EDSU Journal of Engineering and Technology (EDSUJET)

# Review for Computer-Aided Validation of Kinematic Models for a 3-Link Articulated Robot

**Inanemoh Jossy[a,*],Chukwuemeka Chijioke Obasi[b],Bello Oshiomeghie Lawal[c],**
[a]Department of Computer Engineering, Edo State University Iyamho
Corresponding author: Inanemoh Jossy, Email: jraw.inanemoh@gmail.com

**Received: 13 August 2025, Accepted: 23 October 2025, Published: 30 November, 2025**

**K E Y W O R D S**

Lagrange formulation
Robotic manipulator
Kinematic modelling
Dynamic simulation
Model validation
Symbolic computation

**A B S T R A C T**

Building upon the foundational work presented in Dynamics of a 3-Link Manipulator: A Computational Model, this study advances the validation of kinematic models through the rigorous application of the Lagrange formulation. A Python-based simulation framework was developed to examine the dynamic behavior of a 3-link articulated robotic manipulator by deriving joint torques from the Lagrangian equations and comparing them with the kinematic predictions obtained from the model. The validation process evaluates kinematic consistency, torque computation, trajectory tracking, energy consumption, and motion smoothness, thereby extending verification beyond purely geometric accuracy. The findings confirm that the Lagrange formulation provides a reliable basis for validating kinematic models in simulation, offering a stronger link between geometric correctness and dynamic performance, and establishing a foundation for future studies in adaptive and learning-based control of robotic manipulators.

## 1. INTRODUCTION

In many fields and applications where precise and exact performance is required, robotic arms perform better than human labour (M. Farman, 2018). The stage for robotic arms is a set of coupled links joined by joints that provide relative movement of the adjacent links. A robotic arm's joints can be either prismatic, which produces linear motion, or revolute, which produc es rotational motion. Kinematic chains, which are made up of joints and links, give the robotic arm its structural support (Gong, 2020).

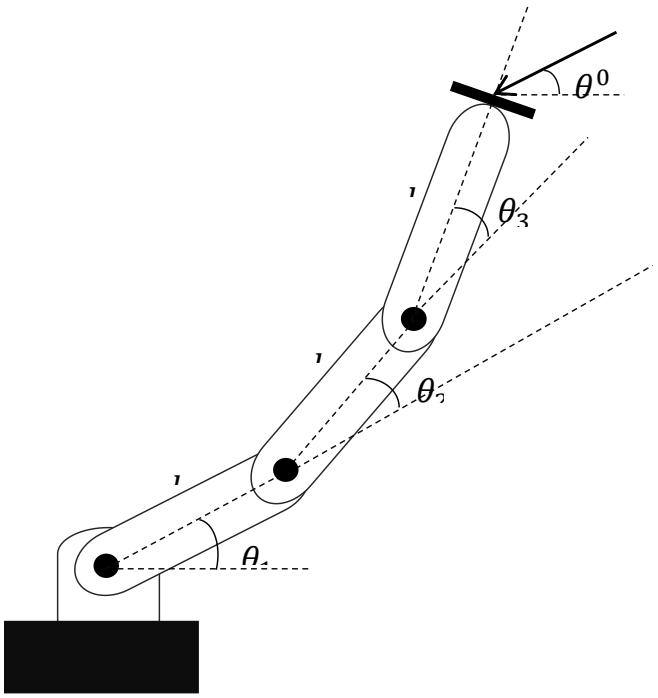


Figure 1.1: Free Body diagram of 3-link robot

The number of joints (axes) on a robotic arm reveals its range of motion (degrees of freedom), while the type of joints dictates its type and potential uses. The manipulator type discussed in this study is a three link or 3-DOF articulated robotic arm that has three revolving joints built into its structure.

The first joint ($\theta_1$) in Fig. 1.1 permits the robotic arm to rotate around the vertical axis (z-axis), while the other two joints ($\theta_2$ and $\theta_3$) permit rotation around the horizontal axis (x-axis)

Kinematics of robotic arms is the study of robotic arm motion without taking into account the forces causing the motion (Banga, Jan. 2022). Using the given joint angles, kinematics is used to determine the end effector's position. On the other hand, it is employed to determine the manipulator's joint angles based on a preferred end effector. Forward kinematics is the term for the former, whereas inverse kinematics is the term for the latter.

