International Journal of Academic Engineering Research (IJAER)
ISSN: 2643-9085
Vol. 10 Issue 4 April - 2026, Pages: 58-64

Design Of A 6-Dof Surgical Robot Manipulator With Mechanical Remote Center Of Motion Based On Inverse Kinematics

Hanafi Fadzillah*), Sumardi

Departement of Electrical Engineering, Engineering Faculty, Diponegoro University, Semarang, Indonesia
*) *Email: Hanafi.fadzillah@gmail.com*

Abstract: *Research around surgical robots in the medical world is popular because of their use in MIS (Minimal Invasive Surgery). This surgery allows the patient to undergo a small incision during surgery so that they do not experience excessive bleeding and recover quickly from hospitalization. Therefore, this research discusses the design of a 6-DOF (Degree of Freedom) robot manipulator with a mechanical RCM (Remote Center of Motion). Mechanical RCM allows the robot's rod end effector to move by maintaining a static point, which in the medical world is called the trocar point. With this RCM, the load on the robot control system can be reduced, and the trocar point obtained is fixed. Differential inverse kinematics that utilizes the Jacobian Matrix is used as a robot control system. Because the differential inverse kinematics nature is an iterative method, the trajectory planning algorithm is designed to integrate with the inverse kinematics method. The end effector position obtained through this control method has an average maximum error of 36 mm, with smooth trajectory results resembling a first-order system response.*

Keywords*: Differential Inverse Kinematics, Remote Center of Motion, Minimal Invasive Surgery, Trajectory Planning*

1. Introduction

The development of robotics technology has progressed rapidly to the medical world, to be precise in terms of surgery and surgery on humans. The surgical robot itself is a robot that helps improve the sensory characteristics of the surgeon and allows for more refined surgeries and MIS (minimally invasive surgery) procedures. [1]. Based on these hopes and opportunities, several companies have started to commercialize robots for surgery, one of the well-known companies is Intuitive Surgical Inc. The capabilities and results offered by this surgical robot are not to be underestimated, this is evidenced by the existence of surgical robots that are already operating in hospitals in developed countries such as the United States, Japan and Singapore. In Japan, this surgical robot technology began to be developed to realize the actual use of telesurgery or long-distance surgery between the main hospital and other hospitals. [2].

Behind these advantages, the use of surgical robots is still has several weaknesses that still have to be faced, namely the cost of acquisition, operation and maintenance of the robot itself which can be considered too expensive from the point of view of developing countries such as Indonesia. Based on a study conducted by Childers, each procedure using the surgical robot Intuitive Surgical Inc. is estimated to cost around 3568 dollars, of which 1866 dollars are for instrumentation and accessories, 1038 dollars for robotic systems, and 663 dollars for service contracts. In addition, an additional fee of 1701 dollars is also required for system maintenance costs in each procedure [3].

From these problems emerged new studies regarding the development of medical surgical robot technology, such as the Raven II surgical robot developed by the University of California. This robot has a 7 DOF configuration (6 DOF + grasp) and works by utilizing a teleoperation system. The teleoperation system consists of two main components, namely a component that acts as a master device controlled by the operator, and a component that acts as a slave device that performs actions according to orders given by the master device [4]. The Raven II robot slave device itself has a mechanical constraint that allows the robot to move the end effector with the remote center of motion [5]. This remote center of motion allows the end effector to enter the patient's body through a point called the trocar point. This trocar point is the main key in carrying out the MIS procedure. MIS is a surgical technique based on access to the internal cavity through several small incisions in the patient's body or so-called trocar points. The existence of this limitation causes the movement of the end effector to be limited to 4 Degree of Freedom (DOF) as shown in Figure 1 [6].

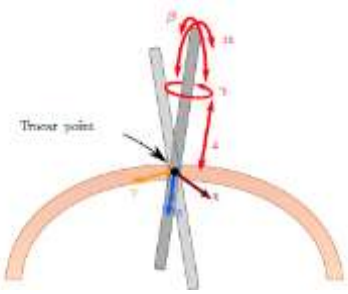


Figure 1. Movement of end effector with remote center of motion constraint

Based on above description, this Final Project discusses the design of a 6-DOF surgical robot manipulator with a remote center of mechanical motion based on inverse kinematics to test trajectory movements along with the level of precision and accuracy of end effector positions.

