ISSN 2347 - 3983

Volume 8. No. 10, October 2020

International Journal of Emerging Trends in Engineering Research

Available Online at http://www.warse.org/IJETER/static/pdf/file/ijeter328102020.pdf

https://doi.org/10.30534/ijeter/2020/328102020

Kinematic Modeling of A Low Cost 4 DOF Robot Arm System

Muhammad Ashraff Mohd Adzeman, Mohd Hairi Mohd Zaman, Muhammad Fikri Nasir, Mohd Faisal Ibrahim, Seri Mastura Mustaza

Department of Electrical, Electronic and Systems Engineering, Faculty of Engineering and Built Environment, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

ashraffadzeman@gmail.com, hairizaman@ukm.edu.my, mfn2704@gmail.com, faisal.ibrahim@ukm.edu.my, seri.mastura@ukm.edu.my

ABSTRACT

Robot arm systems are often used worldwide to enhance the quality and productivity of production in the manufacturing industry. Typical applications of robot arm systems include assembly, painting, welding, and pick and place operations. The number of degree-of-freedom (DOF) for most robots in the industry is limited to 6 DOF. An increase in DOF causes a robot arm design to become considerably complicated. This study mainly aims to perform modeling of a low cost 4 DOF robot arm. Modeling simulation of a robot arm was performed using MATLAB. The transformation matrix for each robot arm joint was obtained using the Denavit and Hartenberg convention technique. The modeling concept used for the actual and simulated robot arm is the forward and inverse kinematic models. The accuracy of the robot arm modeling was evaluated by calculating the error value between the modeling of the actual and simulated robot arms. This study successfully modeled a robot arm, although other easy modeling approaches should be investigated in the future.

Key words : Robot arm modeling, DH parameter, degree of freedom, DOF, kinematic.

1. INTRODUCTION

A robot arm is an automatic operation device designed to move similar to human hands and arms to achieve specified tasks, such as object grasping, handling, and other tasks. Moreover, robot arms can replace humans in dangerous or hazardous working environments. Robot arms are characterized as having a long-run operation with no end, which is essential for the enhancement of production efficiency and merchandise quality, as well as mechanization and automation of business production. Therefore, robot arms are widely utilized in manufacturing, healthcare, agriculture, and other fields [1]–[5].

Currently, the most common industrial robots used are robot arms, which are known as serial manipulators. Manipulators are constructed as a series of links (mechanical extensions) connected by motor-actuated joints and mounted onto a fixed-point base [6]. Kinematics play a significant role in robots, particularly for research on industrial manipulators' behavior. Therefore, the first step involved in any robot system is the analysis and modeling of the robot arm kinematics [7], [8]. The analysis and modeling processes include forward and inverse kinematics. Forward kinematics is the process for obtaining end-effector position and posture by using a given joint angle. By contrast, inverse kinematics is the process for obtaining the value of a joint angle using a given end-effector position and posture.

The main problem that frequently occurs when dealing with robot arms is obtaining the kinematic model that is considerably challenging to find. The reason is the lack of specific information, such as robot parameters and encoder, provided by robot manufacturers. Additionally, the simulation process is complicated because of the lack of an accurate model. For the simulation part, the software that can model kinematics includes MATLAB, Roboanalyser, and Gazebo. Some software requires users to have the Robot Operating System (ROS) and the Ubuntu operating system to use Gazebo.

Therefore, this study focuses on the modeling and simulation of the OpenManipulator robot arm from Robotis, a low-cost robot with four revolute joints. Some of the parts of this robot model can be 3D printed, and the 3D files can be obtained from the Robotis website [9]. Additionally, the current research focuses on the simulation using MATLAB. The comparison data between the modeling kinematic of the actual robot and simulation will also be presented in this paper.

1.1 Robot Arm Design

The design of robot arms is one of the crucial aspects of modeling this equipment. Several factors related to robot design are the number of servo motor, type of motor, and type of robot arm. The structure of robot arm, for example, Robotis OpenManipulator robot arm, typically consists of a base (1), waist (2), shoulder (3), upper arm (4), elbow (5), lower arm (6), and wrist (7), as depicted in Figure 1 [9], [10].

