Model and simulation of arm robot with 5 degrees of freedom using MATLAB

M Sabri1*, Rahmad Fauzi2, M S Fajar2, Geubrina H S1, Faris A M Sabri1

1Computational & Experimental System Mechanics Research Centre, Mechanical Engineering Department Universitas Sumatera Utara, Medan, Indonesia
2Electrical Engineering Department, Universitas Sumatera Utara, Medan, Indonesia

*Email: m.sabri@usu.ac.id

Abstract. The arm robot with five degrees of freedom is commonly used for industrial or education. For Industrial uses, robot are used for the main step to make the production time more efficient and it can make a product with a good quality. Considering the benefits of arm robot with five degrees in industrial, the robot simulation is used to know the robot torque that will improve and optimizing the movement of the arm robot so the robot can help industry to produce in effective and efficient way. In this paper, the arm robot with five degrees of freedom will analyze the kinematic forward movement and inverse kinematic of the Robot simulation. Interactive simulation system is created using MATLAB software, which is one of the most widely used platforms for modeling and simulating robot systems. Finally, the simulation results verify the kinematic effectiveness in the robot movement and the robot torque according to the robot motion.

1. Introduction

Robotics is an interdisciplinary branch of science and engineering that involve mechanical engineering, electrical engineering, computer science, and others. Robotics is concerned with the design, construction, operation, use of robots and computer systems for their control, sensory feedback and information processing. A robot is a physical machine that has a motor, sensor and controller. All you have to do is program them to do physical tasks (eg lift, maintain machines, paint, etc.) and they do these tasks independently.

Robot is widely used in industrial production plant and processing as the first step to make a product with a good quality. In an arm robot, there are joints where the joints use a motor to drive the joint, this robot simulation consists of joints that have a major impact on the kinematic movement of the robot.

The simulation system is made visually clear and highly interactive, which can simulate various physical systems in the robot. Compared with the physical experiment, it is lower in cost, higher reliability, and is very appropriate for industrial education and research as a measure to save money. Currently, simulation has been recognized as an important tool in the world of robotics and technology [5]. A large number of robotics simulation software uses the MATLAB application where the MATLAB application provides a platform for robotic systems, and has been used extensively. In this paper, an interactive simulation of a robot system will be developed using a platform that has been provided in MATLAB to obtain a good model in robot kinematics.

In a simulation system with the MATLAB platform, it is very easy to monitor the robot's kinematics and the torque used in each joint robot. The robot simulation system will also be modeled using Simulink in MATLAB with the MATLAB robotics library platform that is already available for simulating an arm robot with 5 degrees of freedom.
2. Robot arm with 5 degrees of freedom

Robot arm of 5 degrees of freedom is commonly used for both education and industrial uses. This robot works with 5 motors for the joint, in the following picture you can see the shape of the robot with 5 degrees of freedom in this simulation.

![Figure 1. Robot arm](image)

In general, a robot with 5 degrees of freedom can also be described in terms of coordinates as shown below.

![Figure 2. Robot coordinates](image)

The geometric structure of the 5 degrees freedom robot is similar to the human arm. Arm length and all five joints have been adapted for the needs of industrial robots in general but are functionally similar to the working principle of the human arm, which can be seen in the following explanation:

![Figure 3. Robot working principle](image)

- Joint 1: waist
- Joint 2: the shoulder
- Joint 3: the elbow
- Joint 4: wrist up and down movement
- Gripper: robot hand part (end-effector)
3. Model and simulation

The model created in this simulation uses Simulink in MATLAB with the Robotics System Toolbox library. The Simulink form that has been formed can be seen in the following image.

![Figure 4. The Simulink form](image)

From this figure, it can be seen that the first step in simulating is to create a path for the movement of the robot where the path is made by determining the points that will be combined and form a path. In this simulation, we will use a path for the movement of the robot that has been combined horizontally and vertically and uses the ball as an object that carried by the robot from one point to another as shown below.

