Article

Not peer-reviewed version

Federated Learning for Distributed Multi-Robotic Arm Trajectory Optimization

Fazal Khan * and Zhou Meng *

Posted Date: 20 August 2025

doi: 10.20944/preprints202508.1459.v1

Keywords: trajectory optimization; federated learning (FL); distributed multi-robotic arms; cooperative manipulation; real-time planning

Article

Federated Learning for Distributed Multi-Robotic Arm Trajectory Optimization

Fazal Khan * and Zhou Meng *

College of Mechanical Engineering, Donghua University, Songjiang, Shanghai, China
* Correspondence: meccol@dhu.edu.cn (F.K.); mz@dhu.edu.cn (Z.M.)

Abstract

Manipulator path and trajectory planning is a significant aspect of robotics that involves defining an optimal trajectory for the manipulator to move from a starting position to a target position while avoiding obstacles and minimizing latency factors such as time, energy, or jerk. This process typically involves algorithms that analyze the manipulator's kinematic and dynamic constraints, including workspace object geometry and environmental obstacles, to generate a collision-free and efficient trajectory. Common techniques include sampling-based methods like Rapidly-exploring Random Trees (RRT) or Probabilistic Roadmaps (PRM), optimization-based approaches, and artificial potential fields that treat obstacles as repulsive forces and goals as attractive forces. Advanced path planning may also incorporate machine learning for adaptive navigation in dynamic environments. The goal is to ensure smooth, precise, and safe motion. Efficient path planning enhances performance, reduces wear and tear, and ensures operational reliability.

Keywords: trajectory optimization; federated learning (FL); distributed multi-robotic arms; cooperative manipulation; real-time planning

---

1. Introduction

1.1. Background

Robotic manipulators have become indispensable in modern industrial automation, performing complex tasks such as assembly, welding, and material handling with high precision. However, as industries increasingly adopt multi-robotic systems for collaborative operations, optimizing their trajectories in shared workspaces presents significant challenges.

1.2. Significance of This Problem

A critical limitation in multi-robotic systems is the need for centralized control, which often requires sharing raw sensor data among manipulators, raising concerns about data privacy and communication overhead. Federated Learning (FL) offers a promising solution by enabling decentralized model training, where manipulators collaboratively optimize trajectories without exchanging raw data[1]. Instead, local updates from reinforcement learning (RL) agents are aggregated into a global model, preserving privacy while improving adaptability[2].

1.3. Adopted Methods

Traditional trajectory planning methods, such as Rapidly-exploring Random Trees (RRT) and Probabilistic Roadmaps (PRM), focus on single-robot systems and often struggle with dynamic environments, real-time coordination, and computational efficiency when scaled to multiple robots[3]. Moreover, ensuring collision-free motion while maintaining synchronization and minimizing latency factors remains an open research problem.

1.4. *Shortcomings of Existing Methods*

However, implementing FL in multi-robotic trajectory optimization introduces challenges such as handling heterogeneous manipulator configurations, maintaining synchronization in dynamic environments, and balancing communication efficiency with real-time performance.

1.5. *Optimization Contribution*

This paper proposes a Federated Learning-based framework for distributed multi-robotic arm trajectory optimization, addressing these challenges through a hybrid approach combining FL with sampling-based motion planning. Our contributions include:

- A privacy-preserving FL architecture where robots train shared trajectory models locally and contribute only gradient updates, eliminating the need for raw data exchange.
- An adaptive RRT algorithm enhanced with pruning optimization to reduce computational overhead while ensuring collision-free paths in dynamic environments.
- Real-time synchronization via EtherCAT, ensuring precise coordination among manipulators with minimal latency[4].
- Integration of multi-sensor fusion such as vision, encoders, force feedback for robust object localization and trajectory adaptation.

Our experiments demonstrate that the proposed framework achieves superior trajectory efficiency, collision avoidance, and scalability compared to centralized and standalone RL approaches. Applications span industrial assembly lines, surgical robotics, and warehouse automation, where collaborative manipulation in constrained spaces is critical. The simulations have undergone significant revisions to improve clarity, technical depth, and organization. The algorithms were restructured to concisely present the core goal of optimizing multi-robotic arm trajectories using federated learning (FL) while preserving data privacy, along with key challenges like synchronization in dynamic environments and handling heterogeneous robot configurations. The introduction was substantially expanded to provide stronger motivation for the work, clearly outlining the limitations of existing centralized approaches and explaining how FL enables collaborative learning without raw data exchange[5]. A dedicated contributions subsection was added to highlight the paper's novel elements, including the FL-based optimization framework, adaptive RRT algorithm with pruning optimization, and real-time synchronization through EtherCAT.

Technical sections were enhanced with deeper explanations of the federated learning architecture, particularly how local reinforcement learning updates are securely aggregated into a global model[6,7]. The trajectory optimization methodology now includes more detailed discussions of sampling-based techniques like RRT and PRM, with emphasis on the improved adaptive RRT algorithm that incorporates dynamic pruning to reduce computational overhead.

2. Synchronization of Multiple Manipulators

Synchronization of multiple manipulators is challenging task in bin picking. a perfect synchronization can assure accuracy. It's quite difficult but precise coordination can enhance the working speed and work ability. This process involves a central controller to control and manage manipulator simultaneously through Ethercat [8–10]. Where precision also required to make sure exact positioning and multiple tasks can perform at same time by distributing approaches where individual controller can communicate in real time by Ethercat.

Adding encoder for the feedback to controller to assure the close loop communication. Multiple manipulators connected together symmetrically and performing action on same time required path planning and strong RRT algorithm. Central unit is responsible for decision making and precise control to avoid collision. When its shared space It also required the communication of multiple manipulators at the same time and advanced algorithm compute synchronization of RRT trajectories.

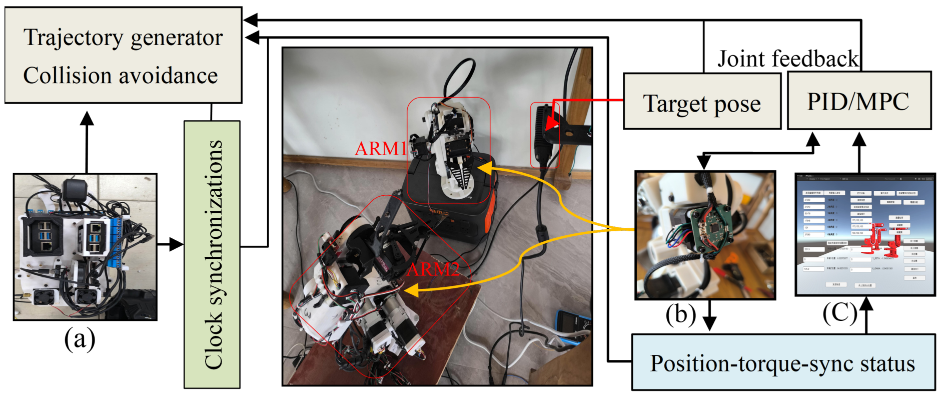

Figure 1. CoordinatedControl Synchronizing Multiple Manipulators Using (a) Controller (PID, MPC), (b) Feedback Controller for Stability, and (c) Visualization for Real-Time Monitoring. The system integrates (a) controllers (PID for precision, MPC for predictive optimization), (b) feedback controllers to correct deviations via encoder data, and (c) visualization tools RRT path plots, state trajectories to enhance operational transparency and debugging.

$$t_{\text{slave}_i} = t_{\text{master}} + \Delta t_{\text{offset}_i} + \Delta t_{\text{propagation}_i} \tag{1}$$

The equation describes how EtherCAT synchronizes the local clock of the i-slave device with the master's reference clock. Here t_{slave} represents the synchronized time of the slave, while t_{master} is the master's reference time. The term Δt_{offset} corrects any initial time difference offset between the slave and master clocks, ensuring they start from a common baseline. Meanwhile $\Delta t_{\text{propagation}}$ accounts for the signal transmission delay as data travels from the master to the slave over the network.

