Frontiers in Computing and Intelligent Systems
ISSN: 2832-6024 | Vol. 15, No. 3, 2026

Design of an Integrated Visualization-Control Digital Twin and Middleware Architecture for Lightweight Robotic Arms

Junjie Xiong, Lei Ma [*], Xin Wang

School of Mechanical Engineering, Xihua University, Chengdu, Sichuan, CO 610039, China
[*] **Corresponding author**: Lei Ma (Email: sheshuyuan@163.com)

Abstract: With the in-depth advancement of the "Industry 4.0" strategy and the rapid development of intelligent manufacturing technologies, the digital twin, as a key technology bridging the physical world and the digital information world, has demonstrated tremendous application value in the fields of remote monitoring, fault diagnosis, and process simulation of industrial robots. However, traditional robotic arm control systems often suffer from issues such as a low degree of visualization, poor real-time performance in remote interactions, and a disconnection between the simulation environment and the physical entity. To address these challenges, this paper designs and implements an integrated visualization and control digital twin system for a five-axis robotic arm based on the MQTT protocol and a decoupled storage and access architecture. At the communication layer, the system discards the traditional point-to-point tightly coupled model and introduces an IoT distributed architecture based on the MQTT protocol. On the host computer side, a high-fidelity digital twin system is developed utilizing the Unity3D engine. By introducing a concurrent task queue and a packet boundary processing algorithm, high-precision bidirectional synchronization between virtual and physical entities, as well as automated operation planning, are achieved. To resolve the cloud concurrency blocking issue caused by massive high-frequency state data, this paper further designs a data persistence middleware based on Node.js asynchronous processes and the InfluxDB time-series database, realizing data stream cleaning, batch processing, and high-concurrency storage. Comprehensive performance tests demonstrate that the system can achieve bidirectional interaction between the digital twin and the physical entity with millisecond-level latency. Furthermore, the cloud middleware effectively reduces computational overhead, providing a lightweight solution with significant industrial application value for the digital and intelligent operation and maintenance of small and medium-sized robotic systems.

Keywords: Digital Twin; IoT Middleware; MQTT; Unity3D; Time-series Database.

1. Introduction

During the progression towards industrialization and modernization, the deep integration of emerging technologies—such as the Internet of Things (IoT), artificial intelligence (AI), cloud computing, and big data—with traditional manufacturing has become an inevitable trend in the global development of intelligent manufacturing [1]. The crux of transitioning traditional manufacturing toward intelligence and modernization lies in the establishment of Cyber-Physical Systems (CPS) within production, specifically achieving the interactive integration of virtual information and physical entities during the manufacturing process. However, traditional industrial robotic arm control modes primarily rely on on-site handheld teach pendants or local industrial personal computers (IPCs) for point-to-point programming. This closed control architecture exhibits significant limitations. Firstly, operators require professional training, resulting in a high technical threshold and poor flexibility, which makes it difficult to adapt to the demands of customized flexible manufacturing [2]. Secondly, traditional human-machine interaction predominantly relies on two-dimensional interfaces and lacks intuitive visual monitoring methods. Consequently, operators cannot intuitively perceive the real-time posture of the robotic arm in three-dimensional space or its spatial relationship with the surrounding environment [3]. Furthermore, traditional control modes are mostly dedicated solutions for isolated scenarios, resulting in severe data silos. The difficulty of uploading data to the cloud in real time for big data analysis often confines equipment maintenance to a state of reactive, post-fault repair, thereby precluding the realization of data-driven, proactive predictive maintenance [4], as illustrated in Figure 1.

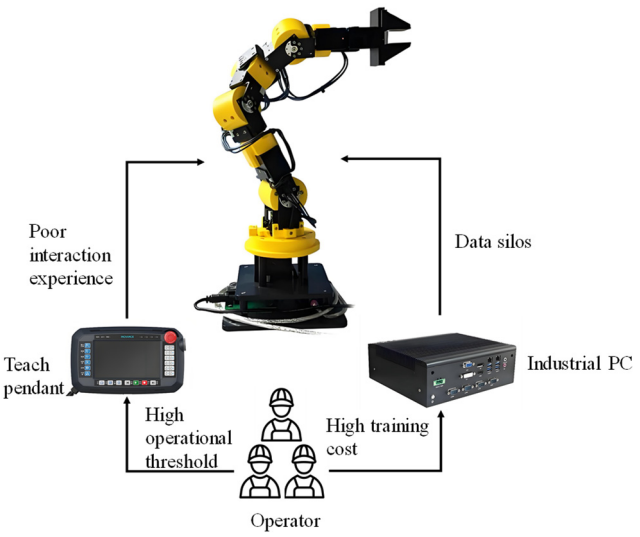


Figure 1. Limitations of the traditional industrial robotic arm control paradigm

To break through spatial and temporal constraints and resolve the disconnection between physical entities and simulation environments, digital twin technology has emerged as an advanced evolutionary form of virtual simulation technology in the Internet of Things (IoT) era [5].

Regarding underlying digital twin platforms and applications, foreign industrial giants dominate the heavy industrial simulation market by virtue of their first-mover advantages. For instance, Siemens' NX Mechatronics Concept Designer (MCD) and Dassault Systèmes' 3DEXPERIENCE platform provide exceptionally powerful full-element physical simulation capabilities [6]. However, such commercial software often suffers from expensive licensing, closed architectures, and high thresholds for secondary development. Meanwhile, domestic scholars have achieved fruitful results in exploring theoretical frameworks for digital twins. For example, the five-dimensional digital twin model proposed by Professor Tao Fei's team at Beihang University provides authoritative theoretical support for the practical application of this technology; numerous domestic research institutes have also made significant breakthroughs in the research and development of shop-floor-level twin management and control platforms [7]. Tang Tang et al. from Tongji University pointed out that the digital twin is a key technology for achieving full-lifecycle closed-loop optimization [8]. Nevertheless, when focusing on small and medium-sized underlying execution equipment, existing domestic digital twin applications are predominantly concentrated on the unidirectional state display level of "data visualization dashboards," generally lacking reverse real-time control capabilities over physical equipment [9]. To seek a balance between development costs and high-fidelity rendering, academia is actively exploring the cross-domain utilization of game engines, such as Unity3D and Unreal Engine, to construct lightweight digital twin systems [10]. At the level of IoT communication and cloud data processing, facing massive, high-frequency equipment telemetry data, traditional direct TCP communication and relational databases are gradually exposing bottlenecks in network congestion and disk read/write operations. Therefore, the introduction of the lightweight MQTT publish-subscribe protocol and an asynchronous storage and access architecture specializing in time-series data has become a key technological development direction for achieving the spatial-temporal decoupling of multi-terminal devices and the value mining of massive historical data.

