Available Online: 12 December 2025 DOI: 10.54254/2977-3903/2025.30494

Research on production line simulation method based on digital twin

***Xiaodie Duan*[1], *Yazhou Chen*[2], *Wasim M.K. Helal*[1, 3*]**

[1]School of Mechanical and Electrical Engineering, Quanzhou University of Information Engineering, Quanzhou, China
[2]College of Marine Equipment and Mechanical Engineering, Jimei University, Xiamen, China
[3]Department of Mechanical Engineering, Faculty of Engineering, Kafrelsheikh University, Kafrelsheikh, Egypt

*Corresponding Author. Email: dr_waseemhelal@eng.kfs.edu.eg

Abstract. Addressing the challenge of insufficient behavioral modeling accuracy in digital twin-driven production line simulation, this study targets the inherent limitations of "state explosion" and "behavioral black box" in traditional Finite State Machines when describing complex, concurrent equipment behaviors. An innovative simulation method is proposed, centering on an Improved Finite State Machine (IFSM) that formally incorporates a set of intermediate states to explicitly characterize component-level collaborative workflows and key transition steps. This is integrated within a three-tier modeling architecture to enable modular encapsulation and reuse of behavioral logic. Experimental validation through a stacker case study demonstrates that the implemented prototype system effectively simulates normal, disturbed, and fault conditions. Compared to traditional FSMs, the proposed method reduces model transition rule complexity and achieves component-level fault localization accuracy while maintaining real-time performance. This study proves the method's feasibility and superiority in enhancing the correctness, stability, and maintainability of production line simulations, providing a viable technical pathway for constructing high-fidelity and high-reliability digital twin systems.

Keywords: digital twin, finite state machine, production line simulation, behavioral model, real-time simulation

1. Introduction

The wave of Industry 4.0 and smart manufacturing is propelling the manufacturing sector toward a data-driven paradigm [1]. In this transformation, digital twins [2, 3] emerge as a pivotal enabling technology. By establishing high-fidelity dynamic mapping between physical entities and virtual models, they provide an unprecedented framework for real-time monitoring, in-depth analysis, and proactive decision-making in production systems [4, 5]. The core value of a truly effective digital twin lies in its behavioral model, a computational framework that accurately captures the dynamic response logic of physical entities under normal, disturbed, and fault conditions [6]. The precision of this behavioral model directly determines the effectiveness of digital twins in advanced applications such as predictive analytics, optimization, and autonomous decision-making [7].

Despite the significant attention the concept of digital twins has garnered, a gap often exists between its theoretical potential and practical implementation. Current efforts predominantly focus on 3D visualization and data dashboards, with notable shortcomings in accurately modeling the underlying production logic [8]. This discrepancy between rich geometric representation and insufficient behavioral fidelity has been identified as a critical limitation for digital twins in cyber-physical production systems [9]. The challenge is particularly acute when describing the discrete event logic, concurrent operations, and complex state transitions inherent to automated production lines [10].

As a classical tool for discrete event system modeling, Finite State Machines (FSMs) have been widely adopted in this field due to their formal semantics and clear state transition logic [11, 12]. However, traditional FSMs struggle with the complexity of modern automated production equipment, primarily due to two inherent limitations [13, 14]; (1) "State explosion" caused by combinatorial complexity: The overall state space of a system grows exponentially with the number of its components, leading to bloated and unmaintainable models [15]; (2) "Behavioral black box" caused by obscured intermediate logic: Critical transition processes and synchronization conditions are hidden within complex transfer conditions, reducing model readability and making it difficult to pinpoint the root cause of failures [16]. For instance, a failure in a stacker's operation might only be represented by a generic "fault" state, obscuring whether the issue lies with the X-axis positioning, the gripper, or another specific component.

Tao et al. [4] and Liu et al. [7] have underscored the necessity of high-fidelity behavioral models, while Kritzinger et al. [10] have highlighted that many implementations remain at a descriptive level, lacking deep behavioral simulation. Zhuang et al. [14] developed a digital twin-based framework for assembly shop-floors using state machines, and Shangguan et al. [16] integrated FSMs for fault diagnosis in satellite systems. However, these approaches often do not explicitly formalize the intermediate states of component collaboration, limiting diagnostic granularity. From architectural and implementation perspectives, Redelinghuys et al. [17] proposed a multi-layer digital twin architecture, and Kampa [18] demonstrated the use of game engines for realistic simulation. While valuable, these studies often rely on conventional logic for behavior modeling, failing to systematically resolve the core issues of state explosion and behavioral opacity in FSMs.