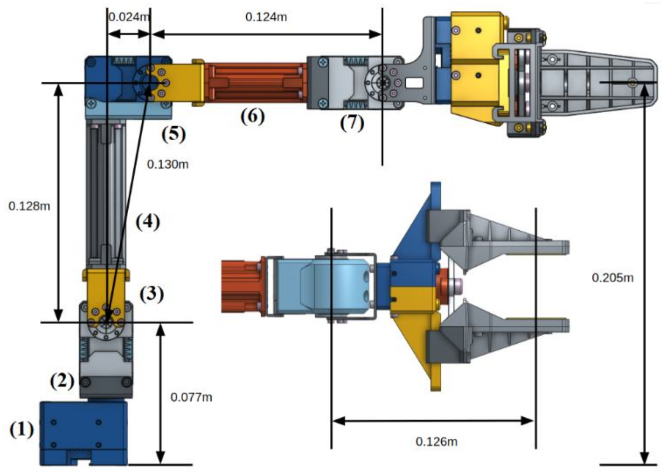

Figure 1: Structure of OpenMaipulator Robot Arm

The number of the motor correspond to the degree-of-freedom (DOF) of the manipulator. In the mechanical context, DOF refers to the mode where one system or device can move. One motor can only move in between 2 or 3 dimensions, but they have over 3 DOF [11]. Series or parallel manipulators or robot arms are built to provide an end effector with six parameters (i.e., 3 are translation and 3 are orientation). This configuration provides a direct relationship between the actuator and manipulator configuration, which can be described as the forward and inverse kinematics, respectively.

Several robot arm designs have been used in industry, and all robots can be classified by their corresponding DOF. Several configurations exist for the different types of robots, and every robot has its own function. Accordingly, robots often have different tasks that can be accomplished. 2-DOF robots are typically used in the agriculture industry. The demand for these robots continues to increase as the demand for food supply increases, but the labor is limited [3]. Particularly, the robot arm is frequently used is 4-DOF. The reason is that this robot can be used in all industries [7]. Meanwhile, Cartesian robots are typically used for fruit picking even though this robot lacks mobility. However, Cartesian robots are fast when picking fruits [12]. Another type of robot arm is a cylindric robot, which has two prismatic joints and one rotational joint. This robot can move in positive and negative z-directions and can rotate in the θ direction and transfer on the y-direction.

A polar or spherical robot has one massive telescopic arm attached to it. The robot moves when rotation occurs at the base, while the robot arm moves in an angular manner from top to bottom. This robot can be used in welding, assembly, painting, and material handling. Meanwhile, the selective compliance robot arm (SCARA) is a type of robot that is useful when tasks are needing repeated and fast movements. This robot can move in a uniform manner and accelerate in a circular motion. Its structure has two rotational joints that are parallel to the prismatic joint. The rotational and prismatic joints move at the horizontal and vertical axes, respectively. The last type of robot arm is a delta or parallel robot. Typical parallel robots have end-effectors with n DOF linked to a fixed base. The connection of this robot is needed to have two kinematic chains that will move the robot. This parallel robot arm typically has a closed-loop kinematic chain mechanism, and the manipulator is attached to a fixed base.

1.2 DH Parameter

In the Denavit and Hartenberg (DH) parameter convention, the link notation can describe the spatial relationship between two relative joints. The relationship of two coordinate link frames needs six parameters (i.e., three translational and three rotational). The frame that connects with another joint has a relationship to the position and geometry with another joint. Figure 2 shows that the link parameters are (α_i) and (a_i), and the offset of the joint (d_1) is fixed to provide the manipulator configuration. Using n of θ_1, this configuration will achieve a specific posture [13], [14]. The posture of every configuration or end effector can be changed if the value of θ_1 changes.

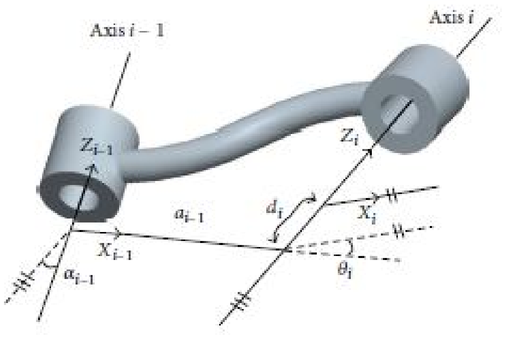

Figure 2: Definition Parameter of a Robot Arm [11]

Several techniques are used for modeling robot arms, one of which is using the DH parameters. Once the link frame has been set, the position and orientation of the *i*-frame in relation to *i-1* are completely defined using four parameters, known as the DH parameters [9]. The parameters taken are as follows:

1. b_i/d_i (Joint offset)
   The length of the intersection point occurs at the common normal line on the axis of the Zi joint. The measured value is at a distance between x_i and x_{i+1}, which is measured along z_i.

