International Journal of Scientific Research and Engineering Development-– Volume 9 Issue 4, Jul – Aug 2026
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Research on Key State Acquisition and Visualization for Industrial Robots Based on Node-RED

Lei Zhou
School of Cixi, Zhejiang Business Technology Institute, Ningbo, Zhejiang, China
Email: pahuoxi@zbti.edu.cn

----------**********----------

Abstract:

Industrial-robot laboratories in vocational colleges often have limited budgets, heterogeneous equipment, and teaching-oriented maintenance requirements. Commercial industrial-Internet platforms provide powerful monitoring functions, but their server requirements, customization costs, and deployment complexity are not well matched to small teaching laboratories. At the same time, fully custom monitoring software requires expertise in industrial communication, database design, service interfaces, and front-end development, which makes it difficult for teaching teams to maintain and reproduce in courses. To address these problems, this study designs a key-state acquisition and visualization system for an ABB IRB 120 industrial robot based on Node-RED. The system uses a three-level robot-PLC-Node-RED architecture: Modbus TCP reads controller registers at the field layer; an S7-1200 PLC performs data conversion, moving-average filtering, threshold judgment, and JSON encapsulation; and MQTT transports standardized messages to Node-RED. Node-RED then parses the payload, routes the state values, stores historical records in SQLite, and presents them through a lightweight dashboard. Following the robot controller manual and industrial-robot maintenance specifications, the study screens six categories of key states from twenty-three output parameters: J1-J3 motor temperatures, J1 motor current, controller internal temperature, operating mode, and gripper input/output status. Mathematical models are introduced for moving-average filtering, normalized deviation, weighted health indexing, and alarm persistence. Four diagrams illustrate the system architecture, low-code data pipeline, state-evaluation model, and dashboard layout. Four tables define parameter attributes, protocol characteristics, design targets and test methods, and teaching cases. All numerical performance indicators are identified explicitly as research and verification targets rather than as completed experimental measurements. The resulting design provides a low-cost, interpretable, and extensible monitoring scheme for small teaching laboratories and converts equipment-maintenance data into reusable teaching resources.

Keywords — **industrial robot, Node-RED, key state acquisition, low-code development, Modbus TCP, MQTT, visualization**

----------**********----------

I. INTRODUCTION

With the deepening integration of intelligent manufacturing and vocational education, industrial robots have become important teaching and training equipment in robotics technology, mechatronics, and electrical automation programs. Students learn robot programming, trajectory execution, and basic maintenance, while colleges seek to keep laboratory equipment safe, reliable, and strongly connected with real industrial practice. Nevertheless, the internal state of a robot often remains a black box in ordinary laboratory teaching. Students can observe the motion of the arm, but they cannot readily see the accompanying changes in motor current, joint temperature, controller temperature, or gripper input and output signals. Teachers and laboratory managers likewise lack a convenient historical record that can reveal early risks such as motor overload, joint jamming, abnormal current rise, or poor controller heat dissipation.

State acquisition and visualization are therefore not merely software-engineering tasks. For a vocational college, they are also a means of converting an opaque industrial process into explainable teaching resources. If joint temperature, current load, operating mode, and IO events are recorded and displayed together, students can connect abstract parameters with concrete robot behavior. Teachers can use real laboratory data to explain load variation, communication conversion, threshold judgment, and preventive maintenance. Laboratory managers can consult historical curves when investigating an abnormal event rather than relying only on an operator's recollection.

Large industrial-Internet platforms already provide mature solutions for large-scale factories. Platforms such as Siemens MindSphere and ABB Ability support heterogeneous device access, remote operation, data aggregation, and analytics. These capabilities are valuable for multi-line production systems, but their typical requirements, including dedicated servers, customized software, long implementation cycles, specialized operation teams, and network infrastructure, can exceed the resources of a teaching laboratory. For a college with a small number of robots and PLCs, adopting such a platform may result in excessive cost and technical overcapacity while still failing to meet classroom needs.