Consequently, a critical research gap remains. There is a conspicuous absence of a universal and structured modeling framework capable of achieving fine-grained behavioral logic depiction at the component level while simultaneously preserving model clarity and maintainability at the system level. This deficiency manifests most acutely in the context of Finite State Machine (FSM) applications, where the core challenge of explicitly modeling and managing the numerous intermediate transition states generated throughout the equipment life-cycle remains systematically unaddressed. Without a formal mechanism to capture these states, model transparency and diagnostic granularity are inherently limited. Furthermore, the construction of a well-layered, highly reusable behavioral model architecture founded upon this explicit state management represents an unresolved endeavor. The absence of such an integrated, structured approach is widely recognized as a significant impediment to realizing the full potential of digital twins in cyber-physical production systems [19-21].

To bridge this gap, this study proposes a digital twin behavior modeling method for production line simulation based on an Improved Finite State Machine (IFSM). The present work is structured to three key points: at the Model Level: a set of intermediate states (x_i) is introduced into the FSM to create an IFSM, explicitly characterizing key transition steps and internal concurrent logic, thereby enhancing model transparency and diagnosability; at the Framework Level: a three-tiered modeling architecture ("Component-Equipment-System") are designed where the IFSM is applied progressively across layers, enabling modular encapsulation and reuse of behavioral logic to effectively control system complexity; at the System Validation Level: an integrated digital twin prototype system is implemented and validated. The developed system integrates industrial Internet of Things (IoT), Message Queuing Telemetry Transport (MQTT) communication, and the Unity3D engine. Using a stacker as a case study, the advantages of our method in terms of model complexity, fault localization accuracy, and real-time performance are demonstrated.

2. Digital twin behavior modeling and finite state machines

2.1. Behavioral modeling in digital twins

The behavioral model of digital twins serves as the core mechanism enabling virtual-real interaction and intelligent decision-making. In recent years, behavioral modeling research has evolved from early geometry-and physics-driven approaches to progressively characterize system dynamics. Based on modeling principles, existing studies can be broadly categorized into two types: data-driven modeling and first-principles modeling.

Data-driven modeling methodologies primarily leverage machine learning and artificial intelligence technologies to identify system behavior patterns from historical and real-time data. For instance, deep learning models are employed to predict equipment remaining service life [15], while reinforcement learning is applied to optimize dynamic scheduling and control strategies [22]. The key advantage of these approaches lies in their exceptional nonlinear fitting capabilities, enabling the discovery of complex correlations that are difficult to obtain through mechanistic analysis. However, they are often regarded as "black-box" models, characterized by opaque decision-making processes and heavy reliance on large volumes of high-quality training data. These limitations result in suboptimal performance under data scarcity or novel operational conditions [23].

In contrast, first-principles modeling (or mechanistic modeling) constructs models based on a deep understanding of a system's physical laws and operational logic. Typical examples include discrete event system specifications, Petri nets, and finite state machines. These models offer significant advantages such as high transparency and strong interpretability, making them particularly suitable for describing manufacturing systems with clear rules and state transition logic [24].

While data-driven approaches are gaining momentum, first-principles models particularly those that clearly articulate discrete logical behaviors of systems remain indispensable for ensuring the determinism, predictability, and traceability of digital twin operations. This study focuses on the precise modeling of discrete event logic in production lines, with particular emphasis on refining and enhancing first-principles modeling frameworks.

2.2. Application and evolution of finite state machines in intelligent manufacturing

FSM is a classic and powerful first-principles modeling tool that continues to play a vital role in intelligent manufacturing. Its clear formal semantics make it particularly suitable for modeling equipment operation modes and production order status transitions. By defining states, transition rules, and actions, the FSM can clearly describe system state transitions and capture critical system behaviors.

FSM consists of three components: a finite set of states, an input set, and a set of state transition rules. Its formal definition is:

$$M = (Q, \sum, \Delta, s_0, F) \tag{1}$$

In Equation (1), Q denotes the set of states; $\sum$ denotes the set of events; Δ denotes the state transition function; s_0 denotes the set of initial states; F denotes the set of final states.

In recent years, the application of FSM has shown a trend of deep integration with specific industrial scenarios. In fault diagnosis, Shangguan et al. [14] successfully integrated state machines into digital twins of complex equipment, achieving precise fault localization and predictive maintenance.

However, traditional FSMs increasingly demonstrate inherent limitations when modeling the complex, concurrent, and fine-grained behaviors typical of modern automated production lines. To overcome these constraints, researchers have primarily expanded FSMs in two key directions:

One approach involves introducing hierarchical and concurrent concepts. The classical Harel Statechart theory provides a theoretical foundation for this, managing complexity through hierarchical states and concurrent orthogonal regions. Inspired by this, many modern modeling languages and frameworks (such as Unified Modeling Language (UML) State Machine) have built-in these features. Another direction involves formal expansion and customization. By introducing extended variables and conditional logic, they significantly enhanced the model's descriptive capabilities.

