Kinematic Analysis for Trajectory Planning of Open-Source 4-DoF Robot Arm

Han Zhong Ting[1], Mohd Hairi Mohd Zaman[2], Mohd Faisal Ibrahim[3], Asraf Mohamed Moubark[4]
Department of Electrical, Electronic and Systems Engineering
Faculty of Engineering and Built Environment,
Universiti Kebangsaan Malaysia
43600 Bangi, Selangor, Malaysia

***Abstract*—Many small and large industries use robot arms to establish a range of tasks such as picking and placing, and painting in today's world. However, to complete these tasks, one of the most critical problems is to obtain the desire position of the robot arm's end-effector. There are two methods for analyzing the robot arm: forward kinematic analysis and inverse kinematic analysis. This study aims to model the forward and inverse kinematic of an open-source 4 degrees of freedom (DoF) articulated robotic arm. A kinematic model is designed and further evaluated all the joint parameters to calculate the end-effector's desired position. Forward kinematic is simple to design, but as for the inverse kinematic, a closed-form solution is needed. The developed kinematic model's performance is assessed on a simulated robot arm, and the results were analyzed if the errors were produced within the accepted range. At the end of this study, forward kinematic and inverse kinematic solutions of a 4-DoF articulated robot arm are successfully modeled, which provides the theoretical basis for the subsequent analysis and research of the robot arm.**

***Keywords*—Robot arm; kinematic analysis; forward kinematic; inverse kinematic; open-source**

I. Introduction

A robot arm is made up of a movable chain of links that are linked together by joints. A hand or end-effector that can move freely in space is typically connected to one end, which is fixed to the ground. Robot arms are capable of performing repetitive tasks at speeds and precision well above those of human operators. Robot arms are used in various fields requiring high precision, for instance medical, industry, and some hazardous places. However, controlling the robot arm has been challenging with a higher degrees of freedom (DoF). Position analysis and trajectory planning of robot arm is an essential step in design and control.

Robotics research has gotten much traction over the years, thanks to advances in robot technology and the growing use of robotics in various fields. The robotics researchers have been focusing on more current configurations, intelligent actions, autonomous robotics, and high-level intelligence. All of these areas are related to robot kinematic, both forward and inverse. However, there are no excellent universal algorithms for producing the forward and inverse kinematic of a robot arm, especially finding the correlation between both kinematic models. The problem involves an error in achieving an accurate correlation between the forward and the inverse kinematic of a robot arm [1], [2], [3], [4], [5], [6], [7], [8], [9]. To plan an accurate robot arm's movement, we have to understand the relationship between the forward and inverse kinematic of the actuators to control the robot's resulting position in an accurate manner. Hence, there remains a scope to investigate further and work towards finding better solutions.

This study aims to develop a suitable methodology for solving the forward and inverse kinematic problem for a 4-DoF articulated robotic manipulator with relative ease. A kinematic model with an accurate correlation between the forward and inverse kinematics of a 4-DoF articulated robotic manipulator is developed using MATLAB software. The performance of the developed kinematic model is assessed and evaluated in Robot Operating System (ROS) on Ubuntu. The robot arm assessed in this research is the OpenMANIPULATOR-X robot arm manufactured by Robotis, which is an open-source robotic platform.

A. Kinematic Modeling

Kinematic is a branch of mechanics that studies the motion of bodies and structures without taking force into account. Geometry is used to research the movement of multi-DoF kinematic chains that make up the robot arm's structure. The relationship between the robot arm's linkages and its position, orientation, and acceleration is studied in robot kinematics. Kinematic analysis is an effective method when planning the robot arm's trajectory can be divided into two types: forward kinematic and inverse kinematic.

Forward kinematic refers to using the kinematic equations of a robot arm to compute the position of the end-effector's frame and joint variables relationship. Meanwhile, inverse kinematic is the reverse process that calculates the joint parameters that achieve an end-effector's position. The reverse operation, on the other hand, is much more challenging in general. The reverse operation is, in general, much more challenging. Forward kinematic has a simple and straightforward solution if compared with the inverse kinematic solution, which has many equations with a highly complex form to be solved [10].

In robotics, the inverse kinematic approach uses the kinematic equations to find the joint parameters that give each of the robot arm's end-effectors the desired configuration (position and rotation). Motion planning determines the robot arm's movement so that its end-effectors move from an initial configuration to the desired configuration. Inverse kinematic transforms the motion plan into joint actuator trajectories for the robot arm. Forward kinematic measures the chain's configuration using joint parameters, while inverse kinematic reverses this calculation to find the joint parameters that achieve the desired configuration [11].

B. Mechanism of Robot Arm

A robot arm can be either serial configuration with an open-loop structure or parallel configuration with a closed-loop structure. The joint type for an industrial robot arm can

be revolute or prismatic, and the link type can be rigid or flexible. Furthermore, a hybrid structure comprising both open and closed-loop mechanical chains is possible. The serial robot arm is characterized by the fact that the first joint is always starting from the fixed base and the end of the chain is free to move in space. Several different configurations of the robot arm can be formed due to the revolute and prismatic joints, and the axes of two adjacent joints can be either parallel or orthogonal. Orthogonal joints intersect by 90° with respect to their common normal, and it can be parallel when one axis rotates 90° [12]. Some of the robot arm mechanisms arise from open, closed, and hybrid open and closed kinematic chains.

