2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)

Development of an Industrial Robot Condition Monitoring System Platform for Smart Manufacturing

First Author: Yanjun Ma[1,a]
Unit: [1]Hudong-Zhonghua Shipbuilding (Group) Co., Ltd
Shanghai 200000, China
*Corresponding author:kfmyj@163.com

Second Author: Lei Yang [2,b]
Unit: [2]Shanghai Maritime University
Shanghai 200000, China
[b]e-mail:13283809960@163.com

***Abstract*-To address the issue of industrial robot condition monitoring in smart manufacturing processes, this study developed a condition monitoring system for industrial robots based on a B/S (Browser/Server) architecture. Utilizing a front-end and back-end separation approach, the system incorporates functionalities such as user management, visualization dashboards, and device management. Additionally, a health assessment module for joint bearings was developed. By employing technologies such as Vue.js and Node.js, the system mitigates the challenges posed by large data volumes, ensuring real-time monitoring and evaluation of the health status of industrial robot joint bearings. This provides an effective tool for equipment maintenance and management.**

Keywords-Industrial robots; Condition monitoring; B/S architecture; Joint bearings

I. INTRODUCTION

With the rapid advancement of smart manufacturing technologies, industrial robots are increasingly utilized on production lines. However, the operational status of industrial robots directly impacts production efficiency and product quality, making real-time monitoring and evaluation of their condition particularly critical. Existing industrial robot condition monitoring systems primarily adopt a Client/Server (C/S) architecture. While this approach can meet basic monitoring requirements, it presents limitations in terms of cross-platform compatibility, deployment and update processes, and cost control. To address these challenges, this study analyzes the specific requirements for industrial robot condition monitoring and designs a system platform based on a Browser/Server (B/S) architecture. By employing a front-end and back-end separation development model, the system enhances scalability and maintainability. The B/S architecture, with its advantages of cross-platform compatibility, ease of deployment and updates, cost efficiency, and ease of access and sharing, offers a more flexible and efficient solution for monitoring the condition of industrial robots.

Industrial robot condition monitoring technology plays a crucial role in improving production efficiency and reliability, ensuring production safety, reducing maintenance costs, and optimizing maintenance schedules. Currently, condition monitoring for industrial robots can be categorized into component-level monitoring and system-level monitoring.Zhang Zhichao[1] developed a cloud-based B/S architecture for remote monitoring and analysis of industrial robots. Using web technologies, real-time monitoring was achieved alongside data algorithm analysis with Python, enhancing production efficiency.Gao Wen[2] implemented a monitoring and warning software system with an intuitive human-machine interface by leveraging C# language, the .NET6 framework, OPC UA protocol, 3D modeling, and machine vision algorithms.Jin Lingyan[3] built a UR10 virtual prototype system using Matlab-Simulink. Through TCP/IP communication protocol, data acquisition of robo tic arms was achieved.Xu Jianming and Pan Xiangfei[4] designed a remote real-time monitoring system for robots using a J2EE architecture and WebSocket communication protocol.Hong Huiwu[5] developed a monitoring platform based on a B/S architecture. The back-end employed a J2EE platform and SSH framework deployed on Tomcat7 servers, while the front-end utilized the Vue framework. Yin Qingsong[6] analyzed the requirements of condition monitoring systems and designed hardware architecture based on ZYNQ using piezoelectric integrated circuit sensors as examples.Liu, X[7] studied the characteristics of acoustic emission signals and their relationship with robotic reducer components, enabling reducer monitoring without dismantling the robotic arm.Ayankoso S[8] uses multi-layer perceptron, CNN, and sparse autoencoder for fault detection in collaborative robots, achieving over 93% accuracy and 96% F1 score, and proposes an integrated method combining trajectory change and anomaly detection to improve online monitoring reliability and safety.Kaigom E[9] proposes a digital twin-based approach that integrates robot models and deep learning to advance fault prediction in collaborative robots.Giacomo N[10] proposes a ROS-based condition monitoring architecture for collaborative robots, which quickly detects faults through data labeling and indexing, and defines health indicators to detect joint anomalies using torque information, enhancing robot reliability.