Addressing the issues of insufficient interactivity, closed architectures, and data silos in existing remote control systems for small and medium-sized robotic arms, this paper takes a five-axis robotic arm as the research object and proposes and implements an integrated visualization and control digital twin system based on the MQTT protocol and a decoupled storage and access architecture. The core research of this paper first lies in breaking through the limitations of traditional industrial configuration software, which can only display data unidirectionally in two-dimensional charts. By leveraging the powerful physical rendering capabilities of the Unity3D engine, a high-fidelity three-dimensional interactive environment is constructed. This environment integrates video monitoring, status data, and control commands into the same 3D space, achieving an immersive reverse control experience that merges the virtual and the real. Secondly, this paper constructs a "device-edge-cloud" fully decoupled communication architecture based on the MQTT protocol, discarding the traditional tightly coupled mode of direct connection to the host computer. By introducing an edge gateway program, protocol standardization and spatial-temporal decoupling between underlying heterogeneous hardware and upper-layer software are achieved. Finally, to achieve full-lifecycle condition monitoring and analysis of the equipment, this paper designs a decoupled storage and access architecture based on the asynchronous middleware Node.js and the time-series database InfluxDB. By introducing memory buffering and asynchronous batch writing strategies, the hidden danger of cloud I/O blocking caused by the concurrent reporting of massive, high-frequency telemetry data is completely eliminated, laying a solid data foundation for the proactive predictive maintenance of the equipment.

2. Overall Distributed Architecture Design

2.1. "Device-Edge-Cloud" Distributed Architecture

This system aims to construct a lightweight digital twin system for a robotic arm that integrates real-time monitoring, remote control, simulation rehearsal, and data analysis. To overcome the limitations of traditional industrial control systems regarding the degree of visualization, interactive experience, and data utilization, and to address challenges such as the difficult integration of heterogeneous devices, stringent real-time requirements, and unstable remote communication, this paper discards the traditional tightly coupled control paradigm of direct connections between host computers and lower-level controllers. Instead, it adopts a three-tier "device-edge-cloud" distributed architecture based on the Internet of Things (IoT) for its top-level design. This architecture completely decouples the logical dependencies among underlying hardware execution, network communication relay, and upper-layer rendering and analysis, thereby endowing the system with exceptional industrial robustness when subjected to the impact of high-frequency concurrent data. The overall logical architecture and data flow of the system are illustrated in Figure 2.

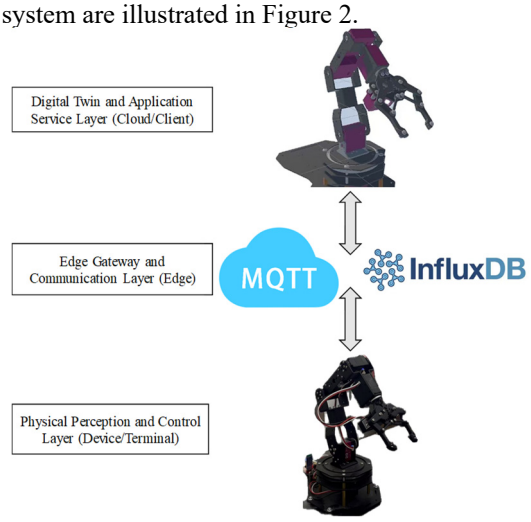


Figure 2. Overall logical architecture and data flow diagram of the system

The lowest level of the system is the physical perception and control layer, namely the "device" layer, which constitutes the physical mapping foundation of the digital twin. Centered on microcontrollers and intelligent bus servos, this layer is primarily responsible for receiving and parsing the end-effector spatial coordinate commands issued by the upper layer, and executing millisecond-level underlying forward and inverse kinematics calculations. Simultaneously, serving as the source of data, this layer collects core load parameters—such as the current physical angles, operating

voltages, and temperatures of each joint—in real-time through high-frequency cycles.

The middle level of the system is the edge gateway and communication layer, namely the "edge" layer. Acting as a bridge connecting the physical world of atoms with the digital world of bits, it undertakes the crucial tasks of protocol conversion and spatial-temporal decoupling. Considering the limited capability of microcontrollers to process complex TCP/IP protocol stacks, this system introduces an independent node running an edge computing gateway program. This node is responsible for parsing the heterogeneous serial data streams uploaded from the underlying layer and encapsulating them into standard JSON-formatted messages. Meanwhile, relying on a high-performance broker deployed in the cloud, this layer constructs a loosely coupled message publish-and-subscribe center, achieving the efficient routing of data flows.

The top level of the system is the digital twin and application service layer, namely the "cloud" layer, which is mainly composed of a 3D rendering client and data service middleware. The client drives the synchronization of the virtual model by subscribing to the real-time state of the robotic arm at a high frequency, and provides an immersive integrated visualization and control interface to issue control commands in reverse. The resident asynchronous middleware and the time-series database work collaboratively to clean and persistently store the massive transient motion data, facilitating full-lifecycle state mining in conjunction with visual dashboards.

2.2. Selection of Core Supporting Technologies

Regarding the selection of key technologies supporting the aforementioned architecture, this paper conducts an in-depth evaluation tailored to the stringent scenarios of robotic arm digital twins. At the communication protocol level, compared to the request-response paradigm of HTTP and the high maintenance costs of WebSocket in multi-terminal collaboration, the system selects the lightweight MQTT publish-subscribe protocol. A characteristic comparison of mainstream IoT and Web communication protocols is presented in Table 1.

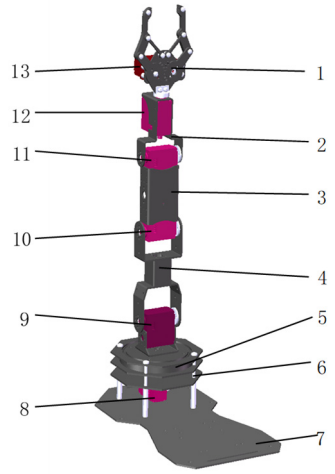
Table 1. Comparison of characteristics among mainstream IoT and Web communication protocols.