In addition, the DoF can be defined as the particular motion of links related to any mechanism or machine. When performing a specific task, the DoF often plays an important role. The total number of DoF is always equal to the number of independent displacements of links. The number of DoF permitted by a joint and their characteristics can be determined by the design constraints of the body or link.

C. Articulated Robot Arm

The articulated robot arm is a robot arm with revolute joints and is also called a revolute, or anthropomorphic robot arm. The anthropomorphic resembles the human arm's design, including shoulder, elbow, and wrist joints [13]. The articulated robot arm can range from a simple two-joints structure to systems with 10 or more interacting joints [14]. The configuration of a 4-DoF articulated manipulator is shown in Fig. 1.

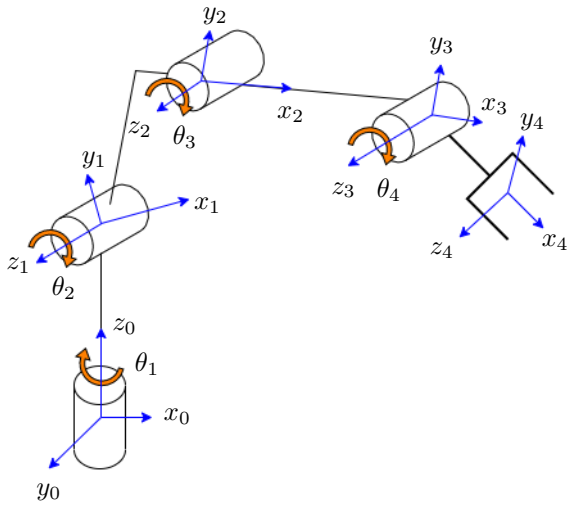


Fig. 1. Configuration of a 4-DoF Articulated Robot Arm.

D. Denavit-Hartenberg (D-H) Convention

In the Denavit-Hartenberg (D-H) convention, the link notation can describe the spatial relationship between two relative joints. The frame that connects with another joint has a relationship to the position and geometry with another joint. Fig. 2 shows that the link parameters are α_i and a_i, and the joint offset (d_i) is fixed to provide the manipulator configuration. This configuration achieves a specific posture using n of θ_i [15]. The posture of every configuration or end-effector can be changed if the value of θ_i changes.

Once the link frame has been set, the position and orientation of the i-frame in relation to $i-1$ are completely

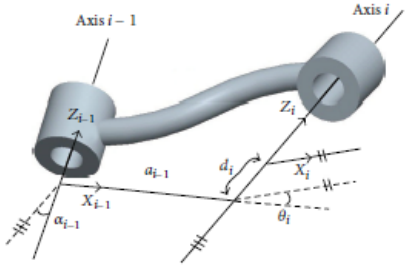


Fig. 2. Definition Parameters of a Robot Arm [15].

defined using four parameters, known as the D-H parameters. The parameters taken are as follows:

1) Joint offset, d_i: This parameter is the intersection point's length at the common normal line on the zi joint axis. The measured value is at a distance between x_i and x_{i+1}, which is measured along zi.
2) Joint angle, θ_i: This parameter is 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, it is positive. This parameter is taken at degrees between x_i and x_{i+1}, measured approximately z_i.
3) Link length, a_i: This parameter is taken at the distance of the common normal to z_i and z_{i+1}, measured along x_{i+1}.
4) Twist angle, α_i: This parameter is the angle 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 D-H parameters, which define the transformation of the i-frame relative to the $i-1$ frame. This matrix can be represented as T_i^{i-1}, and can be calculated as,

$$T_i^{i-1} = \begin{bmatrix} c(\theta_i) & -s(\theta_i)c(\alpha_i) & s(\theta_i)s(\alpha_i) & a_i c(\theta_i) \\ s(\theta_i) & c(\theta_i)c(\alpha_i) & -c(\theta_i)s(\alpha_i) & a_i s(\theta_i) \\ 0 & s(\alpha_i) & c(\alpha_i) & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where c and s represent sin and cos, respectively. For n DoF consisting of $n+1$ frames, the sum of the numbers for the transformation matrix is n [16]. The concatenated matrix provides the required transformation from the n frame that corresponds to the end-effector to the 0-frame mounted on the base as follows:

$$T_n^0 = T_1^0 \cdot T_2^1 \cdot T_3^2 \ldots T_n^{n-1} \quad (2)$$

E. Closed-Form Solution

A closed-form solution is an expression for an exact solution given with a finite amount of data [17]. If an equation solves a given problem in terms of functions and mathematical operations from a given commonly accepted set, it is said to be a closed-form solution. An infinite sum, for example, will not be considered a closed-form solution. The closed-form solution

can be applied in solving the inverse kinematic of the robot arm. The solution obtained has the advantages of being exact, includes all solution sets, and low computational cost.

II. Methods

Some considerations should be emphasized to ensure the robot arm can maneuver correctly, including its software and hardware. Accordingly, software should be stable and easy to use to prevent difficulties. Furthermore, hardware plays a role in forming a solid robot arm and able to move as planned. The MATLAB software (MathWorks) has been used to develop the algorithm in this study.

The primary hardware utilized in this study is the OpenMANIPULATOR-X robot arm manufactured by Robotis. The OpenMANIPULATOR-X robot arm is made up of pieces from the DYNAMIXEL-X series and 3D printing. The daisy chain method is adopted by DYNAMIXEL, which has a modular design. It enables users to add or remove joints for their own use easily. Fig. 3 shows the structure of the OpenMANIPULATOR-X robot arm [18].

