




Article

Hybrid System for Fault Tolerance in Selective Compliance Assembly Robot Arm: Integration of Differential Gears and Coordination Algorithms

Claudio Urrea , Pablo Sari  and John Kern 

Electrical Engineering Department, Faculty of Engineering, University of Santiago of Chile, Las Sophoras 165, Estación Central, Santiago 9170020, Chile; pablo.sari@usach.cl (P.S.); john.kern@usach.cl (J.K.)

* Correspondence: claudio.urrea@usach.cl; Tel.: +56-2-27183350

Abstract: This study presents a fault-tolerant control system for Selective Compliance Assembly Robot Arm (SCARA) robots, ensuring operational continuity in cooperative tasks. It is evaluated in five scenarios: normal operation, failures without reconfiguration, and with active reconfiguration. The system employs redundant actuators, differential gears, torque limiters, and rapid detection and reconfiguration algorithms. Simulations in MATLAB R2024a demonstrated reconfiguration times of 0.5 s and reduced trajectory errors (0.0042 m on the X-axis for Robot 1), achieving efficiency above 99%. Nonlinear Model Predictive Controllers (NLMPCs) and Adaptive Sliding Mode Control (ASMC) were compared, with NLMPC excelling in stability and ASMC in precision. The system showcased high productivity in pick-and-place tasks, even under critical failures, establishing itself as a robust solution for industrial environments requiring high reliability and advanced automation.

Keywords: fault-tolerant control; SCARA robotics; cooperative coordination; active redundancy; MATLAB simulations; differential gears



Academic Editor: Tamás Haidegger

Received: 1 December 2024

Revised: 27 December 2024

Accepted: 22 January 2025

Published: 24 January 2025

Citation: Urrea, C.; Sari, P.; Kern, J. Hybrid System for Fault Tolerance in Selective Compliance Assembly Robot Arm: Integration of Differential Gears and Coordination Algorithms.

Technologies **2025**, *13*, 47.

<https://doi.org/10.3390/technologies13020047>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, cooperative robotics has significantly impacted various industrial sectors, enhancing efficiency and precision in complex tasks [1,2]. These systems coordinate multiple robots to achieve common goals, increasing flexibility and productivity in environments like automated factories, precision assembly, and advanced manufacturing [3,4].

Selective Compliance Assembly Robot Arm (SCARA) robots are valued for their precision and speed in repetitive movements, especially in assembly and object-handling tasks [5,6]. Designed for high-efficiency horizontal movements, they are ideal for cooperative environments requiring speed and accuracy [7]. However, these systems face challenges in robustness and reliability, particularly when failures occur during cooperative operations [8].

A robotic system's ability to remain operational despite failures is crucial in collaborative industrial applications, where disruptions can significantly impact productivity and quality [9]. Collaborative tasks demand high precision and synchronization to prevent interruptions [10]. Critical failures, such as actuator or sensor issues, can desynchronize robots, causing errors that compromise system efficiency and accuracy [11,12].

Fault tolerance is another significant challenge in cooperative robotics. Many systems stop affected robots during failures to avoid damage, ensuring safety [13]. However, this

can trigger chain reactions, halting the entire system and severely impacting industrial operations that rely on continuous workflows [14,15].

Designing systems capable of detecting, diagnosing, and reconfiguring robots during failures is essential. These strategies minimize downtime and enable remaining robots to adapt, maintaining reliability and efficiency [16]. Robust systems must ensure individual and collective operation during unexpected events [17,18].

Despite advances in fault-tolerant control, current methods face limitations in response time and complexity. Many rely on complex algorithms requiring significant processing, delaying reconfiguration during critical situations like actuator burnout. Passive methods, though robust, lack the flexibility for dynamic environments [19,20]. This study proposes a hybrid solution combining mechanical and electronic components to minimize interruptions.

Failures in actuators operating under continuous pressure in cooperative setups compromise both individual and collective robot performance. These failures often stem from mechanical issues, like gear wear, or electrical faults, like short circuits [21,22].

Fault-Tolerant Control (FTC) has become a critical research topic in modern robotics, particularly in applications demanding operational continuity and precision [23]. Innovative solutions are categorized as Passive Fault Tolerant Control (PFTC) or Active Fault Tolerant Control (AFTC). PFTC systems are robust against known failures without active detection, while AFTC systems integrate detection, diagnosis, and reconfiguration mechanisms, adapting dynamically in real time [8,24].

A study in [25] introduces an adaptive visual control scheme for manipulators in uncalibrated environments using Lyapunov analysis to ensure stability and image error convergence. This adaptive approach compensates actuator failures, enhancing industrial tracking performance. Another scheme combines Multiple Adaptive Neuro-Fuzzy Inference System (M-ANFIS) with a Proportional Integral Derivative (PID) controller, detecting faults with neural networks and adapting in real time to ensure robustness [26].

A Neural Dynamic Fault-Tolerant (NDFT) scheme for redundant dual-arm robots employs Variable Parameter Recurrent Neural Networks (VP-RNN) and Quadratic Programming (QP) to compensate for failures while maintaining high-precision trajectory and orientation [27]. Other methods include fixed-time nonsingular sliding mode control with disturbance observers, effectively managing actuator failures [28,29], and Interval Type-2 Takagi-Sugeno Fuzzy Models (IT2 T-S) for handling uncertainties and external disturbances [30].

For space manipulators, integrated fault diagnosis and fault-tolerant control using adaptive extended Kalman filters and sliding mode control enhance state estimation accuracy under noisy conditions [31]. Distributed neuroadaptive schemes employing neural networks ensure stability and accuracy for non-holonomic multi-robot systems [22].

For rigid robotic systems, methods like Funnels with Radial Basis Function Neural Networks (RBFNN) maintain tracking error limits despite dead zones and failures, ensuring system stability [32]. Disturbance observer-based methods support continuum robots in high-precision applications like surgery [33]. Image-Based Visual Servoing (IBVS) with iterative learning fault observers and sliding mode controllers manages uncertainties effectively, reducing chattering phenomena [34].

For nonlinear systems, ref. [35] introduces an Interval Type-2 Takagi-Sugeno Fuzzy Model (IT2 T-S Fuzzy Model), which has proven flexible in handling uncertainties and actuator failures. Numerical simulations validate its ability to manage external disturbances and highlight its applicability to different robotic manipulator configurations. For the aerospace industry, ref. [36] presents a distributed neuroadaptive scheme using back-

stepping techniques and neural networks to ensure the stability of robot formations, even in the presence of actuator failures.

A review in [37] discusses advanced planning and control strategies for cooperative manipulator systems, emphasizing their adaptability in dynamic environments. Specific approaches include the use of deep learning for trajectory optimization [38] and reinforcement learning to improve fault detection and adaptation [39]. These methods have demonstrated robustness in experimental validations, particularly in multi-robot systems requiring high degrees of precision and resilience [40].

Recent research has also focused on the integration of digital twin technologies for SCARA robots, enabling real-time simulation and validation of fault-tolerant systems [41]. Advanced domain adaptation techniques for fault diagnosis have been developed to enhance system reliability under varying operational conditions [42,43].

Further studies explore innovative control strategies, including hybrid neural-fuzzy systems [44], predictive control models for multi-robot systems [45], and adaptive reinforcement learning approaches [46]. These methods aim to optimize fault recovery times and ensure sustained performance in dynamic and uncertain environments.

For industrial applications, ref. [47] highlights the significance of coordinated motion planning to mitigate collision risks, while [48] investigates real-time collaborative localization techniques for multi-robot setups. These approaches leverage advanced sensing and data fusion technologies to enhance accuracy and robustness in cooperative scenarios.

The development of fault-resilient hardware, such as dual-actuator configurations and torque-limiting mechanisms, further supports operational continuity in high-stakes environments [49,50]. Recent innovations also include bio-inspired control methods and cerebellum-based predictive models for dynamic task handling [51,52].

Finally, refs. [53–59] present a comprehensive analysis of energy-efficient fault-tolerant systems, emphasizing the integration of sustainable practices in modern robotics. These studies collectively highlight the growing need for hybrid solutions that combine mechanical, electronic, and computational advancements to address the challenges of fault tolerance in cooperative robotics.

Table 1 compares characteristics of these methods with the proposed hybrid system, which integrates immediate detection and reconfiguration capabilities, ensuring scalability for multiple robots without compromising coordination or performance.

Table 1. Comparison of previous studies and the proposed approach in terms of response time, computational complexity, and scalability.

Previous Studies	Fault Response Time	Computational Complexity	Scalability for Multiple Robots
PFTC Systems	Fast; but limited to known faults	Low (no active detection required)	Moderate; adapts to small and medium systems
Adaptive Visual Scheme	Slow in complex scenarios	High (Lyapunov analysis and compensation)	Limited; performance issues with many robots
Neural Dynamic Scheme	Medium; effective for complex faults	High (recurrent neural networks and QP)	High; well-suited for dual robot configurations
Takagi-Sugeno Fuzzy Model	Medium; effective against disturbances	Medium (Type-2 fuzzy model)	Moderate; better performance in well-defined systems
Visual Servo Control	Medium to slow; affected by chattering	Medium (sliding modes)	Limited; challenging to coordinate multiple robots
Proposed Study	Fast; immediate detection and reconfiguration	Low to Medium (simple mechanical and electronic components)	High; optimized design for multiple robots without compromising coordination and performance

This study addresses the gaps in current solutions by proposing a hybrid system that combines mechanical innovations and computational efficiency to achieve faster, simpler

fault-tolerance solutions for SCARA robots. The detailed methodology and experimental results validate the effectiveness of the approach in enhancing robustness and reliability in industrial applications.

In the specific case of a cooperative system composed of two SCARA robots, the stoppage of one due to an actuator failure can significantly affect synchronization, increasing the risk of accidents within its environment. This can lead to collisions between the robots, potentially causing severe damage to both the equipment and the process. To address this issue, this study proposes a hybrid system of redundant actuators that can not only detect and respond rapidly to a failure but can also enable the robots to continue operating without interruptions. Although current systems based on diagnostics and automatic reconfiguration are effective in certain contexts, they have limitations in response speed and implementation complexity. Therefore, this study proposes an approach that does not rely solely on complex diagnostics or advanced algorithms but incorporates mechanical and control solutions to provide greater safety and robustness.

An active redundancy system for the actuators of SCARA robots is proposed, utilizing a combination of primary and secondary (redundant) actuators in each joint. By employing a differential gear system along with torque limiters, the system ensures that when an actuator fails, the secondary actuator is immediately activated to continue operations without interruptions, preventing any impact on the synchronization and coordination between the robots.

This approach offers a simpler solution compared to purely algorithmic methods, as it integrates a mechanical component, a coordination algorithm for cooperative tasks, and a control system. This combination provides a system capable of responding immediately to failures without relying exclusively on complex diagnostics.

To implement active redundancy in the joints, a differential gear system is used, allowing the two actuators to operate jointly without interfering with each other. This ensures that the torques generated by both actuators are distributed in a controlled manner, preventing a failure in one actuator from transferring undesired loads to the other. Additionally, torque limiters are included between each actuator's shaft, ensuring that when one actuator fails, the other can take over control without the system experiencing overloads or additional damage.

This combination not only enables smooth control transfer between actuators but also reduces the need for complex diagnostic systems and advanced reconfiguration algorithms. As a solution based on simple mechanical and electronic components, the system can respond quickly to a failure, minimizing latency in fault detection and robot reconfiguration. This simplicity also lowers maintenance and repair costs while extending the robot's lifespan by efficiently redistributing loads.

The proposed system stands out for its hybrid approach, which does not rely exclusively on artificial intelligence techniques or complex algorithms. It ensures an immediate response to failures without requiring the system to stop. This feature is particularly relevant in cooperative environments where the interruption of one robot can cause a domino effect, impacting other robots that depend on it to complete interdependent tasks. By ensuring that each SCARA robot can continue operating without interruption, the overall system's robustness and reliability are significantly enhanced.

For this purpose, the main objective of this study is to implement a fault-tolerant system for cooperative robots based on active actuator redundancy. To achieve this, the robots are designed in SolidWorks, and a virtual scenario is developed in MATLAB R2024a, where various tests are conducted by simulating actuator failures. Additionally, the system's performance is analyzed under different scenarios to ensure operational continuity, precision, and robustness in industrial environments.

To achieve this, this document is structured into six sections:

Section 1 presents the introduction and state of the art, highlighting the relevance of cooperative robotics systems and the challenges they face when performing tasks, especially in the event of a failure.

Section 2 describes the materials and methods used in the study.

Section 3 details the kinematic and dynamic modeling of the SCARA robots, focusing on the design of prototypes in SolidWorks 2022 and the mechanical components: servomotors, differential gears, and torque limiters.

Section 4 implements the fault-tolerant control system and describes the simulations carried out in a virtual environment using MATLAB R2024a.

Section 5 analyzes the results obtained in Section 3.

Section 6 presents the conclusions and future projections for this study.

2. Materials and Methods

This section details the materials and methods used in this study. An ASUS TUF GAMING A15 laptop (Taipei, Taiwan) was employed with the following specifications: AMD Ryzen 7 6800H processor with Radeon Graphics at 3.2 GHz, 8 GB of RAM, and an NVIDIA GeForce RTX 3050 Ti Laptop GPU.

Additionally, the following software was used:

- SolidWorks 2022: Design of the various robotic prototypes and their components, such as motors, torque limiters, and the differential gear system.
- MATLAB R2024a: Validation of the SCARA cooperative system through various virtual tests involving actuator failures.

The experimental design focuses on developing a hybrid fault-tolerant SCARA robotic system that combines mechanical components, such as differential gears and torque limiters, with an algorithm to maintain constant coordination between robots while performing a task. The system's objective is to sustain operation in the presence of various failures in the robot's primary actuators by utilizing a redundant actuator. To validate the design, simulations are conducted in a virtual environment, evaluating the effectiveness of the fault detection, diagnosis, and reconfiguration algorithms.

For the experimental development, the following variables must be considered:

- Independent variables: Cooperative task, primary actuators with and without faults, numerical sensitivity threshold configuration, fault detection parameters, and the coordination algorithm.
- Dependent variables: Efficiency of the cooperative system in the presence of faults (response time, trajectory accuracy, coordination between robots).
- Controlled variables: Physical parameters of the robots, mechanical and electrical characteristics of the actuators, and the logic of the algorithms.

The simulations are carried out in a virtual environment developed in the MATLAB R2024a software, where the kinematic and dynamic models of the robots are first defined. The prototypes designed in SolidWorks are exported, and the cooperative task is programmed in a complex scenario. To validate each of the variables mentioned above, total failures in the primary actuators of the robots are simulated, and the system's behavior is analyzed using the following metrics: system response time, measured from fault detection to system reconfiguration; precision, which is the deviation in the trajectory of the end effector; and efficiency, analyzed through operational continuity and synchronization between the robots during the task. To analyze the system's ability to detect, diagnose, and reconfigure without interruptions, performance indices are used.

The system incorporates three algorithms to ensure the proper functioning of the robots in the event of actuator failures. The first algorithm detects failures in the robots by comparing the deviation between the desired and measured Cartesian positions of the end effector, while the diagnostic determines the faulty actuator through joint error analysis. The second algorithm handles the reconfiguration of the robots by deactivating the faulty actuator and activating the redundant actuator in real time. The third algorithm manages coordination between robots to avoid collisions and ensure the correct execution of the cooperative task.

3. Fault-Tolerant System

This section addresses the methodology used for the implementation of the fault-tolerant system, whose main objective is to monitor and ensure that the robots continue to operate correctly, even when some of their components fail. This capability is crucial in SCARA robots, used for executing cooperative tasks, as a failure in one of the robots could compromise the entire system, affecting the precision, synchronization, and quality of the task performed.

This study proposes a novel approach with redundant actuators in each joint of the robot. In the event of a failure in the primary actuator, the secondary actuator replaces it, allowing the process to continue correctly. For this purpose, the system is structured in several stages: fault detection, diagnosis, reconfiguration, and control, as shown in Figure 1.

