Research on modelling and trajectory planning for a 5-DOF robot manipulator

Group members:

1. Nguyễn Quốc Thịnh
2. Dương Xuyên Hạn
3. Đào Minh Nguyên

Abstract

- This paper presents the development of a Graphic User Interface (GUI) in MATLAB to simulate and control a 5-DOF robotic arm through communication between a microcontroller unit and MATLAB.
- Firstly, kinematic characteristics of the manipulator are analyzed, and the forward kinematics equations are established using Denavit-Hartenberg (D-H) method.
- Secondly, the working space is visualized in MATLAB and the inverse kinematics equations are derived using algebraic and geometric methods.
- Thirdly, velocity kinematics "Jacobian" and trajectory planning of the robotic arm are developed to analyze and control the robot’s motion.
- Finally, the manipulator’s movement is both simulated in MATLAB and tested on the real model to exhibit the effectiveness of the proposed method.

1. Introduction

- Robots turn abstract goals into physical action: sending power to motors, monitoring motions, and guiding things towards the goal.
- There is a fact that the cost of industrial robots today is relatively high compared to the financial ability of Viet Nam.
- To solve this problem, there are two proposed solutions: first is to find and buy used industrial robots and restore the controller for the robot and secondly find new compact with low-cost educational robot arms, analyze and simulate the motions of that physical model.
- There is a large amount of literature which discuss the robot molding and analysis of industrial robots.
- The paper is organized as follow: section II gives a full description of the manipulator, section III gives the mathematical kinematic model, section IV covers the interface software, section V presents the experimental results, and section VI concludes this paper.

2. Robot description

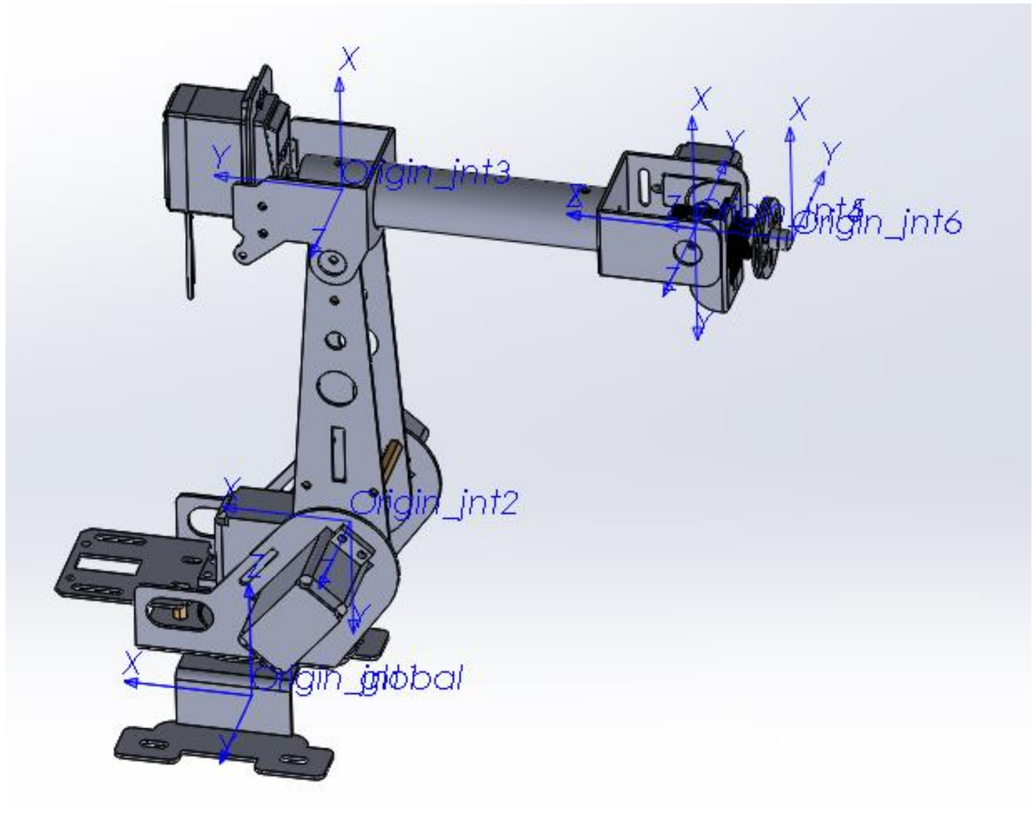


Fig. 1. Robot Arm Model

The robot arm model has 5 degrees of freedom (DOF). These joints provide shoulder rotation, shoulder back and forth motion, elbow motion, wrist up and down motion, wrist rotation. The limits of the joints are listed on Table. 1:

Table 1. The joint limits

	Upper Limit	Lower Limit
θ_1	$-\pi/2$	$\pi/2$
θ_2	$-\pi/4$	$\pi/4$
θ_3	$-\pi/4$	$\pi/4$
θ_4	$-\pi/2$	$\pi/2$
θ_5	$-\pi$	π

3. Kinematics

- The relationship between the end-effector position and joints variables are the results of the forward and inverse kinematics problems.
- Forward kinematics will demonstrate where the position of robot’s manipulator hand will be if all joints are known.
- If the position and orientation of end-effector is known, inverse kinematics will calculate all the possible cases of joints values.