A conventional custom monitoring system brings a different difficulty. A development team normally has to write upper-computer communication code, design database tables, implement service interfaces, and build a front end. This is time-consuming for a small teaching team and is not easy for students to reproduce in a course. Students often see the final interface but not the complete path from field register, communication protocol, data conversion, threshold logic, database record, to visual component. Low-code development offers an alternative path. Node-RED is a flow-based programming environment built on Node.js. It provides nodes for MQTT, HTTP, WebSocket, JSON processing, functions, databases, and dashboard widgets. Its node-and-wire representation makes the data path and business logic explicit, which is useful both for rapid engineering iteration and for classroom demonstration.

This paper is based on the college-level research project Research on Key State Acquisition and Visualization for Industrial Robots Based on Node-RED. It addresses three questions. First, how should key state parameters be selected from equipment manuals and maintenance specifications under the constraints of a teaching laboratory? Second, how can a robot, a PLC, and Node-RED form a stable and explainable cross-protocol data chain? Third, how can a lightweight visualization system satisfy both equipment maintenance and classroom teaching? The main contributions are: (1) a three-level robot-PLC-Node-RED architecture adapted to vocational-college resources; (2) mathematical formulations for moving-average filtering, normalized deviation, weighted health indexing, and persistent alarm judgment; and (3) a reusable design method consisting of parameter mapping tables, structural diagrams, dashboard layouts, test plans, and teaching cases.

II. RELATED RESEARCH AND TECHNICAL FOUNDATIONS

A. State-Monitoring Requirements of Industrial Robots

Industrial-robot state monitoring may include temperature, current, torque, vibration, position error, speed, operating mode, safety-circuit status, and end-effector state. A mass-production factory is usually most concerned with overall equipment effectiveness, downtime loss, production prediction, and quality correlation. A teaching laboratory has different priorities: operation safety, equipment life, parameter interpretability, and the educational value of the monitoring process. Consequently, collecting every available parameter is not necessarily the best design. Parameters should be selected according to fault impact, teaching value, acquisition difficulty, communication load, and the maintenance actions that the data can support.

Motor temperature and current are especially useful for detecting overload, jamming, and heat-dissipation problems. A persistent temperature rise may result from excessive load, insufficient lubrication, reducer abnormality, or poor

environmental ventilation. A sudden current increase may indicate load shock, abrupt deceleration, or mechanical jamming. Controller internal temperature reflects the thermal condition of the control cabinet, fans, filters, and surrounding installation environment. Operating mode and gripper IO status do not directly express equipment health, but they reconstruct the operating context of an abnormal event. Their educational value is high because students can connect the program state, physical action, and parameter change.

B. Complementary Roles of Modbus TCP and MQTT

Modbus TCP is a widely used industrial field protocol based on a master-slave model and register and coil addressing. It is simple, mature, and well supported by PLCs. Its deterministic polling model is suitable for reading robot-controller registers from an S7-1200 PLC. However, its data model is low-level and strongly coupled with device addresses. If a dashboard were directly bound to Modbus registers, later device replacement or address adjustment would require extensive front-end modification.

MQTT is a lightweight publish-subscribe protocol. A broker decouples message producers from consumers through topics, making it easier to connect one data source to several consumers, such as a dashboard, a database writer, and an alarm module. QoS levels allow different transmission guarantees for different parameter categories. In this study, Modbus TCP is used between the robot controller and PLC, while MQTT is used from the PLC or edge side to Node-RED. The two protocols are therefore complementary: Modbus TCP supports field-level acquisition, and MQTT supports information-level decoupling and visualization.

C. Characteristics of Node-RED Low-Code Development

Node-RED was originally designed for rapid IoT application development. A developer works in a browser, drags nodes onto a canvas, configures parameters, and connects ports to form a data-processing flow. For this project, its advantages are threefold. First, the required ecosystem is relatively complete: MQTT input, JSON parsing, function calculation, switch routing, SQLite storage, and dashboard widgets are available. Second, the flow diagram is itself technical documentation; students can trace how data move from topic subscription to chart display. Third, modification cost is low, so interface layout, alarm thresholds, and query functions can be adjusted according to teaching feedback.