2. θ_i (Joint angle)
   The angle between the orthogonal projection on the common normal line, x_i and x_{i+1}, and the normal plane to the z_i joint axis. The rotation direction is closely related to the parameter value; when the rotation is counterclockwise, the rotation is positive. This parameter is taken at degrees between x_i and x_{i+1}, measured approximately z_i.

3. L_i/a_i (Link length)
   This parameter is taken at the distance of the common normal to z_i and z_{i+1}, measured along x_{i+1}.

4. α_i (Twist angle)
   The angle is between the orthogonal projections on the axis of the joint z_i and z_{i-1} to the normal plane on the common normal line. This value can be obtained from the degree between z_i and z_{i+1}, measured by x_i.

A transformation matrix can be obtained based on the DH parameter, which defines the transformation of the *i*-frame relative to the *i-1* frame. This matrix can be represented as ${}^{i-1}_{i}T$, and can be calculated as,

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\Theta_i & -s\Theta_i c\alpha_i & s\Theta_i s\alpha_i & a_i c\Theta_i \\ s\Theta_i & c\Theta_i c\alpha_i & -c\Theta_i s\alpha_i & a_i s\Theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (1)$$

For *n* DOF consisting of *n+1* frames, the sum of the numbers for the transformation matrix is *n* [13]. The concatenated matrix provides the required transformation from the n frame that corresponds to the final effector to the 0-frame mounted on the base as follows:

$$ {}^{0}_{n}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T \dots {}^{n-1}_{n}T \quad (2)$$

2. METHODOLOGY

Some considerations should be emphasized to ensure the robot arm to maneuver correctly, including the software and hardware used. Accordingly, software should be stable and easy to use to prevent difficulties. Furthermore, hardware plays a role in forming a robot arm that is strong and moving as planned. Figure 3 shows the general connection involving all hardware and software used in this study.

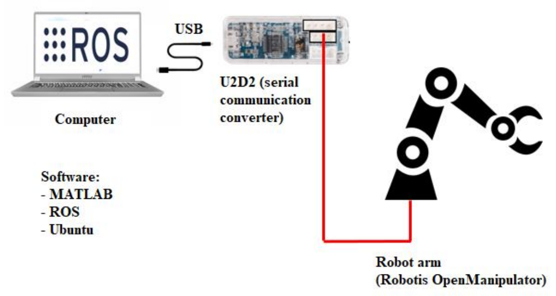
ROS
USB
Computer
U2D2 (serial communication converter)
Software:
- MATLAB
- ROS
- Ubuntu
Robot arm
(Robotis OpenManipulator)

Figure 3: Connection between Computer and Robot Arm

The three software used in this study is the Robot Operating System (ROS), Ubuntu OS, and MATLAB. ROS is an open-source robot framework software based on the Linux operating system. Briefly, ROS is a peer-to-peer communication system based on the concept of executable programming or a node that can communicate with others [16]–[18]. Ubuntu OS is a Debian Linux-based operating system. For the computer version, a graphical user interface type Unity or Gnome has been provided, while the server version has a set command-line interface.

For the hardware part, the primary hardware used in this study is the robot arm model OpenManipulator produced by the Robotis company. Table 1 lists several specifications of OpenManipulator [9].

Table 1: Specifications of the OpenManipulator Robot Arm

Items	Unit	OpenManipulator
Degree of freedom	N/A	4
Payload	g	500
Weight	kg	0.7
Gripper stroke	mm	20–75
Speed (joint)	RPM	46
Repeatability	mm	<0.2
Main controller	N/A	PC, 'OpenCR'

2.1 Kinematic Modeling

Using the transformation matrix calculation on the DH parameter, several things should be prioritized. To facilitate understanding, Figure 4 shows a flowchart detailing the activities performed during kinematic modeling.