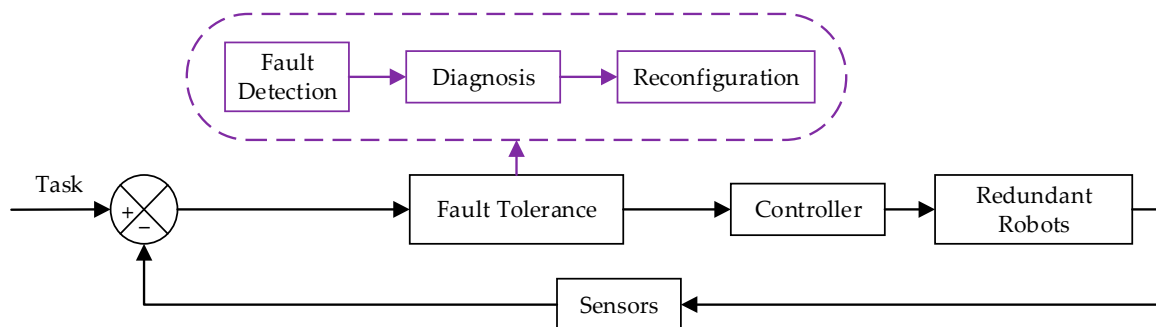


Figure 1. Fault-tolerant system diagram for the cooperative SCARA system.

3.1. Selective Compliance Assembly Robot Arm with Actuator Redundancy

The robotic system consists of two SCARA robots configured to perform cooperative tasks. Each is structured with four joints that enable movements in the three-dimensional plane. To ensure operational continuity in case of failures, each joint is equipped with a primary actuator responsible for movement during normal operation and a redundant actuator that remains inactive until the primary actuator fails (see Figure 2).

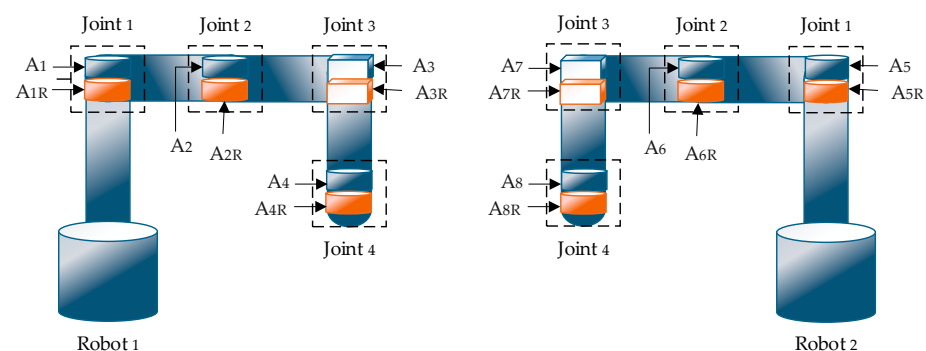


Figure 2. Arrangement of primary and redundant actuators in SCARA robots.

where:

Joint 1 and Joint 2: Rotational joints controlling the horizontal movement of the arm.

Joint 3: Regulates the vertical displacement of the end-effector.

Joint 4: Regulates the orientation of the end-effector.

A_1, A_2, A_3, A_4 : Primary actuators corresponding to the first, second, third, and fourth joints of Robot 1, respectively.

$A_{1R}, A_{2R}, A_{3R}, A_{4R}$: Redundant actuators corresponding to the first, second, third, and fourth joints of Robot 1, respectively.

A_5, A_6, A_7, A_8 : Primary actuators corresponding to the first, second, third, and fourth joints of Robot 2, respectively.

$A_{5R}, A_{6R}, A_{7R}, A_{8R}$: Redundant actuators corresponding to the first, second, third, and fourth joints of Robot 2, respectively.

3.1.1. 3D Modeling and Design of SCARA Robots in SolidWorks 2022 Software

The design of the SCARA robots is carried out in SolidWorks 2022 software (SolidWorks: <https://www.solidworks.com/> (accessed on 29 November 2024)), where the main components were modeled to replicate the characteristic mobility of this type of robot (see Figure 3). The design ensures that the robots can perform precise movements in the three-dimensional plane, essential for cooperative tasks.



Figure 3. SCARA robot components: (a) base; (b) first arm; (c) second arm; (d) shaft.

Once all the individual components are modeled in 3D, the robot assembly is performed using SolidWorks assembly tools:

Position relations: Constraints such as aligning axes, inserting pivots, and limiting degrees of freedom are applied to replicate the robot's real movement.

Motion simulation: To validate the design, the motion simulation tool in SolidWorks 2022 is used, allowing verification of the arm's range of motion and the accuracy of the end-effector positioning.

Figure 4 shows the SCARA system prototype.

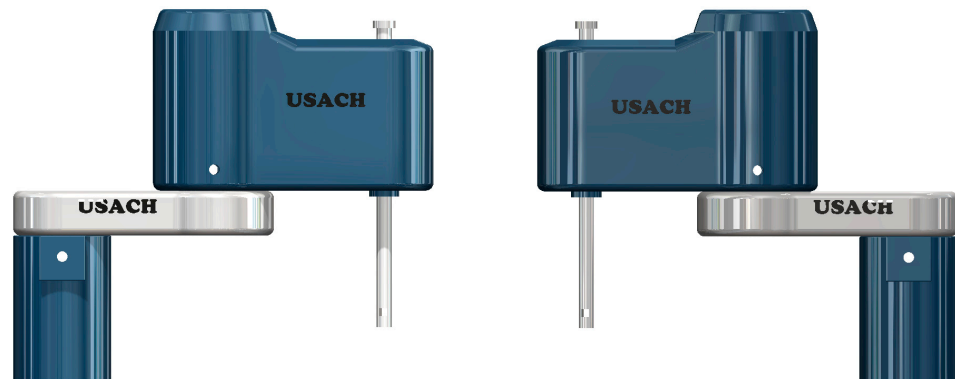


Figure 4. SCARA system prototype.

The different mechanical components developed in SolidWorks are available at the following link: <https://zenodo.org/records/14214239>.

The SolidWorks 2022 software includes a tool called physical properties, which allows obtaining the characteristics of the robot's links, such as their masses, dimensions, and moments of inertia, as shown in Table 2.

Table 2. Dimensions and weights of the SCARA robot links.

	Parameter	Value	Unit of Measure
Robot 1 and Robot 2	a_1	0.2	m
	a_2	0.2259	m
	d_1	0.1916	m
	d_4	0.02	m
	I_{zz1}	0.1	kg·m ²
	I_{zz2}	1.33	kg·m ²
	I_{zz3}	0.02	kg·m ²
	I_{zz4}	0.01	kg·m ²
	m_1	2.29	kg
	m_2	9.71	kg
	m_3	1.9	kg
	m_4	0.14	kg

where:

a_1, a_2, d_1, d_4 : Length of the robot arms.

m_1, m_2, m_3, m_4 : Masses of links 1, 2, 3, and 4, respectively.

$I_{zz1}, I_{zz2}, I_{zz3}, I_{zz4}$: Moments of inertia of links 1, 2, 3, and 4, respectively, along the z-axis.

The physical parameters of the robot were obtained directly from SolidWorks 2022, using the MMGS unit system (millimeter, gram, second), with precision set to four decimal places (± 0.0001 mm). This precision was selected to ensure a high level of accuracy in the robot's kinematic and dynamic calculations.

3.1.2. Mathematical Modeling of Selective Compliance Assembly Robots Arm

The mathematical modeling of SCARA robots is important as it allows for analyzing their kinematic and dynamic behavior. In this study, since both robots share the same characteristics and dimensions, only one robot will be analyzed, although the complete system is composed of two. The procedure follows the one outlined in [50]. Figure 5 shows the kinematic diagram of the SCARA robot, from which the Denavit–Hartenberg parameters are determined and presented in Table 3.

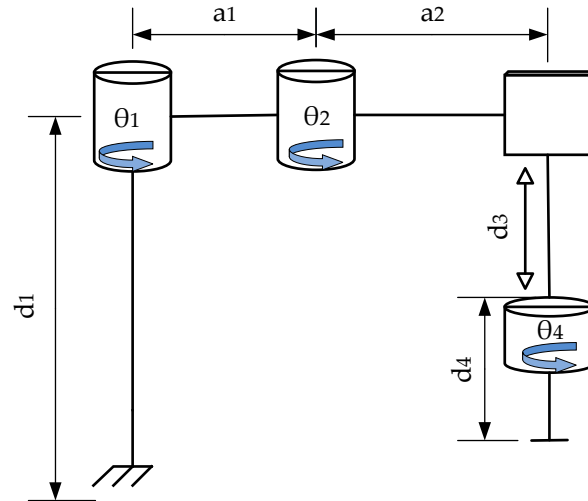


Figure 5. Kinematic diagram of the SCARA robot.

where:

a_1 : Length of the first arm.

a_2 : Length of the second arm.

d_1 : Height from the base to the first rotation axis (axis Z_0).

d_3 : Linear displacement of the end-effector along the axis Z_3 .

d_4 : Height of the end-effector.

θ_1 : Rotation angle of the first link (rotation around the axis Z_0).

θ_2 : Rotation angle of the second link (rotation around the axis Z_1).

θ_4 : Rotation angle of the end-effector (rotation around the axis Z_3).

Table 3. Denavit–Hartenberg parameters.

Joint _{<i>i</i>}	θ_i	d_i	a_i	α_i
1	θ_1	0.1916	0.2	0
2	θ_2	0	0.2259	π
3	0	d_3	0	0
4	θ_4	0.02	0	0

The forward kinematics allow calculating the position and orientation of the end-effector from the joint angles [60].

$$P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} = \begin{bmatrix} 0.2 \cos(\theta_1) + 0.2259 \cos(\theta_1 + \theta_2) \\ 0.2 \sin(\theta_1) + 0.2259 \sin(\theta_1 + \theta_2) \\ 0.1716 - d_3 \end{bmatrix} \quad (1)$$

where:

p_x, p_y, p_z : Positions of the end-effector on the x , y , and z axes, respectively.

In contrast, inverse kinematics determine the joint angles required to reach a desired position of the end-effector in the three-dimensional plane [50]. To determine the different parameters, it is based on the geometric analysis of the robot, considering a projection on the XY plane axis. From the desired position of the end-effector, a trigonometric relationship is established to determine the orientation of the links.

In Equation (2), the term $\arctan\left(\frac{p_y}{p_x}\right)$, calculates the direct angle toward the target position in the XY plane. The second term $\arctan\left(\frac{0.2259\sin(\theta_2)}{0.2+0.2259\cos(\theta_2)}\right)$ introduces a geometric correction that considers the additional displacement generated by the second link, whose length is 0.2259 m and whose orientation depends on θ_2 .

The following are the equations to determine each of the different parameters of inverse kinematics:

$$\theta_1 = \arctan\left(\frac{p_y}{p_x}\right) - \arctan\left(\frac{0.2259\sin(\theta_2)}{0.2 + 0.2259\cos(\theta_2)}\right) \quad (2)$$

$$\theta_2 = \arccos\left(\frac{p_x^2 + p_y^2 - 0.0910}{0.0904}\right) \quad (3)$$

$$d_3 = 0.1716 - p_z \quad (4)$$

$$\theta_4 = \varnothing + \theta_1 + \theta_2 \quad (5)$$

where:

\varnothing : Desired angle of the end-effector in the X, Y plane.

For the dynamic analysis, the inertia, Coriolis, and gravitational forces acting on the robot's joints are considered, as they allow analyzing its behavior during task execution. The equations used for this analysis have been taken directly from [50]:

$$M_i(q_i)\ddot{q}_i + C_i(q_i, \dot{q}_i)\dot{q}_i + G_i(q_i) = \tau_i \quad (6)$$

$$M_i = \begin{bmatrix} 0.1843\cos(\theta_2) + 2.034 & 0.0922\cos(\theta_2) + 1.464 & -0.03 & 0 \\ 0.0922\cos(\theta_2) + 1.464 & 1.464 & -0.03 & 0 \\ 0 & 0 & -0.03 & 0 \\ 0 & 0 & 0 & 0.14 \end{bmatrix} \quad (7)$$

$$C_i = \begin{bmatrix} 0 & 0.0461\sin(\theta_2)(2\dot{\theta}_1 + \dot{\theta}_2) & 0 & 0 \\ 0.0922\sin(\theta_2)(2\dot{\theta}_1 + \dot{\theta}_2) & 0.0461\dot{\theta}_1\sin(\theta_2) & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (8)$$

$$G_i = \begin{bmatrix} 0 \\ 0 \\ -20.01 \\ 0 \end{bmatrix} \quad (9)$$

$$q_i = \begin{bmatrix} \theta_1 \\ \theta_2 \\ d_3 \\ \theta_4 \end{bmatrix}; \dot{q}_i = \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{d}_3 \\ \dot{\theta}_4 \end{bmatrix}; \ddot{q}_i = \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \\ \ddot{d}_3 \\ \ddot{\theta}_4 \end{bmatrix} \quad (10)$$

where:

C_i : Coriolis matrix.

G_i : Gravitational forces matrix.

M_i : Inertia matrix.

q_i : Position vector.

\dot{q}_i : Velocity vector.

\ddot{q}_i : Acceleration vector.

τ_i : Torque vector.

$\dot{\theta}_1, \dot{\theta}_2, \dot{d}_3, \dot{\theta}_4$: Angular velocities of $\theta_1, \theta_2, \theta_4$ and linear velocity d_3 .

$\ddot{\theta}_1, \ddot{\theta}_2, \ddot{d}_3, \ddot{\theta}_4$: Angular accelerations of $\theta_1, \theta_2, \theta_4$ and linear acceleration of d_3 .

3.2. Fault Tolerance in Selective Compliance Assembly Robots

Currently, robotic systems have a significant impact on the industry, making it crucial to implement the concept of fault tolerance so that robots can continue operating without interruption, even in the presence of internal and external failures. This study focuses on actuator failures in cooperative SCARA robots, as these components are extremely critical for the precise and efficient movement of the robots. Such failures could severely compromise the performance of joint tasks, as the robot might lose precision and coordination in its movements. This issue is particularly critical in cooperative environments where strict synchronization is required: multiple robots performing complex tasks together, such as in assembly or part-handling chains.

The fault-tolerant strategy consists of three stages:

1. Fault detection.
2. Diagnosis.
3. Automatic system reconfiguration.

3.2.1. Fault Detection

The fault detection process is based on an algorithm that continuously compares the actual positions with the expected positions of the end-effector, which are calculated using forward kinematics. The logic is as follows:

1. Calculation of desired Cartesian positions: The position of the end-effector (p_{xd}, p_{yd}, p_{zd}) in the three-dimensional plane is obtained using forward kinematics based on the desired joint angles ($\theta_{1d}, \theta_{2d}, d_{3d}, \theta_{4d}$) and the geometric characteristics of the robots.
2. Obtaining actual Cartesian positions: To determine the actual positions (p_{xm}, p_{ym}, p_{zm}) of the robots' end-effectors, the transform sensor from the Simulink library is used. This sensor measures the positions (p_{xd}, p_{yd}, p_{zd}) of the end-effectors in real time, enabling continuous and precise measurement during system operation.
3. Comparison and calculation of Cartesian error (e_x, e_y, e_z): The error is calculated as the difference between the actual position measured by the sensors and the desired position.

$$e_x = p_{xd} - p_{xm}; e_y = p_{yd} - p_{ym}; e_z = p_{zd} - p_{zm} \quad (11)$$

$$E = \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} \quad (12)$$

where:

p_{xd}, p_{yd}, p_{zd} : Desired x, y, and z position of the end-effector.

p_{xm}, p_{ym}, p_{zm} : Measured x, y, and z position of the end-effector.

$\theta_{1d}, \theta_{2d}, d_{3d}, \theta_{4d}$: Desired position of joints 1, 2, 3, and 4, respectively.

e_x, e_y, e_z : Cartesian error in x, y, and z.

E: Cartesian error vector.

- Evaluation against a threshold: If the error $\|E\|$ exceeds a defined margin ϵ , the system detects a fault. This threshold is adjusted to avoid false detections caused by noise or other minor disturbances in the system.

$$\|E\| = \sqrt{e_x^2 + e_y^2 + e_z^2} \quad (13)$$

$$\text{If } \begin{cases} \|E\| > \epsilon & \text{a fault has occurred} \\ \|E\| < \epsilon & \text{no fault has occurred} \end{cases} \quad (14)$$

This flowchart is presented in Figure 6, detailing how the desired and actual positions are processed to detect anomalies.