2.1. Non-Repetitive Trajectories

Because the object location and size is different so need more calculations work to pick and place. When they must follow non-repeatable trajectory its more complex trajectory path, each manipulator required different sequence to perform task. Unlike repeatable trajectories non-repeatable trajectories required continues recalculations to perform collision free task[11]. Since the trajectory is not predefined task and motion planning RRT algorithm helps the control system to automatically choose minimum path to perform task.

$$x(t) = \sum_{k=0}^{n} a_k(t) \cdot t^k \tag{2}$$

The expression $x(t)$ defines a vector-valued function as a polynomial in ¢ where each term $a_k(t) \cdot t^k$ consists of a time-dependent coefficient vector $a_k(t)$ multiplied by t^k. This allows $x(t)$ to model dynamic, time-varying behaviors, such as trajectories or curves, where the coefficients themselves can change over time for greater flexibility.

$$\begin{aligned} x(t) &= a_1 t + a_2 t^2 + a_3 \sin(\omega t), \\ y(t) &= b_1 t + b_2 \cos(\omega t) + b_3 t^3, \\ z(t) &= c_1 \sqrt{t} + c_2 e^{-\lambda t} + c_3 \ln(1+t). \end{aligned} \tag{3}$$

The equations describe the motion of a point in 3D space over time ¢ Combines linear motion $a_1 t$ and oscillation $a_3 \sin(\omega t)$ Includes linear motion $b_1 t$, oscillation $b_2 \cos(\omega t)$ and cubic growth $b_3 t^3$ Features a square root term $c_1 \sqrt{t}$, exponential decay $c_2 e^{-\lambda t}$ and logarithmic growth $c_3 \ln(1+t)$. Together, they model complex motion with linear, oscillatory, and non-linear effects[12,13].

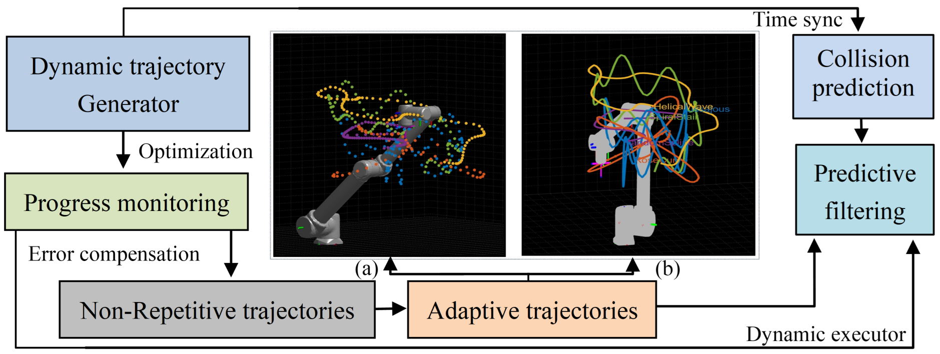

Figure 2. Real-Time Optimization, Error Compensation, and Collision Prediction for Motion Systems—combining (a) intermittent/periodic control (irregular, cycle-skipping) for efficiency and (b) continuous control (smooth, unbroken cycles) for stability.

2.2. Repeatable Trajectories

Probabilistic Road map: Probabilistic Road map is referred to network of dots within occupied environment. This technique is used by manipulator where algorithm construct a graph in robotic configuration space. It consists two phases learning phase and query phase.

Learning phase: learning face creates random nodes in graph for collision free movement Query phase: this phase creates nearby paths for robot by connecting closest dots to each other These techniques will provide initial information about starting point to final point. As PRM relies on these nodes created randomly by algorithm and then connected together. Algorithm relies on sampling-based motion planner to constructing graph and connected random nodes. this can also include the analyzing obstacle in environment. Apart from usable space there is hurdles and obstacles in path. multiple planner Nodes connecting can be specified according to roots[14].

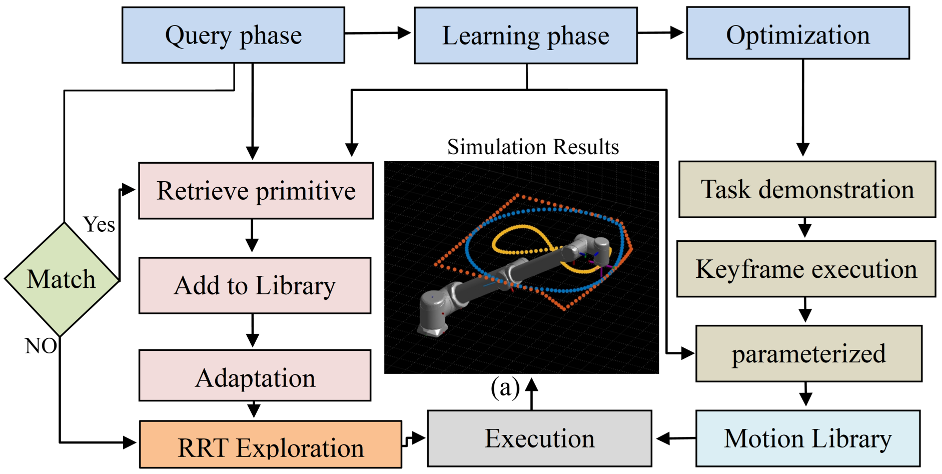

Figure 3. Probabilistic Roadmaps (PRM) and Rapidly Exploring (RRT) for Repeatable Path Planning.(a)represents a unique one-time path without repetition.