Evaluation Dimension	HTTP / REST	WebSocket	Native TCP / UDP	MQTT
Core communication architecture	Request / Response	Full-duplex	Point-to-point	Publish / Subscribe
Packet header overhead	Extremely high	Relatively high	Extremely low	Extremely low
Bidirectional real-time performance	Poor	Extremely high	Extremely high	High
Network fault tolerance & keep-alive	Stateless	Weak	Extremely weak	Extremely strong
Quality of Service (QoS) guarantee	None	None	TCP with acknowledgment, UDP without	Built-in three-level QoS (0, 1, 2) control
Multi-terminal system decoupling degree	Low	Moderate	Extremely low	Extremely high

The fixed packet header of MQTT, which is merely 2 bytes, enables it to effortlessly transmit and receive multidimensional status frames at high frequencies without strain. Its built-in Quality of Service (QoS) mechanism and spatial-temporal decoupling characteristics ensure the reliable delivery of physical safety commands even under severe network jitter [11]. At the host computer platform level, compared to the inadequate 3D rendering capabilities of traditional industrial configuration software and the performance bottlenecks of Web3D front-end technologies when handling high-frequency physical rigid body calculations, this paper selects the Unity3D game engine. Its built-in PhysX physics engine and material rendering system not only authentically restore metallic textures but also perfectly implement rigid body dynamics calculations and continuous collision detection, ensuring that the digital twin is genuinely subject to the laws of physical mechanics. At the database level, in the face of continuous high-frequency sensor data generated by each joint of the robotic arm, the frequent splitting of underlying indexes in traditional relational databases would lead to severe disk I/O bottlenecks. Therefore, the system selects the InfluxDB time-series database, which specializes in IoT scenarios. Based on time-series data indexing, it achieves extremely efficient high-frequency writing and temporal aggregation queries, providing the most solid data foundation for subsequent equipment health assessment and predictive maintenance [12].

3. Abstraction of Physical Entities and Underlying Execution Mechanism

3.1. Robotic Arm Body and Intelligent Drive Unit



1. End-effector gripper、2. Wrist joint connector、3. Forearm joint、4. Upper arm joint、5. Waist rotary platform、6.Rotary platform connector、7. Base、8. Base servo motor、9. Upper arm servo motor、10. Forearm servo motor、11. Wrist pitch servo motor、12. Wrist rotation servo motor 、Gripper actuation servo motor

Figure 3. Structural diagram of the robotic arm

Although the research focus of this paper centers on the upper-layer digital twin engine and cloud data services, the high-precision execution and state feedback mechanisms of the underlying physical entities are the decisive prerequisites for ensuring the physical fidelity of the virtual model. The physical carrier selected for this system is a five-degree-of-freedom (5-DOF) robotic arm adopting a vertically articulated serial structure. The overall assembly consists of a base, an upper arm, a forearm, and an end-effector connected in series sequentially, forming a stable kinematic chain [13], as shown in Figure 3. This configuration simulates the kinematic and physiological characteristics of the human arm, possessing the advantages of a compact structure, flexible movement, and a large workspace.

Regarding the joint drive units, to satisfy the digital twin system's demand for real-time state perception of the underlying equipment, the system abandons traditional open-loop analog servos and instead opts for SP-20D intelligent serial bus servos. The key parameters of the bus servos are presented in Table 2. Unlike traditional PWM servos, which can only passively receive signals, the intelligent bus servo integrates internal sensors for voltage, temperature, and position, thereby possessing robust closed-loop feedback capabilities. The host computer can conduct polling-based monitoring of the physical states of all servos via bus interaction, providing the most fundamental data source guarantee for subsequent virtual-real synchronization and multidimensional health monitoring.

Table 2. Key parameters of the SP-20D intelligent bus servo.

Parameter Category	Parameter Name	Specifications	Remarks
Electrical Characteristics	Operating voltage	5V-8.4V	Wide voltage input, 7.4V recommended
	Stall current	2.3A-3A	Under high load conditions
	No-load current	100mA	During static holding
Mechanical Characteristics	Stall torque	20 kg·cm	At 7.4V power supply
	Operating speed	0.16 sec/60°	At 7.4V power supply
	Gear type	All-metal gears	Improves wear resistance and lifespan
	Product weight	63g (±1g)	Lightweight design
Control Characteristics	Communication protocol	UART asynchronous serial	Half-duplex, baud rate 115200

3.2. Kinematics Calculation and Active Safety Defense

To enable the digital twin system to issue operation commands to the physical entity through intuitive Cartesian spatial coordinates, an efficient inverse kinematics algorithm based on the spatial geometric analysis method is embedded into the underlying controller. The core objective of the inverse kinematics calculation is to inversely solve for the rotation angles of each joint based on the target coordinates $(x,\ y,\ z)$ and the end-effector orientation angle α in three-dimensional space. Addressing the limited floating-point computing capacity of the microcontroller, this algorithm ingeniously performs spatial projection and linkage geometric decoupling, reducing the dimensionality of complex 3D spatial addressing into a 2D vertical plane triangle-solving problem that contains the motion axes of the upper arm and forearm. Assuming the distance and height from the center point of the executing gripper to the base are y' and z' ,respectively, the decoupled mathematical formulas of the linkage mechanism are shown in Equation (1) and Equation (2).

$$y' = \sqrt{x^2 + y^2} - L_3 \cos(\alpha) \tag{1}$$

$$z' = z - L_0 - L_3 \sin(\alpha) \tag{2}$$

Where L_0 represents the height of the base, and L_3 denotes the length of the end-effector gripper. After obtaining the side-length relationships of the triangle following dimensionality reduction, the algorithm precisely deduces the internal angle relationships of the upper arm and the forearm utilizing the law of cosines, thereby determining the rotation angles of each linkage relative to the vertical axis. The topological relationship of its geometric solution is illustrated in Figure 4.

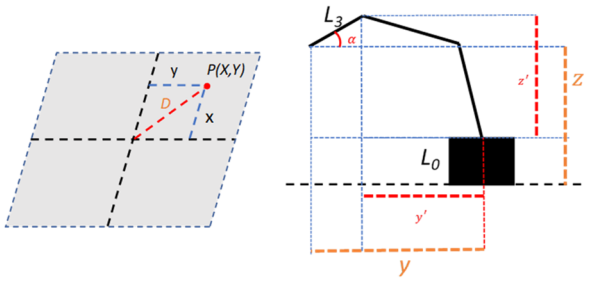


Figure 4. Topological diagram of the geometric solution

Since the joint angle θ_i obtained from the inverse kinematics solution is a continuous mathematical variable, whereas the underlying actuators can only recognize digital pulse signals within a specific range, the system further establishes a rigorous linear mapping model. The control pulse width range accepted by the intelligent servo is set from 500 to 2500, which completely corresponds to the physical range of motion of the servo from 0° to 270°. Its mathematical conversion logic is shown in Equation (3):

$$P_i = 1500 \pm \left(\frac{2000 \cdot \theta_i}{270} \right) \tag{3}$$

More critically, although the digital twin system can perform collision rehearsals in a virtual environment, the underlying physical system still needs to possess absolute self-protection capabilities. To this end, during the algorithm execution phase, the firmware embeds multi-level boundary checks and reachability evaluation mechanisms [14]. Once the calculation process triggers an out-of-bounds exception, the lower-level control program immediately intercepts the motion commands and halts the PWM pulse width updates. Its specific safety defense interception logic is shown in Table 3. This robust, low-level closed-loop logic of "evaluate first, execute later" completely circumvents the risk of physical equipment damage caused by network latency or invalid parameters.