2. Method

International Journal of Academic Engineering Research (IJAER)
ISSN: 2643-9085
Vol. 10 Issue 4 April - 2026, Pages: 58-64

2.1. Hardware Design

The manipulator designed in this final project resembles the design of the Raven II arm robot, where this robot has 6 DOF (degrees of freedom) with one grasper as shown in Figure 2. [4]. This unique design is intended to achieve movement of the rod end effector with a remote center of motion. This remote center of motion allows the rod on the end effector to move while maintaining a point called the trocar point. Modified DH robot parameters can be seen in Table 1.

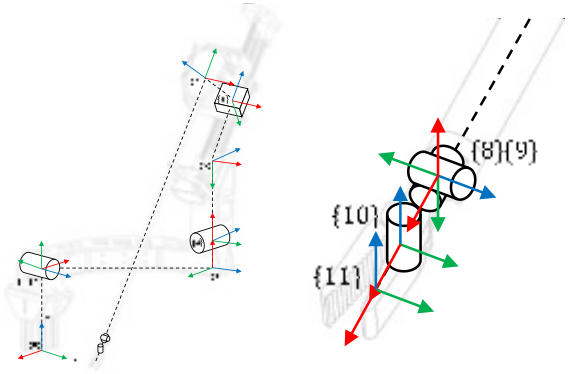


Figure 2. Diagram of the manipulator kinematics

Table 1. Modified DH parameter of the manipulator

j	α_{j-1} (°)	a_{j-1} (mm)	d_j (mm)	θ_j (°)	Type	Min	Max
1	0	0	148	180	Fixed	-	-
2	90	0	0	0	Revolute	0°	90°
3	0	240,21	179,33	90	Fixed	-	-
4	105	38.5	0	0	Revolute	−55°	55°
5	0	150,32	0	90	Fixed	-	-
6	52	0	90	0	Prismatic	39,37	188,37
7	90	0	125	0	Fixed	-	-
8	90	0	473,5	90	Revolute	−45°	225°
9	90	0	0	90	Revolute	0°	180°
10	90	9,5	0	0	Revolute	−90°	90°
11	0	7,5	0	0	Fixed	-	-

Based on Table 1, it can be seen that the robot is designed with 11 joints where 5 joints are fixed type, 5 joints are revolute type and 1 joint is prismatic type. Joints 2, 4 and 6 function to adjust the position of the end effector while joints 8, 9, 10 function to adjust the orientation of the end effector. This link configuration allows the robot to move following the constraints of the remote center of motion.

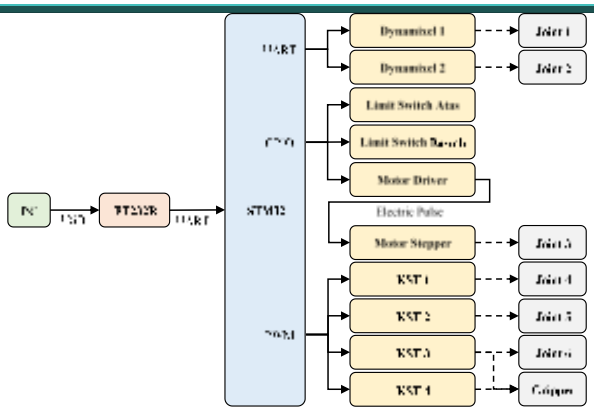


Figure 3. Manipulator electrical system configuration

The electric manipulator system is centered on a PC with a python language program to compute joint robot values as shown in Figure 3. The joint values obtained are transmitted to STM32 via FT232R to actuate actuators. Apart from that, there is also a Limit Switch to keep the movement of joint 3 from exceeding the required value.

2.2. Control Method Design

The inverse kinematics method used in this final project is the differential kinematics method, where this method will provide an equation connecting the velocity of the joint (the derivative of the joint value) and the linear velocity and angle of the end effector (the derivative of the pose of the end effector). [7]. This mapping will be represented by the Jacobian matrix [8].

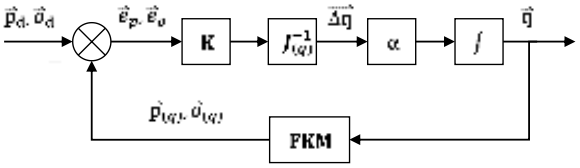


Figure 4. Block diagram of the differential inverse kinematics method

Simply put, the differential inverse kinematic method can be explained through the block diagram in Figure 4. In addition, this method is also a trajectory planning algorithm for the movement of the end effector robot. Because the nature of this method is an iterative method, the system will change the joint value little by little depending on the error and parameter update rate of the system until the expected final pose is reached.

