JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 38, XXX-XXX (2022)
DOI: 10.6688/JISE.20220X_38(X).00XX

Exploring the Application of Cloud Computing and Information Security in Financial Robot Software Architecture and Self-Localization Algorithms

PING-LI[1], LI-MIN ZHANG [2,*] AND XI-BEI HUANG[1]
[1]*School of Information and Mechatronic Engineering, Hunan International Economics University, Changsha,410205, China; lip290520305@163.com ; touchangel@outlook.com*
[2]*College of Electrical and Information Engineering, Hunan Institute of Traffic Engineering, Hengyang 421001, Hunan, China*
**Corresponding author: Limin Zhang, e-mail: zlmin2024@163.com*

The meteoric rise of cloud computing and information security technologies has markedly heightened the importance of financial robots, particularly in ensuring security and precision. This work embarks on designing a sophisticated financial robot software architecture, grounded in cloud computing principles, to bolster system security and operational efficiency. At the heart of this design lies a self-localization algorithm grounded in multi-sensor information fusion (MSIF), meticulously crafted to tackle localization accuracy challenges inherent in financial robot software. This algorithm harnesses the formidable computational power of cloud computing platforms coupled with the robust safeguards of information security technologies, thereby elevating both performance and security. Expert assessments reveal that the developed financial robot software architecture routinely achieves functionality implementation scores exceeding 8. The system boasts an average task scheduling delay of under 5μs, with 99.83% of transmission delays confined to less than 1.5ms. Furthermore, the information fusion localization algorithm excels in maintaining heading angle error near 0 degrees. Localization errors in both X and Y axes are kept below 0.8 meters, with a notable reduction in errors as iteration counts increase. These findings underscore the efficacy of the proposed solution in markedly enhancing the security and accuracy of financial robot software, addressing software drift issues, and offering a dependable security risk assessment framework for corporate financial management.

Keywords: cloud computing; information security; financial robot; self-localization algorithm; software architecture

1. INTRODUCTION

The continuous advancements in cloud computing and information security technologies have led many enterprises, particularly in the financial sector, to adopt cloud computing for data storage and processing. Financial robot software has become indispensable for these companies, as it enhances work efficiency, reduces error rates, and lowers operational costs [1]. However, constructing efficient and reliable software architecture for financial robot software that enables quick and accurate task execution poses a major challenge. Ensuring data security during data transmission and processing is also crucial to prevent the leakage or compromise of sensitive information [2]. In practical applications, financial robot software is utilized to automate various tasks, including bill management

and report generation. To achieve these functionalities, a self-localization algorithm is employed to determine the location of the financial robot and perform corresponding tasks based on its current position. This aspect is of paramount importance and value in enhancing the efficiency and security of financial management, thereby contributing to the advancement of modern enterprise financial management systems.

Cloud computing plays a vital role in the architecture of financial robot software by providing high-performance computing and storage resources. This technology enables financial robots to effectively handle large-scale financial data and complex computational tasks [3]. When designing the software architecture of financial robots, careful consideration must be given to fully leverage cloud computing resources to achieve task parallel processing and distributed storage, thereby enhancing overall performance and scalability. Given the highly sensitive and valuable nature of financial data, stringent protection measures are essential during storage, transmission, and processing. The software architecture of financial robots should incorporate various security measures, such as data encryption, identity authentication, and access control, to ensure the confidentiality, integrity, and availability of financial data. Self-localization refers to the ability of financial robots to accurately determine their position and environmental state while performing tasks [4]. Self-localization algorithms leverage a wide range of sensors and data sources, including images, sounds, and location information, to locate the robot and perceive its environment. Through precise self-localization algorithms, financial robots can effectively adapt to different financial scenarios and meet diverse task requirements.

The research background of the financial robot software architecture and self-localization algorithm, based on cloud computing and information security, encompasses several key aspects. Firstly, it addresses the transformational demands within the financial industry driven by the need for advanced technologies and solutions. Secondly, it highlights the significance of leveraging cloud computing resources to enhance financial robots' capabilities, enabling them to efficiently handle large-scale financial data and complex computational tasks. Thirdly, it emphasizes the importance of ensuring information security, as financial data is highly sensitive and valuable, requiring robust measures to safeguard its confidentiality, integrity, and availability. The research in this area aims to provide technical support for the development and application of financial robots, thereby contributing to the advancement of financial technology [5]. A primary focus of the research is the design and optimization of the architecture of financial robot software. This involves maximizing the utilization of cloud computing resources to achieve task parallel processing and distributed storage, resulting in improved overall performance, scalability, and reliability of financial robots. Additionally, the research delves into the implementation of effective information security measures within the financial robot software architecture. Techniques such as data encryption, identity authentication, access control, and the identification and remediation of security vulnerabilities are explored to safeguard financial data from unauthorized access and potential threats. The collective efforts in developing cloud computing and information security-based financial robot software are expected to enhance the efficiency, security, and innovation capabilities of financial businesses, empowering them to address contemporary challenges and opportunities in the financial landscape.

The primary objective of this work is to develop a robust, efficient, and intelligent

cloud computing-based financial robot software platform aimed at enhancing modern enterprise financial management in terms of efficiency and security. Delving into the synergy of cloud computing and information security technologies, this work pioneers the design of a novel architecture tailored for financial robot software. At its core, a self-localization algorithm based on multi-sensor information fusion (MSIF) is introduced, addressing prevalent issues of localization accuracy within financial robot software. The innovation lies in leveraging cloud computing to enable the financial robot software to maintain high-efficiency operation even under heavy loads. Concurrently, the integration of information security technologies ensures the security and reliability of data transmission. The proposed self-localization algorithm significantly enhances localization precision by amalgamating data from multiple sensors, with remarkable efficacy in eliminating heading angle errors. This fusion of advanced computational capabilities and robust security measures exemplifies a transformative approach to improving both the operational efficiency and data integrity of financial robots. The research marks a significant stride in resolving the longstanding challenges of localization accuracy, thereby providing a reliable and secure framework for the deployment of financial robot software in demanding environments.

This work presents several significant contributions, starting with the design of a cloud-computing-based architecture for financial robot software. Expert evaluations have validated both the functional implementation and performance aspects of this architecture. Additionally, the introduction of the MSIF self-localization algorithm has led to a substantial improvement in localization accuracy by integrating data from multiple sensors. Comparative experiments against traditional sensor-based localization methods and other filtering algorithms highlight the superior precision and stability of the MSIF approach. These findings not only demonstrate the efficacy of the proposed solutions in enhancing the security and accuracy of financial robot software but also offer a reliable solution for corporate financial management, incorporating robust information security risk assessment capabilities. This research provides crucial theoretical and practical foundations for the future development of financial robot technologies, marking a significant advancement in the field.

This work is structured into five main parts, each addressing different aspects of the study. Section 1 introduces the background and research objectives of the financial robot software architecture and self-localization algorithm, contextualizing the study within the broader field of financial technology and highlighting its specific goals. Section 2 provides a comprehensive overview of the current research landscape and accomplishments related to financial robot software architecture and self-localization algorithms, both domestically and internationally. This section serves as a basis for understanding the existing body of knowledge and identifying potential research gaps. Section 3 outlines the research methods employed in this work, including the design of a cloud computing-based and information security-oriented robot software architecture. Moreover, it introduces the utilization of a self-localization algorithm based on MSIF. Experts are invited to evaluate its functionality and rationality to ensure the robustness and appropriateness of the proposed architecture. Furthermore, various algorithm performances are thoroughly tested and assessed. Subsequently, Section 4 involves conducting extensive tests on the designed software architecture and self-localization algorithm, followed by a detailed analysis of the obtained results. This empirical evaluation provides valuable insights into the performance and effectiveness of the proposed solutions. Finally, Section 5 presents the conclusions drawn from the

study, including an assessment of the prospects, limitations, and potential future research directions. This work establishes a secure, efficient, and intelligent financial robot software platform by investigating and developing cloud computing-based financial robot software architecture and self-localization algorithms. This platform is expected to significantly enhance support and services for modern enterprise financial management, contribute to the advancement of cloud computing and information security technology, and offer valuable insights for enterprise digital transformation.

2. LITERATURE REVIEW

Currently, research on cloud computing-based and information security-oriented financial robot software architecture and self-localization algorithms is gaining momentum and making significant progress. With the increasing demand and adoption of financial technology, these research outcomes contribute to the enrichment of theory and practice in the field of financial technology and hold great promise for future advancements. The study of cloud computing-based and information security-oriented financial robot software architecture and self-localization algorithms is expected to bring further innovation and advancements to the financial industry. Furthermore, the field can be further developed through international exchanges and collaborations, fostering the advancement of financial robot technology and facilitating its innovative applications.

Ding et al. (2022) focused on the design and optimization of financial robot software architecture to suit the cloud computing environment. Their work revolved around distributed computing, task scheduling, and resource management, aiming to enhance the performance and scalability of financial robot software [6]. In parallel, other studies have addressed fault-tolerant mechanisms and high availability design within the architecture to ensure the stable operation of the financial robot software. In the domain of information security, domestic researchers have also begun to address security requirements and technical measures within the financial robot software architecture. Their research directions encompass data encryption, identity authentication, access control, security vulnerability analysis, and remediation [7, 8]. Additionally, some studies have focused on secure mechanisms for privacy protection and data sharing, aimed at fulfilling compliance and security requirements for financial data processing [9].

In the domain of self-localization algorithms for financial robots, domestic researchers have initiated investigations into the integration of multiple sensors and data sources to achieve localization and environmental perception for financial robots [10]. The research encompasses various areas such as image processing, speech recognition, and localization techniques, with the goal of enhancing the perception and decision-making capabilities of financial robots to adapt to diverse financial scenarios and task requirements [11]. Furthermore, some domestic research institutions and financial organizations have taken practical steps in applying cloud-based and information security-oriented financial robot software architecture and self-localization algorithms in real-world scenarios. These applications span fields such as financial data processing, intelligent customer service, and risk management [12]. Through practical application verification, researchers can assess the performance and effectiveness of the proposed architecture and algorithms in real-world environments.

Foreign researchers have conducted comprehensive studies on financial robot software architecture, with a particular focus on cutting-edge technologies such as distributed computing, fault tolerance mechanisms, resource management, and micro-services architecture. These technologies aim to achieve high performance, scalability, and resilience in financial robot software [13]. Additionally, some studies explore the application of emerging technologies, such as containerization and serverless computing, in the design of financial robot architecture [14]. In the context of information security, researchers attach significant importance to addressing issues in financial robot software architecture and propose a range of solutions. These encompass data privacy protection, identity authentication, access control, encryption algorithms, and other security technologies to ensure the safety and compliance of financial data [15]. Furthermore, research is focused on vulnerability analysis and security testing of financial robot software to detect and rectify potential security risks [16, 17].

Researchers from various fields have been actively applying machine learning, computer vision, deep learning, and other advanced technologies to study the perception and localization capabilities of financial robots. Their research objectives include image recognition, speech recognition, and natural language processing, aiming to enhance the environmental perception and intelligent decision-making abilities of financial robots [18, 19]. These scholars have conducted multiple real-world application cases in diverse areas such as financial transactions, risk management, and investment advice. Through collaborative verification, they have theoretically validated the feasibility of architectures and algorithms and improved the performance and effectiveness of financial robots in practical scenarios [20]. Cross-border cooperation between research institutions, financial organizations, and technology companies has been increasing, leading to joint advancements in the research and application of cloud-based and information security-oriented financial robots. The comparative analysis of financial robot software architecture and self-localization algorithms is illustrated in Table 1:

Table 1. Comparison of financial robot software architecture and self-localization algorithms.

