norashikin M. Thamrin², Megat Syahirul Amin Megat Ali², Idris Zainal Abidin³

¹School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA, Shah Alam, Malaysia

²Microwave Research Institute, Universiti Teknologi MARA, Shah Alam, Malaysia

³Malaysia e-Commerce, Cytron Technologies Sdn Bhd, Bukit Mertajam, Malaysia

Article Info

Article history:

Received May 21, 2024

Revised Oct 17, 2024

Accepted Nov 19, 2024

Keywords:

Edge detection

Forward kinematics

Image processing

Inverse kinematics

Levenberg-Marquardt

Robot manipulator

Robot operating system

ABSTRACT

Colour sorting robots automate industries, but translating image data to robot movement is expensive and complicated. Vision sensors require a lot of processing power, which can slow down and strain the robot. Real-time colour sorting hardware and software integration complicates things. This work uses robotic operating system (ROS) to solve vision-guided colour sorting problems in Cartesian space. Ubuntu 20.04, ROS Noetic, a Raspberry Pi, a camera, and six servos. In Jupyter Lab, unified robotic description format (URDF) is used to build a virtual kinematic model, and Levenberg-Marquardt (LM) optimisation guides object manipulation. OpenCV image processing uses colour conversion, Canny edges, and midpoint estimation to detect coloured objects efficiently. The average servo movement error is 0.46 degrees, and the robot manipulator's final destination positioning error is 1.65 mm. The average object edge detection error is 0.33 mm, and the red, green, blue (RGB) colour distance is 57.84. ROS-based robot manipulator achieves impressive Cartesian space colour sorting accuracy despite image processing challenges, enabling real-world deployment.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Norashikin M. Thamrin

Microwave Research Institute, Universiti Teknologi MARA

Shah Alam, Selangor, Malaysia

Email: norashikin@uitm.edu.my

1. INTRODUCTION

Malaysia is improving its industry via robotics and automation. But still behind Thailand and Vietnam [1]. The government promotes advanced consumer product production using robotics [2]. The flexible, multitasking robots aid in sorting chores [3], especially in recycling centres [4]. Traditional sorting robot manipulators have fixed item locations, joint angles, and sequencing [5]. Due to difficulty recognising and managing things, manipulators without vision are less productive [6]. Manipulators can also use machine vision to assess object position, colour, and size. Operations are more efficient, and worker costs are lower [7]-[9]. Robotic image processing demands instantaneous visual decision-making [10]. Thus, smooth functioning demands powerful hardware and efficient image-processing techniques [11]. OpenCV is essential to robotic image processing, offering picture filtering, feature extraction, object detection, tracking, and posture estimation [12]-[15]. It operates effortlessly on Windows, Linux, and MacOS for robot deployment at diverse locations. It runs complicated robots on resource-constrained embedded devices and powerful desktops [16], [17]. Colour is essential for image processing and object detection. Red, green, blue (RGB) colour spaces strongly impact object detection. A robotic arm using a PixyCMU camera sensor detected

coloured items with 80% accuracy [18] and dimensional shapes with 98.33% accuracy [19] despite lighting sensitivity. Image segmentation, object recognition, and retrieval use edge detection in computer vision [20], [21]. The Canny edge detector is best for accuracy, noise robustness, and efficiency [22].

Planning and controlling visual robot movement data requires precise algorithms [23]. Low dexterity makes robot manipulators grasp poorly [24]. Enhancing robot object capture perception is research [25]. Robots measure size and strength with stereoscopic, monovision, and 3D cameras [26]. To properly manoeuvre robotic manipulators, kinematic motion planning analyses joint movements and workspace limits [27]. In forward kinematics (FK), joint angles and link lengths define the end-effector position, while in inverse kinematics (IK), joint angles determine the desired pose [28]. Practical applications require FK and IK, but FK is easier to implement [29]. IK solves nonlinear equations using Newton-Raphson or optimisation [30]. IK scenarios benefit from the efficient Levenberg-Marquardt (LM) algorithm [31]. IK is computationally intensive. Therefore, robot complexity, precision, and real-time constraints are a concern. Sensors and servos hinder robot manipulator troubleshooting. Robotic operating system (ROS) GPU speeds up IK computations for optimisation and sophisticated kinematic models. ROS, computational optimisation, specific libraries, and processing capacity enable accurate robot operations.

Previous research emphasises the benefits of machine vision and robotic arms for quick colour sorting but also highlights real-time image processing, colour space selection, edge detection, and robot-environment interaction issues. RGB and Canny edge detection will be used to construct an OpenCV colour and edge detection method. Second, image object and destination positions can be converted to Cartesian coordinates for robot manipulator motion planning. Forward and enhanced IK are implemented on a ROS-based robot manipulator using the LM algorithm to improve motion efficiency.