Once path building complete and the graph dots been created next process continues and can connect multiple dots together and same planner can use multiple times. This can save efforts and time. Rapidly Exploring Random trees: this technique is one of time query planner technique that use sampling-based motion planning algorithm to explore high dimensional spaces. RRT advancements a tree from starting configuration when graph been created start to end make it particularly effectively, RRT make it effective for real dynamic world. sometimes its bidirectional and if required to reuse need to make query plan again, if two different graphs created and at some point, they connect together.

$$q_{\text{new}} = q_{\text{near}} + \eta \cdot \frac{q_{\text{rand}} - q_{\text{near}}}{\|q_{\text{rand}} - q_{\text{near}}\|},$$
$$\text{where} \begin{cases} q_{\text{rand}} \sim \text{RandomSample}(0), & \text{(random config)} \\ q_{\text{near}} = \arg\min_{q \in T} \|q - q_{\text{rand}}\|, & \text{(nearest node)} \\ \text{CollisionFree}(q_{\text{near}}, q_{\text{new}}), & \text{(feasibility)} \\ \|q_{\text{new}} - q_{\text{goal}}\| < \epsilon \Rightarrow \text{terminate.} & \text{(goal check)} \end{cases} \tag{4}$$

The RRT algorithm expands a tree by generating new nodes q_{new} from the nearest existing node q_{near} moving a step size η toward a random sample q_{rand}. Key conditions q_{rand} is randomly sampled q_{near} is the tree's closest node, the path must be collision-free, and the process stops when q_{new} nears the goal within tolerance ϵ. This balances exploration and goal-directed growth for efficient path finding.

3. Actuation and Redundancy

Hardware very much relies on actuation where it converts energy to motions and often exploits to redundancy means having more degree of freedom that actually required this enhance flexibility and robustness robotic system being used has 2-4 robotic arm with 6DOF each.

Fully actuated: within working range within fully actuated condition manipulator all joints are fully actuated and has independently control over all the degree of freedom DOF in working process. Under actuated: in this condition robot don't have degree of freedom instead of redundancy achieve by dynamic coupling fewer actuator then degree of freedom. conditions also apply.

For collision-free movement encoders and stop switches is used to stop over moving the manipulator. By this way end effector can be positioned to maximum reachable points[19]. In shared-space each manipulator moving in different position there is high chance to collide or hit with obstacles, system required high end safety to avoid collision. To avoid collision feedback required by master control to access full control to each manipulator.

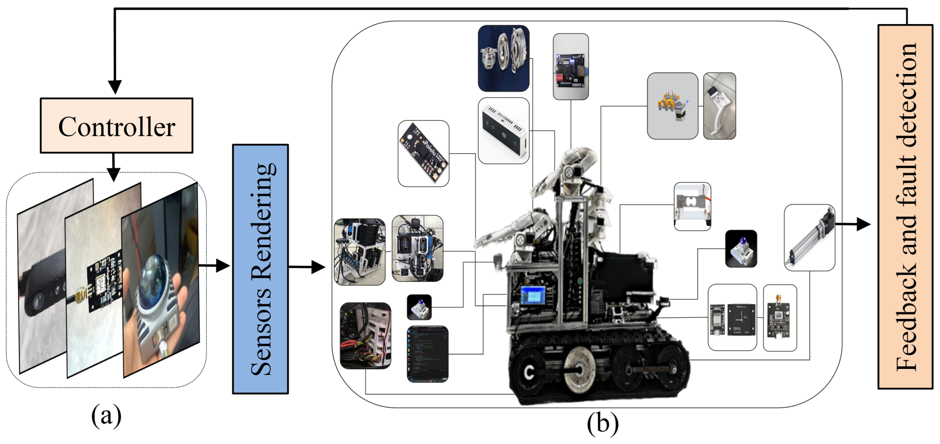

Figure 4. Actuation and Redundancy in Robotic Systems Enhancing Performance and Control Strategies.(a)heterogeneous sensor fusion setup LiDAR/IMU/RGB-D.(b)Multi-sensor fusion system integrating LiDAR, IMU, and RGB-D data for unified perception and reporting..

$$\begin{aligned} M(q)\dot{q} + C(q,\dot{q})\dot{q} + G(q) &= \tau \\ M(q)\dot{q} + C(q,\dot{q})\dot{q} + G(q) &= B\tau \end{aligned} \tag{5}$$

The equations represent the dynamics of robotic manipulators, capturing the relationship between joint motion and applied torques. The first equation describes a fully actuated robotic arm, where the inertial forces $M(q)\dot{q}$, Coriolis/centripetal forces $C(q,\dot{q})\dot{q}$, and gravitational forces $G(q)$ are balanced by direct joint torques τ. Here $M(q)$ is the inertia matrix encoding mass distribution, $C(q,\dot{q})$ accounts for velocity-dependent effects, and $G(q)$ represents gravity's influence. Underactuated redundancy generalizes this to an underactuated system via the actuation matrix $B\tau$ which maps available control inputs τ to effective joint torques $B\tau$.

3.1. Manipulator Rendering

Dynamic environment is rendered with different sizes and shapes of geometries such as boxes or cylindrical shapes, process start with importing manipulator kinematics system use a technique called frugal that is provide initial information about the shape and size of geometry [15,16]. System sends signals and calculate the pose of arm to pick the geometry and perform next task. By using the Kalman filter State equation

$$x_k = F_k x_{k-1} + B_k u_k + w_k \tag{6}$$

The equation describes a discrete-time state-space model that describes how a system evolves from one time step $k-1$ to the next k. Here x_k represents the current state vector (position, velocity) which depends on system Dynamics $F_k x_{k-1}$. The state transition matrix F_k linearly transforms the previous state x_{k-1} to predict the new state. Control Input $B_k u_k$. The matrix B_k maps external control actions u_k (motor commands) to their effect on the state. Process Noise w_k. This term accounts for uncertainties or disturbances (assumed Gaussian, with zero mean and covariance) modeling unpredictable effects like friction or sensor errors. Measuring equation observation model:

$$z_k = H_k x_k + v_k \tag{7}$$

The equation describes the measurement model in state estimation and filtering problems (Kalman filters). z_k: The measurement vector at time step k representing observed data (sensor readings like position, velocity, or camera detections). H_k: The observation matrix, which maps the true state to the expected measurement. It defines how the state variables relate to the sensor outputs. x_k: The true state vector (manipulator poses, object position) being estimated. v_k: The measurement noise, accounting for sensor inaccuracies. $w_k \sim N(0, Q_k)$ = process noise (Gaussian with covariance Q_k

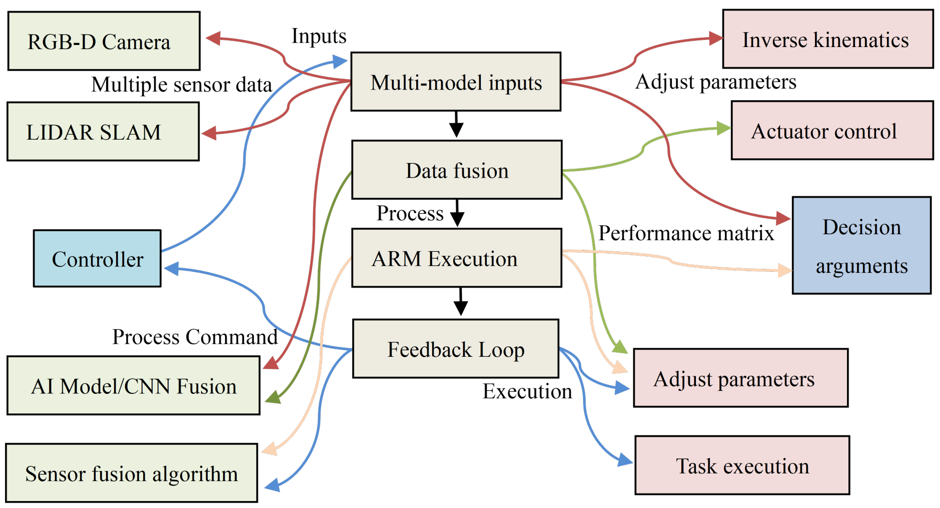

Figure 5. Multi-modal Initialization and Multi-Sensor Fusion Architecture for Adaptive Trajectory Diversity.