3.1 Forward kinematics

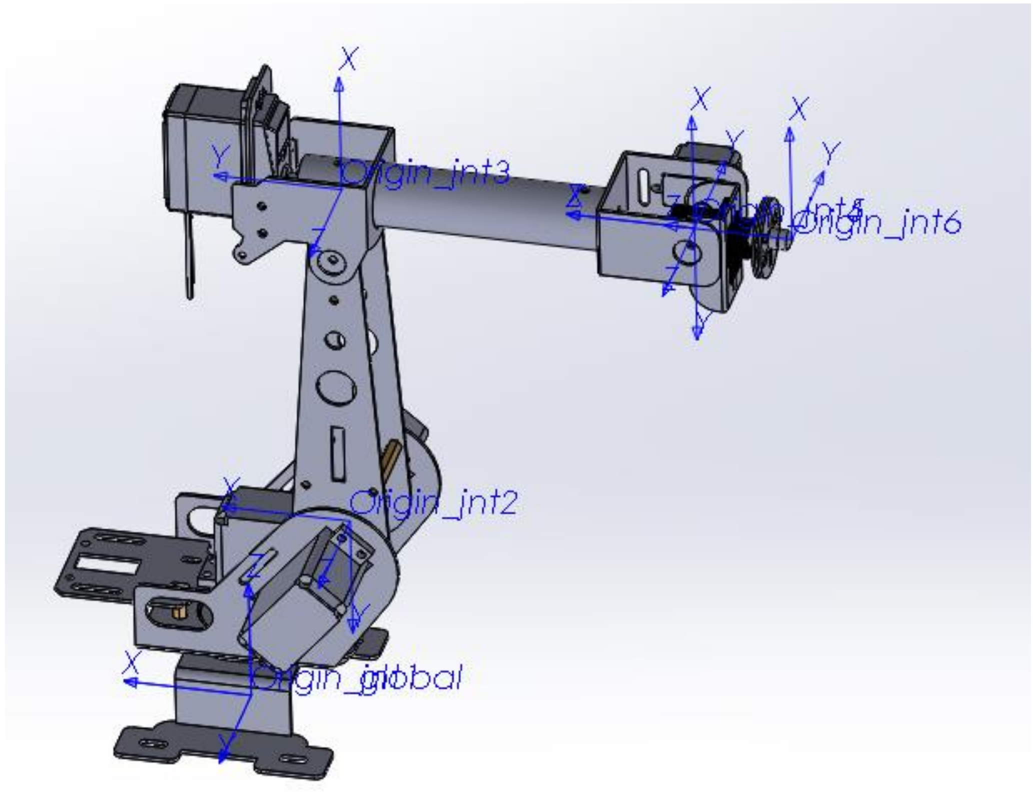


Fig. 2. The coordinate frames attached into the robot

There are many methods can be used to derive the kinematics calculation. The Denavit-Hartenberg analyses is one of the most used [4]. In this study, the D-H convention is used with DH table (Table 2) and coordinate frames attached as Fig. 2.

Table 2. DH table based on coordinate frames

Frame No.	a_i (mm)	α_i (deg.)	d_i (mm)	θ_i (deg.)
1	-37	-90	76	θ_1
2	135.5	0	0	θ_2
3	-126	0	0	θ_3
4	0	-90	0	θ_4
5	0	0	-35	θ_5

3.1 Forward kinematics

The formula of homogeneous transformation matrix [4] is:

$$ {}^{i-1}_{i}T = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha_i & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha_i & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

By substituting the parameters in Table 2; the transformation matrices from base to end-effector can be obtained:

$$ {}^{0}_{1}T = \begin{bmatrix} c\theta_1 & 0 & -s\theta_1 & a_1 c\theta_1 \\ s\theta_1 & 0 & c\theta_1 & a_1 s\theta_1 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} {}^{1}_{2}T = \begin{bmatrix} c\theta_2 & -s\theta_2 & 0 & a_2 c\theta_2 \\ s\theta_2 & c\theta_2 & 0 & a_2 s\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} {}^{2}_{3}T = \begin{bmatrix} c\theta_3 & -s\theta_3 & 0 & a_3 c\theta_3 \\ s\theta_3 & c\theta_3 & 0 & a_3 s\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^{3}_{4}T = \begin{bmatrix} c\theta_4 & 0 & -s\theta_4 & 0 \\ s\theta_4 & 0 & c\theta_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} {}^{4}_{5}T = \begin{bmatrix} c\theta_5 & -s\theta_5 & 0 & 0 \\ s\theta_5 & c\theta_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

3.1 Forward kinematics

From these matrices, we can compute the transformation matrix between coordinate from the base {0} to end-effector {5}:

$${}^0_5T = {}^0_1T.{}^1_2T.{}^2_3T.{}^3_4T.{}^4_5T = \begin{bmatrix} {}^0_5R & {}^0_5p \\ 0 & 1 \end{bmatrix}$$ With the orientational and translational matrix: $${}^0_5R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad {}^0_5p = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

Where the components of orientational matrix are:

$$\begin{aligned}
r_{11} &= s\theta_1 s\theta_5 + c\theta_{234} c\theta_1 c\theta_5 \\
r_{12} &= c\theta_5 s\theta_1 - c\theta_{234} c\theta_1 s\theta_5 \\
r_{13} &= -s\theta_{234} c\theta_1 \\
r_{21} &= c\theta_{234} c\theta_5 s\theta_1 - c\theta_1 s\theta_5 \\
r_{22} &= -c\theta_1 c\theta_5 - c\theta_{234} s\theta_1 s\theta_5 \\
r_{23} &= -s\theta_{234} s\theta_1 \\
r_{31} &= -s\theta_{234} c\theta_5 \\
r_{32} &= s\theta_{234} s\theta_5 \\
r_{33} &= -c\theta_{234}
\end{aligned}$$

And the components of translational matrix are:

$$\begin{aligned}
p_x &= c\theta_1(a_1 + a_3 c\theta_{23} + a_2 c\theta_2 - d_5 s\theta_{234}) \\
p_y &= s\theta_1(a_1 + a_3 c\theta_{23} + a_2 c\theta_2 - d_5 s\theta_{234}) \\
p_z &= d_1 - a_3 s\theta_{23} - a_2 s\theta_2 - d_5 c\theta_{234}
\end{aligned}$$

=> With the transformation matrix developed above, we can define the relationship between the base frame {0} and the TCP (Tool Center Point) frame {5} of the manipulator.
From that, the position (${}^0_5p$) and orientation (${}^0_5R$) of the TCP can be calculated easily with all input joints value.