2. KINEMATICS EQUATION

2.1. Kinematic model

The traditional Denavit-Hartenberg (DH) modelling method is widely used to establish a link coordinate system for each joint in robotic manipulators. This approach assigns specific parameters to define the relative positions and orientations of adjacent links, ensuring consistency in kinematic analysis. The DH parameters of the 6-DOF robot manipulator used in this work are presented in Table 1.

Table 1. DH parameters of 6-DOF robot manipulator used in this work

Joint	Θ_i (°)	α_{i-1} (°)	a_i (mm)	d_i (mm)
J_0	0	0	0	143.5
J_1	Θ_1	$\Pi/2$	0	28
J_2	Θ_2	0	83	0
J_3	Θ_3	0	83	0
J_4	Θ_4	$-\Pi/2$	0	72.5
J_5	Θ_5	$-\Pi/2$	0	116

2.2. Forward kinematic

Robot manipulator operation relies on FK to translate joint angles into end-effector position and orientation. It checks the precision of servo movement and detects errors that cause rotation instability. Applying homogeneous transformation matrices that represent rotations and translations based on the DH parameters in Table 1 can sequentially transform the position and orientation of each link's origin until the 6-DOF robot manipulator's end-effector. Therefore, the coordinate for the centre point of the end-effector (ECP) can be calculated by using the transformation matrix, T_1^6 as (1). In (1) can be expressed in the transformation matrix, T form, as (2). Derive the six-dimensional coordinates of the manipulator's pose by transforming the matrix, T , to achieve a comprehensive solution for the positive kinematics of the manipulator.

$$T_1^6 = T_0^1 T_1^2 T_2^3 T_3^4 T_4^5 T_5^6 \quad (1)$$

$$T = \begin{bmatrix} a_x & b_x & c_x & p_x \\ a_y & b_y & c_y & p_y \\ a_z & b_z & c_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

2.3. Optimised inverse kinematics using Levenberg-Marquardt

The LM optimisation of IK can be used to minimise the difference between the current end-effector position and the desired target position. The inverse kinematic controls the servo motors to the desired location. Firstly, the position error, P_e is calculated using (3) by subtracting the initial coordinate (x_0, y_0, z_0)

with the target coordinate (x_t, y_t, z_t) . Then, the LM pseudo-inverse (LMPI) is implemented using (4). In this phase, the initial joint variable values, e.g., all joint variable values, are set to 0. LMPI is an iterative procedure that will stop when the convergence joint variable values with the given end-effector position are achieved.

$$P_e = [(x_t - x_0), (y_t - y_0), (z_t - z_0)] \quad (3)$$

$$LMPI = (T_i^j + \lambda I)^{-1} \quad (4)$$

where, I is identity matrix, λ is damping parameter, and T_i^j is transformation matrix for each joint. Then, the joint velocity, JV_i for each joint of the robot manipulator is calculated using (5):

$$JV_i = LMPI \times P_e \quad (5)$$

JA_i is added with the JV_i for each joint angle to obtain the new angle, JA_New_i using (6):

$$JA_New_i = JA_i + JV_i \quad (6)$$

If the position error is less than the tolerance value, that is 0.0001, then the optimisation is converged.

3. METHOD

3.1. Hardware structure of the robot manipulator

Figure 1 shows the 6-DOF Yahboom DOFBOT arm made of anodised aluminium alloy used in this study. A Raspberry Pi processor, Arduino Mega expansion board, and I²C servo bus are included. The servo bus controls five 15 kg YB-P15M servo motors and one Yahboom 6 kg gripper motor. The manipulator also has a 640×480 vision sensor with a 30 fps frame rate and 110-degree view angle. With an error margin of less than 1 degree, each servo motor can cover 180 degrees. The processor runs Ubuntu 20.04 and ROS Noetic. A graph paper is placed before the robot manipulator to test the pick-and-place target location (XY-coordinate). This graph paper states that the smallest coordinate resolution is 1 mm. This robot manipulator uses six servos: *ServoID_1*, *ServoID_2*, *ServoID_3*, *ServoID_4*, *ServoID_5*, and *Servo_Gripper*. A laser beam marks the end-effector's final graph paper position. DOFBOT robot manipulator translations peak at 34.4 cm. This study's longest linear distance for testing is 16.6 cm due to the Z-axis, which must always face down to track the end-effector through the laser beam.