Low code does not mean that industrial design can be arbitrary. Address mapping, byte order, data types, engineering units, invalid-value handling, timestamp consistency, and alarm persistence still require professional care. Node-RED reduces implementation friction in the presentation and integration layers, but it does not remove the need for rigorous engineering validation.

III. OVERALL SYSTEM DESIGN

A. Design Principles

Five principles guide the design. Safety: the acquisition system must not change the robot's original control logic or bypass manufacturer safety mechanisms. Lightweight deployment: existing laboratory software and hardware should be used where possible, avoiding large servers and unnecessary commercial-platform modules. Interpretability: communication path, data format, filtering method, and threshold rules should be visible and explainable. Teaching adaptation: the interface should serve maintenance staff and students at the same time. Extensibility: parameter mapping and protocol adapters should allow the later addition of other devices, such as a FANUC LR Mate 200iD using FOCAS.

B. Architecture

The system adopts a robot-PLC-Node-RED architecture, as shown in Fig. 1. The field layer includes the ABB IRB 120 robot, its controller, joint motors, temperature feedback, current feedback, and gripper IO. The control layer uses an S7-1200 PLC configured in TIA Portal as a Modbus TCP master; it reads registers, converts data types, filters signals, judges thresholds, and packages JSON messages. The communication layer contains a local Mosquitto MQTT broker. The processing layer in Node-RED parses JSON, routes data, triggers alarms, and writes records to SQLite. The visualization layer provides

temperature curves, a current gauge, mode icons, IO indicators, alarm logs, and historical queries. The application layer serves equipment maintenance, classroom teaching, student projects, and later extension to more devices.

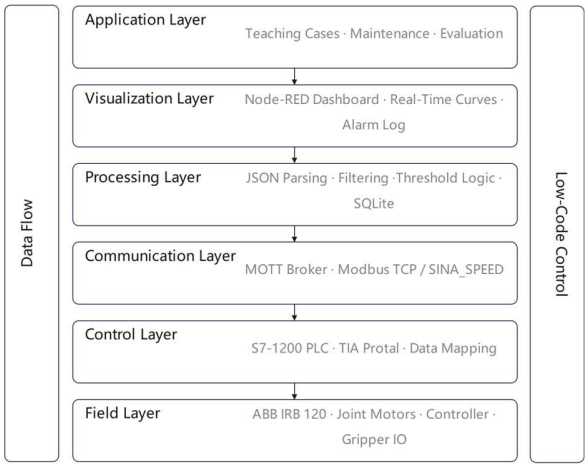

Fig. 1 Architecture of the Node-RED-based industrial-robot state-acquisition and visualization system

The central idea is to separate industrial real-time processing from teaching-oriented presentation. The PLC is close to the field and is appropriate for cyclic acquisition and basic judgment. Node-RED is outside the real-time motion-control loop and is appropriate for data organization, storage, interaction, and visualization. This division prevents Node-RED from becoming a weak link in motion control while preventing the PLC program from becoming a complicated user-interface development task.

IV. KEY STATE PARAMETER SCREENING AND MAPPING

A. Screening Basis

The controller manual for the ABB IRB 120 is used as the source of available output parameters. Twenty-three output items are classified and compared with industrial-robot maintenance requirements. Four dimensions are considered: fault impact, teaching value, acquisition difficulty, and communication load. Six categories of key states are selected: J1-J3 motor temperatures, J1 motor current, controller internal temperature, operating mode, and gripper IO status. Temperature and current belong to health-related parameters; operating mode and gripper status belong to process-related parameters.

The use of J1-axis current in the first implementation stage is a practical choice, not a claim that monitoring one joint is sufficient for complete health evaluation. If communication resources permit, the design should be extended to other joint currents and load-related variables. J1 current is used here to explain data conversion, filtering, threshold judgment, and dashboard mapping in a controlled way.

B. Parameter Attributes

Table 1 lists the parameter category, data type, sampling frequency, and visualization form. Sampling frequency follows the dynamic characteristics of each signal. Temperature changes relatively slowly, so 1 Hz is sufficient to observe trends. Current and IO status can change rapidly, so 2 Hz is assigned. Operating mode is used for context reconstruction and is sampled at 1 Hz.