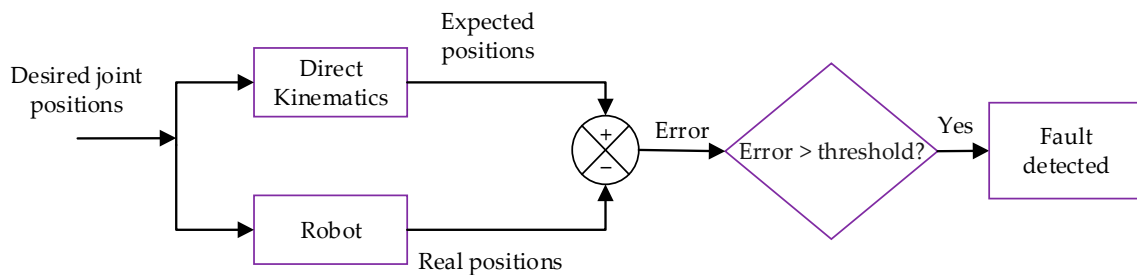


Figure 6. Fault detection in the SCARA system.

3.2.2. Fault Diagnosis

Once a fault is detected in the system, it is crucial to determine the cause and identify which actuator or joint is generating the error. An algorithm analyzes the joint errors in each joint following this logic:

- Calculation of desired joint positions: The desired positions $(\theta_{1d}, \theta_{2d}, d_{3d}, \theta_{4d})$ for each joint are obtained from the trajectories previously defined in the control system.
- Obtaining actual joint positions $(\theta_{1m}, \theta_{2m}, d_{3m}, \theta_{4m})$: These are measured in real time using the Joint Sensor in Simulink, integrated into the revolute joints of the model. This sensor provides the current angular values of each joint.
- Comparison and calculation of joint error: The error is calculated as the difference between the real position measured by the sensors and the desired position.

$$e_{\theta_1} = \theta_{1d} - \theta_{1m}; e_{\theta_2} = \theta_{2d} - \theta_{2m}; e_{d_3} = d_{3d} - d_{3m}; e_{\theta_4} = \theta_{4d} - \theta_{4m}; \quad (15)$$

where:

$\theta_{1m}, \theta_{2m}, d_{3m}, \theta_{4m}$: Measured position of joints 1, 2, 3, and 4, respectively.

$e_{\theta_1}, e_{\theta_2}, e_{d_3}, e_{\theta_4}$: Errors in joints 1, 2, 3, and 4, respectively.

- Evaluation against a threshold: Each joint error is compared with a specific threshold defined for each joint. This threshold is set based on the allowed tolerances for the correct operation of the system. If the error exceeds the threshold, the system diagnoses that the corresponding actuator is failing.

$$\text{If } \begin{cases} \|e_{\theta_i}\| > \epsilon & \text{actuator is defective} \\ \|e_{\theta_i}\| < \epsilon & \text{actuator is in good condition} \end{cases} \quad (16)$$

where:

ϵ : Threshold.

e_{θ_i} : Error in joint i .

This flowchart is presented in Figure 7, where it details how the desired and actual positions are processed to detect a faulty actuator.

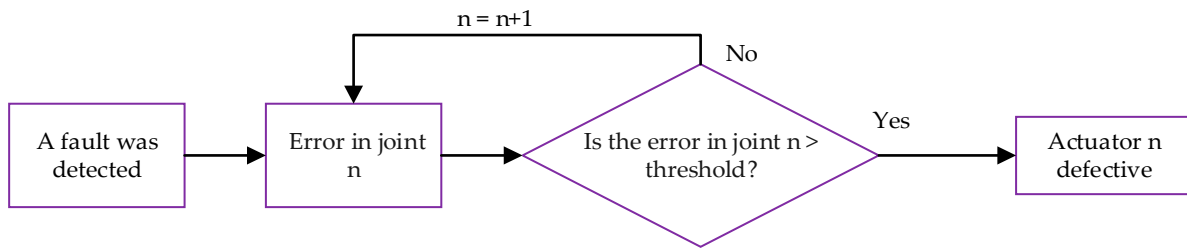


Figure 7. Fault diagnosis SCARA system.

3.2.3. System Reconfiguration

System reconfiguration begins immediately after diagnosing the failed actuator. Since the system is designed with a redundant configuration, each robot joint has two actuators: a primary actuator and a redundant actuator. This setup ensures that in the event of a critical failure in the primary actuator, the system can continue operating without interruptions. When a failure in the primary actuator is detected, the system automatically switches to the redundant actuator. This process involves:

1. Deactivating the failed actuator: The defective actuator is deactivated to prevent interference with system operation.

$$A_i = 0 \quad (17)$$

2. Activating the redundant actuator: The redundant actuator is activated and takes control of the affected joint, dynamically adjusting to the current operating conditions.

$$A_{ir} = 1 \quad (18)$$

where:

A_i : Primary actuator i .

A_{ir} : Redundant actuator i .

3. Parameter adjustment: The redundant actuator adapts its settings to maintain system accuracy, performance, and stability within the previously established operational limits.

$$\theta_{ir} = \theta_{id} - \theta_{Ai} \quad (19)$$

where:

θ_{ir} : Redundant joint i position.

θ_{Ai} : Position where the primary actuator i failed.

θ_{id} : Desired position of joint i .

Automatic and real-time reconfiguration minimizes productivity loss and eliminates major impacts on the process. The reconfiguration strategy ensures that the robotic system continues operating without performance loss, maintaining work quality by avoiding serious errors that could result from a failure in the primary actuator (see Figure 8).

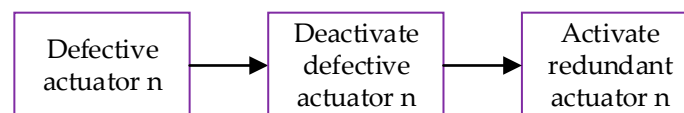


Figure 8. System reconfiguration.

3.3. Gear System for Activation and Deactivation of Actuators in the Different Joints

To equip SCARA robots with redundant actuators, differential gears are used to enable controlled distribution of torque and movement to the actuators. This type of mechanism is employed in applications where combining mechanical efforts is necessary to ensure precise synchronization of the actuators.

The differential gear is a planetary gear system with a central gear that helps distribute torque among the other connected gears. The main advantage of this system is that it allows the shafts to rotate at different speeds while maintaining proportional torque distribution. This is particularly useful in systems where a primary actuator performs the work, and a redundant actuator remains inactive until a failure in the primary actuator is detected. Figure 9 shows the differential gear system to be used, which was designed in SolidWorks 2022.

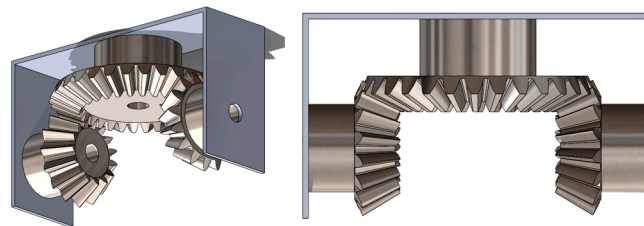


Figure 9. Differential gear system.

The differential gears in SCARA robots allow two actuators to be connected to the same joint. Under normal conditions, the primary actuator moves the joint, while the redundant actuator remains inactive. If the primary actuator fails, the control system activates the redundant actuator to take full control of the joint. This ensures that the robot continues operating without interruptions, even in the event of mechanical failures in one of its primary actuators.

3.3.1. Servomotor

A servomotor is an electromechanical device designed to control the position, speed, and acceleration of a mechanical system. In SCARA robots and their differential system, servomotors are used to drive the planetary gears, enabling efficient torque distribution and precise joint control.

Servomotors are composed of several key elements:

DC Motor: Provides rotational movement.

Gear System: Multiplies torque and reduces speed to achieve high precision.

Position Sensor: Measures the shaft's position and transmits this information to the controller.

Controller: Adjusts the motor's movement to correct deviations from the desired position.

Feedback Circuit: Maintains closed-loop control between input and output, ensuring instant corrections for precise positioning.

The primary importance of servomotors in differential gear systems lies in controlling the movement of the planetary gears with the precision needed to ensure the system's proper operation. Servomotors enable continuous adjustments and real-time feedback, ensuring that, even in the event of a failure in one of the actuators, SCARA robots maintain synchronization and joint control. The servomotor prototype design is created using SolidWorks 2022 software (see Figure 10).

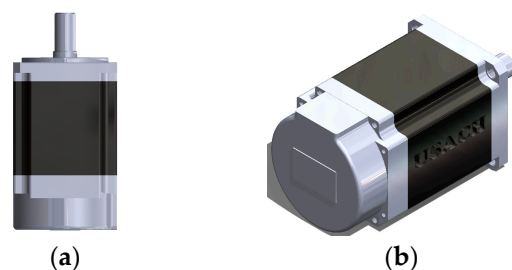


Figure 10. Servomotor prototype: (a) front view; (b) isometric view.

Servomotors can exhibit errors that significantly affect the system and are sometimes difficult to diagnose. One of the most common issues is overheating. This occurs when the motor is overworked or when there are problems with ventilation or the cooling system. Excessive heat can impair the motor's performance and potentially cause permanent damage to the device.

Another frequent fault is the wear and tear of mechanical components such as gears, bearings, and other moving parts, which deteriorate over time and usage. This can result in vibrations, noise, or even cause the motor to operate unevenly or imprecisely. Electronic-related failures can also occur, specifically in the internal circuits, which may be damaged by voltage issues, short circuits, or electrical interference. If a servomotor's controller fails, the motor might stop functioning correctly or become uncontrolled, which can be hazardous in sensitive applications.

The differential gear system prototype design is shown in Figure 11, created with SolidWorks 2022 software.

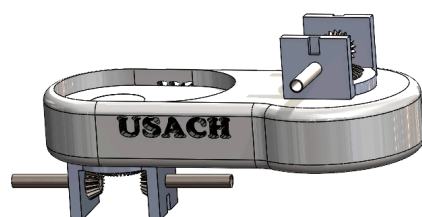


Figure 11. Differential gear system prototype.

3.3.2. Torque Limiter

A torque limiter is a device engineered to safeguard machinery and motors from damage caused by overloads. Its core function is to disengage the torque once a preset limit is surpassed, thereby preventing overheating or component failure.

The torque limiter connects to the shaft through a pin, facilitating power transfer. This pin, which may be either a parallel pin or a spring pin, fits into both the shaft and the groove-shaped resin part within the torque limiter's housing.

The driving end of the system is linked to the torque limiter's plastic sleeve, while the output side attaches to the aluminum casing. When the torque on the output side is below the designated limit, power flows seamlessly, with no internal slippage occurring within the limiter.

If the torque on the output side exceeds the threshold, the torque limiter activates by slipping internally, interrupting the power from the motor. This mechanism shields the system, minimizing the risk of damage to the motor and related mechanical parts. Once the torque falls back within safe levels, the torque limiter restores power transmission without internal slipping.

These devices are commonly used in automatic doors, electric windows, motorized curtains, and other motor-driven systems. A notable example is the TOK TLEU torque limiter.

The TOK TLEU stands out for its compact and lightweight design, high durability, and stable torque output. Unlike conventional friction-based limiters, this model uses magnetic coupling to generate torque, ensuring quiet operation. It offers adjustable torque settings ranging from 5 to 40 Nm, in 5 Nm increments, and can operate at a maximum speed of 2000 rpm.

Torque limiters provide three major advantages: protects components and motor during overloads, operates quietly due to magnetic force rather than friction, and automatically restores power transmission once torque returns to safe levels.

The design of the torque limiter used in the SCARA robot is shown in Figure 12, created with SolidWorks 2022 software.

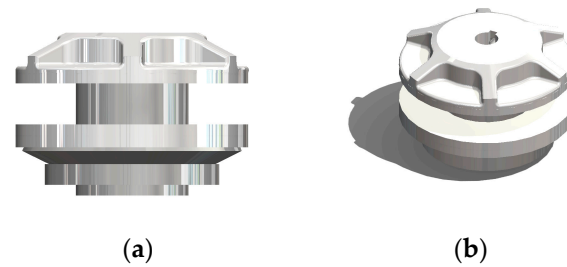


Figure 12. Torque limiter prototype: (a) front view; (b) isometric view.

3.3.3. Mathematical Analysis of the Servomotor System with Differential Gear

The mathematical analysis of the behavior of the servomotors, along with the differential gear system, is presented below. This system enables the distribution of torque between the primary and redundant actuators in SCARA robots. Figure 13 illustrates the electrical and mechanical model of two servomotors, each represented by an armature circuit and a dynamic model that includes inertia, friction, and torque.

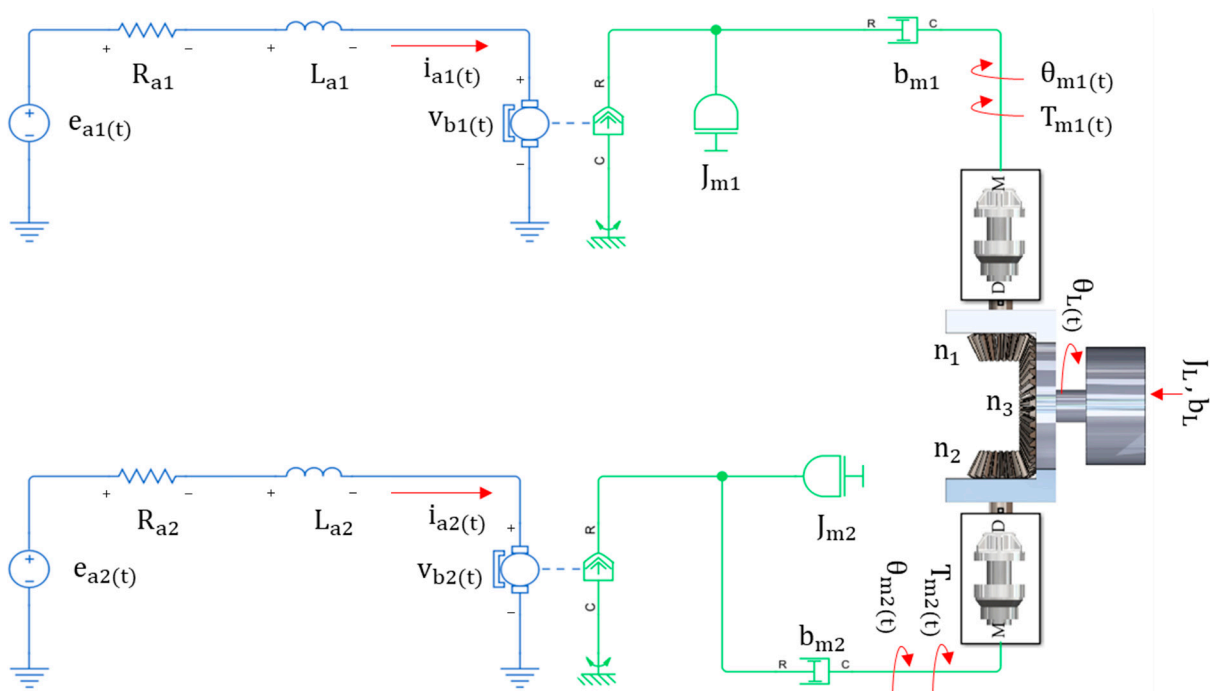


Figure 13. Electromechanical circuit of the differential system.

These servomotors act on a differential gear, which distributes movement and effort among the robot's joints, ensuring efficient and precise control. A detailed analysis of this system is essential to understand how the servomotors interact with the differential

gear and how variations in torque and speed are managed to ensure synchronized and fault-free operation.