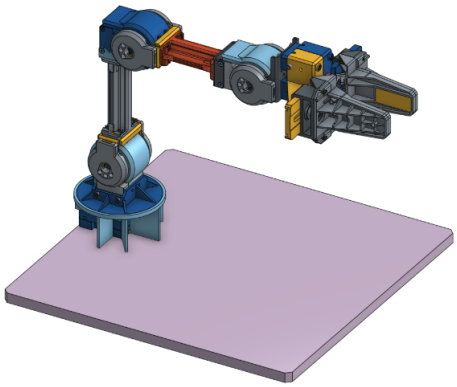


Fig. 3. Structure of the OpenMANIPULATOR-X Robot Arm [18].

A. Forward and Inverse Kinematic Analysis

Fig. 4 shows the flowchart of the forward and inverse kinematic analysis. Forward kinematic modeling was performed using the Denavit-Hartenberg (D-H) convention method. For inverse kinematic modeling, the closed-form solution method was used to obtain all sets of possible solutions. Next, correlation analysis between the forward and inverse kinematic was performed to determine whether there are any relationship between both kinematic models. Finally, performance evaluation was carried out using trajectory planning and waypoint tracking algorithm in MATLAB and ROS on Ubuntu.

B. Modeling of Forward Kinematic

The D-H convention method used to model the forward kinematic of the robot arm can be divided into three processes. The first process is the configuration analysis of the robot arm to obtain its D-H parameters. Fig. 5 shows the OpenMANIPULATOR-X robot arm's configuration analysis, while Table I shows the D-H parameters obtained after the configuration analysis is performed.

There is an offset between joint 2 and joint 3 on the robot arm, resulting in an offset angle. The offset angle is identified and considered in D-H parameters. Fig. 6 shows the calculation of the offset angle, θ_0, between joints 2 and 3.

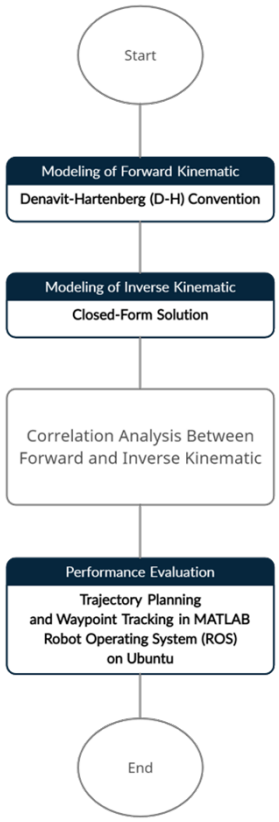


Fig. 4. Flowchart of the Process of Kinematic Modeling.

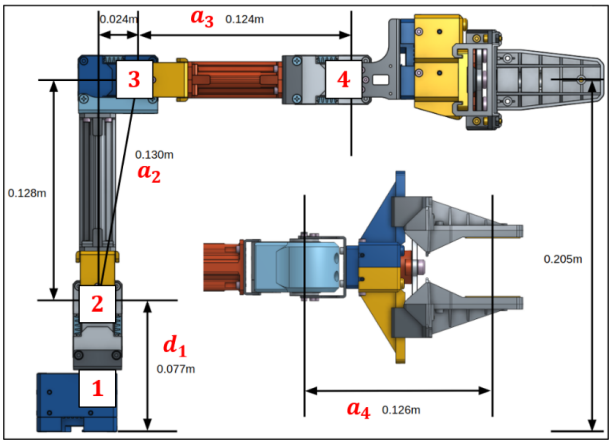
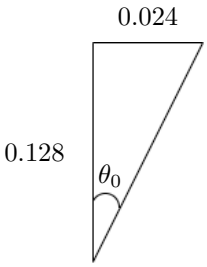


Fig. 5. Configuration Analysis of the OpenMANIPULATOR-X Robot Arm.

TABLE I. D-H Parameters

Joint	θ_i (°)	α_i (°)	a_i (m)	d_i (m)
1	θ_1	90	0	0.077
2	$\theta_1 - \theta_0$	0	0.130	0
3	$\theta_3 + \theta_0$	0	0.135	0
4	θ_4	0	0.126	0



$$\theta_0 = tan^{-1}\left(\frac{0.024}{0.128}\right) \approx 11^\circ$$

Fig. 6. Calculation of Offset Angle, θ_0 between Joint 2 and 3.

The second process in the D-H convention involves the calculation of the transformation matrix. After obtaining the value of the D-H parameters, the parameters will be included in the transformation matrix. Since 4 frames or 4 DoF were used, the limit for the linked matrix is T_4^0. The transformation matrix equations can be formulated as follows:

$$T_1^0 = \begin{bmatrix} c(\theta_1) & 0 & s(\theta_1) & 0 \\ s(\theta_1) & 0 & -c(\theta_1) & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

$$T_2^1 = \begin{bmatrix} c(\theta_2) & -s(\theta_2) & 0 & a_2c(\theta_2) \\ s(\theta_2) & c(\theta_2) & 0 & a_2s(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

$$T_3^2 = \begin{bmatrix} c(\theta_3) & -s(\theta_3) & 0 & a_3c(\theta_3) \\ s(\theta_3) & c(\theta_3) & 0 & a_3s(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$T_4^3 = \begin{bmatrix} c(\theta_4) & -s(\theta_4) & 0 & a_4c(\theta_4) \\ s(\theta_4) & c(\theta_4) & 0 & a_4s(\theta_4) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