The Current study demonstrates that integrating FSM with digital twin technology for production line behavior simulation represents a valuable and dynamic research direction. However, our in-depth analysis reveals a critical research gap that remains unaddressed:

This study primarily focuses on applying FSM for high-level system control and monitoring, while existing studies on detailed modeling have failed to provide systematic solutions that balance component-level behavioral characterization with holistic model structuring. Specifically, existing approaches inadequately address the explicit and formal modeling of critical intermediate states generated during collaborative operations of equipment components. These intermediate states are essential for achieving precise fault diagnosis, process traceability, and simulation analysis. The lack of dedicated management for intermediate states leads to two outcomes when dealing with complex equipment: models either revert to the "black box" logic of traditional FSMs, or while implicitly incorporating these states, they lack unified modeling frameworks, resulting in difficulties in model comprehension, maintenance, and reuse.

To address this gap, this study proposes an IFSM and its hierarchical modeling framework, with the following innovations: By formally defining the intermediate state (x_i), the transparency issue of behavioral logic is fundamentally resolved.

3. IFSM mathematical model and hierarchical modeling framework

The framework is designed to systematically address the state explosion and logical opacity issues encountered by traditional FSMs when describing complex production line behaviors. Complex systems, especially in manufacturing, are often managed through hierarchical decomposition, which separates concerns and reduces model complexity by abstracting system details at different levels [25]. Such layered architectural patterns are equally fundamental to structuring scalable digital twins [5]. Inspired by this established principle, we construct a three-level hierarchical modeling architecture (component-device-system) and apply the improved finite state machine at each level.

3.1. Formal definition of the Improved Finite State Machine (IFSM)

Traditional FSMs often provide overly broad state definitions when modeling devices with complex internal processes and concurrent behaviors, failing to adequately capture the coordinated operations of their internal components. To address this, we an IFSM has been developed by extending the classical FSM framework. The formal definition of IFSM is structured as a six-tuple: This paper improves the FSM by defining it as an IFSM (Improved Finite State Machine), with the following key enhancements:

(1) A variable is added to the FSM "x_i" mathematical model to x_i represent the intermediate state. The variable is combined with the initial state and the terminal state to describe all the state sets of the device.

(2) The production line is horizontally divided into multiple modules and vertically divided into different levels. Each behavior object is independently established with a finite state machine model, and multiple device-level models are composed into a system-level model, s_0 that the behavior modeling of the digital twin production line is clear.

(3) Propose a highly structured implementation framework, encapsulate the motion components as the research object, design the simulation rules (event-triggered scheduling and conversion algorithm), process the state machine conversion, realize the

high reuse of finite state machine, and enhance the robustness and maintainability of the system.

The IFSM mathematical model is as follows:

$$M = (Q, \sum, \Delta, s_0, x_i, F) \tag{2}$$

In Equation (2), Q denotes the set of states; $\sum$ denotes the set of events; Δ denotes the state transition function; s_0 denotes the set of initial states; F denotes the set of final states; $x_i \in Q$ above and will not s_0 be F repeated here. The intermediate state x_i set represents the transitional states between the initial state and s_0 the final state F. These intermediate states accurately describe the specific behavioral logic of internal components x_i within an object. When it is combined with the initial state and final state , they establish an object behavior model and modularize the object's overall structure, facilitating management of multiple objects. Additionally, they enable timely feedback on component anomalies and pinpoint specific faulty devices or abnormal states.

4. System implementation and experimental analysis

4.1. Production line equipment behavior modeling

The material system is a vital component of the production line, with materials' lifecycle spanning the entire production process. This section demonstrates the IFSM behavior modeling through the material feeding process in the system.

As shown in Figure 1, the stacker crane consists of a three-axis module and clamping jaws for material handling. When it is viewed as a system, the three-axis module's X, Y, and Z axes, along with the clamping jaws, are considered components. The operational sequence is as follows:

(1) Receiving the material withdrawal instruction and arrive at the material withdrawal place;

(2) The gripper reaches the material taking position and picks up the material;

(3) Picking up the material and transport it to the discharge point;

(4) Releasing the clamp and drop the material;

(5) Returning to the initial state.

Figure 1. Pile mechanism

The IFSM model for each component in the stacker crane is shown in Figure 2.

(1) The first state (s_0) indicates the standby idle waiting state, where the stacker crane has not received an order command from the Human Machine Interface (HMI) or Manufacturing Execution System (MES), and remains in the initial state.