3.2 Workspace

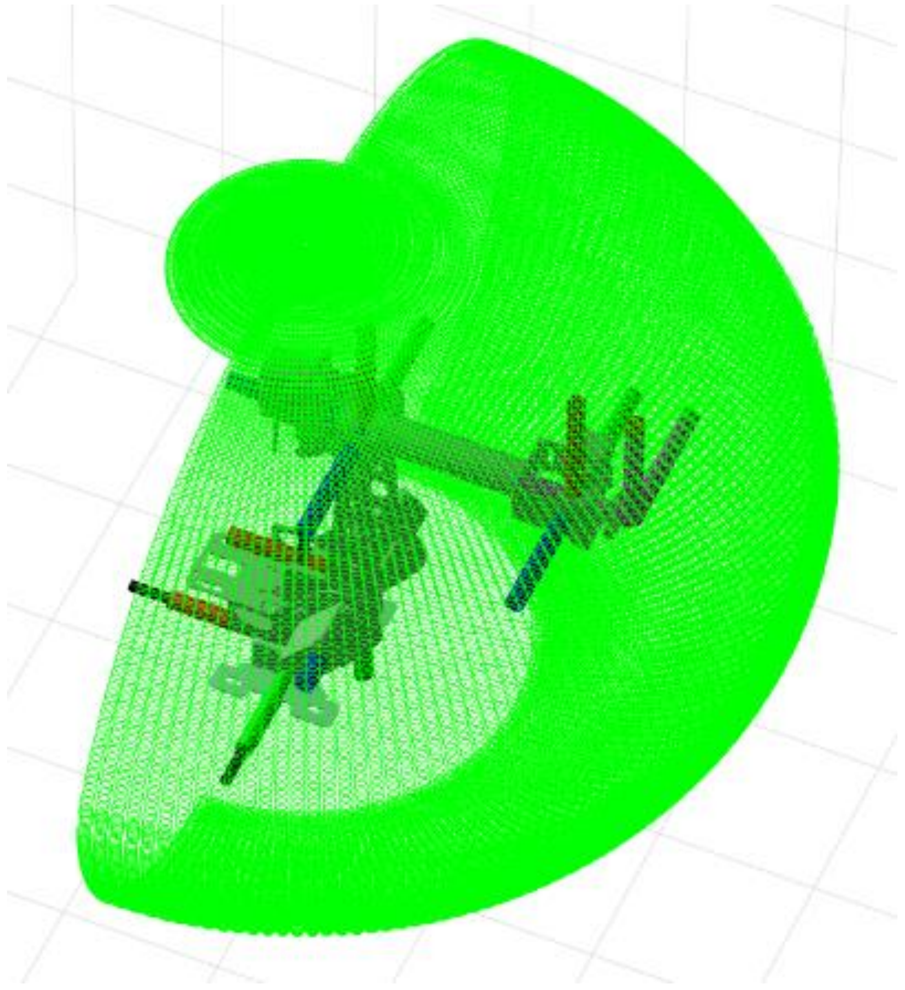


Fig. 3. The workspace of the robot arm model

- The workspace of the manipulator depends on allowed ranges of joints rotation. The proposed method is applied to the joint space of the manipulator to estimate the working space [4].

- For our robot model, the algorithm yields values for the four joint variables θ_1, θ_2, θ_3, θ_4 (θ_5 not included) continuously and generates a number of samples of the workspace.

- The joints values are substituted into forward kinematics to calculate the end-effector position (p_x,p_y,p_z) and drawn on Fig. 3.

3.3 Inverse kinematics

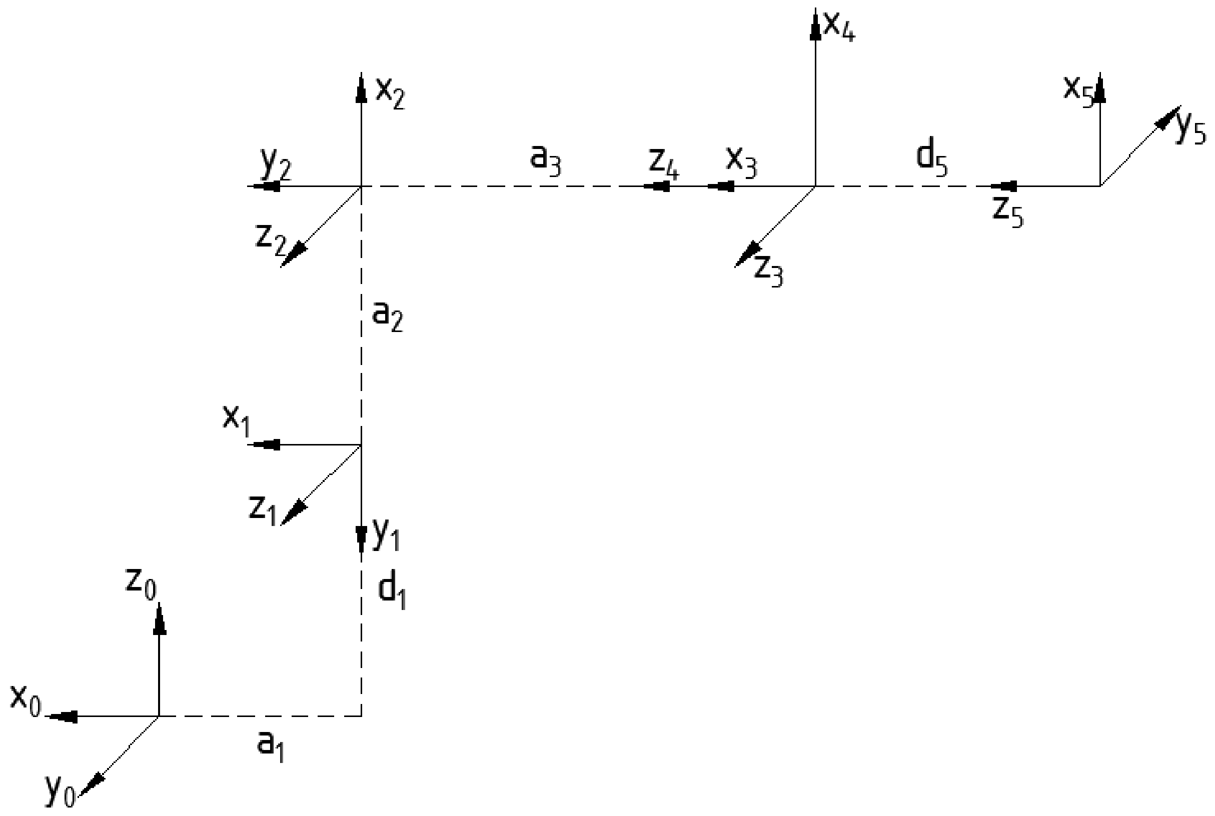


Fig. 4. The coordinate systems from base {0} and TCP {5} based on DH table.

- The problem of inverse kinematics is that, with given the orientation and position of the Tool Center Point (TCP), we can find all possible cases of manipulators' joints values which the robot arm can reach to that desired TCP [4].

- Both the analytical and geometrical solving method will be demonstrated in this thesis to solve the inverse kinematics problem.

