Robot Arm for Pick and Place Test Tube

Approaches to find the correct locations

Name: Tom Dang

Contents

Appendix

Equations

Figure

Table

Abstract

This report presents the comprehensive design, kinematic modeling, and dynamic analysis of a three-degree-of-freedom (3-DOF) robotic arm for an automated test tube pick-and-place task. The methodology encompasses the mechanical design in SolidWorks, followed by a rigorous mathematical analysis. Forward kinematics were established using general homogeneous transformation matrices, while analytical solutions were successfully derived for the inverse kinematics problem.

Two primary control strategies for path generation were investigated: task space and joint space control. The analysis explored various trajectory generation techniques for joint space movement, including linear interpolation, cubic splines, and quintic polynomials, to assess their impact on motion smoothness. Dynamic analysis revealed that while the manipulator is functional, Joint 2 and Joint 3 experience peak torques of approximately 0.2 N*m and 0.55 N*m, respectively. These values are well below the 5 N*m maximum torque capacity of the joints, indicating the motors are oversized.

The findings validate the kinematic models and control strategies. Torque analysis shows that there are further adjustments depending on the path for optimization: picking more loads or minimizing the volume space used. These are 2 possible paths to optimize toward. The design is restricted to demonstration in the modeling and simulation. The physical equivalent with not in real life requires more work on redesigning the motor and gear drive to make the robot arm work properly.

Assumptions

- The tool (end effector) is not the focus of the robot arm, the design is simplified as an extension of the third link, which is called end effector link in this report.
- The material used in this design PP high density, which is a built-in material data in SolidWorks software.
- The working environment is defined as a robot standing in between two parallel conveyor belts. One contains an array of test tubes that might be empty or contain chemicals to deposit per request, while the other contains an empty that has slots for deposited test tubes to be placed in.
- Maximum torque of each joint is 5 N*m

Phase 1: Robotic Design and Forward Kinematics

In this phase design, the aim is to initialize a design of robot with a clear purpose and vision for development, along with the first design to start the iteration as the phase moves on to further kinematic and dynamic analysis. This phase is treated as a preparation stage for the first modeling of the robot arm using MATLAB.

Design Spatial Manipulator

The robot arm is designed to be a 3 DOF robot with all revolute joints for picking and placing the test tube. The mass of the test tube is not a not a significant concern in this case, given its low mass and its center of mass of close is almost identical as a point mass.

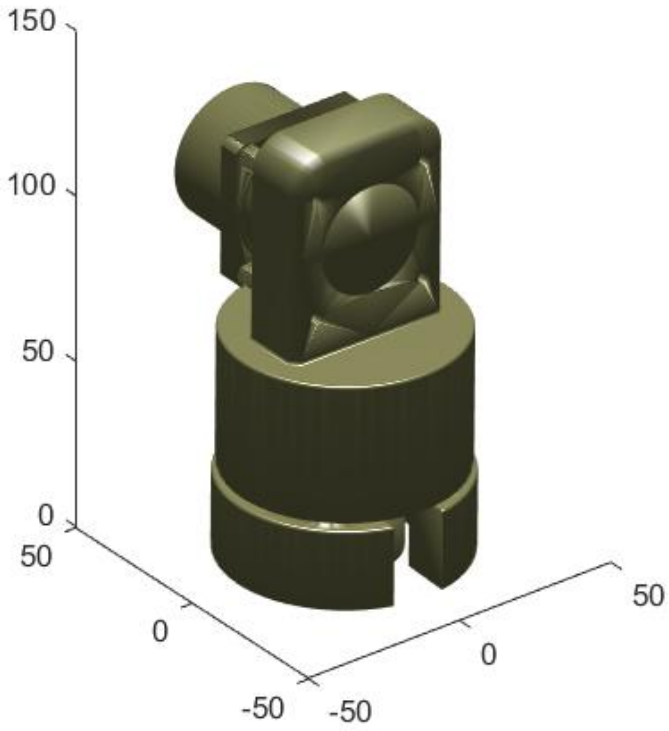


Figure 1: Base component of the robot arm

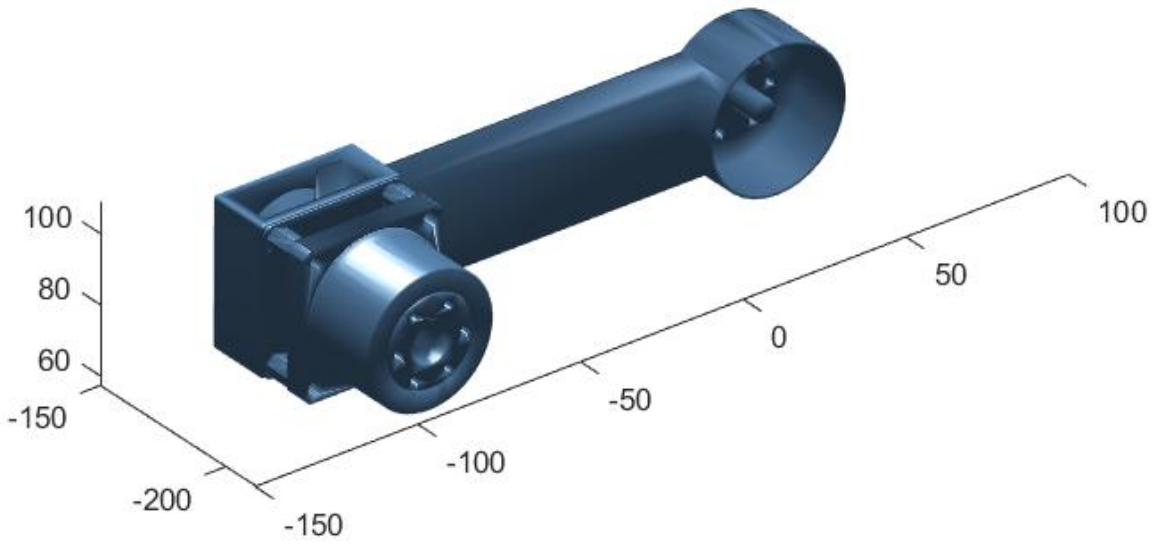


Figure 2: Middle link of robot arm

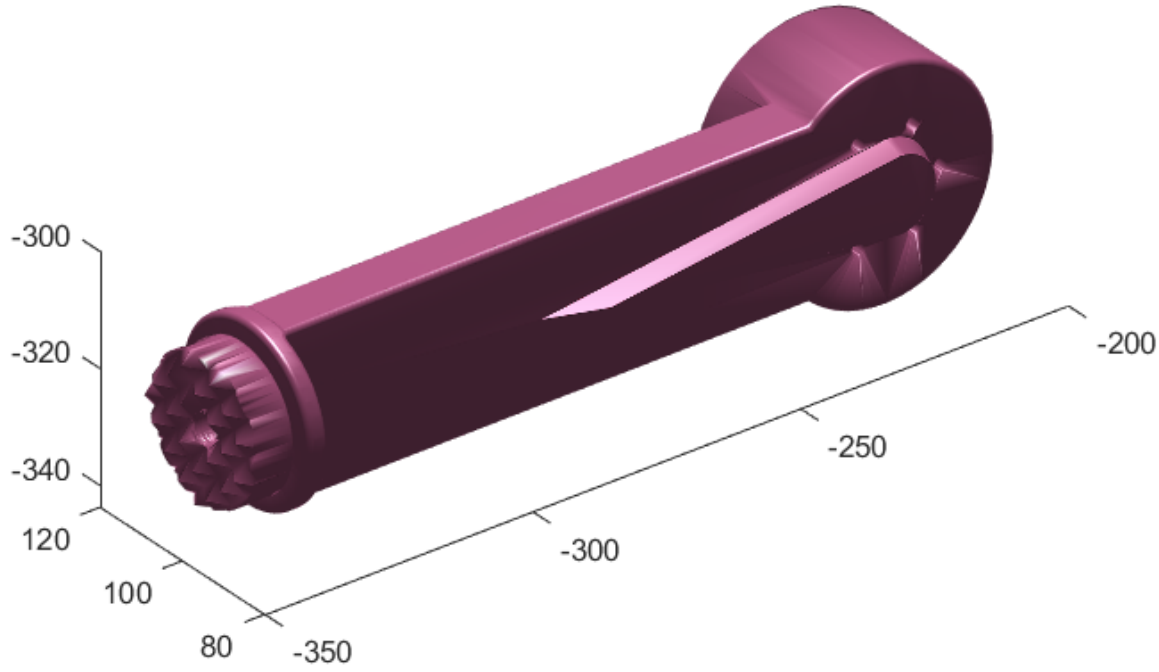


Figure 3: End effector link of robot arm

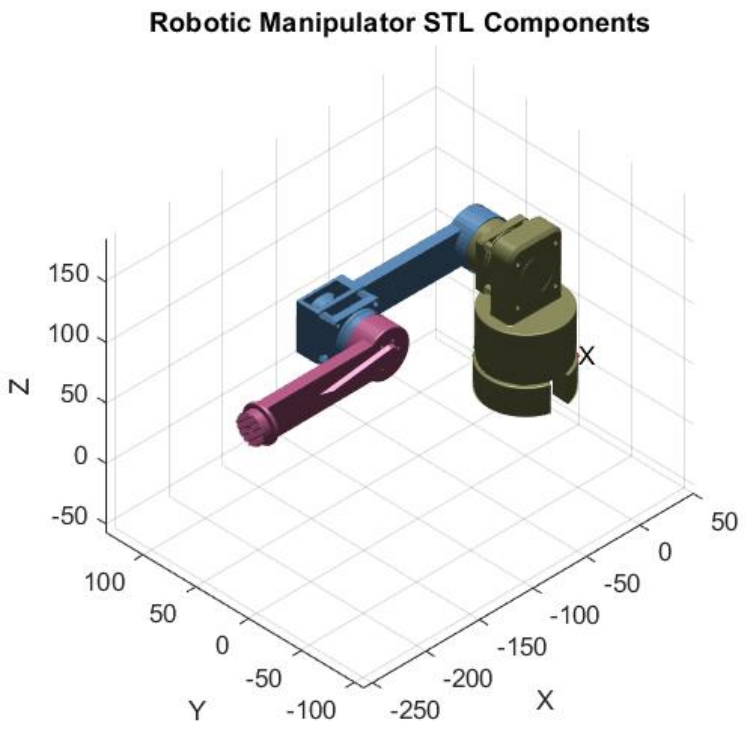


Figure 4: A diagram of a fully assembled robot arm

The base of the robot arm is designed to elevate the 2-link robot arm to an offset to prevent obstacles from the ground. In this design, the assumption is that the base can be elevated off the ground and given more space for the 2-link robot arm to have more space for rotation and necessary movement for a fixed trajectory. Since all the forces are acting on the Z axis and the base is rotating around the Z axis, there is no torque happening in the base link. The base link served as a weight to move the whole robot arm's center of mass to within its volume when the robot is performing its task.

The middle link and end effector link are reconstructed from a simple 2-link robot arm in 2D plane. To simplify the design, the 2 links have identical design, which means the same mass distribution and center of gravity and powered by the same motors.

Design Specification

The whole robot will be designed with PP high density, a built-in material for SolidWorks software. PP is well suited for environment with highly toxic chemicals and has acceptable mechanical properties, such as modulus of elasticity, fatigue strength and price for manufacturing. The material in the base is subjected to changes, depending on the alignment of the center of mass of the robot arm within its volume and the methods of

installation are considered in mechanical design, along with human factors and ergonomic designs.