(2) The second state is operational mode (s_x, indicating material retrieval and discharge states), involving material retrieval from the automated warehouse and discharge at a preset location. When the event HMI or MES order == true is triggered, the

stacker crane initiates operation to complete both retrieval and discharge. During discharge, the event Material Placement == true is activated, sending a feedback signal to the stacker crane indicating the operation is complete.

(3) The third state is the end state (F). When the stacker crane completes its operation, it triggers the Equipment Placement == true event, which drives the stacker crane back to the initial state.

As depicted in Figure 2, the IFSM explicitly models the sequential and concurrent movements of the X, Y, Z axes and the gripper through a series of intermediate states (x_1 to x_{13}), which are triggered by specific events (ε_1 to ε_9). For instance, starting from the initial state (s_0), the trigger HMI or MES order == true (event ε_1) activates the X-axis movement to the pickup point. This is followed by the coordinated advancement of the Y and Z axes (states x_2, x_3) upon verification of the X-axis's arrival (event ε_2). This fine-grained modeling delineates each step of the operation, such as gripper deployment (x_4), material retrieval confirmation (ε_4), and the subsequent sequence for transportation and discharge. The explicit inclusion of these intermediate states is pivotal for enabling precise fault localization, as any deviation or failure can be traced to a specific state and component.

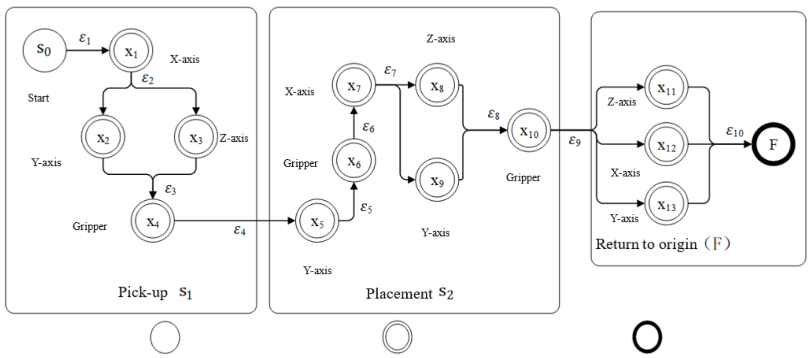


Figure 2. IFSM model for palletizing machine components

When the device fails to operate normally, its operational status is shown in Figure 3, where s_3 indicates an unplanned shutdown and s_4 denotes a pause.

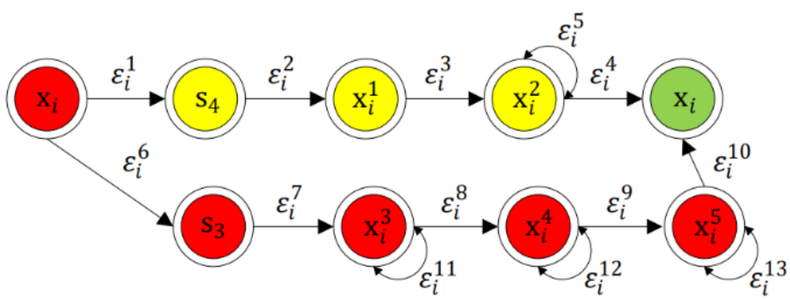


Figure 3. IFSM behavior model during abnormal conditions

When equipment suddenly stops ε_i^1 working, the system triggers an event to perform anomaly detection. If the cause is material shortage, it transitions ε_i^2 to state s_4. State s_4 indicates the equipment x_i^1 is in pause ε_i^3 mode awaiting x_i^2 material replenishment ε_i^4. If material is insufficient, it alerts relevant personnel ε_i^5 to load materials. The system then ε_i^6 loads materials for designated ε_i^7 personnel. It verifies material x_i^3 quantities, performs loading operations, and completes ε_i^{11} loading to transition back ε_i^8 to normal working state x_i. If x_i^4 loading is in progress ε_i^{12}, it remains in this state. ε_i^9 State s_3 signifies equipment shutdown x_i^5. Anomaly detection reveals faults, with ε_i^{13} specific fault identification ε_i^{10} followed by maintenance by personnel or network systems to determine causes. If detection fails, maintenance continues. Identified equipment requires repair, which is completed or ongoing. Maintenance concludes with fault cause documentation and time recording. Logging processes are ongoing until completion, after which normal state x_i is executed. The improved finite state machine, by adding intermediate states, provides clearer descriptions of production line workflows, equipment lifecycle behaviors, and inter-level relationships. When anomalies occur during production, it enables precise identification of problematic states or faulty equipment.