3.2. Multi-Modal Initialization

Control system align information from different sensors fusion and gathering data to a achieve a robust estimation shape of object size. Vision system provide orientation of object input can be given by pictures from different sizes after that system can analyze size of object by self and adjust gripper according to shape of object. additionally, pressure sensors also added to gripper for smooth gripping.

$$Q^* = \begin{cases} J(Q_k) + \lambda \cdot 1_{\text{collision}}(Q_k, \epsilon) \\ Q_k = I(q_{k0}, q_{kT}, t) \\ q_{k0}, q_{kT} \sim p(q|M) \end{cases} \tag{8}$$

This equation defines multi-modal trajectory optimization for robotic arms. The optimal trajectory Q^* is chosen from candidates Q_k each generated by interpolating between sampled start q_{k0} and goal q_{kT} configurations. The selection minimizes a cost $J(Q_k)$ (path length) plus a collision penalty $\lambda \cdot 1_{\text{collision}}$. This ensures safe, efficient motion in cluttered environments by evaluating multiple path options

- Trajectory generation: $Q_k = I(q_{k0}, q_{kT}, t)$
- Configuration sampling: $q_{k0}, q_{kT} \sim p(q|M)$
- $J(Q_k) + \lambda \cdot 1_{\text{collision}}$: Cost + safety trade-off

where λ is applied to penalize trajectories that collide with obstacles in the environment ϵ, and λ is a weighting factor. The control system integrates data from multiple sensors, fusing inputs from vision systems, LiDAR, and pressure sensors to create a robust estimation of an object's size and shape.

3.3. Collision-Free Motion Planning

Multiple manipulators connected practically providing continuously feedback to master control to get exact positioning to improve coordination in complex dynamic environment, visual information can be process individually by each system and trajectory path can be determine by connectivity of multiple system[17].

3.4. Constrained Optimization for Each Manipulator

By using multiple constraints such as environmental constraints, these constraints execute complex task while staying in manipulators physical limits. task constraints and coordination constraints can get possible results by formulate motion planning and mathematically optimization of trajectories with explicit constraints. As manipulator perform different task by using diffident path trajectories.

4. Motion Variation

When multiple-manipulator work in shared space Object manipulation required shortest time as well as required collision free time to complete task in possible shortest time, where placing position is predefined but picking place is not predefined, system is required to recognized connection between each manipulator to perform task, adding encoder helps the manipulators for exact positioning and system can make a trajectory planning by collaboration of each manipulator, each manipulator control system is synchronized together and connected to master control.

Operating manipulators in share space required carefully synchronize in dynamic environment. create the path trajectories for separately to each manipulator this condition applies for multiple-manipulator at the same space and each manipulator required to work closely.

$$\begin{aligned} &\tau_i(0) = q_{i,\text{start}}, \quad \tau_i(T) = q_{i,\text{goal}} \\ &\forall i, \quad A_i(q_i(t)) \cap \left(\bigcup_{j \neq i} A_j(q_j(t)) \right) = \emptyset \quad \forall t, i \end{aligned} \tag{9}$$

Equations describe the constraints for a multiple manipulators path planning problem. specifies the boundary conditions for each agent i: the initial position $\tau_i(0)$ is set to the starting configuration $q_{i,\text{start}}$ and the final position $\tau_i(T)$ must reach the goal configuration $q_{i,\text{goal}}$ by time T. This ensures that each agent begins and ends its trajectory at predefined locations.

For every agent i, the area or volume $A_i(q_i(t))$ occupied by the agent at any time t must not intersect with the combined areas or volumes occupied by all other agents $j \neq i$ at the same time. In other words, no two agents can occupy the same space simultaneously, ensuring safe and conflict-free paths.

5. System Structure and Updates in Design

Modern manipulator considering several factors sophisticated hierarchical new architecture design been added for both performance and adaptability. Consisting three layers: Physical layer: physical layer with modular interfaces to unified control connect such as Ethercat Coordination layer: coordination layer is responsible for handling real-time task distributions GPU-accelerated trajectory planning and collision-free motion planning. Cognitive layer: its human-robot interaction through machine learning models. with manipulator to enhance work efficiency Adopting a sudden new environment and making new trajectories is challenging task for control system. Essential details can be provided by sensor fusion and strategy based real time decision can be solve the removing error in trajectories.

5.1. Action-Space

Action space is defined by robot structure which allows each arm to work under working space and distance from object to gripper. Data integration can be best strategy to avoid collision between multiple-manipulators, this technique makes the pick and place process different in trajectories as well as in time so it will work one by one. Gripper designing varying size of object, opening/closing time can be adjusted by command. And position and pointing can be adjust according to position of object.

5.2. Motion Planning Failure

Motion planning failure occur if RRT algorithm fail to generate viable path for motion. different path trajectory required different path trajectory planning at the meantime some RRT trajectories can recover from prediction of previously used trajectories in real time. training data can be analyzing multiple time to determine whether mission can be successful or not.in real world environment scenario mission failure cannot be predicted in advanced yet to be perform task and analyze previous trajectories. this data can be used in variation in training data.

5.3. Rapidly-Exploring Random Tree

RPT expansion is use in the parabolicity compute sampling-based RRT method. It builds a search tree from a static beginning point and then gradually expands it using random sampling nodes to explore the whole environmental space until it reaches a target point. A continuous trajectory route between the starting and destination point is created by this technique. Adding the probabilistic computing and quick search capabilities of the RRT method are its main advantages. The term "completeness" describes the algorithm's thorough coverage of the search space and connecting nodes through a process, where it allows for path planning between any two places by conducting an exhaustive and unhindered study of the full map region given enough time and iterations. Previous-recent sampling point, New Node, grows with radius r under collision detection circumstances. cost sums are computed sequentially from the starting coordinate point to the closest node and to the new node and every node in this dynamic environmental space is a subset of the new node. The new node's parent is selected to be the node with the closest distance. if there are no barriers separating it from the closest node the new node is added to the tree structure. this iterative process keeps on until criteria are satisfied and workable solution is discovered.

5.4. Adaptive Pruning Optimization Strategy

With the addition of an adaptive pruning threshold and eliminate suboptimal nodes from motion. enhanced RRT algorithm integrates an adaptive pruning optimization technique that intelligently maintain computational efficiency without decreasing solution quality. Depending on real conditions the algorithm dynamically modifies this threshold encountered throughout the search process rather of relying just on the cost of the current best path to set pruning criteria[18]. In order to dynamically modify the lopping threshold, it also takes into account the features of obstacles avoidance in diverse contexts, including the numbers of obstacles the path's complexity, and varied dimensions lowering the pruning threshold.