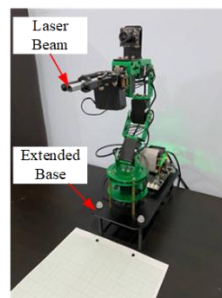


Figure 1. A DOFBOT Yahboom 6-DOF robot manipulator is used in this work

3.2. Object and colour detections

Image processing identifies things, the OpenCV library was used. Image processing-based color and object detection are shown in Figure 2. The code started RGB webcam recording. To simplify processing, pictures, Figure 2(a), were converted to grayscale, Figure 2(b). Image processing algorithms process one-third of data, making them computationally light and accelerating real-time apps. Grayscale and Gaussian images were blended to remove sensor limitations, light fluctuations, and edge detection mistakes. A low-pass filter blurs edges and removes high-frequency noise to smooth the image, as in Figure 2(c). Object boundaries are found using Canny edge detection, Figure 2(d). Hysteresis thresholding finds essential edges. Strong edges with gradient magnitudes above the high threshold are accepted instantaneously. Between low

and high thresholds, edge connections must be more substantial. Greater image intensity robustness and noise reduction. Next, the object's midpoint was calculated for accurate pick-and-place. This experiment uses squares, and (7) locates the midpoint of the square, as shown in Figure 2(e).

$$M1 = \frac{\sqrt{(Y1 - X1)^2 + (Y2 - X2)^2}}{2} \quad (7)$$

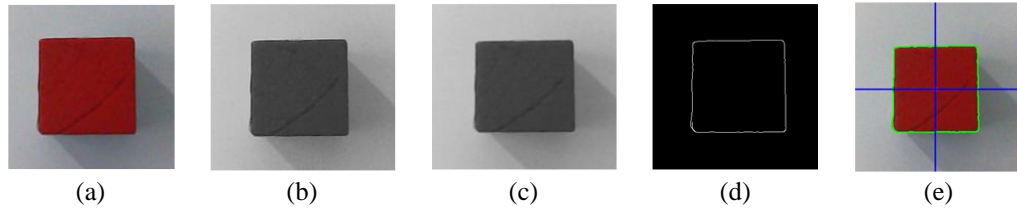


Figure 2. The captured image: (a) is converted, (b) a grayscale image, (c) Gaussian filter is applied for edge detection improvement, (d) Canny edge detection is applied to detect the edge of the object, and (e) the centroid of the detected object is computed

3.4. Image-to-coordinate translation for motion planning

Image-to-coordinate translation involves converting the information captured by the robot's vision system (images) into precise coordinates the manipulator can use to navigate its workspace and interact with the objects. A homogeneous transformation matrix is used for this purpose. It enables one to find the position of a point in the robotic arm base frame, given the position of a point in the camera reference frame. To do this, the camera lens is parallel to the surface. The pixel resolution must be known as *pixel_res*. It can be calculated by using (8):

$$pixel_res = \frac{width\ length\ in\ cm}{width\ length\ in\ pixel} \quad (8)$$

The computation of the XY-coordinates, (x_{coord}, y_{coord}) can be transformed into SI by using (9) and (10), with x_{pixel} and y_{pixel} are the total number of pixels to reach these points.

$$x_{coord} = x_{pixel} \times pixel_res \quad (9)$$

$$y_{coord} = y_{pixel} \times pixel_res \quad (10)$$

The relationship between the object's position from the robot and its position in the image frame can be calculated using (11):

$$P^R = H_C^R \times P^C \quad (11)$$

where, P^R is the position of the object from the robot frame, P^C is the position of the object from the camera frame, and H_C^R is the homogenous relationship between P^R and P^C .

Then, the homogenous relationship between the object location from the viewpoint of the robot frame and camera frame, H_C^R , can be defined as (12):

$$H_C^R = \begin{bmatrix} R_C^R & d_C^R & * & * \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (12)$$

where, H_C^R is the homogenous relationship between P^R and P^C , R_C^R is the rotation from the robot frame to the image frame, d_C^R is the displacement from the robot frame to the image frame, and * leave as blank.

The rotation of the robot frame to the camera frame, R_C^R this can be calculated using (13):

$$R_C^R = \begin{bmatrix} \cos(\theta_x) & -\sin(\theta_x) & 0 \\ \sin(\theta_x) & \cos(\theta_x) & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta_y) & -\sin(\theta_y) \\ 0 & \sin(\theta_y) & \cos(\theta_y) \end{bmatrix} \quad (13)$$

where, θ_x is the x rotation of the robot angle to the camera angle and θ_y is the y rotation of the robot angle to the camera angle.

The displacement of the robot frame and camera frame, d_C^R is determined by using (14). The displacement can be measured using a ruler, which measures the displacement in the X-axis and Y-axis between the robot and camera frames.

$$d_C^R = [x_d, y_d] \quad (14)$$

where, x_d is the displacement from the robot frame to the camera frame along the X-axis and y_d is the displacement from the robot frame to the camera frame along the Y-axis.