3.3 Inverse kinematics

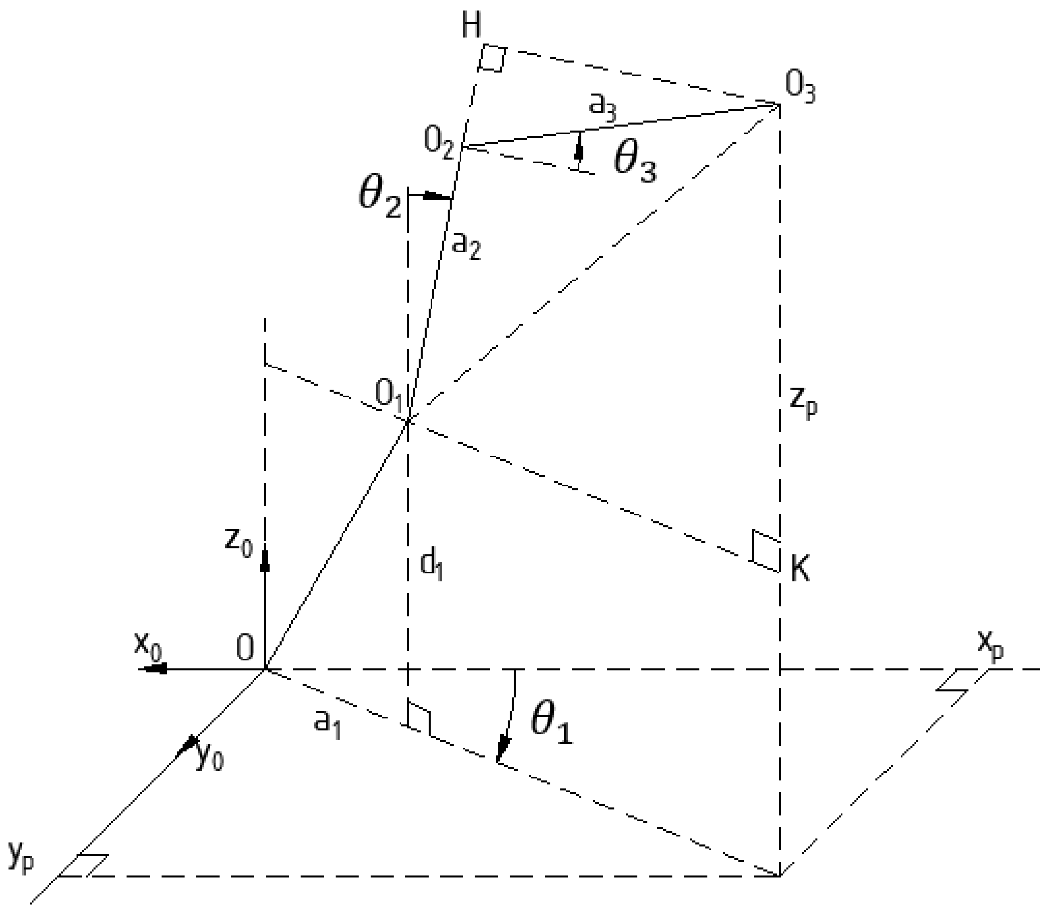


Fig. 5. Coordinate system projection on base frame {0]

The summary of inverse kinematics problem can be derived in equations of 5 manipulators' joint value:

$$\theta_1 = -atan2\left(y_p, -x_p\right); \;\; \theta_1 = -atan2\left(y_p, -x_p\right) + \pi$$
$$\theta_2 = -\frac{\pi}{2} + atan2(d_4 cos\theta_3, a_2 + d_4 sin\theta_3) + atan2(z_p - d_1, \sqrt{x_p^2 + y_p^2} - a_1)$$
$$\theta_3 = atan2\left(D, \pm\sqrt{1 - D^2}\right)$$
$$\theta_4 = atan2(b_{23}, b_{13})$$
$$\theta_5 = atan2(-b_{32}, -b_{31})$$

From these solutions, we can see that there are 4 possible cases solutions of 5 DOFs since there are 2 solutions of θ_1 and 2 solutions for θ_3.

3.4 Validation of forward and inverse kinematics

- In order to validate the forward and inverse mathematical kinematics model build above, the simulation is developed based on MATLAB Robotics Toolbox, after constructing the 3D model in Solidworks with coordinate frames, the robot is imported into MATLAB in URDF format.

- By using MATLAB Robotics Toolbox with URDF file, the simulation process is developed to validate the kinematics model with.

3.4 Validation of forward and inverse kinematics

The final results of error between simulation and calculation is demonstrated by Fig. 6. The horizontal axis represents the number of randomly chosen points, and the vertical axis represents the position error in millimeters.

