Konya Mühendislik Bilimleri Dergisi, c. 13, s. 2, 568-579, 2025
Konya Journal of Engineering Sciences, v. 13, n. 2, 568-579, 2025
ISSN: 2667-8055 (Electronic)
DOI: 10.36306/konjes.1595958

REAL-TIME TRAJECTORY TRACKING OF ROBOTIC MANIPULATOR BASED ON COMPUTED TORQUE CONTROL

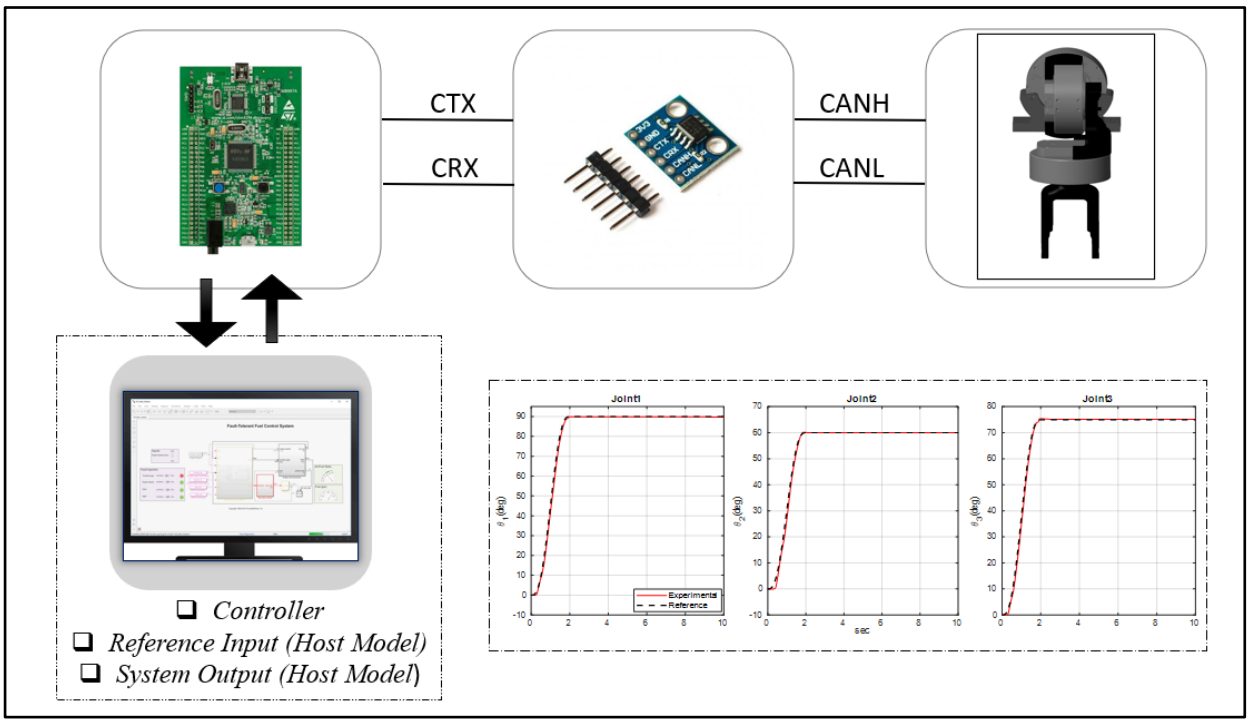
[1,*]Tuğçe YAREN, [2]Selçuk KİZİR

[1] Balikesir University, Electrical and Electronic Engineering Department, Balikesir, TÜRKİYE
[2] Kocaeli University, Mechatronic Engineering Department, Kocaeli, TÜRKİYE
[1] tugce.yaren@balikesir.edu.tr, [2] selcuk.kizir@kocaeli.edu.tr

Highlights

- Developed a 3-DOF manipulator model using the SOA framework, reducing complexity.
- Designed and implemented real-time Computed Torque Control for precise tracking.
- Demonstrated controller's robustness through static and dynamic trajectory tracking

Graphical Abstract



Flowchart of the proposed method

*Corresponding Author: Tuğçe YAREN, tugce.yaren@balikesir.edu.tr

Konya Mühendislik Bilimleri Dergisi, c. 13, s. 2, 568-579, 2025
Konya Journal of Engineering Sciences, v. 13, n. 2, 568-579, 2025
ISSN: 2667-8055 (Electronic)
DOI: 10.36306/konjes.1595958

REAL-TIME TRAJECTORY TRACKING OF ROBOTIC MANIPULATOR BASED ON COMPUTED TORQUE CONTROL

[1,*]Tuğçe YAREN, [2]Selçuk KİZİR

[1] Balikesir University, Electrical and Electronic Engineering Department, Balikesir, TÜRKİYE
[2] Kocaeli University, Mechatronic Engineering Department, Kocaeli, TÜRKİYE
[1] tugce.yaren@balikesir.edu.tr, [2] selcuk.kizir@kocaeli.edu.tr

(Received: 03.12.2024; Accepted in Revised Form: 06.05.2025)

ABSTRACT: This study presents obtaining the mathematical model of a three-degree-of-freedom robotic manipulator using spatial operator algebra (SOA), designing a controller based on the obtained model, and implementing the designed controller in real time. SOA is a technique that provides a mathematical foundation for reducing the complexity of robotic systems, analyzing, and optimizing them. The control of the robotic arm is achieved using the computed torque control method calculated based on mathematical model derived from SOA. The performance of the controller is rigorously evaluated through real-time trajectory tracking experiments, where it consistently achieved high precision in following predefined trajectories, maintaining tracking errors below 2.5 degrees. The effectiveness of the controller is further validated in disturbance rejection tests, where it effectively maintained trajectory accuracy despite manual external perturbations. These tests demonstrate the controller's capability to handle dynamic tasks and disturbances, showcasing the practical applicability and robustness of the SOA-based computed torque control scheme.

Keywords: *Computed Torque Control, Modelling, Robotics, Spatial Operator Algebra, Trajectory Tracking*

1. INTRODUCTION

Since the 1960s, the emergence of robotics in various application areas has led to the need to design nonlinear controllers instead of linear control. Currently, numerous nonlinear controllers are employed in robots. While there are various studies on nonlinear controllers in the literature, computed torque control, adaptive control, and robust control, which are specialized applications of feedback linearization for mechanical systems such as robots, are preferred.