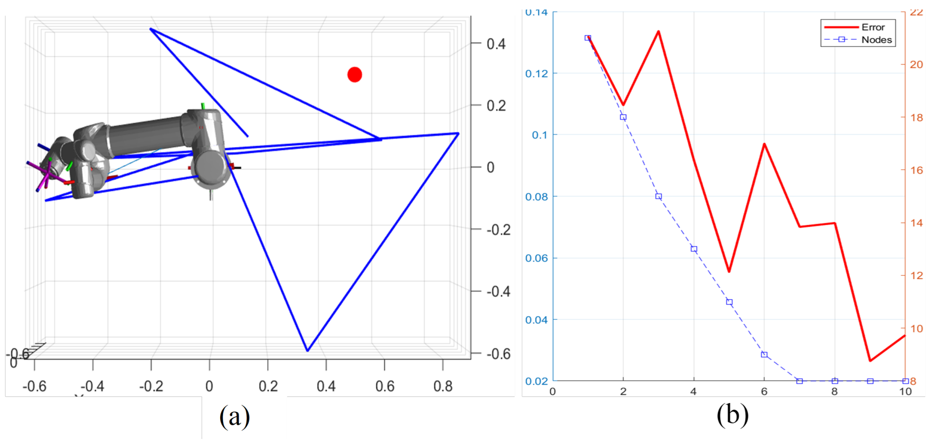

Figure 6. Adaptive Pruning Metrics Error Reduction and Node Elimination.(a)representing data points or measurements in a process iteration count 10, an error value 0.035 rad.(b)error decreasing from 0.14 rad to 0.02 rad over 10 iterations, indicating improving system performance.

The optimization method in algorithm also incorporates a local path re-evaluation in mechanism. the optimization method can efficiently narrow the search space and By directly altering the tree structure boost the algorithm's performance even further. The ongoing pruning and path optimization process can add enhanced RRT method can reduce the path length and increase path trajectory smoothness.

$$\min\left|\int\left[\sum_i\left(e_i^2 + A_i w_i^2 + \gamma a_i^2\right)\right]dt\right| \tag{10}$$

This equation optimizes robotic arm movements by minimizing tracking errors $\|e(t)\|_P$, control effort $\|u(t)\|_W$, and sudden joint changes $\|\dot{q}(t)\|_Q$ over time. It ensures precise, energy-efficient, and smooth pruning motions. The integral balances these factors across the task duration, while missing constraints (like arm dynamics) would refine the system further.

Table 1. Iteration Results Nodes, Error (rad), and Error Reduction

Iteration	Nodes	Error (rad)	Error Reduction
1	25	0.1700	—
2	18	0.1647	3.1%
3	15	0.1382	16.1%
4	13	0.0851	38.4%
5	11	0.0791	7.0%
6	9	0.0937	18.5%
7	8	0.0332	64.6%
8	8	0.0659	98.5%
9	8	0.0380	42.3%
10	8	0.0656	72.6%

in order to preserve and maintain more possible route possibilities could be required to delay in pruning places with a lot of impediments to determine whether to keep or change the pathways in the current search tree.

6. Connectivity

Connecting refers to multiple-manipulator collision-free trajectory in controlled dynamic environment.it required multiple data for trajectories and path planning. Multi-layer trajectories required multiple RRT algorithms at back end, optimizing global planner and localization which is individually planner for each manipulator. connected multiple system to master control at the same time and continuous feedback to create collisions-free dynamic environment. A manipulator control system can connect to other system by physical interfaces which provided by system in use.

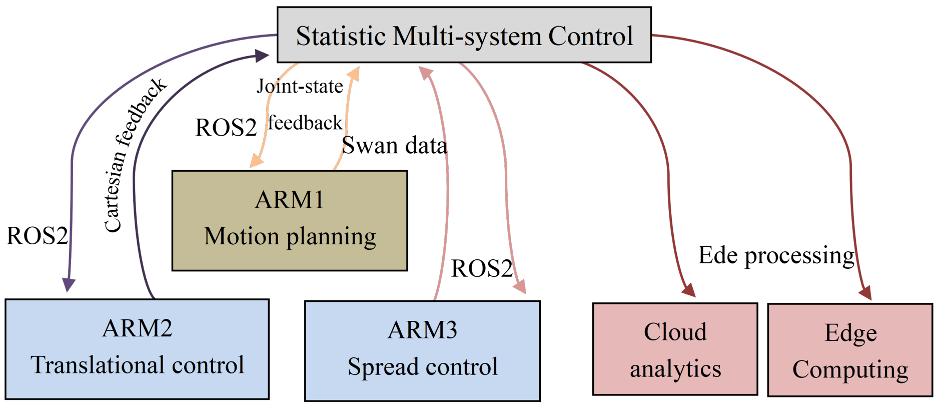

Figure 7. Multi-Agent Robotic Coordination Real-Time Edge Processing & Cloud Analytics.

6.1. Motion-Planning in Joint Space

In shared space manipulator-system can connect to another manipulator-system by physical interface. trajectory planning and motion constraints in 3D visualization helps to avoid obstacle and also beneficial for system to locate exact position of object and update trajectory after every step. Path trajectory in a working space can divide according to available working space. RRT algorithm and motion constraints assist the motion planner to reduce complexity.

6.2. System-to-System Communication

Multiple system manipulation is challenging task for control system in shared working space. Mathematical algorithm for motion planning employs layered communication architecture for communication and self-decision making enables possibilities to communicate multiple manipulator-system with each other. Communication assist in coordination for planning, control and execution. Connected system with master control, system exchange related information with other system which can be processed and analyze or present and previous position of manipulator collectively information exchange creates close-loop communication between one system to another by sending request information and exchanging information. Master control-System inquiry about recent activities to other systems.

6.3. Statistic Multi System Control

This is a modern statistical technique that centralized control system, it is involving the global command to control each manipulator connected with master control. however, for clarification this refer to multi-control system, that provide a central command structure that can exceed to several numbers of control-system connected with each other by global command[19]. Control depends on communication range within range many systems can connect with each other.

6.4. Computational Limits

Manipulator motion planning confront fundamental computational limits that need to carefully scale with manipulator degree of freedom When numbers of systems connected with each other in dynamic control, computational limit become bottleneck for system to process to next step. System required complete structure which dot to dot connected with each other, path trajectory and planning will be analyzed individually but each system connected to master-control to get equally feedback to each manipulator. Adding Self decision-making possibilities required data provided by integration of sensors fusion data from each manipulator, localization required central control to be in action.