Table 3. Description of inverse kinematics boundary checks and safety defense mechanisms.

Status Code	Logical Trigger Condition	Physical Meaning and Defense Mechanism
0	Execution completed	Successful calculation: The target point is within the physical range, and the angles have been updated.
1	$z' < -L_0$	Ground collision interception: The target point is detected below the physical surface of the base; commands are forcibly halted to protect the workspace surface.
2	$\sqrt{y'^2 + z'^2} > L_1 + L_2$	Reachability violation: The target point exceeds the physical limit of the sum of the link lengths.
3,5	$\cos(A)$or $\cos(\beta)$exceeds limits	No geometric solution: The link lengths cannot form a closed triangle, typically caused by the target point being too close or a kinematic singularity.
4	$\theta_4$exceeds [-135°, 135°]F	Forearm mechanical limit: The calculated angle exceeds the physical safety travel of the forearm servo.
6	θ_5 exceeds [0°, 180°]	Upper arm mechanical limit: The calculated angle exceeds the range of motion of the upper arm servo.
7	$\theta_3$exceeds [-90°, 90°]	Wrist orientation limit exceeded: The compensation angle required to maintain the target posture exceeds the limit of the wrist servo.

3.3. Asynchronous Serial Communication Protocol Design

In the communication link of bidirectional virtual-real interaction, to ensure the atomicity of command parsing and the convenience of cross-platform debugging during long-distance transmission, the underlying system specifically designed an asynchronous serial communication protocol based on ASCII strings. This protocol strictly defines frame header identifiers and frame tail verification characters for different task levels [15]. For instance, complex inverse kinematics commands controlling the spatial addressing of the end-effector begin with "$KMS:", while state readback and direct drive commands are identified by "{...PRAD!}" and "#", respectively. Ultimately, all data frames utilize "!" as the unique mandatory terminator.

This protocol design with explicit start and end character markers possesses significant technical advantages. On the one hand, the microcontroller can employ an interrupt mechanism coupled with a circular buffer for asynchronous reception. Only when the frame terminator "!" is detected does it trigger the segmentation and parsing logic of the state machine, thereby avoiding command fragmentation and misaligned execution caused by TCP stream transmission or network jitter. On the other hand, through different frame header symbols, the lower-level controller can perform logical stratification and priority scheduling of tasks in the first instance, ensuring that heavy floating-point inverse kinematics calculations and ultra-fast PWM register updates do not interfere with each other. This robust underlying communication design lays a solid foundation for the stable operation of the entire digital twin system under the impact of high-frequency state streams.

4. Design of the Robotic Arm Digital Twin Interaction System

The core challenge of a digital twin system lies in achieving a high degree of consistency between the physical entity and the virtual model in geometric, physical, and behavioral dimensions. This chapter focuses on elaborating the construction process of the five-axis robotic arm digital twin system based on the Unity3D engine, covering high-fidelity 3D model preprocessing, the establishment of an asynchronous communication multi-threaded architecture, and the implementation scheme of virtual-real synchronization and the integrated visualization and control interface.

4.1. Twin Model Construction and Multi-body Physical Simulation

Geometric fidelity and physical accuracy are the physical foundations for realizing virtual-real mapping and bidirectional interaction. Geometric fidelity ensures the visual intuitiveness of the robotic arm's posture in the virtual space, while physical accuracy determines the reliability of the system when executing trajectory planning. This system adopts the quadric error metric algorithm to perform mesh decimation preprocessing on non-critical parts. While maintaining external visual characteristics, it significantly reduces the total polygon count, ensuring a high frame rate for system rendering. Subsequently, a tree-nested structure corresponding to the physical entity, from the base to the gripper, is established in the Unity environment. By resetting the centers of rotation, the system ensures that the virtual rotation axes and the physical servo axes completely coincide in three-dimensional space.

To endow the digital twin with authentic mechanical response characteristics, the system conducts in-depth physical modeling of each component of the robotic arm based on Unity's built-in industrial-grade PhysX physics engine. First, rigid body components are added to each joint link, and mass is assigned according to the material density of the physical entity. Based on this, the system automatically calculates the center of gravity and the inertia tensor, ensuring that the dynamic performance during motion complies with the laws of classical mechanics. The core parameter configurations of the robotic arm link rigid bodies are shown in Table 4.

Table 4. Core parameter configuration of robotic arm link rigid bodies.

Parameter Name	Setting Value	Physical Meaning and Configuration Motivation
Mass	0.2-1.0	Dynamic balance: Set according to the actual weight of the link (Unit: kg). The upper arm is set to 1.0, and the forearm and wrist are sequentially decreased to simulate the real moment of inertia gradient.
Drag	0.0-0.1	Dynamic balance: Set according to the actual weight of the link (Unit: kg). The upper arm is set to 1.0, and the forearm and wrist are sequentially decreased to simulate the real moment of inertia gradient.
Angular Drag	0.05-0.2	Mechanical friction: Simulates friction at the joint bearings. Setting it to 0.05 can effectively suppress minor oscillations of the virtual model upon sudden stops, enhancing visual smoothness.
Use Gravity	Enabled	Real load: Allows gravity to participate in calculations. When conducting "power failure tests" or "torque estimation" simulations, this option is a prerequisite for perceiving physical authenticity.
Interpolate	Enabled	Real load: Allows gravity to participate in calculations. When conducting "power failure tests" or "torque estimation" simulations, this option is a prerequisite for perceiving physical authenticity.
Collision Detection	Enabled	Anti-penetration: Due to the fast movement speed of the robotic arm, Continuous Collision Detection (CCD) is recommended to prevent the links from directly passing through obstacle meshes during high-speed motion.

Regarding collision detection, the system employs a compound collider strategy that strikes a balance between performance and precision [16]. For the base and core link segments with relatively regular geometric shapes, low-computational-overhead box colliders are uniformly configured to enhance scanning efficiency; conversely, for the end-effector, which possesses a complex structure and directly engages in operational tasks, high-precision mesh colliders are configured to achieve accurate interference judgment during the grasping process of miniature objects, as illustrated in Figure 5.

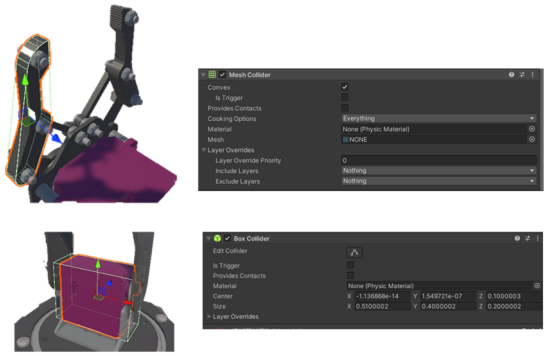


Figure 5. Diagram of the compound collider strategy.