TABLE 1
Attribute Description Of Key State Parameters For The Industrial Robot

No.	Parameter	Category	Data type	Sampling rate	Visualization
1	J1 motor temperature	Health	Float, °C	1 Hz	Temperature curve and threshold line
2	J2 motor temperature	Health	Float, °C	1 Hz	Temperature curve and threshold line
3	J3 motor temperature	Health	Float, °C	1 Hz	Temperature curve and threshold line
4	J1 motor current	Health/load	Float, A	2 Hz	Gauge and load trend
5	Controller internal temperature	Health	Float, °C	1 Hz	Temperature curve and status color

No.	Parameter	Category	Data type	Sampling rate	Visualization
6	Operating mode	Process	Enumeration	1 Hz	Icon and text label
7	Gripper IO status	Process	Boolean/enumeration	2 Hz	IO indicator and event record

V. DATA ACQUISITION AND PROCESSING METHODS

A. Modbus TCP Reading and PLC Preprocessing

The S7-1200 PLC acts as a Modbus TCP master and cyclically reads holding registers exposed by the robot controller. Parameters may differ in register length and encoding, so the PLC program must process byte order, integer and floating-point conversion, scale conversion, and invalid-value filtering. For example, if a temperature is supplied as a scaled integer, it must be converted into degrees Celsius using the documented scale. If a value occupies two 16-bit registers as a 32-bit floating-point number, high-word and low-word order must be handled consistently. The objective is to ensure that every value entering the upper layers has a consistent engineering unit and a known timestamp.

The PLC also performs preprocessing. Moving-average filtering is applied to fluctuating temperature and current values. Discrete signals are debounced for a short period to prevent IO jitter from producing false alarms. Filtered values, timestamps, device identifiers, parameter codes, and status codes are packaged as JSON messages and published to MQTT topics. This preprocessing reduces unnecessary upper-layer computation and preserves a clear boundary between field logic and visualization logic.

B. Moving-Average Filtering Model

Let $x_i(k)$ denote the raw value of parameter i at sampling time k, and let the window length be N. The filtered value is

$$y_i(k)=\frac{1}{N}\sum_{j=0}^{N-1} x_i(k-j) \qquad (1)$$

The window length N balances smoothness and response speed. A large N produces a smoother curve but delays the response to a sudden load change; a small N preserves responsiveness but weakens noise suppression. As an initial engineering setting, N = 10 is suggested for temperature and N = 5 for current. These values should be compared under empty-load, light-load, and repeated-positioning conditions during the validation stage.

C. Normalized Deviation and Weighted Health Index

Because the selected variables have different units and ranges, they must be normalized before aggregation. Let U_i and L_i denote the upper and lower safe limits for parameter i, and let y_i be the filtered value. A normalized deviation is defined as

$$d_i = clip[(y_i - L_i)/(U_i - L_i), 0, 1] \quad (2)$$

Here clip(.,0,1) restricts the result to the interval [0,1]. For temperature and current, where larger values generally indicate greater risk, U_i may be taken from the manufacturer or maintenance specification and L_i from a normal lower bound. Enumeration states such as operating mode are displayed separately rather than forced into the continuous health index.

A weighted health index is then constructed as

$$H(k) = 1 - \sum_{i=1}^{n} w_i d_i(k),\ \sum_{i=1}^{n} w_i = 1,\ w_i \geq 0 \quad (3)$$

where n is the number of continuous parameters participating in health evaluation and w_i is the weight of parameter i. A value of H(k) close to 1 indicates that the monitored variables remain close to their normal intervals. A persistent decline below a preset level triggers a warning. Weights can be determined from manufacturer recommendations, maintenance records, teaching emphasis, and expert review; historical failure data should be used to calibrate them when available.