The base is constrained to rotating around 180 degrees of any 2 consecutive quadrants. The middle link has no range limit as the motor of the end effector link has no collision with the base link during 360 rotation tests. The end effector link has 180 degrees of rotation range. The actual range for rotation is subjected to change to maximize the options for the best inverse kinematics solution on later phases.

Forward Kinematics

The objective of the forward kinematics is building and preparing a set of solutions for each component's behavior in respected to the base frame. There are 2 approaches of building the forward kinematics equations: Denavit-Hartenberg (DH) parameters and general homogenous transformations matrices.

The DH parameters are a standardized way to describe the geometry of robotic manipulators. There are 4 parameters for each joint: link length (a), link twist (α), link offset (d), and joint angle (θ). The general homogeneous transformation matrices encapsulate the rotation of its components and its position in a respected frame, specifically base frame in this case. This simulation prefers the general homogeneous transformation matrices methods due to the work of repositioning and orientating the assets in MATLAB has a significant similarity to the concept of homogeneous transformation.

Note: $C_n = \cos\ \theta_n$ and $S_n = sin\theta_n$. Similarly, $C_{nm} = \cos(\theta_n)\cos(\theta_m) - \sin(\theta_n)\sin(\theta_m)$ and $S_{nm} = \sin(\theta_n)\cos(\theta_m) + \sin(\theta_m)\cos(\theta_n)$

$$ {}^{0}_{1}T = \begin{matrix} C_1 & -S_1 & 0 & 0 \\ S_1 & C_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix} $$

Equation 1: Transformation of base link at base frame

$$ {}^{1}_{2}T = \begin{matrix} C_2 & 0 & S_2 & 1.3 \\ 0 & 1 & 0 & 40 \\ -S_2 & 0 & C_2 & 95 \\ 0 & 0 & 0 & 1 \end{matrix} $$

Equation 2: Transformation matrix of middle link from base frame

$$ {}^{0}_{2}T = \begin{matrix} C_1C_2 & -S_1 & C_1S_2 & 1.3C_1 - 40S_1 \\ S_1C_2 & C_1 & S_1S_2 & 40C_1 + 1.3S_1 \\ -S_2 & 0 & C_2 & 95 \\ 0 & 0 & 0 & 1 \end{matrix} $$

Equation 3: Transformation of middle link at base frame

$$
{}^{2}_{3}T = \begin{matrix} C_3 & 0 & S_3 & -133.3 \\ 0 & 1 & 0 & -27.5 \\ -S_3 & 0 & C_3 & 0.5 \\ 0 & 0 & 0 & 1 \end{matrix}
$$

Equation 4: Transformation matrix of end effector link from middle link frame

$$
{}^{0}_{3}T = \begin{matrix} C_1C_{23} & -S_1 & C_1S_{23} & -133.3C_1C_2 + 0.5C_1S_2 + 1.3C_1 - 12.5S_1 \\ S_1C_{23} & C_1 & S_1S_{23} & -133.3S_1C_2 + 0.5S_1S_2 + 1.3S_1 - 12.5C_1 \\ -S_{23} & 0 & C_{23} & 0.5C_2 + 133.3S_2 + 95 \\ 0 & 0 & 0 & 1 \end{matrix}
$$

Equation 5: Transformation of end effector link from base frame

$$
{}^{3}_{ee}T = \begin{matrix} 1 & 0 & 0 & -126.994 \\ 0 & 1 & 0 & -12.2355 \\ 0 & 0 & 1 & 2.8614 \\ 0 & 0 & 0 & 1 \end{matrix}
$$

Equation 6: Transformation matrix of end effector from end effector link frame

$$
{}^{0}_{ee}T = \begin{matrix} C_1C_{23} & -S_1 & C_1S_{23} & C_1(2.8614S_{23} - 126.994C_{23} - 133.3C_2 + 0.5S_2 + 1.3) - 0.2645S_1 \\ S_1C_{23} & C_1 & S_1S_{23} & S_1(2.8614S_{23} - 126.994C_{23} - 133.3C_2 + 0.5S_2 + 1.3) - 0.2645C_1 \\ -S_{23} & 0 & C_{23} & 126.994S_{23} - 2.8614C_{23} + 0.5C_2 + 133.3S_2 + 95 \\ 0 & 0 & 0 & 1 \end{matrix}
$$

Equation 7: Transformation of end effector from base frame

DH parameters can be found using point swarm optimization with the policy of the smallest average error percentage between 50 samples and 12 parameters subjected to optimization, including link length (a), link twist (α), link offset (d), and joint angle (θ) of each joint. Although this method cannot provide the exact equivalent result of the general homogeneous matrices method, the values are presented for preferences.

Index (n)	Link length (a)	Link offset (d)	Link twist (α)	Joint angle (θ_n)
1	1.34560	95.042768	-1.570526	θ_1
2	-133.301463	-64.703505	0.002879	θ_2
3	-101.773902	74.503106	-0.288565	θ_3
ee	-25.218347	-9.951452	0.206569	0

Table 1: DH parameters for robot arm

The parameters although violate the convention that link length must be positive, but this gives an acceptable error across multiple runs of point swarm optimization. The error is constantly off by 27 mm in Euclidian distance, which is due to the policy function in the point swarm optimization. The behavior of the line to relatively like the general homogeneous matrix method, which is another good selection as there are more selections, which give lower error rate, but the behavior is not like the intended one.

The parameters can be tuned by using different optimization techniques with different policy functions to yield to lower error rate and more similar behavior to the general homogeneous matrix method.

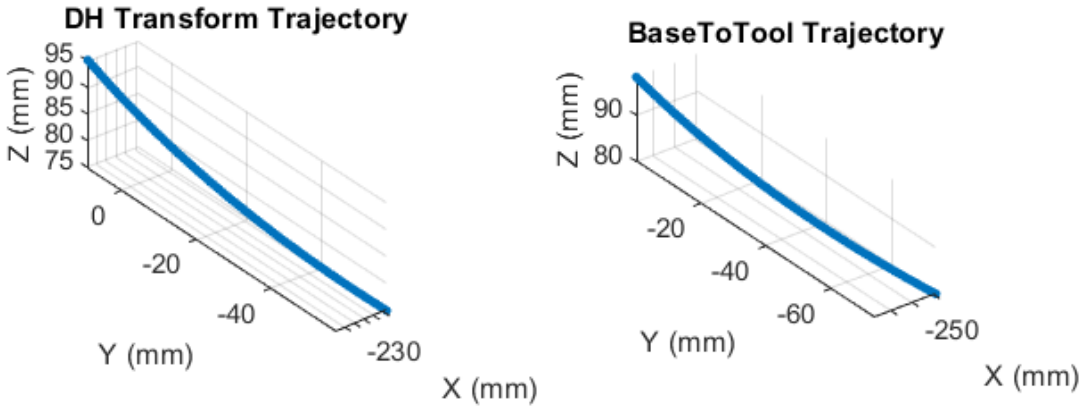


Figure 5: Comparison between DH parameters (left) and general homogeneous transformation matrix method (right)

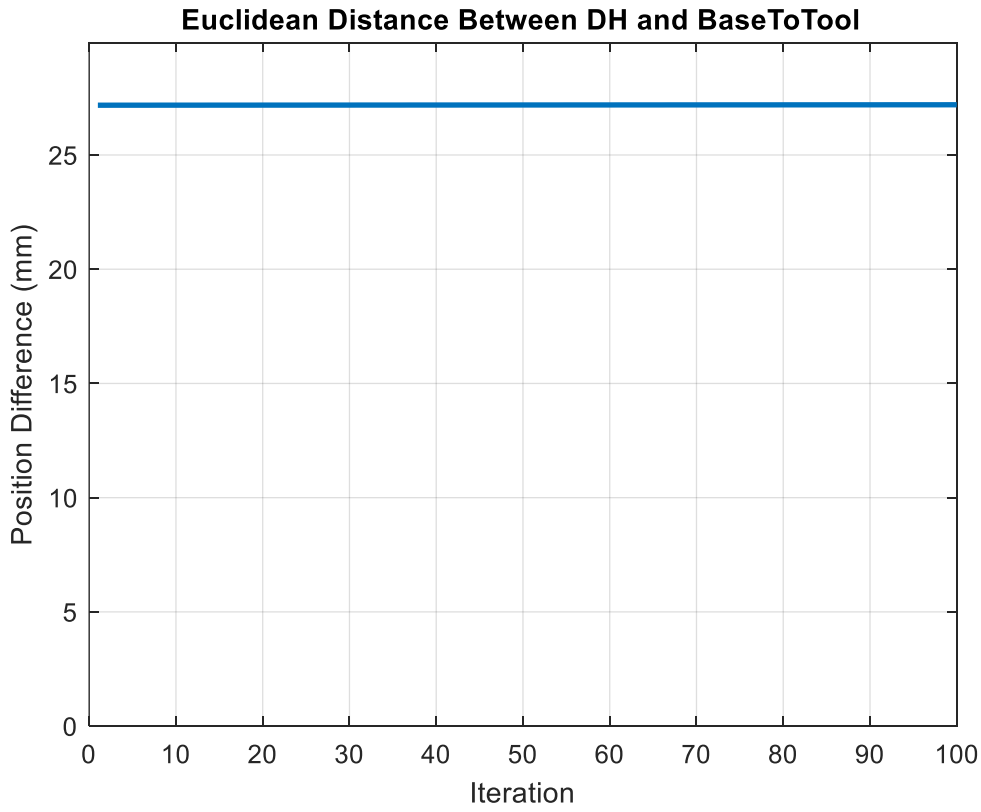


Figure 6: Euclidian error between DH parameters and general homogeneous transformation

Phase 2: Inverse Kinematics and Workplace Analysis

Inverse Kinematics

$$\begin{matrix} x \\ y \\ z \end{matrix} = \begin{matrix} C_1(2.8614S_{23} - 126.994C_{23} - 133.3C_2 + 0.5S_2 + 1.3) - 0.2645S_1 \\ S_1(2.8614S_{23} - 126.994C_{23} - 133.3C_2 + 0.5S_2 + 1.3) - 0.2645C_1 \\ 126.994S_{23} - 2.8614C_{23} + 0.5C_2 + 133.3S_2 + 95 \end{matrix}$$

Equation 8: Position equations in relation to the angles of joints

Let $R = 2.8614^2 + 126.994^2 = 16135.95$ and $C = 133.3^2 + 0.5^2 = 17769.14$

Step 1: Solve for θ_1

$$S_1 x - C_1 y = -0.2645$$

Which yield the solution is

$$\theta_1 = atan2(x, -y) \pm \arccos\left(\frac{-0.2645}{\sqrt{x^2 + y^2}}\right)$$

Equation 9: Inverse kinematic solution for θ_1

Where the condition for real solutions is

$$x^2 + y^2 \geq 0.2645^2$$

Step 2: Solve for θ_2 for each θ_1 (up to 2 solutions per θ_1)

Let $U = C_1 x + S_1 y - 1.3, V = z - 95, M = 133.3U - 0.5V, N = -0.5U - 133.3V$ and $L = \frac{1}{2}(R - C - U^2 - V^2)$ which yield the equations of

$$C_2 M + S_2 N = L$$

Which yield the solution is

$$\theta_2 = atan2(N, M) \pm \arccos\left(\frac{L}{\sqrt{M^2 + N^2}}\right)$$

Equation 10: Inverse kinematic solution for θ_2

Where the condition for real solutions is

$$M^2 + N^2 \geq L^2$$

Step 3: Solve for θ_3 for each pair of (θ_1, θ_2)

Let $P = U + 133.3C_2 - 0.5S_2$ and $Q = V - 0.5C_2 - 133.3S_2$

Rearrange the term to get $S_{23} = \frac{2.8614P+126.994Q}{R}$ and $C_{23} = \frac{2.8614Q-126.994P}{R}$