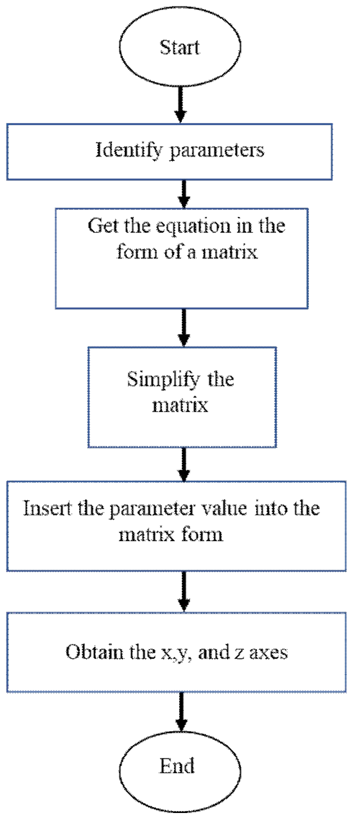
Start
Identify parameters
Get the equation in the form of a matrix
Simplify the matrix
Insert the parameter value into the matrix form
Obtain the x,y, and z axes
End

Figure 4: Flowchart of the Kinematic Modelling

The parameters of the robot should be first obtained. Table 2 shows a list of parameters that will be used for robot arm modeling purposes. These parameters are obtained based on the analysis method presented in Figure 4.

Table 2: Actual DH Parameters of OpenManipulator Robot Arm

Joint	**θ_i**	**α_i**	**d_i (m)**	**L_i (m)**
1	θ_1	90°	0	0.063
2	$90^\circ + \theta_2$	0°	0.13	0
3	θ_3	0°	0.124	0
4	0°	0°	0.126	0

After obtaining the parameter values, the parameters will be included in the transformation matrix in (1). Given that *n* used is 4 frames or better known as 4 DOF, the limit for the linked matrix is ${}^0_4T$. The equated matrix equations can be calculated as follows:

$$ {}^0_4T = {}^0_1T\,{}^1_2T\,{}^2_3T\,{}^3_4T \qquad (3)$$

$$ {}^0_1T = \begin{bmatrix} c\theta_1 & -s\theta_1 c(90) & s\theta_1 s(90) & (0)c\theta_1 \\ s\theta_1 & c\theta_1 c(90) & -c\theta_1 s(90) & (0)s\theta_1 \\ 0 & s(90) & c(90) & 0.063 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4)$$

$$ {}^1_2T = \begin{bmatrix} c\theta_2 & -s\theta_2 - 1 & 0 & (0.13)(c\theta_2) \\ s\theta_2 + 1 & c\theta_2 & 0 & (0.13)(s\theta_2 + 1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (5)$$

$$ {}^2_3T = \begin{bmatrix} c\theta_3 & -s\theta_3 + 1 & 0 & (0.124)(c\theta_3) \\ s\theta_3 - 1 & c\theta_3 & 0 & (0.124)(s\theta_3 - 1) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (6)$$

$$ {}^3_4T = \begin{bmatrix} c(0) & -s(0)c(0) & s(0)s(0) & 0.126c(0) \\ s(0) & c(0)c(0) & -c(0)s(0) & 0.126s(0) \\ 0 & s(0) & c(0) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (7)$$

After obtaining the preceding values, the next step is to combine them in sequence and simplify.

$$ {}^2_4T = {}^2_3T\,{}^3_4T \qquad (8)$$

$$ {}^1_4T = {}^1_2T\,{}^2_4T \qquad (9)$$

$$ {}^0_4T = {}^0_1T\,{}^1_4T \qquad (10)$$

After completing the simplifying process, the transformation matrix for (10) can be the same as (11). The values for p_x, p_y and p_z are the values for the end-effector coordinates *x*, *y*, and *z*, respectively.

$$ {}^0_4T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (11)$$

2.2 Simulation

Several steps or procedures should be followed to ensure that this simulation process runs smoothly. The flowchart in Figure 5 shows that this simulation procedure is performed in a systematic sequence. Several steps or procedures should be followed to ensure that this simulation process runs smoothly. The flowchart in Figure 5 shows that this simulation procedure is performed in a systematic sequence.

