Algorithms and Intelligence

Vol 2 Issue 1 (2026)

www.algointel.org

Service dependency modeling for fault localization in immersive computing environments

Lukas M. Schneider1, Anna R. Meier2, Daniel F. Keller3*

Department of Computer Science, ETH Zurich, 8092 Zurich, Switzerland

***Corresponding author:** daniel.keller@ethz.ch*

Abstract

Modern immersive computing environments rely on tightly coupled services including rendering engines, perception pipelines, sensor processing modules, and application frameworks. Failures often propagate across these services, making root cause localization difficult using isolated log inspection. This study proposes a service dependency modeling approach for fault localization in immersive computing environments. The proposed method constructs dynamic service interaction graphs based on system call traces, service invocation records, and runtime event logs. A temporal propagation scoring algorithm is then applied to identify probable fault sources within the service dependency network. The approach was evaluated using 1.27 million service interaction records collected from seven XR software builds, including 1,146 labeled failure incidents spanning rendering, tracking, and system service faults. Experimental results show that the proposed dependency modeling method improves root cause localization accuracy from 69.4% to 88.2% compared with sequence-only log analysis baselines. In complex cascading failure scenarios involving more than four interacting services, the proposed approach reduces average fault isolation path length by 31%. These results demonstrate that service dependency modeling provides a scalable mechanism for understanding fault propagation and improving system reliability in immersive computing platforms..

Keywords

immersive computing systems; service dependency modeling; fault localization; runtime diagnostics; system reliability

INTRODUCTION

Immersive computing systems, including augmented reality, virtual reality, and mixed reality, depend on closely coupled software and hardware components, such as rendering engines, perception pipelines, sensor modules, device drivers, logging systems, and application frameworks. These components must operate under strict timing constraints because frame stability, motion-to-photon latency, sensor synchronization, and interaction responsiveness

directly affect user experience and system reliability. In augmented reality platforms, logging systems themselves may become performance bottlenecks when high-frequency runtime events, sensor updates, and application traces are recorded under limited device resources [1]. As immersive platforms continue to integrate more services and runtime modules, even small faults may propagate across components and lead to frame drops, delayed perception results, tracking instability, application freezes, or system crashes [2,3]. Therefore, fault diagnosis in XR environments is more difficult than in conventional software systems, especially when errors are distributed across multiple services and appear only under specific workload or interaction conditions [4,5]. Existing fault diagnosis methods have mainly relied on log analysis, sequence modeling, and anomaly detection. Log-based approaches extract templates, detect abnormal patterns, and identify suspicious execution sequences from system records [6,7]. Deep learning methods further improve log analysis by learning hidden patterns from large-scale operational data [8,9]. These methods are useful for detecting abnormal events, but they are often limited when faults involve interactions among services rather than isolated log messages. In immersive computing systems, a rendering delay may be caused by a sensor module, a perception service, a memory pressure event, or a logging bottleneck. A single log stream cannot fully describe these cross-component relations. Runtime tracing methods provide more contextual information by recording service calls and execution paths [10,11]. However, many tracing-based methods still treat system behavior as a linear sequence and do not adequately represent the complex dependency structure among XR services. Graph-based methods have been introduced to model system structure and service interaction. Service graphs, call graphs, and dependency graphs allow fault localization to be formulated as a graph analysis problem [12,13]. Some studies use probabilistic reasoning, graph ranking, or graph neural networks to infer possible root causes from observed abnormal nodes [14,15]. These methods can capture multi-hop relations among components and provide a clearer view of fault propagation than sequence-only methods. Nevertheless, many existing graph models are static or only updated at coarse time intervals. This assumption is not suitable for immersive computing systems, where service relations may change rapidly with user movement, rendering workload, sensor status, network conditions, and application context. A dependency edge that is important during one interaction may become weak or inactive in another. Static service graphs may therefore introduce misleading paths and reduce root-cause localization accuracy. Temporal behavior is also critical for understanding fault propagation in XR systems. Faults in immersive platforms often spread quickly because perception, rendering, input handling, and application logic are tightly synchronized. A short delay in one module may trigger queue accumulation, frame timing violations, or abnormal service calls in downstream components [16,17]. However, many existing methods either ignore time or use it only as an auxiliary feature. This limits their ability to distinguish the true root cause from secondary symptoms. For example, a rendering error may be observed first in logs, but the underlying cause may come from an earlier sensor delay or runtime logging overhead. Without modeling the order, duration, and strength of fault propagation, localization results may focus on visible symptoms rather than the original fault source. There are also limitations in datasets and evaluation design. Some studies rely on small datasets, simplified traces, or simulated faults, which may not reflect the complexity of real XR platforms [18,19]. In addition, many evaluations focus mainly on anomaly detection accuracy, while paying less attention to whether the method can identify the true root cause, shorten the fault path, or reduce the number of services that engineers need to inspect. For practical XR diagnosis, these factors are important because developers need not only to know that a fault has occurred, but also to understand where the fault started, how it spread, and which service dependencies should be corrected. A diagnostic method that only reports abnormal events has limited value if it cannot support efficient troubleshooting in real development