One of the most popular approaches used in the control of robotic manipulators is the computed torque control method. This method provides the necessary torques to successfully track the desired joint-space trajectories. This approach is also referred to in the literature as "inverse dynamics control". This is because this approach relies on the use of the inverse dynamic model to linearize and decompose the nonlinear dynamics of the robot. In cases where all parameters are known with high accuracy, the computed torque control (CTC) method demonstrates high performance. However, the effectiveness of CTC may decrease due to parameter uncertainties such as incorrect inertia matrix and unmodeled dynamics such as joint flexibility [1].

CTC method integrates a PD control term with a feedback dynamic compensation term, which is determined using the system's actual velocity and desired acceleration signals. This structure functions as a PD controller enhanced with a feedback inner loop, allowing the CTC controller to provide improved trajectory tracking and resistance to disturbances. Despite its advantages, the method has two primary limitations [2]. Firstly, the dynamic compensation relies on a model with constant parameters, while the system's parameters typically change during trajectory execution. As a result, the controller's ability to achieve effective dynamic balancing is compromised. To enhance dynamic compensation, adaptive control strategies are often recommended. Such approaches are particularly effective in mitigating external disturbances by leveraging adaptive or robust control frameworks.

Secondly, in the CTC approach, linear PD controllers with fixed proportional and derivative constants

***Corresponding Author:** Tuğçe YAREN, tugce.yaren@balikesir.edu.tr

are utilized to minimize tracking errors. However, when nonlinear factors such as modeling inaccuracies and friction are present in the manipulator's dynamics, the CTC controller may struggle to deliver the desired level of tracking precision. To address this, alternative strategies have been developed to dynamically tune the PD gains of the CTC controller. For instance, Yang et al. [3] proposed the use of intelligent optimization techniques to enhance the PD gain coefficients, thereby improving trajectory tracking performance. Despite these improvements, the practical implementation of such controllers remains challenging due to their intricate design and significant computational demands. A simpler and more effective alternative involves the use of nonlinear PD controllers. These controllers are capable of mitigating uncertainties and demonstrate superior performance compared to traditional PD controllers, all while maintaining a relatively straightforward structure [4].

CTC has been applied to different types of robots, including serial, parallel, humanoid, and collaborative robots. Due to its ability to provide high dynamic performance, this control method has been extensively investigated in the literature, particularly in serial robots. Shang and Cong [2] developed a nonlinear computed torque controller and applied it to a high-speed planar parallel manipulator. The stability of the parallel manipulator system was proven using the Lyapunov theorem, and it was demonstrated that the developed controller guarantees both the asymptotic convergence of tracking error and the error rate to zero. Hayat et al. [5] proposed a model-independent robust-adaptive controller for Euler-Lagrange systems with a quantitative performance analysis in terms of state errors. The proposed controller structure operates as an adaptive computed torque control method. Real-time control of 3-DOF and 7-DOF serial robots was successfully achieved. Lee et al. [6] developed a computed torque control method combined with H∞ control, which demonstrates successful tracking performance even in fast tracking control where full dynamics computation is not easy. Real-time control of the Stewart platform was successfully implemented using the developed method. Due to the high computational load, applying this control method can be challenging even on very high-speed DSPs (Digital Signal Processors). The proposed method addresses this issue. Polydoros et al. [7] proposed a machine learning approach for modeling inverse dynamics and provided information about its application to a physical robotic system. In their study, a collaborative robot was real-time controlled using the computed torque control method. The controller was designed based on the inverse dynamic model obtained using the proposed approach. They obtained an advantage in the computed torque method by obtaining the dynamic model using machine learning due to the problems in obtaining the dynamic model analytically. In general, it is concluded in the literature that the computed torque control method is preferred in robot control applications, but updates are made to its structure to eliminate its disadvantages.

In the CTC, the inverse dynamic model of the system is the most important building block, and the methods commonly used for dynamic modeling of systems are the Lagrange-Euler, Newton-Euler, and Hamilton equations. These methods are fundamental techniques used to model and analyze the motion and behavior of mechanical systems. Each method has its advantages, disadvantages, and application areas, so selecting the right method depends on the system characteristics and analysis requirements. However, obtaining a dynamic model may not be straightforward for every system. As system complexity increases, the difficulty of obtaining the model also increases. The kinematic and dynamic analysis of high-degree-of-freedom hybrid systems, which use serial and parallel robots together on a fixed or mobile platform, is a highly challenging, complex, and time-consuming problem [8]. Considering these structures, classical methods can be both inadequate and costly in terms of computational power.

Obtaining the kinematic and dynamic models of complex and high-degree-of-freedom robots using recursive methods instead of classical methods both simplifies the modeling process and reduces computation time by eliminating unnecessary calculations. Spatial operator algebra (SOA) is one of the prominent structures in this field. Spatial operators can be easily applied to multi-manipulator systems thanks to their recursive structure. It is a more systematic and easily programmable high-performance computing algorithm compared to other kinematic and dynamic analysis methods [9].

SOA is a method based on coordinate-free vector representation, which provides great convenience for the designer as it allows the designer to choose a set of free axes in the configuration. Since this

algorithm does not use gradient-based derivatives, discontinuities and sharp changes in trajectory functions are not a problem. Therefore, systems analyzed using this method are very suitable for real-time programming and control. The Jacobian matrix can be numerically obtained using a systematic and easily programmable method with SOA. It also provides the designer with an analytical expression for the inverse of the mass matrix. The SOA algorithm can provide shorter cycle times. With shorter cycle times, robot arms and robotic systems can be controlled more effectively and powerfully. Thus, the use of the SOA method in robot control enables both robustness and speed to be achieved simultaneously. In summary, SOA is a highly efficient calculation algorithm that is more systematic and easily programmable compared to other kinematic and dynamic analysis methods [10, 11].