After the homogenous relationship is calculated, the object's position in the real world can be obtained through (15):

$$P^R = H_C^R \times \begin{bmatrix} x_{coord} \\ y_{coord} \\ 0 \\ 1 \end{bmatrix} \quad (15)$$

3.5. Performance analysis










The robot manipulator's pick-and-place accuracy for sorting must be tested using end-effector position precision and camera image object recognition accuracy. We evaluated the servo motor angle and end-effector location. The latter checks the object's edge colour and size. The robotic manipulator's whole motion was carefully analysed for servo motors. Smoothness variations like binding or jerking could suggest mechanical faults were checked. The servo motor's accuracy was tested using its embedded encoder, a feedback sensor that captures output angle data corresponding to input commands. In (16) was used to compute the servo motor absolute error. An average error value was calculated after ten replications to ensure statistical robustness and eliminate random fluctuations.

$$|S_e| = \theta_i - \theta_o \quad (16)$$

where, S_e is angle error of the servo motor, θ_i is input angle of the servo motor, and θ_o is output angle of the servo motor.

The robot manipulator uses IK to analyse the end-effector's position to reach target positions in Cartesian space (x, y). After calculating joint angles using IK, the manipulator moves. After that, the end-effector's final position is accurately recorded. In (17) quantifies accuracy using the Euclidean distance (E_d). This distance represents the difference between the target position (x_1, y_1) and the end-effector's location (x_2, y_2). A camera on a robot manipulator measured RGB values of paper prints for colour detection analysis. The camera's RGB values are compared to the true values to determine colour detection accuracy. As shown in Table 2, the test is repeated with different colours.

Table 2. The reference colours used in the colour detection experiment

RGB value	HEX Value	Colour	RGB Value	HEX Value	Colour
(255,0,0)	#FF0000		(65,135,90)	#41875A	
(0,255,0)	#00FF00		(50,220,40)	#32DC28	
(0,0,255)	#0000FF		(100,100,100)	#646464	
(0,20,40)	#001428		(210,165,135)	#D2A587	
(40,20,0)	#281400				

The error (R_e, G_e, B_e) between measured and real RGB values is calculated using (17). Colour distance error is a quantitative measure of the difference between two RGB colours. In (19) and (20) calculate the colour distance, $|\Delta D|$ error using the Euclidean equation for RGB and its percentage.

$$E_d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (17)$$

$$|R_e G_e B_e| = (R_m - R_r), (G_m - G_r), (B_m - B_r) \quad (18)$$

where, R_m, G_m, B_m are the measured RGB value by the camera and R_r, G_r, B_r are the real RGB value

$$|\Delta D| = \sqrt{(R_m - R_r)^2 + (G_m - G_r)^2 + (B_m - B_r)^2} \quad (19)$$

where, R_m, G_m, B_m are the measured RGB value by the camera and R_r, G_r, B_r are the real RGB value.

$$|\Delta D\%| = \left(\frac{C_m - C_r}{C_r} \right) \times 100 \quad (20)$$

where, C_m is the component of measured RGB value by the camera (R_m, G_m, B_m) and C_r is the component of real RGB value (R_r, G_r, B_r).

This study evaluates Canny edge detection in a camera system. The test subject is a cubic object carefully positioned orthogonal to the camera's field of view for edge detection. The cubic's estimated edge profile is then extracted using the Canny edge detection algorithm. The cubic width is calculated based on the number of pixels in the detected edge region. In (8) calculates pixel resolution, which (21) uses to convert pixel count to width. This estimated width is compared to the cubic object's actual width. The experiment is meticulously replicated to find its average accuracy value to reduce random fluctuations and improve evaluation robustness. Ten different-sized objects are used in this experiment.

$$w_c = n \times \text{pixel_res} \quad (21)$$

where, n is the number of pixels that fall within the detected region.

4. RESULTS AND DISCUSSION

4.1. Servo angle analysis

Table 3 shows the results of the servo angle performance analysis. Each servo motor was rigorously tested during dynamic motion by the robot manipulator. To achieve this, a PC terminal was used to command *Servo_1*, *Servo_2*, *Servo_3*, *Servo_4*, and *Servo_5* to position themselves at 40°, 50°, 10°, 30°, and 50°. The encoders reported the achieved angles after servo movements. The measured values were (40°, 51°, 10°, 29°, and 50°), indicating servo errors of (0°, 1°, 0°, -1°, and 0°). The calculated average absolute error across all trials was 0.4°, which was remarkable. Further analysis showed that each servo motor had a maximum error of +1° or -1°. Motor positions and speeds have less jitter or rapid fluctuations, which accounts for this high accuracy. The robot manipulator's digital bus servo motors' low jitter shows their precision and stability, proving its suitability for high-precision applications. It benefits pick and place operations in colour sorting, which require high positional accuracy and control.