Based on the flow chart in Figure 4, when the robot system is initiated, 2 kinds of input are first required, namely the expected robot pose ($\vec{p}_d$ dan $\vec{o}_d$) and the current robot pose ($\vec{p}_{(q)}$ dan $\vec{o}_{(q)}$). The current robot poses are obtained from the current forward kinematics. Then by comparing the expected pose and the current pose, an error pose will be ($\vec{e}_p$ and $\vec{e}_o$). error pose is given a gain to determine the performance and speed of the system to reach the expected set point. After obtaining the error pose and its gain value, proceed with the inverse kinematic

International Journal of Academic Engineering Research (IJAER)
ISSN: 2643-9085
Vol. 10 Issue 4 April - 2026, Pages: 58-64

calculation using the formula shown in Equation 1 where $\dot{q}$ is changes in joints value, J is jacobian matrix and $\dot{p}$ is changes in position [9].

$$\dot{q} = \begin{bmatrix} \dot{p}_e \\ \omega_e \end{bmatrix} = J^{-1}(q) \cdot \dot{p} \qquad (1)$$

The $\dot{q}$ matrix is a 6×1 matrix where each element represents a change in the value of the joint variable. Because matrix J is a square matrix, finding the inverse value of J is possible, while the $\dot{p}$ matrix is a 6×1 matrix where each element represents a change in the value of the end effector pose. After obtaining the $\dot{q}$ value, the data needs to be changed first, which was originally a change in the joint value to the joint value by multiplying it by α (update rate with a value range of 0 to 1) and adding up the previous joint value. After that, the joint value is sent to the robot to be actuated, then feedback is taken on the actualized joint value to carry out the forward kinematics process and repeat the inverse kinematic process. This process is carried out repeatedly so that the end effector pose value is below the specified error threshold. This algorithm also acts as a trajectory planning for the movement of the end effector. The parameters α and gain are the parameters that determine the movement of the end effector from one point to another. The smaller the value of α, the movement of the end effector will be slower, but smoother without jolting, while the gain parameter affects how fast the system will fix the errors that occur.

2.3. Software Design

The development of the inverse kinematics program is based on the block diagram in Figure 4, while Figure 5 is a flowchart for software design.

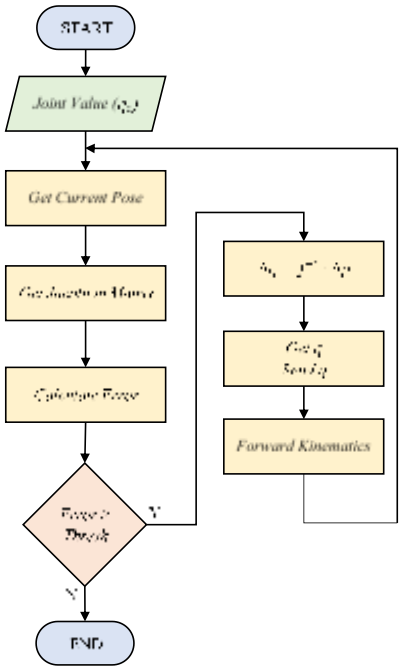


Figure 5. ***Differential inverse kinematics Flowchart***

The differential inverse kinematics method is an iterative system so this method will continue to repeat itself until the end effector pose approaches the expected set point with an error below the specified threshold. The Joint Value input is only done once, that is, during the initialization phase of the robot so that in subsequent iterations, the Get Current Pose process gets a new joint input from the forward kinematics process.

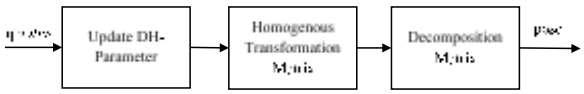


Figure 6. Block diagram of ***forward kinematics***

Forward Kinematics of the Manipulator can be explained through the Block Diagram in Figure 6. The joint value input (q) is used to update the DH parameter so that a new robot posture is obtained. The previously updated DH parameters are used to update the homogeneous transformation matrix. By decomposing the transformation matrix, we can know the pose of the robot in each frame, including the pose in the last frame, namely the pose of the end effector.