4.2. Asynchronous Communication Architecture and Data Stream Packet Boundary Processing

As the connecting hub of the digital twin system, the communication module is responsible for establishing a bidirectional, low-latency, and high-reliability data transmission link between the physical and virtual spaces. The system constructs a core controller based on the MQTTnet library, employing a decoupled publish/subscribe model to forward data via a cloud broker. The Unity client monitors the angle status information fed back from the lower-level controller in real-time by subscribing to specific topics and achieves remote manipulation of the physical entity by publishing commands to the control topics, as illustrated in Figure 6.

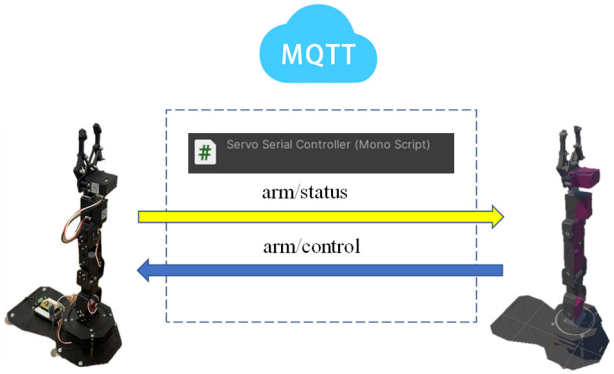


Figure 6. Diagram of the communication architecture.

In Unity development, MQTT callback functions are typically triggered within independent network sub-threads. Since Unity's underlying graphics and transformation APIs are not thread-safe, directly manipulating virtual models within these callback functions would lead to main-thread conflicts and trigger exceptions. To resolve this issue, the system introduces a concurrent queue architecture, where each piece of raw data received by the network thread is encapsulated as a specific operational task and pushed into the queue; subsequently, the Unity main thread cyclically extracts

and executes these tasks during its update cycle. This design achieves deep decoupling between network I/O and graphics rendering, ensuring that the main rendering loop remains stable even under network fluctuations or the impact of large data volumes.

Furthermore, as MQTT is based on the TCP protocol, the characteristics of stream-based transmission often lead to "packet sticking" (coalescing) or "partial packet" phenomena when facing rapid and continuous joint coordinate reporting from the lower-level controller. To ensure the atomicity of command parsing, the system maintains a global string buffer within the controller. Instead of using time-based segmentation, it cyclically retrieves the unique terminator defined in the protocol to perform recursive scanning of the buffer. For data packets containing composite status feedback, the algorithm further performs secondary segmentation based on characteristic identifiers, ensuring that every complete command package arriving within a frame can be accurately stripped, thereby achieving zero-leakage data synchronization.

4.3. Real-time Synchronization Mapping and Smoothing Follow-up Algorithm

The essence of a digital twin system lies in converting the raw sensor pulse-width data, parsed from the communication layer, into the physical power that drives the movement of the virtual model through kinematic mapping and temporal smoothing. The system first employs a linear transformation to map the specific range of PWM pulse-width signals, fed back by the underlying servos, into a normalized scalar space of 0 to 1. Its normalization mapping algorithm is shown in Equation (4):

$$V_{norm} = \frac{PWM_{val} - 500}{2000} \tag{4}$$

The normalized values are subsequently converted into the local Euler angles corresponding to each joint. Considering the differences in mechanical assembly polarity, the algorithm performs direction mirroring and linear interpolation calculations for specific servo IDs when driving model rotation. Due to the discrete nature of network transmission, directly setting the model's pose based on the received data would result in visual jitter. Therefore, the system introduces a time-based interpolation smoothing mechanism and adds a regulation factor to adapt to varying real-time requirements. Let the current frame progress be t; the smoothing logic of the algorithm is shown in Equation (5):

$$t = Clamp01\left(\frac{(T_{current} - T_{start}) \times \alpha}{T_{duration}}\right) \tag{5}$$

The algorithm, combined with the time differential and the pre-defined motion cycle, achieves synchronous stepping of the model angles. This design ensures that even when the sampling frequency of the lower-level controller is low, the virtual model can still approach the target position with exceptional smoothness. In addition to real-time follow-up mapping, the system further implements automated task planning based on Cartesian space. Operators only need to define the 3D physical coordinates of the target anchor point within the virtual environment; the system then constructs an asynchronous state machine and issues macro commands, containing coordinates and execution time, to the edge gateway. The underlying physical robotic arm independently completes complex inverse kinematics closed-loops and servo control. This computation-storage separation architecture, where the upper-layer computer handles task planning and the lower-layer controller manages inverse kinematics execution, not only significantly alleviates the bandwidth burden on the communication bus but also enhances the industrial practical value of the digital twin system.

4.4. Integrated Visualization-Control Interface and Multidimensional Status Monitoring

Traditional industrial robot monitoring interfaces usually separate the 3D view from the 2D control panel, leading to a visual disconnection for the operator. Based on the Unity UI system, this system designs an immersive, integrated visualization-control interface following the Head-Up Display (HUD) principle, where control widgets are suspended as semi-transparent layers above the 3D main viewport. Users can both observe the robotic arm's posture from all angles via mouse interaction and directly control the corresponding servos through the independent slider array on the left,as shown in Figure 7.

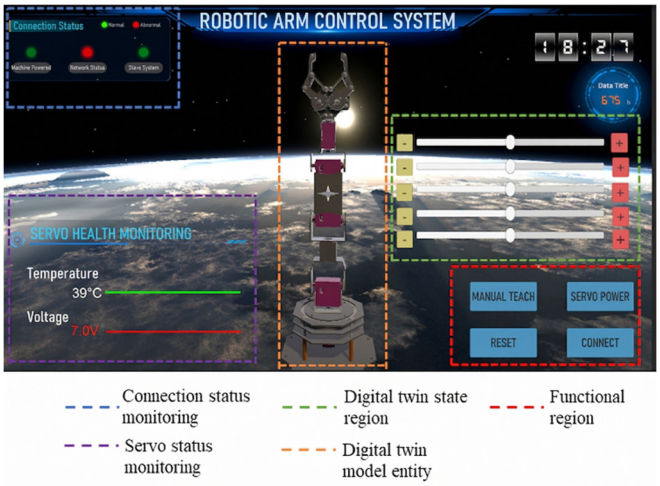


Figure 7. UI layout in Unity3D.