After obtaining values for T_1^0, T_2^1, T_3^2 and T_4^3, the next step is to combine them in a sequence to simplify them.

$$T_4^2 = T_3^2 \cdot T_4^3 \tag{7}$$

$$T_4^1 = T_2^1 \cdot T_4^2 \tag{8}$$

$$T_4^0 = T_1^0 \cdot T_4^1 \tag{9}$$

The final process in the D-H convention is to obtain the position vector for the end-effector. After completing the simplification process, the transformation matrix equation is as follows:

$$T_4^0 = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

where $p(p_x, p_y, p_z)$ is the position vector of the end-effector while $n(n_x, n_y, n_z)$, $o(o_x, o_y, o_z)$ and $a(a_x, a_y, a_z)$ are orthogonal unit vectors that determine the orientation of the frame for the end-effector. p_x, p_y, and p_z are the values for x, y, and z coordinates of the end-effector.

C. Modeling of Inverse Kinematic

The closed-form solution method involving geometric and algebraic solutions is used to model the robot arm's inverse kinematic. The solution is divided into two processes. The first process is the calculation to get the angle for joint 1. The movement of the robot arm on the $x-y$ surface depends only on the angle of joint 1.

Regarding the x-y surface projection of the robot arm movement, the angle for joint 1, θ_1 can be calculated as follows:

$$\theta_1 = tan^{-1}\left(\frac{p_y}{p_x}\right) \tag{11}$$

Since θ_1 is the rotational angle for the robot arm's base, the angle range is between -180° and 180°. The quadrant for θ_1 is identified to determine the sign for each trigonometric ratio in a given quadrant. Table II shows the signs for the trigonometric ratio in each quadrant.

TABLE II. SIGNS FOR THE TRIGONOMETRIC RATIO IN EACH QUADRANT

p_x	p_y	**Quadrant**	θ_1
+	+	1	θ_1
-	+	2	$\theta_1 + 180^\circ$
-	-	3	$\theta_1 - 180^\circ$
+	1	4	θ_1

The second process involves calculating the angles for joints 2, 3, and 4 (θ_2, θ_3, and θ_4). For obtaining the solution for θ_2, θ_3, and θ_4, the 3-dimensional (3D) space which is consisting of the x, y, and z coordinate axes, is simplified to a 2-dimensional (2D) surface. The x and y-axis are merged as a new axis and known as the r-axis, as shown in Fig. 7.

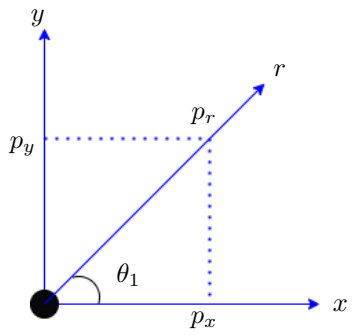


Fig. 7. Combination of x and y-Axis as r-Axis.

The x-coordinate and the y-coordinate for the end-effector are combined as r-coordinate using the Pythagorean equation as follows:

$$p_r{}^2 = p_x{}^2 + p_y{}^2 \tag{12}$$

$$p_r = \sqrt{p_x{}^2 + p_y{}^2} \tag{13}$$

Fig. 8 shows the projection of the r-z surface for the movement of the robot arm. The r_3 coordinates and the z_3 coordinates can be calculated as follows:

$$r_3 = p_r \tag{14}$$

$$z_3 = p_z - d_1 \tag{15}$$

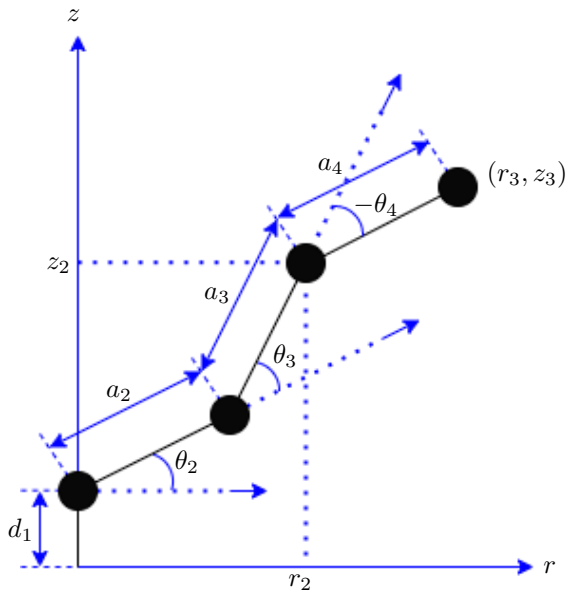


Fig. 8. Projection of the $r - z$ Surface for the Movement of the Robot Arm.

The range of the total sum of angles for θ_2, θ_3, and θ_4 are between $-90°$ and $90°$ as shown in Fig. 9.

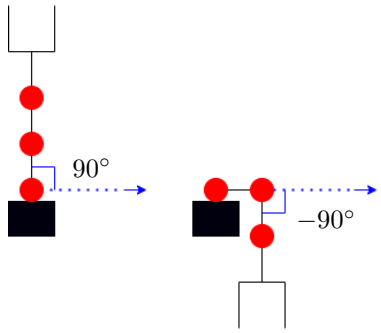


Fig. 9. Maximum and Minimum of the Sum of Angle for Joint 2, 3 and 4.