![Figure 5. The movement](image)

The robot kinematics is represented by a set of algebraic constraint equations on the joint configuration vector \( q = [q_1, q_2, ..., q_n]^T \in \mathbb{R}^n \), where \( n \) is the number of joints. Limitation of the joint motion range is not taken into account in this paper. Let us define the residual of \( i \)th constraint equation \( e_i \in \mathbb{R}^3 \) as

\[
e_i(q) \equiv \begin{cases} 
    d p_i - p_i(q) & \text{(position constraint)} \\
    \alpha(d R_i R_i^T(q)) & \text{(orientation constraint)}
\end{cases}
\]

(1)

Where \( p_i \in \mathbb{R}^3 \) and \( d p_i \in \mathbb{R}^3 \) are the position and its desired value of the link of interest, respectively, and \( R_i \in SO(3) \) and \( d R_i \in SO(3) \) are the orientation and its desired value of the link of interest, respectively. In addition, \( \alpha(R) \in \mathbb{R}^3 \) for an arbitrary \( R \in SO(3) \) means the equivalent angle-axis vector, the way to compute which is described in Appendix A. The constraints include ones due to tasks and the others due to mechanisms such as closed loops, while they are not distinguished hereafter. Let the number of all the constraints \( 3m \) and define the residual vector \( e(q) \in \mathbb{R}^{3m} \) as

\[
e(q) \equiv [e_1^T(q) e_2^T(q) ... e_m^T(q)]^T
\]

(2)
IK comes down to solve the following nonlinear equations:

\[ e(q) = 0 \]  (3)

The conventional IK that is based on the NR method tries to reach \( q = q^* \) which satisfies Eq. (3), from a certain initial value \( q_0 \) by the following updating rule:

\[ q_{k+1} = q_k - \nabla e(q_k)^{-1} e_k \]  (4)

where \( k \) is the iteration step, and \( e_k \equiv e(q_k) \). The basic Jacobian matrix \( J(q) \) can also replace \( \nabla e \) as

\[ \nabla e(q_k) \approx -J_k \]  (5)

Where \( J_k \equiv J(q_k) \). The above formulation implies the following three assumptions:

1) The number of constraints and the DOF of the robot are the same, i.e., \( n = 3 \) m.
2) \( J_k \) is always regular.
3) Equation (3) can be solved.

Even if one of them is violated, the iteration will decrease. To discuss an IK solver that will not diminish in unsolvable cases, let's replace the original equation (3) with the following minimization (QP) problem:

\[ E(q) \equiv \frac{1}{2} e^T W_E e \rightarrow \text{min.} \]  (QP)

Where \( W_E = \text{diag} \{w_{E,i}\} (w_{E,i} > 0 \ for \ \forall i = 1 \sim 3m) \) is the weighted matrix to reflect the priority level of each constraint / constraint and also to absorb the physical differences in metrics between each constraint. Note that this is not an equivalent substitute but implies that safety takes priority over finding the correct solution by avoiding abnormal calculations for practical applications.

Here, we point out the following important facts. When the gradient method is applied for the minimization problem, \( \nabla E = 0^T \) is satisfied at the convergent point. From the definition of \( E(q) \),

\[ \nabla E = e^T W_E \nabla e \approx -e^T W_E J, \]  (6)

and thus, the convergent point is the singular point if not the solution of (3). Therefore, the singularity problem cannot be avoided if the original equation cannot be solved.

The NR method is still available for problems (QP) in a slightly different form than (4). By distinguishing (6), so we get

\[ \nabla^2 E = \nabla e^T W_E \nabla e + \sum_{i=1}^{n} \frac{\partial^2 e}{\partial q_i} W_E e \]  (7)

where (5) is applied. \( \frac{\partial J}{\partial q_i} \) is referred as the Hessian manipulator [1]. The NR method upgrade rule based on (7) is given as

\[ q_{k+1} = q_k + (J_k^T W_E J_k - \sum_{i=1}^{n} \frac{\partial J_k}{\partial q_i} W_E e_k)^{-1} g_k \]  (8)

\[ g_k \equiv J_k^T W_E e_k \]  (9)

This method was proposed by Deo and Walker. However, the Hessian manipulator requires large computational costs, and moreover, it does not guarantee a decrease in \( E \) because the matrix coefficient
is not necessarily positive so it does not converge globally. Now, our aim is not to know the exact curvature of $E$ but to know the descent direction of $E$. Hence, it is better to use a corresponding positive matrix instead of $\nabla^2 E$. If the last term of (7) is omitted, it is equivalent to the Gaussian–Newton (GN) method where the update rule is defined as

$$q_{k+1} = q_k + (J_k^T W W J_k)^{-1} g_k$$ (10)

which is equivalent with the usage of the weighted norm-minimizing generalized inverse matrix of $J_k$ rather than $\nabla e(q_k)^{-1}$ in (4). Although the updating rule (10) is not available if $J_k$ is not full, at the singular points, it is resolved if an inverted MP matrix is used. The problem is that the coefficient matrix becomes ill-posed, and the computation becomes unstable in the vicinity of the singular points. A method to use (8) only near the singular points and (10) in the other area is proposed. It requires to judge if the robot configuration is near the singular points every step of iteration, and thus, is disadvantageous in terms of computational cost.