6.5. Translation Control

Translation control refer to end-factor of manipulator precisely transition control in cartesian moment and maintaining orientation stability. Let us consider we have distance target point µ—µd as object functionality so each time lap τ the mean position incremental translation 1µn(τ) of the complying subset (p n < N) calculated as root mean square •2 as given data:

$$\Delta\mu_n(\tau) = \frac{\epsilon}{n}\sum_{i=1}^{n} \nu_i(\tau) \tag{11}$$

This equation calculates the average incremental change $\Delta\mu_n(\tau)$ by scaling the sum of individual terms $\nu_i(\tau)$ by $\frac{\epsilon}{n}$. It's used in adaptive systems like robotics or machine learning to adjust parameters trajectories or weights based on collective feedback, where ϵ controls the adjustment rate and n normalizes the response.

6.6. Spread Control

Spread control responsible for managing multiple-manipulator degree of freedom. this technique optimize coverage while preventing collision. trajectory algorithm maximizes uniform spacing and minimize concentration. this technique helps in collision-free movement, optimal coverage and also load balancing.

$$\mathbf{u}_{\text{spread}} = \mathbf{K}_s(\mathbf{q} - \mathbf{q}_{\text{nom}}) + \mathbf{D}_s\dot{\mathbf{q}} \tag{12}$$

This equation calculates control forces $\mathbf{u}_{\text{spread}}$ to maintain a robotic arm's desired joint configuration $\mathbf{q}_{\text{nom}}$. The stiffness term $\mathbf{K}_s(\mathbf{q} - \mathbf{q}_{\text{nom}})$ pushes joints toward their target positions, while the damping term $\mathbf{D}_s\dot{\mathbf{q}}$ ensures smooth, stable motion. Together, they enable the arm to preserve an optimal posture keeping joints spread for stability during tasks without abrupt movements. The gain matrices $\mathbf{K}_s$ and $\mathbf{D}_s$ determine the system's responsiveness.

7. Distributed Optimization Algorithms

DOA refer to enabling multiple-systems to connect to each other and breakdown the algorithm to subparts. As using multiple optimization algorithms methods and getting multiple results here we distributed optimization algorithms into three general categories in this section: ADMM techniques: Such as DFO methods and sequential convex programming. These groups are categorized according to their suitability for multi-manipulator environmental scenarios and distributed via common junction procedures. Here, we provide a concise overview of each category by examine a representative distributed algorithm optimization. Following an analysis of each separable optimization problem, the following is discussed: Communication and computation: Multiple-manipulator trajectory planning systems require careful balancing of computational algorithm demands. in order to achieve best performance, solution, quality and latency constraints delay should no more then1µs for low-level control to 10ms for high-level computational[20,21]. This interaction between communication and computation is a essential design consideration. using adaptive resolution mechanisms that dynamically transition between high-fidelity cloud plans and efficient edge fallbacks depending on network dynamics. robust architectures edge-cloud use hybridization to distribute processing across nodes. Master control handle raw sensor data and immediate control (sub-millisecond), edge nodes manage local planning (10-100ms), and cloud resources handle large scale computations such as policy learning (>100ms)

$$\mathbf{x}_i^{k+1} = \mathbf{x}_i^k - \alpha \underbrace{\nabla f_i(\mathbf{x}_i^k)}_{\text{Local Objective}} \underbrace{-\beta \sum_{j\in N_i}(\mathbf{x}_i^k - \mathbf{x}_j^k)}_{\text{Consensus}} + \gamma \underbrace{\mathbf{u}_{\text{spread}}^k}_{\text{Physical Control}} \tag{13}$$

- Local Optimization $\nabla f_i(x_i^k)$: The term represents the gradient of robot i's local objective function minimizing tracking error or energy consumption. The weight α scales how aggressively the robot pursues this individual goal.
- Consensus Coordination $\sum_{j\in N_i}(x_i^k - x_j^k)$: This term ensures alignment with neighboring robots $j \in N_i$ by reducing differences between their states $x_i^k - x_j^k$. The weight β controls the strength of coordination, critical for tasks like collaborative lifting or formation control.
- Physical Control u_{spread}^k: Implied but not fully shown, the placeholder u_{spread} suggests additional physical constraints (joint limits, stability) are applied, scaled by γ. In practice, this might include terms like $K_s(q - q_{\text{nom}})$ to maintain safe joint configurations.

7.1. DFO Method

We point out that when extending gradient descent to limited optimization a popular projection on gradient descent a common technique used, which entails projecting the iterates to the feasible set[22,23]. We discuss into more depth on the extensions of gradient descent techniques to limited optimization. These known as first-order techniques because they typically only need to compute the gradient, or the first derivative of the goal and constraint functions. When applying the unconstrained optimization variable is updated in the following way. DFO algorithms avoid this basic issue by permitting each manipulator to use only its local gradients and collaborating with its neighbor to reach a consent on a common solution.

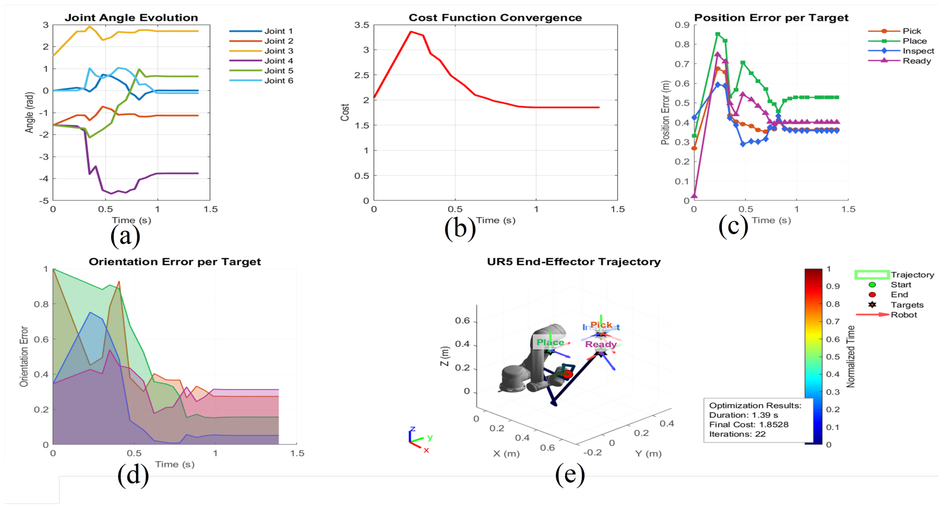

Figure 8. Dynamic Task Allocation and Performance Metrics for Multi- Task Assessment, Environment Scalability and Adaptability Metrics for Multi-Manipulator Task Execution.(a) Joint Angle Evolution tracks angular displacement rad over time s, (A: 0.5 rad, B: 1.0 rad at distinct timesteps).(b)Cost Function Convergence plots optimization progress cost: 1.5 → 0.5 → 0.0 indicating error reduction.(c)Position Error per Target Measures positional deviation during tasks 0.8m → 0.1m, reflecting improving accuracy in phases like "Pick" or "Inspect".(d) Orientation Error per Target Orientation Error per Target improved precision 0.5s. (e) UR5 End-Effector Trajectory robotic arm's end-effector ranges from 0m to 0.6m..

7.2. Sequential Convex Programming

Sequential Convex Programming (SCP) solving provides an iterative framework for Solving challenging non-convex motion planning and optimization issue by computing a series of iterates that indicate the resolution of a number of approximations of the original problem is known as sequential convex programming.