This study focuses on the architecture design, functional development, and practical application evaluation of an industrial robot condition monitoring system. Comprehensive testing and analysis of the system platform confirm its feasibility and effectiveness in actual production environments. The research results not only provide a novel solution for industrial robot condition monitoring but also offer strong support for the development and application of smart manufacturing technologies.

II. SYSTEM ARCHITECTURE

A. Development Environment

The condition monitoring system platform is based on a B/S architecture and adopts a front-end and back-end separation model. The specific development environment is

979-8-3315-0976-7/25/$31.00 ©2025 IEEE

outlined in Table 1 below.

Table 1 System Development Environment

Type	Specific Selection
Development Platform OS	Windows10（x64）
Development Tools	VScode
Programming Language	JavaScript
Database	MySQL
Server	Node.js -v18.16.0
Framework/Library	Vue.js -v2.7.8
Script Files	Python -v3.11.3
Automation Build Tool	Webpack

B. Software System Architecture

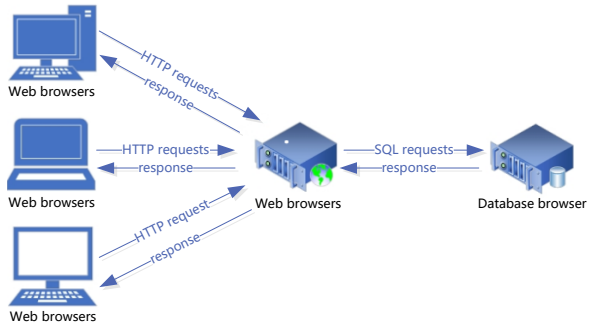


Figure 1. B/S Architecture Diagram

Figure 2. MVVM Architecture

The B/S architecture is illustrated in Figure 1. The system's front-end adopts the MVVM (Model-View-ViewModel) pattern, as shown in Figure 2. This pattern divides the application into three main components: Model, View, and ViewModel. Utilizing the Vue.js framework, the MVVM pattern, depicted in Figure 3, enables two-way data binding between the data layer and the view layer, thereby simplifying the development process and improving efficiency.

The front-end technology stack includes Vue.js, Vue-router, Vuex, Axios, Element-UI, ECharts, and Three.js, all of which collectively contribute to the development of the web application:

1.Vue.js: Serves as the core framework, simplifying DOM manipulation through declarative rendering.

2.Vue-router: Optimizes page navigation and enables seamless single-page application (SPA) routing.

3.Vuex: Manages the application state and facilitates data sharing across components.

4.Axios: Acts as an HTTP client to streamline data interaction between the front-end and back-end.

5.Element-UI: Provides a comprehensive UI component library for rapid interface development.

6.ECharts: Implements advanced data visualization for interactive dashboards.

7.Three.js: Renders 3D graphics, enhancing the user interaction experience.

Additionally, Webpack is used for resource bundling, supporting code splitting and hot updates to boost performance and maintainability.

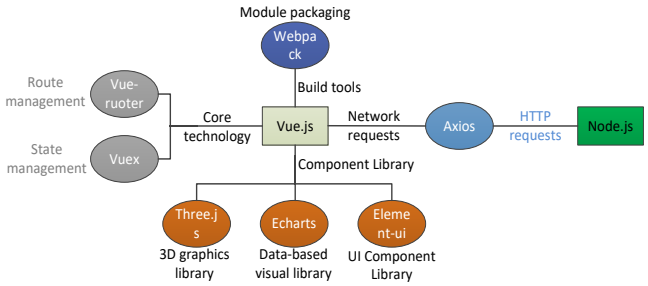


Figure 3. Vue.js Technology Architecture Diagram

In the back-end, Node.js is chosen as the server runtime environment, and API interfaces are built based on the Express framework. Node.js, with its event-driven and non-blocking I/O design philosophy, can efficiently handle a large number of concurrent requests, making it well-suited for building high-performance web applications. The middleware mechanism of the Express framework, as shown in Figure 4, allows developers to process HTTP requests and responses, implementing functions such as authentication, logging, and error handling, thereby enhancing the flexibility and maintainability of the system.