θ_2, θ_3, and θ_4 can be calculated using the following equations:

$$\phi = \theta_2 + \theta_3 + \theta_4 \tag{16}$$

$$r_2 = r_3 - a_4 \ cos \ \phi \tag{17}$$

$$z_2 = z_3 - a_4 \ sin \ \phi \tag{18}$$

$$cos \ \theta_3 = \frac{r_2{}^2 + z_2{}^2 - \left(a_2{}^2 + a_3{}^2\right)}{2 \ a_2 \ a_3} \tag{19}$$

$$\theta_3 = \pm \ cos^{-1}\left(\frac{r_2{}^2 + z_2{}^2 - \left(a_2{}^2 + a_3{}^2\right)}{2 \ a_2 \ a_3}\right) \tag{20}$$

$$r_2 = a_2 \ cos \ \theta_2 + a_3 \ cos \left(\theta_2 + \theta_3\right) \tag{21}$$

$$z_2 = a_2 \ sin \ \theta_2 + a_3 \ sin \left(\theta_2 + \theta_3\right) \tag{22}$$

$$r_2 = cos \ \theta_2 \left(a_2 + a_3 \ cos \ \theta_3\right) - sin \ \theta_2 \left(a_3 \ sin \ \theta_3\right) \tag{23}$$

$$z_2 = cos \ \theta_2 \left(a_3 \ sin \ \theta_3\right) + sin \ \theta_2 \left(a_2 + a_3 \ cos \ \theta_3\right) \tag{24}$$

$$cos \ \theta_2 = \frac{\left(a_2 + a_3 \ cos \ \theta_3\right) r_2 + \left(a_3 \ sin \ \theta_3\right) z_2}{r_2{}^2 + z_2{}^2} \tag{25}$$

$$sin \ \theta_2 = \frac{\left(a_2 + a_3 \ cos \ \theta_3\right) z_2 + \left(a_3 \ sin \ \theta_3\right) r_2}{r_2{}^2 + z_2{}^2} \tag{26}$$

$$\theta_2 = tan^{-1}\left(\frac{sin \ \theta_2}{cos \ \theta_2}\right) \tag{27}$$

$$\theta_4 = \phi - \left(\theta_2 + \theta_3\right) \tag{28}$$

There are two possible configurations for the robot arm, namely “elbow up” and “elbow down”. Both configurations are considered in the calculation of θ_2, θ_3, and θ_4.

The angle range for θ_2, θ_3, and θ_4 is between $-90°$ and $90°$, respectively. Based on the configuration of the robot arm in ROS, the angle range for θ_2 becomes $0°$ and $180°$ and the angle range for θ_3 becomes $-180°$ and $0°$. The angle limit for each joint in ROS ranges between $-90°$ and $90°$. Thus, the angle range for θ_2 to be considered in kinematic modeling is between 0 and $90°$, while for θ_3 is between $-90°$ and $0°$.

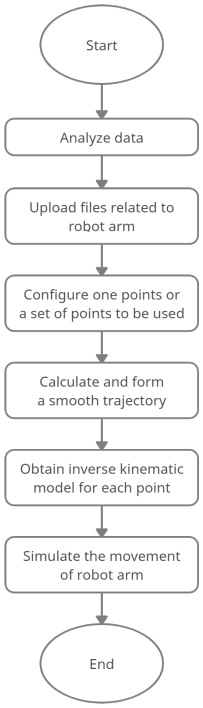


Fig. 10. Simulation Procedures for Trajectory Planning and Waypoint Tracking.

D. Trajectory Planning and Waypoint Tracking

The simulation of trajectory planning and waypoint tracking is done using MATLAB software with the help of Robotics System Toolbox, Simulink and Simscape Multibody. Fig. 10 shows the simulation procedure for trajectory planning and waypoint tracking. The first step to be completed is to analyze the data using MATLAB software. This procedure aims to identify and overcome the dependencies of the simulation files. Next, the algorithm includes uploading files related to the robot arm into MATLAB software. After that, the algorithm configures one point or a set of points to be used. The points are obtained from the modeling of the robot arm and incorporated into the algorithm. This step is very important to determine the similarity with the actual robot arm.

The next step is to calculate and form a smooth trajectory. This process runs the trajectory process and uses the points configured in the previous step. Trajectory planning is one of the critical processes in robots, where it involves the smooth movement of the end-effector from the initial position to the target position [18]. When an arbitrary endpoint is given, the algorithm computes the optimal feasible path for the robot arm's movement based on the kinematic constraints.

The final process involves performing inverse kinematic of the points found on the robot arm workspace. There are six values in the weight vector. The first three values are for rotational purposes, and the last three values are for transition purposes. Then, the algorithm configures a number greater than the number of points previously configured to ensure the robot arm's smooth movement. After that, it uses a kinematic solver for each end-effector position and uses the previous configuration to make an initial guess. After completing all the processes, the movement of the robot arm is simulated.

III. Results and Discussion

A. Modeling of Forward Kinematic

Fig. 11 shows the flowchart of the forward kinematic modeling algorithm. D-H parameters are included in the developed algorithm to obtain the end-effector's coordinate of the robot arm, and the transformation matrix is calculated to obtain the equations for the end-effector's coordinates. Next, random values of the four joint angles are generated by using the "randi" function. This function is able to generate uniformly distributed random integers from the specified interval. Finally, the end-effector's coordinates are calculated by inserting the random values for the four joint angles into the transformation matrix equations. Table III shows the results of the forward kinematic modeling developed.