Study	Method	Advantages	Limitations	Research Gaps
Reference [6]	Cloud Computing Environment for Financial Robot Software Architecture Design and Optimization	Enhances performance and scalability, focuses on distributed computing and task scheduling	Primarily focuses on performance optimization, potentially neglecting information security and localization issues	Requires a more integrated approach to security and localization algorithm optimization
Reference [7-9]	Security Technologies such as Data Encryption, Identity Authentication, and Access Control	Emphasizes information security, covers data encryption and authentication	May lack attention to performance optimization in a cloud computing environment	Need for research combining cloud computing environments with enhanced system performance and security

Reference [10-12]	Self-Localization Algorithm Using Multi-Sensor Information Fusion	Improves localization accuracy, enhances environmental perception	May be limited by sensor and algorithm accuracy in practical applications	Further validation needed for algorithm effectiveness in real-world financial scenarios
Reference [13]	Distributed Computing, Fault Tolerance Mechanisms, Microservices Architecture	High performance, strong scalability, good adaptability	May overlook data privacy protection and information security	Integration with data privacy protection measures needed to enhance overall security
Reference [14-17]	Containerization and Serverless Computing Technologies	Utilizes emerging technologies, increases architectural flexibility and manageability	Potential technical integration and stability issues in practical applications	Need to align with specific requirements of the financial industry
Reference [18-20]	Image Recognition, Speech Recognition, Natural Language Processing	Enhances environmental perception and intelligent decision-making capabilities	High technical demands, potential challenges in practical application	Need to improve adaptability and practicality for specific financial tasks

In summary, the current research emphasis in China revolves around financial intelligence, information security, and cloud computing. However, further investigation is required in the areas of financial robot software architecture and self-localization algorithms. Nevertheless, foreign research has demonstrated significant expertise in software architecture, robot intelligence, and adaptive algorithms, providing valuable references and insights for related research in China.

3. METHODS

3.1 Research Framework

The overarching implementation framework is depicted in Fig. 1.

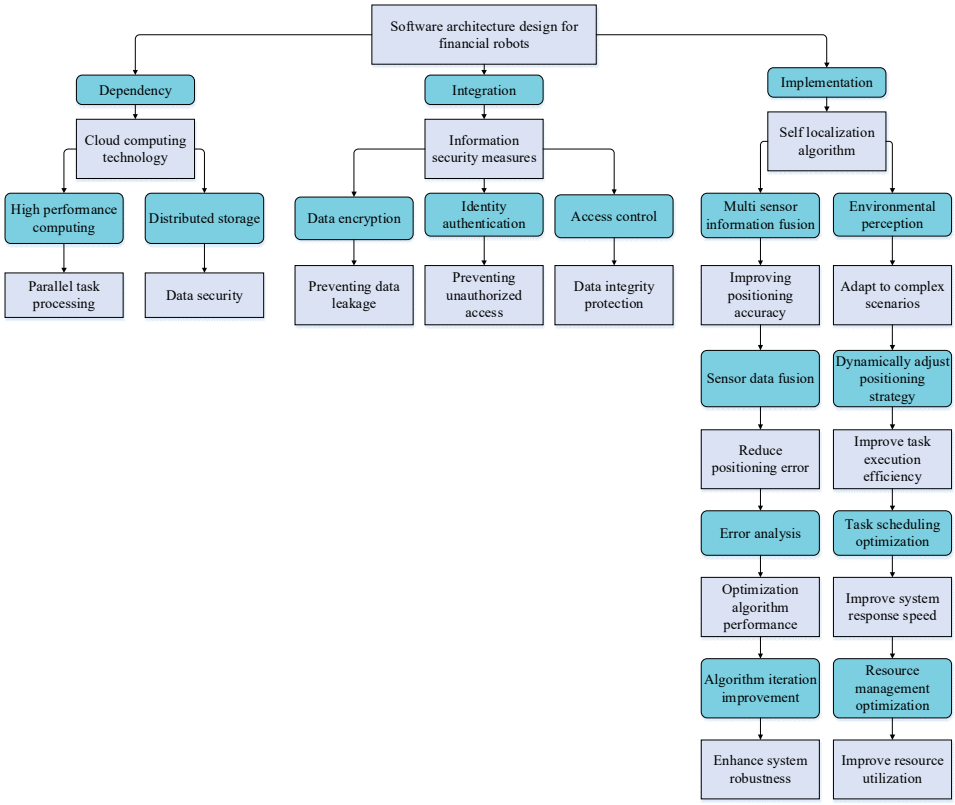


Fig. 1 Overall implementation framework

Fig. 1 elucidates the multifaceted design of the financial robot software architecture and the intricate interconnections between its components. At the heart of this framework is the software architecture, which leverages cloud computing technology to enable high-performance computing and distributed storage. This setup supports parallel processing of tasks and ensures robust data security. Integrated within this architecture are critical information security measures such as data encryption, identity authentication, and access control. These measures are designed to prevent data breaches, unauthorized access, and to safeguard the integrity of data. The self-localization algorithm stands as a pivotal element within the financial robot software framework. By employing multi-sensor information fusion technology, this algorithm enhances localization accuracy and equips the robot to navigate complex environments. The fusion of sensor data effectively reduces localization errors, while dynamic adjustment of localization strategies further amplifies task execution efficiency. Error analysis and task scheduling optimization contribute to refining the algorithm's performance and accelerating system response times. In this implementation framework, the iterative improvement of the self-localization algorithm, driven by error analysis, fortifies the system's robustness. Concurrently, optimization of task scheduling and resource management enhances resource utilization, ensuring high efficiency and stability when processing vast amounts of financial data and managing complex computational tasks. Overall, the framework provides a clear representation of the key elements

and their interactions within the financial robot software architecture, delivering comprehensive technical support for achieving efficient, secure, and reliable financial robot operations.

3.2 Software Architecture Design of Financial Robot

Financial robots, sophisticated intelligent devices deployed within the financial sector, are designed to autonomously or semi-autonomously perform a variety of financial tasks. These tasks encompass, but are not limited to, automated trading, risk management, data analysis, and client services. As cloud computing and information security technologies have evolved, the scope and significance of financial robots have escalated considerably. In the contemporary financial landscape, these robots significantly boost the efficiency and accuracy of financial services through their adept data processing and precise decision-making capabilities, while simultaneously mitigating the risks associated with manual operations. The applications of financial robots in the financial domain are diverse and impactful:

(1) Automated Trading: Financial robots execute buying and selling operations based on preset trading strategies and real-time market data, enabling automation of trading and high-frequency trading practices.

(2) Risk Management: Utilizing big data analytics and machine learning algorithms, financial robots continuously monitor market fluctuations, assess risks, and implement appropriate measures to aid financial institutions in effective risk management.

(3) Client Interaction: Through natural language processing and voice recognition technologies, financial robots engage with clients, offering round-the-clock customer support and financial consulting services.

(4) Data Analysis and Modeling: Financial robots swiftly handle vast quantities of financial data, performing intricate analyses and modeling to provide a scientific foundation for financial decisions.

(5) The role of financial robots is increasingly pivotal in enhancing service quality, optimizing resource allocation, reducing operational costs, and improving client satisfaction. They have emerged as a critical focus in the advancement of modern financial technology, underscoring their growing influence in the field.

The design of financial robot software architecture encompasses creating the software system for financial robots, including defining the system's modules, components, and their relationships [21, 22]. This design ensures that the financial robots can efficiently perform various financial tasks, including processing accounting vouchers, reconciling account balances, and generating reports [23, 24]. Design principles such as modularity, layering, and loose coupling can be adopted when designing software architecture to reduce complexity and enhance maintainability. Moreover, the incorporation of suitable technologies and tools can support the development and deployment process, improving efficiency and minimizing overhead [25, 26].

The design process for financial robot software architecture involves the following specific steps:

Step 1: Requirements definition

The first step is to define the requirements of the financial robot software system. These requirements encompass usability, developability, real-time capability, interactivity, and dynamism.

Step 2: Communication plan design

This step involves presenting a financial robot software architecture implemented using the Robot Operating System (ROS). ROS is a widely adopted framework known for its capabilities in facilitating cross-language and cross-platform interprocess communication. Its Remote Procedure Call (RPC) framework effectively addresses the challenge of interprocess communication between hosts and programming languages [27, 28]. Fig. 2 illustrates the proxy node-based communication scheme.

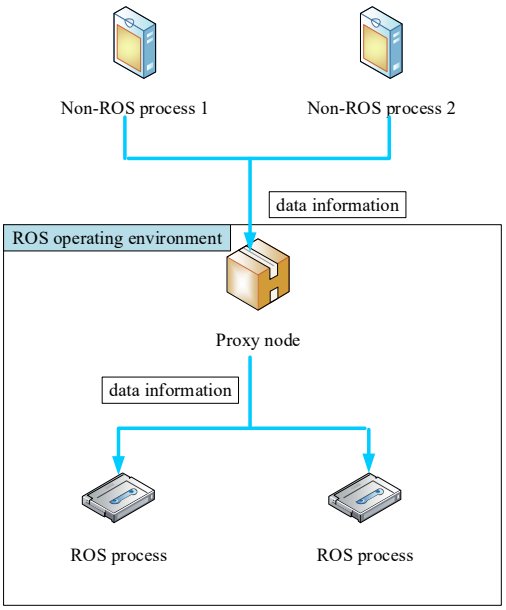


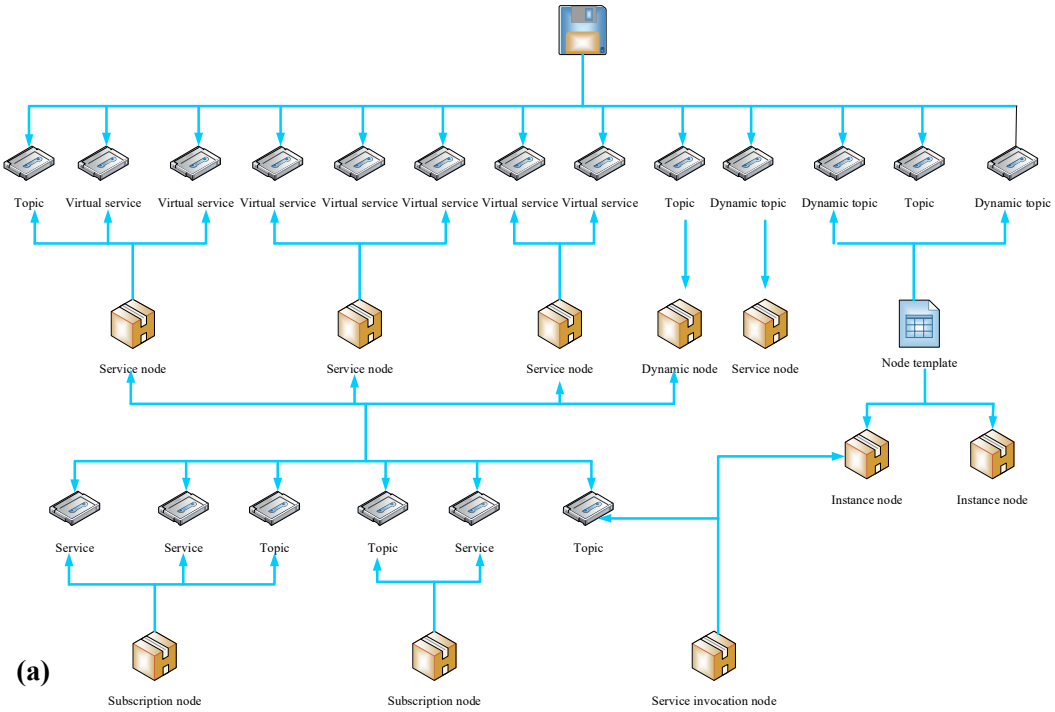
Fig. 2 Proxy node-based communication scheme

Fig. 2 demonstrates the versatility of using a more generalized RPC method for dynamic encapsulation, enabling the utilization of ROS topics and services in non-ROS environments. This is achieved through the deployment of a proxy node in a runnable ROS environment, thereby facilitating access to ROS topics and services in other environments.