Table 3. Result of servo motors analysis

#/Servo_ID	Absolute error (°)					Average absolute error (°)
	1	2	3	4	5	
1	0	1	0	-1	0	0.4
2	1	1	0	-1	0	0.6
3	1	1	0	-1	0	0.6
4	1	0	0	-1	0	0.4
5	1	0	0	0	0	0.2
6	1	0	0	0	1	0.4
7	0	0	0	0	1	0.2
8	0	1	0	0	1	0.4
9	0	1	0	0	1	0.4
10	1	-1	-1	-1	1	1
Total average error (mm)						0.46

4.2. End-effector's position analysis

Based on the IK with LM optimisation, Table 4 shows the end-effector's final position calculated by Ed. Using 1-millimetre units, this table shows the end-effector's trajectory on a Cartesian plane with X and Y axes. Trials 8-10 had more significant errors when the end-effector extended further along the X -axis and the *Servo_2* angle exceeded 99°. These increased errors may be due to the control system's inability to manipulate the manipulator at long distances precisely or the robot's physical constraints. Although these occasional deviations occurred, 70% of experiments yielded promising results with errors less than 1 mm, proving the system's accuracy. The average error across all trials was 1.65 mm, bolstering the system's potential for moderate positional precision applications.

4.3. Object colour estimation analysis











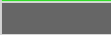
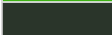
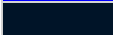
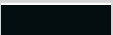


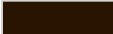
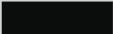
Table 5 compares the robot manipulator's camera's measured colour values and hexadecimal codes to the reference colours. Visual inspection shows that measured colours are consistently darker than

references. Without additional lighting during the experiment, this discrepancy is perceived. The experiment used only the room's ambient lighting, reducing the illumination of the analysis area. Thus, the observed colour shift is attributed to this reduced lighting level. Even though the darkness is visible, the colour difference is negligible. Nevertheless, the measured colours are still within the experiment's parameters and acceptable for accurate colour detection. To address this concern, we acknowledge the importance of investigating the variability in colour detection under different lighting conditions under a systematic experiment in future.

Table 4. Result of end-effector's position analysis

#	Target position		Final position		Euclidean error from the target position, E_d (mm)
	X_{coor} (mm)	Y_{coor} (mm)	X_{coor} (mm)	Y_{coor} (mm)	
1	0	0	0	0	0
2	0	10	0	10	0
3	0	20	0	20.5	0.5
4	0	30	0	31	1
5	0	40	0	41	1
6	20	0	20	0	0
7	30	0	30	0	0
8	40	0	43	4	5
9	50	0	49	2.9	3
10	60	0	63	5.2	6
Average error (mm)					1.65

Table 5. Result of colour detection by the robot manipulator's camera

Reference colour		Measured colour		Reference colour		Measured colour	
HEX	Colour	HEX	Colour	HEX	Colour	HEX	Colour
#FF0000		#E62323		#41875A		#1B522E	
#00FF00		#37FF1C		#32DC28		#3EB215	
#0000FF		#0028FA		#646464		#2A352A	
#001428		#040D10		#D2A587		#A07459	
#281400		#0B0D0C					

Using a similarity-based experiment, Table 7 analyse the robot manipulator's colour detection abilities. Table 6 shows higher E_d and ΔD values, which indicate more significant colour detection discrepancies. The smallest distance, 25.32 (from #001428), indicates accurate detection, while #646464's maximum distance was 95. The more comprehensive range suggests that colour detection performance varies widely across colours. Notably, the observed errors, ranging from 25 to 95 in colour distance, are acceptable. The average colour distance across all measurements is 57.84, confirming the excellent performance. The minimum deviation of 4.6% suggests little difference, while #646464's maximum of 21.3% matches the previously observed more significant error for this colour. This higher percentage difference is attributed to "poor lighting" during the experiment.

Table 6. The result of colour detection by the robot manipulator's camera is based on similarity values

#	Input colour			Measured colour			Euclidean distance, E_d			Ed for RGB, ΔD
	R_r	G_r	B_r	R_m	G_m	B_m	R_e	G_e	B_e	
1	255	0	0	230	35	35	25	35	35	55.45
2	0	255	0	55	255	28	55	0	28	61.72
3	0	0	255	0	45	250	0	45	5	45.28
4	0	20	40	4	13	16	4	7	24	25.32
5	40	20	0	11	13	12	29	7	12	32.16
6	65	135	90	27	82	46	38	53	44	78.67
7	50	220	40	62	178	41	12	42	1	43.69
8	100	100	100	42	53	42	58	47	58	94.54
9	210	165	135	160	116	89	50	49	46	83.77
Average Ed										57.84

4.4. Object's width estimation analysis

Table 7 shows the Canny edge detection results. The absolute error is calculated by comparing the estimated object width to its reference values. This table shows that each object has absolute errors of 0.2 to

0.8 mm. Due to poor lighting and the camera's inability to detect the larger object's edge, *Object 1* has the highest absolute error of 0.8 mm, indicating a more significant edge detection discrepancy. However, the consistency of the errors suggests that the Canny edge detection algorithm estimates object widths accurately.