Beyond achieving posture synchronization, the system also designs a dynamic health monitoring module based on custom UI rendering technology. This module can perceive and record the temperature and voltage fluctuations of the servos in real-time. To resolve the issues of layer occlusion and performance overhead inherent in traditional rendering components within the UI interface, the system developed a custom component inheriting from the underlying graphics base class, achieving dynamic construction of vertex data by overriding core functions. The system maps the raw sampling values to the scale space and calculates the final vertex coordinates in conjunction with the pixel dimensions of the container. Its vertical axis mapping formula is shown in Equation (6):

$$P_{ui}.y = \frac{V_{raw} - y_{min}}{y_{max} - y_{min}} \times Height_{rect} \tag{6}$$

This custom mesh rendering significantly reduces the communication overhead from CPU to GPU by submitting the vertex data of a fixed-length temporal buffer in a single pass. Based on the parsed high-frequency sensor data, the system implements a three-tier health rating logic for the servos combined with the rate of temperature rise. When the temperature is below 60°C, the system maintains a normal

white display; between 60°C and 75°C, the UI presents an orange highlight to warn of high loads; once the 75°C critical threshold is exceeded, the system switches to a red flashing mode and prompts emergency power-off protection. This visualization-control system tightly binds the geometric state of the equipment with its physical health, providing an extremely intuitive interactive window for preventative maintenance.

5. High-Concurrency Oriented Asynchronous Data Persistence Middleware Architecture

Industrial digital twin applications rely not only on the real-time mapping of instantaneous states but also require long-term accumulation and value extraction from massive time-series data. To completely eliminate the limitations of severe "data silos" in traditional industrial equipment and passive post-incident repairs, this system constructs a high-performance data service framework on the cloud that operates independently of the Unity main rendering thread. The core of this framework lies in a middleware architecture designed with the separation of storage and computation.

5.1. Cloud Message Broker and Topic Architecture Design

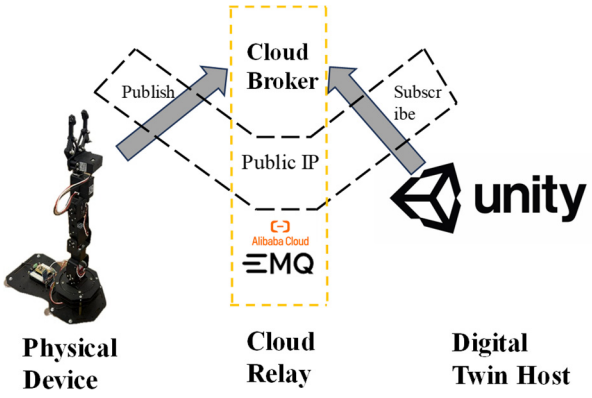


Figure 8. Distributed "Cloud-Edge-Device" network architecture diagram of the system.

For a communication system based on the MQTT protocol, the concurrent processing capability of the broker platform directly determines the flow latency of underlying high-frequency sampling data and upper-layer control commands. Compared to the Mosquitto single-threaded architecture, which easily encounters performance bottlenecks under massive high-concurrency, or the overly "heavy" architecture of traditional enterprise-grade RabbitMQ, this system ultimately selects EMQX as the core MQTT message broker. Leveraging the concurrency model of the Erlang language, this platform can process the parallel access of massive IoT devices with extremely high throughput and millisecond-level data forwarding latency. In terms of network architecture, the cloud server is allocated a fixed public IP and opens a standard communication gateway, completely overcoming the issues of frequent IP address changes and firewall blocking common in traditional LAN debugging, as illustrated in Figure 8.

Regarding the data synergy of multi-terminal heterogeneous devices, the system constructs a rigorous topic hierarchy and standardized payload specifications on the cloud. The control topics (e.g., arm/control) serve as the sole entry point for the upper-layer computer to issue motion logic. Its payload adopts an ASCII character stream format with specific prefixes such as $KMS: or #. This plaintext protocol design not only enhances the visualization for cross-platform debugging but also provides robust frame reassembly capabilities in jittery network environments using the terminator !. Status topics (e.g., arm/status) are pushed at high frequency by the underlying robotic arm. By encapsulating fixed-length composite messages such as #IDTxxVyy!, they accurately carry the temperature and voltage feedback of each joint, providing a standardized data stream entry for subsequent regex extraction in the middleware.

5.2. Asynchronous Data Persistence Middleware Design

Due to constraints in computing power and memory, the underlying hardware only possesses basic MQTT networking capabilities and cannot directly initiate complex HTTP requests to invoke database APIs. To this end, the system developed an asynchronous persistence middleware service on the cloud server. Regarding the selection of development languages, compared to traditional synchronous blocking architectures like Java, this system selects Node.js to construct the cloud middleware. Relying on the single-threaded, fully asynchronous non-blocking I/O model and the event loop mechanism of the V8 engine, Node.js can easily mitigate massive concurrency impacts with extremely low memory footprints, efficiently completing real-time cleaning of MQTT messages and asynchronous disk writing for the database.

On the data cleaning side, the middleware utilizes regular expressions (Regex) to intercept and parse the high-frequency influx of fragmented ASCII character streams. For example, temperature and voltage are extracted via the matching pattern /T(\d+)V(\d+)/, and pulse widths are extracted via the pattern /P(\d+)/, which are then converted into actual physical angles based on mechanical physical constants. Upon successful extraction, the data is immediately instantiated as a standard Point object for the time-series database and automatically tagged with dimensions including the servo ID. To minimize the disk I/O pressure on the cloud server, the system eschews the inefficient single-trigger write mode in favor of an in-depth batch asynchronous buffering strategy. The core parameter configurations of the cloud asynchronous data persistence middleware are shown in Table 5.

This ingenious "peak clipping and valley filling" mechanism—triggering a disk write every 5000 ms as a forced flush or whenever the queue accumulates 1000 data entries—perfectly transforms high-frequency random writes into low-frequency sequential writes. This significantly releases the CPU computing power of the cloud server and dramatically enhances the system's throughput and availability under extreme network traffic peaks.

5.3. Industrial-grade Time-series Data Monitoring and Visualization

Upon successful data persistence into the database, the system introduces Grafana, an industry-leading open-source visualization and analysis platform, for front-end presentation [17]. Utilizing the Flux query engine of InfluxDB, the system performs deep feature mining on the equipment's operating status across three dimensions: thermodynamics, electricity, and kinematics, as shown in Table 6.

Table 5. Core parameter configuration of the cloud asynchronous data persistence middleware.