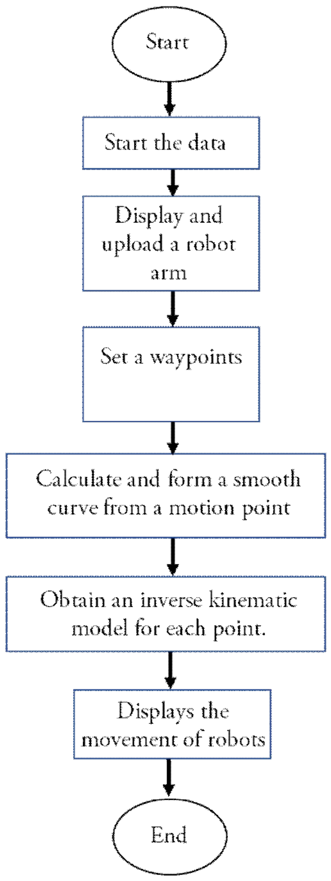
Start
Start the data
Display and upload a robot arm
Set a waypoints
Calculate and form a smooth curve from a motion point
Obtain an inverse kinematic model for each point.
Displays the movement of robots
End

Figure 5: Flowchart of the Simulation

The first step that should be completed is to initialize data using MATLAB. This procedure aims to identify and address the dependencies of the simulation files. Next, the algorithm included intends to display and upload the robot into the MATLAB system. After that, the algorithm configures a set of waypoints to be used, for example, eight waypoints. The point is obtained from the robot modeling of the arm and incorporated into the algorithm. This step is crucial to determine the similarities with real arm robots.

The next step is to calculate and form a smooth curve. This process runs the trajectory process and uses the waypoint configured in the previous step. The trajectory is one of the essential processes in robots, which involves moving the final effector smoothly from initial to target position [11].

The final process involves performing inverse kinematics for the points found in the robot working space. There are six values in the weights vector. The first three values are for rotational purposes, and the last three are for transitional purposes. Then, the algorithm configures a number greater than the number of waypoints configured previously to ensure the smooth movement of the robot arm. Thereafter, it uses a kinematic solver for each final effector position and uses the previous configuration to make the initial guess. After the completion of all processes, the robot movement is simulated.

3. RESULTS AND DISCUSSION

3.1 Actual Robot

The coordinate values of x, y, and z can be obtained from the DH parameter values based on the calculation techniques discussed in the methodology section. Table 3 shows the waypoint values selected for joints 1 to 4, used in this study. Data in Table 3 were entered into the (3) to obtain the coordinates x, y, and z. The values of x, y, and z can be obtained from (11), in which the values of p_x, p_y, and p_z correspond to x, y, and z, respectively. These values are listed in Table 4.

Table 3: Total of 8 Waypoints for Joints 1 to 4

Waypoint	Joint			
	1	2	3	4
1	−0.818	0.101	0.362	−0.425
2	−0.331	−0.781	0.245	0.563
3	0.256	−0.973	0.448	0.535
4	0.611	−0.831	1.16	−0.319
5	1.124	−1.267	1.293	−0.02
6	0.439	−1.275	0.655	0.641
7	0.063	0.092	0.626	−0.686
8	0.815	0.1	0.362	−0.431

Table 4: Coordinates for Actual Robot Arm

Waypoint	Axes		
	x	y	z
1	0.199	−0.197	0.142
2	0.163	−0.052	0.244
3	0.149	0.033	0.229
4	0.145	0.1	0.139
5	0.068	0.121	0.133
6	0.113	0.048	0.206
7	0.266	0.015	0.117
8	0.21	0.199	0.142

3.2 Simulated Robot

The parameter values are studied based on the kinematic modeling of real robots, and tests are conducted using MATLAB for the robot arm simulation. Table 5 shows the values obtained. Each waypoint configuration moves the robot arm according to the trajectory line based on the values in Table 5. Figure 6 shows the sequence of the simulated robot movement.