Step 3: Establishment of bidirectional communication between ROS nodes

Step 4: Design of the architectural view

The architectural view of the financial robot software design includes the component connector view and the framework deployment view [29], as indicated in Fig. 3.



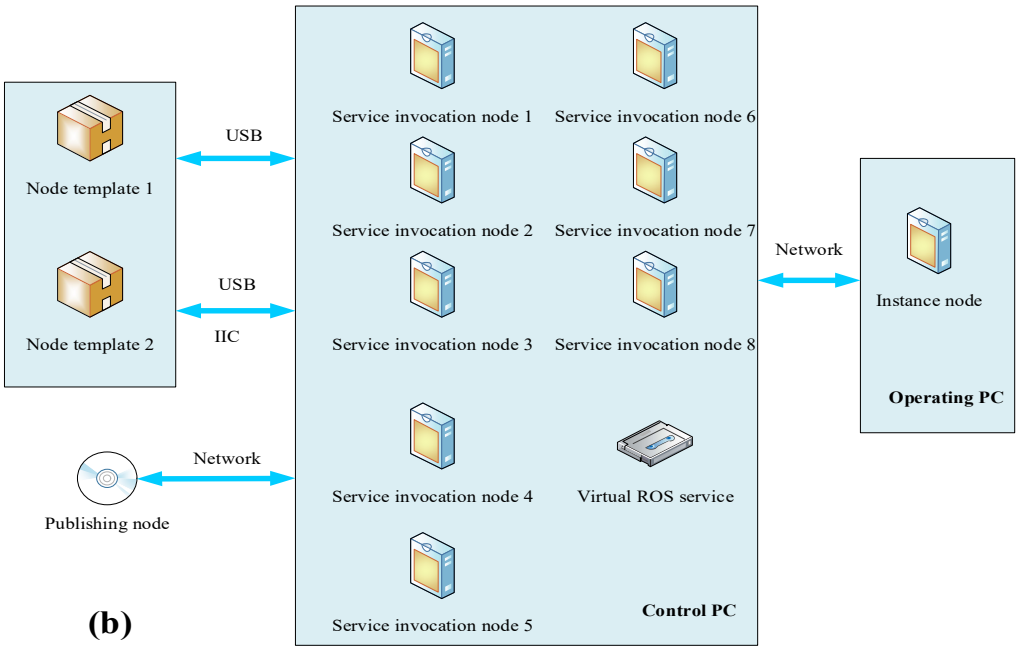


Fig. 3 Financial robot software architecture design view ((a) component connection; (b) software architecture deployment)

Fig. 3 illustrates the comprehensive financial robot software architecture, comprising the component connection view and the software architecture deployment. These two aspects work together to complement the entire software system, which is subsequently subjected to performance testing. The design of the financial robot software architecture is meticulously crafted to guarantee the system's security, accuracy, and efficiency. Anchored in cloud computing technology, this architecture is segmented into several key modules, each assigned specific functions and governed by stringent information security protocols for data transmission and interaction. The core modules and their functions are outlined as follows:

(1) Data collection module: This module is tasked with aggregating data from diverse sources—market trends, transaction records, customer information, and more. Utilizing MSIF technology, it ensures comprehensive and precise data capture, enhancing the reliability of subsequent analyses.

(2) Data processing module: Charged with the cleansing, filtering, and preprocessing of collected data, this module employs big data analytics and machine learning algorithms to analyze and model the information. The processed data serves as the foundational input for decision-making processes.

(3) Decision support module: Operating on preset trading strategies and risk management frameworks, this module integrates real-time data to generate actionable trading decisions and risk assessment reports. It leverages High-

Performance Computing (HPC) technologies to ensure both the timeliness and efficiency of decision outputs.

(4) Execution module: Responsible for transmitting the trading directives and risk control measures, as formulated by the Decision Support Module, to the trading and risk management systems. This module also monitors the execution phase to ensure the precision and promptness of trades and risk mitigation actions.

(5) Monitoring and feedback module: This component provides continuous surveillance of the financial robot's operational state, gathering and analyzing runtime data to produce performance reports and improvement recommendations. Additionally, it facilitates user interaction, collecting feedback to enhance the overall user experience.

(6) Information security module: Employing state-of-the-art security technologies—such as encryption algorithms, firewalls, and intrusion detection systems—this module safeguards the confidentiality, integrity, and availability of data, thereby ensuring the secure operation of the entire system.

Together, these modules form a robust framework designed to optimize the functionality and security of financial robots, supporting their advanced capabilities in the financial sector.

Step 5 involves selecting a suitable hardware platform.

Step 6 entails choosing an appropriate operating system.

Step 7 pertains to selecting a control platform.

Step 8 focuses on conducting software performance testing.

Step 9 involves evaluating the software's functional implementation by experts.

Step 10 encompasses analyzing the obtained results.

3.3 A robot-self Localization Algorithm based on MSIF

Implementing self-localization algorithms in financial robots is essential for several compelling reasons. Firstly, the autonomous navigation and precise positioning required during task execution are crucial for the efficient operation of financial robots. In environments like banks and stock exchanges, these robots often need to navigate autonomously between various workstations, deliver critical documents, or assist in customer service. Without accurate localization capabilities, there is a significant risk of deviations in path planning and task execution, potentially leading to reduced efficiency or operational errors. Furthermore, self-localization algorithms significantly enhance the accuracy and stability of financial robots' positioning. Complex financial environments often present numerous obstacles and interference factors, such as dense crowds or intricate layouts. Traditional localization methods may struggle with these complexities, resulting in substantial errors. However, by utilizing MSIF techniques, self-localization algorithms can integrate data from multiple sensors, offering more precise and reliable positional information. This

multi-source data fusion effectively reduces localization errors, ensuring that robots can maintain accurate positioning even in challenging environments. In addition to improving accuracy, self-localization algorithms also enhance the safety of financial robots. Safety is a paramount concern in financial settings. Inaccurate positioning could lead to collisions with people or objects, causing property damage or injury. A precise self-localization algorithm ensures that robots can navigate around obstacles and complete tasks safely and efficiently, thereby safeguarding the entire system. Moreover, these algorithms bolster the robots' autonomous learning and adaptation capabilities. Through continuous localization and navigation, robots can gather environmental data and incrementally refine their localization algorithms, enhancing their self-learning abilities. This adaptability allows financial robots to operate efficiently in ever-changing environments, meeting diverse task requirements. In conclusion, the application of self-localization algorithms in financial robots not only improves positioning accuracy and safety but also enhances their self-learning and adaptability. This ensures that they can complete tasks efficiently and safely in complex financial environments. Therefore, researching and developing efficient self-localization algorithms is crucial for enhancing the overall performance of financial robots.

The MSIF-based robot self-localization algorithm is a sophisticated technique used for robot localization, utilizing multiple sensors such as laser rangefinders, vision cameras, and inertial measurement units to gather environmental and robot state information. This information is subsequently fused and processed to determine the precise location of the robot in the environment [30]. A fundamental aspect of this algorithm is the inclusion of a motion model of the robot, which aids in inferring the robot's position at different time intervals. The complexity and overhead associated with the MSIF-based robot self-localization algorithm primarily involve sensor selection and integration, the design of the data fusion algorithm, data calibration and correction procedures, computation and storage overhead, algorithmic tuning, and performance evaluation. To ensure optimal system performance, it is essential to carefully balance these factors when designing and implementing the algorithm, taking into account system requirements while mitigating complexity and overhead. The positioning algorithm is formulated in three key components:

(1) Multi-sensor information fusion.

The structural diagram of the multi-sensor information fusion model is depicted in Fig. 4.

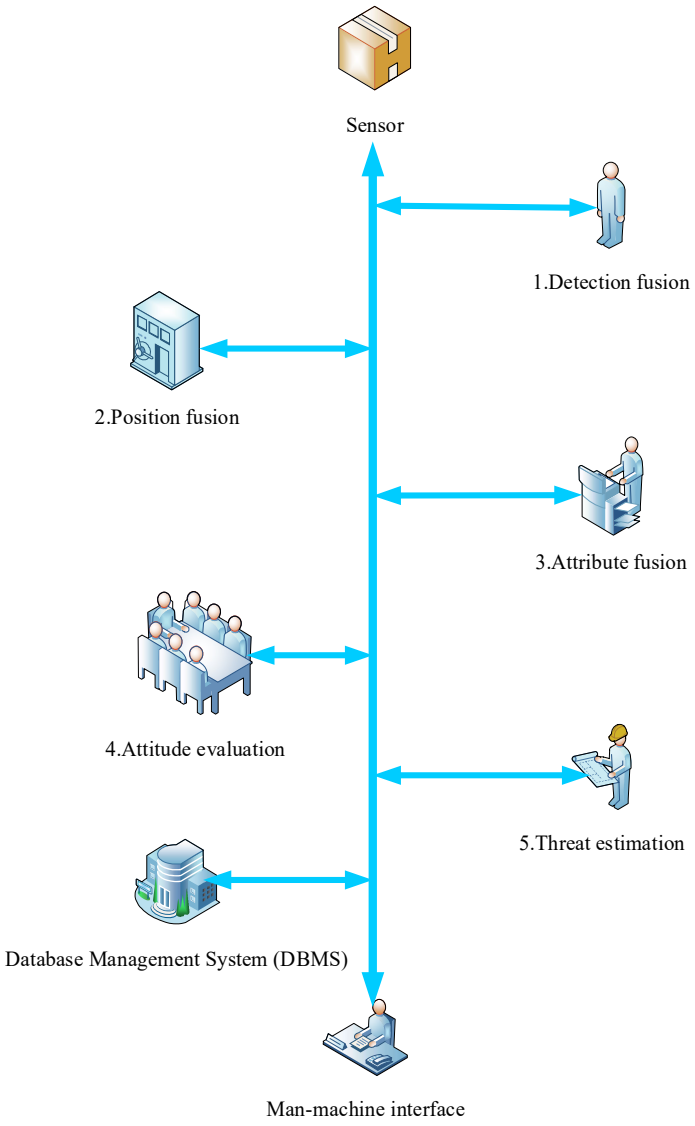
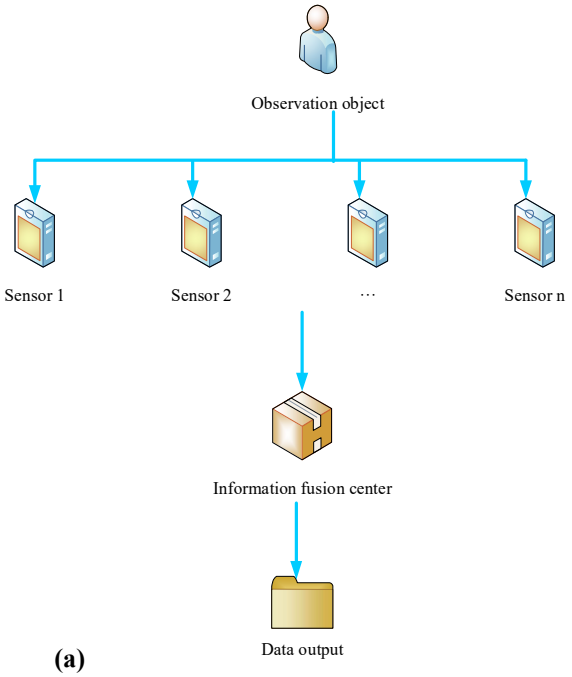


Fig. 4 Structure of multi-sensor information fusion model

Fig. 4 illustrates a comprehensive view of information fusion, which is categorized into five levels: detection fusion, position fusion, attribute fusion, attitude evaluation, and threat estimation. Detection fusion involves directly detecting or assessing signals. Position fusion is of utmost importance as it involves both temporal and spatial fusion. This level can be further classified into three types: centralized, decentralized, and hybrid fusion structures. Attribute fusion consists of data, feature, and decision layers. Attitude evaluation entails analyzing and inter-

preting complex environments to arrive at accurate inferences. Finally, threat estimation involves evaluating decision uncertainties and randomness. Fig. 5 depicts the two modes of information fusion.