To analyze the system, the electrical and mechanical equations of each motor are formulated:

$$\begin{aligned} e_{a1} &= R_{a1}i_{a1} + L\frac{di}{dt} + v_{b1} \\ e_{a2} &= R_{a2}i_{a2} + L\frac{di}{dt} + v_{b2} \end{aligned} \quad (20)$$

$$\begin{aligned} T_{m1} &= k_{t1}i_{a1} \\ T_{m2} &= k_{t2}i_{a2} \end{aligned} \quad (21)$$

$$\begin{aligned} v_{b1} &= k_{b1}\dot{\theta}_{m1} \\ v_{b2} &= k_{b1}\dot{\theta}_{m2} \end{aligned} \quad (22)$$

The torque in each motor is calculated using the following formulae:

$$\begin{aligned} T_{m1} &= J_{m1}\ddot{\theta}_{m1} + b_{m1}\dot{\theta}_{m1} \\ T_{m2} &= J_{m2}\ddot{\theta}_{m2} + b_{m2}\dot{\theta}_{m2} \end{aligned} \quad (23)$$

where:

T_{m1}, T_{m2} : Torque generated by motor 1 and motor 2.

v_{b1}, v_{b2} : Back electromotive force of motor 1 and motor 2.

k_{t1}, k_{t2} : Torque constant of motor 1 and motor 2.

$\dot{\theta}_{m1}, \dot{\theta}_{m2}$: Angular velocity w_{m1} and w_{m2} .

J_{m1}, J_{m2} : Rotor's moment of inertia.

b_{m1}, b_{m2} : Viscous friction coefficient.

To determine the relationship between the actuators and the gear system, the following equations are used:

$$\begin{aligned} J_{e1} &= -J_{m1} - J_L\left(\frac{\theta_L}{\theta_{m1}}\right)^2 \\ J_{e2} &= -J_{m2} - J_L\left(\frac{\theta_L}{\theta_{m2}}\right)^2 \end{aligned} \quad (24)$$

$$\begin{aligned} T_{m1} &= J_{e1}\ddot{\theta}_L + b_{e1}\dot{\theta}_L \\ T_{m2} &= J_{e2}\ddot{\theta}_L + b_{e2}\dot{\theta}_L \end{aligned} \quad (25)$$

$$T_{m1} = \left[-J_{m1} - J_L\left(\frac{\theta_L}{\theta_{m1}}\right)^2 \right] \ddot{\theta}_L \quad (26)$$

$$T_{m2} = \left[-J_{m2} - J_L\left(\frac{\theta_L}{\theta_{m2}}\right)^2 \right] \ddot{\theta}_L$$

$$e_{a1} = L \frac{\left[-J_{m1} - J_L\left(\frac{\theta_L}{\theta_{m1}}\right)^2 \right]}{k_{t1}} \ddot{\theta}_L - R \frac{\left[-J_{m1} - J_L\left(\frac{\theta_L}{\theta_{m1}}\right)^2 \right]}{k_{t1}} \dot{\theta}_L + k_{b1}\dot{\theta}_L \quad (27)$$

$$e_{a2} = L \frac{\left[-J_{m2} - J_L\left(\frac{\theta_L}{\theta_{m2}}\right)^2 \right]}{k_{t2}} \ddot{\theta}_L - R \frac{\left[-J_{m2} - J_L\left(\frac{\theta_L}{\theta_{m2}}\right)^2 \right]}{k_{t2}} \dot{\theta}_L + k_{b1}\dot{\theta}_L$$

The characteristics of the servomotor are as follows:

Torque constant: 0.056 [Nm/A].

Voltage constant: 0.0554 [Nm/A].

To calculate the value of the friction coefficient b in both motors, the no-load speed and current values are used.

$$b = \frac{T_e}{w_r} \quad (28)$$

$$\omega_r = \frac{V_a - R i_a}{k_t} = \frac{24 - (1.15)(0.5)}{0.056} = 418.30 \text{ [rad/seg.]} \quad (29)$$

The torque generated solely by the rotation of the shaft is known.

$$T_e = \frac{k_t i_a}{k_t} = 0.0280 \text{ [Nm]} \quad (30)$$

$$b = \frac{T_e}{\omega_r} = 6.6937 \times 10^{-5} \text{ [Nm]} \quad (31)$$

To analyze the operation of the actuators, it is important to examine their energy performance, which is defined by the relationship between the electrical and mechanical power in each motor.

$$\begin{aligned} P_{e1} &= e_{a1} i_{a1} \\ P_{e2} &= e_{a2} i_{a2} \end{aligned} \quad (32)$$

$$\begin{aligned} P_{m1} &= T_{m1} \dot{\theta}_{m1} \\ P_{m2} &= T_{m2} \dot{\theta}_{m2} \end{aligned} \quad (33)$$

$$\begin{aligned} n_1 &= \frac{P_{e1}}{P_{m1}} \times 100\% \\ n_2 &= \frac{P_{e2}}{P_{m2}} \times 100\% \end{aligned} \quad (34)$$

where:

P_{e1}, P_{e2} : Electrical power of motor 1 and motor 2.

P_{m1}, P_{m2} : Mechanical power of motor 1 and motor 2.

n_1, n_2 : Efficiency of motor 1 and motor 2.

3.4. Coordination Algorithm

In a robotic system where, multiple robots collaborate on the same task, it is crucial to implement a coordination algorithm that ensures the efficient and safe operation of the robotic cell. This algorithm must manage each robot's movements to maintain coordination and prevent collisions, which could severely affect system performance and durability. To achieve proper coordination, it is essential to have detailed knowledge of the different positions and working areas each robot can reach. Assigning a task outside the reach of its end-effector could compromise execution, causing delays or system failures.

This study focuses on a complex scenario where the robots must collect objects of different colors located within their respective working areas and transport them to the common working area, where both robots can operate together. Real-time communication between the robots is vital in this context. They must exchange information about their positions and plan their movements in a coordinated manner to avoid collisions while transporting or manipulating objects in the common area. Figure 14 shows the working areas: Robot 1's area is in red, Robot 2's area is in green, and the common working area is in light blue.

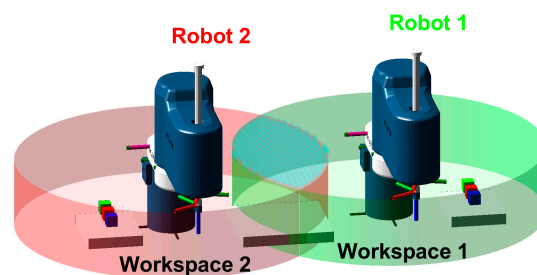


Figure 14. Working area of Robot 1 and Robot 2.

Continuing with the cooperative task, Figure 15 shows the collisions in position and orientation that can occur due to poor communication between robots when they need to jointly transport a large object, such as a table. In this case, it is critically important for the robots to constantly exchange information about the positions and orientations of their end-effectors. Without this precise communication, they would be unable to move the object correctly at the same time, jeopardizing both the task and the integrity of the system.



Figure 15. Cooperative system in collision: (a) position collision; (b) orientation collision.

Figure 16 shows the proposed algorithm for coordinating the robots in performing a cooperative task.

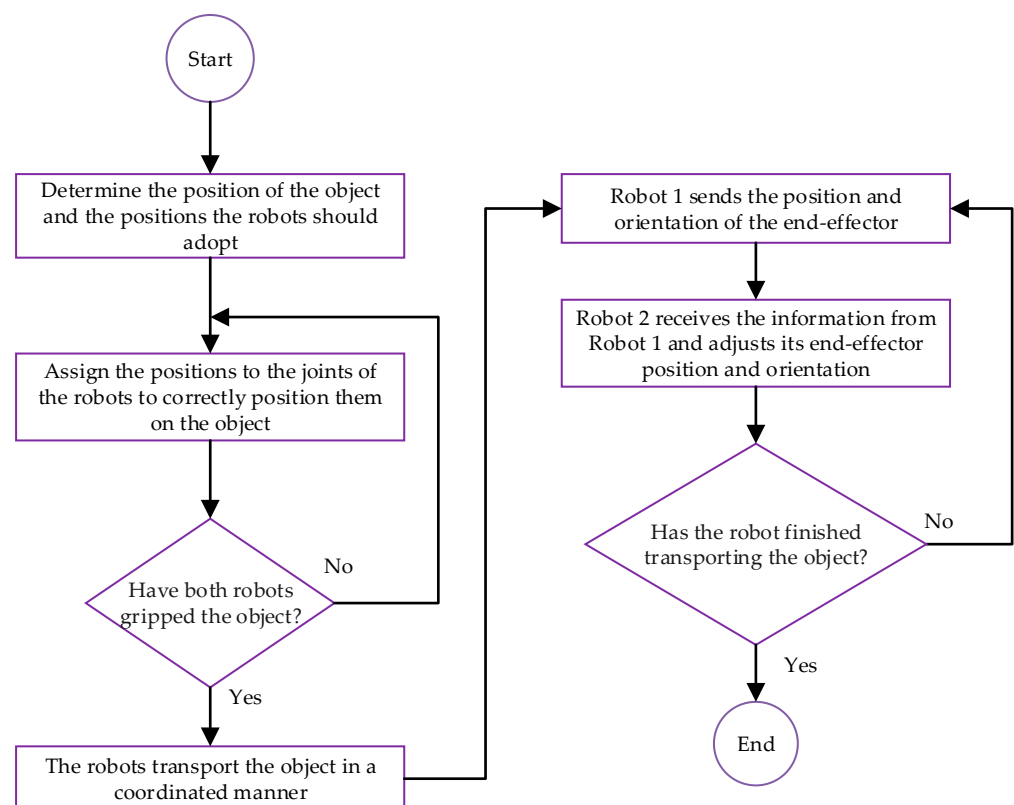


Figure 16. Algorithm for coordinating the robots when performing cooperative tasks.

The process begins with determining the object's position and the Cartesian positions the robots must adopt. Next, joint positions are assigned to the robots so that their end-effectors are properly positioned over the object. It is then verified whether both robots have gripped the object; if the response is affirmative, the robots transport it in a coordinated manner. Otherwise, the process enters a delay until the object is secured.

During transport, Robot 1 sends the position and orientation of its end-effector to Robot 2, which adjusts its parameters accordingly. The process continues until the transport is completed.

3.5. Controllers

The robots, in addition to having good coordination among themselves when performing a task, must also use controllers capable of minimizing the different failures that may arise during the task, such as disturbances and vibrations. Therefore, the following controllers are used:

3.5.1. Nonlinear Model Predictive Controller

Nonlinear Model Predictive Controller (NLMPC) is an advanced controller used to handle systems with complex nonlinearities in their dynamics. Its main objective is to optimize the system's behavior over time by anticipating possible deviations and adjusting control actions optimally within a prediction horizon.

Starting from the dynamic equation presented in Equation (6), the control law is formulated, where the state vector x consists of the four joint positions q , the four velocities \dot{q} , and the four control torques, represented by τ .

$$\begin{aligned} \dot{x} &= f(x, \tau) \\ x &= \begin{bmatrix} q \\ \dot{q} \end{bmatrix} \end{aligned} \quad (35)$$

The cost function is represented by

$$J = \sum_{k=0}^{N_p} \left[(q_k - q_{ref,k})^T Q (\dot{q}_k - \dot{q}_{ref,k}) + \tau_k^T R \tau_k \right] \quad (36)$$

where:

q_k : Predicted joint position at step k .

$q_{ref,k}$: Reference joint position at step k .

\dot{q}_k : Predicted joint velocity at step k .

$\dot{q}_{ref,k}$: Reference joint velocity at step k .

τ_k : Control torques at step k .

Q y R : Weighting matrices 4×4 .

Constraints are established for positions, velocities, and torques.

$$\begin{aligned} q_{\min} &\leq q_k \leq q_{\max} \\ \dot{q}_{\min} &\leq \dot{q}_k \leq \dot{q}_{\max} \\ \tau_{\min} &\leq \tau_k \leq \tau_{\max} \end{aligned} \quad (37)$$

The NLMPC solves the following optimization problem at each time step, applying the first torque of the optimal sequence to the robot inputs, and the process repeats at the next time step.

$$\min_{\tau_0, \dots, \tau_{N_p-1}} J \quad (38)$$

Due to its advantages, this type of controller is widely used in applications with critical constraints, such as advanced robotics, complex chemical processes, and autonomous vehicles.

3.5.2. Sliding Mode Controller

The Sliding Mode Controller (SMC) is a control technique specifically used in robotic manipulators due to its robustness against uncertainties and external disturbances. Its operation forces the system to slide over a sliding surface, thus ensuring accurate and precise trajectory tracking.

This technique is based on the general dynamics of the robot, expressed in Equation (6). For this study, the following control equation is proposed:

$$M(q)(\ddot{q}_{\text{ref}} - \Lambda(q_{\text{ref}} - q) + C(q, \dot{q})\dot{q} + G(q) + k \cdot \text{sign}(s) = \tau \quad (39)$$

The term $M(q)(\ddot{q}_{\text{ref}} - \Lambda(q_{\text{ref}} - q))$ is used to compensate for the natural dynamics of the robot and adjust the reference acceleration, ensuring that the system responds correctly to changes in the desired trajectory. On the other hand, the term $k \cdot \text{sign}(s)$ introduces robustness against disturbances and variations in the dynamic parameters of the robots.

The sliding surface s is defined as a function of the joint positions q and velocities \dot{q} , where

$$s = \dot{q} - \dot{q}_{\text{ref}} + \Lambda(q - q_{\text{ref}}) \quad (40)$$

where:

\ddot{q}_{ref} : Reference acceleration.

k : Control gain vector that determines the magnitude of the discontinuity to ensure robustness.

$\text{sign}(s)$: Sign function applied component-wise. If $s > 0$, its value is 1; otherwise, it is -1 .

s : Sliding surface vector.

Λ : Positive gain matrix to adjust convergence toward the surface.

q_{ref} : Desired joint position vector.

The main objective of the controller is to force $s = 0$ to ensure that q converges to q_{ref} , thereby minimizing tracking error.

To reduce errors caused by chattering, the sign function is smoothed using the following equation:

$$\text{sign}(s) = \frac{s}{\epsilon + |s|} \quad (41)$$

where:

ϵ : is a small value that smooths the sign function, preventing abrupt changes in the control signal.

4. Simulations in the Virtual Environment of the Selective Compliance Assembly Robot Arm Cooperative System

A virtual environment is designed in MATLAB 2024a with the aim of evaluating and perfecting the performance of the two SCARA robots when performing cooperative tasks, as shown in Figure 17. The robots are programmed to manipulate and transport cube-shaped objects, which are initially placed on tables located to the right and left of the robots. Finally, the cubes must be placed on a common table situated within the shared workspace of the robots. The initial and final positions of the centers of the objects are shown in Tables 4 and 5.

To address this scenario and analyze the behavior of the robots in the event of a potential failure in one of their actuators, five cases are proposed:

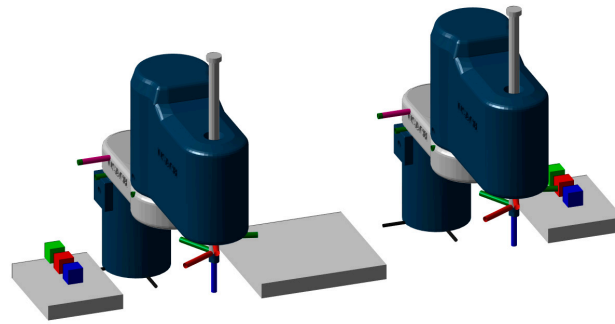


Figure 17. Virtual environment of the SCARA cooperative system.

First case: The robots are studied under normal operating conditions, meaning no actuator fails, and both robots' function with all their primary actuators while the redundant actuators remain idle. In this case, it is evaluated whether the robots can correctly perform the assigned task in the following order: first, the robots pick up the green objects and transport them to the box located in the shared workspace area; then, they perform the same operation with the red objects, and finally with the blue objects. This process is shown in Figure 18.

Table 4. Initial positions in the three-dimensional plane of the objects to be transported.

	Robot 1			Robot 2		
	X (m)	Y (m)	Z (m)	X (m)	Y (m)	Z (m)
Green Objects	0.2	0.1	0.1	−0.2	0.1	0.1
Red Objects	0.2	0.15	0.1	−0.2	0.15	0.1
Blue Objects	0.2	0.2	0.1	−0.2	0.2	0.1

Table 5. Final positions in the three-dimensional plane of the objects to be transported.