Jain [12] described a computational modeling architecture developed to meet a wide range of robot modeling needs, including analysis, simulation, and embedded modeling for robotic systems. The architecture is based on the theoretical framework of UOC for computational dynamics, with calculations performed by fast, structure-based algorithms. The work was conducted at the JPL Laboratory (in collaboration with NASA). Wensing et al. [13] introduced an algorithm based on spatial vector algebra that enables operational space control of sliding-base systems to be performed at higher speeds. The algorithm reduced the computational load $O(n^3 + m^3)$ from to (nd), where n is the system's degrees of freedom, m is the number of constraints, and d is the depth of the kinematic connectivity tree. The accuracy of the algorithm was demonstrated through simulation at a speed of 3.6 m/s, controlling a quadrupedal robot model. They emphasized the ease of creating control structures with SOA modeling. Nakanishi et al. [14] conducted the control of a 16-DOF bipedal robot using inverse dynamics and PD control methods in a simulation environment. Their primary focus in this study was to develop a general and computationally efficient inverse dynamics algorithm for a robot with a free-floating base and constraints. Additionally, effective access to the parameters required for the controller can be achieved through a SOA-based dynamic formulation.

In this paper, the aim was to obtain a dynamic model using the SOA method, design a nonlinear controller based on the obtained model, and implement the designed controller in real-time. To provide a systematic approach in real-time robot control, it was planned to leverage the advantages of the SOA method. A 3-DOF robotic manipulator was utilized to validate the designed controllers in real-time. The real-time application of CTC based on SOA was implemented on a 3-DOF structure, and the applicability of the SOA algorithm in real-time control structures was observed. The equations of motion are derived using SOA in Section 2. The SOA-based CTC are designed in Section 3. In Section 4, the performance of the designed controller is tested.

2. MATERIAL AND METHODS

2.1. Dynamic Modelling of the 3-DOF Manipulator

The right and left views of the manipulator, along with a frame assignment diagram, are presented in Fig. 1. The design of the manipulator emulates the human shoulder, which can be modeled as having three degrees of freedom: flexion/extension, abduction/adduction, and internal/external rotation. Each joint of the manipulator is motor-driven: the first joint facilitates rotation around the z-axis, the second joint around the y-axis, and the third joint around the x-axis. These movements are all coordinated in relation to a fixed frame, mirroring the complex interactions found in human shoulder movements.

The mathematical model of this structure was derived using the SOA algorithm. Consequently, the frame assignment adheres to the conventions used in SOA, facilitating the analysis and simplification of the manipulator's dynamics. The fixed frame serves as a reference point from which all measurements and movements are defined, ensuring consistency in the model's mathematical formulation.

Table 1 details the system parameters, including the mass, link lengths, and inertia properties of the manipulator's links. The inertia matrices are derived from the SolidWorks CAD model.

The link length vectors of the manipulator are given in Eq. (1).

$$\overrightarrow{l_1} = \begin{bmatrix} 0 \\ -l_2 \\ l_1 \end{bmatrix} \quad \overrightarrow{l_2} = \begin{bmatrix} -l_4 \\ -l_3 \\ 0 \end{bmatrix} \quad \overrightarrow{l_3} = \begin{bmatrix} -l_5 \\ 0 \\ 0 \end{bmatrix} \tag{1}$$

The skew-symmetric form of the link length vectors is arranged in Eq. (2).

$$\widehat{l_1} = \begin{bmatrix} 0 & -l_1 & -l_2 \\ l_1 & 0 & 0 \\ l_2 & 0 & 0 \end{bmatrix} \quad \widehat{l_2} = \begin{bmatrix} 0 & 0 & -l_3 \\ 0 & 0 & l_4 \\ l_3 & -l_4 & 0 \end{bmatrix} \quad \widehat{l_3} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & l_5 \\ 0 & -l_5 & 0 \end{bmatrix} \tag{2}$$

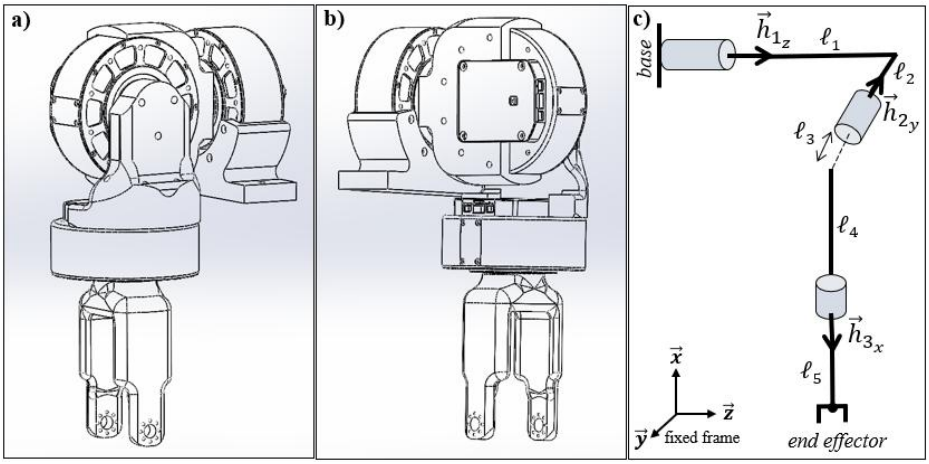


Figure 1. a) Right view of the manipulator, b) left view of the manipulator, c) frame assignment.

Table 1. System parameters

Parameter	Value	Unit
$[m_1\ m_2\ m_3]$	[0.850 0.805 0.250]	kg
$[l_1\ l_2\ l_3\ l_4\ l_5]$	[0.0625 0.0235 0.0195 0.11365 0.1105]	m
$[I_{xx}^1\ I_{yy}^1\ I_{zz}^1]$	[837,720.17 1,252,091.31 809,756.36]	gmm²
$[I_{xx}^2\ I_{yy}^2\ I_{zz}^2]$	[1,087,199.53 1,087,199.53 1,129,158.39]	gmm²
$[I_{xx}^3\ I_{yy}^3\ I_{zz}^3]$	[62,771.32 81,874.85 122,408.01]	gmm²

The link velocity propagation operators can be seen in Eq. (3).

$$\emptyset_{1,0} = \begin{bmatrix} I_3 & 0_3 \\ 0_3 & I_3 \end{bmatrix} \quad \emptyset_{2,1} = \begin{bmatrix} I_3 & 0_3 \\ -\widehat{l_1} & I_3 \end{bmatrix} \quad \emptyset_{3,2} = \begin{bmatrix} I_3 & 0_3 \\ -\widehat{l_2} & I_3 \end{bmatrix} \quad \emptyset_{t,3} = \begin{bmatrix} I_3 & 0_3 \\ -\widehat{l_3} & I_3 \end{bmatrix} \tag{3}$$