In order to precisely and accurately manage the positioning of its end effector, this study focuses on computer-aided validation of kinematic models for a 3-link articulated robotarm shown in Fig. 1.1

With the advent of artificial intelligence and control theory, modern robotic arms now possess enhanced degrees of freedom (DOFs), enabling them to perform complex tasks with high precision. The transition from hydraulically driven systems to electromechanical and pneumatic actuators has further enhanced their efficiency and safety in human environments (Younis, M. A., 2020).

In industrial manufacturing, robotic manipulators have revolutionized production lines. For example, automotive assembly plants employ six-axis articulated arms for welding, painting, and component installation with micron-level precision (Collins, J., et al., 2021), while the electronics industry relies on SCARA robots for high-speed circuit board assembly with repeatability of ±0.02mm (Muelaner, J. E., et al., 2018). The healthcare sector

benefits from surgical robots like the da Vinci platform, which enables minimally invasive procedures with sub-centimeter incisions (Taylor, R. H., & Stoianovici, D., 2020). Beyond these domains, robotic manipulators are increasingly deployed in agriculture for delicate fruit harvesting (Bac, C. W., et al., 2022), space exploration for extravehicular operations (Wu, L., et al. , 2020), and underwater maintenance in harsh environments (Santhakumar, M., & Asokan, T., 2019).

Kinematic modeling, which relates joint configurations to end-effector poses, is the bedrock of robotic motion planning. It encompasses forward kinematics (FK) for end-effector positioning and inverse kinematics (IK) for joint trajectory synthesis (Kucuk, S., & Bingul, Z., 2016). Traditional methods like the Denavit-Hartenberg (DH) convention assume ideal conditions rigid links, perfect alignment, and deterministic motion yet real-world complexities such as joint compliance and structural flexibility introduce discrepancies (Collins, J., et al, 2020).These challenges are exacerbated in high-degree-of-freedom (DOF) systems, where computational overhead and dynamic coupling complicate real-time control (Villani, L., et al., 2019).

To address these limitations, this study employs the Lagrange formulation, an energy-based approach that captures kinetic and potential energy interactions to derive dynamic equations of motion (Shaik, R., 2021). For a 3-link articulated manipulator, this method enables precise simulation of joint torques, energy consumption, and trajectory tracking under varying operational scenarios. Unlike black-box machine learning techniques, this project emphasizes first-principles modeling, leveraging Python's SymPy for symbolic derivation, NumPy for numerical analysis, and Matplotlib for visualization to validate kinematic and dynamic behavior.

As robotic systems grow in complexity, the need for accurate validation of these dynamic models becomes critical. This study focuses on validating the kinematic behavior of a 3-link articulated robotic manipulator using the Lagrange formulation. Unlike black-box machine learning approaches, this project emphasizes classical mechanics to simulate, analyze, and verify the manipulator's motion and torque characteristics under various operational scenarios.

Moreover, modern simulation tools such as Python's SymPy, NumPy, and Matplotlib provide powerful platforms for symbolic derivation, numerical computation, and visualization of robot dynamics. These tools allow researchers to analyze how different configurations and torque inputs influence the motion trajectory, energy usage, and overall system stability. They also support rapid prototyping and iterative development without the risks and costs associated with physical testing.

This research builds upon this foundation by employing the Lagrange formulation to not only derive the equations of motion for the 3-link manipulator but also to simulate its behavior across different motion trajectories. The study aims to assess how accurately the model simulate joint torques and how effectively it manages energy consumption and ensures smooth motion. Such validations are very important, especially in fields like precision manufacturing, where small deviations can lead to significant product defects or operational inefficiencies.

## 2. VALIDATION OF KINEMATIC MODELS FOR A 3-LINK ARTICULATED ROBOTIC MANIPULATOR USING LAGRANGE FORMULATION

In (Obasi et al., 2020), forward kinematics involves determining the position of a robot's end-effector from given joint parameters (joint angles and link lengths). For a planar manipulator with three revolute joints (3R planar manipulator), the computation is carried

out in two dimensions (X-Y plane), assuming all links rotate about the z-axis perpendicular to the plane of motion.