Which yield the solution is

$$\theta_3 = atan2(S_{23}, C_{23}) - \theta_2$$

Equation 11: Inverse kinematic solution for θ_3

Workspace Analysis

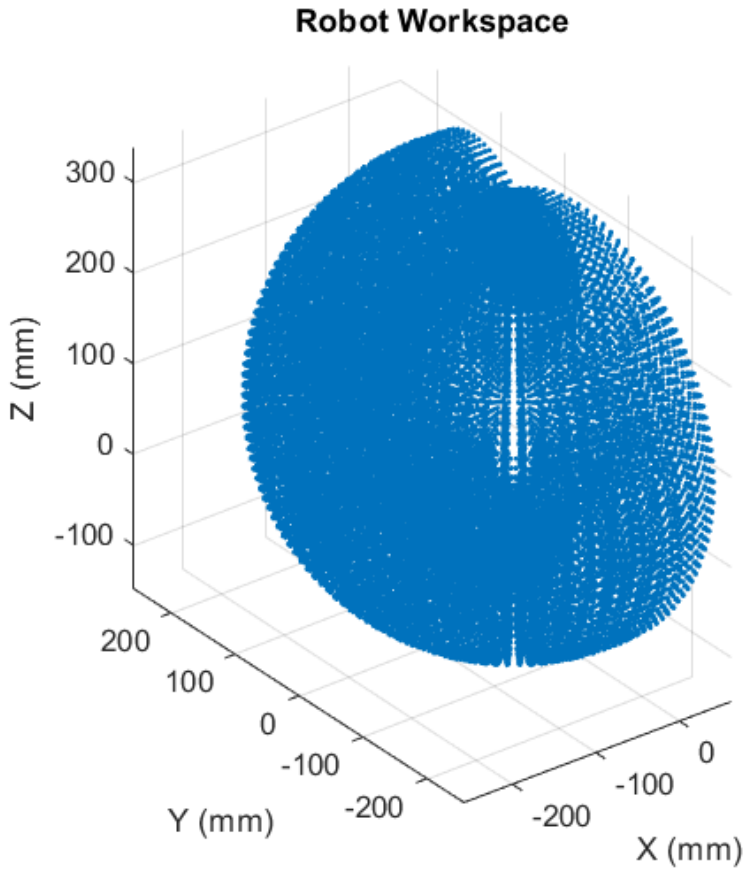


Figure 7: 3D visualization of the workspace for the robot arm

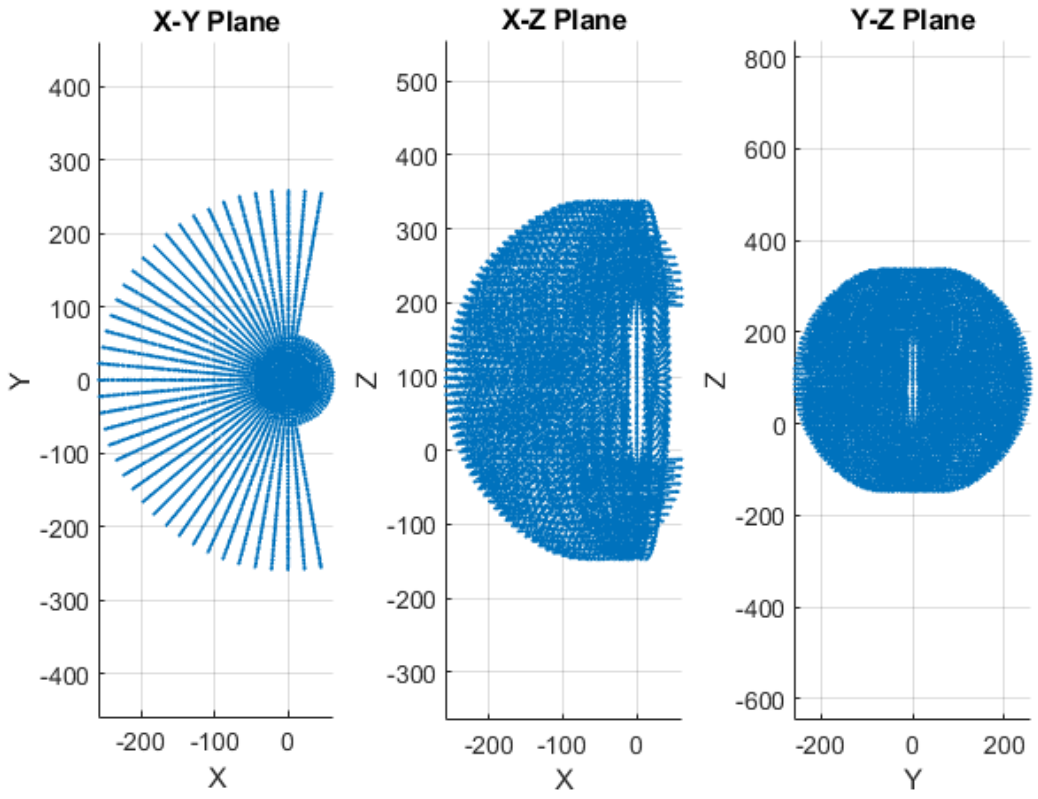


Figure 8: 3 different views of the robot arm's workspace

Kinematics Reconstruction

The derived inverse kinematics equations are then used to validate. The validation strategy is used a set of angles to forward kinematics, which the location in respect to the angles is then found. Using that location to apply inverse kinematics and cross check with the original angle sets. There is a maximum of 4 sets of angles that can be found through inverse kinematics, the angle set with the smallest error to the original is picked. The ideal scenario is that the inverse kinematics solutions are solvable to every location that a set of angles forward kinematics through the same system. The error in location inference must be zero, but the angle can be varied in a few cases because of singularities inherited in trigonometric solutions of inverse kinematics.

The test run verify angle set from -180 degrees to 180 degrees with the spacing of 5 degrees for all 3 joints, which is then generated 389017 test cases for the inverse kinematics. It scores 100% on being able to find at least 1 solution to a given configuration. Position error is approximately 0 mm, which matches the ideal scenario. However, the angular error

distribution is quite concerning if the robot is enforced based on the joint space, instead of task space.

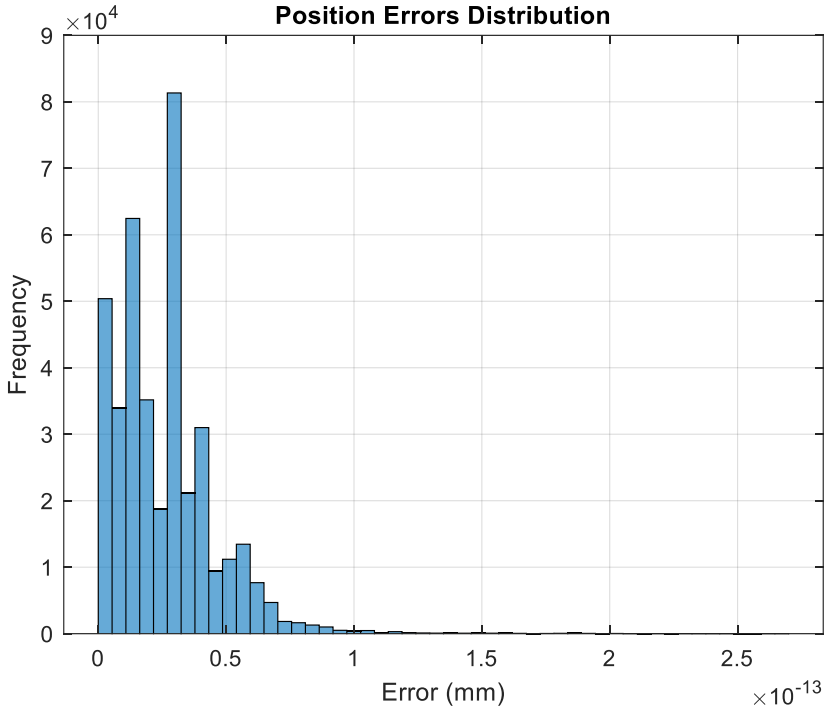


Figure 9: Error distribution on positioning inference using inverse kinematics

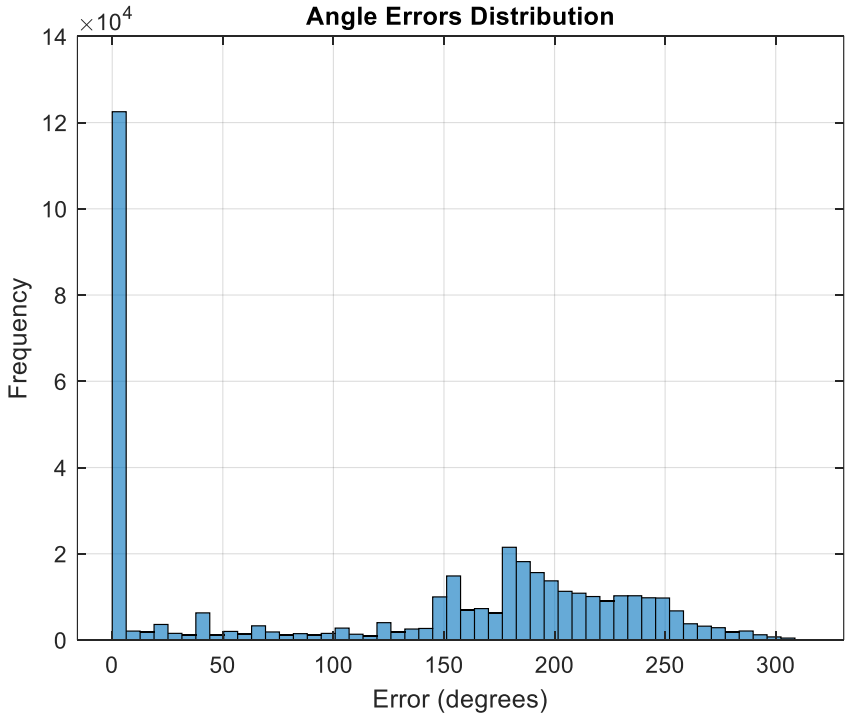


Figure 10: Error distribution on angle selected by the closest to the original angles

Phase 3: Path Generation and Jacobian Matrix

In this section, the robot arm is controlled by two methods: via task space or joint space. Task space control relies on the accuracy of the Jacobian matrix that gives information about the linear velocities of the end effector. Joint space control relies on the knowing of the angles that are within the constraints of joint range and inferable through a targeted location. Both approaches have their own advantages and disadvantages when scaling the problem to pick and place multiple objects in sequence or a specific pattern.

Path Generation

The input for a path generation is a target location on 3D space. The general procedures are given below for each approach.

For task space driven solution:

For every animation frame:

- Step 1: compute the direction between the current position to the target position to infer the linear velocity
- Step 2: compute the analytical Jacobian matrix for the linear velocities of the end effector to get the angular velocities of each component
- Step 3: update the visualization of each component in space
- Step 4 (optional): create traces of end effector of sample points
- Step 5: compute the angular velocity of the end effector
- Step 6: compute the force applied on the end effector during motion
- Step 7: infer the joint torque required based on payload mass and center of mass of each component

After the animation ends, plot and visualize all data collected during the animation.