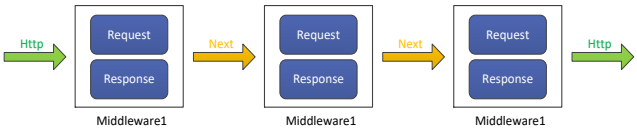


Figure 4. Express Middleware Mechanism

As shown in Figure 5, in terms of data flow, the system collects data from underlying devices via sensors and stores it in a MySQL database. When users send data requests through the browser, the web front-end uses HTTP Axios to send corresponding requests to the back-end. The back-end server executes the required logical operations based on the requests, performing CRUD (Create, Read, Update, Delete) operations on the database or invoking relevant Python programs.

Data exchange between the front-end and back-end uses the JSON format, ensuring reliable data transmission through HTTP clients like Axios and server-side HTTP frameworks. The Node.js environment employs the Python-Shell module to execute Python scripts, enabling interaction between the front-end, back-end, and script files, thereby providing robust support for data processing.

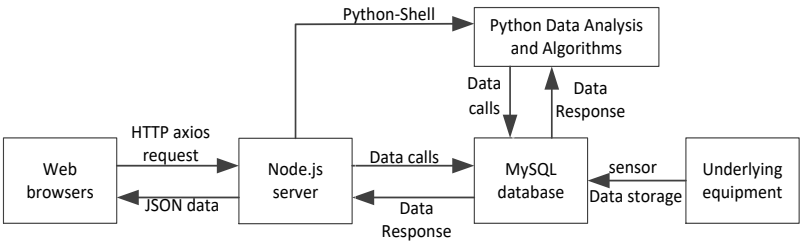


Figure 5. Data Flow Diagram

III. System Platform Functional Design and Implementation

A. Interactive System Development

In the development of the industrial robot condition monitoring system platform, the interactive system is a key element for enhancing user experience and operational convenience. The platform provides an intuitive user interface that allows users to access the system through a straightforward registration and login page. Upon logging in, users are directed to the production visualization platform, which integrates real-time production data, including current time, daily processing status, inventory information, alarm notifications, equipment operation status, and equipment utilization rates, offering operators a comprehensive production monitoring view.

The system's user management functionality enables administrators to manage user accounts, including setting permission levels and account statuses, as illustrated in Figure 6. The production visualization platform displays information such as current time, daily processing status, inventory details, alarm notifications, equipment operation status, and equipment utilization rates, as shown in Figure 7.

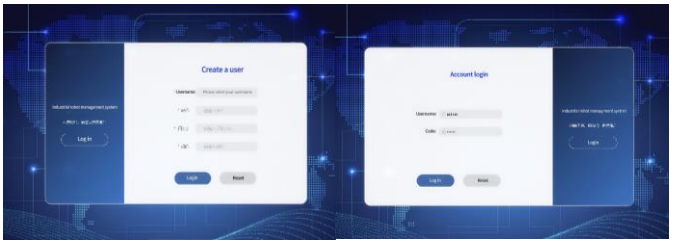


Figure 6. Registration and Login Interface

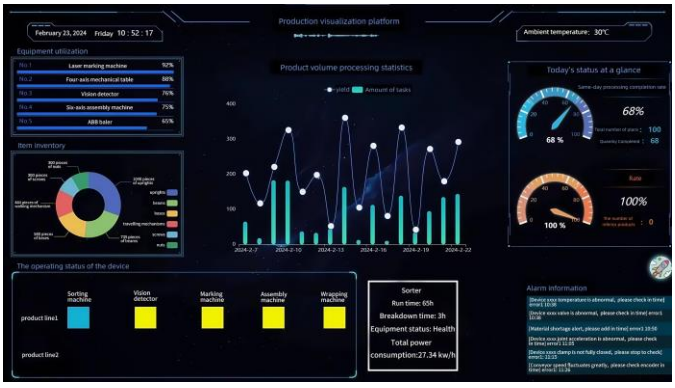


Figure 7. Visualization Dashboard

The permission system is designed with multiple levels to ensure system security and flexibility, where the SuperAdmin (super administrator) holds the highest level of permissions and can manage the permissions of other users, as shown in Figure 8.