Let the three joint angles be denoted by $\theta_0, \theta_1, \theta_2$, and the corresponding link lengths as $L_0, L_1, L_2$. The position of the end effector $(x, y)$ can be derived by summing the vector contributions of each link.

**2.2 Simulation Procedure**

The geometric derivation as proposed by (Obasi et al., 2020), each link contributes to the position of the end-effector by rotating its length along the sum of all previous joint angles. For link 0 joint,

$$x_0 = L_0 \cos(\theta_0)\ ,\ \ y_0 = L_0 \sin(\theta_0) \quad 2.10$$

For link 1 joint,

$$x_1 = x_0 + L_1 \cos(\theta_0 + \theta_2), y_1 = y_1 + L_2 \text{ sine}(\theta_1 + \theta_1) \quad 2.11$$

For link 3 joint (End Effector),

$$x = x_2 + L_2 \cos(\theta_0 + \theta_1 + \theta_2)\ , y = y_2 + L_2 \sin(\theta_0 + \theta_1 + \theta_2) \quad 2.12$$

Substituting from above;

$$x = L_0 \cos(\theta_0) + L_1 \cos(\theta_0 + \theta_2) + L_2 \cos(\theta_0 + \theta_1 + \theta_2) \quad 2.13$$

$$y = L_0 \cos(\theta_0) + L_1 \cos(\theta_0 + \theta_1) + L_2 \sin(\theta_0 + \theta_1 + \theta_2) \quad 2.14$$

According (Obasi et al., 2020), these are the standard forward kinematic equations for a 3R planar robot. They are essential for trajectory planning and control in robotic systems. The derivation of the forward kinematics of 3R manipulators has been fundamental to the control of robotic systems. As shown in (Obasi et al., 2020), the computational modeling of such kinematics aids in predicting the end-effector location in real time, which is critical for tasks like pick-and-place, drawing, and welding in constrained environments. (Benitti, 2021) also reinforces this model by illustrating its implementation using homogeneous transformation matrices in the study of robotic arm motion. Inverse kinematics (IK) deals with the computation of the joint angles required for a robotic arm to reach a desired position in space. For a three-link planar manipulator, the goal is to determine the joint angles $\theta_0, \theta_1, \theta_2$ given the end-effector position $(x, y)$ and known link lengths $L_0, L_1, L_2$ (Obasi et al., 2020). To simplify the derivation, we assume the third link contributes only to the orientation of the end-effector, and we focus on finding $\theta_0$ and $\theta_1$ based on the position $(x, y)$ of the wrist point (end of second link), then solve for $\theta_2$.

Computing for wrist position, if the end-effector's orientation angle $\phi$ is known (relative to the base), the position of the wrist (joint between second and third link) is,

$$x_w = x - L_2 \cos(\phi) \quad 2.15$$

$$y_w = y - L_2 \sin(\phi) \quad 2.16$$

Applying geometric relations,

The distance from the base to the wrist is given as

$$r = \sqrt{x_w^2 + y_w^2} \quad 2.17$$

$$\alpha = \cos^{-1}\left(\frac{L_1^2 + r^2 - L_2^2}{2L_1 r}\right) \quad 2.18$$

$$\beta = \tan^{-1}\left(\frac{y_w}{x_w}\right) \quad 2.19$$

Then,

$$\theta_1 = \beta - \alpha \quad 2.20$$

Using the cosine law,

$$\theta_2 = \cos^{-1}\left(\frac{L_1^2 + r^2 - L_2^2}{2L_1 L_2}\right) \quad 2.21$$

Depending on the configuration (elbow-up or elbow-down), $\theta_2$ may be,

$$\theta_2 = \pi - \cos^{-1}\left(\frac{L_1^2 + r^2 - L_2^2}{2L_1 L_2}\right) \quad 2.22$$

For the end effector, the joint angle can be computed by the orientation of the end-effector provides,

$$\theta_2 = \emptyset - (\theta_0 + \theta_1) \quad 2.23$$

The Lagrangian $L$ is defined as the difference between kinetic and potential energies:

$$L = T - V \quad 2.24$$

Using the Euler-Lagrange equation:

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{q}_i}\right) - \frac{\delta L}{\delta q_i} = \tau_i \quad 2.25$$

**2.3 Kinetic and Potential Energies**

The Kinetic and Potential Energies. Let $m_1, m_2, m_3$ be the masses of the links, and the acceleration due to gravity respectively.