4.2. Production line system behavior modeling

The behavioral logic of system modules exhibits similarities across device-level and system-level implementations, with distinctions lying in triggering conditions and corresponding transition states. This paper establishes a universal finite state machine model by abstracting common characteristics. Production line equipment operates in various states during normal operation, including unplanned shutdowns (e.g., failure-induced halts), scheduled shutdowns (post-task completion), and sudden disturbances (e.g., order-induced pauses). Sudden disturbance states require specific order disruption rules to restore equipment functionality, as illustrated in Figure 4.

s_0: The states are defined as follows: Initial state. s_1: Indicates equipment or systems undergoing setup. s_2: Idle state: Signals completed preparations s_3 with all devices in standby mode. Production s_4^5 state: Denotes equipment entering active s_4^6 operation. Disturbance state: Occurs due to material shortages, order disruptions, or schedule changes. Planned shutdown: Marks completed production. Unplanned shutdown: Results from equipment failures during operation. The transition probabilities are: ε_1 (preparation completion), ε_2 (order received), ε_3 (operation completion/failure), ε_4 (processing finished/order completed), ε_5 (equipment shutdown), ε_6 (equipment malfunction), ε_7 (repair completion), ε_8 (production suspension), ε_9 (equipment restart after replenishment).

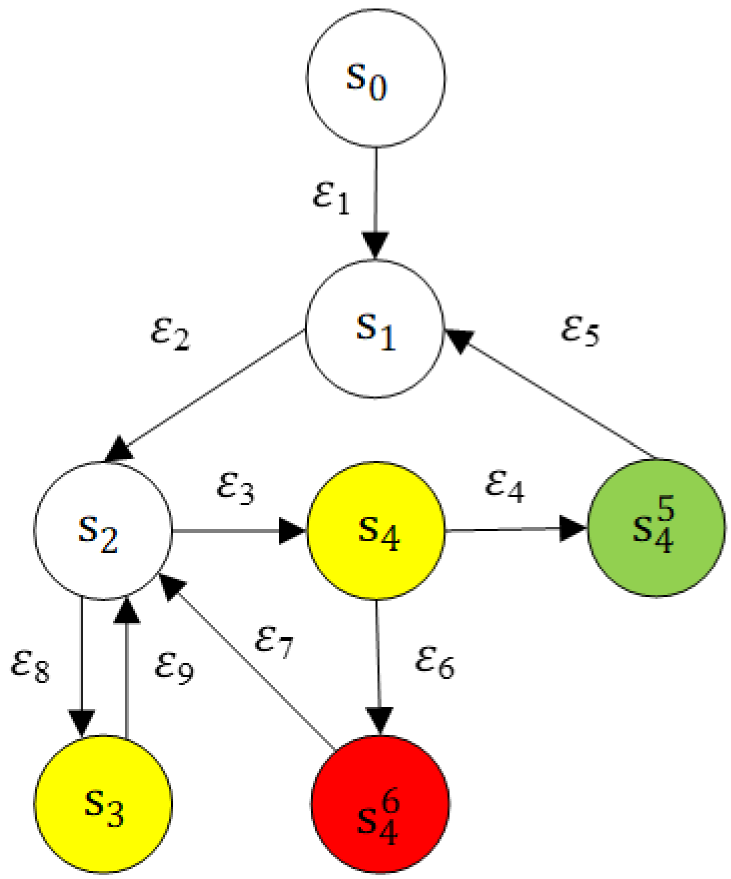


Figure 4. Finite state machine behavior model

4.3. Design of IFSM-based behavioral model classes

The IFSM behavioral model for C#-designed automated production lines defines operational rules and logic between production line components, ensuring consistency between physical entities and virtual models. The IFSM model comprises four modules: System Management, State Management, Transition Condition Management, and Motion Object Management. The System Management class handles state addition, deletion, and transitions. The State Management class stores states and manages transition conditions and internal logic. The Transition Condition Management class implements transition mechanisms, while the Motion Object Management class inherits from the State Management class to realize motion behaviors. These four modules collectively implement the overall object behavior logic. The inter-module relationships ensure coordinated collaboration to achieve comprehensive object behavior implementation (specific implementation methods are detailed in the appendix). The relationships between module classes are illustrated in Figure 5.