D. Alarm Judgment and Persistence

To reduce false alarms caused by instantaneous disturbance, the system combines threshold crossing with a persistence condition. Let D_i be the normalized alarm threshold, T_i the required duration, tau the sampling period, and $m_i(k)$ the number of consecutive out-of-limit samples. An alarm is raised when

$$Alarm_i = 1, if\ d_i(k) \geq D_i\ and\ m_i(k) \geq \lceil T_i/\tau \rceil \quad (4)$$

The ceiling function ⌈ ⌉ converts the required duration into the minimum number of consecutive samples. This design tolerates short noise while responding to persistent abnormality. In practice, the system should retain both raw and filtered values so that a preprocessing error cannot hide a genuine anomaly.

E. Protocol Selection

Table 2 compares the roles of Modbus TCP and MQTT in this system. They are not replacements for each other; each is assigned to the layer where its characteristics are most appropriate.

TABLE 2
Applicability Comparison Of Modbus Tcp And Mqtt In This System

Dimension	Modbus TCP	MQTT
Primary role	Industrial field-device communication	Lightweight publish-subscribe messaging
Communication model	Master-slave request/response	Broker-mediated publish/subscribe
Data representation	Register and coil addresses	Topic plus JSON payload
Timing behavior	Cyclic polling; relatively predictable	Lightweight; depends on QoS and network
Coupling	Strongly coupled to device addresses	Producers and consumers are decoupled
Suitable link	Robot controller to PLC	PLC/edge side to Node-RED
Extension	Each device requires address mapping	A new consumer can subscribe to a topic

For MQTT transmission, core health parameters use QoS 1 to obtain at-least-once delivery, while less-critical process parameters may use QoS 0 to reduce acknowledgment overhead. Every message contains a timestamp, device identifier, parameter code, numerical value, unit where applicable, and status code. These fields make later tracing and classroom explanation possible.

VI. NODE-RED VISUALIZATION SYSTEM IMPLEMENTATION

A. Flow Design

Fig. 2 shows the core Node-RED processing flow. An MQTT input node subscribes to the robot-state topic. A JSON node converts the text payload into an object. A Function node extracts fields, checks value validity, converts units where necessary, and adds business flags. A Switch node routes data according to parameter type. Chart nodes draw temperature curves, a Gauge node displays motor current, and Text or Icon nodes show operating mode and IO state. SQLite nodes store key events and periodic records, while Notification nodes issue alarm prompts.

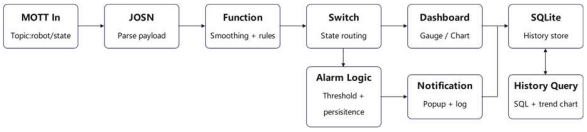

Fig. 2 Node-RED low-code data-processing pipeline

Compared with a completely custom implementation, the flow has clearer node responsibilities. A teacher can explain it in groups: the subscription layer answers where the data come from; the parsing layer answers what the fields mean; the logic layer answers how an abnormal state is identified; and the presentation layer answers how results are displayed. Students can modify node parameters under guidance and immediately observe the effect, which is difficult in a compiled upper-computer project.

B. Data Structure and Historical Storage

SQLite is selected for local historical storage because it is simple to deploy, does not require an independent database server, and is suitable for a single-machine teaching environment. The historical table should contain at least device ID, parameter code, value, unit, status code, acquisition time, and write time. To support retrieval over the previous thirty days at hourly or daily granularity, SQL aggregation can be performed during query time instead of pre-materializing every possible aggregate table.

Historical records also support classroom cases. For example, a current curve from a repeated-positioning experiment can be exported and analyzed in relation to load changes, speed settings, acceleration, and trajectory smoothness. This use of real laboratory data is more persuasive than an isolated explanation of current or load concepts.

C. Visualization Interface Structure

Fig. 3 maps the state-evaluation model to dashboard components, and Fig. 4 shows the dashboard layout. The interface is divided into a health-monitoring area, operation-monitoring area, alarm-log area, historical-query area, and teaching-explanation area. The health area emphasizes temperature trends and threshold lines. The operation area displays current load, operating mode, and IO status. The alarm area records time, parameter, exceeded value, duration, and handling state. The historical area supports time-range and granularity selection. The teaching area explains the meaning of each parameter and the typical risk it reflects.