Kinetic Energy (T)

$$T = \frac{1}{2}m_1\left(v_{x1}^2 + v_{y1}^2\right) + \frac{1}{2}m_2\left(v_{x2}^2 + v_{y2}^2\right) + \frac{1}{2}m_3\left(v_{x3}^2 + v_{y3}^2\right) \quad 2.26$$

Potential Energy (V)

$$V = m_1 g y_1 + m_2 g y_2 + m_3 g y_3 \quad 2.27$$

The work of (Obasi et al., 2020) offers a practical implementation of inverse kinematics using a geometric method. Their derivation based on Pythagoras' and cosine laws allows the robot to calculate required joint angles for a known task-space location. Such computations are vital in motion planning and robotic path optimization. (Ben-Ari) Ben-Ari and Mondada (2018) further explain how the analytical approach supports real-time robotic control in constrained environments supports better validation of kinematic equations and enables control design, making it more robust for real-world applications.

**2.91 Lagrangian Modeling of a 3-Link Robotic Manipulator**

A planar 3-link articulated robotic manipulator consists of three rigid links connected by revolute joints. Let the generalized coordinates be:

$$\theta_1(t), \theta_2(t), \theta_3(t) \quad 2.28$$

Where $\theta_i(t)$ is the time-dependent angle of the $i^{th}$ joint.

Kinematic Modeling: Center of Mass and Velocities.

Let each link $i$ have:

length $l_i$, mass $m_i$ center of mass located at the midpoint.

$$x_1 \frac{l_1}{2}\cos(\theta_1) \quad 2.29$$

$$x_2 = l_1\cos(\theta_1) + \frac{l_2}{2}\cos(\theta_1 + \theta_2) \quad 2.30$$

$$x_3 = l_1\cos(\theta_1) + l_2\cos(\theta_1 + \theta_2) + \frac{l_3}{2}\cos(\theta_1 + \theta_2 + \theta_3) \quad 2.31$$

$$y_1 = \frac{l_1}{2}\sin(\theta_1) \quad 2.32$$

$$y_2 = l_1\sin(\theta_1) + \frac{l_2}{2}\sin(\theta_1 + \theta_2) \quad 2.33$$

$$y_3 = l_1\sin(\theta_1) + l_2\sin(\theta_1 + \theta_2) + \frac{l_3}{2}\sin(\theta_1 + \theta_2 + \theta_3) \quad 2.34$$

The linear velocities of the centers of mass are obtained via time derivatives, or from symbolic derivation using angular velocities:

$$v_{x1} = -\omega_1 . \frac{l_1}{2}\sin(\theta_1) \quad 2.35$$

$$v_{y1} = \omega_1 . \frac{l_1}{2}\cos(\theta_1) \quad 2.36$$

$$v_{x2} = -\omega_1 l_1 \sin(\theta_1) - (\omega_1 + \omega_2) . \frac{l_2}{2}\sin(\theta_1 + \theta_2) \quad 2.37$$

$$v_{y2} = -\omega_1 l_1 \cos(\theta_1) + (\omega_1 + \omega_2) . \frac{l_2}{2}\cos(\theta_1 + \theta_2) \quad 2.38$$

$$v_{x3} = -\omega_1 l_1 \sin(\theta_1) - (\omega_1 + \omega_2) . l_2 \sin(\theta_1 + \theta_2) - (\omega_1 + \omega_2 + \omega_3) . \frac{l_3}{2}\sin(\theta_1 + \theta_2 + \theta_3) \quad 2.39$$

$$v_{y3} = \omega_1 l_1 \cos(\theta_1) + (\omega_1 + \omega_2) l_2 \cos(\theta_1 + \theta_2) - (\omega_1 + \omega_2 + \omega_3) . \frac{l_3}{2}\cos(\theta_1 + \theta_2 + \theta_3) \quad 2.40$$