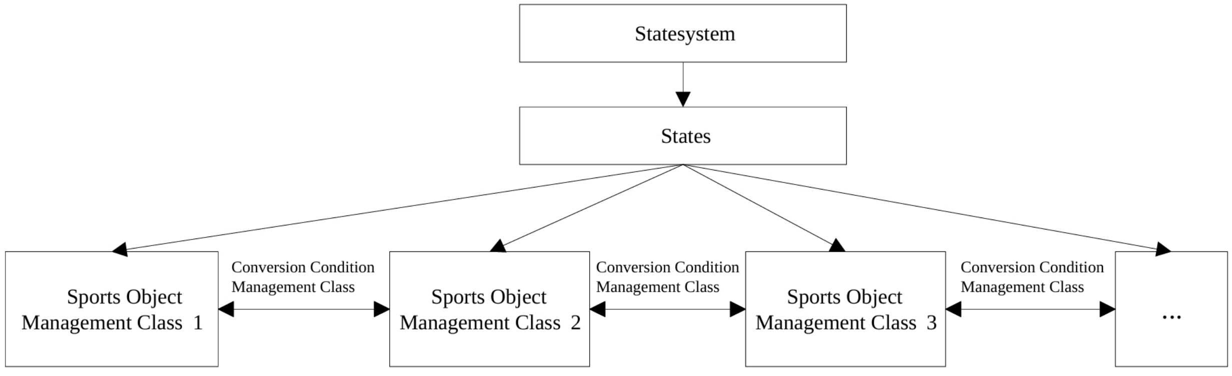
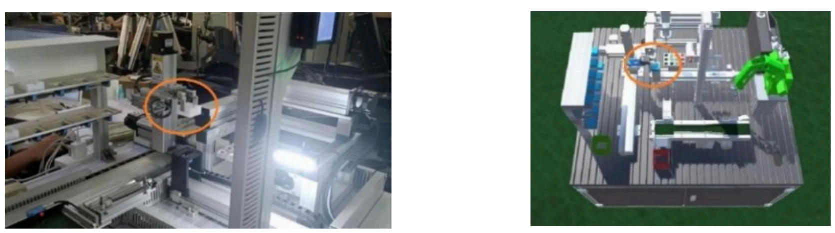


Figure 5. Module class relationships

4.4. Testing and verification

The industrial gateway collects data from various equipment including processing systems, conveyor belts, storage facilities, HMI devices, and barcode scanners to simulate production lines. This data is then uploaded to servers via the MQTT JSON protocol for storage, analysis, and application. The client application, built using the Unity 3D engine and MQTT protocol, enables device data subscription and message processing through M2Mqtt library installation, client creation and configuration, server connection, topic subscription, and message reception/publishing. The LitJson library is employed for data parsing, converting messages into JsonData objects for real-time monitoring and anomaly handling within the Unity 3D engine. During verification, the process begins with activating smart manufacturing equipment and servers, followed by launching the Unity3D engine on the virtual environment to establish server connections. MQTTX client configurations are then tested to ensure real-time operation.

After successful test communication, configure the product type and place an order in the Unity3D client. The system will receive and analyze real-time device status and data, then map them to the virtual model for synchronization as shown in Figure 6, where circles indicate the real-time operational status of the device.



(a) physical entity (b) virtual entity

Figure 6. Real-time virtual-to-physical mapping

To ensure that the behavior simulation service module in the digital twin system can simulate as expected and produce accurate and reliable results it should be tested and verified. This service encompasses three core components: real-world interface, node interface, simulation model, motion event management, and transition condition management, which work collaboratively to execute simulations. The simulation model interface, motion event management interface, and transition condition management interface provide functionalities for adding, deleting, and refreshing elements. The simulation model interface also supports saving, initiating, pausing simulations, and downloading simulation programs to hardware devices. Motion events are managed through C# programming, while transition condition management enables serial or parallel device behavior configurations. The specific behavior simulation services are shown in Figure 7 and Figure 8.