3. Result and Discussion

3.1. Position Control Test with Inverse Kinematic

The results of position control test with inverse kinematics on the X axis can be seen in Figure 7 and Figure 8. Figure 7 shows the error data of the actual position of the end effector against the set point. From Figure 7 it can be seen that the resulting error is still quite high with a maximum value of 43 mm and a minimum value of 9 mm. Figure 8 shows the boxplot of the inverse kinematics test on the X axis. The boxplot helps to see the distribution of data on errors that occur starting from the minimum value, quartile 1 (Q1), quartile 2 (Q2) or median, quartile 3 (Q3), and the maximum value.

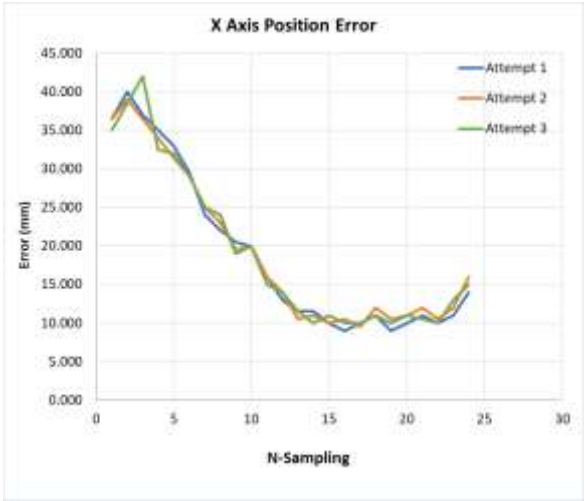


Figure 7. X axis position error

Based on Figure 7 and Figure 8, it can be concluded that the control of the end effector on the X axis has a poor level of accuracy but a high level of precision. This is because, in each data sampling the end effector position never reaches the expected set point value which is shown from the position error.

However, in each attempt the position of the end effector is always similar from one attempt to another.

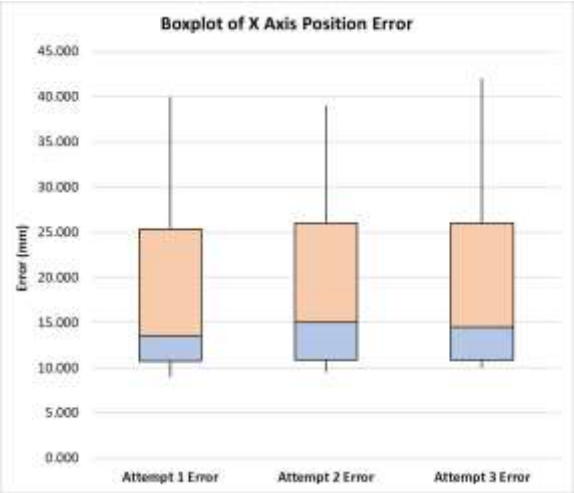


Figure 8. Boxplot of X axis position error

From each attempt, the MAE and RMSE values are obtained as follows,

$\text{MA}E_1 = 18.938$ $RMSE_1 = 21.580$
$MAE_2 = 19.146$ $RMSE_2 = 21.527$
$\text{MA}E_3 = 19.125$ $RMSE_3 = 21.675$

Of the three MAE and RMSE values, the average MAE was 19.09 and RMSE was 21.594. From these two values it can be concluded that the average error on the X axis is 19.06 mm. However, with the difference between the average MAE value and the average RMSE of around 2.5, it indicates that there is a deviation in value between one data sampling and the other.

The results of position control with inverse kinematics on the Y axis can be seen in Figure 9 and Figure 10.

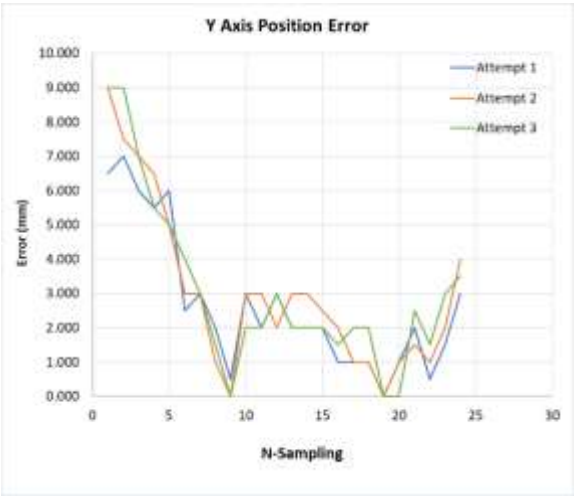


Figure 9. Y axis position error

Based on Figures 9 and 10, it can be concluded that the control of the end effector on the Y axis has a better level of accuracy and a fairly high level of precision when compared to the previous test. This is because, in each sampling data, the position of the end effector is close to the expected set point value. In addition, in each test the position of the end effector is always similar from one test to another.