For joint space driven solution:

- Step 1: perform inverse kinematics. Filter the solution based on defined angle constraints. Then rank the solution based on 2 criteria: minimum position error and minimum joint movement
- Step 2: check the payload (state of the robot arm picking up, holding, placing down test tube)

- Step 3: compute in batches sets of angles that reach the target position using trapezoid method to gain constant angular velocity for each segment
- Step 4: perform the animation. For each frame
    - Perform homogeneous transformations on each component to update position
    - Update visualization of each component in space
    - (optional) Create trace of end effector of sample points
    - Compute the Jacobian matrix numerically (using finite differences) and analytically (using derivative method) and compute the similarity
    - Compute the linear and angular velocity of the end effector from the trajectory
    - Compute the force applied on the end effector during motion
    - Infer the joint torque required based on payload mass and center of mass of each component
- Step 5: plot and visualize all data collected during the animation

These are the data extracted after running the simulation

Locations	Angle (task space) (estimated)	Angle (joint space)
[0,50,100]	[271.2586, -67.8983, 157.0553]	[270.3031, -66.4559, 156.2506]
[150, 150, 100]	[225.1587, -33.5191, 70.4444]	[225.0714, -32.8272, 68.7608]
[150, 150, 0]	[225.0726, -49.7366, 51.9287]	[225.0714, 1.3005, -53.4534]
[150, 150, 100]	[225.0714, -32.7685, 68.7672]	[225.0714, -32.8272, 68.7608]
[68.7060, 192.1789, 100.4706]	[249.9493, -35.6395, 74.9198]	[250.4017, -35.5452, 74.7208]
[-19.3230, 195.1438, 100.4706]	[275.8537, -38.1019, 80.2447]	[-84.2678, 161.3129, 80.2779]
[-96.8593, 161.3129, 100.3521]	[300.9819, -40.6076, 85.5507]	[-58.9370, -40.5876, 85.5075]
[-150, 100, 100]	[326.2601, -43.0145, 90.5202]	[-33.6060, -42.9887, 90.4630]
[-150, 100, 0]	[326.3928, -64.7885, 75.4288]	[-33.6060, 9.1737, -77.2541]
[-150, 100, 100]	[326.3940, -43.0849, 90.4883]	[-33.6060, -42.9887, 90.4630]
[0, 50, 100]	[273.6992, -66.4677, 156.2642]	[270.3031, -66.4559, 156.2506]

Table 2: Locations and angles for each approach

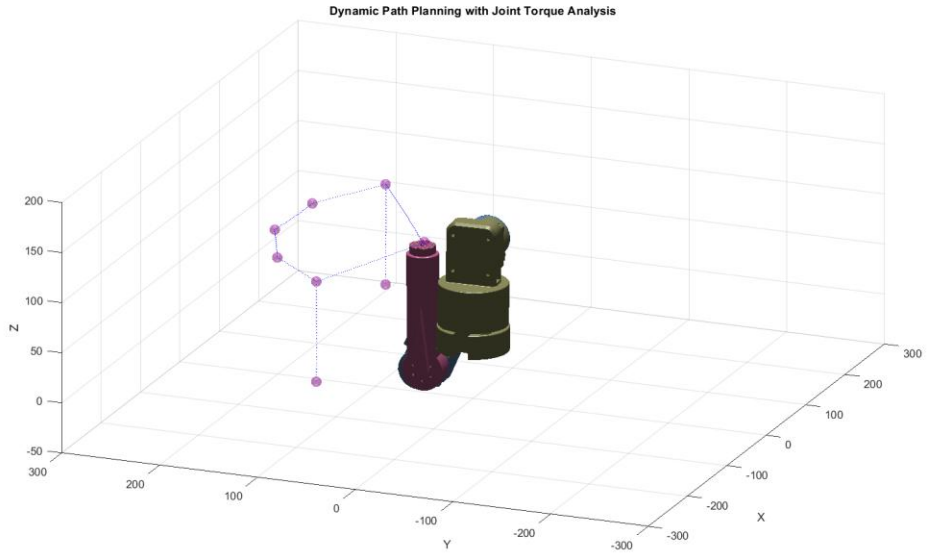


Figure 11: Task space driven trajectory generation

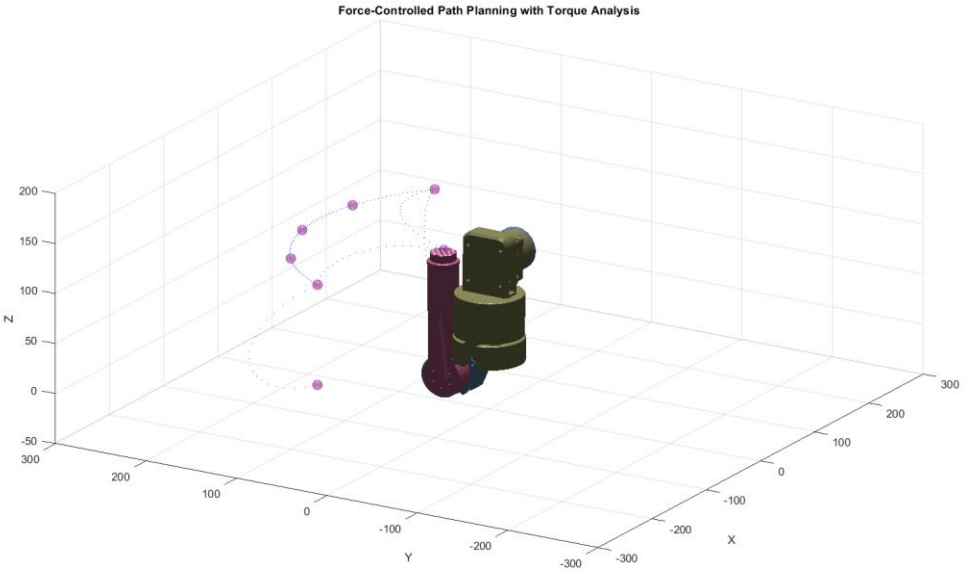


Figure 12: Joint space driven trajectory generation

Trajectory Generation

Task space analysis

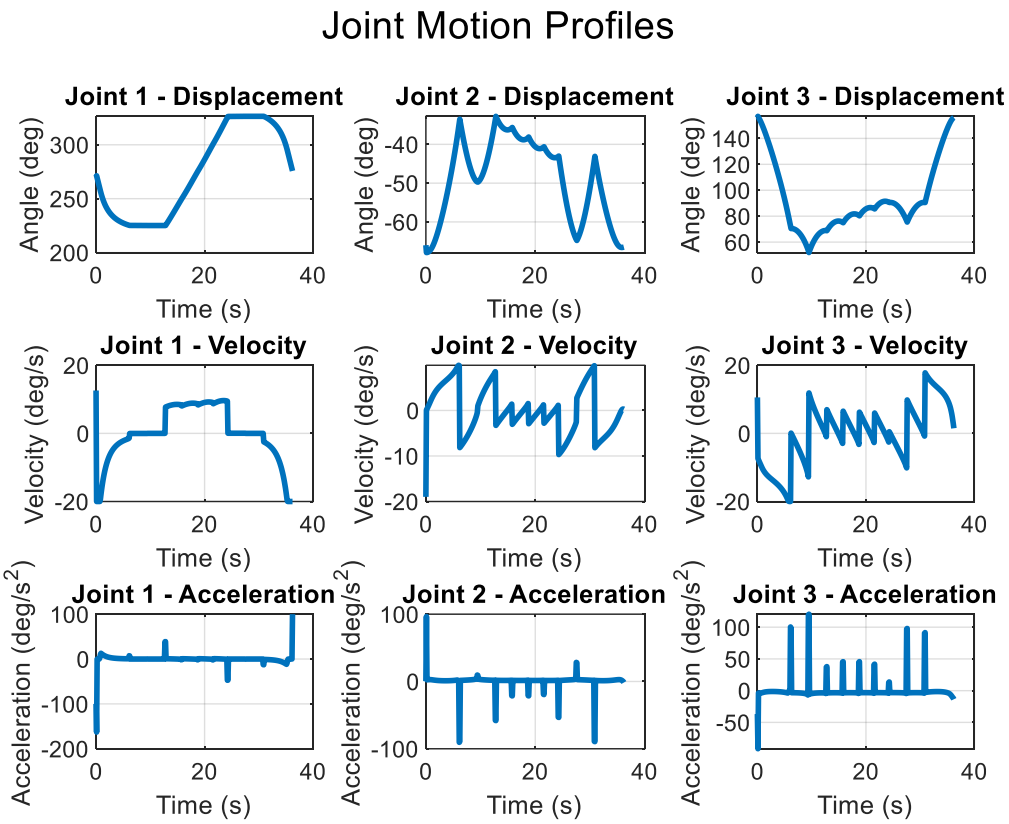


Figure 13: Joint motion profiles (task space driven)

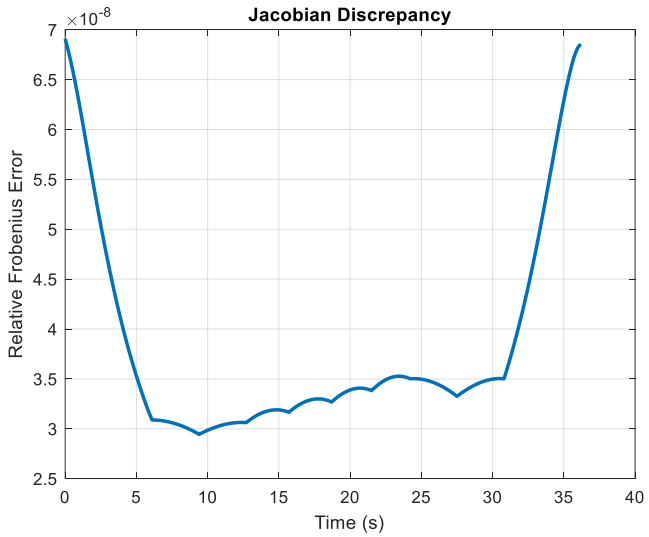
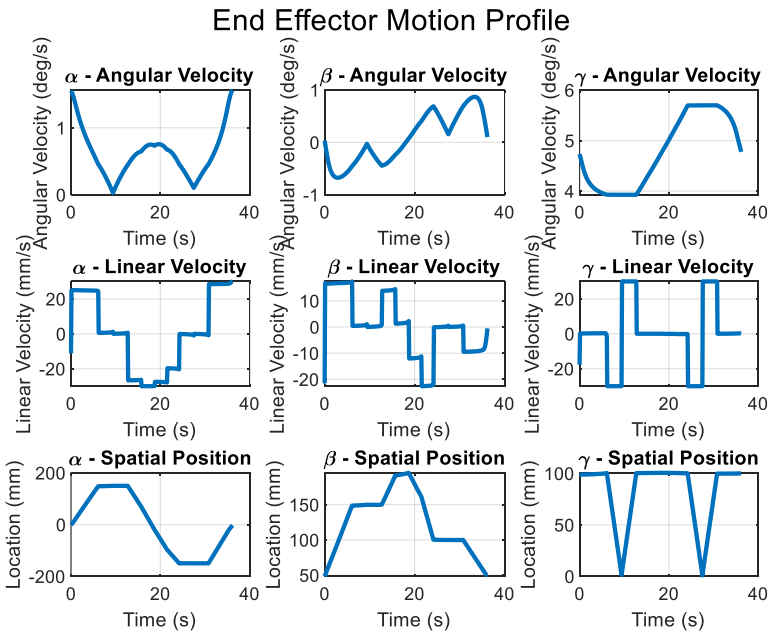


Figure 14: Jacobian similarity score between analytical and geometrical approaches (task space driven)

The task space derives the angular velocity based on the multiplication of the inverse of Jacobian matrix that contains the linear velocity and the linear velocity vector of the end effector. However, the angular velocities are limited to 20 degrees per second to avoid a high spiking when differentiating for acceleration. The linear velocities are shown clearly in Figure 11: Task space driven trajectory generation. The joint motion profile does not clearly how the linear displacements are calculated. In the row of angular velocities graph, there are ripples shown in second 16 and 24. The cause for that behavior stemmed from the fact that the program tries to minimize the positioning error close a targeted location below a preset tolerance. Most of the target locations are effectively a path of a continuous curve that is generated between 2 points of an arc. At the end, the spiking behavior mostly resulted from the program's iterative calculation on Jacobian matrix to feed forward a targeted location.