(a)

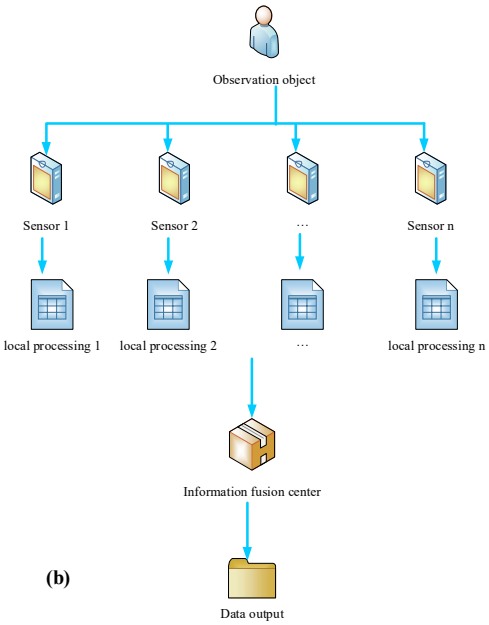


Fig. 5 Two modes of information fusion ((a) centralized structures; (b) distributed structure)

Fig. 5 depicts the two modes of information fusion: decentralized and centralized. In decentralized information fusion, data processing is distributed, while the remaining steps are similar to those in centralized information fusion. However, decentralized information fusion exhibits superior efficiency and accuracy in data processing.

(2) Design Bayesian filter positioning algorithm and Kalman filter (KF) positioning algorithm)

Bayesian filtering localization involves treating the localization algorithm as a Bayesian estimation problem. The robot's position is denoted as (a, b, θ), where (a, b) represents the robot's two-dimensional coordinates, and θ stands for the heading direction. At time T, the robot's motion state is recorded as a_t. The expression of the Bayesian rule is given by Equation (1):

$$B(a_t) = \frac{p(Z_t|a_t, Z_{0\cdots t-1})p(a_t|Z_{0\cdots t-1})}{P(Z_{0\cdots t-1})} \quad (1)$$

In Equation (1), $B(a_t)$ represents the robot's motion state, and the state sequence before time t is denoted as $a_{0,\cdots,t-1} = (a_0, a_1, \cdots, a_{t-1})$, while the sensor observation sequence is $Z_{0\cdots t-1} = (Z_0, Z_1, \cdots, Z_{t-1})$. The term $p(a_t|Z_{t-1})$represents the probability density. According to Bayesian filtering, Equation (1) can be transformed into Equation (2).

$$B(a_t) = \eta p(a_t|Z_t) \int p(a_t|a_{t-1})\, B(a_{t-1}) da_{t-1} \quad (2)$$

In Equation (2), $p(a_t|Z_t)$represents the sensor observation model, and $p(a_t|Z_t)$ signifies the system state transition model. When designing the algorithm, it is formulated as state prediction and state update, as expressed in Equations (3) and (4):

$$B(a_t) = \int p(a_t|a_{t-1})\, B(a_{t-1}) da_{t-1} \tag{3}$$

$$B(a_t) = \eta p(a_t|Z_t) B(a_t) \tag{4}$$

The steps of the KF positioning algorithm are illustrate in Fig. 6.

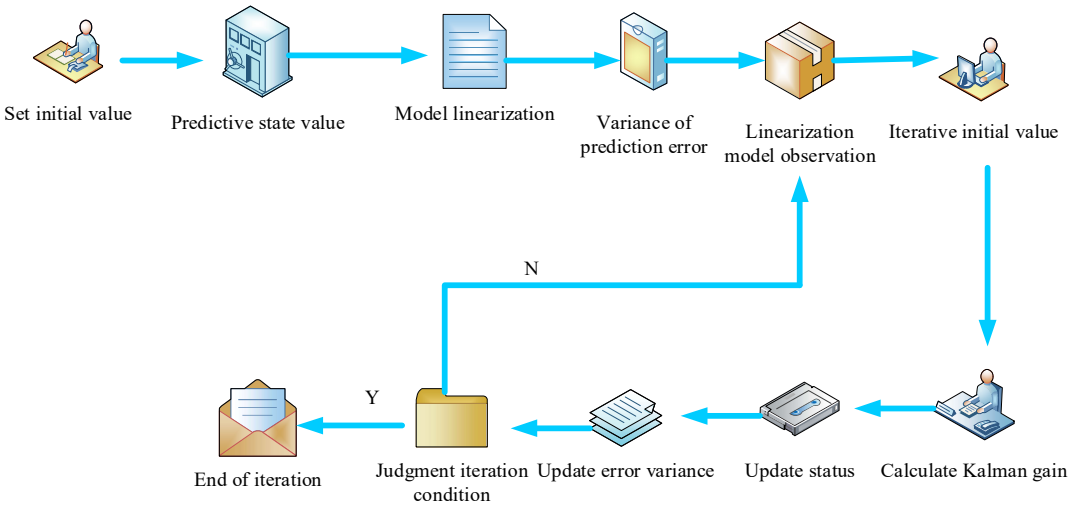


Fig. 6 Steps of KF positioning algorithm

Fig. 6 the steps involved in the KF localization prediction algorithm. The algorithm begins by setting the initial values. Next, the state values are predicted. Subsequently, the model is linearized, and the linearized model is used to predict the error and variance. Afterwards, the initial values and iteration count are set, and the linearized model is iterated and observed. Next, the Kalman gain is calculated, and state updates, as well as error and variance updates, are performed. Finally, the result is evaluated against the exit loop condition. If the condition is satisfied, the iteration is terminated; otherwise, the iteration process continues.

The system state variable c is assumed to be a real number; v represents input; m refers to process noise. The system state transition at time t is represented by Equation (5).

$$\mathrm{C_t} = \mathrm{Ac_{t-1}} + \mathrm{Bv_{t-1}} + \mathrm{m_t} \tag{5}$$

Here, A indicates the gain matrix; B refers to the input gain. The observation equation at time t is given by Equation (6).

$$z_t = Dc_t + u_t \tag{6}$$

In Equation (6), u_t means the observation noise at time t; z stands for the observed variable; D represents the actual gain. The observation noise and process noise follow the Gaussian distributions and can be written as Equations (7) and (8), respectively:

$$P(m) = N(0, Q) \tag{7}$$

$$P(u) = N(0, R) \tag{8}$$

Here, Q and R are constants representing the covariance matrix of the two types of noise.

The application structure of KF based on recursive estimation theory is displayed in Fig. 7.

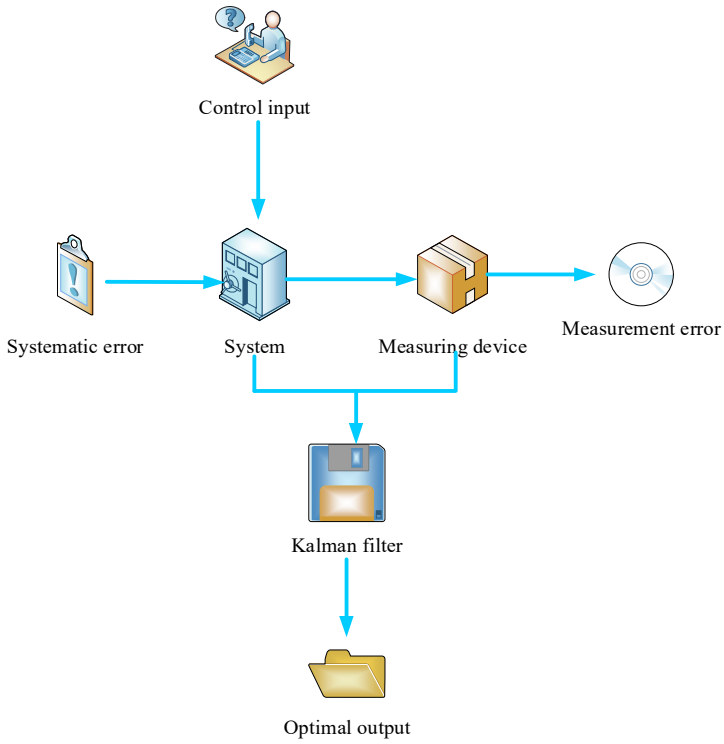


Fig. 7 Application structure of KF based on recursive estimation theory

Fig. 7 illustrates the recursive estimation theory proposed by the KF, which utilizes a state-space representation. The KF algorithm, in its recursive form, can handle multidimensional stationary and non-stationary stochastic processes. As a result, it has found widespread applications in modern engineering fields, including navigation, control, guidance, and communication.

(3) Structural design of the MSIF positioning algorithm and localization information fusion

At time t, the sensors detect n features denoted as $L = \{L_1, L_2, \cdots, L_n\}$, but due to prediction, k features are estimated as $M = \{M_1, M_2, \cdots, M_k\}$. The residual between the observed and estimated values is given by Equation (9).

$$v = z(t) - \hat{z}(t) \tag{9}$$

Here, $z(t)$ and $\hat{z}(t)$ refer to the actual observed value and the estimated measurement value at time t. The covariance matrix is computed as Equation (10)

$$S = HPH^T + R \tag{10}$$

In Equation (10), H represents the Jacobian matrix; P means the covariance matrix of errors; R indicates the covariance matrix of observation noise. The relationship between the measured value and the actual value is described by Equation (11).

$$D^2 = u^T S^{-1} u < \chi_\alpha^2(k) \tag{11}$$

In Equation (11), $\chi^2(k)$ represents the chi-square distribution with k degrees of freedom, α refers to the significance level, and 1- α represents the confidence level, which is typically set at 0.95.

Fig. 8 illustrates the process of the fusion robot positioning algorithm.

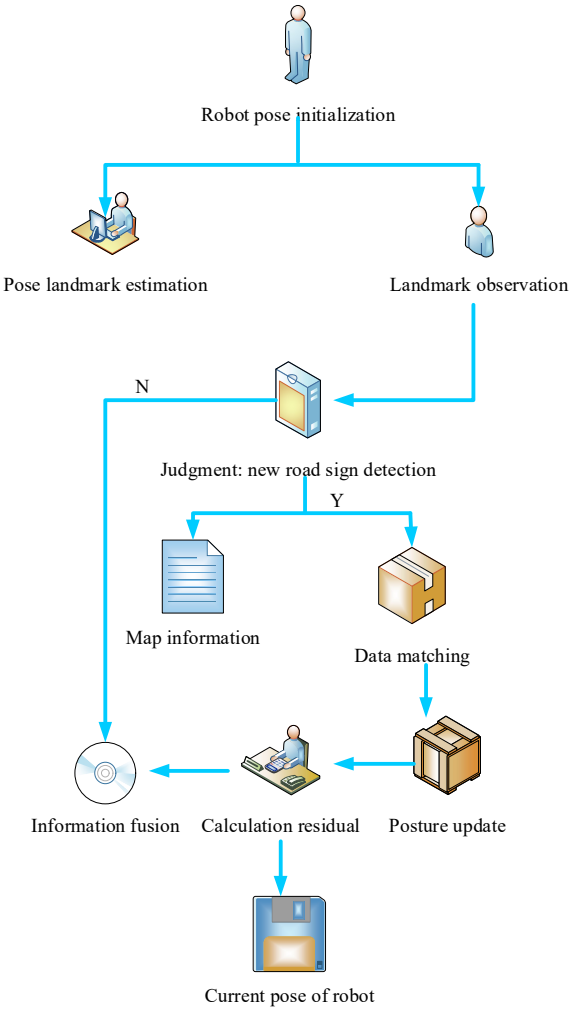


Fig. 8 Fusion robot positioning algorithm process

Fig. 8 indicates that the core of the fusion algorithm lies in matching and fusing data information, leading to the precise determination of the robot's current position and orientation through residual calculations.