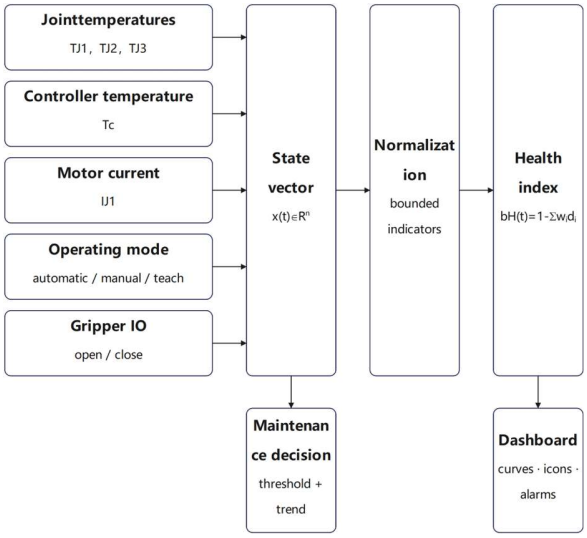

Fig. 3 State-evaluation model and its mapping to visualization components

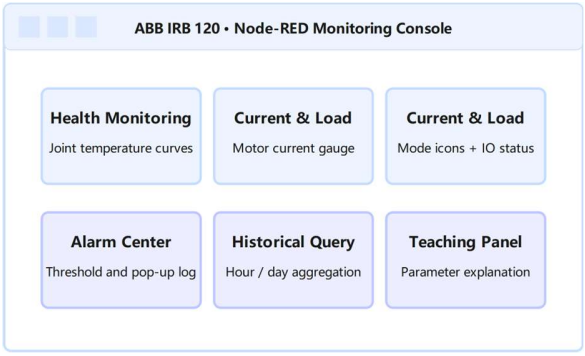

Fig. 4 Lightweight dashboard layout for industrial-robot state monitoring

The interface follows the principle of maintenance first and teaching enhancement. Maintenance users need to identify anomalies quickly, so key indicators use prominent color and readable fonts. Students need to understand meaning, so each region includes a concise explanation. Alarm information does not display only an abstract code; it also shows parameter name, current value, threshold, duration, and suggested action. This structure helps users move from observation to diagnosis.

VII. PERFORMANCE TARGETS AND TEST PLAN

A. Design Targets

Table 3 lists the design targets to be verified in the project. These values originate from the project research objectives. They are not presented as completed experimental measurements. The validation plan is organized around these targets so that later implementation can produce auditable evidence.

Table 3
System Design Targets And Proposed Verification Methods

Category	Indicator	Design target	Test method
Timeliness	Data-update delay	< 1 s	Compare PLC send time with dashboard refresh time
Timeliness	Alarm response time	≤ 1 s	Inject a simulated out-of-limit value and measure popup time
Interaction	Interface operation response	< 0.5 s	Click queries and page switches and record response time

Category	Indicator	Design target	Test method
Reliability	Parameter acquisition accuracy	≥ 98%	Compare controller-indicated values with system records
Resource use	PLC resource occupancy	≤ 30%	Record CPU and communication load using PLC diagnostics
Stability	Continuous test duration	72 h	Run empty-load, light-load grasping, and repeated positioning
Traceability	Historical data retention	Last 30 days	Query hourly/daily aggregates and verify record integrity

B. Test Conditions

Three typical operating conditions are designed: empty-load operation, light-load grasping, and repeated positioning. Empty-load operation provides baseline temperature rise and current fluctuation. Light-load grasping verifies the correlation between gripper IO events and current variation. Repeated positioning exposes periodic load behavior and steady temperature trends. Each condition should be recorded continuously, and different MQTT QoS configurations should be compared with respect to delay, message count, and interface fluency.

C. Risks and Controls

Three categories of risk require control. First, field safety: emergency stop, speed limitation, workspace separation, and teacher supervision must be maintained; the monitoring system must never be used to override robot safety functions. Second, data risk: raw values and filtered values should be recorded in parallel so preprocessing does not conceal genuine anomalies. Third, network risk: network state and message sequence should be retained so that system delay can be distinguished from packet loss or broker congestion.