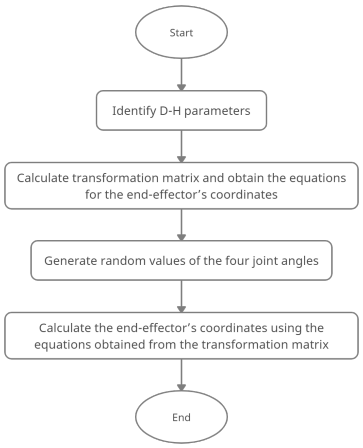


Fig. 11. Flowchart of Forward Kinematic Modeling Algorithm.

TABLE III. Results of Forward Kinematic Modeling

Case	Joint Angles (°)				End-Effector's Coordinates		
	θ_1	θ_2	θ_3	θ_4	p_x	p_y	p_z
1	103	15	-5	21	-0.1489	0.3508	0.115
2	56	3	-13	79	0.1716	0.2544	0.1531
3	65	68	-23	-20	0.1185	0.2542	0.3347
4	-22	34	-21	53	0.2804	-0.1133	0.2733
5	48	8	-65	32	0.2125	0.236	-0.0963
6	11	70	-8	-67	0.2512	0.0488	0.2966
7	166	42	-35	-40	-0.3407	0.0849	0.0918
8	82	38	-58	36	0.0632	0.3583	0.1246
9	-158	56	-79	14	-0.3158	-0.1276	0.0965
10	-54	75	-37	9	0.1465	-0.2017	0.3691

B. Modeling of Inverse Kinematic

Fig. 12 shows the flowchart of the inverse kinematic modeling algorithm. The coordinates of the end-effector are entered into the algorithm to obtain the joint angles that achieve a particular end-effector position. Next, the angle value for joint 1 is calculated, followed by calculating the angle values for

joints 2, 3, and 4. Finally, all solution sets for the joint angles are obtained.

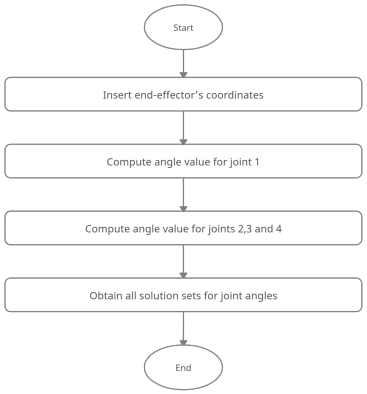


Fig. 12. Flowchart of Inverse Kinematic Modeling Algorithm.

Table IV shows the set of solutions for joint angles in integers obtained using MATLAB software. Referring to case 1 from the results of the forward kinematic modeling in Table V, a total of 66 sets of solutions can be obtained for the end-effector's coordinates ($p_x = -0.0834$, $p_y = 0.3611$ and $p_z = 0.1744$) including the desired solution set, which is the 65^{th} set of solution ($\theta_1 = 103°$, $\theta_2 = 15°$, $\theta_3 = -5°$ and $\theta_4 = 21°$).

TABLE IV. SET OF SOLUTIONS FOR JOINT ANGLES IN THE FORM OF INTEGER

Solution	Joint Angles (°)			
	θ_1	θ_2	θ_3	θ_4
1:	103	28	-2	-28
2:	103	38	-20	-19
3:	103	26	2	-28
4:	103	39	-24	-15
5:	103	24	5	-28
6:	103	40	-27	-13
7:	103	23	7	-28
8:	103	41	-29	-10
9:	103	21	9	-27
10:	103	41	-31	-8
11:	103	20	10	-26
12:	103	42	-32	-5
13:	103	19	12	-25
14:	103	42	-34	-3
15:	103	18	13	-25
16:	103	42	-35	-1
17:	103	17	14	-24
18:	103	42	-36	1
19:	103	16	14	-22
20:	103	42	-36	2
21:	103	15	15	-21
22:	103	42	-37	4
23:	103	14	16	-20
24:	103	42	-38	6
25:	103	14	16	-19
26:	103	41	-38	8
27:	103	13	16	-18
28:	103	41	-38	9
29:	103	12	17	-16
30:	103	41	-39	11
31:	103	12	17	-15
32:	103	40	-39	13
33:	103	11	17	-13

Solution	Joint Angles (°)			
	θ_1	θ_2	θ_3	θ_4
34:	103	40	-39	14
35:	103	11	17	-12
36:	103	39	-39	16
37:	103	11	17	-10
38:	103	39	-39	17
39:	103	10	16	-9
40:	103	38	-38	18
41:	103	10	16	-7
42:	103	37	-38	20
43:	103	10	15	-5
44:	103	37	-37	21
45:	103	10	15	-3
46:	103	36	-37	22
47:	103	9	14	-1
48:	103	35	-36	23
49:	103	9	13	0
50:	103	34	-35	24
51:	103	9	12	2
52:	103	33	-34	25
53:	103	10	11	5
54:	103	32	-33	26
55:	103	10	9	7
56:	103	31	-31	27
57:	103	10	8	9
58:	103	29	-30	27
59:	103	11	6	11
60:	103	28	-28	28
61:	103	12	3	14
62:	103	26	-25	28
63:	103	13	0	17
64:	103	24	-22	28
65:	103	15	-5	21
66:	103	21	-17	27

To obtain more accurate angles with more decimal points, the number of solution sets increases. The number of solution sets increases by about 10 times with the increase of one decimal point for the joint angles, as shown in Table V.