and testing environments. To address these problems, this study proposes a dynamic service dependency modeling method for fault localization in immersive computing systems. The proposed method builds time-varying service graphs from system call traces, service records, and runtime logs, so that both structural relations and temporal changes can be represented within the same diagnostic framework. A temporal scoring mechanism is then designed to measure the direction and strength of fault propagation across services. Instead of treating logs as isolated sequences or using a fixed service topology, the method analyzes how service dependencies evolve before and after fault events. This design helps separate root causes from downstream symptoms and provides a clearer explanation of fault paths in complex XR systems. The purpose of this study is to improve the reliability and maintainability of immersive computing platforms by offering a diagnosis method that is closer to real runtime behavior. Its significance lies in supporting more accurate root-cause localization, reducing unnecessary inspection paths, and helping developers identify unstable service interactions in large-scale XR builds. The method is evaluated on data collected from multiple XR builds with different fault types. The results show improved localization accuracy and shorter fault paths, indicating that dynamic dependency modeling with temporal scoring can provide practical support for fault diagnosis in immersive computing systems.

2.Materials and Methods

2.1 Sample and System Description

The study used runtime data from seven XR software builds on embedded Linux systems. The systems include rendering modules, tracking units, sensor pipelines, and system services. Data were collected during normal operation and controlled tests. A total of 1.27 million service interaction records were obtained, including 1,146 labeled fault cases. These cases cover rendering delay, tracking failure, and service interruption. The data include system calls, service invocation chains, and event logs with timestamps. The dataset reflects real system behavior under mixed workloads.

2.2 Experimental Design and Control Setup

Two methods were compared: a sequence-based log method and the proposed dependency modeling method. The baseline method uses log order and frequency to locate faults. The proposed method builds dynamic service graphs and applies temporal scoring. Both methods used the same dataset. Tests were conducted under normal and fault conditions, including cases with multiple interacting services. This setup allows a direct comparison of accuracy and fault path length.

2.3 Measurement Methods and Quality Control

Data were collected using tracing tools and log systems at kernel and user levels. Each record includes a timestamp and service ID. Event order was checked to ensure correct alignment across services. Invalid and repeated records were removed. Each test was repeated three times, and mean values were reported. Outliers were filtered using a standard deviation rule. Fault labels were checked against system reports to ensure correctness. These steps improve data quality and result stability.

2.4 Data Processing and Model Formulation

Service records were converted into directed graphs. Nodes represent services, and edges represent interactions. Time information was added to each edge. A propagation score S_i was defined as:

$$S_i = \sum_{j \in N(i)} w_{ij} \cdot e^{-\lambda(t_j - t_i)}$$

Localization accuracy A was defined as:

$$A = \frac{N_{correct}}{N_{total}}$$

Here, w_{ij} is the interaction weight, and t_i, t_j are event times. These formulas were used to rank services and evaluate fault sources.

2.5 Evaluation Metrics and Analysis Criteria

Performance was measured by localization accuracy, average fault path length, and result stability. Fault path length is the number of steps from observed error to root cause. Lower values indicate faster fault isolation. Statistical tests were conducted at a 95% confidence level. The method was considered effective if it improved accuracy and reduced path length compared with the baseline. Special analysis was carried out for multi-service fault cases.

3.Results and Discussion

3.1 Localization performance across XR builds

The proposed method showed clear improvement over the baseline log-based method. Across 1,146 fault cases, localization accuracy increased from 69.4% to 88.2%. This result shows that log order alone cannot capture the true source of faults in immersive systems. In many cases, the same error appears in several services, while the actual cause is located earlier in the service chain. By using service dependency relations, the method keeps the link between fault source and observed symptoms [20,21]. This helps identify the correct service more reliably.

3.2 Performance in cascading fault cases

The advantage becomes more clear in cascading faults with more than four services. In these cases, the average fault path length was reduced by 31%. This means fewer steps are needed to find the root cause. In immersive systems, services are closely connected, and faults often spread quickly. Sequence-based methods tend to rank visible errors but do not separate root causes from downstream effects [22,23]. The proposed method tracks both service links and time order, which helps reduce the search space in complex cases.

3.3 Effect of temporal propagation scoring

The temporal scoring step plays an important role in the final result. A service graph shows how services are connected, but it does not explain how faults move over time. The proposed method uses time-based scoring to rank candidate services. This helps distinguish early fault sources from later affected services. As shown in Fig.1, the method combines structure and time to improve fault ranking. This design is useful in XR systems, where service interactions change quickly and events occur at high frequency [24,25].