The pseudocode for the self-localization algorithm based on MSIF is outlined in Table 2.

Table 2. Pseudocode for the self-localization algorithm based on MSIF.

Step	Pseudocode

1. Initialization	-
1.1 Define Variables	sensor_data = []
	position = [x0, y0]
	heading = theta0
	state_estimate = [position, heading]
1.2 Parameter Settings	sensor_noise = [...]
	control_noise = [...]
	dt = 0.025
	N = 100
2. Data Acquisition	-
2.1 Data Retrieval	sensor_data = get_sensor_data()
3. Prediction Phase	-
3.1 Motion Model	state_predict = motion_model(state_estimate, dt)
3.2 Control Noise	state_predict += control_noise
4. Update Phase	-
4.1 Observation Model	observation = observation_model(sensor_data)
4.2 Kalman Gain	K = calculate_kalman_gain()
4.3 State Update	state_estimate = state_predict + K * (observation - state_predict)
5. Data Fusion	-
5.1 Fusion Process	state_estimate = sensor_fusion(state_estimate, sensor_data)
6. Iterative Calculation	-
6.1 Loop Iteration	for i in range(N):
	sensor_data = get_sensor_data()
	state_predict = motion_model(state_estimate, dt)
	state_predict += control_noise
	observation = observation_model(sensor_data)
	K = calculate_kalman_gain()

	state_estimate = state_predict + K * (observation - state_predict)
	state_estimate = sensor_fusion(state_estimate, sensor_data)
7. Output Results	-
7.1 Position Output	position, heading = state_estimate
7.2 Error Analysis	error = calculate_error(state_estimate, true_state)

Localization via information fusion harnesses a myriad of sensors and varied informational sources to compute positional estimates with exceptional precision. This complex process mirrors the workings of a feedback control system, perpetually adjusting parameters and refining models to elevate the accuracy of localization, thereby ensuring robust environmental mapping. Fig. 9 provides an illustrative depiction of the structural framework of this intricate information fusion localization system.

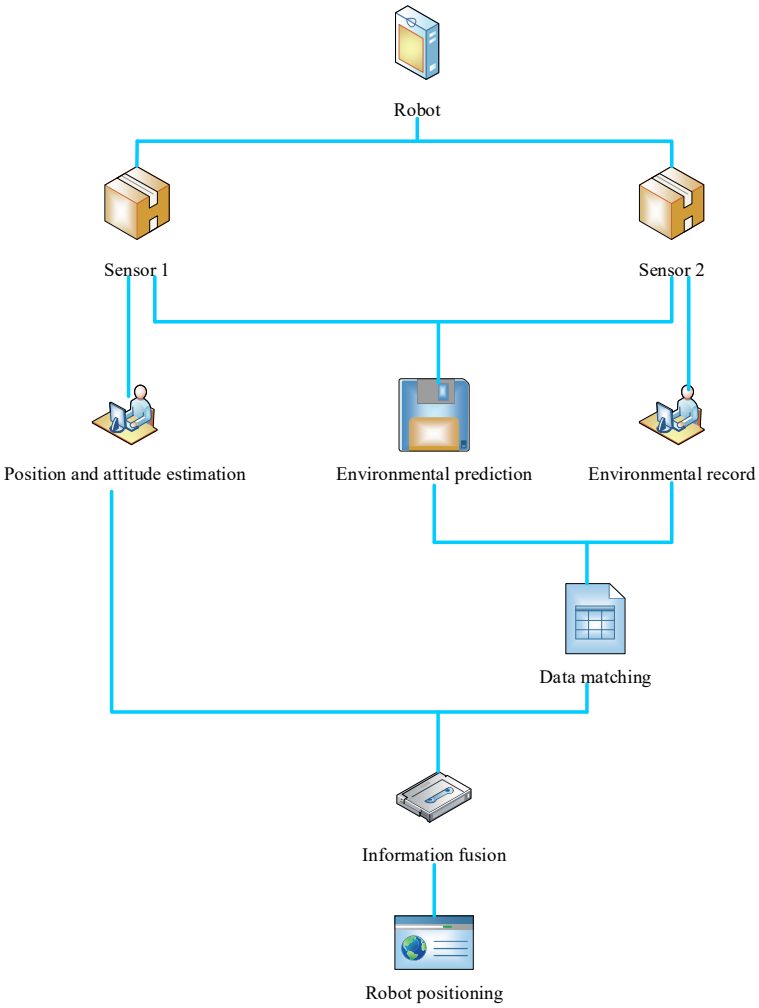


Fig. 9 Information fusion localization architecture

Fig. 9 showcases the information fusion structure, enabling the robot to incorporate data from multiple sensors. The robot initiates its motion from a known point and tracks its movement using odometry control information. However, as odometry inherently introduces errors, the robot's position becomes uncertain after covering a certain distance. To mitigate uncertainty and rectify accumulated errors, the Iterative Extended KF algorithm is adopted to fuse information from internal and external sensors and constantly update the robot's state, enabling relative localization within the environmental map. Prior to information fusion, preprocessing of diverse sensor information is imperative, involving the elimination of erroneous data and the extraction of reliable information to guarantee the accuracy and efficacy of the information fusion process.

4 Experimental Design and Performance Evaluation

4.1 Experimental Materials and Data Collection

(1) Data collection for financial robot software architecture

The assessment of the financial robot software architecture's performance pivots on meticulously gathered data, sourced from The Software Testing Dataset Repository, enabling an exhaustive examination of the software's capabilities.

The data collection employs Selenium automation scripts, written in Java, designed to interact with web pages via Selenium WebDriver, mimicking user actions within a browser. The dataset encompasses supplementary details, such as issue creation dates, last edited dates, issue titles, and tags. The test suite comprises a compilation of multiple test cases, each tailored to rigorously scrutinize diverse facets of the software application. Every test case is designated with a unique Test Id, Title, and Description. Below is a translation of a selection of test cases from the dataset:

Test Id 1: Essential fields must be verified and marked with an asterisk (*). This involves checking mandatory fields across forms, screens, pop-ups, etc., ensuring they are duly marked. Additionally, any agreed-upon color-coding for these fields and asterisks should be visible.

Test Id 2: Verification error messages must be accurately displayed in the correct positions. Inputting invalid values into any field on the webpage should trigger an error message. The triggered error message must align with the expected design as outlined in the Functional Requirement Document or any other stipulated knowledge base documentation.

Test Id 3: All error messages should be presented using a uniform Cascading Style Sheets (CSS) style, such as red. Any error message triggered during validation should be color-coded using the same CSS style to maintain consistent color-coding throughout the application.

Test Id 4: General confirmation messages should utilize a CSS style distinct from error messages, such as green. Any messages other than standard error messages (e.g., warning messages) should be displayed using a different CSS style. For instance, if error messages are shown in red, warning messages should use an alternate color like green.

Test Id 5: Tooltip text should be meaningful. Tooltips help users quickly understand specific web page components/features. These tooltips should convey meaningful information about the components/tools on the page and should be concise.

Test Id 6: Dropdown fields should have the first item set to blank or text like "Select". All dropdown menus on any page should have the first item set to blank or text like "Select," indicating that nothing is selected by default unless otherwise specified by design.

Test Id 7: The “Delete Function” for any records on a page should prompt confirmation. Clicking the delete button on any webpage should not immediately trigger the deletion action. Instead, the system should request user confirmation through appropriate

standard channels. For example, clicking the delete button should trigger a pop-up window with an appropriate confirmation message and “OK” and “Cancel” buttons.

Test Id 8: If a page supports the addition/deletion/updating of records, it should provide options to select/unselect all records. Whenever a page supports addition/updating/deletion functionality, the page should offer users an option to select all selectable items using a select-all button. It should also provide another option to deselect all selected items using a deselect-all option.

Test Id 9: Amount values should display the correct currency symbol. Fields that update or display monetary values should include th appropriate currency symbol. These symbols should correspond to the currency being displayed by default or the currency being input.

Test Id 10: A default page sorting should be provided. The page should have a default sorting order.

The dataset continues with further test cases, covering an expansive range of test points from user interface elements (such as button functionality, field alignment, font size, and style) to backend functionalities (including database performance, security testing, file upload, and email functionalities). Each test case is meticulously designed to ensure that all aspects of the application adhere to expected standards and functional requirements, providing a comprehensive framework for evaluating the robustness and reliability of the financial robot software architecture.

(2) Testing the MSIF-based robot self-localization algorithm

In the rigorous assessment of the MSIF-based self-localization algorithm, a dual-wheel robot with a wheelbase of 0.5 meters serves as the test platform. This robot navigates within a specified motion environment, meticulously designed to evaluate the MSIF algorithm's performance in real-world applications. The experimental setup comprises the robot moving at a velocity of 2 meters per second within a 100-meter by 100-meter area. To capture precise environmental data, the sensor sampling interval is set at 0.025 seconds. The observation angle ranges from -30° to 30°, allowing for a maximum observable distance of 30 meters. As the robot traversed its path, it maintained an angular velocity of $g=20\times\pi/180$ rad/s. The experiment also accounted for control noise at 0.6 meters per second and observation noise at 0.1 meters per second.

4.2 Experimental Environment

To address the computational requirements of this software architecture system, an embedded chip architecture is recommended, with options such as ARM and x86. Both architectures offer robust computational capabilities, ease of development, high versatility, and strong scalability. For optimal support and compatibility, a platform with a Linux operating system and a suitable compilation toolchain is advised.

This work adopts the Low-Latency Kernel as the operating system, chosen for its ability to meet the real-time requirements of this system. This operating system exhibits advantages such as an average wake-up latency of only 5 μs and a maximum latency of approximately 160 μs, making it highly suitable for handling heavy workloads. Regarding

the robot control platform (Control PC), the PCM-3365EW-S9A1E industrial control board is recommended. This control board features a compact size of approximately 96 × 90 × 28.8 mm, a power supply requirement of 5V ± 5%, and a maximum power consumption of about 8W, ensuring efficiency and effective power management. Moreover, it can operate within a wide temperature range, from -40°C to 85°C. The control board is equipped with 4GB DDR3L memory and supports a solid-state drive through the mSATA interface. Additionally, it offers various interfaces, including a 10/100/1000 Mbps adaptive Ethernet port, VGA, LVDS, HDMI display interfaces, USB 2.0 interfaces, RS-232/422/485, I2C, and 8-bit GPIO interfaces. These diverse interfaces cater to the requirements of various embedded devices.

4.3 Indicator Settings

(1) Experimental platform for financial robot software architecture

The experimental platform for evaluating the financial robot software architecture, which is based on cloud computing and information security, encompasses two main aspects: software real-time performance and ROS communication performance. In the real-time performance test, a comparison is conducted between the previously designed software system (A) and the newly proposed software system (B). The evaluation is based on the average response time and maximum response time of each system. In the communication performance test, 1,000 experiments are conducted per group, with a total of 10 groups. This comprehensive approach ensures robustness in evaluating the system's communication performance based on communication latency. The comparison groups encompass various scenarios, including Transmission Control Protocol connections (M1), ROS topics (M2), ROS non-persistent connections (M3), ROS persistent connections (M4) with a low-latency scheduling strategy, and ROS persistent connections with normal latency scheduling (M5). Key evaluation indicators include average latency (C), maximum latency (D), percentage of delays exceeding 0.5 ms (E), percentage of delays exceeding 1 ms (F), and percentage of delays exceeding 1.5 ms (G).

For expert evaluation, a panel of 15 experts is invited to rate the functional implementation and performance of the financial robot software architecture via email. Specific evaluation criteria are outlined in Table 3.

Table 3. Function and performance evaluation criteria for financial robot software architecture.