	Robot 1			Robot 2		
	X (m)	Y (m)	Z (m)	X (m)	Y (m)	Z (m)
Green Objects	0.25	0.1	0.1	0.2	0.1	0.1
Red Objects	0.25	0.15	0.1	0.2	0.15	0.1
Blue Objects	0.25	0.2	0.1	0.2	0.2	0.1

Second case: The cooperative system is evaluated by applying total failures in the robots' actuators. First, a total failure is induced in the first actuator of Robot 1 while it is transporting the red object to Robot 2, and later, a failure occurs in the second actuator of Robot 2 while it is transporting the blue object at 20 s. The goal is to analyze how these failures affect the robots' ability to complete the task accurately and cooperatively.

In this case, the robots can still move the objects, but due to the loss of control in the affected actuators, they fail to position them in the correct location within the shared workspace. This scenario allows for the evaluation of the system's robustness in the face of critical failures and observes the robots' behavior in degraded operating conditions. Additionally, the coordination algorithm's performance is verified; although the robots do not place the objects in the correct location, the algorithm prevents collisions between them, ensuring they do not interfere or collide with each other. These processes are shown in Figures 19 and 20. Meanwhile, Figures 21 and 22 show the joint positions of the primary and redundant actuators, where it is verified that a failure occurred in the actuator.

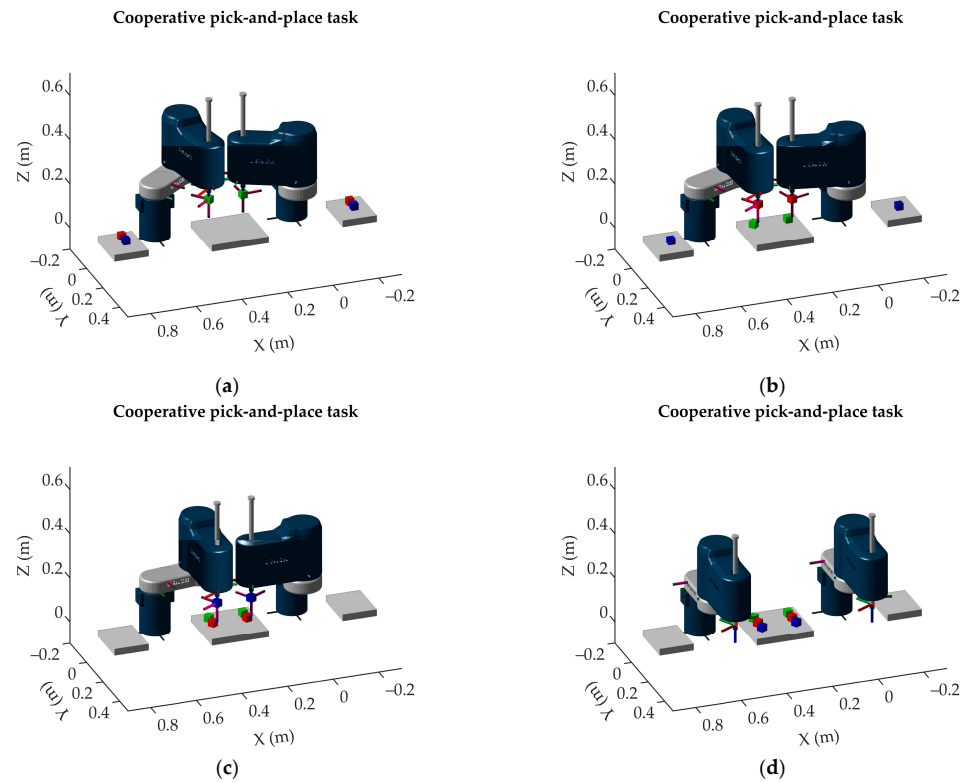


Figure 18. Cooperative object manipulation process by the SCARA robots: (a) movement of green objects; (b) movement of red objects; (c) movement of blue objects; (d) completion of the cooperative task.

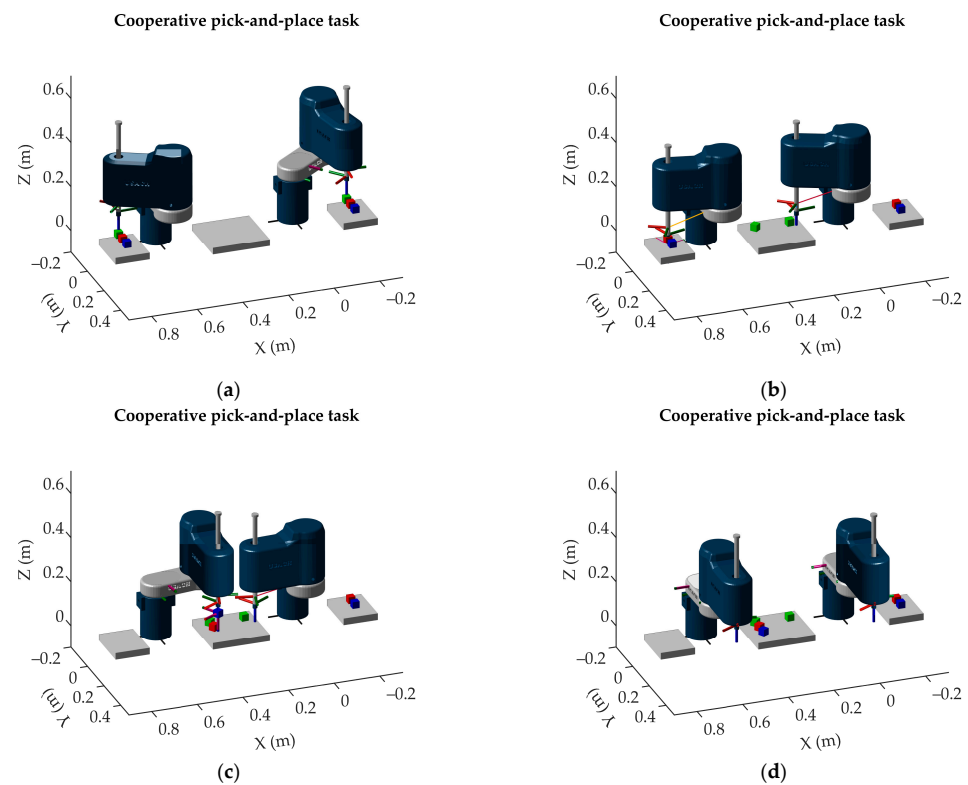


Figure 19. Cooperative object manipulation process by the SCARA robots with a failure in Actuator 1 of Robot 1: (a) movement of green objects; (b) the robots transport red objects with the first actuator of Robot 1 defective; (c) movement of blue objects with the first actuator of Robot 1 defective; (d) completion of the inefficient cooperative task.

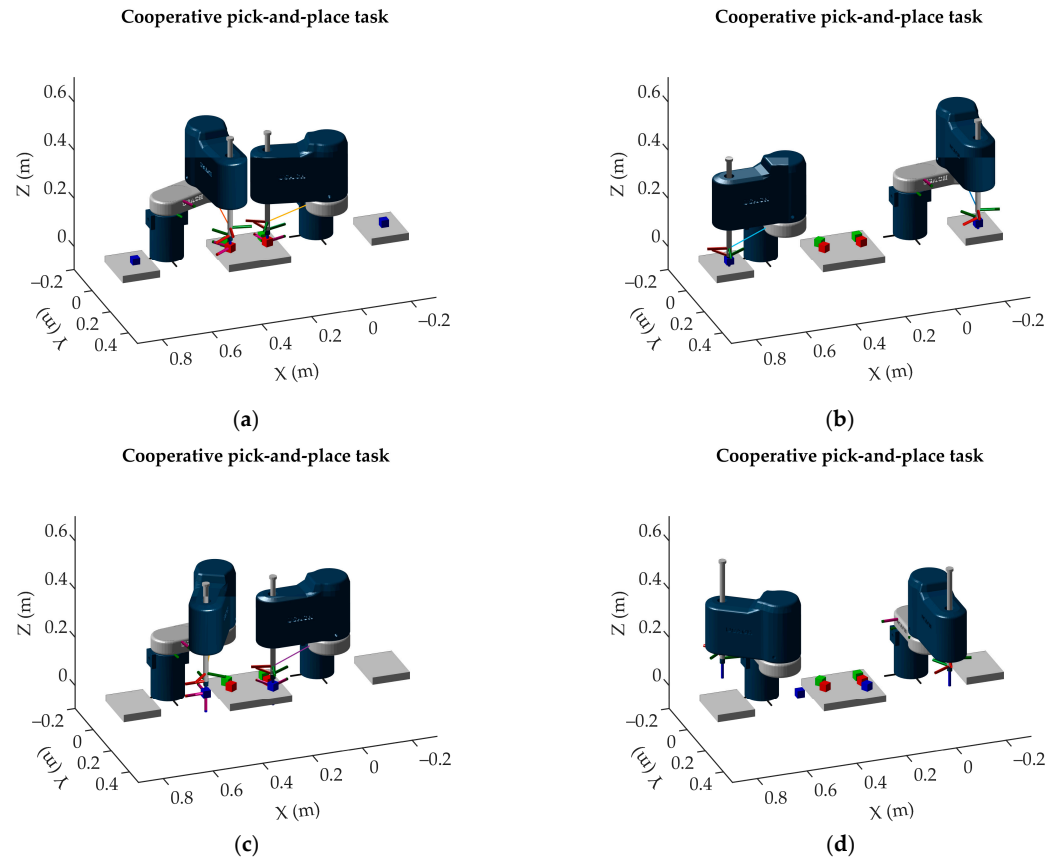


Figure 20. Cooperative object manipulation process by the SCARA robots with a failure in actuator 2 of Robot 2: (a) movement of red objects; (b) the robots hold blue objects; (c) movement of blue objects with the second primary actuator of Robot 2 defective; (d) completion of the inefficient cooperative task.

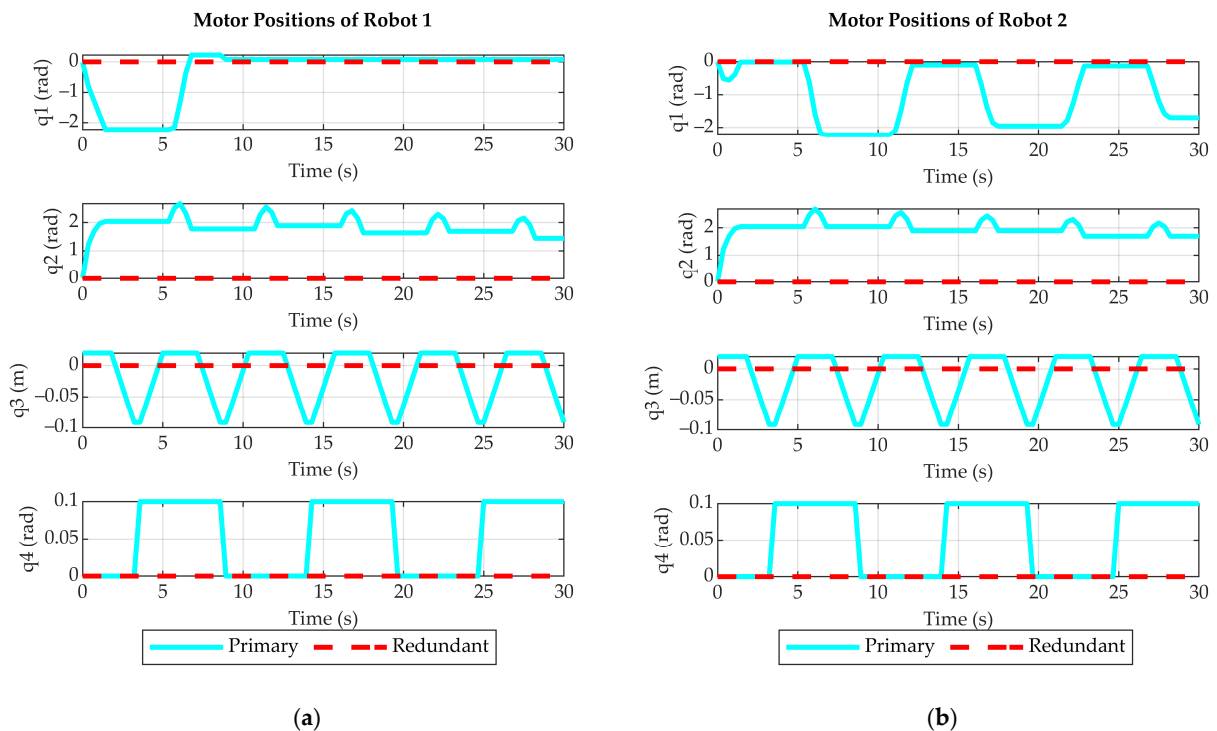


Figure 21. Positions in the primary and redundant actuators of the SCARA robots with a failure in Actuator 1 of Robot 1: (a) Robot 1; (b) Robot 2.

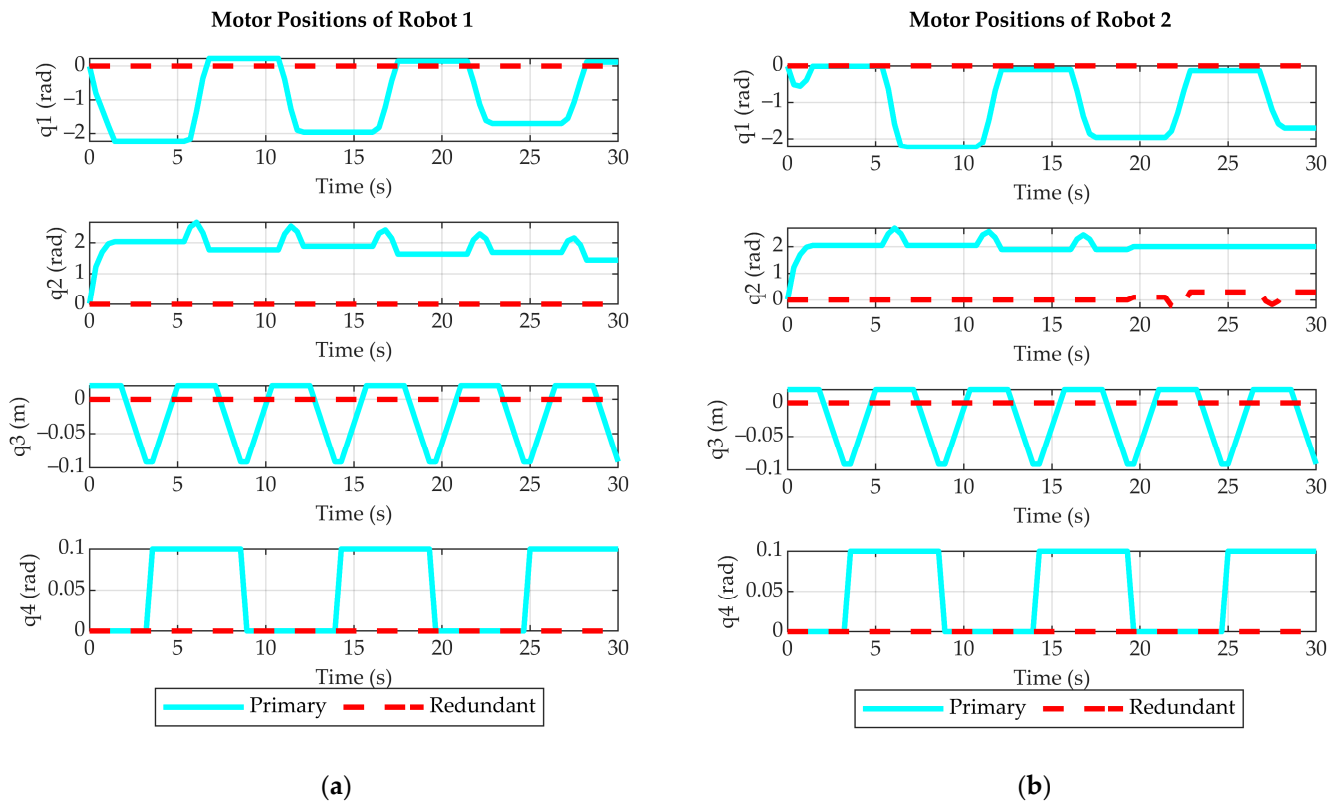


Figure 22. Positions in the primary and redundant actuators of the SCARA robots with a failure in the second primary actuator of Robot 2: (a) Robot 1; (b) Robot 2.