Table 7. Result of object's width estimation

Object	Reference object's width (mm)	Estimated object's width, w_e (mm)	Absolute error (mm)
1	90.0	90.8	0.8
2	54.0	54.3	0.3
3	45.5	45.3	0.2
4	30.0	30.3	0.3
5	75.0	75.2	0.2
6	62.5	62.7	0.2
7	81.0	81.5	0.5
8	40.0	40.2	0.2
9	67.5	67.8	0.3
10	50.0	50.3	0.3
Average error (mm)			0.33

Overall, the proposed system outperforms traditional sorting robots using real-time image processing with ROS and OpenCV, enabling dynamic adjustments and higher sorting accuracy. It balances efficiency and cost-effectiveness, achieving precision with affordable hardware like Raspberry Pi and Arduino, unlike more expensive systems that require high computational power. Its modularity and flexibility allow adaptation to various sorting tasks, making it a versatile solution for industrial applications.

5. CONCLUSION

A 6-DOF sorting robot was created and tested using servos, vision, advanced kinematic models, and optimisation. In this investigation, the arm's pick-and-place precision was 0.46° absolute servo error and 1.65 mm positioning error. The average Ed of 57.84 in RGB values in colour detection trials is good but needs better lighting or image processing to improve accuracy. These results affect industrial automation, robotic system integration precision and real-time jobs like colour-based sorting. Advanced robotic applications in adaptive and autonomous contexts can benefit from LM IK. This study showed depth cameras reduced robot object recognition and spatial awareness errors. Machine learning or more significant optimisation for real-time decision-making can improve manipulator accuracy and efficiency. Using the robotic manipulator for industrial lines or precision agriculture beyond sorting could demonstrate its adaptability and impact. Further improvements can automate and address the growing need for intelligent and efficient robotic solutions in many sectors by expanding robotic system adoption.

ACKNOWLEDGEMENTS

The authors thank the School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA for providing the research facilities. This research is not supported by any grants.





REFERENCES

- [1] K. S. Ching, L. Y. Lian, L. W. Yun, M. P. Jun, and W. B. Jian, "the Industrial Revolution 4.0 in Malaysia: Challenges, Barriers, Obstacles, and Recommendations for Action By," 2020.
- [2] T. Z. Gen, D. C. Y. Kuen, V. M. Rao, and R. L. Oliveira, "Industry 4.0 Technology Adoption in Malaysian Manufacturing: Strategies for Enhancing Competitiveness," pp. 1–50, 2022.
- [3] A. Mohan, A. Jatin, and P. Vashisht, "Enhancing Robotic Arm Performance: Integrating Arduino Control and Aerodynamic Principles for 6 Degrees of Freedom," *International Journal of Innovative Research in Engineering and Management*, vol. 10, no. 3, pp. 141–145, 2023, doi: 10.55524/ijirem.2023.10.3.21.
- [4] N. Fangerow, D. Aschenbrenner, C. Colloseus, and R. Khoury, "Robot-assisted automated sorting techniques for plastic recycling," *Procedia CIRP*, vol. 120, pp. 1232–1237, 2023, doi: 10.1016/j.procir.2023.09.154.
- [5] Y. Li, "A Design of Robot System for Rapidly Sorting Express Carton with Mechanical Arm Based on Computer Vision Technology," *Highlights in Science, Engineering and Technology*, vol. 52, pp. 168–177, 2023, doi: 10.54097/hset.v52i.8885.
- [6] K. Peng and Z. Wang, "The Research on the Motion Control of the Sorting Manipulator based on Machine Vision," *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 3, pp. 206–210, 2023, doi: 10.14569/IJACSA.2023.0140323.
- [7] X. Zhang *et al.*, "Design and operation of a deep-learning-based fresh tea-leaf sorting robot," *Computers and Electronics in Agriculture*, vol. 206, 2023, doi: 10.1016/j.compag.2023.107664.
- [8] S. K. Chakraborty *et al.*, "Development of an optimally designed real-time automatic citrus fruit grading–sorting machine leveraging computer vision-based adaptive deep learning model," *Engineering Applications of Artificial Intelligence*, vol. 120, 2023, doi: 10.1016/j.engappai.2023.105826.