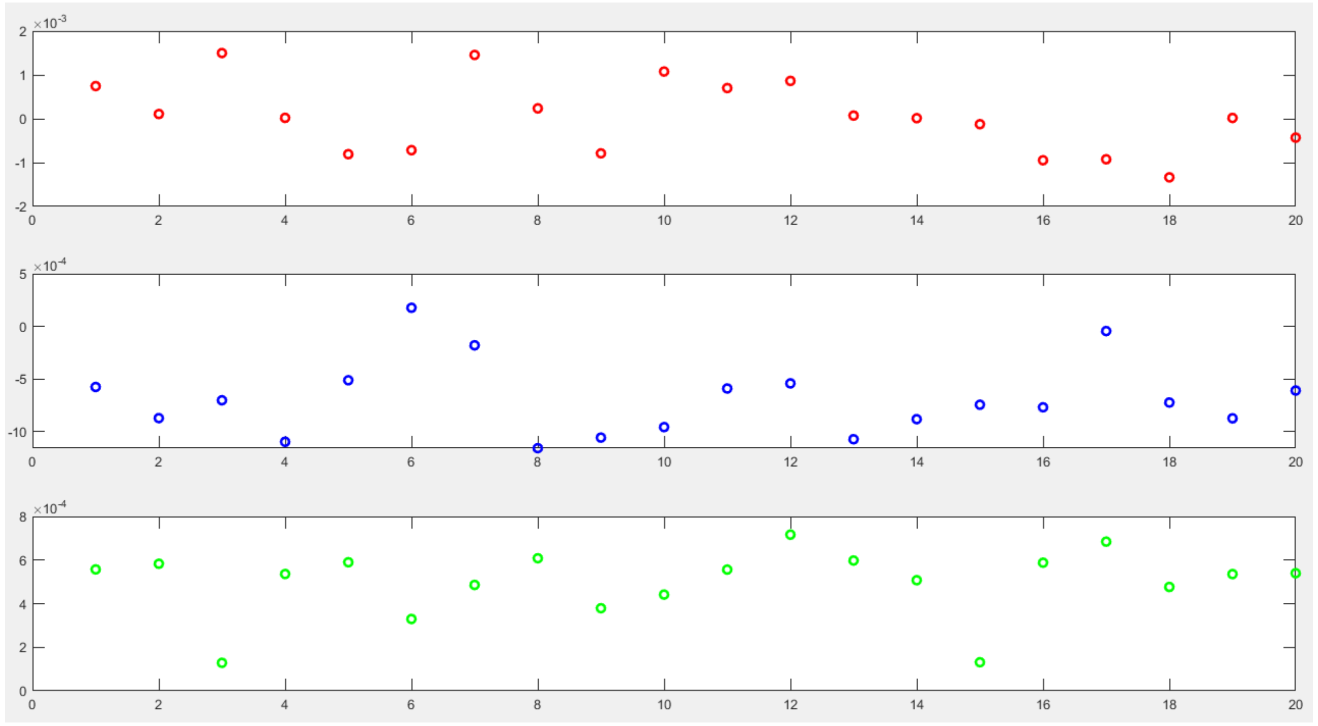


Fig. 6. The error of forward kinematics
(red: x axis, blue: y axis, green: z axis)

3.4 Validation of forward and inverse kinematics

A set of points generated by the specific circle path are taken to validate the inverse kinematics problem.

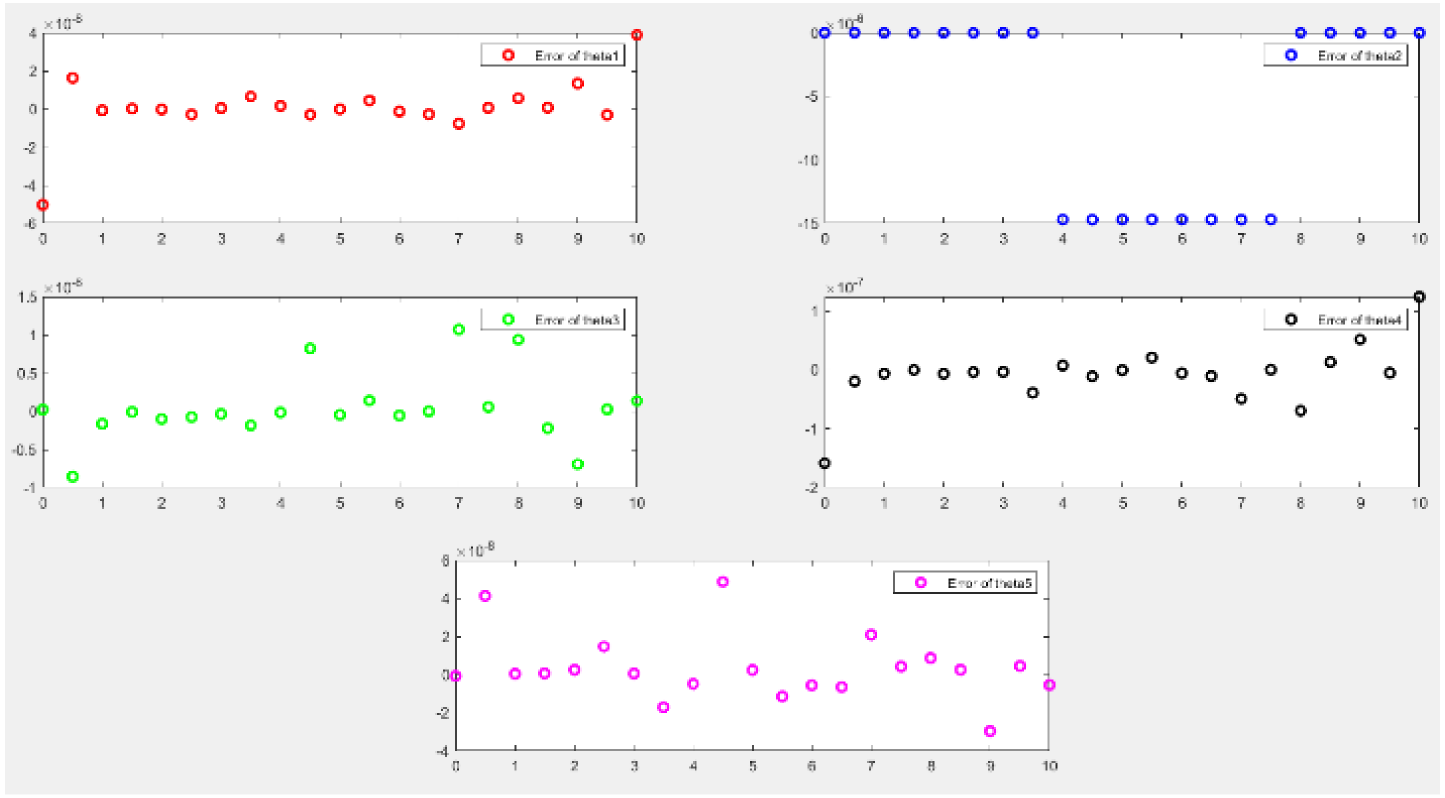


Fig. 7. The error of inverse kinematics of θ_1, θ_2, θ_3, θ_4, θ_5

From Fig. 6 and Fig. 7, the error is relatively small that it would not cause any major difference.

3.5 Velocity kinematics/Arm Jacobian

- The Jacobian is one of the most important quantities in the analysis and control of robot motion. It is used for smooth trajectory planning and execution in the derivation of the dynamic equation.