Primary indicators	Functional indicator H	Performance indicators I
Secondary indicators	Function integrity degree H1	Data processing speed I1
	Function development degree H2	Concurrent performance I2 Scalability I3
	Function execution degree H3	System stability I4 Security I5 User-friendliness I6

The functionality of the proposed financial robot software architecture is evaluated using a rating scale of 1 to 10 according to the indicators described in Table 1. This quantitative assessment aims to provide valuable recommendations for system improvement, based on experts' ratings and feedback. It ensures a systematic and reliable evaluation, guiding future enhancements to improve the architecture's performance and functionality.

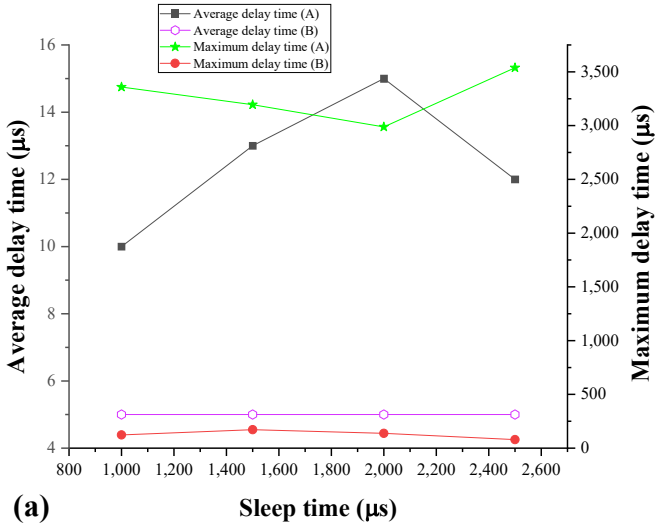
(2) A robot self-localization algorithm based on MSIF

The MSIF-based robot self-localization algorithm is rigorously tested from four perspectives: algorithm validity, the influence of the heading angle, localization accuracy, and the number of iterations for state updates.

4.4 Experimental Result

(1) Experimental platform of financial robot software architecture

Fig. 10 portrays the performance test results of the financial robot software architecture system, showcasing its capabilities and effectiveness under high system load. This includes average and maximum latency, as well as a detailed analysis of communication performance in cloud computing and information security settings.



(a)

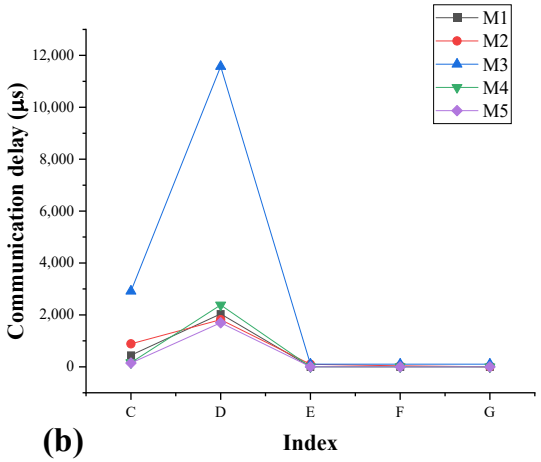


Fig. 10 Financial robot software architecture system performance test ((a) real-time test; (b) communication performance test)

Fig. 10 depicts the remarkable performance of the designed system under high system load, maintaining an average latency of 15μs and a maximum latency of 172μs. This result represents a significant improvement compared to previous software systems. Furthermore, the communication performance analysis reveals that ROS topics with a low-latency scheduling strategy exhibit an average delay of 884.8μs, while ROS services with persistent connections demonstrate consistent performance. Remarkably, in 99.83% of cases, the transmission delay remains below 1.5ms, indicating reliable and efficient communication. Fig. 11 summarizes the expert evaluation scores, showing the ratings for functionality and performance.

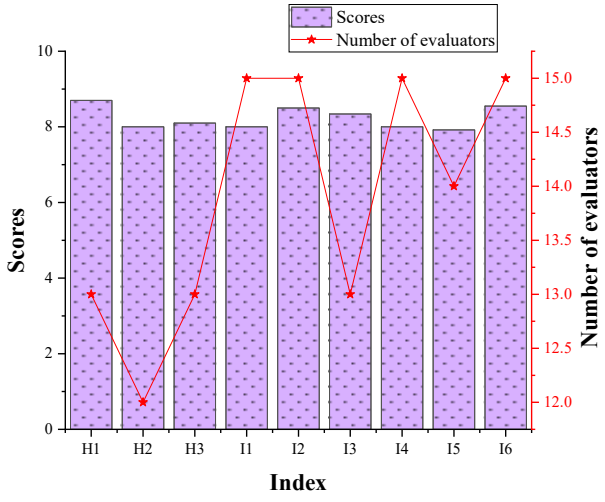
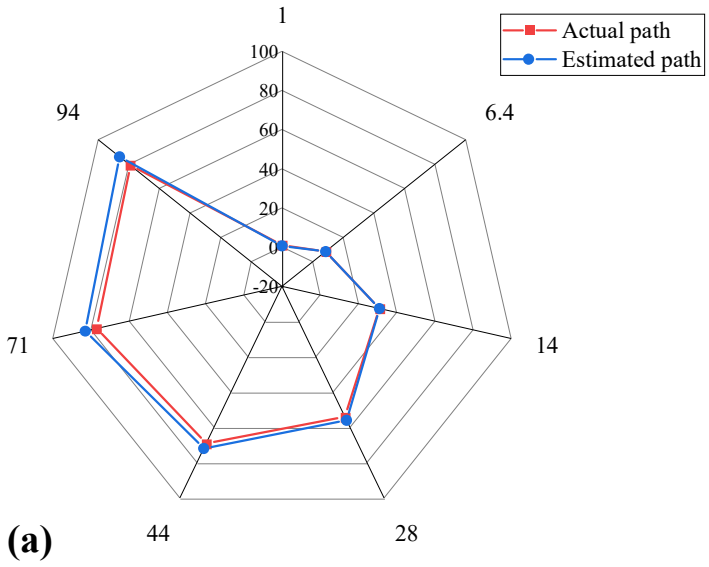


Fig. 11 Functionality and performance scoring of financial robot software architecture

Fig. 11 reveals that the current design of the financial robot software architecture receives ratings of 8 or above for both functionality implementation and performance. This indicates that the designed architecture, based on cloud computing and information security, successfully fulfills its intended functionality and performance and meets the requirements of downstream users. Notably, the ratings for functionality completeness and user-friendliness exceed 8.2, highlighting the overall superior performance of the system.

(2) MSIF-based robot self-localization algorithm

The robot self-localization algorithm undergoes performance testing in a 100m*100m environment. Fig. 12 compares the performance of traditional sensor-based localization methods with information fusion-based localization methods in robot positioning.



(a)

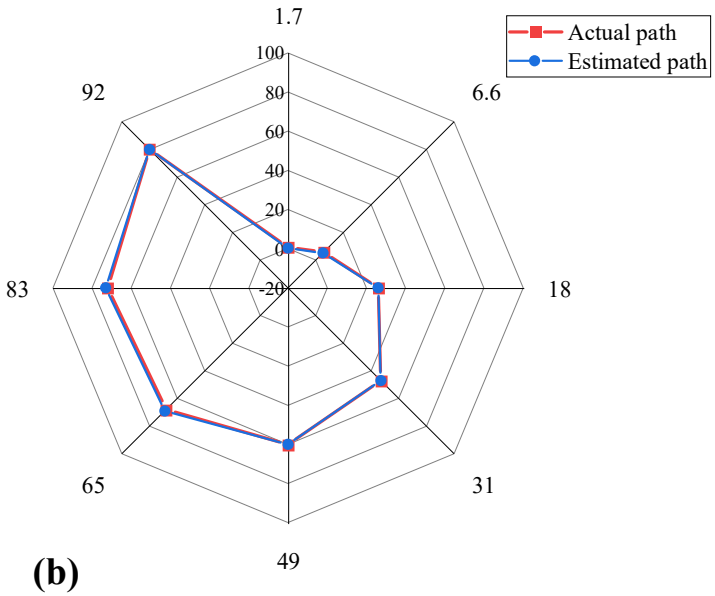


Fig. 12 Performance comparison of traditional sensor positioning method and information fusion positioning method in robot positioning ((a) traditional sensor localization; (b) information fusion localization)

Fig. 12 shows that while the traditional sensor localization method produces position and orientation estimates that generally align with actual trends, it exhibits notable errors. In contrast, using information fusion techniques to integrate and process sensor data effectively reduces localization errors, resulting in a substantial alignment between the predicted and actual paths. By leveraging data from multiple sensors and performing information fusion, the accuracy and reliability of the robot's self-localization are improved, ensuring precise navigation along pre-planned routes. Fig. 13 provides a detailed description of the correspondence between the predicted and actual trajectories in the robot's self-localization process, as well as the measurement of directional errors.

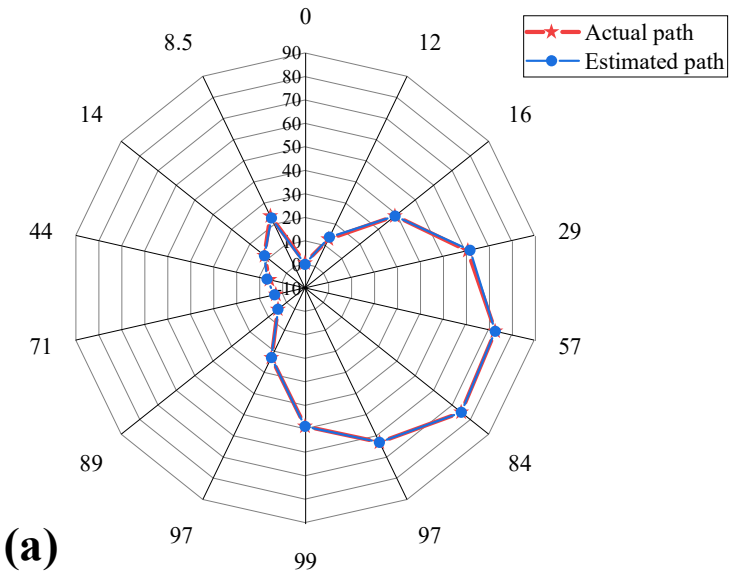


Fig. 13 Correspondence between predicted trajectory and actual trajectory and measurement of direction error during robot self-localization ((a) the self-localization trajectory of the robot; (b) direction error)

Fig. 13 provides a detailed depiction of the close correspondence between the predicted trajectory and the actual trajectory during robot self-localization. Furthermore, the measurement of directional error shows minimal fluctuations and remains close to zero. These findings indicate the algorithm's high stability and reliability in accurately determining the robot's position and direction. Fig. 14 shows the performance of the KF and Bayesian filtering localization methods in robot positioning, along with an error analysis.