The file management module supports uploading, downloading, browsing, and deleting operations, facilitating the management of industrial robot-related documents and files, as illustrated in Figure 9.

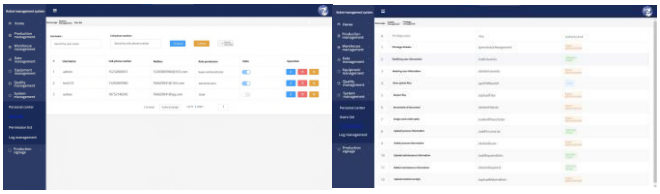


Figure 8. User Management and Permissions List

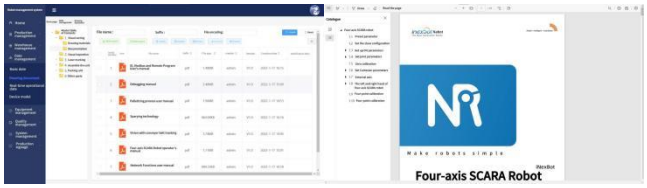


Figure 9. File Management and File Browsing

Additionally, the maintenance management module allows users to edit and manage equipment maintenance information, including creating, editing, deleting, and exporting maintenance records, as shown in Figure 10. Overall, the development of the interactive system focuses on providing a comprehensive and user-friendly platform to meet the demands of industrial environments for equipment monitoring and management.

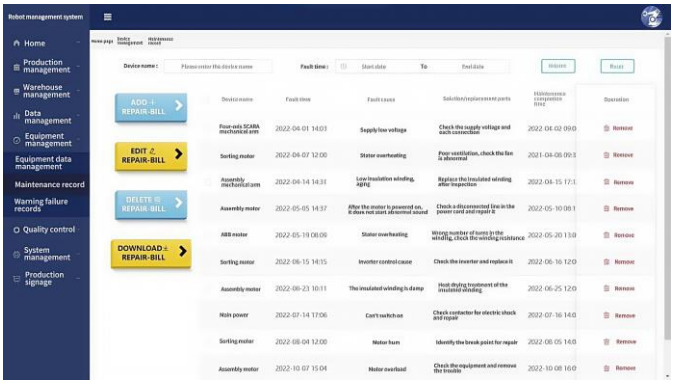


Figure 10. Maintenance Management

Figure 10 demonstrates the maintenance management interface, which enables users to create, edit, delete, and export equipment maintenance records, providing comprehensive support for maintenance operations. In the interactive system, device information is retrieved and displayed from the Device_Information data source. Equipment models are rendered using the Three.js engine, optimizing the loading and rendering efficiency of 3D models, as shown in Figure 11.Real-time data management is implemented using a WebSocket long-polling mechanism, ensuring dynamic updates of data. Combined with ECharts for data visualization, this enhances users ability to monitor and analyze equipment status in real time, as illustrated in Figure 12.

a. Device Information Management b. Device Model Management

Figure 11. Device Information and Model Management

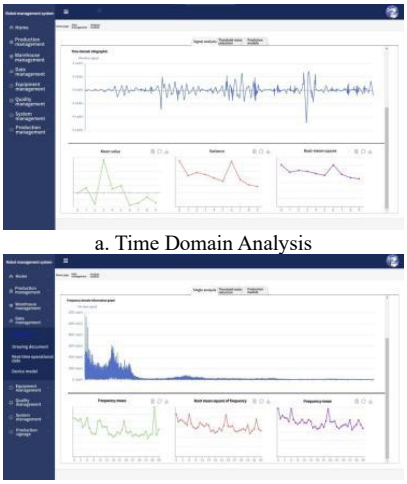
Figure 12. Real-Time Data Management

B. Development of Joint Bearing Health Assessment System

Joint bearings are critical moving components of industrial robots, and their health directly impacts the performance and lifespan of the robots. The developed joint bearing health assessment system enables real-time monitoring and evaluation of joint bearing health based on vibration signal analysis.

The system first collects vibration signals from joint bearings using sensors and performs analyses in the time domain, frequency domain, and time-frequency domain. Users can configure parameters, such as selecting the analysis time range, device information, and the desired features to extract. The system utilizes tools like ECharts to visually display analysis results in the form of charts, making it easier for users to understand and analyze the data, as shown in Figures 13 and 14.