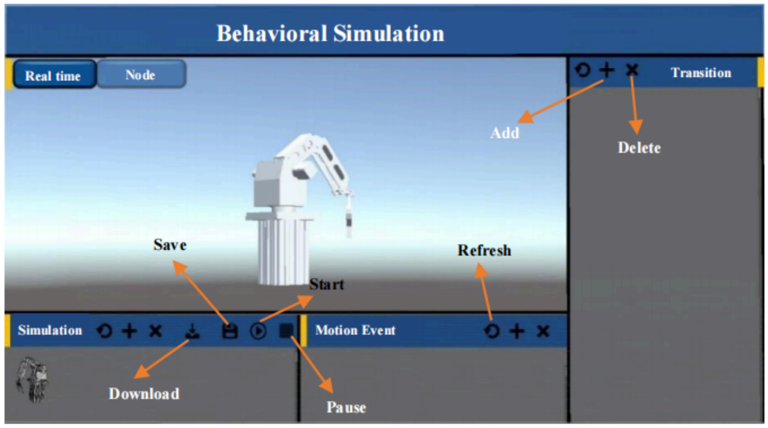


Figure 7. Behavior simulation real scene window

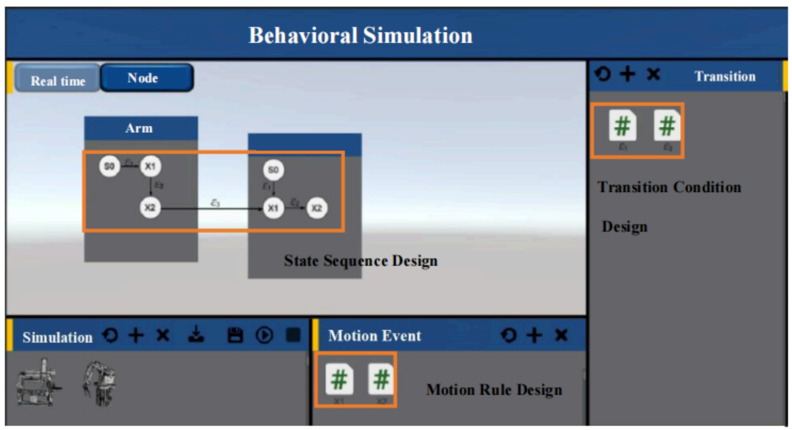


Figure 8. Behavior simulation node window

5. Conclusion

This paper addresses the challenge of insufficient behavioral modeling accuracy in digital twin-driven production line simulations. It thoroughly analyzes the two major limitations of traditional finite state machines in describing complex and concurrent equipment behaviors: "state explosion" and "logical black box". To overcome these challenges, the study proposes an innovative and systematic solution.

The main investigations of this paper can be summarized as follows:

An IFSM model has been developed. By formally introducing an intermediate state set (x), this model enables detailed deconstruction and explicit representation of the collaborative workflow among device components. This innovation fundamentally improves the transparency and diagnosability of behavioral models, allowing faults to be precisely traced to specific components and process steps.

A hierarchical framework for digital twin behavior modeling was established in this study. The proposed three-tier architecture comprising component, device, and system levels systematically incorporates the IFSM at each tier to effectively manage complex system behaviors. Through focus separation and modular encapsulation, the framework significantly improves model reusability, maintainability, and scalability.

Furthermore, an integrated digital twin prototype system was implemented and validated. The developed system integrates industrial IoT, MQTT communication, and the Unity3D engine. In a case study involving a typical stacker crane, along with systematic comparative experiments, the proposed method demonstrated clear advantages. Experimental results confirm that the IFSM not only maintains real-time performance but also reduces the complexity of model conversion rules and achieves component-level fault localization accuracy, thereby offering robust support for precision operations and maintenance in production lines.

References

[1] Zhong, R. Y., Xu, X., Klotz, E., & Newman, S. T. (2017). Intelligent manufacturing in the context of industry 4.0: A review. *Engineering, 3*(5), 616-630.

[2] Hassija, V. (2025). A survey on digital twins: Enabling technologies, use cases, application, open issues, and more. *IEEE Journal of Selected Areas in Sensors, 2*, 84-107.

[3] Yao, J. F., Yang, Y., Wang, X. C., Tan, J., & Zhang, Q. (2023). Systematic review of digital twin technology and applications. *Visual Computing for Industry, Biomedicine, and Art, 6*(1), Article 10. https: //doi.org/10.1186/s42492-023-00138-3

[4] Tao, F., Zhang, H., Liu, A., & Nee, A. Y. C. (2019). Digital twin in industry: State-of-the-art. *IEEE Transactions on Industrial Informatics, 15*(4), 2405–2415. https: //doi.org/10.1109/TII.2018.2873186

[5] Fuller, A., Fan, Z., Day, C., & Barlow, C. (2020). Digital twin: Enabling technologies, challenges and open research. *IEEE Access, 8*, 108952–108971. https: //doi.org/10.1109/ACCESS.2020.2998358

[6] Grieves, M. (2022). Intelligent digital twins and the development and management of complex systems. *Digital Twin, 1*(1), Article 2. https: //doi.org/10.12688/digitaltwin.17441.1

[7] Liu, M., Fang, S., Dong, H., & Xu, C. (2021). Review of digital twin about concepts, technologies, and industrial applications. *Journal of Manufacturing Systems, 58*, 346–361. https: //doi.org/10.1016/j.jmsy.2020.06.017

[8] Jones, D., Snider, C., Nassehi, A., Yon, J., & Hicks, B. (2020). Characterising the digital twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology, 29*, 36–52. https: //doi.org/10.1016/j.cirpj.2020.02.002