7.3. Multiple-Pose Distributed Optimization

A revolutionary method Multiple-pose distributed optimization which is a dynamically paradigm-shifting approach to selects the most efficient distributed formulation based on real-time system conditions for robotic coordination. This framework adaptively transition between three basic optimization poses. ADMM decomposition for loosely-coupled separatable issues and consensus approaches for tightly-coupled problems. federated averaging in dense networks, for star-topology systems with central coordination[24]. In order to automatically trigger pose transitions the system continuously monitors key metrics like network connectivity, problem separability, and residual convergence.

$$q_i^{k+1} = \arg\min_{q_i} \left(J_i(q_i) + \sum_{j \in N_i} \left(\lambda_{ij}^k q_i + \frac{\rho}{2} \left\| q_i - \frac{q_i^k + q_j^k}{2} \right\|^2 \right) \right) \tag{14}$$

This equation describes the distributed optimization process for a robotic arm or multiple arms at each iteration $k+1$. The goal is to find the optimal joint configuration q_i^{k+1} that minimizes a local cost function $J_i(q_i)$ (tracking error or energy consumption) while ensuring coordination with neighboring robots. The second term enforces consensus among robots by penalizing deviations from the average of their previous configurations $\frac{q_i^k+q_j^k}{2}$, weighted by a penalty parameter ρ. The Lagrange multipliers λ_{ij}^k help balance the trade-off between local optimization and global agreement.

8. Multi-Manipulator Task Assessment

Multi-manipulator task assessment provides a comprehensive framework for estimating and optimizing of collaborative manipulator operations through real-time performance and minimize a trajectory cost and avoid colliding from starting point to desired state configuration. also included the monitoring and adaptive control inputs. The system continuously tracks key metrics such as desired reference trajectory including task completion time by using synchronized ROS2 clocks.

8.1. Flexible and Scalable Advances in Dynamic Task Allocation

Leverage adaptive algorithms to efficiently distribute workloads in real-time, ensuring optimal resource utilization. these modernizations enable seamless task reassignment and scalability, adapting to changing demands while maintaining system performance and reliability. At its core a classified architecture processes energy efficiency and workload balance, creating a multidimensional performance profile[25]. Through joint state data feature extraction pipelines that quantify trajectory smoothness, torque profiles, and completion ratios. For LSTMs temporal pattern analysis, we use to feed into machine learning models like random forests for failure prediction.

$$\mathcal{T} = \sum_{i=1}^{N} \left(\underbrace{w_i \|x_i - x_i^*\|^2}_{\text{Tracking Error}} + \underbrace{\alpha \sum_{j \neq i} \|x_i - x_j\|_{\min}^{-2}}_{\text{Collision Avoidance}} + \underbrace{\beta \|\tau_i\|^2}_{\text{Effort Minimization}} \right) \tag{15}$$

The variables x_i and x_j represent the positions of the i and j agents in the system, respectively. The tracking error term $\|x_i - x_j^*\|^2$ measures how far each agent is from its desired target position x_i^*. The collision avoidance term $\|x_i - x_j\|_{\min}^{-2}$ ensures agents maintain a minimum safe distance by heavily penalizing proximity. enforces a lower bound on separation. The effort term $\|\tau_j\|^2$ quantifies the control input force or torque applied to the i agent, minimizing energy use. Together, these terms ensure precise, safe, and efficient multi-agent coordination.

8.2. Multi-Manipulator Collaborative Work

Multi-manipulator collaborative work including object identification, visual place recognition, depth estimation, 3D mapping, reinforcement learning, multi-manipulator learning involves applying deep learning techniques to estimate functions from data. in shared workspaces consent grasping that dynamically balances applied forces across end-effectors, and collision-aware path trajectory motion planning that integrates swarm potential fields for safe motion[26]..

At the algorithmic level, these systems leverage distributed task allocation through optimized assignment methods such as RRT algorithm, and modern implementations utilize quality-of-service-controlled communication architecture to synchronize state information at 100ms intervals while maintaining μs-level hardware synchronization through Time-Sensitive Networking, with performance continuously monitored via metrics like force balance error (<0.15 target) and collision probability (<0.1%).

$$M_0\ddot{x}_0 + C_0\dot{x}_0 + G_0 = \sum_{i=1}^{n} J_i^T F_i \tag{16}$$

M_0 represents the object's inertial properties in task space. $\ddot{x}_0$ acceleration vector of the system's generalized coordinates. C_0 accounts for Coriolis and centrifugal effects due to rotational motion and G_0 handles gravitational forces. The right side of the equation sums the contributions from each robot, where J_i^T transforms end-effector forces F_i into equivalent joint torques.

9. Conclusion

This study presents a comprehensive framework for path planning and coordinated control of single and multiple robotic manipulators in dynamic, shared environments. By integrating sampling-based algorithms such as Rapidly exploring Random Trees (RRT) and Probabilistic Roadmaps (PRM), along with real-time synchronization via EtherCAT, the system achieves collision-free, adaptive motion trajectories for both repetitive and non-repetitive tasks. Advanced control architectures including PID, MPC, and adaptive pruning optimization—enhance the efficiency and safety of manipulation, particularly in complex scenarios requiring multi-arm cooperation. The implementation of feedback loops through encoders, Kalman filters, and multi-sensor fusion (vision, depth, force) significantly improves object recognition, pose estimation and trajectory adaptation. Moreover, the proposed multi-layered system architecture (physical, coordination, and cognitive layers) ensures robustness, scalability, and real-time responsiveness. Distributed optimization algorithms like ADMM and DFO, coupled with cloud-edge hybrid computing, enable high-performance multi-agent collaboration with minimized latency. Ultimately, this work provides a scalable and intelligent robotic control framework capable of dynamic task adaptation, efficient motion planning, and real-time coordination laying the foundation for future industrial and autonomous robotic systems that require precision, safety, and autonomy in cluttered or unpredictable environments.

Author Contributions: Fazal khan; conceived the study, developed the federated learning framework, implemented the software, conducted experiments on multiple robotic arms, and wrote the initial draft of the manuscript. Zhou meng ;supervised the research, provided critical feedback, secured funding, and contributed to revising and editing the manuscript. All authors reviewed and approved the final version of the paper.

Institutional Review Board Statement: This study was conducted in accordance with the research guidelines of Donghua University. Since the work involved simulated and laboratory-based experiments on robotic arms without human participation or sensitive data, formal ethics approval was not required. All procedures followed standard safety and operational protocols for robotic systems. The research adhered to the university's policies on experimental integrity and data reproducibility. No conflicts of interest or risks were identified. Funding and resources were provided by Donghua University under its institutional research framework.

Acknowledgments: This research was supported by Donghua University. The authors gratefully acknowledge the technical and infrastructural support provided by the Mechatronics Lab at Donghua University, which was essential for conducting experimental work on multi-robot coordination and control. Special thanks to Dr. Zhou Meng for their expert guidance and for providing access to the advanced robotic testbeds, which were critical in validating the proposed framework. The lab's high-precision motion-capture systems and collaborative robotics platforms enabled rigorous experimental testing and accurate data acquisition, significantly improving the reliability of this study.