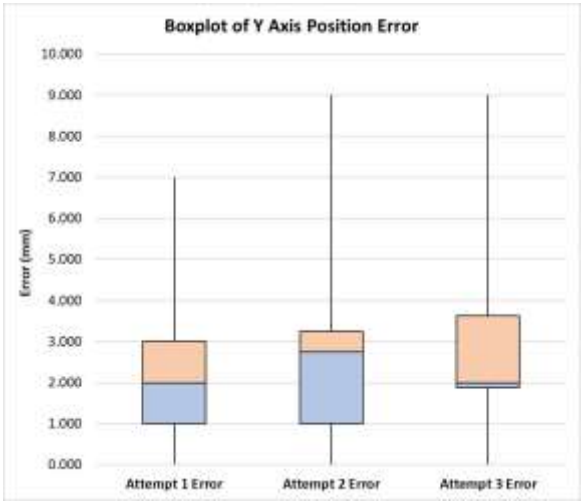


Figure 10. Boxplot of Y axis position error

From each attempt, the MAE and RMSE values are obtained as follows,

$\text{MA}E_1 = 2.667$ $RMSE_1 = 3.332$
$MAE_2 = 3.000$ $RMSE_2 = 3.813$
$\text{MA}E_3 = 3.042$ $RMSE_3 = 3.886$

Of the three MAE and RMSE values, the average MAE was 2,903 and RMSE was 3,677. From these two values it can be concluded that the average error on the Y axis is 2,903 mm. In addition, with the difference between the average MAE value and the average RMSE of around 0.7, it indicates that there is a slight deviation in value between one data sampling and the other.

The results of position control with inverse kinematics on the Z axis can be seen in Figure 11 and Figure 12.

Figure 11. Z axis position error

Based on Figures 11 and 12, it can be concluded that the control of the end effector on the Z axis has a poor level of accuracy but a fairly high level of precision similar to that of the X axis.

International Journal of Academic Engineering Research (IJAER)
ISSN: 2643-9085
Vol. 10 Issue 4 April - 2026, Pages: 58-64

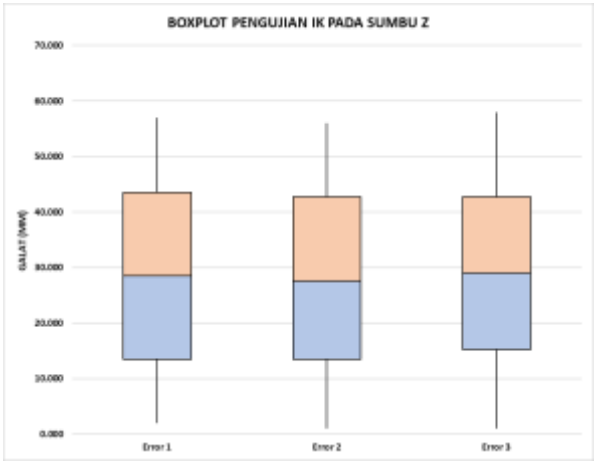


Figure 12. Boxplot of Z axis position error

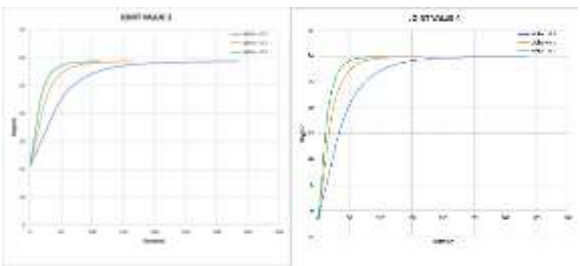
From each attempt, the MAE and RMSE values are obtained as follows,

$\mathrm{MAE}_1 = 28.708$ $RMSE_1 = 33.540$
$MAE_2 = 28.125$ $RMSE_2 = 32.832$
$\mathrm{MAE}_3 = 28.958$ $RMSE_3 = 33.555$

Of the three MAE and RMSE values, the average MAE was 28,597 and RMSE was 33,309. From these two values it can be concluded that the average error on the Z axis is 28,597 mm. In addition, with the difference between the average MAE value and the average RMSE of around 4.8, it indicates that there is a deviation in value between one data sampling and the other.