The axes of rotation vectors are given in Eq. (4).

$$\overrightarrow{h_1} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad \overrightarrow{h_2} = \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} \quad \overrightarrow{h_3} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} \tag{4}$$

The axes of rotation matrices are given in Eq. (5).

$$\overrightarrow{\overrightarrow{H_1}} = \begin{bmatrix} \overrightarrow{h_1} \\ \vec{0} \end{bmatrix} \quad \overrightarrow{\overrightarrow{H_2}} = \begin{bmatrix} \overrightarrow{h_2} \\ \vec{0} \end{bmatrix} \quad \overrightarrow{\overrightarrow{H_3}} = \begin{bmatrix} \overrightarrow{h_3} \\ \vec{0} \end{bmatrix} \tag{5}$$

Manipulator rotation axis matrix is given in Eq. (6).

$$H = \begin{bmatrix} \overrightarrow{\overrightarrow{H_1}} & \overrightarrow{\overrightarrow{0}} & \overrightarrow{\overrightarrow{0}} \\ \overrightarrow{\overrightarrow{0}} & \overrightarrow{\overrightarrow{H_2}} & \overrightarrow{\overrightarrow{0}} \\ \overrightarrow{\overrightarrow{0}} & \overrightarrow{\overrightarrow{0}} & \overrightarrow{\overrightarrow{H_3}} \end{bmatrix} \qquad H \in R^{18x3} \tag{6}$$

The manipulator propagation matrix can be seen in Eq. (7). I and 0 is the 6x6 identity and zero matrix, respectively.

$$\emptyset = \begin{bmatrix} I & 0 & 0 \\ \emptyset_{2,1} & I & 0 \\ \emptyset_{3,1} & \emptyset_{3,2} & I \end{bmatrix} \qquad \emptyset \in R^{18x18} \tag{7}$$

Tip point propagation matrix is given in Eq. (8).

$$\emptyset_t = [0_{6x6} \quad \emptyset_{t,3}] \tag{8}$$

The Jacobian matrix of the manipulator can be obtained from Eq. (9).

$$J = \emptyset_t \emptyset H \tag{9}$$

The forward kinematic equation of the manipulator can be obtained from Eq. (10).

$$\overrightarrow{\overrightarrow{V_t}} = J\underline{\dot{\theta}} \tag{10}$$

where V_t is the velocity vector of the tip point and $\underline{\dot{\theta}}$ is the stacked link spatial velocities of the manipulator. The inverse kinematic equation of the manipulator can be obtained from Eq. (11).

$$\underline{\dot{\theta}} = J^{-1}\overrightarrow{\overrightarrow{V_t}} \tag{11}$$

The mass matrices of the links are given in Eq. (12).

$$M_1 = \begin{bmatrix} \mathrm{I}_1 & m_1\hat{l}_1 \\ -m_1\hat{l}_1 & I_3 m_1 \end{bmatrix} \quad M_2 = \begin{bmatrix} \mathrm{I}_2 & m_2\hat{l}_2 \\ -m_2\hat{l}_2 & I_3 m_2 \end{bmatrix} \quad M_3 = \begin{bmatrix} \mathrm{I}_3 & m_3\hat{l}_3 \\ -m_3\hat{l}_3 & I_3 m_3 \end{bmatrix} \tag{12}$$

where I_x is the inertia matrix of link x, m_x is the mass vector of link x, and I_3 is the 3x3 identity matrix. The inertia matrix of the joints is assumed to be the identity matrix. The manipulator mass matrix is given in Eq. (13).

$$M = \begin{bmatrix} M_1 & 0 & 0 \\ 0 & M_2 & 0 \\ 0 & 0 & M_3 \end{bmatrix} \tag{13}$$

The generalized mass matrix of the manipulator can be derived from Eq. (14).

$$\boldsymbol{M} = H^T \emptyset^T M \emptyset H \tag{14}$$

Bias terms of the manipulator are given in Eq. (15) (including Coriolis and gravity).

$$\underline{C} = H^T \emptyset^T (M \emptyset \underline{a} + \underline{b}) \tag{15}$$

where $\underline{a}$ is the bias spatial accelerations of the manipulator, $\underline{b}$ is the bias spatial forces of the manipulator. The inverse dynamic equation of the manipulator can be obtained from Eq. (16).

$$\underline{\tau} = \boldsymbol{M}\underline{\ddot{\theta}} + \underline{C} + J^T \vec{F}_t \tag{16}$$

where F_t is spatial forces at the tip point of the manipulator. The inverse dynamics model obtained in Eq. (16) is used in controller design.

2.2. Controller Design

The computed torque control approach is based on the use of inverse dynamics model, as can be seen from Fig. 2. Since it is a model-based approach, obtaining the dynamic model precisely is of critical importance.

The CTC consists of two loops: feedback inner loop and PD control outer loop. CTC control equation obtained according to the block diagram is given in Eq. (17).

$$u = M(\theta)\left[\ddot{\theta}_d + K_v \dot{e} + K_p e\right] + C\left(\theta, \dot{\theta}\right) + G(\theta) \tag{17}$$

where e is joint position error and $\dot{e}$ is joint velocity error, are shown in Eq. (18). CTC has two gain parameters, proportional (K_p) and derivative (K_v).

$$e = \theta_d - \theta \qquad \dot{e} = \dot{\theta}_d - \dot{\theta} \tag{18}$$

where $\theta_d, \dot{\theta}_d, \ddot{\theta}_d$ is the desired position, velocity and acceleration, respectively. Two separate matrix, C and G, appearing in Eq. (17), are included in the $\underline{C}$ matrix in the SOA model. The gains K_p and K_v are obtained through trial-and-error method as 100 and 70, respectively.