### 3.1.1 Modeling Assumptions

To simplify the modeling process while ensuring accuracy, the following assumptions were made. The manipulator links were treated as rigid bodies with uniform mass distribution, and the joints were modeled as ideal and frictionless, neglecting effects such as backlash and elasticity. External disturbances such as air resistance and unmodeled loads were not considered. The manipulator was assumed to operate in a planar configuration, restricting its motion to two dimensions.

### 3.1.2 System Modeling of the Manipulator

The manipulator was modeled as a 3-link planar articulated system with revolute joints; each characterized by parameters such as link length, mass, inertia, and joint variables. The Denavit–Hartenberg (DH) convention was employed to establish the kinematic representation, which provided the mathematical foundation for describing joint transformations and end-effector positions. Both forward and inverse kinematic formulations were considered to ensure that trajectories could be defined for simulation and subsequently validated against dynamic results.

The physical parameters of the manipulator are presented in Table 3.1, which summarizes the link lengths, masses, inertias, and allowable joint ranges.

**Table 3.1: Parameters of the 3-Link Manipulator**

| Parameters | Link 1 | Link 2 | Link 3 |
|---|---|---|---|
| Length (m) | 0.40 | 0.35 | 0.30 |
| Mass (kg) | 1.20 | 1.00 | 0.80 |
| Inertial$(kg/m^2)$ | 0.05 | 0.04 | 0.03 |
| Jojnt Rang ($^0$) | -180 to +180 | -180 to +180 | -180 to +180 |

A schematic representation of the manipulator is provided in Figure 1, showing the revolute joints, link dimensions, and coordinate frames.

### 3.1.4 Simulation Framework and Implementation

A Python-based simulation framework was developed to implement the derived equations. The simulation environment was designed to compute joint torques for predefined motion trajectories, including linear and circular end-effector paths. Numerical integration methods, specifically the Runge–Kutta 4th-order (RK4) method with a fixed step size of 0.01 s, were used to solve the dynamic equations.

The following implementation conditions were applied:

i. Initial joint velocities were set to zero.
ii. Joint torque limits were constrained to ±10 Nm.
iii. Joint angle ranges were restricted between –180° and +180°.

The workflow of the simulation process is presented in Figure 3.1, illustrating steps from defining manipulator parameters to validation of dynamic results against kinematic predictions.

Torque analysis formed a critical part of the methodology, as it allowed for the comparison of torque requirements across different simulated trajectories. For each joint, torque variations were recorded and examined to assess their consistency with the manipulator's kinematic predictions. This process highlighted how different trajectory types influenced actuator effort and exposed any deviations between dynamic and kinematic torque estimations.

### 3.1.5 Validation Techniques

Validation was performed by comparing results from the dynamic model against kinematic predictions and benchmarks from literature. Four complementary techniques were

employed in this process. First, a trajectory consistency check was carried out by comparing the end-effector paths obtained from the kinematic and dynamic models to ensure geometric accuracy. Second, a torque deviation analysis was conducted to evaluate the differences between the computed torque values of the two models. Third, a benchmark comparison was implemented by cross-checking the torque magnitudes and profiles obtained in this study with those reported in published works, thereby verifying the reliability of the model. Finally, an error analysis based on the Mean Square Error (MSE) metric was used to provide a quantitative measure of trajectory accuracy. Together, these techniques ensured a robust validation of the manipulator's kinematic and dynamic models.
These techniques are summarized in Table 3.2.