The analytical Jacobian matrix is used for this calculation as it is less redundant for the program to keep iterating the homogeneous transformation matrix, extracting the rotating axis and positioning vector to compute the cross product. One drawback of using analytical Jacobian is that the derivative is directly tied to the final homogeneous transformation, which results in more work during the process of tuning parameters of kinematic equations. In this run, the relative Frobenius error[1] is extremely low.



[1] Relative Frobenius error can be understand as flattening the matrix into a long vector and then calculating the Euclidean of that vector. Essentially, it is a dimensionless measure of proximity of 2 matrices. Smaller number representing a better approximation of the reference matrix.

Figure 15: End effector motion profile (task space driven)

While angular velocities are effectively capped, remaining well with the 20 degrees per second limit, the linear velocities demonstrate distinct, step-like changes. These abrupt shifts indicate precise and rapid acceleration and deceleration to achieve desired linear speeds, suggesting that the end effector is executing a series of distinct linear movements rather than a continuous, smooth acceleration profile. Combining with a low error rate, it validates the effectiveness of control strategy, including the use of an analytical Jacobian matrix for calculation, even with the inherent complexity of iterative Jacobian calculations and the challenge of tuning kinematic parameters. The low Frobenius error confirms that the system achieves a highly precise and reliable execution of the desired end effector trajectories.

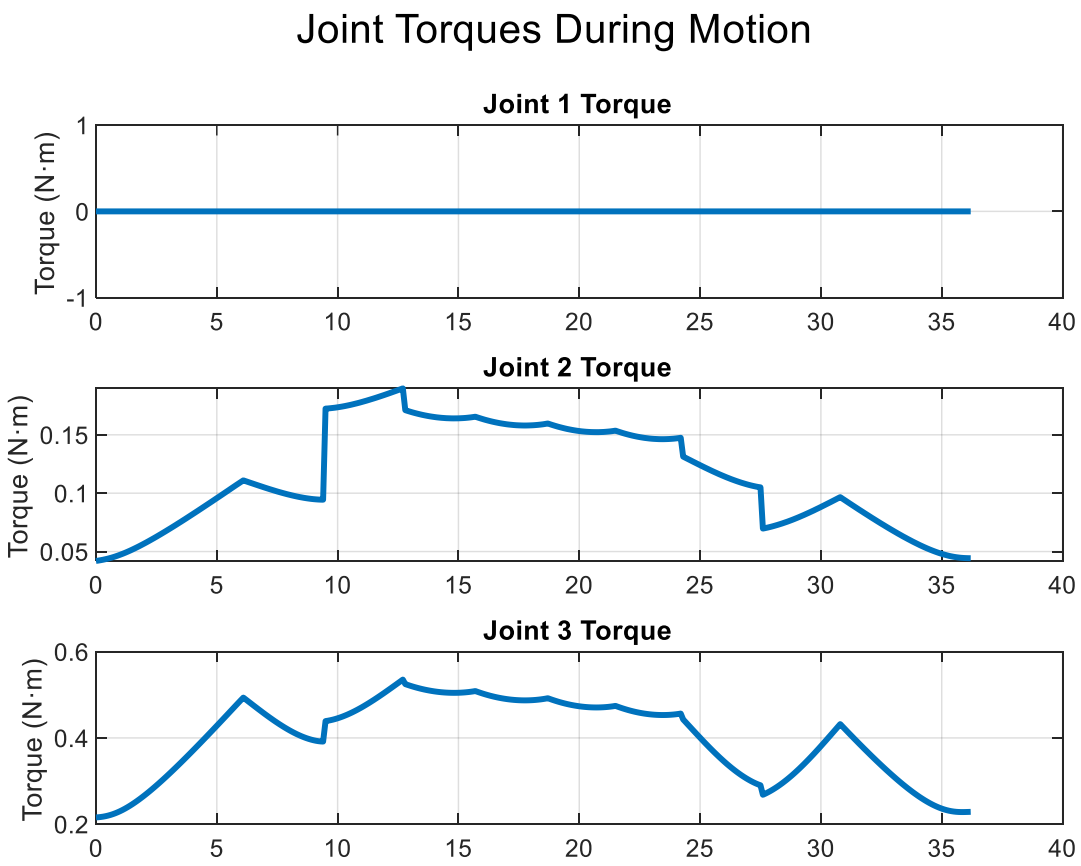


Figure 16: Joint torques during motion (task space driven)

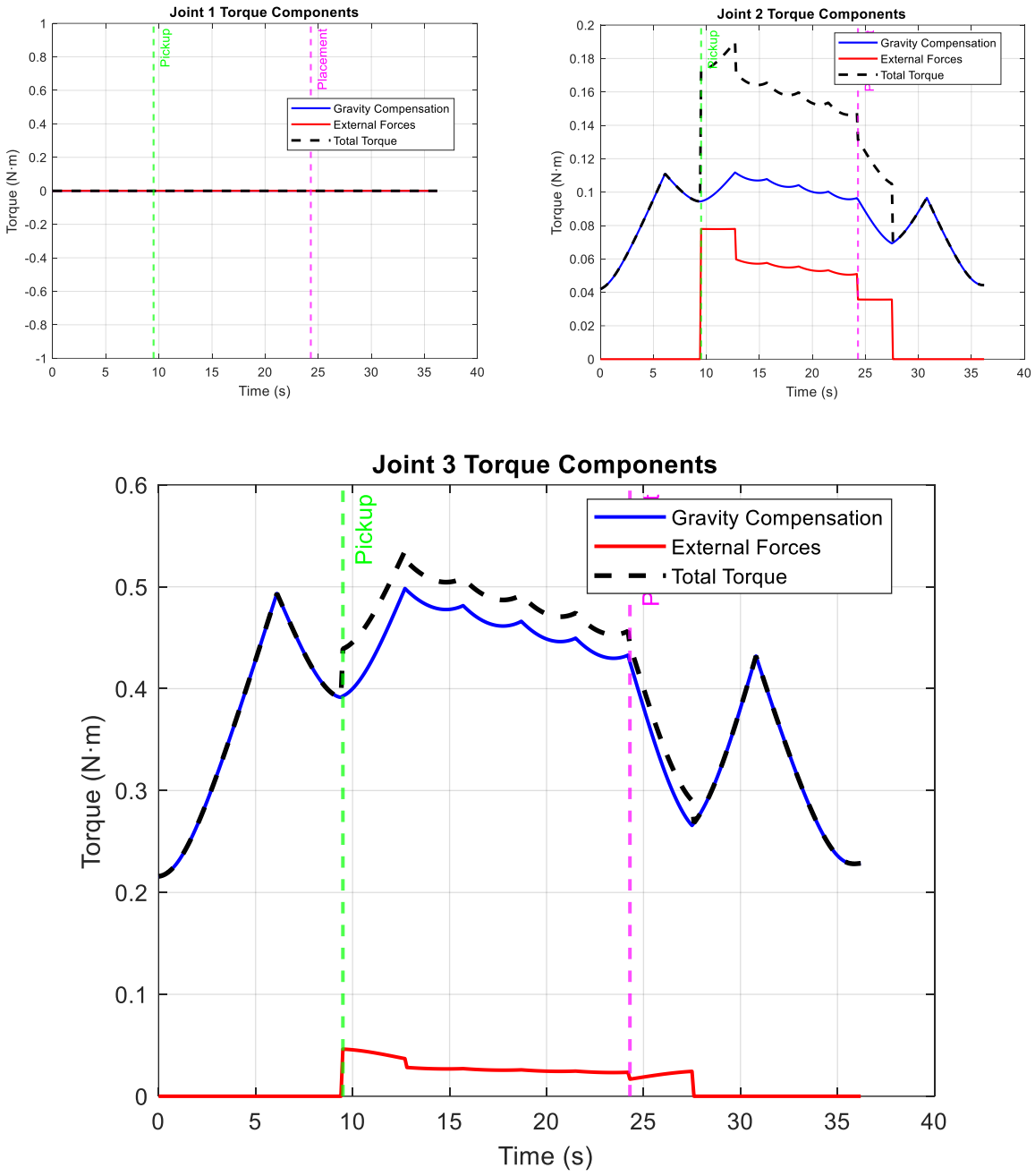


Figure 17: Detail torque analysis (task space driven)

As anticipated by the manipulator's design, Joint 1 (the base) exhibits negligible torque throughout the motion. This is consistent with a vertical rotation axis for this joint, which minimizes gravitational effects and suggests it's not directly involved in force transmission for the specified task.

In contrast, Joint 2 (the middle link) experiences significant torque, with its peak value reaching approximately 0.2 N*m. This measured peak torque is within the stated maximum acceptable torque of 5 N*m for the motor supporting Joint 2's rotation. This indicates a

good match to the mechanical design of the robot arm. The motor can hold more torque than it currently has.

Joint 3 (the end effector link) is the most heavily loaded, showing peak torques around 0.55 N*m. This high torque is primarily driven by the gravitational forces acting on its own link's mass and the added mass of the test tube, particularly during the "Pickup" and "Placement" phases. This 0.55 N*m peak torque for Joint 3 is way lower than the maximum torque capacity of the joint, which suggests the oversize the motor capacity.

To deal with the oversized motor, based on the amount of torque supplied on Joint 2 and 3, modifications to the robot arm's design are necessary. Strategies, such as increasing the maximum allowable loads and shortening the end effector length, can help maximize the usable torques from the motor, which improve overall usability of the robot arm. Furthermore, it's crucial to acknowledge that the length of the links is a critical factor influencing torque, as it creates a larger moment arm for both gravitational and payload forces. As the torque is derived via length and center of mass of component, the torque shown in the analysis underestimates the actual amount of torque that one motor can exert on. The current design does not account for reducers in motor, screws to tighten components together strong, and many more problems with the implementation of gearbox system in the drivers.