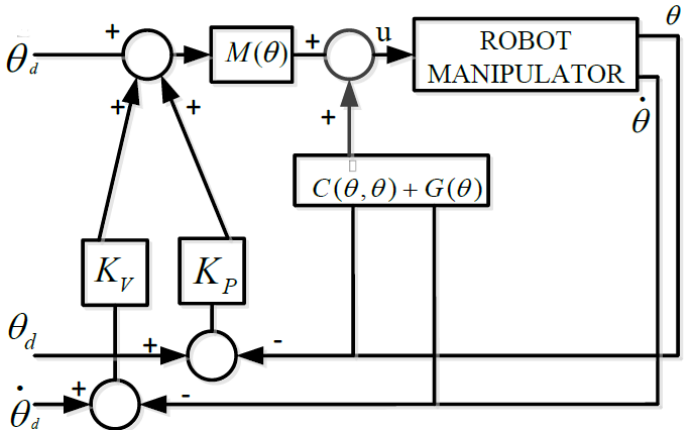


Figure 2. Computed torque control block diagram [15]

3. RESULTS AND DISCUSSION

3.1. Simulation Model

Fig. 3 illustrates the Simulink simulation model constructed based on the CTC strategy as Fig. 2. The dynamic model in the CTC was obtained with the SOA algorithm. A Simmechanics model of a 3-DOF manipulator was used to test the controller's effectiveness on the system.

Inputs to the system include desired joint angles, velocities, and accelerations, derived from predefined reference trajectories. The model integrates feedback of joint angles and velocities, calculating position and velocity errors to adjust the control inputs effectively. These inputs are processed through the SOA algorithm, which computes the necessary torque commands to achieve precise joint positioning and movement, ensuring robust control performance across various operational scenarios.

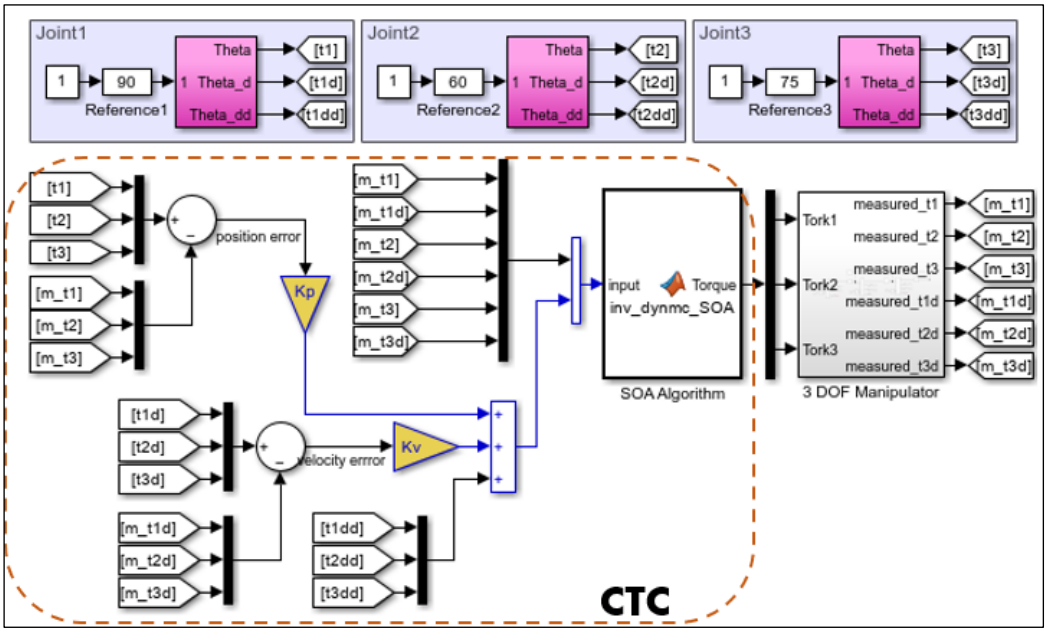


Figure 3. Computed torque control simulation model

The controller performance was tested by applying a trajectory reference signal in simulation. In this test, the robot's trajectory, which meets the initial and final conditions and provides a continuous change over time, is modeled by the 5th-degree polynomial seen in Eq. (19). This polynomial allows the robot to make a smooth and continuous transition from the starting point (t_0) to the endpoint (t_f) within a specific time interval.

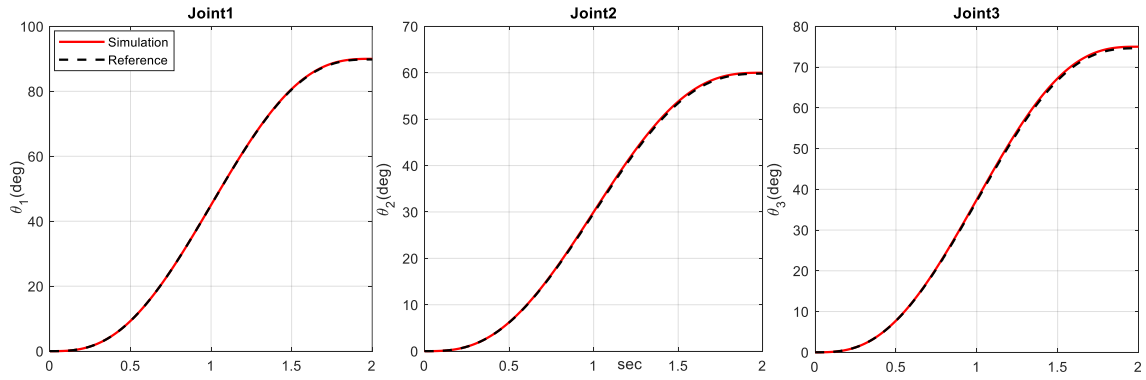


Figure 4. CTC simulation results

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{19}$$

where the coefficients are: $a_0 = \theta_0$, $a_1 = \dot{\theta}_0 = 0, a_2 = \frac{1}{2}\ddot{\theta}_0 = 0, a_3 = \frac{1}{2t_f^3}(20)(\theta_f - \theta_0), a_4 = \frac{1}{2t_f^4}(30)(\theta_0 - \theta_f), a_5 = \frac{1}{2t_f^5}(12)(\theta_f - \theta_0)$. θ_0 represents the robot's starting position, while θ_f represents the final position.

Initially, the robot's velocity and acceleration were considered to be zero (a_1, a_2). Based on this polynomial, the desired trajectories are defined as follows: $\theta_{1d}(t_0) = 0°, \theta_{1d}(t_f) = 90°, \theta_{2d}(t_0) = 0°, \theta_{2d}(t_f) = 60°, \theta_{3d}(t_0) = 0°, \theta_{3d}(t_f) = 75°$ with $t_0 = 0\ s, t_f = 2\ s$. The trajectory tracking results of the simulation are shown in Fig. 4. The simulation results indicate that all joints successfully achieved their desired positions with minimal tracking errors.