**Table 3.2: Validation Metrics Used in the Study**

| Metric | Description | Purpose |
|---|---|---|
| Trajectory Consistency | Compare kinematic vs. dynamic paths | Verify geometric accuracy |
| Torque Deviation | Compare kinematic vs. dynamic paths | Evaluate actuation consistency |
| Benchmark Comparison | Cross-check with literature | Ensure model reliability |
| Error Analysis (MSE) | Quantitative error measure | Assess numerical accuracy |

### 3.1.6 Performance Evaluation

Finally, the performance of the manipulator was evaluated using two major criteria, namely energy consumption and motion smoothness. Energy consumption was quantified by calculating the total work done by the actuators, which was obtained through the integration of torque over the angular displacement for each joint. Motion smoothness was assessed by analyzing the velocity, acceleration, and jerk profiles of the manipulator along the simulated trajectories, with lower jerk values indicating smoother motion. These evaluation measures provided deeper insight into the quality of motion and the efficiency of the dynamic model in capturing realistic manipulator behavior.

Through this structured methodology, the study not only validates the manipulator's kinematic models but also extends the evaluation to include torque computation accuracy, energy efficiency, and motion quality. This ensures that the validation framework goes beyond geometric accuracy and provides a reliable foundation for future research in adaptive control and learning-based methods for robotic manipulators.

The research employed both symbolic and numerical modeling of a three-link robotic manipulator using the Lagrange formulation. The methodology followed a structured process that began with manipulator modeling, proceeded to dynamic formulation, and was then implemented in computational tools before being validated through comparative simulations.

### 3.1.7 Numerical Simulation

Equation 2.35 and 2.36 were represented using python. The system was numerically simulated using the following parameters:

**Table 3.3: Simulation Parameters**

| Parameter | Value |
|---|---|
| Link lengths | $l_1 = l_2 = l_3 = 1.0m$ |
| Masses | $m_1 = m_2 = m_3 = 1.0kg$ |
| Gravitational acceleration | $g = 9.82m/s^2$ |
| Initial angle $(\theta_1, \theta_2, \theta_3)$ | $30^0, -30^0, 45^0$ |
| Initial angular velocity $(\omega_1, \omega_2, \omega_3)$ | 0,0,0 |

The system of differential equations $\frac{d\theta_1}{dt} = \omega_1, \frac{d\omega_1}{dt} = \tau_1(\theta_1, \theta_2, \theta_3, \omega_1, \omega_2, \omega_3)$ was solved using the $solve\_ivp$ function over a time range of 0 to 10 seconds.