$$q_{k+1} = q_k + H_k^{-1} g_k$$ (11)

$$H_k \equiv J_k^T W W J_k + W_N$$ (12)

Where $W_N = \text{diag}\{w_{N,i}\}(w_{N,i} > 0 \text{ untuk } i = 1 \sim n)$ is called the damping factor. The IK solution using the LM method was proposed by Goldenberg et al. The LM method does not require a Hessian manipulator calculation and guarantees a decrease in $E$ since $H_k$ is always regular and positive, and the increment term of (11) is always oriented to the descending direction. This is the simplest form of Tikhonov’s regulation and is interpreted that the following mixed minimization problems is solved in each step of iterations:

$$\frac{1}{2} r_k^T W r_k + \frac{1}{2} \Delta q_k^T W_N \Delta q_k \rightarrow \text{min.}$$ (13)

Where $\Delta q_k \equiv q_{k+1} - q_k$, and $r_k \equiv e_k - J_k \Delta q_k$. Hence, $q_k$ converges to a certain value even in redundant cases, taking the minimum deviation. To summarize this section, the minimization approach with the LM method is more preferable than the other methods from the viewpoint of numerical robustness in unsolvable cases, which necessarily causes a convergence to singular points, even on redundant robots.

In the LM method, the choice of the $W_N$ damping factor greatly affects the convergence performance. Many methods for selecting $W_N$ have been proposed in such a way that they use a constant value of the manipulability measure of emphasis of the singular coefficient of the minimum matrix value below the threshold suppression of the number of conditions of the coefficient matrix below the explicit threshold of adding the adjoin vector configuration of the remaining priority constraints. high and so on. The least squares method error is damped which defines the damping factor as

$$W_N = \lambda E_k 1$$ (14)

where $\lambda$ is a constant coefficient and has no significant effect; $\lambda = 1$ is acceptable, $E_k \equiv E(q_k)$, and 1 is the $n \times n$ identity matrix. If the original equation (3) can be solved, $E_k$ quadratically converges to 0 while $q_k$ converges with the solution, and thus, the iteration is expected to be superlinearly convergent. However, if the solution is near one from the singular point, the matrix coefficients will be numerically incorrect as they approach the solution. We propose to modify it as

$$W_N = E_k 1 + \bar{W}_N$$ (15)
where \( W_N = \text{diag}\{\bar{w}_{N,i}\} \) and \((\bar{w}_{N,i} > 0 \text{ for } \forall i = 1 \sim n)\) are small biasing values. Suppose all \( \bar{w}_{N,i} \) equal with \( \bar{w}_N \), that is, \( W_N = \bar{w}_N 1 \) for simplicity, and the singular value decomposition is conducted to \( W_E^{1/2} J_k \) as

\[
W_E^{1/2} J_k = U \Sigma V^T \tag{16}
\]

where \( W_E^{1/2} \) means \( \text{diag}\{\sqrt{W_N,i}\} \), \( U \) and \( V \) are orthonormal matrices, and \( \Sigma = \text{diag}\{\sigma_i\} \) is a quadratic matrix in which the singular values are diagonally assigned in descent order, that is, \( \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \). Since

\[
H_k = V \sum U^T U \sum V^T + (E_k + \bar{w}_N) 1 \tag{17}
\]

the number of conditions \( H_k \), \( \kappa \) is

\[
\kappa = \frac{\sigma_1^2 + E_k + \bar{w}_N}{\sigma_n^2 + E_k + \bar{w}_N} \tag{18}
\]

It states the following facts.

1. If Eq. (3) is solvable and the solution is far from the singular points, \( W_N \) quadratically converges to \( \bar{w}_N \). If \( \bar{w}_N \) is sufficiently small, the iteration is superlinearly convergent as well as the Chan et al’s method.