3.2. Trajectory Planning Test

Figure 13 shows a comparison of the joint value (q) with different alpha α values (update rates). It can be seen that each joint reaches the same value but with a different number of iterations. This is caused by the alpha value. When updating the joint value, the difference in the joint value (joint speed) is multiplied by the alpha value before adding up the joint value so that the joint value is always proportional to the joint speed. If the alpha value is 0.3, the change in joint value is 0.3 from the joint speed. This is the reason why the smaller the alpha value, the smaller the oscillation that occurs in the joint value. The alpha value serves to adjust the changes in the joint value in each iteration. As a result, more iterations are needed to achieve the expected joint value.



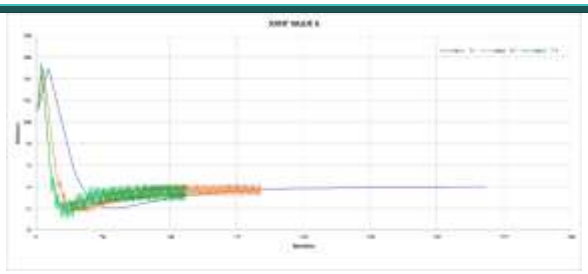


Figure 13. Joint value with alpha variations

Figure 14 shows a comparison of computational or simulated positions with different alpha α (update rate) values. From the Figure it can be seen that the positions on the X, Y, and Z axes reach the expected point, namely (30, 230, 30) but with a different number of iterations. As in the previous section, the difference in the number of iterations is affected by the alpha value. Then, the slightly oscillating Z-axis position is caused by changes in the oscillating value of joint 6 because the large value of joint 6 greatly affects the position on the Z-axis. The position also ends with the same value even though with a different number of iterations where a small alpha value will have the highest number of iterations. Both Figures 13 and 14 both form a first-order system response function which causes the movement of the end effector changes to be smooth.

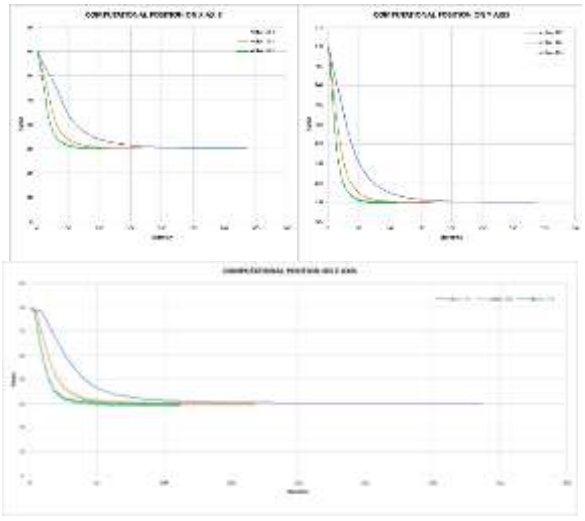


Figure 14. Computational position with alpha variations

From the trajectory planning test, it is obtained that the joint value is similar between the value sent from the computational results and the actual value that is actuated by the actuator. Therefore, the existence of positional errors that occur in position test is hypothesized to be caused by mechanical errors from the robot. This is because the computational position achieved is close to the given set point position, which is different from the actual position obtained. This position was obtained through forward kinematics which was built in python and the numpy library which incidentally is a language and library used for scientific computing. From there it can be inferred that the problem starts with a mechanical error that occurs due to differences in the actual robot framework

compared to the design robot framework, causing errors in robot modeling in computations. The causes of this mechanical error are several things, including errors that occur from 3D Printing because most robot frames have PLA+ filament material, changes in the shape of the robot frame because it has to support the robot's weight, and also inaccuracy in cutting and forming the robot frame. However, it cannot be denied that the cause of the position error is also caused by the development of the inverse kinematic algorithm.