Third case: The system is evaluated with the fault-tolerant mechanism, adding the task from the first and second cases. The same failures as in the second case are induced (failure in the first actuator of Robot 1 and the second actuator of Robot 2), but now the fault-tolerant system is active. Upon detecting a failure, the system immediately deactivates the faulty primary actuator and activates the redundant actuator, allowing the robots to continue with their tasks.

Although there is a slight deviation in the robots' trajectory before the redundant actuator is activated, the system responds correctly, adjusting the positions and trajectories to complete the movement of the objects without compromising precision or coordination. This scenario demonstrates the system's adaptability to adverse conditions and evaluates how the fault-tolerance mechanism maintains stability in cooperative tasks, avoiding collisions and completing the task without interference between the robots or critical errors in the final positioning of the objects. Figure 23 shows the measured and performed trajectories of the robots, and Figure 24 illustrates the activation of redundant actuators when the primary actuators fail.

Fourth case: The behavior of the SCARA robotic system is evaluated in the presence of multiple actuator failures in both robots, with the aim of analyzing how the system responds and adapts to these simultaneous failures.

For this, the following failures are induced in Robot 1:

1. At 8 s, Actuator 1 fails while the robot is heading towards the red box.
2. At 14 s, Actuator 2 fails during the transport of the red box.
3. At 25 s, Actuator 3 fails while the robot is transporting the blue box.
4. At 11 s, Actuator 4 fails while the robot continues transporting the red box.

Trajectory executed by the robots to collect the objects

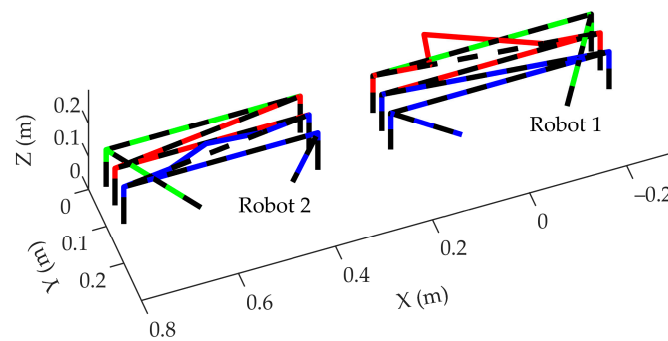


Figure 23. Desired and measured trajectories of the SCARA robots.

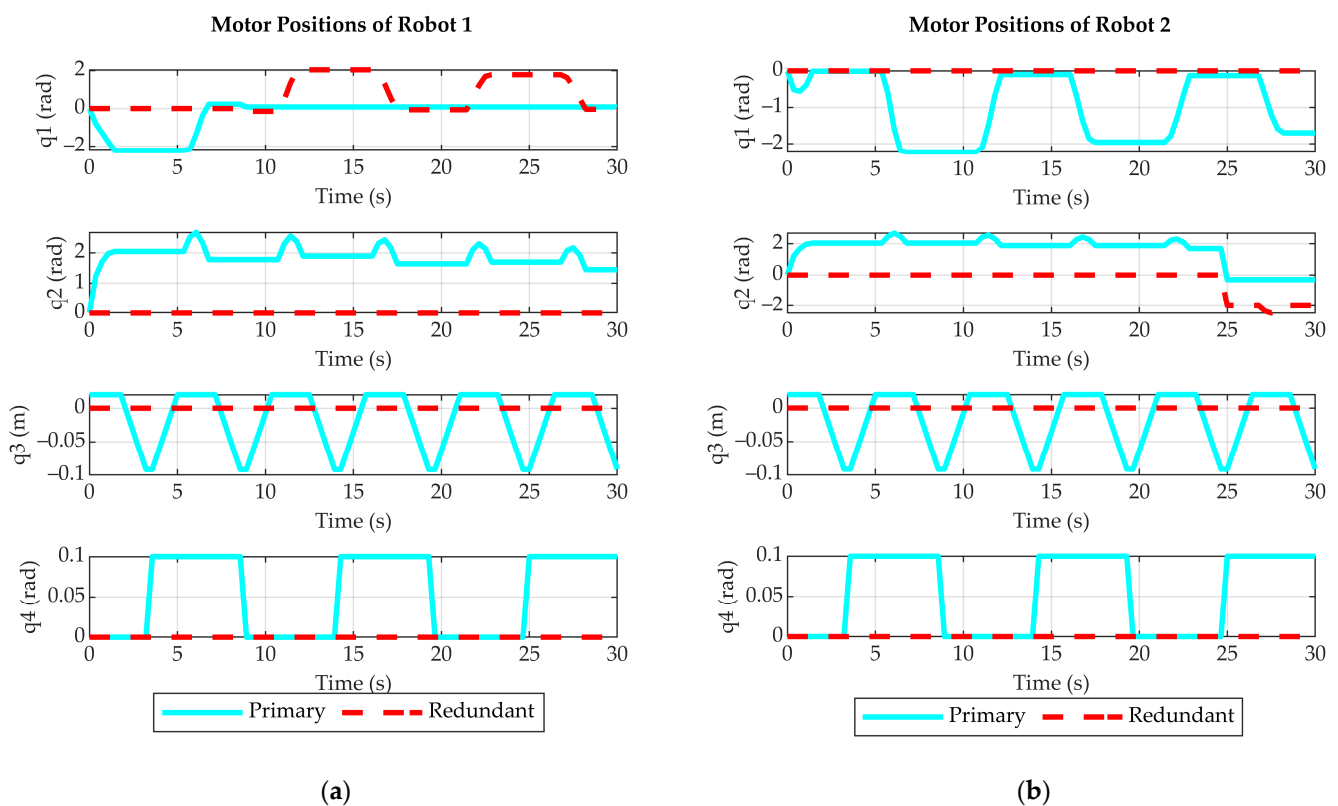


Figure 24. Positions in the primary and redundant actuators of the SCARA robots with a failure in the first primary actuator of Robot 1 and the second primary actuator of Robot 2: (a) Robot 1; (b) Robot 2.

In Robot 2, the following failures are induced:

1. At 14 s, Actuator 1 fails while transporting the red box.
2. At 19 s, simultaneous failures are induced in actuators 2 and 3 while the robot is heading towards the blue box.
3. At 4 s, actuator 4 fails while transporting the green box.

The fault tolerance system is active, which allows observing in Figure 25 how failures are managed in real time while the robots perform the tasks.

When the failures are induced, the system reacts quickly, as shown in Figure 26, i.e.:

- Deactivation of the defective actuator: When an actuator fails, the system automatically deactivates it to prevent the failure from affecting the robot's performance.

- Activation of redundant actuators: The system activates the corresponding redundant actuator to maintain the continuous operation of the robots, minimizing any interruption in the task.

Trajectory executed by the robots to collect the objects

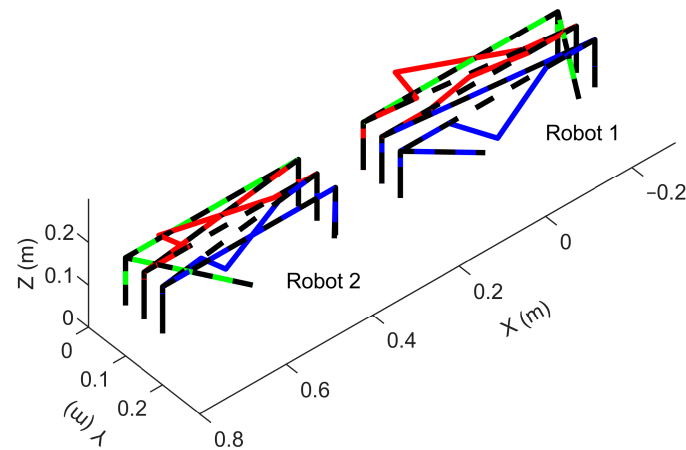


Figure 25. Desired and measured trajectories of the SCARA robots with multiple actuator failures.

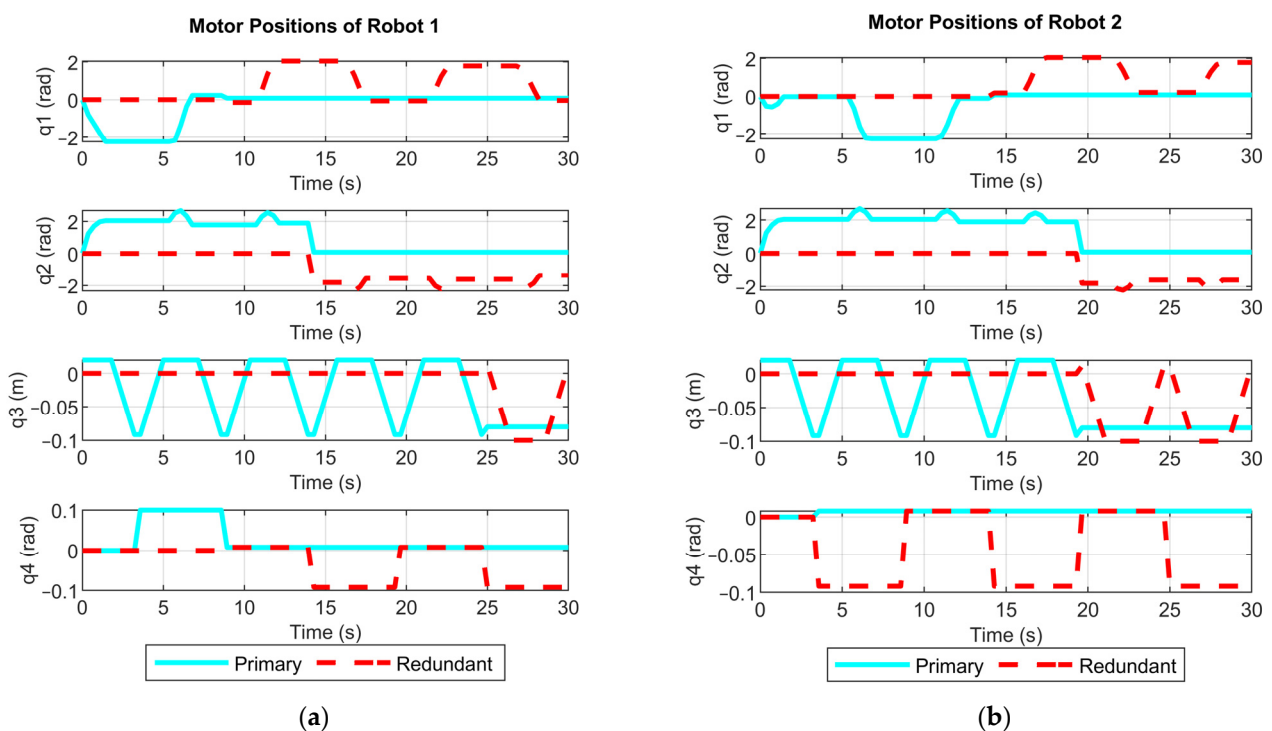


Figure 26. Joint positions of the robots with multiple failures: (a) Robot 1; (b) Robot 2.

Fifth case: A new stage is added to the cooperative task. Once the robots have finished collecting the objects of different colors, they must collaborate to transport the box containing these objects together. Table 6 presents the initial and final positions of the box, indicating where the robots must move it. Figure 27 shows the task in which the robots coordinate to pick up the box, lift it, and transport it to the final position. They then carefully place it down and release it before returning to their initial (home) position.

Table 6. Initial and final positions of the box to be transported by both robots.

Table			
	X (m)	Y (m)	Z (m)
Initial position	0.2	0.1	0.1
Final position	0.2	0.35	0.1

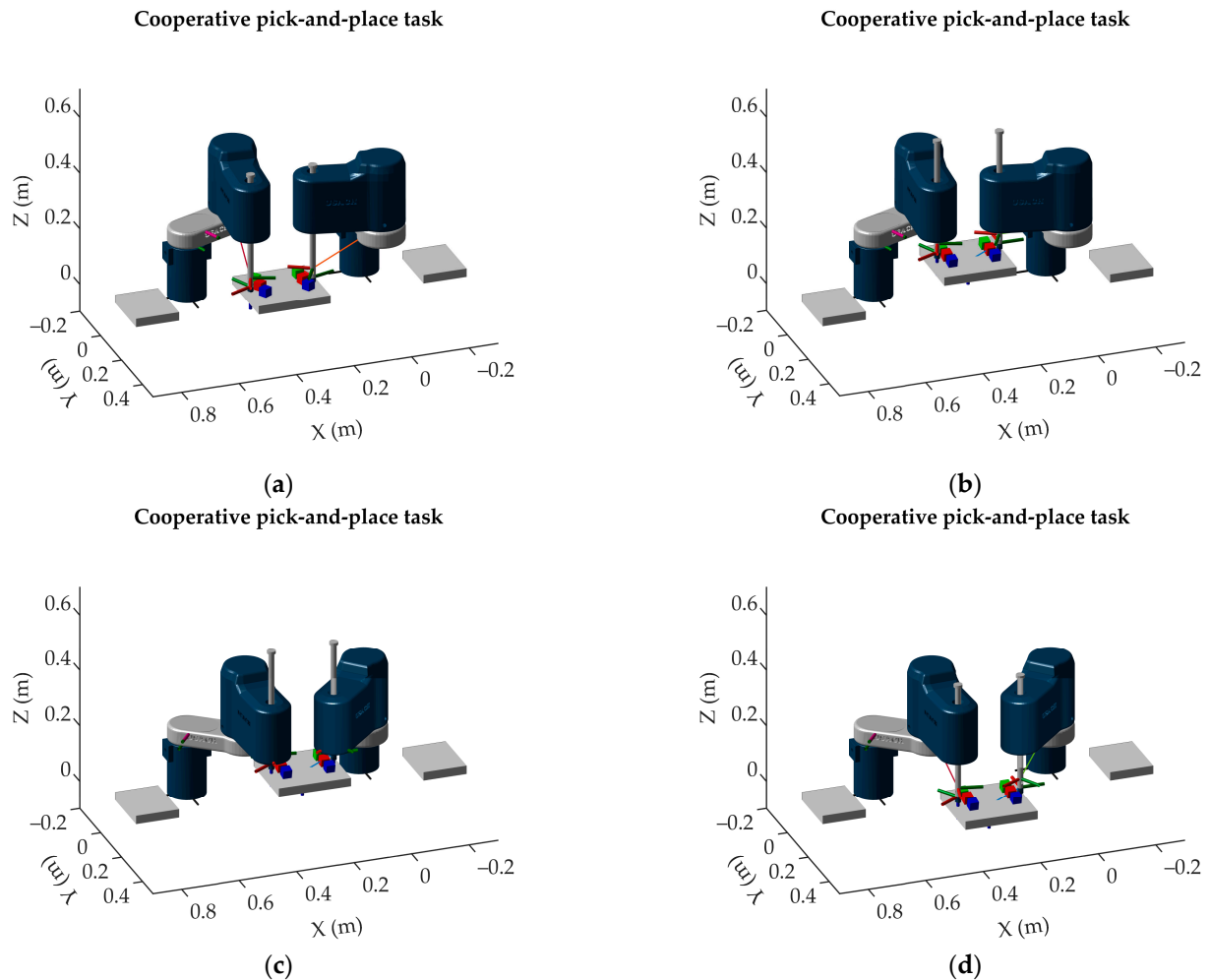
**Figure 27.** Cooperative process for transporting a table between both robots: (a) robots hold the table; (b) robots lift the table; (c) robots transport the table; (d) completion of the cooperative task.

Figure 28 shows the trajectories performed by the robots using the fault-tolerant system in combination with the NLMPC and ASMC position controllers under actuator failure conditions. The reference trajectory, corresponding to the case without actuator failures, is shown in red. Meanwhile, the trajectories obtained with the fault-tolerant system are illustrated in green for the NLMPC controller and blue for the ASMC controller.

Figure 29 shows the trajectory error on the different axes, comparing the deviations observed for each configuration.