3.2. Experimental Results

The STM32F4 discovery board was used as the microcontroller. The code was generated using the Waijung Blockset by Aimagin [16] in MATLAB Simulink.

The motor used in the system is the MyActuator RMD-X8 Pro intelligent actuator. This actuator integrates a high-torque BLDC motor, an internal motor controller, an absolute encoder, and a 6:1 planetary reduction gear into a single compact unit. It supports control via the CAN bus interface, enabling users to specify target position, velocity, or torque for seamless operation [17]. The key specifications of the RMD-X8 Pro are provided in Table 2.

RMD X8-PRO communication is carried out using the CAN bus protocol. To enable communication with any processor, a CAN module is required. In this paper, an STM32F4 development kit and an SN65HVD230 CAN module were used for communication with the motors. The communication structure established between the computer and the motor is depicted in Fig. 5. With this structure, commands can be sent from the computer to the motor, and data can be collected from the motor. The motor has three operating modes: position, velocity, and torque. In this study, the control applications were implemented in the torque mode.

Table 2. RMD-X8 Pro parameters

Parameter	Value
Weight (g)	750
Gear ratio	6:1
Nominal Torque (Nm)	12
Peak Torque (Nm)	35
Torque density (Nm/kg)	46.7
Joint Velocity (rad/s)	23.6

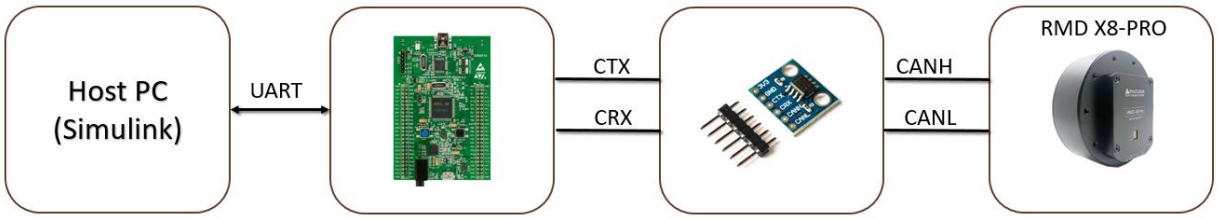


Figure 5. Communication structure

Controller performance was tested by applying different reference signals. In the first real-time test, the reference signal applied in the simulation study was applied to the manipulator ($\theta_{1d}(t_0) = 0°, \theta_{1d}(t_f) = 90°, \theta_{2d}(t_0) = 0°, \theta_{2d}(t_f) = 60°, \theta_{3d}(t_0) = 0°, \theta_{3d}(t_f) = 75°$ with $t_0 = 0\ s, t_f = 2\ s$).

The trajectory tracking experimental results of the first test are shown in Fig. 6. The experimental results indicate that all joints successfully achieved their desired positions with minimal tracking errors. This demonstrates the high accuracy and stability of the controller under steady-state conditions.

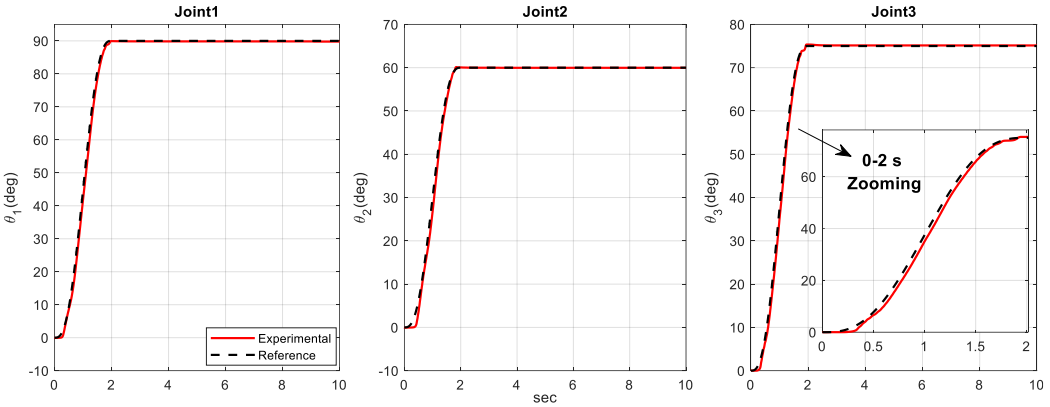


Figure 6. First test results - the joint positions.

The corresponding torque responses in Fig. 7 show smooth and efficient control efforts, with no significant oscillations, indicating energy-efficient actuation. These results highlight the controller's capability for stable and precise joint motion under static reference conditions.

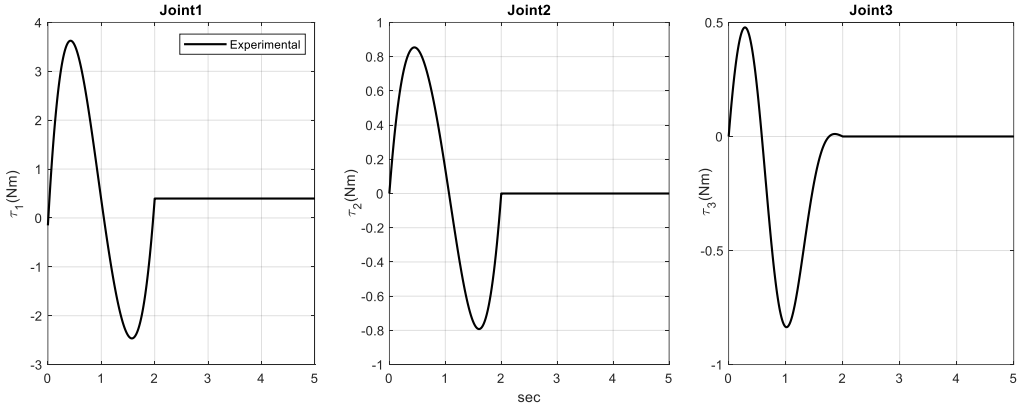


Figure 7. First test results - the joint torques.

In the second test, a sinusoidal reference signal with varying amplitudes for each joint. The desired signal amplitudes are 90° at the first joint, 30° at the second, and 60° at the third. The results confirm that the controller can accurately follow dynamic trajectories, even with continuous changes in the reference (see Fig. 8). The smooth sinusoidal tracking demonstrates the system's adaptability and robustness in handling varying control demands. This highlights the controller's potential for applications requiring periodic or repetitive motions, such as industrial pick-and-place tasks or dynamic path following.