- [9] J. Rubi and M. M. Ahammad, "Utilisation of Robotic Systems in Automating Manufacturing Applications While Reducing the Amount of Labour and Production Costs and Time Associated With the Process," *Technoarete Transactions on Industrial Robotics and Automation Systems*, vol. 2, no. 1, 2022, doi: 10.36647/ttiras/02.01.a004.
- [10] C. Yang, Y. Wang, S. Lan, L. Wang, W. Shen, and G. Q. Huang, "Cloud-edge-device collaboration mechanisms of deep learning models for smart robots in mass personalization," *Robotics and Computer-Integrated Manufacturing*, vol. 77, 2022, doi: 10.1016/j.rcim.2022.102351.
- [11] C. Jalendra, B. K. Rout, and A. Marathe, "Vision sensor based residual vibration suppression strategy of non-deformable object for robot-assisted assembly operation with gripper flexibility," *Industrial Robot*, vol. 49, no. 5, pp. 851–864, 2022, doi: 10.1108/IR-09-2021-0197.
- [12] Y. Song, B. Chen, X. Liu, H. Weijun, X. Xiangyu, and Y. Yuqi, "Audio and video editing system design based on OpenCV," *Информатика. Экономика. Управление - Informatics. Economics. Management*, vol. 1, no. 2, pp. 0101–0120, 2022, doi: 10.47813/2782-5280-2022-1-2-0101-0120.
- [13] R. TH. Hasan and A. Bibo Sallow, "Face Detection and Recognition Using OpenCV," *Journal of Soft Computing and Data Mining*, vol. 2, no. 2, Oct. 2021, doi: 10.30880/jsedm.2021.02.02.008.
- [14] M. Abdullah-Al-Noman, A. N. Eva, T. B. Yeahyea, and R. Khan, "Computer Vision-based Robotic Arm for Object Color, Shape, and Size Detection," *Journal of Robotics and Control (JRC)*, vol. 3, no. 2, pp. 180–186, 2022, doi: 10.18196/jrc.v3i2.13906.
- [15] I. Martinez-Alpiste, P. Casaseca-de-la-Higuera, J. M. Alcaraz-Calero, C. Grecos, and Q. Wang, "Smartphone-based object recognition with embedded machine learning intelligence for unmanned aerial vehicles," *Journal of Field Robotics*, vol. 37, no. 3, pp. 404–420, 2020, doi: 10.1002/rob.21921.
- [16] A. Marcireau, S. H. Ieng, and R. Benosman, "Sepia, Tarsier, and Chameleon: A Modular C++ Framework for Event-Based Computer Vision," *Frontiers in Neuroscience*, vol. 13, 2020, doi: 10.3389/fnins.2019.01338.
- [17] Z. Qiu, J. Liu, H. Sun, L. Lin, and Y. W. Chen, "CoSTHR: A Heart Rate Estimating Network With Adaptive Color Space Transformation," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, 2022, doi: 10.1109/TIM.2022.3170976.
- [18] M. Q. Abbood, "Detection of Objects Geometries and Colors using MATLAB," *AIP Conference Proceedings*, vol. 2660, 2022, doi: 10.1063/5.0107738.
- [19] Mohd. A. Ansari and D. K. Singh, "Significance of Color Spaces and Their Selection for Image Processing: A Survey," *Recent Advances in Computer Science and Communications*, vol. 15, no. 7, 2021, doi: 10.2174/2666255814666210308152108.
- [20] X. Wang, S. Wang, Y. Guo, K. Hu, and W. Wang, "Coal gangue image segmentation method based on edge detection theory of star algorithm," *International Journal of Coal Preparation and Utilization*, vol. 43, no. 1, pp. 119–134, 2023, doi: 10.1080/19392699.2021.2024173.
- [21] M. Wang, C. Sun, and A. Sowmya, "Efficient corner detection based on corner enhancement filters," *Digital Signal Processing: A Review Journal*, vol. 122, 2022, doi: 10.1016/j.dsp.2021.103364.
- [22] H. Sekkat, S. Tigani, R. Saadane, and A. Chehri, "Vision-based robotic arm control algorithm using deep reinforcement learning for autonomous objects grasping," *Applied Sciences (Switzerland)*, vol. 11, no. 17, 2021, doi: 10.3390/app11177917.
- [23] P. A. Zachares, M. A. Lee, W. Lian, and J. Bohg, "Interpreting Contact Interactions to Overcome Failure in Robot Assembly Tasks," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, 2021, pp. 3410–3417, doi: 10.1109/ICRA48506.2021.9560825.
- [24] H. M. Balanji, A. E. Turgut, and L. T. Tunc, "A novel vision-based calibration framework for industrial robotic manipulators," *Robotics and Computer-Integrated Manufacturing*, vol. 73, 2022, doi: 10.1016/j.rcim.2021.102248.
- [25] C. Dai, S. Zhuang, G. Shan, C. Ru, Z. Zhang, and Y. Sun, "Automated End-Effector Alignment in Robotic Micromanipulation," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 5, pp. 3932–3941, 2022, doi: 10.1109/TMECH.2022.3150800.
- [26] Z. Zhou, L. Li, A. Fürsterling, H. J. Durocher, J. Mouridsen, and X. Zhang, "Learning-based object detection and localization for a mobile robot manipulator in SME production," *Robotics and Computer-Integrated Manufacturing*, vol. 73, 2022, doi: 10.1016/j.rcim.2021.102229.
- [27] G. Colucci, A. Botta, L. Tagliavini, P. Cavallone, L. Baglieri, and G. Quaglia, "Kinematic Modeling and Motion Planning of the Mobile Manipulator Agri.Q for Precision Agriculture," *Machines*, vol. 10, no. 5, 2022, doi: 10.3390/machines10050321.
- [28] D. Manolescu and E. L. Secco, "Design of a 3-DOF Robotic Arm and Implementation of D-H Forward Kinematics," *Congress on Intelligent Systems*, 2023, pp. 569–583, doi: 10.1007/978-981-19-9225-4_42.
- [29] M. H. Sayour, S. E. Kozhaya, and S. S. Saab, "Autonomous Robotic Manipulation: Real-Time, Deep-Learning Approach for Grasping of Unknown Objects," *Journal of Robotics*, vol. 2022, 2022, doi: 10.1155/2022/2585656.
- [30] K. Ayusawa, A. Murai, R. Sagawa, and E. Yoshida, "Fast Inverse Kinematics Based on Pseudo-Forward Dynamics Computation: Application to Musculoskeletal Inverse Kinematics," *IEEE Robotics and Automation Letters*, vol. 8, no. 9, pp. 5775–5782, 2023, doi: 10.1109/LRA.2023.3300207.
- [31] J. Zhang, Y. Shi, Y. Ma, L. Xu, J. Yu, and J. Wang, "IKOL: Inverse Kinematics Optimization Layer for 3D Human Pose and Shape Estimation via Gauss-Newton Differentiation," *Proceedings of the 37th AAAI Conference on Artificial Intelligence, AAAI 2023*, vol. 37, pp. 3454–3462, 2023, doi: 10.1609/aaai.v37i3.25454.