Joint space analysis (linear interpolation on angular velocity)

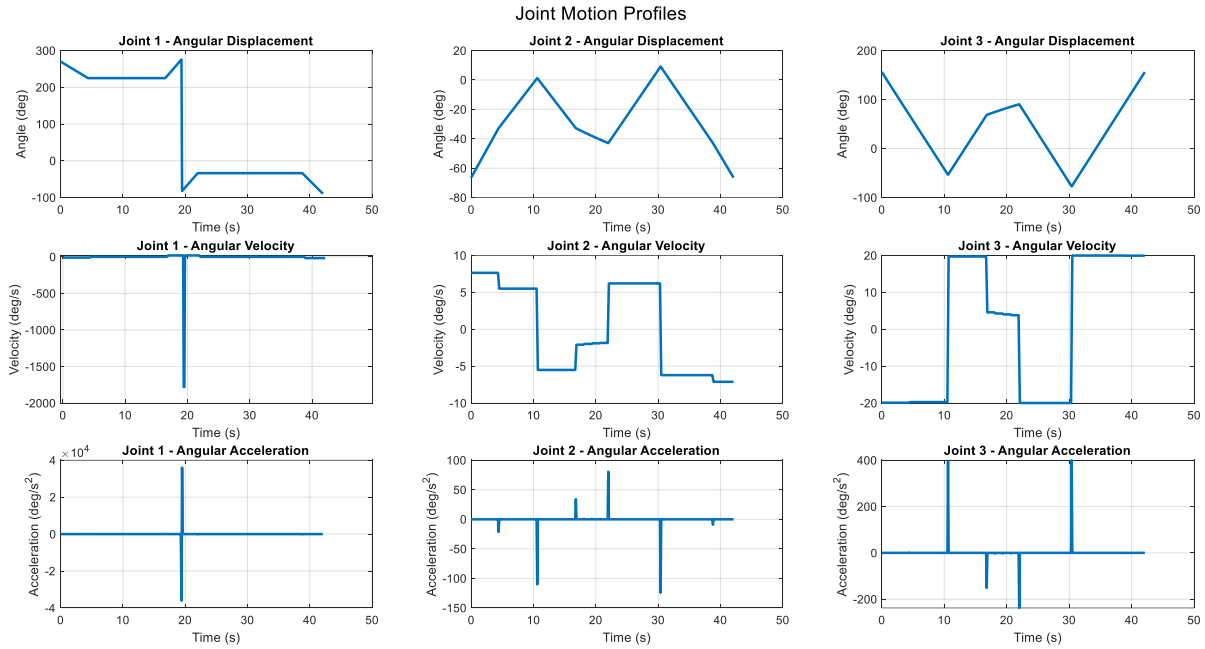


Figure 18: Joint motion profiles (joint space driven)

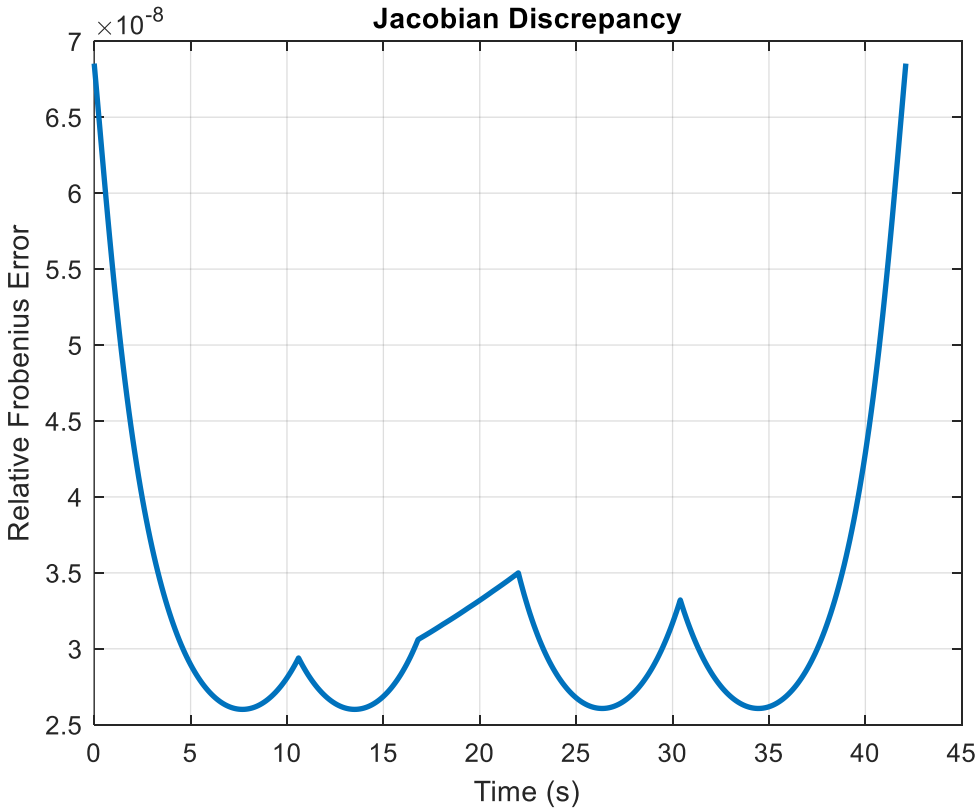


Figure 19: Jacobian similarity score between analytical and geometrical approach (joint space driven)

Generating path via joint space is mathematically redundant since the input is a targeted location. The program is required to use to a set of equations for inverse kinematics, which is then goes through a selection process to choose the best solution for path generation. The path generation relies on the trapezoid to create a constant angular velocity on each joint to reach the target position. Jacobian matrix is then generated by using both cross product and derivative method for comparison after the animation. The linear and angular velocities of the end effector are then calculated through the analytical Jacobian matrix. Torque analysis is then calculated the same way in the task space driven solution. The overall is redundant due to the need to collect data and analyze performance. The process can be dependent on analytical solutions only which provide a faster calculation speed and cheaper memory consumption.

Interpolating between the current angles and the targeted angles yields constant velocities between them but suffer a disadvantage of high pulses of angular acceleration. Physically, that means the motor and the structure are both under a tremendous amount of mechanical stress, which can lead to catastrophic mechanical failure, serve vibrations or the inability of the motor to perform the commanded movement due to insufficient inertial torque capacity. Therefore, the most critical improvement involves developing a new path

generation algorithm that prioritizes smooth and continuous acceleration profiles. To get a smooth acceleration, using higher order trajectory generation methods like cubic splines, quintic polynomials, or S curve profiles, which ensure continuity in velocity and acceleration. The linear interpolation is simpler but leads to acceleration discontinuities.

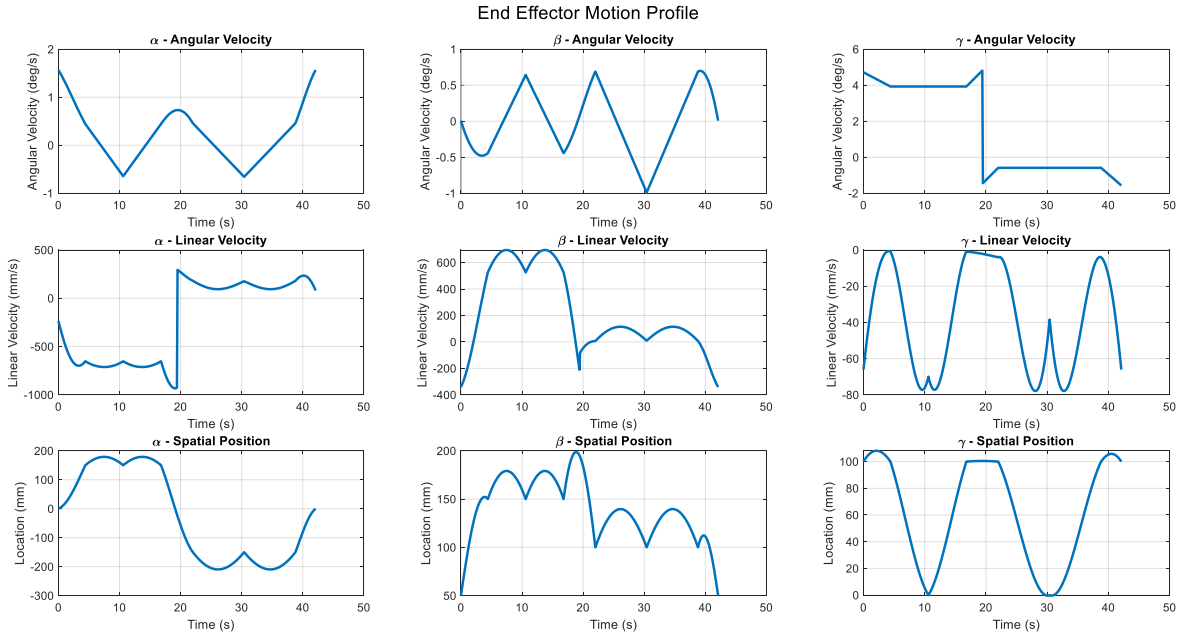


Figure 20: End effector profile (joint space driven)

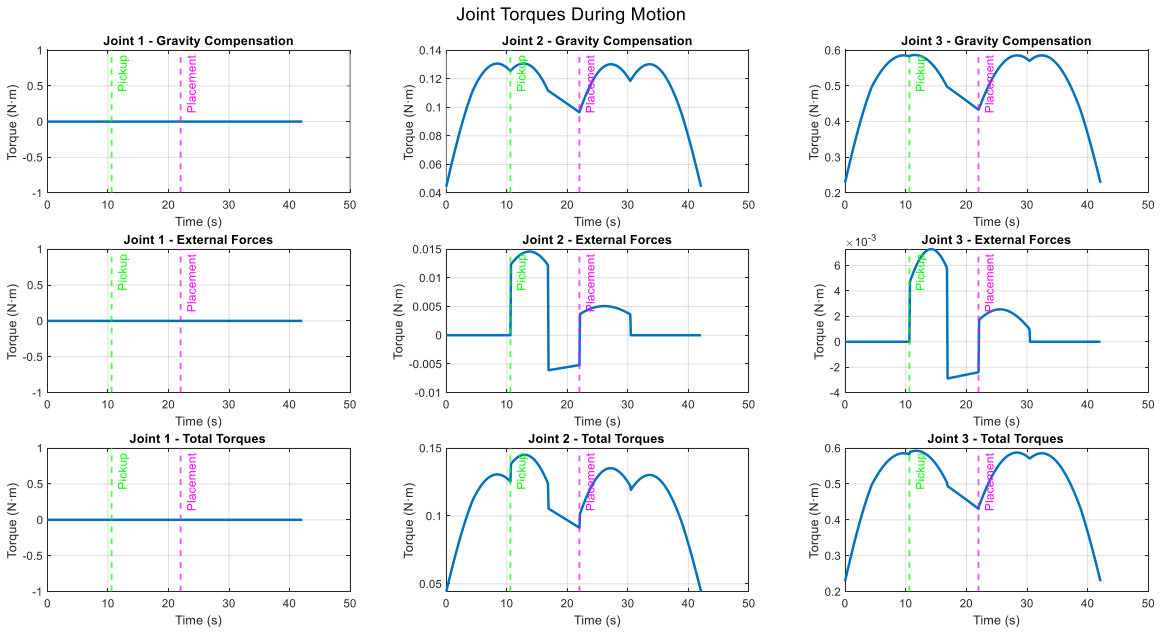


Figure 21: Joint torques during motion (joint space driven)

Joint space analysis (cubic spline and quintic polynomials) for 4-second intervals