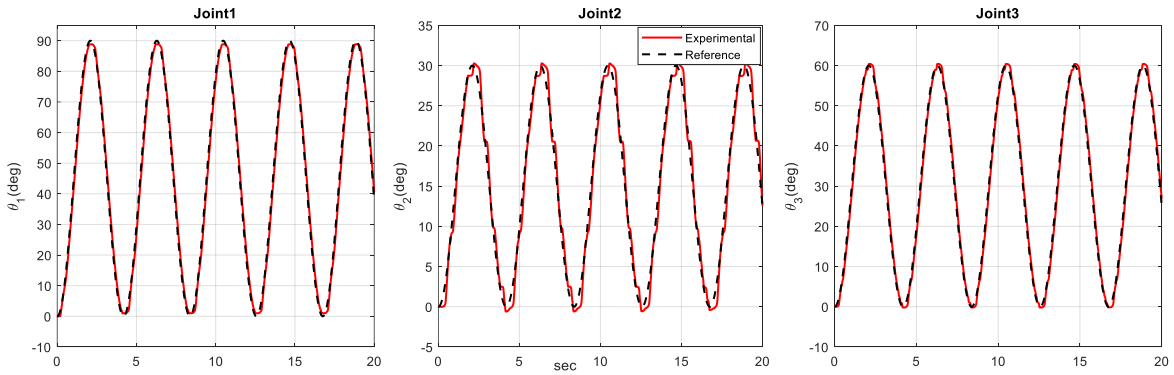


Figure 8. Second test results - the joint positions.

Fig. 9 presents a detailed analysis of the tracking errors for each joint during the second test. The maximum tracking error observed was approximately 2.5° for the second joint while the tracking errors mostly within ±2.5 degrees for all joints. The tracking errors averaged within ±1.25°, indicating the system's ability to track alignment with the desired trajectories.

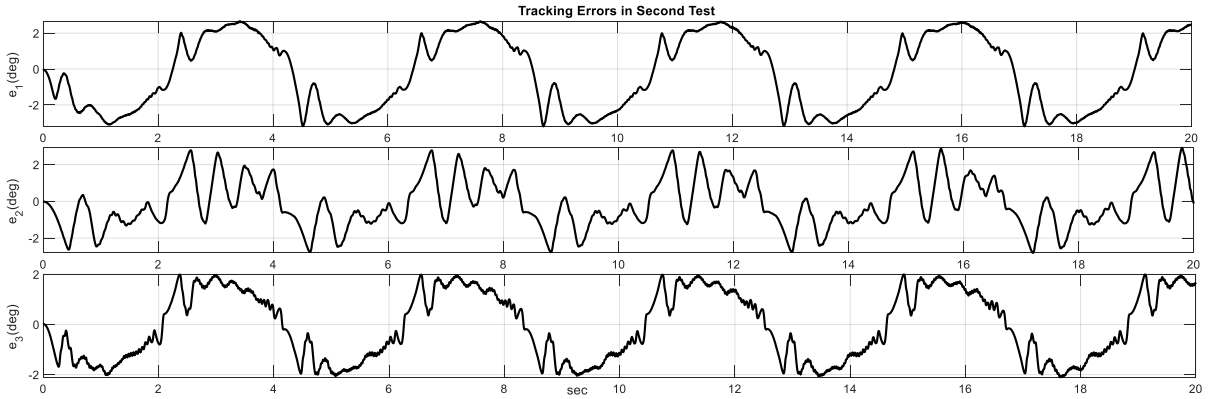


Figure 9. Second test results – tracking errors.

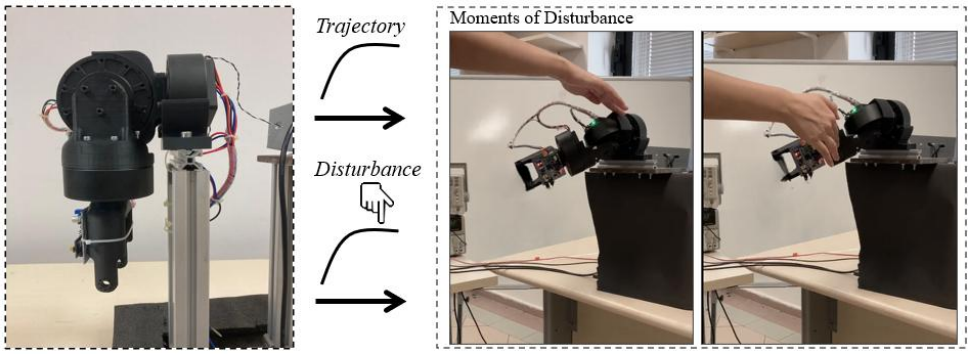


Figure 10. Demonstration of disturbance application in third test

The third test was designed to evaluate the controller's ability to reject disturbances by applying physical obstructions to the joints during trajectory tracking, as illustrated in Fig. 10. During this test, manual forces were intermittently applied to each joint to simulate unexpected physical interferences, challenging the controller's ability to maintain the predefined trajectory.

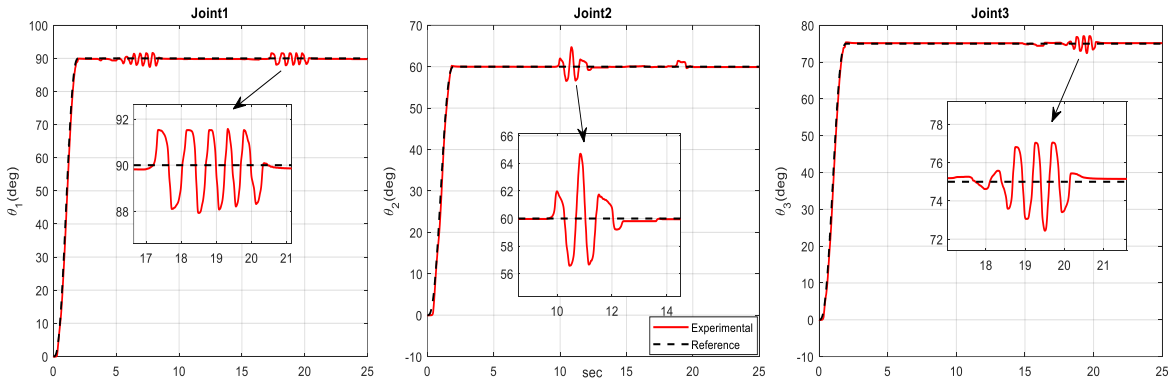


Figure 11. Third test results - the joint positions.