3.3. Pyramid Pattern Drawing Test

The test for drawing a pyramid pattern is carried out in a working space that can be reached by the end effector robot. The position of the end effector will be moved towards 10 set point as shown in Table 2 to form a pyramid pattern. This test will analyze the level of accuracy and precision of the end effector in reaching the set point, as well as the trajectory planning algorithm's ability to draw lines.

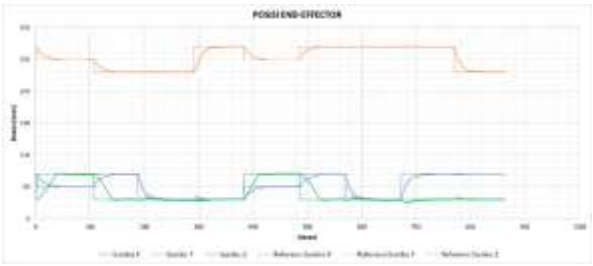
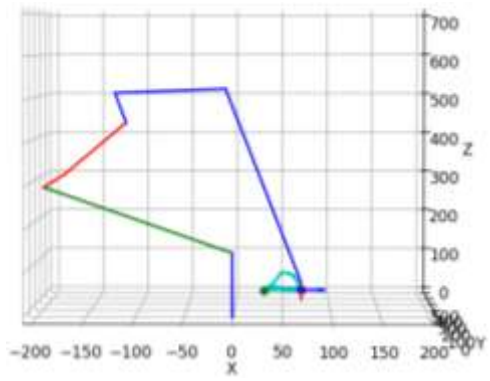
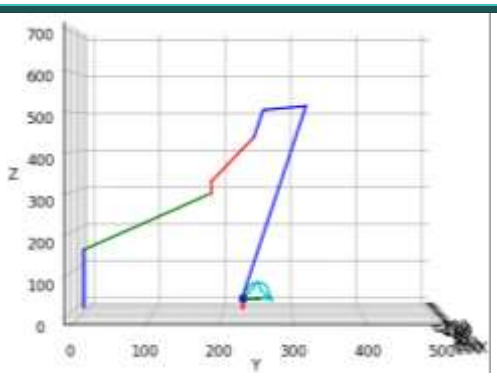


Figure 15. Computational position of the pyramid pattern drawing test

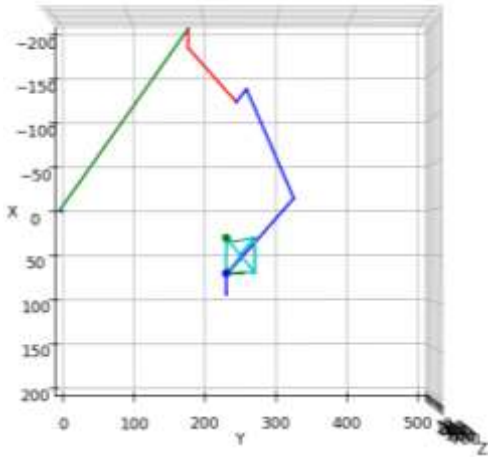
The result is that the end effector robot succeeds in drawing a pyramid but with imperfect drawing results as shown in Figure 16. Each line that connects the corner points of the pyramid is not straight and bends slightly even though it finally ends at the expected point. This is because the movement of one point to another on the Cartesian axis is not linear but forms a function like a first-order system as shown in Figure 15. The bending of this line becomes more visible when the displacement of the point moves not only in one axis, but 2 or 3 at a time as that occurs on the vertical side of the pyramid.



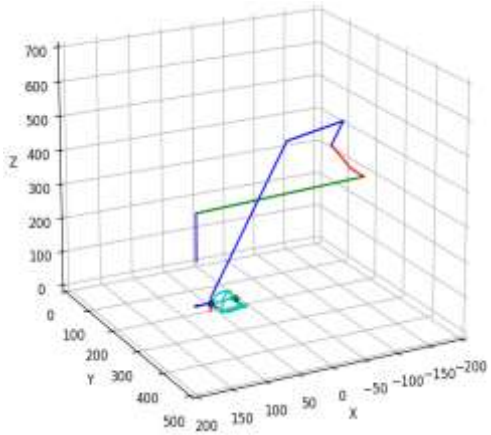
(a)



(b)



(c)



(d)

Figure 16. Pyramid pattern drawing test (a) Front view (b) Side view (c) Top view (d) 3D view

As with what happened in the previous test, the position of the end effector in Figure 15 is the position of the end effector calculated from forward kinematics. the resulting actual position has a fairly low accuracy with the maximum error that occurs in this test of 19.5 mm for the X axis, 10 mm for the Y axis, and 37 mm for the Z axis. However, this robot has high precision because each time the end effector returns to the same set point, the actual end effector is in a position close to its previous position.