TABLE V. RESULTS OF INVERSE KINEMATIC MODELING

Case	End-Effector's Coordinates			Joint Angles (°)				Number of Solution Sets		
	p_x	p_y	p_z	θ_1	θ_2	θ_3	θ_4	0 DP	1 DP	2 DP
1	-0.0833	0.361	0.1744	103	15	-5	21	66	668	6672
2	0.1716	0.2544	0.1531	56	3	-13	79	179	1769	17686
3	0.1185	0.2542	0.3347	65	68	-23	-20	76	762	7616
4	0.2804	-0.1133	0.2733	-22	34	-21	53	134	1346	13454
5	0.2125	0.236	-0.0963	48	8	-65	32	132	1318	13174
6	0.2512	0.0488	0.2966	11	70	-8	-67	147	1462	14617
7	-0.3407	0.0849	0.0918	166	42	-35	-40	156	1546	15468
8	0.0632	0.3583	0.1246	82	38	-58	36	120	1192	11902
9	-0.3158	-0.1276	0.0965	-158	56	-79	14	176	1752	17518
10	0.1465	-0.2017	0.3691	-54	75	-37	9	64	630	6310

C. Correlation Analysis of Kinematic Modeling

Referring to the forward and inverse kinematic modeling results, as shown in Tables III and V, the solution is precisely the same when comparing the joint angles from the forward and inverse kinematic. The same solution set of joint angles could be obtained using the inverse kinematic model, as applied in the forward kinematic model. Thus, the correlation between the developed forward and inverse kinematic model is said to be accurate.

D. Performance of End-Effector Coordinates

Fig. 13 and 14 display the example of simulation results of trajectory planning and waypoint tracking for the set of solutions for the joint angles. The simulation results show that the solution set of joint angles which produce the most feasible trajectory are $\theta_1 = 103°$, $\theta_2 = 42°$, $\theta_3 = -35°$ and $\theta_4 = 0°$. Furthermore, Table VI depicts the comparison of the end-effector's coordinates obtained from the modeling process in MATLAB and output from ROS. The most significant errors occurred in case 1 and case 5. Errors occur between the end-effector's coordinates obtained from the results of modeling and output from ROS due to the rounding calculations that unavoidable in modeling and the kinematic constraints on the simulated robot arm in ROS as follows:

1) Joint Bounds:
   This constraint is satisfied if the robot configuration vector maintains all joint positions within the bounds specified.
2) Cartesian bounds:
   This constraint is satisfied if the end-effector origin's position relative to the target frame remains within the bounds specified.
3) Orientation target:
   This constraint requires the end-effector orientation to match a target orientation within an angular tolerance in any direction. The target orientation is specified relative to the body frame of the reference body.
4) Pose target:
   This constraint requires the end-effector's pose to match a target pose within a distance and angular tolerance in any direction. The target pose is specified relative to the body frame of the reference body.
5) Position target:
   This constraint requires the end-effector position to match a target position within a distance tolerance in

any direction. The target position is specified relative to the body frame of the reference body.

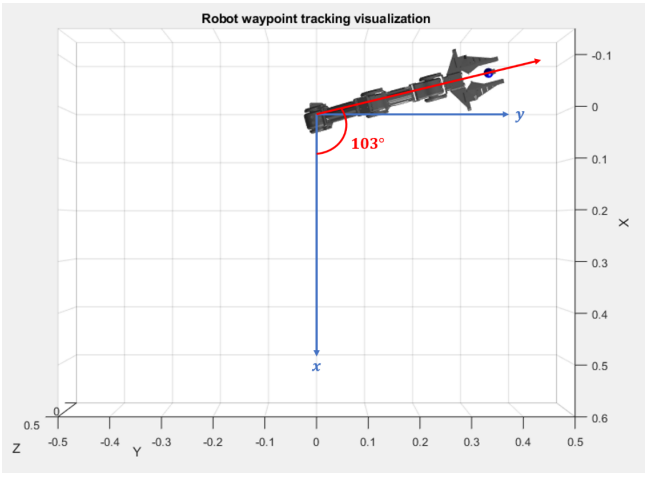


Fig. 13. Simulation Results of Trajectory Planning and Waypoint Tracking for Joint Angle 1.

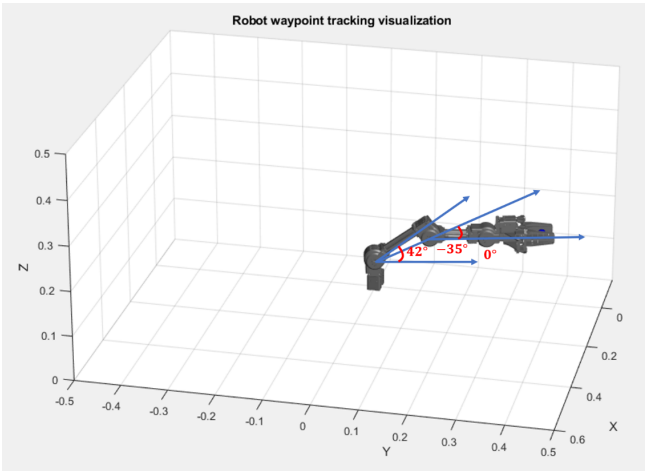


Fig. 14. Simulation Results of Trajectory Planning and Waypoint Tracking for Joint Angles 2, 3 and 4.

TABLE VI. PERFORMANCE OF END-EFFECTOR COORDINATES FROM MODELING AND ROS.