The system exhibited maximum overshoots of 1.78% for joint-1, 7.33% for joint-2, and 2.67% for joint-3 in response to disturbances, as shown in Fig. 11. The settling times to return to the reference band were

3.5 seconds for joint-1, 2.6 seconds for joint-2, and 2.9 seconds for joint-3, respectively. Although the disturbance caused temporary deviations of approximately 2-3 degrees, the controller successfully restored the joints to their reference positions. This response demonstrates the resilience of the controller under external perturbations and its ability to maintain stability and accuracy, even in non-ideal conditions. Such robustness is crucial for real-world applications where external disturbances are inevitable.

4. CONCLUSIONS

In this study, the 3-DOF robotic manipulator was mathematically modeled, and the system's equations of motion were derived using the Spatial Operator Algebra (SOA) framework. Leveraging the complete and verified SOA dynamic model, a Computed Torque Control (CTC) strategy was designed and successfully implemented on a microcontroller. To evaluate the controller's performance, three distinct tests were conducted: parabolic and sinusoidal trajectory tracking, as well as disturbance rejection.

The experimental results demonstrated the controller's high precision in tracking both static and dynamic reference signals. Furthermore, its robust performance under external disturbances validated its reliability for real-world applications. These findings underscore the CTC controller's effectiveness as a practical and robust solution for advanced robotic systems.

An important conclusion from this study is the successful real-time integration of the SOA algorithm into control applications. The feasibility of combining SOA with CTC was verified experimentally, opening up new possibilities for real-time applications in robotics. Future research could explore enhancing the control strategy to improve energy efficiency or extending the methodology to manipulators with higher degrees of freedom. Additionally, integrating advanced control techniques, such as Nonlinear Model Predictive Control (NMPC), with the SOA framework could further enhance the system's performance and adaptability.

Declaration of Ethical Standards

The authors declare that all ethical guidelines including authorship, citation, data reporting, and publishing original research.

Credit Authorship Contribution Statement

Autor 1: Design, experiments, result analysis, writing, editing; Autor 2: Design, editing, supervision.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Funding / Acknowledgements

The study was supported by the Scientific Research Projects Coordination Unit of Kocaeli University through a research project (BAP project #2021-2483).

Data Availability

Research data has not been made available in a repository.

REFERENCES

[1] T. Yaren and S. Kizir, "Efficiency assessment of SOA-based computed torque control: A

comparative analysis with NE-based approach", *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 238, no. 8, pp. 1410-1424, 2024.

[2] W. Shang and S. Cong, "Nonlinear computed torque control for a high-speed planar parallel manipulator", *Mechatronics*, vol. 19, no. 6, pp. 987–992, 2009.

[3] Z. Yang, J. Wu and J. Mei, "Motor-mechanism dynamic model based neural network optimized computed torque control of a high speed parallel manipulator", *Mechatronics*, vol. 17, no. 7, pp. 381–390, 2007.

[4] Y. Xu, J. M. Hollerbach and D. Ma, "A nonlinear PD controller for force and contact transient control", *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 15–21, 1995.

[5] R. Hayat, M. Leibold and M. Buss, "Robust-Adaptive Controller Design for Robot Manipulators Using the H∞ Approach", *IEEE Access*, vol. 6, pp. 51626–51639, 2018.

[6] S. H. Lee, J. B. Song, W. C. Choi and D. Hong, "Position control of a Stewart platform using inverse dynamics control with approximate dynamics", *Mechatronics*, vol. 13, no. 6, pp. 605–619, 2003.

[7] A. S. Polydoros, E. Boukas and L. Nalpantidis, "Online multi-target learning of inverse dynamics models for computed-torque control of compliant manipulators", In IEEE Int Conf Intell Robot Syst, 2017, pp. 4716-4722.

[8] H. Ozakyol, C. Karaman and Z. Bingul, "Kinematic and Dynamic Analysis and Design Toolbox of High-DOF Hybrid Multibody Systems", In IECON 44th Annu. Conf. IEEE Ind. Electron. Soc., IEEE, 2018, pp. 2558–63.

[9] G. Yıldız, "Nonlinear Control Methods Of Industrial Serial Robots", Master Thesis, Istanbul Technical University, Istanbul, Türkiye, 2014.

[10] A. Jain, "Structure-Based Computational Modeling Architecture for Robotics", In IEEE International Conference on Robotics and Automation, 2013.

[11] T. Yaren and S. Kizir, "Real-Time Nonlinear Model Predictive Control of a Robotic Arm Using Spatial Operator Algebra Theory", *Journal of Field Robotics*, 2025. DOI: 10.1002/rob.22514

[12] T. Yaren and S. Kizir, "Advanced SOA-NMPC Controller Design Minimising Real-Time Computational Burden for Dynamic Obstacle Avoidance in Robotic Manipulators", *International Journal of Systems Science*, 2025. DOI: 10.1080/00207721.2025.2479769

[13] P. M. Wensing, L. R. Palmer and D. E. Orin, "Efficient recursive dynamics algorithms for operational-space control with application to legged locomotion", *Auton Robots*, vol. 38, pp. 363–81, 2015.

[14] J. Nakanishi, M. Mistry and S. Schaal, "Inverse Dynamics Control with Floating Base and Constraints", In Proc. 2007 IEEE Int. Conf. Robot. Autom., 2007, pp. 1942–7.

[15] A. Elşavi, "Fractional-order impedance control of 2-DOF seri robot", Master Thesis, Kocaeli University, Kocaeli, Türkiye, 2019.

[16] S. Kizir, T. Yaren and E. Kelekçi, "Matlab Simulink Destekli Gerçek Zamanlı Kontrol: Teori ve Mühendislik Uygulamaları", Ankara, Türkiye: Seçkin Yayıncılık; 2019.

[17] T. Kang, J. Kim, D. Song, T. Kim and S. J. Yi, "Design and Control of a Service Robot with Modular Cargo Spaces", In 18th Int. Conf. Ubiquitous Robot, 2021, pp. 595–600.