Module Category	Core Parameters & Strategies	Set Value / Matching Rules	Engineering Significance and Mechanism Description
MQTT Subscription Side	Keep-alive interval	60 sec	Maintains a healthy TCP connection between the cloud and edge gateway, preventing "zombie" connections and misjudgments.
	Reconnection period	1000 ms	Responds to complex network jitter, achieving rapid and seamless reconnection after disconnection.
	Quality of Service (QoS)	0	Exchanges a minimal probability of packet loss for the highest throughput real-time performance, preventing cloud message accumulation.
Data Cleaning Side	Telemetry status extraction	/T(\d+)V(\d+)/	Efficiently extracts temperature and voltage values from high-frequency, fragmented ASCII character streams.
	Pulse width extraction & conversion	/P(\d+)/ etc.	Extracts pulse width and converts it into actual physical angles at a rate of 7.407 PWM/degree.
	Time-series object boxing	Add id tag	Instantiates standard InfluxDB objects, automatically appending the servo serial number as a dimensional tag.
InfluxDB Writing Side	Forced flush period	5000 ms	Enables the Batching strategy, periodically persisting memory buffer queues to disk to reduce I/O frequency.
	Batch capacity	1000 records	Immediately triggers writing when data accumulation reaches 1000 records, perfectly achieving "peak clipping and valley filling."
	Max retry delay	15000 ms	Exponential backoff strategy for transient database service rejections, enhancing system availability under extreme conditions.

Table 6. Specific functional design and engineering analysis value of Grafana.

Monitoring Dimension	Core Physical Quantity	Data Processing and Visualization Configuration
Thermodynamic Features	Multi-joint temperature	Downsampling aggregation based on filter and mean; renders comparative line charts labeled by id tags.
Electrical Features	Bus voltage	High-frequency voltage detection; deeply integrates Grafana threshold alarms (adaptive red highlighting when < 6.5V).
Kinematic Features	Posture angles	Extracts actual feedback angles; sets ultra-short 5-second auto-refresh rates to plot time-series trajectories.

Regarding thermodynamic features, the system utilizes downsampling aggregation functions to render multi-joint temperatures within the same time dimension into intuitive comparative line charts. This profoundly reveals the differences in thermal gradients between the underlying base and the end-effector grippers, providing a data reference for setting thresholds in over-load protection algorithms. For electrical features, the system conducts high-frequency voltage detection on the power bus to capture the voltage sag phenomena triggered during multi-axis simultaneous movement, deeply integrating the Grafana threshold alarm mechanism. Once the voltage curve drops below the 6.5V critical boundary, the panel background adaptively presents red highlighting to prevent the risk of battery depletion. In terms of kinematic features, the system extracts actual feedback angles and configures an ultra-short auto-refresh rate to depict the time-series trajectories of multi-axis workflows. Engineering personnel can overlay and compare these with theoretical commanded trajectories, thereby achieving precise quantitative assessment of physical

tracking errors and mechanical gear backlash. The Grafana front-end monitoring dashboard is shown in Figure 9.

Figure 9. Grafana front-end monitoring dashboard

6. System Integrated Testing and Performance Analysis

6.1. Virtual-Real Mapping Precision and Control Consistency Test

To comprehensively verify the engineering feasibility of this digital twin system in core virtual-real mapping precision and closed-loop control, multi-dimensional quantitative tests were conducted in a collaborative environment consisting of a public cloud server and actual physical equipment. Regarding virtual-real synchronization and control functions, the system focused on the repeatability of joint space based on data closed-loops. By issuing spatial addressing commands from the upper-layer computer and directly reading feedback from the underlying physical magnetic encoders, the system automated 30 repetitive cycles of core operational trajectories. Experimental data indicate that for lightly loaded joints such as the base, forearm, and wrist, the physical deviations after overcoming static friction and gear backlash are perfectly suppressed within the sampling resolution limits of the underlying hardware. Even for the upper arm joint, which bears the maximum gravitational torque of the system, infinitesimal physical deformations are restricted within a minute fluctuation range by the underlying closed-loop algorithm. These results conclusively prove that the system possesses high-fidelity physical authenticity and control consistency, fully capable of meeting the automated operation simulation requirements for small and medium-sized equipment.

6.2. Network Communication Latency and Middleware Concurrency Stress Test

In terms of wide-area network (WAN) communication and cloud middleware stress testing, the Round-Trip Time (RTT) statistical results demonstrate excellent real-time responsiveness. Regarding the network flow efficiency of the MQTT bus, the test recorded the complete closed-loop time from control command issuance to status confirmation message return, as shown in Figure 10.

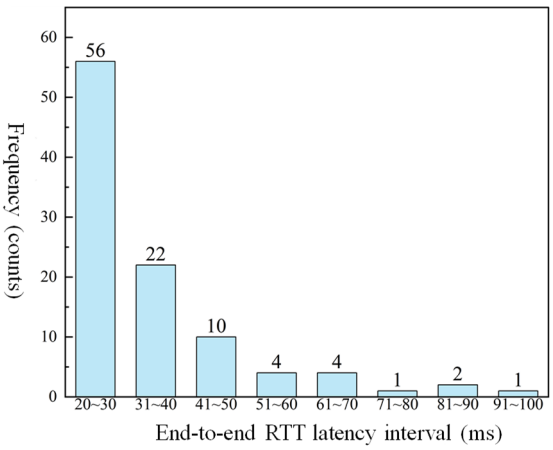


Figure 10. Frequency distribution histogram of full-link RTT latency.

Statistics show that 78% of the command response samples are highly concentrated in the core interval of 20 ms to 30 ms, with an average RTT of only 35.2 ms. This indicator is well below the 100 ms "visual imperceptibility" threshold for industrial human-machine interaction, fully confirming the high determinism and ultra-low latency characteristics of the EMQX message queue architecture in cross-network environments.

Regarding the stress resistance of cloud data persistence, comparative experiments strongly demonstrate the superiority of the separation of storage and computation middleware architecture. Under extreme conditions simulating high-frequency concurrent reporting from five-axis joints, as shown in Figure 11:

The traditional single-sync write mode caused dense computing spikes in the cloud server's CPU, with the average load climbing to approximately 4.5%. Conversely, when the

system switched to the Node.js-based asynchronous batch write mode, high-frequency random write operations were perfectly transformed into low-frequency sequential writes within the memory buffer. Consequently, the cloud processor load was compressed dramatically and remained stable at around 1.2%. This significant performance leap completely eliminates the risks of resource depletion and data overflow brought by massive transient concurrent I/O, proving that the middleware architecture is fully competent for aggregating industrial-grade high-frequency time-series data.