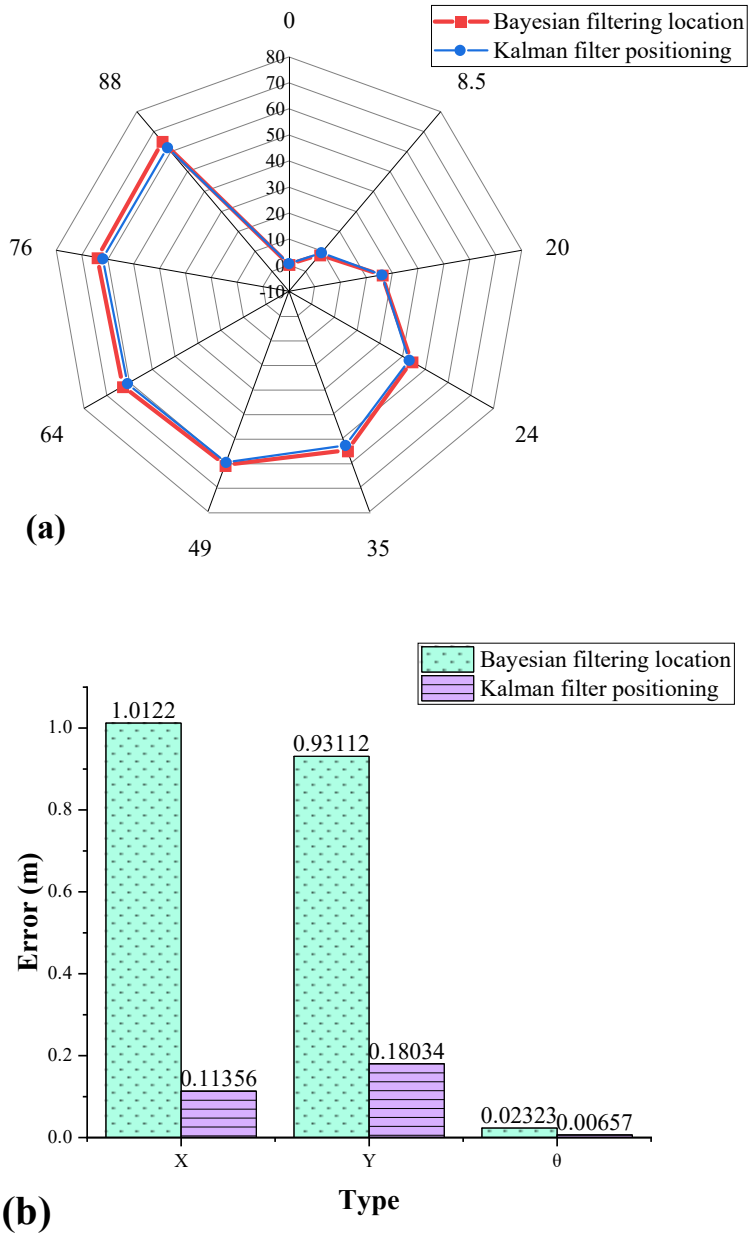


Fig. 14 Performance and error of KF localization and Bayesian filter localization in robot localization ((a) the comparison of localization results; (b) error comparison)

Fig. 14 shows that both filtering algorithms initially achieve favorable localization results. However, the KF algorithm outperforms the information fusion-based algorithm as time progresses, demonstrating a closer alignment between actual localization and predictions. Additionally, the error analysis highlights that the use of the information fusion-

based localization algorithm significantly enhances robot localization accuracy, thereby meeting system application requirements. Therefore, in the field of robot localization, both KF and information fusion-based algorithms are highly effective, providing reliable support for practical system applications. Fig. 15 shows the variation in localization accuracy and error with different numbers of iterations, analyzing the impact of iteration count on the performance of the localization algorithms.

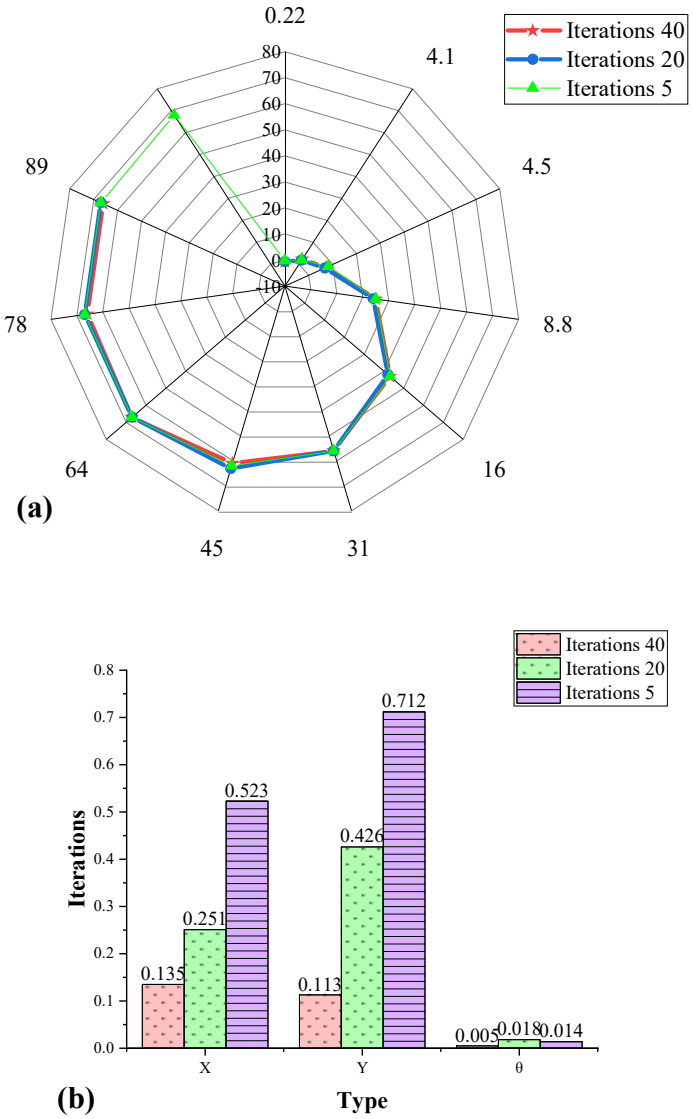


Fig. 15 Impact of iteration number on positioning algorithm performance ((a) comparison of localization curves; (b) error comparison)

Fig. 15 demonstrates that the estimated curves closely align with the actual curves under different numbers of iterations. The error analysis reveals that localization accuracy improves as the number of iterations increases. However, it is important to note that computation time also increases proportionally. Therefore, in practical scenarios, a careful trade-off and selection process is essential to enhance the algorithm's convergence stability while improving localization accuracy. By skillfully choosing the number of iterations, computational costs can be minimized, and optimal results can be achieved while maintaining the desired level of localization accuracy. In summary, optimizing the number of iterations is crucial in robot localization, as it enables the development of more accurate and efficient localization algorithms.

(3) Performance evaluation of financial robot software architecture

To dissect the performance of the financial robot software architecture, a meticulously crafted analytical model is devised. This model is designed to simulate and assess the software's performance across a spectrum of scenarios, targeting key metrics such as response time, communication latency, and localization accuracy. The following outlines the assumptions, parameter settings, and theoretical underpinnings of this model.

(1) Model assumptions:

- System assumptions: Envision the financial robot software architecture as anchored in a cloud computing platform, augmented by a low-latency kernel operating system designed to fulfill stringent real-time processing requirements. This configuration ensures rapid and efficient system response to dynamic operational demands.
- Hardware assumptions: The embedded chip architecture, such as ARM or x86, is hypothesized to deliver substantial computational capacity and versatile adaptability, facilitating high-performance operation within the software framework.
- Communication assumptions: It is assumed that the communication infrastructure employs the ROS (Robot Operating System) framework. Data transmission is managed through various strategies, encompassing persistent and non-persistent connections, to evaluate their impact on system performance.
- Environmental assumptions: The robot is set to navigate within a 100m × 100m operational area, traveling at a steady velocity of 2m/s. Sensor data is sampled every 0.025s, providing critical input for the system's performance evaluation.

(2) Parameter settings:

- Operating system: A low-latency kernel operating system is utilized, featuring an average wake-up latency of 5μs and a maximum latency of 160μs. This choice optimizes the system's ability to handle high-frequency tasks with minimal delay.
- Embedded control board: The PCM-3365EW-S9A1E board is employed, equipped with 4GB DDR3L memory and a maximum power consumption of 8W. It supports a variety of interfaces, including Ethernet, USB, and RS-232, enhancing its connectivity and functionality.
- Sensor settings: The sensors are configured with a maximum observation range of 30m, an observation angle spanning -30° to 30°, and an angular velocity of

20×π/180 rad/s. Control noise is set at 0.6m/s, while observation noise is maintained at 0.1m/s, ensuring accurate environmental data capture.

- Communication strategy: A comparative analysis of different communication strategies (M1 to M5) is conducted, focusing on metrics such as average latency, maximum latency, and the percentage of delays exceeding 0.5ms, 1ms, and 1.5ms. This comparison provides insights into the effectiveness and efficiency of various communication protocols within the system.

The theoretical foundation of this model is anchored in the following principles:

- Cloud computing and information security theory: Cloud computing technology amplifies system computational capacity and fortifies security, ensuring the dependable transmission and processing of financial data. This foundation supports the scalable and secure handling of vast datasets essential for financial operations.
- MSIF theory: By integrating data from multiple sensors, the MSIF theory enhances the precision and stability of robot localization. This theoretical framework enables more accurate and reliable positioning by synthesizing diverse sensor inputs.
- Real-time operating system theory: Employing a low-latency kernel operating system allows for real-time data processing and swift response times. This theory underpins the system's ability to handle dynamic and time-sensitive tasks efficiently.

To validate the efficacy of the analytical model, simulation experiments were conducted, encompassing a range of scenarios and parameter variations to gauge the model's performance. The results of these simulations, reflecting different conditions and parameter adjustments, are detailed in Table 4:

Table 4. Simulation results across various scenarios and parameter variations.

Scenario	Average Latency (μs)	Maximum Latency (μs)	Latency > 0.5ms (%)	Latency > 1ms (%)	Latency > 1.5ms (%)
M1	884.8	172	0.17	0	0
M2	15	160	0.17	0	0
M3	172	172	0.17	0	0
M4	172	172	0.17	0	0
M5	172	172	0.17	0	0

Analyzing the data presented in Table 4 reveals several key insights: Among all communication strategies, the ROS topic approach (M2) exhibits the lowest average and maximum latency, indicating superior real-time performance. Predominantly, the transmission delay remains below 1.5 milliseconds, suggesting that the system maintains reliable and

efficient communication even under heavy loads. The information fusion-based localization algorithm markedly reduces positioning errors, particularly in directional accuracy, maintaining deviations close to zero degrees.

To verify the accuracy and practicality of the analytical model, a comparative analysis between simulation results and experimental outcomes is conducted. Both the simulation and experimental platforms utilized the embedded control board PCM-3365EW-S9A1E, operating with a low-latency kernel operating system. A meticulous comparison of simulation and experimental data reveals a high degree of consistency in average and maximum latency, affirming the model's precision. In terms of localization accuracy, the information fusion-based algorithm demonstrated high precision in both simulation and practical experiments, validating its practical utility. The combined results from simulations and experiments lead to the conclusion that the proposed analytical model effectively assesses the performance of the financial robot software architecture and provides actionable improvement recommendations. By optimizing system design and algorithms, further enhancements in the security and accuracy of financial robot software can be achieved, offering robust support for enterprise financial management.

4.5 Discussion

In the exploration of financial robot software architecture and self-localization algorithms, a novel approach featuring cloud-based information security architecture and a multi-sensor information fusion self-localization algorithm has been proposed. The outcomes of this research, validated through performance testing and expert evaluations, reveal a remarkable advancement over previous systems. Under high system loads, the designed system demonstrates an average latency of 15μs and a maximum latency of 172μs, reflecting substantial improvements. The low-latency scheduling strategy of ROS topics shows an average delay of 884.8μs, while ROS services exhibit consistent performance with persistent connections. Notably, 99.83% of transmission delays are below 1.5ms, underscoring the system's reliability and efficiency in communication. When compared to prior studies, the innovation and advantages of this research become apparent. Chang (2022) proposed a multi-fusion perception architecture aimed at enhancing autonomous navigation, intelligent control, and detection capabilities for autonomous mobile robots [31]. Although the multi-machine and heterogeneous coordination framework performed well in indoor environments, it did not address latency performance under high loads. In contrast, the system presented here maintains low latency even under heavy loads, highlighting the benefits of cloud computing in boosting system performance. Chen (2023) examined machine vision and multi-sensor fusion for drone patrol path planning, emphasizing ultrasonic applications in obstacle avoidance and presenting a comprehensive hardware and software system [32]. However, the focus was on drone path planning and obstacle avoidance, not on self-localization for financial robots in complex environments. The multi-sensor fusion approach used in this research effectively reduces localization errors, enabling precise navigation in intricate settings. Wong et al. (2023) introduced a vision-based system for detecting foreign objects inside marine vessels, employing Mahalanobis distance-driven anomaly detection and human-machine collaboration for enhanced accuracy [33]. Although this system excels in marine environments, its reliance on visual sensors may present limitations in the complex indoor environments of financial robots. The multi-sensor fusion employed here improves localization precision and relia-

bility across varied sensor data. Matsuda et al. (2022) proposed a multi-robot mutual localization relay method to enhance indoor environmental monitoring accuracy through interactions between parent and child robots [34]. While this approach achieved notable results in group robot localization, its complex architecture may not be advantageous for single financial robot applications. The MSIF-based self-localization algorithm introduced here is simpler and more efficient, suitable for dynamic environments. Aljohani et al. (2023) developed an IoT and machine learning-based framework for natural disaster management, demonstrating excellence in urban flood prediction with various models to enhance performance and reliability [35]. Despite its significance in disaster management, it diverges from the self-localization needs of financial robots in secure environments. The multi-sensor fusion localization algorithm presented here offers effective solutions for improving localization accuracy and system reliability, particularly for risk assessment in enterprise financial management. In conclusion, the innovations in the financial robot software architecture and self-localization algorithm are evident not only in system design and performance enhancements but also in the effective resolution of localization accuracy issues in complex environments through multi-sensor information fusion. Compared to previous studies, this research stands out for its exceptional performance under high loads and its proven advantages in functionality and user-friendliness. The detailed discussion and comparison affirm the significant practicality and innovativeness of the research outcomes, providing robust technical support for enterprise financial management.