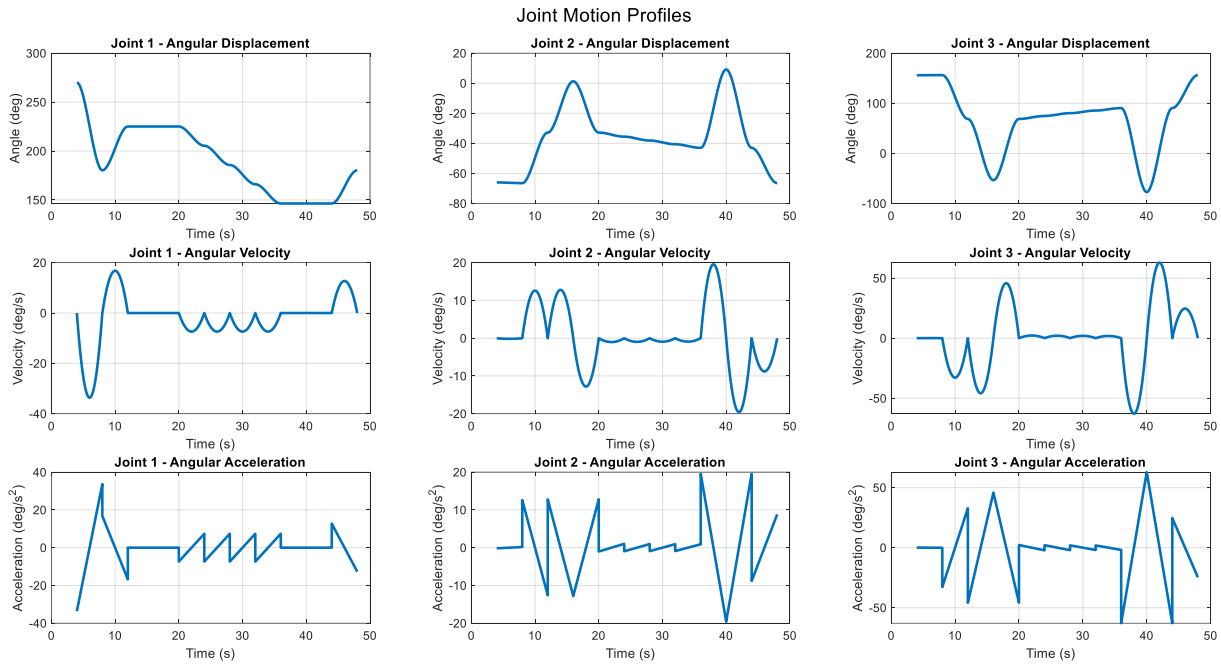


Figure 22: Joint motion profiles (joint space driven) with cubic spline

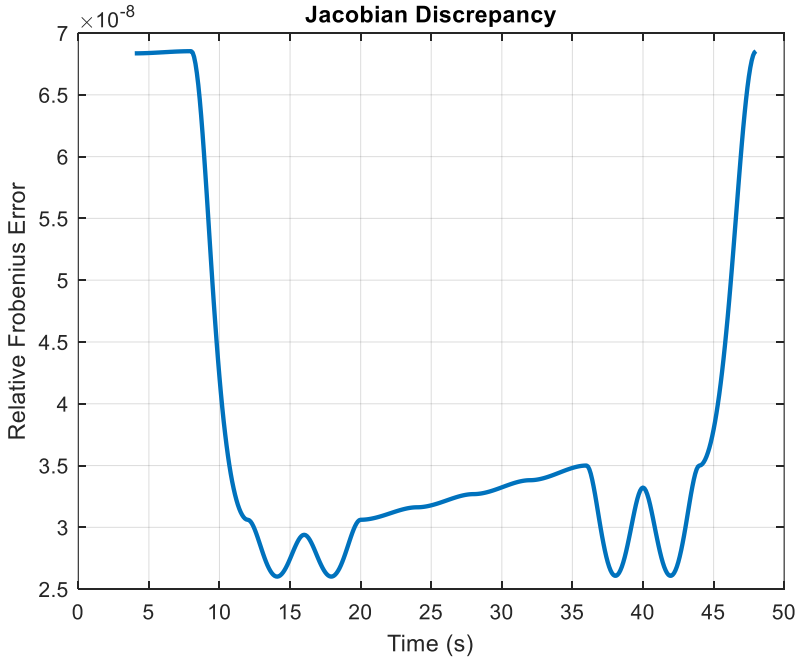


Figure 23: Jacobian similarity score between analytical and geometrical approach (joint space driven) using cubic spline

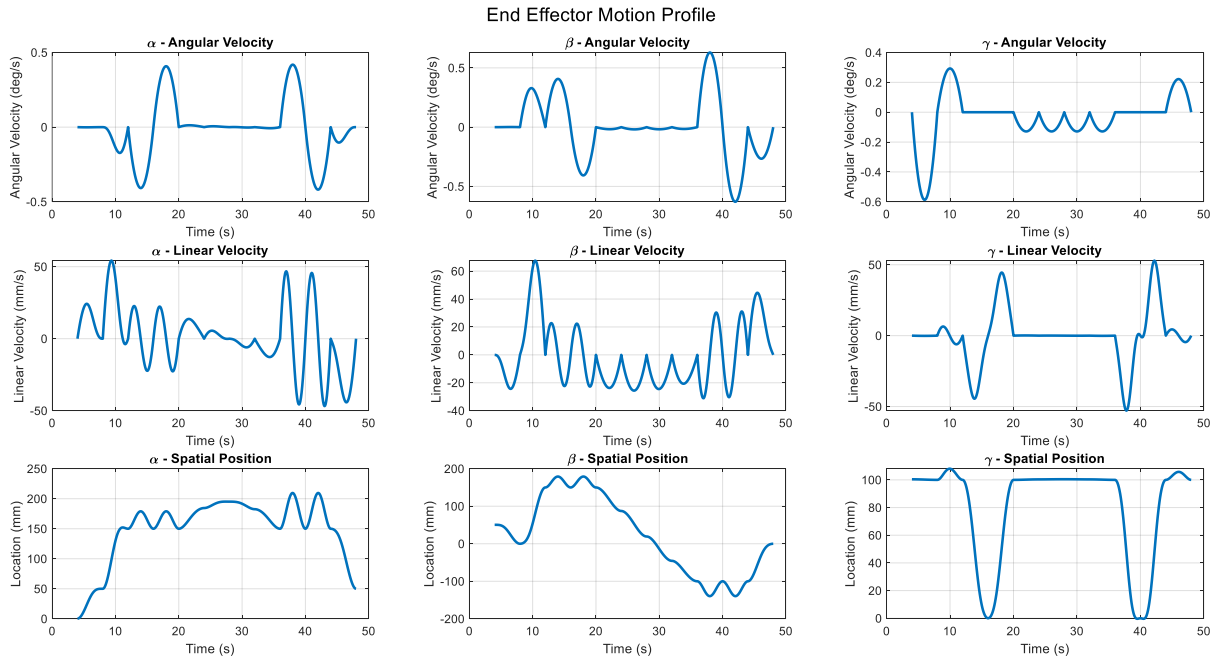


Figure 24: End effector motion profile (joint space driven) using cubic spline

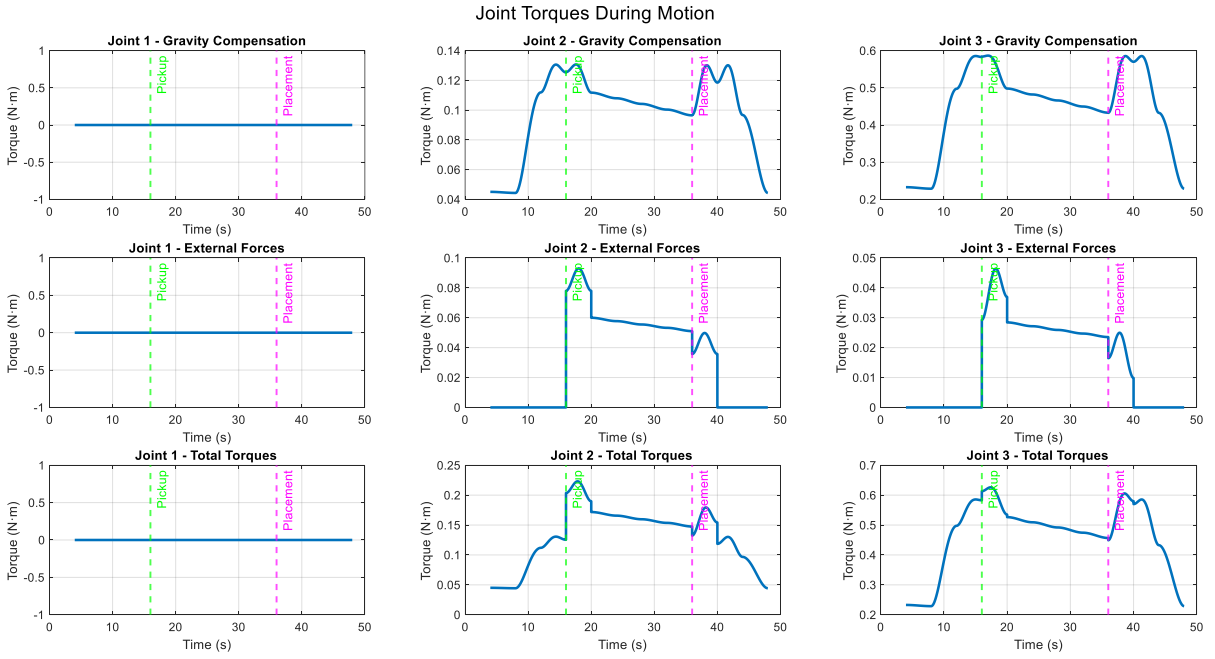


Figure 25: Joint torques during motion (joint space driven) using cubic spline

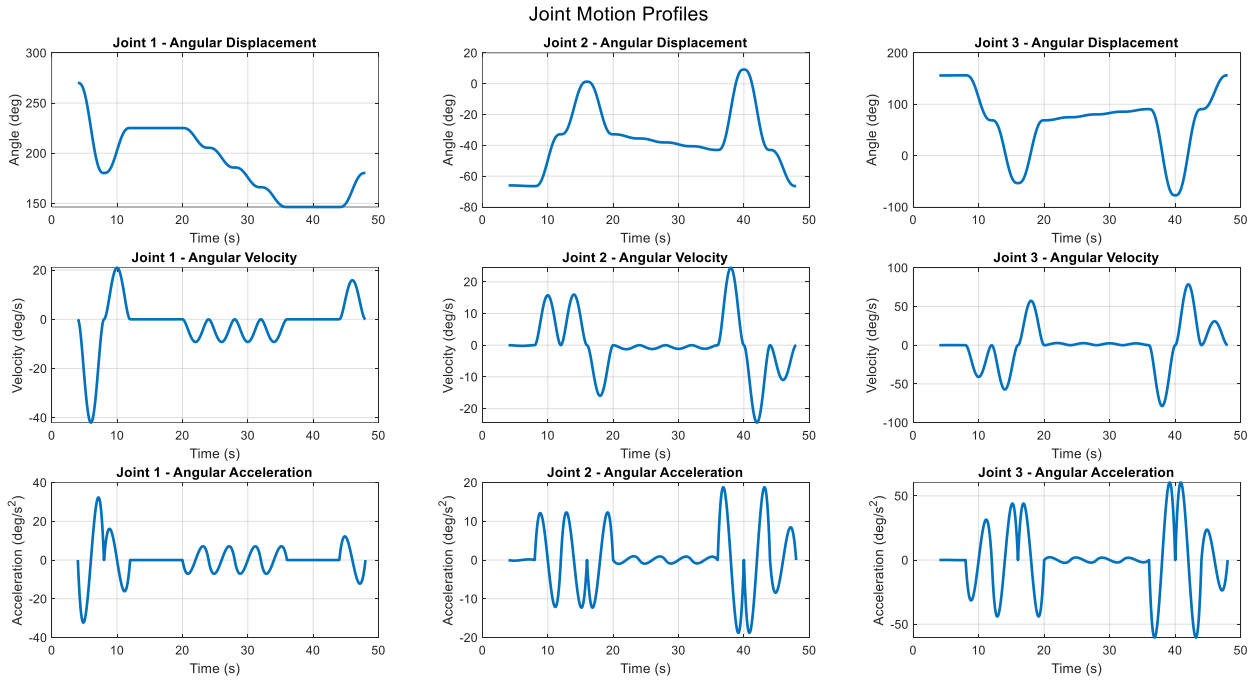


Figure 26: Joint motion profiles (joint space driven) using quintic polynomial

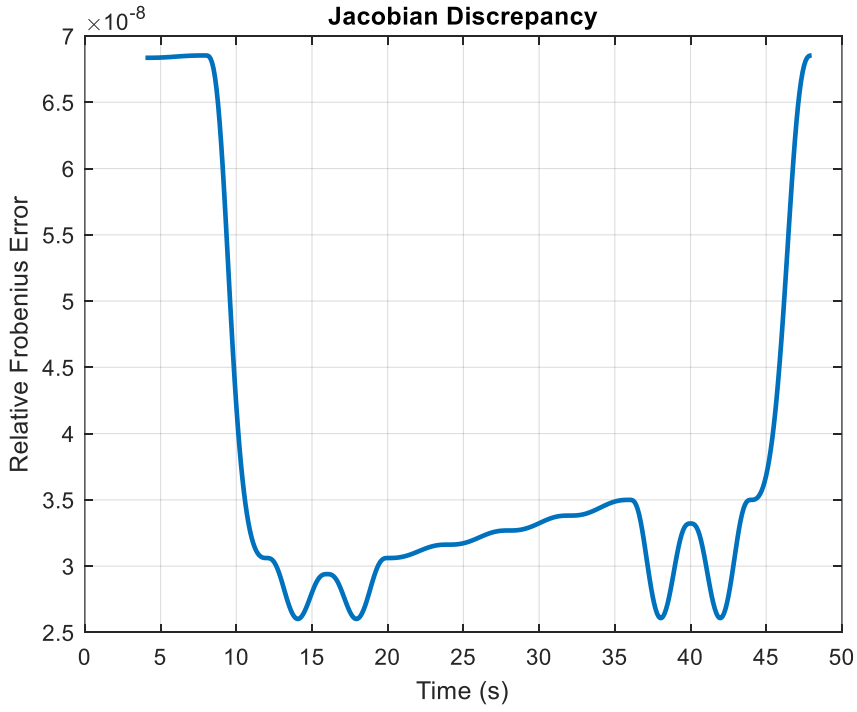


Figure 27: Jacobian similarity score between analytical and geometrical approach (joint space driven) using quintic polynomial