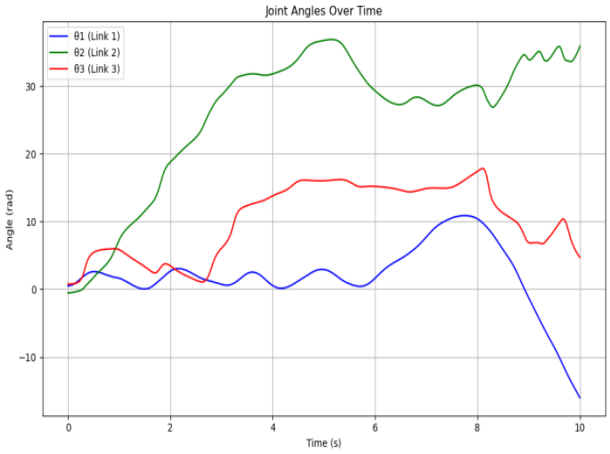


Figure 4.1: Joints Angles over Time

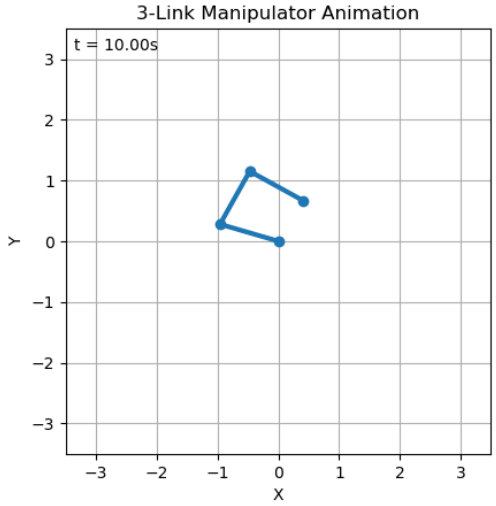


Figure 4.2: Joints Angles over Time

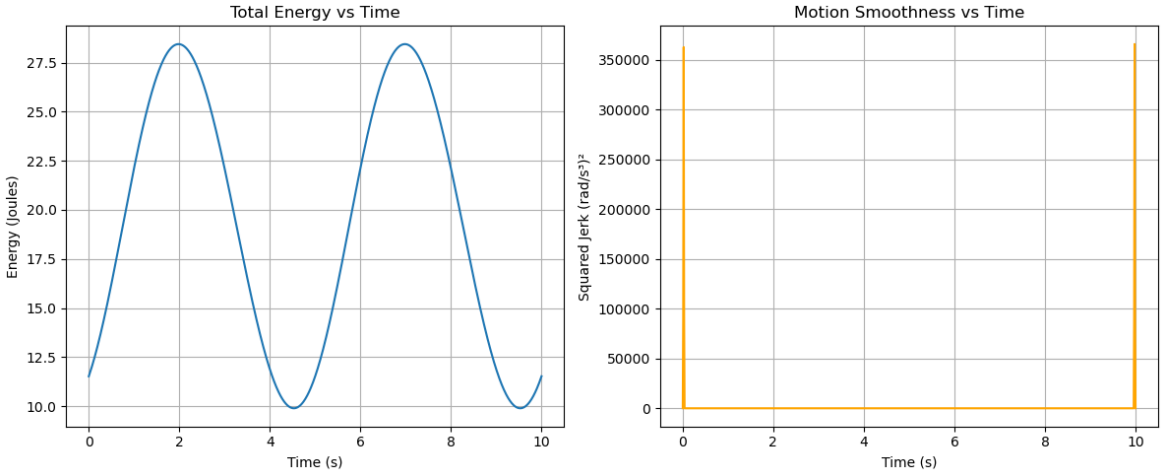


Figure 4.1: Total Energy and Motion Smoothness vs Time

## 3. RESULTS AND DISCUSSION

Joint angle responses for the three links are shown in Figure 4.1. The results indicate that each joint follows a distinct trajectory depending on its role in the manipulator's motion. Joint 2 (green curve) exhibited the most significant angular displacement, rising to approximately 30 radians during the simulation, reflecting its important contribution to overall end-effector movement. Joint 3 (red curve) demonstrated moderate angular excursions around 15 radians, while Joint 1 (blue curve) showed comparatively minor variations but ended with a sharp decline after 8 seconds. This decline suggests that Joint 1 bore a stabilising role early in the trajectory but later experienced increased torque demands. These results confirm that the Lagrange-based dynamic formulation successfully captures the differing contributions of each joint to the manipulator's motion.

A visual representation of the manipulator's movement over time is shown in Figure 4.2. The simulation snapshot illustrates the final configuration of the manipulator at $t$ = 10 seconds. The figure demonstrates smooth and continuous changes in the manipulator's joint positions, confirming that the computed joint torques produced physically consistent trajectories. This visual validation reinforces the correctness of the simulation framework in

replicating realistic manipulator kinematics and dynamics.

Energy consumption and motion smoothness were evaluated as performance metrics, as shown in Figure 4.3. The energy profile (left plot) indicates periodic oscillations between 10 J and 28 J over the 10-second simulation, corresponding to the alternating phases of acceleration and deceleration of the manipulator's joints. The smooth sinusoidal-like variation in energy suggests that the system effectively converts between kinetic and potential energy during motion. The motion smoothness profile (right plot) reveals high jerk values at the start and end of the trajectory, peaking at approximately. $3.5 \times 10^5 (rad/s^3)^2$. This behaviour is attributed to the abrupt initiation and termination of motion, while the intermediate period remained smooth with minimal jerk. These findings highlight the importance of trajectory planning in reducing sudden changes in acceleration, thereby minimising actuator stress and improving overall manipulator performance.

The results demonstrate that the Lagrange formulation accurately models the manipulator's dynamics, producing joint trajectories consistent with theoretical expectations, realistic visual motion, and quantifiable performance metrics. The energy and smoothness evaluations provide further insight into the efficiency and stability of the manipulator, underscoring the robustness of the proposed validation framework.