- To investigate target with specified velocity, each joint velocity at the specified joint positions needs to be found. This is accomplished using Jacobian, which are used to relate joint velocities to the linear and angular velocities of the end-effector [1]. For the AL5B robot arm the Jacobian matrix is equal 6x5.

$$J = \begin{bmatrix} J_{vi} \\ J_{wi} \end{bmatrix} = \begin{bmatrix} J_{v1} & J_{v2} & J_{v3} & J_{v4} & J_{v5} \\ J_{w1} & J_{w2} & J_{w3} & J_{w4} & J_{w5} \end{bmatrix} = \begin{bmatrix} J_1 & J_2 & J_3 & J_4 & J_5 \end{bmatrix}$$

Where:

$$J_{vi} = z_{i-1}^{0} \times (O_n^0 - O_{i-1}^0)\,,\ J_{wi} = z_{i-1}^{0}\,,\, i \in [1,2,\dots,5]$$

3.5 Velocity kinematics/Arm Jacobian

Finally, we obtain:

$$J_1 = \begin{bmatrix} -s\theta_1(a_1 + a_3c\theta_{23} + a_2c\theta_2 - d_5s\theta_{234}) \\ c\theta_1(a_1 + a_3c\theta_{23} + a_2c\theta_2 - d_5s\theta_{234}) \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad J_2 = \begin{bmatrix} -c\theta_1(a_3s\theta_{23} + a_2s\theta_2 + d_5c\theta_{234}) \\ -s\theta_1(a_3s\theta_{23} + a_2s\theta_2 + d_5c\theta_{234}) \\ d_5s\theta_{234} - a_2c\theta_2 - a_3c\theta_{23} \\ -s\theta_1 \\ c\theta_1 \\ 0 \end{bmatrix}$$

$$J_3 = \begin{bmatrix} -c\theta_1(a_3s\theta_{23} + d_5c\theta_{234}) \\ -s\theta_1(a_3s\theta_{23} + d_5c\theta_{234}) \\ d_5s\theta_{234} - a_3c\theta_{23} \\ -s\theta_1 \\ c\theta_1 \\ 0 \end{bmatrix} \qquad J_4 = \begin{bmatrix} -d_5c\theta_{234}c\theta_1 \\ -d_5c\theta_{234}s\theta_1 \\ d_5s\theta_{234} \\ -s\theta_1 \\ c\theta_1 \\ 0 \end{bmatrix} \qquad J_5 = \begin{bmatrix} -d_5c\theta_{234}(c\theta_1 + s\theta_{234}s\theta_1) \\ 0 \\ 0 \\ -s\theta_{234}c\theta_1 \\ -s\theta_{234}s\theta_1 \\ -c\theta_{234} \end{bmatrix}$$

4. Trajectory Planning

The simplest trajectory is a polynomial with boundary conditions on position, velocity and acceleration. The form for the time scaling s(t), t ϵ [0,T] is a quintic polynomial of time:

$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

$$\dot{s}(t) = a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4$$

$$\ddot{s}(t) = 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3$$

Solving for the constraints: $s(0)=\dot{s}(0)=0$; $s(T)=1$; $\dot{s}(T)=0$; $\ddot{s}(0)=\ddot{s}(T)=0$, results in the robot's motion no longer having infinite jerk - which is the derivative of acceleration, at the start and end point [1]

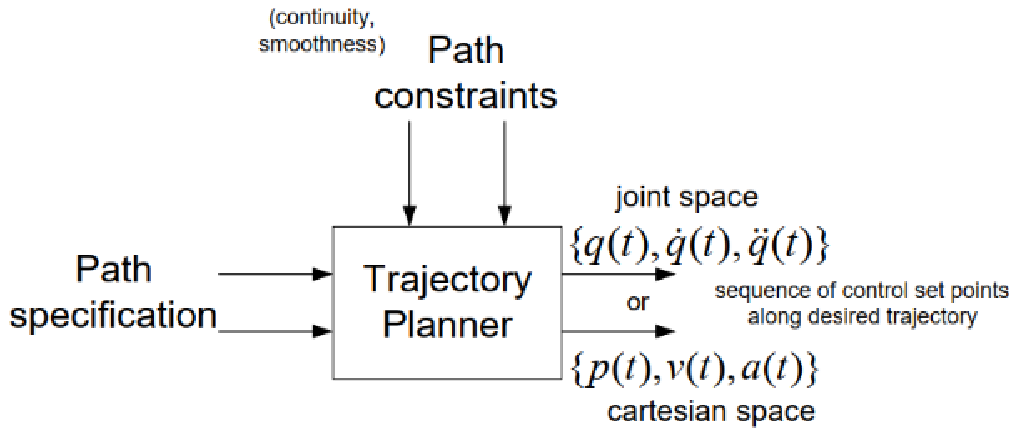
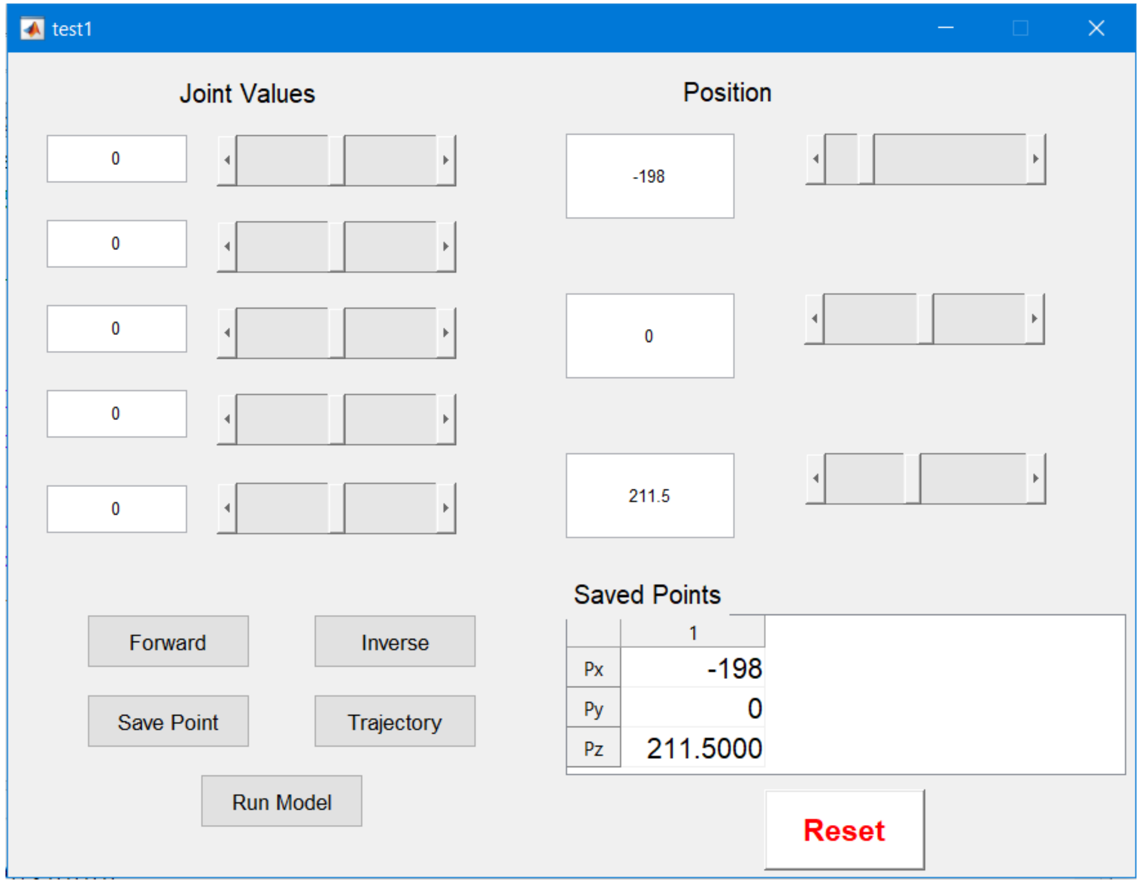