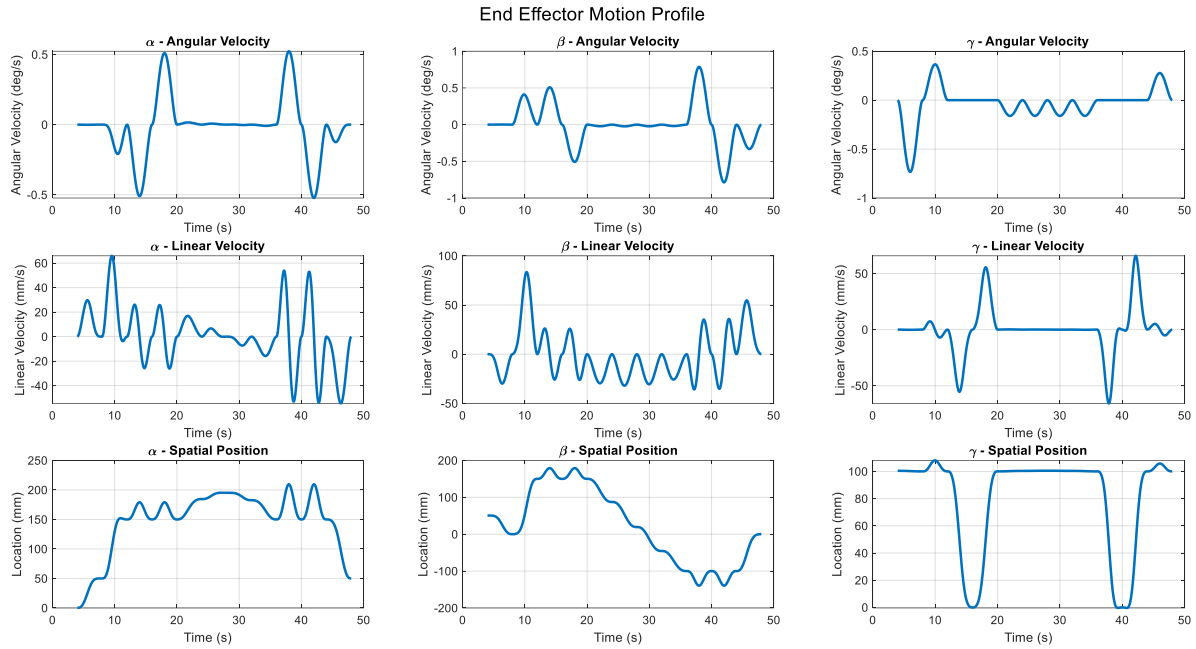


Figure 28: End effector motion profile (joint space driven) using quintic polynomial

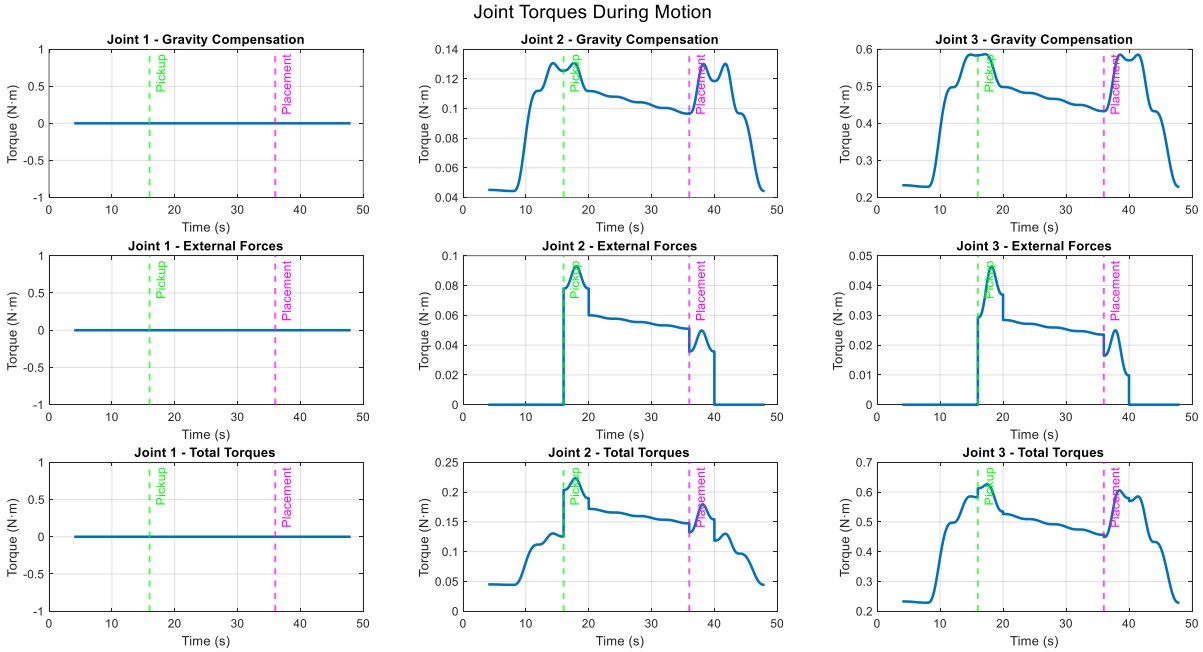


Figure 29: Joint torques during motion (joint space driven) using quintic polynomial

Jacobian Matrix

The Jacobian matrix can be derived from the derivative methods. Refer to Equation 8: Position equations in relation to the angles of joints as the base equations for derivation. The Jacobian matrix for linear velocity of end effector is simply partial derivatives of the position equations respected to the angle of joints. After fully derived and simplified the expression, the equations can be summarized as:

$$J_v = \begin{matrix} -S_1A - 0.2645C_1 & C_1B & C_1D \\ C_1A - 0.2645S_1 & S_1B & S_1D \\ 0 & E & F \end{matrix}$$

Equation 12: Jacobian matrix for linear velocities of end effector

Where:

- $A = 2.8614S_{23} - 126.994C_{23} - 133.3C_2 + 0.5S_2 + 1.3$
- $B = 2.8614C_{23} + 126.994S_{23} + 133.3S_2 + 0.5C_2$
- $D = 2.8614C_{23} + 126.994S_{23}$
- $E = 126.994C_{23} - 2.8614S_{23} - 0.5S_2 + 133.3C_2$
- $F = 126.994C_{23} - 2.8614S_{23}$

Equation 13: Short-hand terms for Jacobian matrix

Alternatively, the shorthand term can be expressed as:

- $A = -E + 1.3$
- $B = D + 133.3S_2 + 0.5C_2$
- $D = 2.8614C_{23} + 126.994S_{23}$
- $E = F - 0.5S_2 133.3C_2$
- $F = 126.994C_{23} - 2.8614S_{23}$

Equation 14: Alternative short-hand terms for Jacobian matrix

The Jacobian matrix that can yield the angular velocities of the end effector can be extracted from the Z axis of the base link (refer to Equation 1: Transformation of base link at base frame), the Y axis of the middle link (refer to Equation 3: Transformation of middle link at base frame), and the Y axis of the end effector link (refer to Equation 5: Transformation of end effector link from base frame), which ultimately yield

$$J_\omega = \begin{matrix} 0 & -S_1 & -S_1 \\ 0 & C_1 & C_1 \\ 1 & 0 & 0 \end{matrix}$$

Equation 15: Jacobian matrix for angular velocities of end effector

In the task space driven solution, the linear velocities of the end effector are determined based on the current location of the end effector and its targeted location. The angular

velocities are inferred by multiplying the inverse matrix of Equation 12: Jacobian matrix for linear velocities of end effector to the linear velocities of the end effector. The inverse technique used in this program is the damped least squares technique to avoid numerical instability

$$\dot{\theta} = J^T(JJ^T + \lambda^2 I)^{-1}\dot{x}$$

Equation 16: Formula for damped least squared inverse matrix techniques

Where $\lambda = 0.1$ and $\dot{x}$ is linear velocities of the end effector.

The full Jacobian matrix to run this robot arm is the combination of Equation 12: Jacobian matrix for linear velocities of end effector and Equation 15: Jacobian matrix for angular velocities of end effector

$$J = \begin{matrix} -S_1A - 0.2645C_1 & C_1B & C_1D \\ C_1A - 0.2645S_1 & S_1B & S_1D \\ 0 & E & F \\ 0 & -S_1 & -S_1 \\ 0 & C_1 & C_1 \\ 1 & 0 & 0 \end{matrix}$$

Equation 17: Full Jacobian matrix for robot arms

Conclusion

This report successfully details the design, modeling, and dynamic analysis of a 3-DOF robotic arm for a specified pick-and-place operation. The work validated the manipulator's functionality through the development and verification of its forward and inverse kinematic models.

The investigation into path generation methods revealed that while both task space and joint space control are viable, the chosen trajectory profile is critical for performance. Simple linear interpolation in joint space, though easy to compute, results in high acceleration pulses that can cause significant mechanical stress. The analysis shows that higher order trajectories, such as cubic splines and quintic polynomials, provide much smoother and more stable motion.

Dynamic analysis of joint torques concludes that the selected motors are oversized for the task, with peak torques reaching only 0.55 N*m, far below the 5 N*m maximum capacity. The mass and length of the middle and end effector links were identified as the primary drivers of this torque, while the base link is acting as a static mass that is locating to the center of mass of the whole robot arm to within the base link’s volume.

Ultimately, identifying the purpose of the design, such as pick and place heavier load or minimize the volume of the robot arm when inactivated, defines the path for further optimization. As the design is done for only demonstrating the behavior of the robot arm in its environment, there is significant rework needed to be done on the design of the motor and its joint arm’s driver mechanism. The simulation also does not account for the gripper design, which plays a larger role in the dynamic analysis of this robot arm. Recommendations on future iterations of the simulation focus on the mitigation of acceleration and velocities pulses, which stemmed from discontinuities of multiple equations and improvement on numerical stability around the discontinuities, namely between near 0 on the right and 360 degrees on the left region in rotation calculation.