BIOGRAPHIES OF AUTHORS






Muhammad Hafidz Hasnor Faiz     is a postgraduate student at the School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA (UiTM). He began coding during his Bachelor's degree, focusing on robotics. He later pursued a Master's degree in artificial intelligence and machine learning. During this time, he also practised data engineering. He frequently programs in Python for applications such as robotics, among others. He is proficient in data engineering languages such as SQL and tools like dbt, snowflake, databricks, apache spark, and more. He can be contacted at email: hafidzhasnor1@gmail.com.






Dr. Ir. Norashikin M. Thamrin    is an Associate Professor at the School of Electrical Engineering, College of Engineering, Universiti Teknologi MARA (UiTM), Malaysia where she has become a faculty member since 2008. She graduated from Universiti Teknologi Malaysia with her bachelor's degree (honours) in electrical-electronic engineering and Master's in Engineering in 2005 and 2007, respectively. She then received her Ph.D. in automation and robotics from Universiti Teknologi MARA (UiTM) in 2017. Her research interest is primarily in automated system development, water security, agriculture, and robotics. She has become the author/co-author of more than 50 publications. She can be contacted at email: norashikin@uitm.edu.my.



Megat Syahirul Amin Megat Ali    is an Associate Professor at the College of Engineering, Universiti Teknologi MARA. He received his B.Eng. (Biomedical) from University of Malaya, Malaysia, in 2006. He then obtained M.Sc. in Biomedical Engineering from University of Surrey, United Kingdom in 2007. He received his Ph.D. in Electrical Engineering from Universiti Teknologi MARA, Malaysia, in 2018. He is a Chartered Engineer, registered with the Engineering Council, United Kingdom, and Senior Member of the Institute of Electrical and Electronics Engineers. He is currently the Deputy Director of Microwave Research Institute, Universiti Teknologi MARA, Malaysia. His main research interests are in biomedical signal processing and machine learning. He can be contacted at email: megatsyahirul@uitm.edu.my.



Idris Zainal Abidin    is a seasoned electronics engineer with a decade of experience. He holds a Bachelor of Engineering in Electrical Engineering from Universiti Teknologi MARA, Shah Alam. His education and professional experience at SIRIM and Cytron Technologies have equipped him with a strong foundation in PCB design, firmware engineering and microcontroller applications. He specialises in projects involving Raspberry Pi and motor drivers and is currently a Technology Marketing Strategist at Cytron. He can be contacted at email: idris@cytron.io.