[9] Negri, E., Fumagalli, L., & Macchi, M. (2017). A review of the roles of digital twin in CPS-based production systems. *Procedia Manufacturing, 11*, 939–948. https: //doi.org/10.1016/j.promfg.2017.07.198

[10] Kritzinger, W., Karner, M., Traar, G., Henjes, J., & Sihn, W. (2018). Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine, 51*(11), 1016–1022. https: //doi.org/10.1016/j.ifacol.2018.08.474

[11] Curcio, E. M., Sgambitterra, E., Lamuta, C., Maletta, C., & Niccoli, F. (2025). Finite state machine-based modeling of shape memory alloy actuators: A multi-physics approach for predicting thermo-mechanical behavior and hysteresis. *Sensors and Actuators A: Physical, 394*, 116990. https: //doi.org/10.1016/j.sna.2024.116990

[12] Zhang, Z., Wang, A., Li, H., & Xia, C. (2025). Algebraic calculation for safety synthesis of networked logical finite state machines. *European Journal of Control, 86*, 101386. https: //doi.org/10.1016/j.ejcon.2024.101386

[13] Son, Y. H., Park, K. T., Lee, D., Jeon, S. W., Noh, S. D., Do Noh, S., & others. (2021). Digital twin–based cyber-physical system for automotive body production lines. *The International Journal of Advanced Manufacturing Technology, 115*(9), 291–310. https: //doi.org/10.1007/s00170-021-07131-1

[14] Zhuang, C., Liu, J., & Xiong, H. (2018). Digital twin-based smart production management and control framework for the complex product assembly shop-floor. *The International Journal of Advanced Manufacturing Technology, 96*(1), 1149–1163. https: //doi.org/10.1007/s00170-018-1617-6

[15] Lu, Y., Liu, C., Wang, K. I. K., Huang, H., & Xu, X. (2020). Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues. *Robotics and Computer-Integrated Manufacturing, 61*, 101837. https: //doi.org/10.1016/j.rcim.2019.101837

[16] Shangguan, D., Chen, L., & Ding, J. (2020). A digital twin-based approach for the fault diagnosis and health monitoring of a complex satellite system. *Symmetry, 12*(8), Article 1307. https: //doi.org/10.3390/sym12081307

[17] Redelinghuys, A. J. H., Basson, A. H., & Kruger, K. (2020). A six-layer architecture for the digital twin: A manufacturing case study implementation. *Journal of Intelligent Manufacturing, 31*(6), 1383–1402. https: //doi.org/10.1007/s10845-019-01516-6

[18] Kampa, A. (2023). Modeling and simulation of a digital twin of a production system for industry 4.0 with work-in-process synchronization. *Applied Sciences, 13*(22), Article 12261. https: //doi.org/10.3390/app132212261

[19] Shangguan, D., Chen, L., & Ding, J. (2019, October). A hierarchical digital twin model framework for dynamic cyber-physical system design. In *Proceedings of the 2019 ACM International Conference on Systems Engineering* (pp. 123–129). ACM. https: //doi.org/10.1145/3358961.3358978

[20] Alhazmi, T., Azzedin, F., & Hammoudeh, M. (2025). MQTT based data distribution framework for digital twin networks. In *Proceedings of the 2025 International Conference on Computing and Network Communications* (pp. 1008–1013). Association for Computing Machinery. https: //doi.org/10.1145/3640101.3640205

[21] Uhlemann, T. H. J., Lehmann, C., & Steinhilper, R. (2017). The digital twin: Realizing the cyber-physical production system for industry 4.0. *Procedia CIRP, 61*, 335–340. https: //doi.org/10.1016/j.procir.2016.11.152

[22] Waschneck, B., Altenmüller, T., Bauernhansl, T., & Kyek, A. (2018). Optimization of global production scheduling with deep reinforcement learning. *Procedia CIRP, 72*, 1264–1269. https: //doi.org/10.1016/j.procir.2018.03.212

[23] Cimino, C., Negri, E., & Fumagalli, L. (2019). Review of digital twin applications in manufacturing. *Computers in Industry, 113*, Article 103130. https: //doi.org/10.1016/j.compind.2019.103130

[24] Harel, D. (1987). Statecharts: A visual formalism for complex systems. *Science of Computer Programming, 8*(3), 231–274. https: //doi.org/10.1016/0167-6423(87)90035-9

[25] Thramboulidis, K. (2015). A cyber-physical system-based approach for industrial automation systems. *Computers in Industry, 72*, 92–102. https: //doi.org/10.1016/j.compind.2015.04.006