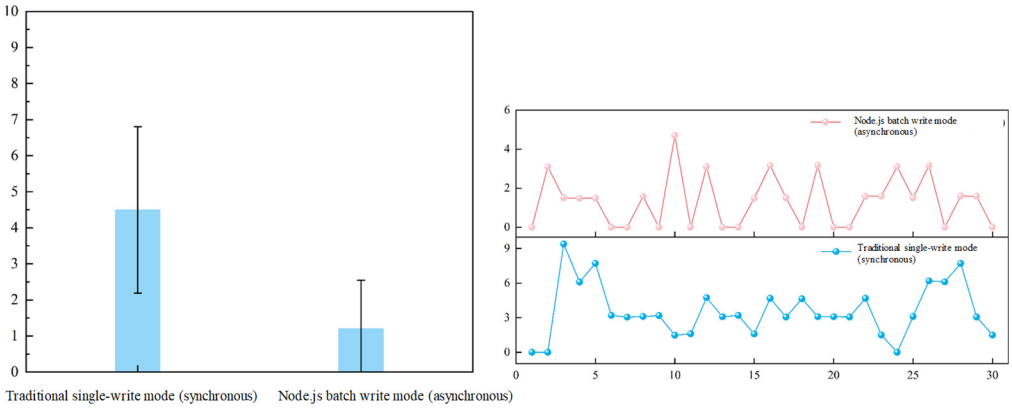


Figure 11. Performance test diagram of the InfluxDB persistence link

7. Conclusion and Outlook

This paper addresses the pain points of traditional industrial robotic arm control systems, such as closed monitoring, poor interaction experience, and low data utilization, by designing and implementing a five-axis robotic arm digital twin and data service middleware system based on a Cloud-Edge-Device architecture. By introducing the Unity3D engine and industrial-grade physical dynamics simulation, this study breaks the display limitations of traditional configuration software that rely solely on 2D charts, successfully constructing a high-fidelity, low-latency, and integrated visualization-control immersive interaction environment. Building upon this architecture, the system utilizes the lightweight MQTT IoT protocol as the communication hub and integrates the Node.js asynchronous non-blocking mechanism with the InfluxDB time-series database to creatively establish a high-performance, storage-computation-separated data gateway.

Comprehensive experiments fully verify that the system not only possesses excellent physical mapping precision exceeding the resolution limits of the underlying hardware but also meets rigorous industrial real-time monitoring standards with sub-second full-link communication latency and high-concurrency cloud capacity. The results of this research provide an efficient and implementable technical path for lightweight industrial equipment to move towards transparent, digital, and intelligent operation and maintenance. Furthermore, it lays a solid hardware and software infrastructure for the future introduction of deep learning algorithms for flexible grasping prediction and predictive maintenance based on long-term historical data.

Admittedly, virtual simulation software for education has certain limitations; for instance, it may lack fine detail when dealing with highly complex physical models or scenes, and real-world production involves many uncontrollable factors that ideal simulation environments cannot perfectly replicate. Despite these limitations, virtual simulation offers unique and immense advantages for the education industry. As technology advances, the constraints of virtual simulation will be mitigated, and its diverse modalities will progressively improve teaching quality while lowering the costs and barriers to learning.

Acknowledgments

The work was supported by Sichuan Provincial Higher Education Talent Cultivation Quality and Teaching Reform Project (JG2024-0780, JG2024-0764), Graduate Education Teaching Reform and Practice Project (YJG202318, YJG 202406, YJG202407, YJG202515), Natural Science Foundation of Sichuan Province (2024ZYD0003, 2025ZY D0173), and Sichuan Provincial Science and Technology Department Project (2025ZYD0173).

References

[1] T. Tan, “Scheduling optimization of guide vane grinding automated production line based on digital twin,” M.S. thesis, Chongqing Univ. Technol., Chongqing, China, 2025.

[2] Y. K. Liu, K. Yang, B. B. Tuo, et al., “Digital twin industrial robots: Conceptual framework, key technologies and case studies,” J. System Simulation, vol. 37, no. 7, pp. 1723–1752, 2025.

[3] Y. Y. Du, Y. Luo, Y. B. Peng, et al., “Three-dimensional visual monitoring of industrial robots based on digital twin,” Comput. Integr. Manuf. Syst., vol. 29, no. 6, pp. 2130–2138, 2023.

[4] F. Tao, H. Zhang, C. Y. Zhang, et al., “From digital twin to digital intelligence twin: Review and outlook,” Comput. Integr. Manuf. Syst., vol. 32, no. 1, pp. 1–17, 2026.

[5] C. B. Zhuang, J. H. Liu, H. Xiong, et al., “Connotation, architecture and development trend of product digital twin,” Comput. Integr. Manuf. Syst., vol. 23, no. 4, pp. 753–768, 2017.

[6] F. Liu, “Design and implementation of intelligent industrial robot system based on ROS,” M.S. thesis, Univ. Chinese Acad. Sci., Shenyang, China, 2017.

[7] F. Tao, M. Z. Zhang, L. S. Nie, et al., “Digital twin workshop: A new paradigm for future workshop operation,” Comput. Integr. Manuf. Syst., vol. 23, no. 1, pp. 1–9, 2017.

[8] F. Qiu, M. Chen, L. Wang, Y. Ying, and T. Tang, “The architecture evolution of intelligent factory logistics digital twin from planning, implement to operation,” Adv. Mech. Eng., vol. 15, no. 9, Art. no. 16878132231198902, 2023.

[9] H. Li, H. Q. Wang, G. Liu, et al., “Concept, system structure and operation mode of industrial digital twin system,” Comput. Integr. Manuf. Syst., vol. 27, no. 12, pp. 3373–3390, 2021.

[10] R. González-Herbón, G. González-Mateos, J. R. Rodríguez-Ossorio, M. Domínguez, S. Alonso, and J. J. Fuertes, “An approach to develop digital twins in industry,” Sensors, vol. 24, no. 3, Art. no. 998, 2024.

[11] J. T. Wang, “Research on IoT device access and cloud monitoring system based on MQTT,” M.S. thesis, Xi'an Technol. Univ., Xi'an, China, 2021.

[12] J. L. Zhou, Y. C. Liu, J. K. Zhu, et al., “Design of intelligent construction machinery management and control platform based on industrial internet of things,” Electron. Test, no. 7, pp. 34–38, 2021.

[13] L. Q. Zhang, “Research on motion and control of snake-like robotic arm,” M.S. thesis, Northeastern Univ., Shenyang, China, 2018.

[14] L. Zhang, H. Du, Z. Qin, Y. Zhao, and G. Yang, “Real-time optimized inverse kinematics of redundant robots under inequality constraints,” Sci. Rep., vol. 14, Art. no. 29754, 2024.

[15] J. Wytrębowicz, K. Cabaj, and J. Krawiec, “Messaging protocols for IoT systems—A pragmatic comparison,” Sensors, vol. 21, no. 20, Art. no. 6904, 2021.

[16] D. Wu, Z. Yu, A. Adili, and F. Zhao, “A self-collision detection algorithm of a dual-manipulator system based on GJK and deep learning,” Sensors, vol. 23, no. 1, Art. no. 523, 2023.

[17] K. R. Yang, “Research on virtual-real data fusion processing technology of industrial digital twin,” M.S. thesis, Xidian Univ., Xi'an, China, 2021.