Table 5: Coordinates for Simulated Robot Arm

Waypoint	Axes		
	x	y	z
1	0.2	−0.2	0.15
2	0.161	−0.051	0.25
3	0.148	0.036	0.231
4	0.147	0.095	0.141
5	0.07	0.122	0.135
6	0.112	0.047	0.211
7	0.268	0.016	0.123
8	0.2	0.2	0.15

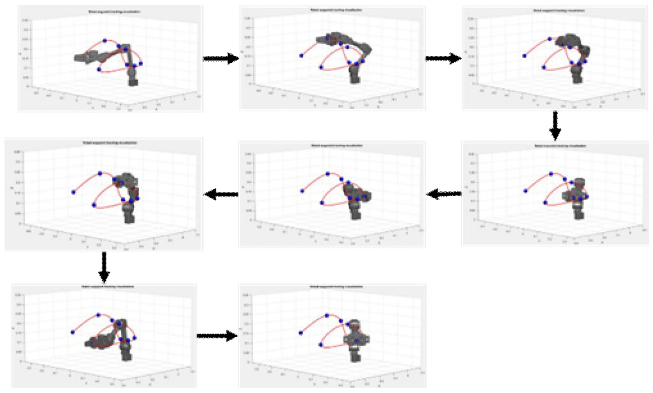
Figure 6: Movements of the Simulated Robot Arm

3.3 Data Comparison

Data from the actual robot arm was compared with the simulated robot arm to evaluate the performance. The first step is to examine the differences in the picture of each coordinate point position of the two robots. Figure 7 shows that the actual robot position and simulation results are similar. This outcome proves that the value of the DH parameter used is accurate.

Apart from the visualization, another performance measurement was conducted by computing the errors based on the values of the x, y, and z coordinates of both robots. The coordinate values for the actual and simulated robot arms are shown in Tables 4 and 5, respectively. The error values obtained are listed in Table 6. The most significant errors occurred at coordinate y.

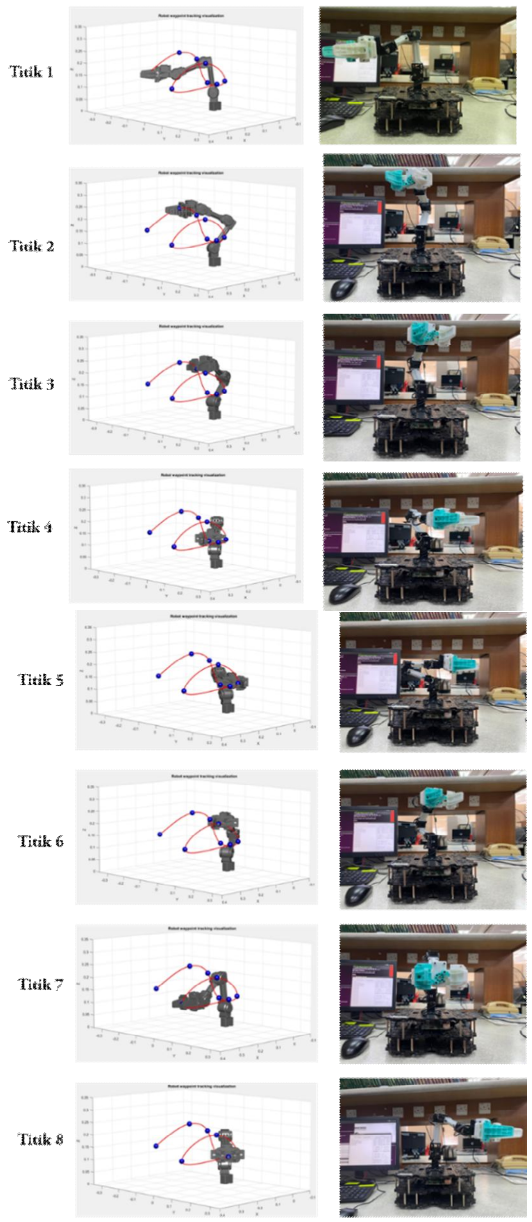

Figure 7: Comparison between the Actual and Simulated Robot Arm

Table 6: Coordinate Errors between the Actual and Simulated Robot Arms

Waypoint	**Error (%)**		
	x	**y**	**z**
1	0.5	1.5	5.333
2	1.242	1.961	2.4
3	−0.676	8.333	0.866
4	1.361	5.263	1.418
5	2.857	0.82	1.481
6	0.893	2.128	2.37
7	0.746	6.25	4.878
8	5	0.5	5.333

4. CONCLUSION

This study successfully modeled a low cost robot arm system and compared it with the simulated robot arm. The DH parameter values were also obtained. However, the developed modeling approach encounters calculation errors and need to be improved in the future.

ACKNOWLEDGEMENT

This work was financially supported by Universiti Kebangsaan Malaysia under the Research University Grant no. GUP-2018-103.