In this study, the focus is specifically on the analysis of actuator failures, without considering other factors that may affect the proper functioning of the SCARA robotic system, such as external disturbances, vibrations, thermal changes, or sensor interference. While these conditions reflect a controlled simulation environment, in real industrial scenarios, external disturbances can also significantly influence robot performance. In future studies,

the incorporation of models that include these disturbances will be considered to evaluate the system's response more accurately under operating conditions closer to reality.

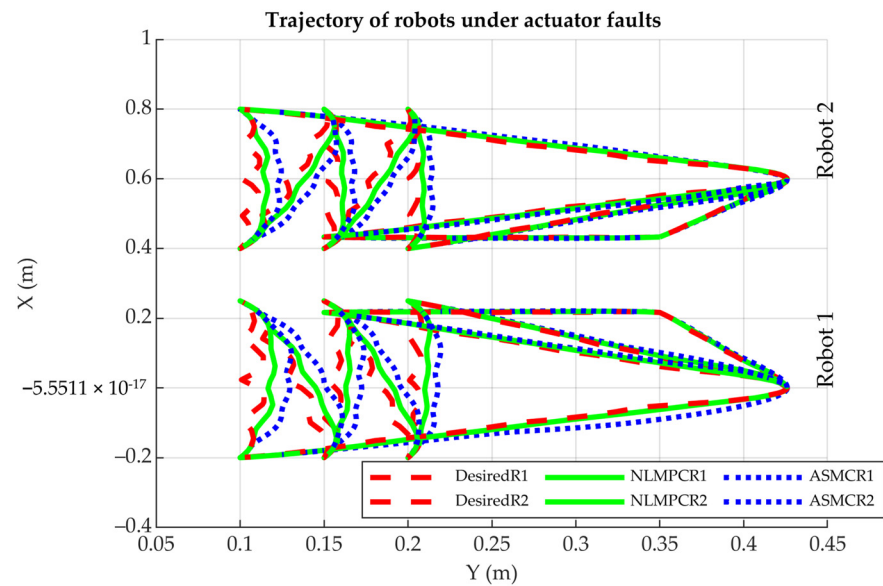
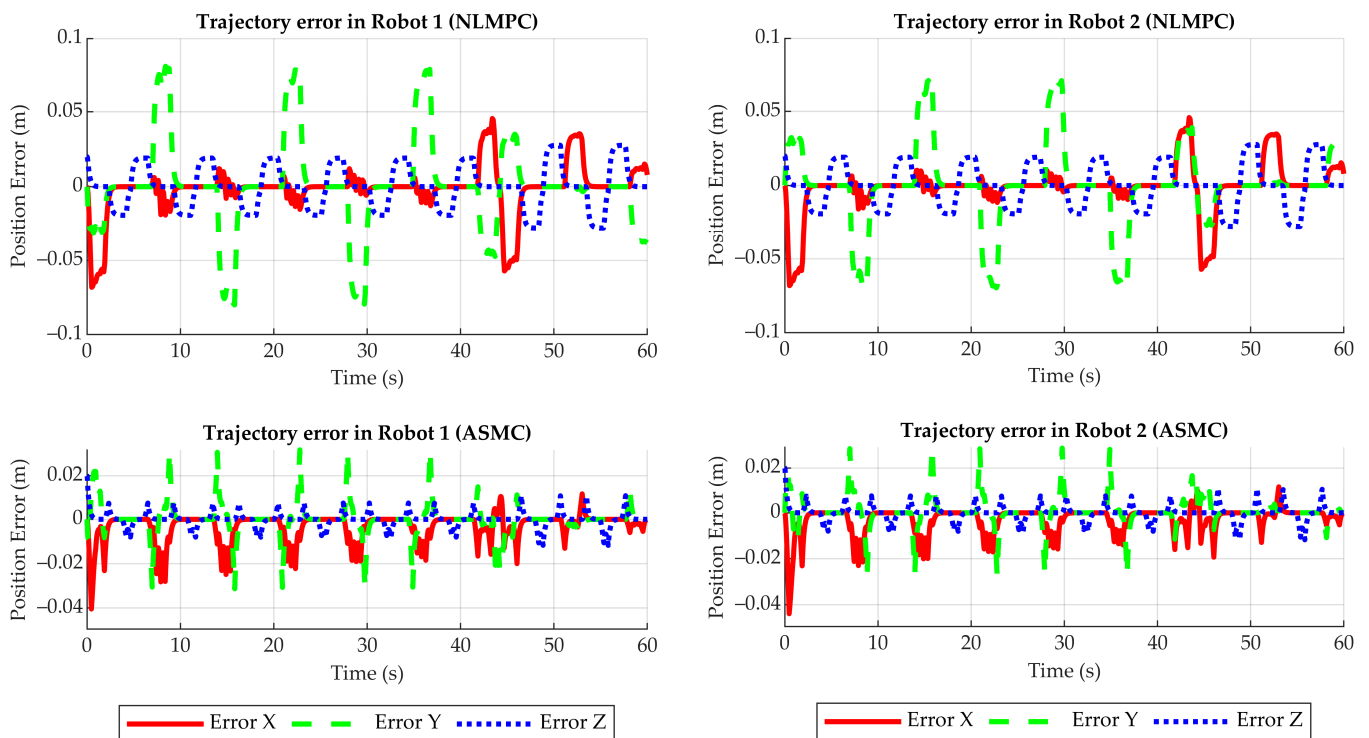


Figure 28. Trajectories performed by the robots in the presence of a failure in Actuator 1 of Robot 1 and Actuator 2 of Robot 2.



(a)

(b)

Figure 29. Position errors in the trajectories performed by the robots in the presence of failures in the primary Actuator 1 of Robot 1 and Actuator 2 of Robot 2: (a) Robot 1; (b) Robot 2.

5. Results Analysis

This section analyzes the results obtained from simulations conducted in the virtual environment developed in MATLAB R2024a. These simulations demonstrate the effec-

tiveness of the proposed fault-tolerant control system for SCARA robots, evaluated in five different scenarios:

Normal operation scenario: The robots successfully performed the tasks of collecting and transporting objects of different colors, maintaining high precision and synchronization. This confirms that the initial system design is suitable for cooperative operations without interruptions or collisions.

Scenario with actuator failures without reconfiguration: When total failures were introduced in the robots' actuators, their ability to position objects accurately was noticeably affected. Although the robots transported the objects, deviations in the trajectories caused significant errors in their final placement. This scenario validated the need for a fault-tolerant system, as active desynchronization due to failures jeopardizes the process.

Scenario with active reconfiguration in redundant actuators: When the fault-tolerant system was activated, the redundant actuators automatically took over the functions of the faulty primary actuators. Although there was a slight initial deviation in the robots' trajectories, the system responded efficiently within 0.5 s of the fault occurrence, adjusting positions and maintaining task precision. This result highlights the robustness of the proposed approach and its ability to maintain operational continuity without interference between the robots. Figure 30 shows the error produced when one of the actuators fails in Robot 1 and Robot 2.

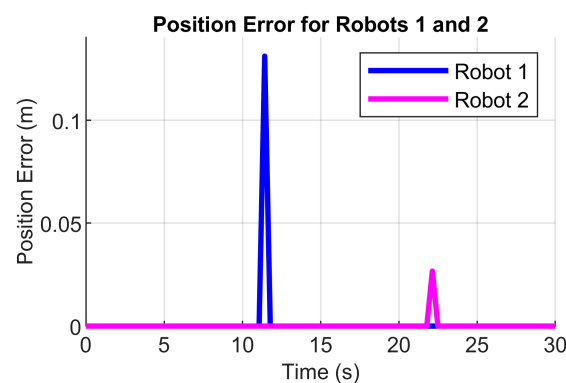


Figure 30. Trajectory errors when an actuator failure occurs in Robot 1 and Robot 2.

When analyzing the performance of Robot 1 and Robot 2 with the different controllers (NLMPC and SMC), some significant differences in terms of efficiency and precision are observed, Table 7. For Robot 1, the NLMPC controller clearly outperforms SMC in almost all aspects: the steady-state error e_{ss} is lower with NLMPC (0.0214 compared to 0.0348), suggesting that NLMPC maintains better final stability. Additionally, the Integral of Squared Error (ISE) is much lower with NLMPC (0.00612 compared to 0.0815), indicating that this controller achieves better overall performance in terms of precision.

Table 7. Initial and final positions of the box to be transported.

	Robot 1		Robot 2	
	NLMPC	SMC	NLMPC	SMC
e_{ss}	0.0214	0.0348	0.0014	0.0056
ISE	0.00612	0.0815	0.0002	0.00036
IAE	0.5948	0.65478	0.8741	1.0215
ITAE	4.2148	14.2147	8.2306	9.8745
IA	0.9574	1.0324	1.5474	1.8945

With Robot 2, the results follow a similar trend. NLMPC once again demonstrates superior performance, with a very low $e_{ss} = 0.0014$, compared to 0.0056 for SMC. This indicates that NLMPC is more effective at minimizing small errors and quickly adjusting to changes. Furthermore, the ISE for Robot 2 with NLMPC (0.0002) is significantly lower than that of SMC (0.00036), reinforcing the idea that NLMPC provides better overall control.

However, when observing other indicators such as the Integral of Absolute Error (IAE) and the Integral of Time-weighted Absolute Error (ITAE), the differences are not as pronounced or, in some cases, SMC even approaches NLMPC in terms of performance. For example, the ITAE for Robot 2 is higher with SMC (9.8745) than with NLMPC (8.2306), but the gap is not as significant as in the previous errors. The Index of Agreement (IA) also shows that SMC tends to require more control effort in both robots.

By inducing multiple actuator failures in both robots, the SCARA robotic system manages them effectively. Despite failures occurring at different times and during critical tasks, the system maintains operational continuity thanks to the rapid deactivation of defective actuators and the activation of redundant ones. The measured trajectories show slight deviations that do not significantly affect overall performance, validating the system's ability to adapt and minimize the impact of failures in real time.

Cooperative task of transporting a box with both robots: The robots collaborated to lift and transport a box with the collected objects. The simulation showed that even after reconfigurations due to failures, the system was able to coordinate the movements of both robots, completing the task without critical errors. This validates the mechanical design and control strategy, ensuring that joint operation is possible even under adverse conditions. When analyzing the errors observed during robot failures (see Figure 31), it was noted that Robot 1 with the NLMPC controller showed average errors of 0.0054 m on the X-axis, 0.0128 m on the Y-axis, and 0.0118 m on the Z-axis. These errors were reduced to 0.0042 m, 0.0040 m, and 0.0017 m, respectively, when using the ASMC controller.

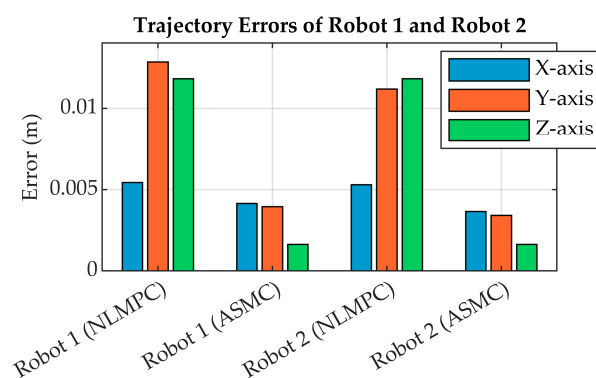


Figure 31. Bar chart of the trajectory errors of the robots on the X, Y, and Z axes using the fault-tolerant system and NLMPC and ASMC controllers.

Robot 2, on the other hand, showed average errors with the NLMPC controller of 0.0053 m on the X-axis, 0.0112 m on the Y-axis, and 0.0118 m on the Z-axis, which were reduced to 0.0037 m, 0.0034 m, and 0.0017 m when using the ASMC controller. These results demonstrate that the ASMC controller significantly improves performance in the presence of faults, reducing errors across all axes and ensuring accuracy in the desired trajectories.

When evaluating the efficiency in the cooperative task (see Figure 32), it is observed that the system achieved a remarkable success rate in all scenarios. For Robot 1 operating with the NLMPC controller, the efficiency was 99.00%, while with ASMC it reached 99.67%. Robot 2, meanwhile, showed an efficiency of 99.06% with NLMPC and 99.71% with ASMC.

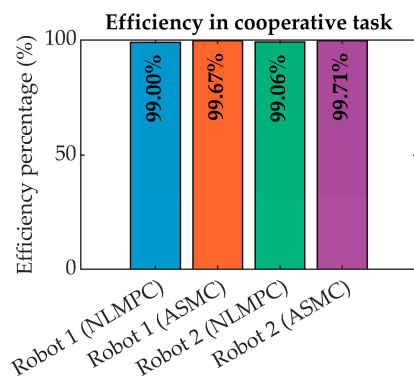


Figure 32. Bar chart of the percentage efficiency in the cooperative task for robots using NLMPC and ASMC controllers.

6. Conclusions and Future Perspectives

In conclusion, this study has validated that the fault-tolerant control system for SCARA robots with redundant actuators is an effective and robust solution for maintaining operational continuity and coordination in cooperative tasks, even in the event of critical failures in the primary actuators. Simulations conducted in a virtual environment developed in MATLAB R2024a confirmed that the system can detect faults, diagnose the problem, and activate redundant actuators within a reconfiguration time of 0.5 s, ensuring that tasks are executed correctly.

Regarding precision, the results showed minimal trajectory errors for the robots, with average values ranging between 0.0042 m and 0.0054 m, depending on the controller used. The ASMC controller stood out for its ability to reduce errors across all axes and ensure greater accuracy compared to NLMPC. Additionally, the overall system efficiency reached levels above 99%, demonstrating its capability to perform complex cooperative tasks, such as object collection and synchronized box transport, without interruptions or errors that could impact the task.

When multiple failures are applied, it is observed that although they temporarily affect movement accuracy, the system recovers quickly, allowing the robots to complete the assigned tasks without significant interruptions. The coordination between both robots is maintained, highlighting the system's ability to manage failures efficiently without compromising overall performance.

The integration of mechanical components, such as differential gears and torque limiters, along with advanced reconfiguration algorithms, minimized the impact of system failures, validating its applicability in high-demand industrial scenarios. Performance indicators such as ISE, IAE, and ITAE confirmed that the system maintains a high level of stability and control, optimizing overall performance. This makes it particularly valuable in sectors like advanced manufacturing, precision assembly, and logistics, where operational interruptions can lead to high costs and delays. Its optimized design reduces maintenance costs and improves sustainability, establishing it as a key tool for modern automation.

As future work, the incorporation of artificial intelligence and machine learning techniques is proposed to further enhance the system's response and adaptability to unexpected failures. The implementation of prediction algorithms based on neural networks could optimize early anomaly detection and enable proactive adjustments to control parameters. Additionally, integrating online learning models could facilitate continuous adaptation to changing conditions in industrial environments, increasing the system's overall efficiency and precision.