VIII. TEACHING APPLICATION DESIGN

A. Conversion into Courses

The system can support courses such as Industrial Robot Operation and Maintenance, PLC Application Technology, Industrial Network and Configuration Technology, and Industrial Data Analysis. In a robot-maintenance course, teachers can use temperature thresholds, motor load, and health-index trends to introduce preventive maintenance. In a PLC course, students can examine Modbus TCP master configuration, register conversion, and logical judgment. In a network course, the system provides a concrete example of why Modbus TCP and MQTT occupy different layers and how JSON payloads bridge them.

B. Training Task Design

Table 4 outlines three teaching cases arranged from observation to configuration to analysis. They can be adapted to different programs and course hours.

Table 4
Teaching-application cases based on the monitoring system

Case	Related course	Task	Learning outcome
State observation and explanation	Industrial robot maintenance	Observe temperature, current, and mode changes; explain parameter relationships	State identification and safety awareness
Communication-chain configuration	PLC application technology	Configure Modbus TCP reading and MQTT topics; complete parameter mapping	Industrial communication and troubleshooting
Health-data analysis	Industrial data analysis / maintenance	Export historical data; calculate deviation and health index; propose maintenance advice	Data processing and evidence-based decision making

C. Mechanism for Research to Support Teaching

Research supports teaching most effectively when the engineering process is converted into actionable learning tasks. The project team can organize the parameter-screening basis, communication-

configuration manual, test report, alarm rules, database structure, and fault cases into modular teaching resources. Students can participate in data organization, interface optimization, documentation, and simple Function-node development. Through these tasks, they learn not only technical facts but also the constraints, trade-offs, and validation habits of engineering design.

IX. DISCUSSION

A. Advantages

The proposed system is low-cost, low-threshold, and teaching-friendly. Node-RED, Mosquitto, and SQLite can be deployed locally without a large cloud platform or dedicated application server. The flow-based environment lowers implementation difficulty and allows a teaching team to iterate quickly. More importantly, industrial data are transformed into visual teaching resources: students can observe state changes that were previously hidden inside the controller.

B. Limitations and Improvements

The design also has limitations. Node-RED is not intended for hard-real-time motion control; its role here is monitoring and analysis. Low-code nodes are highly encapsulated, and complex algorithms may still require Function scripts or external services. The weights of the health index initially depend on expert judgment and manufacturer recommendations and must be calibrated with accumulated data. The current project focuses on one robot type; extension to other brands requires an additional protocol-adaptation layer, for example a FANUC FOCAS interface for an LR Mate 200iD. Security also requires further work, including user authentication, operation rights, network isolation, and backup policy.

C. Prospects for Wider Use

From the perspective of teaching organization, a robot laboratory usually serves multiple classes, projects, and teachers. Without unified state records, equipment failure may be reconstructed only from oral description, making it difficult to know the temperature, current, mode, and IO state at the time of the event. With this system, laboratory managers can retain continuous records, teachers can select representative segments as cases, and students can discuss and reproduce analysis using the same data. One acquisition process can therefore support maintenance decisions, classroom explanation, skill training, and project evaluation.

From an engineering perspective, the scheme can be connected with laboratory safety management. If controller temperature remains in a warning interval, the system can prompt inspection of cabinet filters and room ventilation. If J1 current is notably higher than the historical level for the same trajectory, it can prompt inspection of the reducer, end-effector load, or trajectory smoothness. If gripper IO state disagrees with program expectation, it can prompt checking of air pressure, gripper sensors, or signal wiring. Such rules turn isolated curves into actionable maintenance actions.

From a program-development perspective, the system can demonstrate the principle that technical implementation becomes a teaching resource. Parameter tables, communication manuals, alarm rules, database structures, interface prototypes, and test reports can be packaged as reusable modules for robotics, mechatronics, and electrical-automation programs. After completing basic tasks, students can extend the system toward interface optimization, data compression, anomaly detection, or multi-device access, forming a progression from course training to skill competition and research practice.