Table 2. Comparison of computational and actual position

no.	*Set point (mm)*	Computational Posisition (mm)	Actual Position (mm)
1	(30, 230, 30)	(30.082, 230.066, 30.250)	(13, 220, 14)
2	(50, 250, 70)	(49.952, 249.983, 69.723)	(36, 248, 34)
3	(70, 230, 30)	(70.083, 229.865, 29.204)	(62, 228, 18)
4	(30, 230, 30)	(30.111, 230.059, 30.259)	(13.5, 220, 13)
5	(30, 270, 30)	(29.941, 270.030, 29.718)	(10.5, 265, 12)
6	(50, 250, 70)	(49.949, 249.995, 69.716)	(36, 247, 33)
7	(70, 270, 30)	(69.777, 269.858, 30.959)	(58, 264, 21)
8	(30, 270, 30)	(30.112, 269.960, 30.255)	(11, 267, 12)
9	(70, 270, 30)	(69.992, 270.040, 29.718)	(58, 265, 22)
10	(70, 230, 30)	(69.865, 230.176, 30.792)	(65, 227, 27)

4. Conclusion

Based on the discussion above, it can be concluded that the 6 DOF surgical robot manipulator system with mechanical ROC to control the end-effector has been successfully designed. The level of accuracy controlling the position of the robot end effector is quite low where the average MAE and RMSE values on the X, Y and Z axes are 19.09 and 21.594, 2.903 and 3.677, 28.597 and 33.309 respectively. However, from the three iterations of the test, the same MAE and RMSE values were obtained so that the position control of the robot's end effector has a low level of accuracy, but with a high level of precision. In addition, the trajectory planning algorithm integrates with the inverse kinematics algorithm and is influenced by the alpha (update rate) results in a smooth position change movement. The algorithm causes the movement of the end effector position in each axis similar to the response of a first order system.

Reference

[1] D. Oleynikov, "Robotic Surgery," *Surg. Clin. North Am.*, vol. 88, no. 5, pp. 1121–1130, 2008, doi: 10.1016/j.suc.2008.05.012.

[2] K. Kikuchi, K. Suda, S. Shibasaki, T. Tanaka, and I. Uyama, "Challenges in improving the minimal invasiveness of the surgical treatment for gastric cancer using robotic technology," *Ann. Gastroenterol. Surg.*, vol. 5, no. 5, pp. 604–613, 2021, doi: 10.1002/ags3.12463.

[3] C. P. Childers, "Estimation of the Acquisition and Operating Costs for Robotic Surgery," *J. Am. Med. Assoc.*, vol. 320, no. 8, pp. 835–836, 2018, doi: 10.1111/ans.13856.

[4] B. Hannaford *et al.*, "Raven-II: An open platform for surgical robotics research," *IEEE Trans. Biomed. Eng.*, vol. 60, no. 4, pp. 954–959, 2013, doi: 10.1109/TBME.2012.2228858.

[5] H. King, S. N. Kosari, and B. Hannaford, "Kinematic Analysis of the Raven-II TM Research Surgical Robot Platform UWEE Technical Report Number UWEETR-2012-0006 Kinematic Analysis of the Raven-II TM Research Surgical Robot Platform," no. 206, 2012.

[6] M. M. Marinho, M. C. Bernardes, and A. P. L. Bo, "Using General-Purpose Serial-Link Manipulators for Laparoscopic Surgery with Moving Remote Center of Motion," *J. Med. Robot. Res.*, vol. 1, no. 4, 2016, doi: 10.1142/S2424905X16500070.

[7] D. Nenchev, *Humanoid Robotics: A Reference*. 2018.

[8] Peter Corke, *Robotics, Vision and Control*, 2nd ed. Cham: Springer International Publishing, 2017.

[9] J. Haviland and P. Corke, "Manipulator Differential Kinematics Part II: Acceleration and Advanced Applications," pp. 1–13, 2022, doi: 10.1109/MRA.2023.3270221.