To address the noise present in vibration signals, the system offers three denoising methods: hard thresholding, soft thresholding, and an improved wavelet thresholding technique. Users can configure parameters such as threshold selection methods, signal dimensions, decomposition levels, and wavelet basis functions to perform denoising on vibration signals, thereby improving the accuracy of signal analysis. Figure 15 shows the denoising effect achieved using the improved wavelet thresholding technique.



a. Time Domain Analysis

b. Frequency Domain Analysis

Figure 13. Time Domain and Frequency Domain Analysis

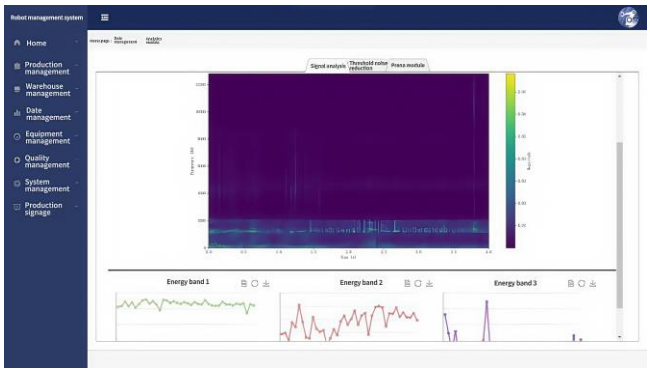


Figure 14. Time-Frequency Domain Analysis

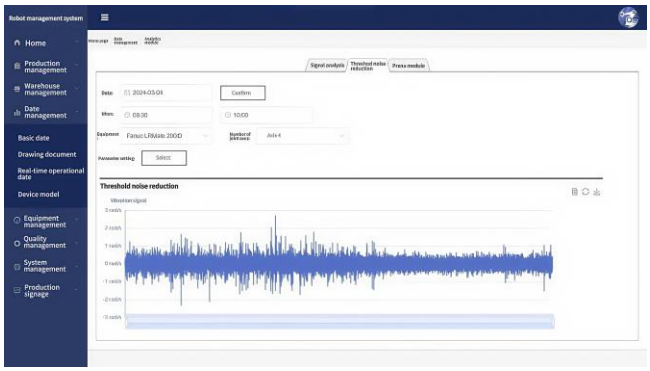


Figure 15. Improved Wavelet Thresholding Denoising

The system uses long-term monitored vibration data to build a joint bearing health assessment model with machine learning algorithms. Users can select pre-trained assessment models and historical monitoring data of the bearings to evaluate the current health status of the bearings. The evaluation results are presented in a visualized format, allowing users to intuitively understand the health condition of the bearings, as shown in Figure 16.

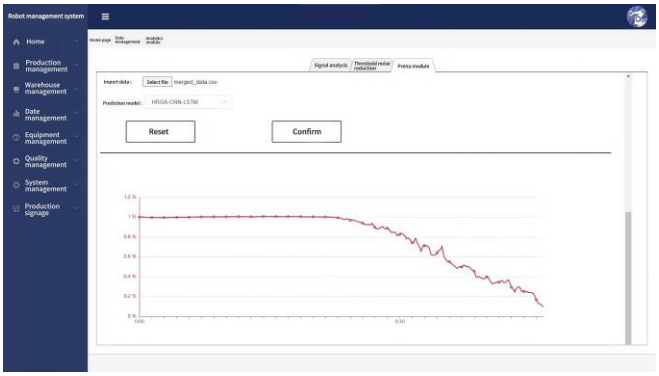


Figure 16. Visualization of Joint Bearing Health Assessment

During the system development process, the efficiency of data transmission and processing was fully considered. To address the issue of large vibration signal data volumes, the system employs data compression and batch processing mechanisms. Data is divided into smaller batches for processing, significantly reducing the bandwidth required for data transmission, alleviating system load, and enhancing concurrency and processing efficiency.

For data transmission, the system uses streaming transmission, avoiding the resource waste and performance bottlenecks caused by loading all data into memory at once. This further improves data transmission efficiency.