Fig. 8. Trajectory Planning Block Diagram [3]

5. Software

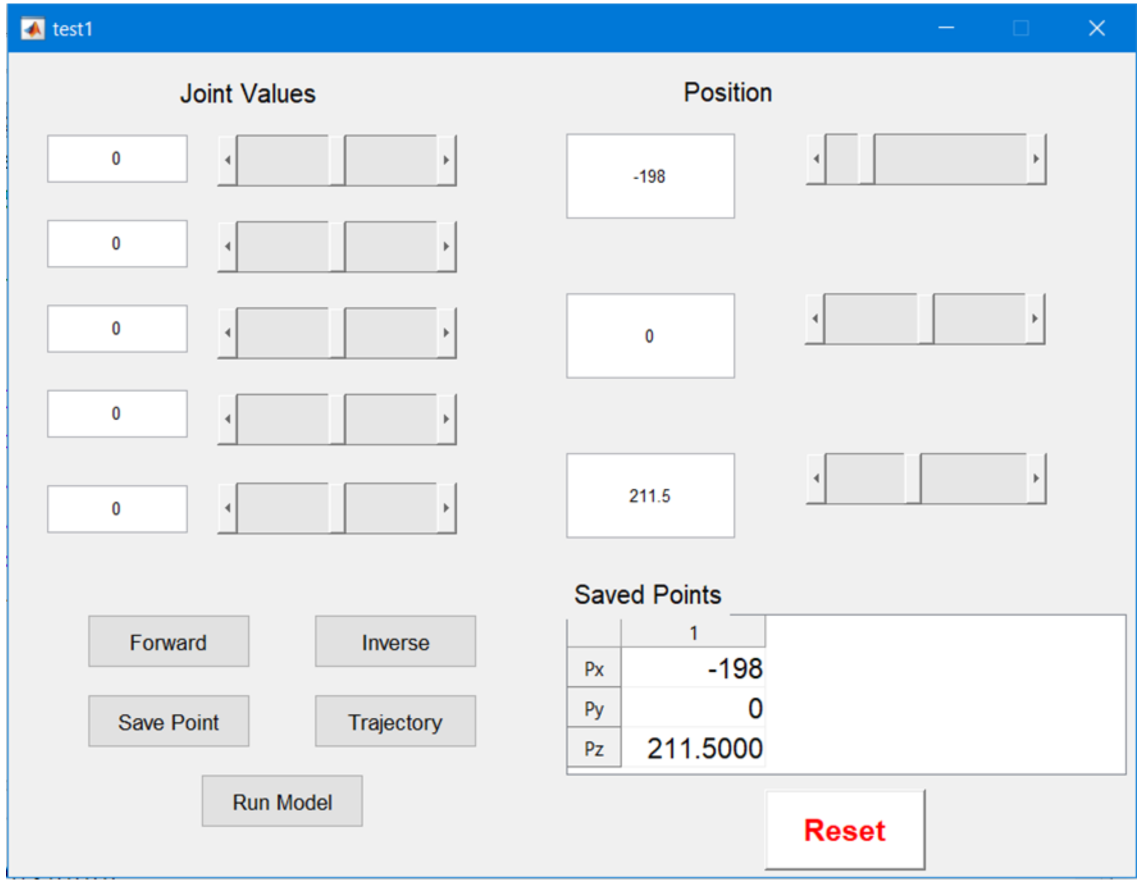


	1
Px	-198
Py	0
Pz	211.5000

Fig. 9. Graphical User Interface

- The GUI (Fig. 9) was developed by MATLAB programming to compute the forward kinematics, inverse kinematics, Jacobian and trajectory planning of the 5-DOF robot arm model.
- The motional simulation of the robot arm was also included of GUI to show the motion of model in Fig. 10 based on the theoretical analysis presented in this paper.

5. Software



	1
Px	-198
Py	0
Pz	211.5000

Joint values: 0, 0, 0, 0, 0. Position: -198, 0, 211.5.

Fig. 9. Graphical User Interface

- Firstly, user can adjust 'Joint Values' in the boxes or sliders to achieve the end-effector position of the robot arm through forward kinematics, and the 'Forward' button is using for plotting the pose of the 3D robot.
- Secondly, user can find the values of the joints at a specific end-effector position by setting the three 'Position' values in the boxes or sliders, the 'Inverse' button is used to show all the possible solutions of the inverse kinematics on the MATLAB Workspace area.
- Thirdly, user can generate trajectory through points by using 'Save Point' button to store the assigned points on 'Saved Points' table and 'Trajectory' button is for plotting the robot's motion and showing its end-effector Cartesian velocity v_x, v_y, v_z (Fig. 10).