Author Contributions: Conceptualization, C.U. and P.S.; methodology, C.U. and P.S.; software, C.U. and P.S.; validation, C.U. and P.S.; formal analysis, C.U. and P.S.; investigation, C.U., P.S. and J.K.; resources, C.U., P.S. and J.K.; data curation, C.U., P.S. and J.K.; writing—original draft preparation, C.U. and P.S.; writing—review and editing, C.U. and P.S.; visualization, C.U. and J.K.; supervision, C.U. and J.K.; project administration, C.U. and J.K.; funding acquisition, C.U. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: This work has been supported by the Faculty of Engineering of the Universidad de Santiago de Chile, Chile.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations, listed in alphabetical order, are used in this manuscript:

AFTC	Active Fault Tolerant Control
ASMC	Adaptive Sliding Mode Control Adaptive Sliding Mode
FTC	Fault Tolerant Control
IA	Index of Agreement
IAE	Integral of Absolute Error
ISE	Integral of Squared Error
ITAE	Integral Time Absolute Error
IT2 T-S Fuzzy model	Interval Type-2 Takagi-Sugeno Fuzzy model
M-ANFIS	Multiple Adaptive Neuro-Fuzzy Inference System
NDFT	Dynamic Neural Fault-Tolerant Control
NLMPC	Nonlinear Model Predictive Controller
PFTC	Passive Fault-Tolerant Control
PID	Proportional Integral Derivative (Controller)
QP	Quadratic Programming
RBFNN	Radial Basis Function Neuronal Networks
SCARA	Selective Compliance Assembly Robot Arm
SMC	Sliding Mode Control
VP-RNN	Variable Parameter Recurrent Neural Networks

References

1. Peers, C.; Zhou, C. Bimanual Telemanipulation Framework Utilising Multiple Optically Localised Cooperative Mobile Manipulators. *Robotics* **2024**, *13*, 59. [\[CrossRef\]](#)
2. Țițu, A.M.; Gusan, V.; Dragomir, M.; Pop, A.B.; Țițu, Ș. Cost Calculation and Deployment Strategies for Collaborative Robots in Production Lines: An Innovative and Sustainable Perspective in Knowledge-Based Organizations. *Sustainability* **2024**, *16*, 5292. [\[CrossRef\]](#)
3. Sandini, G.; Sciutti, A.; Morasso, P. Collaborative Robots with Cognitive Capabilities for Industry 4.0 and Beyond. *AI* **2024**, *5*, 1858–1869. [\[CrossRef\]](#)
4. Samewoi, A.R.; Azlan, N.Z.; Khan, M.R. FAT-based adaptive and velocity feedback control of cooperative manipulators handling a flexible object. *Automatika* **2023**, *64*, 1010–1025. [\[CrossRef\]](#)
5. Sękala, A.; Blaszczyk, T.; Foit, K.; Kost, G. Selected Issues, Methods, and Trends in the Energy Consumption of Industrial Robots. *Energies* **2024**, *17*, 641. [\[CrossRef\]](#)
6. Wu, G.; Niu, B.; Li, Q. Trajectory Tracking Control of Fast Parallel SCARA Robots with Fuzzy Adaptive Iterative Learning Control for Repetitive Pick-and-Place Operations. *Electronics* **2023**, *12*, 4995. [\[CrossRef\]](#)
7. Vu, D.-T.; Nguyen, N.-K. A Comparative Study of Various Control Strategies for a 4-DOF Scara Robot. *Int. J. Electr. Electron. Eng.* **2024**, *11*, 175–183. [\[CrossRef\]](#)

8. Wang, S.; Wang, H.; Li, Y.; Li, Q. Distributed finite-time neuroadaptive fault-tolerant formation control for multi-robot systems. *Appl. Ocean Res.* **2024**, *150*, 104067. [CrossRef]
9. Yue, Q.; Ding, Y. Artificial Tendons Improve Fault-Tolerance of Robotic Arms Under Free-Swinging Failures. In Proceedings of the 2024 4th International Conference on Computer, Control and Robotics, ICCCR 2024, Shanghai, China, 19–21 April 2024; Institute of Electrical and Electronics Engineers Inc.: Piscataway Township, NJ, USA, 2024; pp. 274–283. [CrossRef]
10. Ghorbanpour, A. Cooperative Robot Manipulators Dynamical Modeling and Control: An Overview. *Dynamics* **2023**, *3*, 820–854. [CrossRef]
11. Liu, A.; Lai, G.; Xiao, H.; Liu, Z.; Zhang, Y.; Chen, C.P. Resilient adaptive trajectory tracking control for uncalibrated visual servoing systems with unknown actuator failures. *J. Frankl. Inst.* **2023**, *361*, 526–542. [CrossRef]
12. Maghami, A.; Imbert, A.; Côté, G.; Monsarrat, B.; Birglen, L.; Khoshdarregi, M. Calibration of Multi-Robot Cooperative Systems Using Deep Neural Networks. *J. Intell. Robot. Syst.* **2023**, *107*, 1–14. [CrossRef]
13. Pan, J.; Qu, L.; Peng, K. Fault-Tolerant Control of Multi-Joint Robot Based on Fractional-Order Sliding Mode. *Appl. Sci.* **2022**, *12*, 11908. [CrossRef]
14. AbuJabal, N.; Rabie, T.; Baziyad, M.; Kamel, I.; Almazrouei, K. Path Planning Techniques for Real-Time Multi-Robot Systems: A Systematic Review. *Electronics* **2024**, *13*, 2239. [CrossRef]
15. Kharrat, M.; Alhazmi, H. Finite-Time Adaptive Fuzzy Control for Stochastic Nonlinear Systems with Input Quantization and Actuator Faults. *J. Math.* **2024**, *2024*, 6778333. [CrossRef]
16. Lai, G.; Zhou, S.; Yang, W.; Wang, X.; Wang, F. Prescribed Fixed-Time Adaptive Neural Control for Manipulators with Uncertain Dynamics and Actuator Failures. *Mathematics* **2023**, *11*, 2925. [CrossRef]
17. Milecki, A.; Nowak, P. Review of Fault-Tolerant Control Systems Used in Robotic Manipulators. *Appl. Sci.* **2023**, *13*, 2675. [CrossRef]
18. Yang, L.; Yuan, C.; Lai, G. Adaptive Fault-Tolerant Visual Control of Robot Manipulators Using an Uncalibrated Camera. *Nonlinear Dyn.* **2022**, *111*, 3379–3392. [CrossRef]
19. Yuksel, T. Fault Tolerant Control of Robot Manipulators with Manfis and Gain Scheduling [Online]. Available online: <https://as-proceeding.com/index.php/icensos/article/view/467> (accessed on 29 November 2024).
20. Zhang, Z.; Cao, Z.; Li, X. Neural Dynamic Fault-Tolerant Scheme for Collaborative Motion Planning of Dual-Redundant Robot Manipulators. In *IEEE Transactions on Neural Networks and Learning Systems*; IEEE: Piscataway Township, NJ, USA, 2024. [CrossRef]
21. Fang, X.; Cheng, R.; Cheng, S.; Fan, Y. Nonsingular Fixed-time Fault-tolerant Sliding Mode Control of Robot Manipulator with Disturbance Observer. *Int. J. Control Autom. Syst.* **2024**, *22*, 2182–2192. [CrossRef]
22. Zhou, H.; Lam, H.-K.; Xiao, B.; Xuan, C. Integrated Fault-Tolerant Control Design with Sampled-Output Measurements for Interval Type-2 Takagi-Sugeno Fuzzy Systems. *IEEE Trans. Cybern.* **2024**, *54*, 5068–5077. [CrossRef]
23. Zhang, T.; Shi, P.; Li, W.; Yue, X. Adaptive EKF enhanced fault diagnosis and fault tolerant control for space manipulators with position measurements only. *J. Frankl. Inst.* **2024**, *361*, 106824. [CrossRef]
24. Alruwaily, Y.; Kharrat, M. Funnel-Based Adaptive Neural Fault-Tolerant Control for Nonlinear Systems with Dead-Zone and Actuator Faults: Application to Rigid Robot Manipulator and Inverted Pendulum Systems. *Complexity* **2024**, *2024*, 5344619. [CrossRef]
25. Xu, S. Disturbance Observer-Based Adaptive Fault Tolerant Control with Prescribed Performance of a Continuum Robot. *Actuators* **2024**, *13*, 267. [CrossRef]
26. Le, Q.D.; Kang, H.-J. Implementation of fault-tolerant control for a robot manipulator based on synchronous sliding mode control. *Appl. Sci.* **2020**, *10*, 2534. [CrossRef]
27. Li, J.; Peng, X.; Li, B.; Li, M.; Wu, J. Image-Based Visual Servoing for Three Degree-of-Freedom Robotic Arm with Actuator Faults. *Actuators* **2024**, *13*, 223. [CrossRef]
28. Tran, X.-T.; Nguyen, V.-C.; Le, P.-N.; Kang, H.-J. Observer-Based Fault-Tolerant Control for Uncertain Robot Manipulators without Velocity Measurements. *Actuators* **2024**, *13*, 207. [CrossRef]
29. Urrea, C.; Saa, D.; Kern, J. Automated Symbolic Processes for Dynamic Modeling of Redundant Manipulator Robots. *Processes* **2024**, *12*, 593. [CrossRef]
30. Urrea, C.; Kern, J. Modeling, simulation and control of a redundant SCARA-type manipulator robot. *Int. J. Adv. Robot. Syst.* **2012**, *9*, 51701. [CrossRef]
31. Cang, N.; Guo, D.; Zhang, W.; Shen, L.; Li, W. A new discrete-time repetitive motion planning scheme based on pseudoinverse formulation for redundant robot manipulators with joint constrains. *Robot. Auton. Syst.* **2024**, *176*, 104689. [CrossRef]
32. Li, Y.; Liu, H.; Gao, H. Online learning fuzzy echo state network with applications on redundant manipulators. *Front. Neurobotics* **2024**, *18*, 1431034. [CrossRef]
33. Yan, J.; Liu, M.; Jin, L. Cerebellum-Inspired Model Predictive Control for Redundant Manipulators with Unknown Structure Information. *IEEE Trans. Cogn. Dev. Syst.* **2023**, *16*, 1198–1210. [CrossRef]

34. Dai, J.; Zhang, Y.; Deng, H. Bidirectional RRT*-Based Path Planning for Tight Coordination of Dual Redundant Manipulators. *Machines* **2023**, *11*, 209. [CrossRef]
35. Xie, Z.; Jin, L. A Fuzzy Neural Controller for Model-Free Control of Redundant Manipulators with Unknown Kinematic Parameters. *IEEE Trans. Fuzzy Syst.* **2023**, *32*, 1589–1601. [CrossRef]
36. Jin, G.; Yu, X.; Chen, Y.; Li, J. SCARA+ System: Bin Picking System of Revolution-Symmetry Objects. *IEEE Trans. Ind. Electron.* **2024**, *71*, 10976–10986. [CrossRef]
37. Chen, M.; Zhang, B.; Li, H.; Gao, X.; Wang, J.; Zhang, J. Lifetime Prediction of Permanent Magnet Synchronous Motor in Selective Compliance Assembly Robot Arm Considering Insulation Thermal Aging. *Sensors* **2024**, *24*, 3747. [CrossRef]
38. Ashok, A.; Jain, C.; Relekar, R.J.; Rajput, R.; Venkatarangan, M.J. Test Tube Assortment Using SCARA Robot Platform. In Proceedings of the 2024 10th International Conference on Control, Automation and Robotics, ICCAR 2024, Singapore, 27–29 April 2024; Institute of Electrical and Electronics Engineers Inc.: Piscataway Township, NJ, USA, 2024; pp. 166–171. [CrossRef]
39. Chen, Q.; Chen, L.; Li, Q.; Shi, J.; Wang, D.; Shen, C. Metric Learning-Based Few-Shot Adversarial Domain Adaptation: A Cross-Machine Diagnosis Method for Ball Screws of Industrial Robots. *IEEE Trans. Instrum. Meas.* **2024**, *73*, 1–10. [CrossRef]
40. Giannoccaro, N.I.; Rausa, G.; Rizzi, R.; Visconti, P.; De Fazio, R. An Innovative Vision-Guided Feeding System for Robotic Picking of Different-Shaped Industrial Components Randomly Arranged. *Technologies* **2024**, *12*, 153. [CrossRef]
41. Zhang, Z.; Guo, Q.; Grigorev, M.A.; Kholodilin, I. Construction Method of a Digital-Twin Simulation System for SCARA Robots Based on Modular Communication. *Sensors* **2024**, *24*, 7183. [CrossRef]
42. Khorasani, A.; Usman, M.; Hubert, T.; Furnémont, R.; Lefeber, D.; Vanderborght, B.; Verstraten, T. Mitigating Collision Forces and Improving Response Performance in Human-Robot Interaction by Using Dual-Motor Actuators. *IEEE Robot. Autom. Lett.* **2024**, *9*, 5982–5989. [CrossRef]
43. Sun, L.; Gu, H. Backlash Elimination Control for Robotic Joints with Dual-Motor Drive. *Actuators* **2024**, *13*, 291. [CrossRef]
44. Sun, L.; Li, X.; Chen, L.; Shi, H.; Jiang, Z. Dual-Motor Coordination for High-Quality Servo with Transmission Backlash. *IEEE Trans. Ind. Electron.* **2022**, *70*, 1182–1196. [CrossRef]
45. Vanich, P.; Tangpornprasert, P.; Virulsri, C. Design of a Single-DoF Prosthetic Hand with Practical Maximum Grip Force and Grasp Speed for ADLs Using Dual-Motor Actuator. *IEEE Robot. Autom. Lett.* **2023**, *8*, 1439–1446. [CrossRef]
46. Lalonde, I.; Denis, J.; Lamy, M.; Girard, A. A Dual-Motor Actuator for Ceiling Robots with High Force and High Speed Capabilities. 2024. Available online: <http://arxiv.org/abs/2405.05162> (accessed on 30 November 2024).
47. Patra, K.; Sinha, A.; Guha, A. Online Capability Based Task Allocation of Cooperative Manipulators. *J. Intell. Robot. Syst.* **2024**, *110*, 23. [CrossRef]
48. Taner, B.; Subbarao, K. Modeling of Cooperative Robotic Systems and Predictive Control Applied to Biped Robots and UAV-UGV Docking with Task Prioritization. *Sensors* **2024**, *24*, 3189. [CrossRef]
49. Qian, L.; Liu, P.; Lu, H.; Shi, J.; Zhao, X. An End-to-End Inclination State Monitoring Method for Collaborative Robotic Drilling Based on Resnet Neural Network. *Sensors* **2024**, *24*, 1095. [CrossRef] [PubMed]
50. Urrea, C.; Sari, P.; Kern, J.; Torres, H. Enhancing Adaptability and Autonomy in Cooperative Selective Compliance Assembly Robot Arm Robots: Implementation of Coordination and Rapidly Exploring Random Tree Algorithms for Safe and Efficient Manipulation Tasks. *Appl. Sci.* **2024**, *14*, 6804. [CrossRef]
51. Ali, Z.A.; Alkhamash, E.H.; Hasan, R. State-of-the-Art Flocking Strategies for the Collective Motion of Multi-Robots. *Machines* **2024**, *12*, 739. [CrossRef]
52. Wang, D.; Lian, B.; Liu, Y.; Gao, B.; Zhang, S. Resilient Cooperative Localization Based on Factor Graphs for Multirobot Systems. *Remote Sens.* **2024**, *16*, 832. [CrossRef]
53. Peng, Z.; Wang, C. Reinforcement Learning-Based Pose Coordination Planning Capture Strategy for Space Non-Cooperative Targets. *Aerospace* **2024**, *11*, 706. [CrossRef]
54. Grosset, J.; Oukacha, O.; Fougères, A.-J.; Djoko-Kouam, M.; Bonnin, J.-M. Fuzzy Multi-Agent Simulation for Collective Energy Management of Autonomous Industrial Vehicle Fleets. *Algorithms* **2024**, *17*, 484. [CrossRef]
55. Yuan, Y.; Yang, P.; Jiang, H.; Shi, T. A Multi-Robot Task Allocation Method Based on the Synergy of the K-Means++ Algorithm and the Particle Swarm Algorithm. *Biomimetics* **2024**, *9*, 694. [CrossRef]
56. Matos, D.M.; Costa, P.; Sobreira, H.; Valente, A.; Lima, J. Efficient multi-robot path planning in real environments: A centralized coordination system. *Int. J. Intell. Robot. Appl.* **2024**, *8*, 1–28. [CrossRef]
57. Rodríguez-Seda, E.J.; Kutzer, M.D.M. Guaranteed real-time cooperative collision avoidance for n-DOF manipulators. *Robotica* **2024**, *42*, 3149–3173. [CrossRef]
58. Lefranc, G.; Lopez-Juarez, I.; Gatica, G. Enhancing FMS Performance through Multi-Agent Systems in the Context of Industry 4.0. *Stud. Inform. Control.* **2024**, *33*, 5–14. [CrossRef]

59. Geng, Z.; Yang, Z.; Xu, W.; Guo, W.; Sheng, X. A Globally Guided Dual-Arm Reactive Motion Controller for Coordinated Self-Handover in a Confined Domestic Environment. *Biomimetics* **2024**, *9*, 629. [[CrossRef](#)]
60. Chavdarov, I. Application of Barycentric Coordinates and the Jacobian Matrix to the Analysis of a Closed Structure Robot. *Robotics* **2024**, *13*, 152. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.