REFERENCES

[1] C. Han, H. Ma, W. Zuo, S. Chen, and X. Zhang, **A general 6-DOF industrial robot arm control system based on Linux and FPGA**, in *Proc. 2018 Chinese Control And Decision Conference*, 2018, pp. 1220–1225.

[2] M. H. M. Zaman, M. F. Ibrahim, N. Zainal, and M. F. Nasir, **Dual-arm robot with mobile robot platform with master-slave configuration for teleoperation application**, *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 9, no. 1.4 Special Issue, pp. 685–688, 2020.

[3] K. Rahul, H. Raheman, and V. Paradkar, **Design and development of a 5R 2DOF parallel robot arm for handling paper pot seedlings in a vegetable transplanter**, *Comput. Electron. Agric.*, vol. 166, pp. 1–13, Nov. 2019.

[4] O. O. Obe, E. A. Ajayi, and O. O. Odewale, **Genetic algorithm based optimal trajectories planning for robot manipulators on assigned paths**, *Int. J. Emerg. Trends Eng. Res.*, vol. 8, no. 8, pp. 4888–4893, 2020.

[5] H. Kevin, H. Tjahyadi, A. Aribowo, and A. S. Putra, **Designing the label giving robot arm in the packing box**, *Int. J. Emerg. Trends Eng. Res.*, vol. 8, no. 8, pp. 4867–4874, 2020.

[6] S. Ay, M. Kutay, and T. Ercan, **A low cost manipulator training set**, in *Proc. 2018 6th Int. Conf. on Control Eng. and Information Tech.*, 2018, pp. 1–6.

[7] A. A. Mohammed and M. Sunar, **Kinematics modeling of a 4-DOF robotic arm**, in *Proc. 2015 Int. Conf. on Control, Automation and Robotics*, 2015, pp. 87–91.

[8] K. Jahnavi and P. Sivraj, **Teaching and learning robotic arm model**, in *Proc. 2017 Int. Conf. on Intelligent Computing, Instrumentation and Control Technologies*, 2017, pp. 1570–1575.

[9] Robotis, **OpenMANIPULATOR-X: 2. Specification**, 2020. [Online]. Available: https://emanual.robotis.com/docs/en/platform/openmanipulator_x/specification/#hardware-specification. [Accessed: 24-Sep-2020].

[10] Z. Bian, Z. Ye, and W. Mu, **Kinematic analysis and simulation of 6-DOF industrial robot capable of picking up die-casting products**,” in *Proc. 2016 IEEE/CSAA Int. Conf. on Aircraft Utility Systems*, 2016, pp. 41–44.

[11] P. Corke. ***Robotics, Vision and Control: Fundamental Algorithms in MATLAB***, 2nd ed.

Switzerland: Springer, 2017.

[12] A. Chi Kit, B. Joshua, L. Shen Hin, and D. Mike, **Workspace analysis of Cartesian robot system for kiwifruit harvesting**, *Ind. Robot Int. J. Robot. Res. Appl.*, vol. 47, no. 4, pp. 503–510, 2020.

[13] S. Singh and E. Singla, **Service arms with unconventional robotic parameters for intricate workstations: Optimal number and dimensional synthesis**, *J. Robot.*, vol. 2016, pp. 1–11, 2016.

[14] A. A. Hayat, R. G. Chittawadigi, A. D. Udai, and S. K. Saha, **Identification of Denavit-Hartenberg parameters of an industrial robot**, in *Proc. Conf. on Advances In Robotics*, 2013, pp. 1–6.

[15] K. L. Li, W. Te Yang, K. Y. Chan, and P. C. Lin, **An optimization technique for identifying robot manipulator parameters under uncertainty**, *SpringerPlus*, vol. 5, no. 1, p. 1771, 2016.

[16] R. Mishra and A. Javed, **ROS based service robot platform**, in *Proc. 2018 4th Int. Conf. on Control, Automation and Robotics*, 2018, pp. 55–59.

[17] L. Garber, **Robot OS: A new day for robot design**, *Computer*, vol. 46, no. 12, pp. 16–20, 2013.

[18] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, **ROS: an open-source Robot Operating System**," in *Proc. Open-Source Software Workshop of the Int. Conf. on Robotics and Automation*, 2009, pp. 679–686.