5. Software

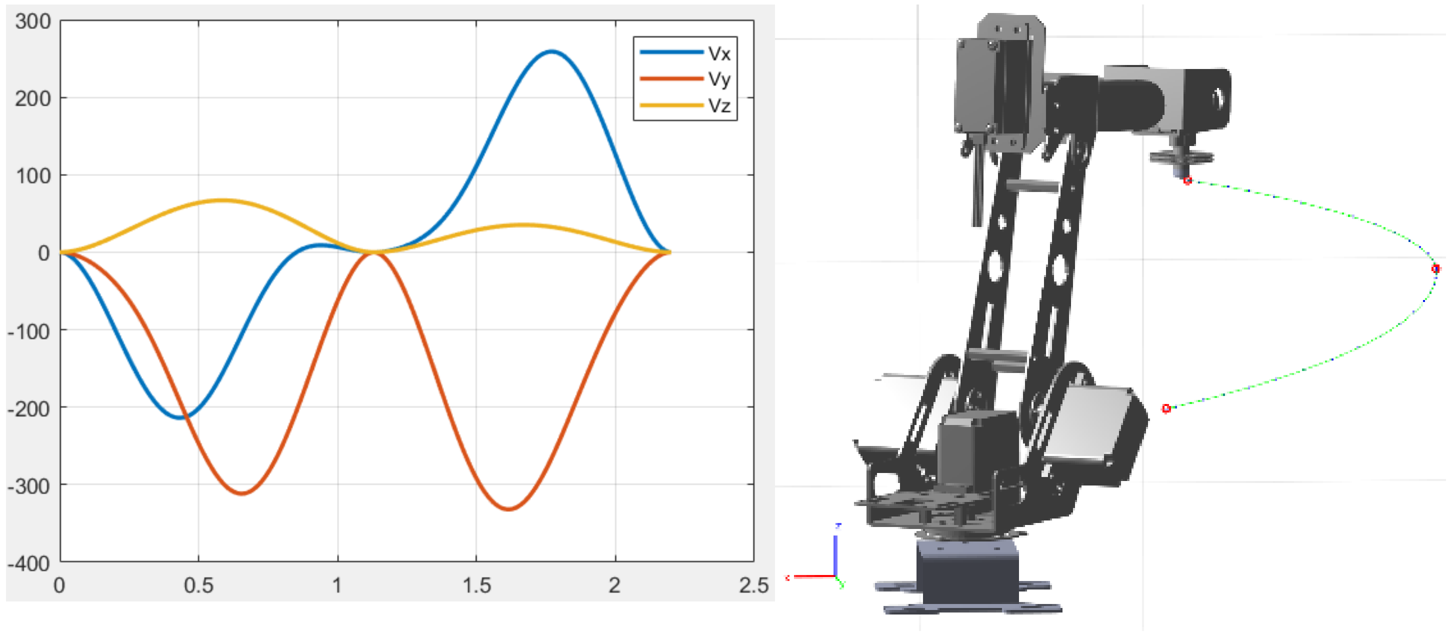


Fig. 10. The end-effector Cartesian velocity and trajectory planning

- Finally, the robot model can be controlled through connection between MATLAB and Arduino (Fig. 11), user can directly control in forward kinematics, inverse kinematics or create path for the end-effector to follow.

6. Experiment

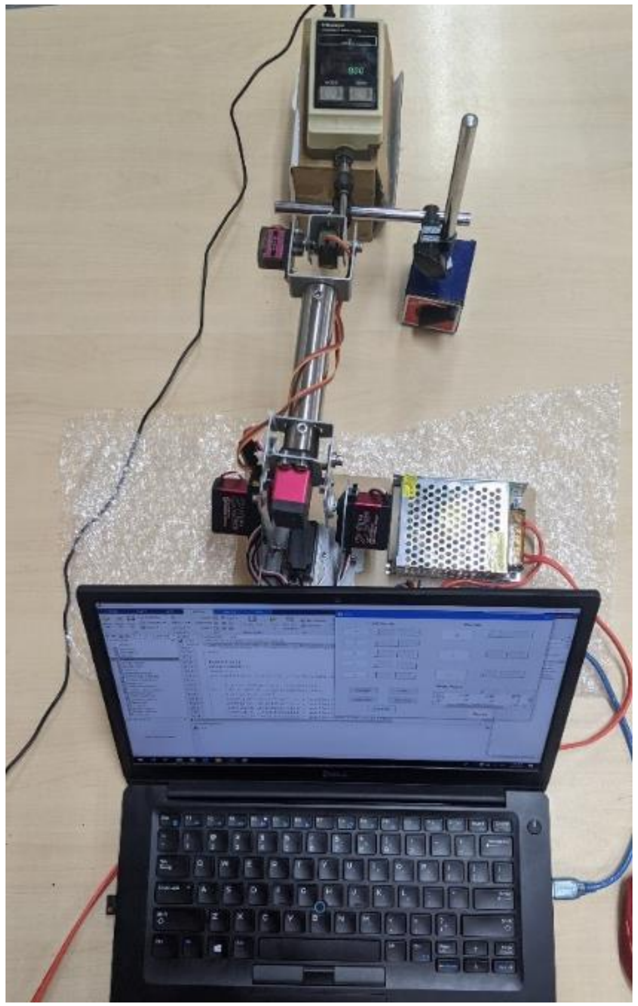


Fig. 11. Experiment setup

- In order to verify the accuracy and precision of the model, we used a digimatic indicator to determine the errors of the end-effector in x, y, z axis (Fig. 11).
- By controlling the robot doing the same task through MATLAB multiple times, and measure the errors in Cartesian coordinate system, we conclude that the error of the end-effector is $\pm 0.5mm$ in three directions.

7. Conclusions

- In this research mathematical modeling and kinematic analysis of our robot model was developed and validated using MATLAB. Kinematics model was achieved with Denavit-Hartenberg (D-H) method. Kinematics and Jacobian solutions were fed into a trajectory planner using MATLAB GUI.

- The algorithm was implemented on the real robot through the connection between MATLAB and Arduino to verify simulation results. Experimental results are determined by a digimatic indicator to evaluate the effectiveness of the simulation program.

- Robotic arms are now widely used in production from arranging, processing to inspecting products. The results of the project can be applied to develop robotic arm product lines for the domestic manufacturing industry.

Acknowledgement: This research is funded by Office for International Study Programs (OISP), Ho Chi Minh City University of Technology (HCMUT), VNU-HCM under grant number **(SVOISP-2020-CK-66).** We acknowledge the support of time and facilities from HCMUT, VNU-HCM for this study.

8. References

1. K. M. Lynch and F. C. Park (2017), *Modern Robotics*, Cambridge University Press.Harekrushna Sutar, et al., *Progress of Red Mud Utilization: An Overview.* Review Article
2. T. C. Phung, T. T. Nguyen, T. D. Nguyen, and N. S. Nguyen, *"A study on designing controller and building control algorithms for 6dof robot applied in education"* in Science and Technology Development Journal - Engineering & Technology, 2018, pp.33–41.
3. M. A. Qassem, I. Abuhadrous and H. Elaydi, *"Modeling and Simulation of 5 DOF educational robot arm"*, 2010 2nd International Conference on Advanced Computer Control, 2010, pp. 569-574, doi: 10.1109/ICACC.2010.5487
4. Mark W. Spong, Seth Hutchinson, and M. Vidyasagar (2020), *Robot Modeling and Control* - 2nd Edition, John Wiley & Sons, Ltd.