References

1. Zhang, W.; Yang, D.; Wu, W.; Peng, H.; Zhang, N.; Zhang, H.; Shen, X. Optimizing federated learning in distributed industrial IoT: A multi-agent approach. *IEEE J. Sel. Areas Commun.* **2021**, *39*, 3688–3703. [CrossRef].
2. Zhang, S.Q.; Lin, J.; Zhang, Q. A Multi-Agent Reinforcement Learning Approach for Efficient Client Selection in Federated Learning. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 9091–9099. [CrossRef].
3. Wu, N.; Jia, D.; Li, Z.; He, Z. Trajectory Planning of Robotic Arm Based on Particle Swarm Optimization Algorithm. *Appl. Sci.* **2024**, *14*, 8234. [CrossRef].
4. Liu, W.; Huang, Z.; Qu, Y.; Chen, L. Path Planning for Robotic Arms Based on an Improved RRT Algorithm. *Open J. Appl. Sci.* **2024**, *14*. [CrossRef].
5. Imakura, A.; Sakurai, T. FedDCL: A Federated Data Collaboration Learning as a Hybrid-Type Privacy-Preserving Framework Based on Federated Learning and Data Collaboration. *arXiv* **2024**, arXiv:2409.18356. [CrossRef].
6. Wu, P.; Su, H.; Dong, H.; Liu, T.; Li, M.; Chen, Z. An obstacle avoidance method for robotic arm based on reinforcement learning. *Ind. Robot* **2025**, *52*, 9–17. [CrossRef].
7. Cao, M.; Mao, H.; Tang, X.; et al. A novel RRT*-Connect algorithm for path planning on robotic arm collision avoidance. *Sci. Rep.* **2025**, *15*, 2836. [CrossRef].
8. Le, Q.D.; Yang, E. Adaptive Fault-Tolerant Tracking Control for Multi-Joint Robot Manipulators via Neural Network-Based Synchronization. *Sensors* **2024**, *24*, 6837. [CrossRef].
9. Yu, P.; Tan, N.; Gu, X. A Hybrid Neurodynamic Scheme for Bimanual Synchronized Tracking Control of Robotic Manipulators With Uncertain Kinematics. *IEEE Trans. Ind. Inform.* **2025**, *21*, 2967–2976. [CrossRef].
10. Wang, D.; Zhang, G.; Zhang, T.; Zhang, J.; Chen, R. Time-Synchronized Convergence Control for n-DOF Robotic Manipulators with System Uncertainties. *Sensors* **2024**, *24*, 5986. [CrossRef].
11. Yu, C. Adaptive Iterative Learning Constrained Control for Linear Motor-Driven Gantry Stage with Fault-Tolerant Non-Repetitive Trajectory Tracking. *Mathematics* **2024**, *12*, 1673. [CrossRef].
12. Widanage, K.N.D.; et al. Non-repetitive-path Iterative Learning and Control for Human-guided Robotic Operations on Unknown Surfaces. *IEEE Trans. Robot.* **2025**. [CrossRef].
13. Wang, L.; Zhu, S.; Wei, M.; Wang, X.; Huangfu, Z.; Chen, Y. Iterative Learning Control with Adaptive Kalman Filtering for Trajectory Tracking in Non-Repetitive Time-Varying Systems. *Axioms* **2025**, *14*, 324. [CrossRef].
14. Xie, S.; et al. Predictive Control for an Ankle Rehabilitation Robot Using Differential Evolution Optimization Algorithm-Based Fuzzy NARX Model. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2025**, *33*, 1886–1895. [CrossRef].
15. Tao, S.; Xiang, F.; Shukla, A.; et al. Maniskill3: GPU Parallelized Robotics Simulation and Rendering for Generalizable Embodied AI. *arXiv* **2024**, arXiv:2410.00425. [CrossRef].
16. Wang, Y.; James, S.; Stathopoulou, E.K.; Beltrán-González, C.; Konishi, Y.; Del Bue, A. Autonomous 3-D Reconstruction, Mapping, and Exploration of Indoor Environments With a Robotic Arm. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3340–3347. [CrossRef].
17. Yang, F.; et al. A Collaborative Drone System with Robotic Arm Based on Lightweight Structure and Multimodal Control. *Proc. 2024 4th Int. Conf. Robot. Autom. Intell. Control (ICRAIC)* **2024**, 375–381. [CrossRef].
18. Bi, A.; Zhang, C. Robot Arm Grasping based on Multi-threaded PPO Reinforcement Learning Algorithm. *Proc. 2024 3rd Int. Conf. Artif. Intell. Hum.-Comput. Interact. Robot. (AIHCIR)* **2024**, 369–373. [CrossRef].
19. Shvalb, N.; Hacohen, S.; Medina, O. Statistical Robotics: Controlling Multi Robot Systems Using Statistical-Physics. *IEEE Access* **2024**, *12*, 134739–134753. [CrossRef].
20. Soleimani Amiri, M.; Ramli, R. Intelligent Trajectory Tracking Behavior of a Multi-Joint Robotic Arm via Genetic-Swarm Optimization for the Inverse Kinematic Solution. *Sensors* **2021**, *21*, 3171. [CrossRef].
21. Zhang, H.; Li, X.; Wang, L.; Liu, D.; Wang, S. Construction and Optimization of a Collaborative Harvesting System for Multiple Robotic Arms and an End-Picker in a Trellised Pear Orchard Environment. *Agronomy* **2024**, *14*, 80. [CrossRef].
22. Desai, N.; Behara, G.; Deb, S.; Sen, D. A Genetic Algorithm Based Adaptive and Hybrid Pathfinding Algorithm to Enhance the Performance of Robotic Bin-Picking. *Proc. 2024 IEEE 21st India Counc. Int. Conf. (INDICON)* **2024**, 1–7. [CrossRef].
23. Guo, P.; Luo, D.; Wu, Y.; et al. Coverage Planning for UVC Irradiation: Robot Surface Disinfection Based on Swarm Intelligence Algorithm. *Sensors* **2024**, *24*, 3418. [CrossRef].
24. Zhang, L.; Zhou, X.; Liu, J.; et al. Instance-level 6D pose estimation based on multi-task parameter sharing for robotic grasping. *Sci. Rep.* **2024**, *14*, 7801. [CrossRef].

25. Duan, S.; Jiang, Z.; Ren, Y.; Wei, H. Coordination of Multi-Robot Systems. *Proc. 2022 Int. Conf. Electron. Devices Comput. Sci. (ICEDCS)* **2022**, 361–367. [CrossRef].
26. Ma, H.; Wei, X.; Wang, P.; Zhang, Y.; Cao, X.; Zhou, W. Multi-Arm Global Cooperative Coal Gangue Sorting Method Based on Improved Hungarian Algorithm. *Sensors* **2022**, *22*, 7987. [CrossRef].