Case	End-Effector's Coordinates						Error (%)		
	Modeling			Output from ROS					
	px	py	pz	px	py	pz	px	py	pz
1	-0.0833	0.361	0.1744	-0.073	0.366	0.1784	14.25	1.34	2.24
2	0.1716	0.2544	0.1531	0.178	0.246	0.155	3.6	3.41	1.23
3	0.1185	0.2542	0.3347	0.127	0.247	0.327	6.69	2.91	2.35
4	0.2804	-0.1133	0.2733	0.282	-0.109	0.271	0.57	3.94	0.85
5	0.2125	0.236	-0.0963	0.221	0.232	-0.087	3.85	1.72	10.69
6	0.2512	0.0488	0.2966	0.257	0.048	0.287	2.26	1.67	3.34
7	-0.3407	0.0849	0.0918	-0.318	0.082	0.091	7.14	3.54	0.88
8	0.0632	0.3583	0.1246	0.061	0.35	0.129	3.61	2.37	3.41
9	-0.3158	-0.1276	0.0965	-0.294	-0.124	0.101	7.41	2.9	4.46
10	0.1465	-0.2017	0.3691	0.153	-0.194	0.362	4.25	3.97	1.96

IV. CONCLUSION

This study successfully modeled both the forward and inverse kinematic model for the 4-DoF articulated robot arm. Correlation between the forward and inverse kinematic model is analyzed, and the performance of the kinematic model is evaluated on the simulated robot arm. The developed kinematic model serves as a theoretical framework for further research, for instance robot arm trajectory planning, and structural optimization of robot design.

ACKNOWLEDGMENT

This work was financially supported by Universiti Kebangsaan Malaysia (grant no. GGPM-2019-051 and GUP-2018-103).

REFERENCES

[1] G. Bao, S. Liu, and H. Zhao, "Kinematics simulation of 4 DOF manipulator," vol. 123, pp. 400–408, 2017.

[2] Z. Chao, F. Wang, C. Zhang, and H. Li, "Inverse kinematics solution and verification of 4-DOF hydraulic manipulator," Journal of Physics: Conference Series, vol. 916, no. 1, 2017.

[3] T. Dewi, S. Nurmaini, P. Risma, Y. Oktarina, and M. Roriz, "Inverse kinematic analysis of 4 DOF pick and place arm robot manipulator using fuzzy logic controller," Int. Journal of Electrical and Computer Engineering, vol. 10, no. 2, 2020.

[4] S. Gómez, G. Sánchez, J. Zarama, M. C. Ramos, J. E. Alcántar, A. Nuñez, J. Torres, S. Santana, and F. Nájera, "Design of a 4-Dof robot manipulator with optimized algorithm for inverse kinematics," Int. Journal of Mechanical and Mechatronics Engineering, vol. 9, no. 6, pp. 929–934, 2016.

[5] A. A. Mohammed, and M. Sunar, "Kinematics modeling of a 4-DOF robotic arm," in Proceedings of the 2015 Int. Conf. on Control, Automation and Robotics, pp. 87—91, 2015.

[6] A. Novitarini, Y. Aniroh, D. Y. Anshori, and S. Budiprayitno, "A closed-form solution of inverse kinematic for 4 DOF tetrix manipulator robot," in Proceedings of the 2017 Int. Conf. on Advanced Mechatronics, Intelligent Manufacture, and Industrial Automation, pp. 25—29, 2017.

[7] L. Rónai, and T. Szabó, "Kinematical investigation and regulation of a 4DOF model robot," Acta Mechanica Slovaca, vol. 20, no. 3, pp. 50–56, 2016.

[8] A. Singh, and A. Singla, "Kinematic Modeling of Robotic Manipulators," in Proceedings of the 2017 National Academy of Sciences India Section A - Physical Sciences, pp. 303—319, 2017.

[9] T. Singh, P. Suresh, and S. Chandan, "Forward and inverse kinematic analysis of robotic manipulators," Int. Research Journal of Engineering and Technology, vol. 4, no. 2, pp. 1459–1469, 2017.

[10] A. El-Sherbiny, M. Elhosseini, and A. Haikal, "A comparative study of soft computing methods to solve inverse kinematics problem," Ain Shams Engineering Journal, vol. 9, no. 4, pp. 2535–2548, 2018.

[11] P. Jha, "Inverse kinematic analysis of robot manipulators," India: National Institute of Technology Rourkela, 2015.

[12] J. M. McCarthy, and G. S. Soh, "Geometric design of linkages," New York: Springer, 2010.

[13] L. Luthsamy, H.F. AL-Qrimli, S.Shazzana Wan Taha, and N.A. Raj, "Design and control of an anthropomorphic robotic arm," Journal of Industrial Engineering Research, vol. 2, no. 1, pp. 1–8, 2016.

[14] I. Al-Naimi, "Introduction to robot manipulators," Robotics and Automation, 2018.

[15] P. Corke, "Robotics, vision and control: Fundamental algorithms in MATLAB," 2nd ed. Switzerland: Springer, 2017.

[16] S. Singh, and E. Singla, "Service arms with unconventional robotic parameters for intricate workstations: Optimal number and dimensional synthesis," Journal of Robotics, pp. 1—11, 2016.

[17] M. v. Hoeij, "Closed form solutions," Florida: Florida State University, 2017.

[18] Robotis, "OpenMANIPULATOR-X," Available online: https://emanual.robotis.com [accessed on 18 May 2021].