The system leverages the Python-Shell module in Node.js to enable interaction between the Node.js environment and Python scripts. Python scripts handle complex signal processing, thresholding denoising, and health assessment algorithms, returning the results to the Node.js environment. The Node.js environment then forwards the results to the front-end for display.

To enhance the intelligence of the joint bearing health assessment system, fault diagnosis and prediction functions are further integrated. By incorporating vibration, temperature, and acoustic sensors, the system can continuously monitor the bearing's operating condition and capture early signs of potential faults. The fault diagnosis module uses signal processing techniques to extract time-domain, frequency-domain, and time-frequency domain features, combined with machine learning algorithms such as Support Vector Machines (SVM) and Random Forests, to identify various fault patterns. The fault prediction module utilizes time-series analysis and deep learning models like LSTM to model historical data and predict the Remaining Useful Life (RUL) of the bearing and the likelihood of fault occurrence. By combining health assessment results, the system can anticipate potential fault risks, provide maintenance alerts, and help optimize maintenance decisions. Through the intelligent integration of fault diagnosis and prediction, the system can more accurately assess bearing health, extend equipment lifespan, and effectively reduce production downtime.

This system achieves real-time monitoring and evaluation of the health status of industrial robot joint bearings, providing an effective tool for the maintenance and management of industrial robot equipment, demonstrating significant application value.

IV. Conclusion

This paper addresses the issue of monitoring the health status of industrial robot joint bearings and develops a condition monitoring system based on a B/S architecture. The system adopts a front-end and back-end separation model, with the front-end built using the Vue.js framework and the back-end developed using the Node.js-based Express framework. Equipment data and user information are stored in a MySQL database.

The system implements various functional modules, including user management, visualization dashboards, device management, maintenance management, file management, and real-time data management. It conducts time-domain, frequency-domain, and time-frequency domain analyses of joint bearing vibration signals, as well as thresholding denoising and health assessment.

During the development process, data transmission and processing efficiency were fully considered. Techniques such as data compression, batch processing, and streaming transmission were employed to effectively address the system load issues caused by large data volumes.

This system achieves real-time monitoring and evaluation of the health status of industrial robot joint bearings, providing an effective tool for equipment maintenance and management, and holds significant application value.

References:

[1] Zhang Zhichao. Design of industrial robot monitoring and analysis system based on cloud platform[D].Anhui University,2022.

[2] Gao Wen. Design of industrial robot monitoring and early warning system based on NET6[D].Chongqing University of Technology,2023.

[3] Jin Lingyan. Research on Reliability Analysis of Motion Accuracy and Joint Status Monitoring of Industrial Robots[D]. Zhejiang Sci-Tech University, 2021.

[4] Xu Jianming,Pan Xiangfei. Research on Monitoring System of Industrial Robot Based on Socket Communication [J]. Computer Measurement and Control, 2017, 25 (07): 70-73.

[5] Hong Huiwu. Research and design of remote monitoring and diagnosis system for industrial robots[D]. Jiangnan University, 2021.

[6] Yin Qingsong. Design of hardware system for state signal acquisition of industrial robots[D]. Anhui University, 2022.

[7] Liu X , Wu X , Liu C ,et al.Research on condition monitoring of speed reducer of industrial robot with acoustic emission[J].Transactions-Canadian Society for Mechanical Engineering, 2016, 40(5): 1041-1049. DOI:10.1139/tcsme-2016-0086.

[8] Ayankoso S ,Gu F ,Louadah H , et al.Artificial-Intelligence-Based Condition Monitoring of Industrial Collaborative Robots: Detecting Anomalies and Adapting to Trajectory Changes[J]. Machines, 2024, 12(9): 630-630.

[9] Ayankoso S ,Kaigom E ,Louadah H , et al.A Hybrid Digital Twin Scheme for the Condition Monitoring of Industrial Collaborative Robots[J].Procedia Computer Science,2024,2321099-1108.

[10] Giacomo N ,Sauro L ,Andrea B .ROS-Based Condition Monitoring Architecture Enabling Automatic Faults Detection in Industrial Collaborative Robots[J].Applied Sciences,2022,13(1):143-143.