2. If Eq. (3) is solvable and the solution is near one of the singular points, \( \kappa \) converges to \( \frac{\sigma_1^2 + \bar{w}_N}{\bar{w}_N} \). Therefore, \( \bar{w}_N \) is necessary to prevent \( H_k \) from degeneracy.

3. If Eq. (3) is unsolvable, \( \kappa \) becomes close to 1 as \( E_k \) increases. It suggests an interesting fact that the larger the norm of \( e_k \) during the iteration, the more stable the computation. This property is also expected from an equation \( \|H_k^{-1} g_k\| \approx \frac{1}{\|e_k\|} \).

4. Result

The results of the movement with the torque obtained using the simulation on the MATLAB can be seen in the graph as follows:

1. Robot movement graphics

To understand more about the movement graphics, we can see the path of robot arm in Figure 5. The graphics below explain the arm robot movement with path that we can see in Figure 5.
Figure 6 show the robot arm’s wrist movement, the graphic start to go up when the joint 1 is moving.

![Figure 7. Movement joint 2](image1)

![Figure 8. Movement joint 3](image2)

![Figure 9. Movement joint 4](image3)

Figure 7 until 9 shows the joints movement that can stabilizing the arm robot movement. So, the arm robot movement will be stable if the graphic of robot simulation looks like Figure 7 until 9.

![Figure 10. Movement gripper 5](image4)

The gripper of robot arm just has two movement, to grip and release. As we can see in the Figure 10 that explain the movement of robot arm’s gripper.
2. **Robot torque graphics**

![Figure 11. Torque joint 1](image)

![Figure 12. Torque joint 2](image)

![Figure 13. Torque joint 3](image)

![Figure 14. Torque joint 4](image)

![Figure 15. Torque gripper 5](image)

By knowing the torque of robot arm, we can make improvisation of the robot arm’s movement. We can verify the kinematic effectiveness and optimizing the torque for robot movement.
5. Conclusion
Robot simulation using MATLAB proved that the simulation help the robot to determine the movement by using torque to the joint’s kinematics or robot arm, so that it can optimize the torque used for robot movement. The simulation can be done by making a path for the arm robot and analyze the movement with MATLAB, with movement graphics and torque graphics as the output of the MATLAB application. From the result, we can conclude that if the torque of the robot can be optimized, the robot movement will be stable and can be improved. It will give a good profit to industry which use the arm robot in their production plant. Arm robot that has a stable and improved movement will finish the work faster with a lot of output rather that using the arm robot that still unstable. So the production in industry will be more effective and efficient.

Acknowledgement:
The data presented and analyzed in this report were collected during research “Model and Simulation of Arm Robot with 5 Degrees of Freedom Using MATLAB” the financial supported by TALENTA Universitas Sumatera Utara Research No. 751/UN5.1.R/PPM/2020.

References
[1] Hourtash A 2005 Proc. 2005 IEEE Int. Symp. Comput. Intell. Robot. Autom 169–174.
[2] Wisanuvej P et al. 2017 Master manipulator designed for highly articulated robotic instruments in single access surgery,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst 209–214.
[3] Patel S and Sobh T 2015 J. Intell. Robot. Syst. vol 77 no 3–4 547–570.
[4] Huang B, Ye M, Hu Y, Vandini A, Lee S-L and Yang G-Z 2018 IEEE Trans. Ind. Informat vol 14 no. 4 1776–1785 Apr. 2018.
[5] Buss S R and Kim J 2005 Selectively damped least squares for inverse kinematics J. Graph. Tools vol 10 no 3 37–49.
[6] Choi Y 2008 Trans. ASME, J. Dyn. Syst., Meas., Control vol 130 no 5 051009-1–051009-7.
[7] Leibrandt K et al. 2017 IEEE Robot. Autom. Lett. vol 2 no. 3 1704–1711 Jul. 2017.
[8] Zhou H, Lu Z, Liu C and Wang H 2016 IEEE International Conference on Signal and Image Processing (ICSIP 2016).
[9] Lenar’cic J and Thomas F 2013 Advances in Robot Kinematics: Theory and Applications. New York, NY, USA: Springer-Verlag.
[10] Nizar M, Munir E, Munawar E and Irvan 2018 IOP Conf. Ser: Journal of Physics 1116(5) 052045
[11] Haryani N, Harahap H, Taslim, Irvan 2020 IOP Conf. Ser: Mater. Sci. Eng. 801(1) 012051