5. CONCLUSION

The MSIF-based robot self-localization algorithm facilitates self-management and path planning for the robot, resulting in enhanced system efficiency and performance. Considering factors such as system performance, data security, user experience, and cost-effectiveness, this comprehensive solution serves as a reference for the research and development of financial robots. The proposed application scheme covers various aspects, including financial data management and processing, automated financial operations, information security risk assessment, financial decision support, and mobile terminal access. These functionalities provide efficient, secure, and reliable solutions for enterprise financial management. Notably, the software architecture of financial robots enables the automation of financial operations, such as invoice processing, bill management, and reimbursement approval. Integration with information security technology ensures the reliability and security of financial operations, reducing errors and delays associated with manual operations and improving the efficiency and accuracy of financial processing.

This work makes notable progress in financial robot software architecture and localization algorithms. However, maintaining data integrity, confidentiality, and availability remains a significant challenge due to data sensitivity and ongoing technological advancements. In the future, it is crucial to develop more robust security mechanisms, such as encryption algorithms and firewalls, to bolster data security and confidentiality. Strengthening identity authentication and access control, implementing encryption algorithms, introducing firewalls, and conducting vulnerability scans are among the measures that can improve data security. Integrating these security mechanisms will contribute to enhancing data security and confidentiality. However, it is important to note that the security landscape is ever-changing, and continuous efforts are required to stay ahead of emerging

threats. Ongoing technology updates and comprehensive security awareness education are essential to ensure that the systems remain resilient against new vulnerabilities and evolving attack vectors.

Acknowledgements

The work was supported in part by Scientific Research Project of Hunan Provincial Department of Education(No.23A0659). Research on the gain effect of school-enterprise collaborative education in the mixed teaching mode ——Using “Computers” Taking the course as an example (No. ND233163).

REFERENCES

1. A. Couturier, and M. A. Akhloufi, A review on absolute visual localization for UAV. *Robotics and Autonomous Systems*, Vol. 135, 2021, pp. 103666.
2. M. R. Bachute, and J. M. Subhedar, Autonomous driving architectures: insights of machine learning and deep learning algorithms. *Machine Learning with Applications*, Vol. 6, 2021, pp. 100164.
3. R. C. Shit, Precise localization for achieving next-generation autonomous navigation: State-of-the-art, taxonomy and future prospects. *Computer Communications*, Vol.160, 2020, pp. 351-374.
4. W. Mao, Z. Liu, H. Liu, F. Yang, and M. Wang, Research progress on synergistic technologies of agricultural multi-robots. *Applied Sciences*, Vol. 11, No. 4, 2021, pp. 1448.
5. M. Khalid, K. Wang, N. Aslam, Cao, Y., Ahmad, N., & Khan, M. K. (2021). From smart parking towards autonomous valet parking: A survey, challenges and future Works. *Journal of Network and Computer Applications*, *175*, 102935.
6. N. Ding, C. Peng, M. Lin, and C. Wu, "A comprehensive review on automatic mobile robots: applications, perception, communication and control," *Journal of Circuits, Systems and Computers*, Vol. 31, no. 08, 2022, pp. 2250153.
7. S. Han, J. Zhang, Z. S. Shaikh, J. Wang, and W. Ren, "Four-Dimensional (4D) Millimeter Wave-Based Sensing and Its Potential Applications in Digital Construction: A Review," *Buildings*, Vol. 13, No. 6, 2023, pp. 1454.
8. S. S. Binyamin and S. Ben Slama, "Multi-Agent Systems for Resource Allocation and Scheduling in a Smart Grid," *Sensors*, Vol. 22, No. 21, 2022, pp. 8099.
9. M. Zdravković, H. Panetto, and G. Weichhart, "AI-enabled enterprise information systems for manufacturing," *Enterprise Information Systems*, Vol. 16, No. 4, 2022, pp. 668-720.
10. Y. Zhang, C. Jiang, B. Yue, J. Wan, and M. Guizani, "Information fusion for edge intelligence: A survey," *Information Fusion*, Vol. 81, 2022, pp. 171-186.
11. J. L. Carvalho, P. C. Farias, and E. F. Simas Filho, "Global Localization of Unmanned Ground Vehicles Using Swarm Intelligence and Evolutionary Algorithms," *Journal of Intelligent & Robotic Systems,* Vol. 107, No. 3, 2023, pp. 45.
12. A. Koval, S. Karlsson, and G. Nikolakopoulos, "Experimental evaluation of autonomous map-based spot navigation in confined environments," *Biomimetic Intelligence and Robotics*, Vol. 2, No. 1, 2022, pp. 100035.

13. A. Sesyuk, S. Ioannou, and M. Raspopoulos, "A survey of 3D indoor localization systems and technologies," *Sensors*, Vol. 22, No. 23, 2022, pp. 9380.
14. A. Dahiya and B. B. Gupta, "A reputation score policy and Bayesian game theory based incentivized mechanism for DDoS attacks mitigation and cyber defense," *Future Generation Computer Systems*, Vol. 117, 2021, pp. 193-204.
15. A. Gaurav, B. B. Gupta, and P. K. Panigrahi, "A novel approach for DDoS attacks detection in COVID-19 scenario for small entrepreneurs," T*echnological Forecasting and Social Change*, Vol. 177, 2022, pp. 121554.
16. S. Dragule, S. G. Gonzalo, T. Berger, and P. Pelliccione, "Languages for specifying missions of robotic applications," *Software Engineering for Robotics*, 2021, pp. 377-411.
17. T. Liu, H. Kang, and C. Chen, "ORB-Livox: A real-time dynamic system for fruit detection and localization," *Computers and Electronics in Agriculture*, Vol. 209, 2023, pp. 107834.
18. A. Ayadi, O. Ghorbel, M. S. BenSalah, and M. Abid, "A framework of monitoring water pipeline techniques based on sensors technologies," *Journal of King Saud University-Computer and Information Sciences*, Vol. 34, No. 2, 2022, pp. 47-57.
19. S. Jia, C. Liu, H. Wu, D. Zeng, and M. Ai, "A cross-correction LiDAR SLAM method for high-accuracy 2D mapping of problematic scenario," *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 171, 2021, pp. 367-384.
20. D. Khan, Z. Cheng, H. Uchiyama, S. Ali, M. Asshad, and K. Kiyokawa, "Recent advances in vision-based indoor navigation: A systematic literature review," *Computers & Graphics*, Vol. 104, 2022, pp. 24-45.
21. P. Dasler, S. Malik, and M. L. Mauriello, “Just Follow the Lights”: A Ubiquitous Framework for Low-Cost, Mixed Fidelity Navigation in Indoor Built Environments," *International Journal of Human-Computer Studies*, Vol. 155, 2021, pp. 102692.
22. S. Khalid, A. Alam, M. Fayaz, F. Din, S. Ullah, and S. Ahmad, "Investigating the effect of network latency on users’ performance in Collaborative Virtual Environments using navigation aids," *Future Generation Computer Systems*, Vol. 145, 2023, pp. 68-76.
23. T. A. Q. Tawiah, "A review of algorithms and techniques for image-based recognition and inference in mobile robotic systems," *International Journal of Advanced Robotic Systems*, Vol. 17, No. 6, 2020, pp. 1729881420972278.
24. A. G. Mohapatra, B. Keswani, S. Nanda, A. Ray, A. Khanna, D. Gupta, and P. Keswani, "Precision local positioning mechanism in underground mining using IoT-enabled WiFi platform," *International Journal of Computers and Applications*, Vol. 42, No. 3, 2020, pp. 266-277.
25. M. M. Rathore, S. Attique Shah, A. Awad, D. Shukla, S. Vimal, and A. Paul, "A cyber-physical system and graph-based approach for transportation management in smart cities," *Sustainability*, Vol. 13, No. 14, 2021, pp. 7606.
26. P. L. Sanchez-Gonzalez, D. Díaz-Gutiérrez, T. J. Leo, and L. R. Núñez-Rivas, "Toward digitalization of maritime transport?," *Sensors*, Vol. 19, No. 4, 2019, pp. 926.
27. X. Yu, W. He, X. Qian, Y. Yang, T. Zhang, and L. Ou, "Real-time rail recognition based on 3D point clouds," *Measurement Science and Technology*, Vol. 33, No. 10, 2022, pp. 105207.

28. E. Karaaslan, B. Sen, T. Ercan, H. Laman, and J. Pol, "Reading vehicular messages from smart road signs: a novel method to support vehicle-to-infrastructure in rural settings," *Transportation Research Record*, Vol. 2675, No. 11, 2021, pp. 150-158.
29. F. D. Von Borstel, J. F. Villa-Medina, and J. Gutiérrez, "Development of mobile robots based on wireless robotic components using UML and hierarchical colored Petri nets," *Journal of Intelligent & Robotic Systems*, Vol. 104, No. 4, 2022, pp. 70.
30. E. I. Adegoke, J. Zidane, E. Kampert, C. R. Ford, S. A. Birrell, and M. D. Higgins, "Infrastructure Wi-Fi for connected autonomous vehicle positioning: A review of the state-of-the-art," *Vehicular Communications*, Vol. 20, 2019, pp. 100185.
31. M. W. Chang, "Real-time multi-fusion perceptron architecture for autonomous drones," *Journal of the Chinese Institute of Engineers*, Vol. 45, No. 7, 2022, pp. 621-631.
32. X. Chen, "UAV patrol path planning based on machine vision and multi-sensor fusion," Open Computer Science, Vol. 13, No. 1, 2023, pp. 20220276.
33. B. Wong, W. Marquette, N. Bykov, T. M. Paine, and A. G. Banerjee, "Human-assisted robotic detection of foreign object debris inside confined spaces of marine vessels using probabilistic mapping," *Robotics and Autonomous Systems*, Vol. 161, 2023, pp. 104349.
34. T. Matsuda, Y. Kuroda, R. Fukatsu, T. Karasawa, M. Takasago, and K. Morishita, "A mutual positioning relay method of multiple robots for monitoring indoor environments," *International Journal of Advanced Robotic Systems*, Vol. 19, No. 5, 2022, pp. 17298806221129842.
35. F. H. Aljohani, A. A. Abi Sen, M. S. Ramazan, B. Alzahrani, and N. M. Bahbouh, "A Smart Framework for Managing Natural Disasters Based on the IoT and ML," *Applied Sciences*, Vol. 13, No. 6, 2023, pp. 3888.