Nevertheless, low code must not be interpreted as meaning that no professional design is required. Node-RED reduces the difficulty of message-flow and interface construction, but the system still needs rigorous parameter definition, address mapping, data validation, exception handling, and security isolation. When a real industrial device is connected, every acquisition change must be checked against manufacturer documentation, tested under supervision, and reviewed by qualified teachers. The monitoring system must not casually modify control parameters or bypass safety circuits.

X. CONCLUSION

This paper has presented a Node-RED-based key-state acquisition and visualization system for an industrial robot in a vocational-college laboratory.

The work defines a robot-PLC-Node-RED architecture, screens six categories of key parameters, formulates moving-average filtering, normalized deviation, weighted health indexing, and persistence-based alarm logic, and designs a lightweight dashboard and historical-query mechanism. Attribute tables, protocol comparison, design targets, and teaching cases provide a reproducible implementation path.

Three conclusions follow from the design. First, combining Modbus TCP and MQTT separates field acquisition from upper-layer visualization and is suitable for a small teaching laboratory. Second, Node-RED's flow-based development exposes the data path, lowering both implementation and explanation barriers. Third, parameter mapping, curves, state indicators, alarm logs, and teaching explanations can convert robot operating states into interpretable maintenance and learning resources. Future work will conduct actual multi-condition tests, calibrate weights, add multi-protocol adaptation, improve security and permissions, and evaluate the system's teaching effectiveness.

REFERENCES

[1] Ren G ,Wang Z ,Liu X , et al.Remaining useful life prediction of industrial robot RV reducer with multiple deep networks and multicore support vector data description[J].Journal of Mechanical Science and Technology,2024,(prepublish):1-15.DOI:10.1007/S12206-024-0703-Y.

[2] Mahmood K ,Otto T ,Riives J , et al.Simulation based feasibility analysis of autonomously movable robot arm[J].Proceedings of the Estonian Academy of Sciences,2021,70(4):422-428.DOI:10.3176/PROC.2021.4.08.

[3] Suling T ,Mingying C ,Yuanyuan L .Research on the Trajectory Planning and Control Technology of Industrial Robots Guided by Computer Visualization[J].Journal of Physics: Conference Series,2021,2074(1):DOI:10.1088/1742-6596/2074/1/012040.

[4] Li Y ,Yijing L ,Hong Z , et al.Research on Key Technology of State Evaluation and Replacement Acquisition System of Intelligent Electric Energy Meter Based on Smart Grid[J].IOP Conference Series: Earth and Environmental Science,2020,558(5):052053-.DOI:10.1088/1755-1315/558/5/052053.

[5] Sensor Research; Recent Studies from State Key Laboratory Add New Data to Sensor Research (Study on Energy Acquisition Scheme and Parameter Optimization for Double-Insulated Ground Wire of Overhead Transmission Lines)[J].Journal of Technology,2020,

[6] Zheng W ,Li H ,Lam K H , et al.A novel rapidly-exploring random tree algorithm with dynamic goal biasing and position-constrained sampling based on equal-interval nodes and cost optimization: application to mobile robots[J].Engineering Applications of Artificial Intelligence,2026,167(P1):113612-113612.DOI:10.1016/J.ENGAPPAI.2025.113612.

[7] Zhao Y ,Mao X ,Yang Y .An Effective Method for Constructing a Robot Operating System Node Knowledge Graph Based on Open-Source Robotics Repositories[J].Electronics,2023,12(19):DOI:10.3390/ELECTRONICS12194022.

[8] Suling T ,Mingying C ,Yuanyuan L .Research on the Trajectory Planning and Control Technology of Industrial Robots Guided by Computer Visualization[J].Journal of Physics: Conference Series,2021,2074(1):DOI:10.1088/1742-6596/2074/1/012040.

[9] Aiko T ,Hayashi E ,Prem G .The Research of an Augmented Reality System for the Implementation of Industrial Robots[J].Journal of Robotics, Networking and Artificial Life,2026,11(3):185-189.DOI:10.57417/JRNAL.11.3_185.

[10] Robotics; Research from Southwest Jiaotong University Provides New Data on Robotics (A Workspace Visualization Method for a Multijoint Industrial Robot Based on the 3D-Printing Layering Concept)[J].Robotics & Machine Learning,2020,2733-.