## 4. CONCLUSIONS

This study presented the validation of kinematic models for a 3-link articulated robotic manipulator using the Lagrange formulation. A computational framework was developed to simulate the dynamic behavior of the manipulator, evaluate its motion characteristics, and assess performance in terms of energy consumption and motion smoothness. The results demonstrated that the Lagrange-based formulation accurately captured the manipulator's dynamics, producing realistic joint trajectories and energy profiles. Visualizations of the manipulator confirmed smooth and continuous motion, while quantitative analysis highlighted the impact of torque variations and jerk on system performance.

The evaluation revealed that energy consumption exhibited periodic oscillations corresponding to the manipulator's alternating acceleration and deceleration phases. Furthermore, jerk analysis showed that the manipulator experienced high transient values at the start and end of the trajectory, indicating that trajectory planning plays a critical role in ensuring smoother motion and reducing mechanical stress on actuators. These findings affirm the robustness of the applied modeling framework in replicating realistic manipulator behavior and provide a baseline for future optimization of robotic systems.

## ACKNOWLEDGEMENTS

I sincerely express my profound gratitude to Almighty God for His grace, strength, and guidance throughout the course of this research and my academic journey. Without His divine help, this work would not have been possible.

I wish to extend special acknowledgement to the Dean of the Faculty of Engineering, Engr. We thank Prof. John Wasiu for his academic leadership and support. My appreciation also goes to the Head of the Department of Computer, Dr Bello O. Lawal, for his dedication and encouragement. I am deeply grateful to my supervisors, Dr Chukwuemeka C. Obasi, for their invaluable guidance, constructive feedback, and continuous support that shaped this work. I also acknowledge the academic and technical staff of the Department of Computer, Edo State University, Uzairue, as well as my colleagues in the department, for their contributions to my learning and research experience.

## REFERENCES

Bac, C. W., et al. (2022). Harvesting Robots for High-value Crops: State-of-the-art Review. *Computers and Electronics in Agriculture*.

Banga, G. S. (Jan. 2022). Robots and its types for industrial applications. *," Mater Today Proc, doi: 10.1016/j.matpr.2021.12.426.*, vol. 60.

Ben-Ari, 2. (n.d.). Elements of Robotics. *Springer Open*, https://doi.org/10.1007/978-3-319-62533-1.

Benitti, F. B. (2021). Exploring the educational potential of robotics in schools: A systematic review. Computers & Education. 58(3), 978-988.

Collins, J., et al. (2020). Sim-to-Real Transfer for Robotic Manipulation. *IEEE Robotics*.

Collins, J., et al. (2021). Sim-to-Real Transfer for Robotic Manipulation. *IEEE Robotics*.

Gong. (2020). A soft manipulator for efficient delicate grasping in shallow water: Modeling, control, and real-world experiments. *International Journal of Robotics Research 10.1177/0278364920917203*.

Kucuk, S., & Bingul, Z. (2016). Robot Kinematics: Forward and Inverse Kinematics. *Industrial Robotics*.

M. Farman, M. A.-S. (2018). Design of a Three Degrees of Freedom Robotic Arm. *Int J Comput Appl*, vol. 179, pp. 12–17, doi: 10.5120/ijca2018916848.

Muelaner, J. E., et al. (2018). Large Volume Metrology for Industrial Applications. *CIRP Annals*.

Obasi et al. (2020).

Santhakumar, M., & Asokan, T. (2019). Nonlinear Modeling of Underwater Manipulators. *Ocean Engineering*.

Shaik, R. (2021). Energy-Based Dynamic Modelling of Multi-Link Manipulators Using Lagrange Methods. *International Journal of Control, Automation and Systems*, 19(3), 320–334.

Taylor, R. H., & Stoianovici, D. (2020). Medical Robotics in Computer-integrated Surgery. *IEEE Trans. Robotics.*

Villani, L., et al. (2019). Force Control in Robotics. *Springer Handbook of Robotics*.

Wu, L., et al. . (2020). Deep Learning for Robot Kinematic Calibration. *IEEE Access*.

Younis, M. A. (2020). Lagrangian-based modeling and motion optimization for medical rehabilitation robots. *Biomedical Engineering Letters,*, 10(4), 475–483.