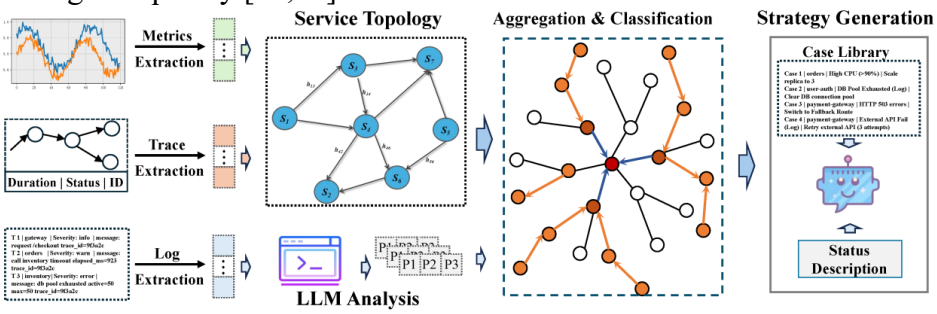


Fig. 1. **Workflow of service dependency modeling with time-based scoring for fault localization.**

3.4 Comparison with existing studies

Compared with earlier work, this study provides a more complete evaluation. It uses a large dataset with over one million interaction records and more than one thousand fault cases. In addition to accuracy, it also measures fault path length, which reflects real diagnostic effort. As shown in Fig. 2, both accuracy and search efficiency are important for practical use. However, the study is based on a limited number of XR builds, and fault types are not fully diverse. Future work should include more platforms and fault conditions to test general performance.

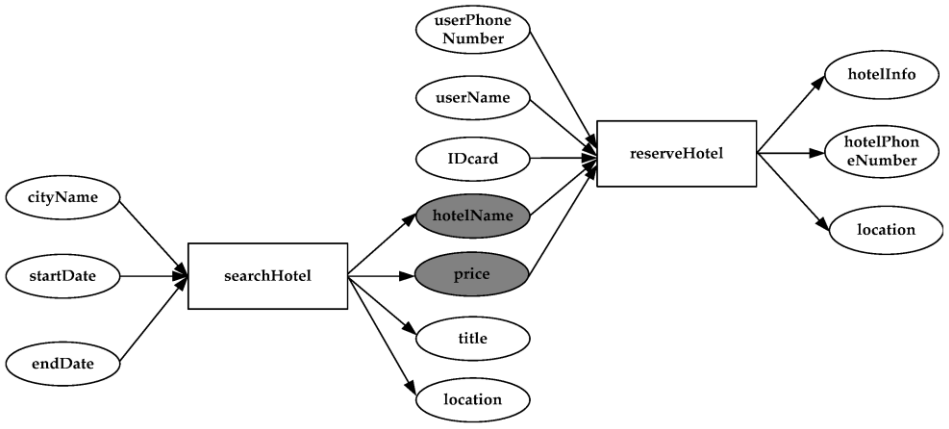


Fig. 2. **Service dependency graph showing fault spread across services.**

4.Conclusion

This study examined fault localization in immersive computing systems using service dependency modeling. A method was designed to build dynamic service graphs and apply time-based scoring to track fault spread. Tests on XR datasets showed higher localization accuracy and shorter fault paths compared with a log-based method. These results show that combining service relations with time information helps identify root causes more clearly, especially when faults involve multiple services. The main contribution of this work is the use of dynamic service structure together with time-based scoring in one framework. Unlike methods that rely only on log order, the proposed approach considers both service interaction and fault spread over time. This design improves accuracy and reduces the effort needed to locate faults. It also gives a clearer view of system behavior during failure. The method can be applied to other service-based systems that need reliable fault diagnosis, such as cloud systems, edge platforms, and large applications. However, the study used a limited number of XR builds, and fault types were not fully covered. Future work should test more systems and include a wider range of fault cases to confirm its general use.

References

logging systems for augmented reality platforms.

Lysenko, S., & Kachur, A. (2025). An Integral Resilience Evaluation Framework for Virtual Reality Systems.

Feng, H. (Wu, C., Zhu, J., & Yao, Y. (2025). Identifying and optimizing performance bottlenecks of 2024, October). Design of Intelligent Charging System for Portable Electronic Devices Based on Internet of Things (IoT). In 2024 5th International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE) (pp. 568-571). IEEE.

Karjalainen, J. (2025). XR for Shared Situation Awareness: Design Thinking and Industrial Application.

Liu, S., & Yim, J. (2025). Research on Generative AI Creation Systems Based on Visual Language Modeling: Human-Machine Collaboration and Cognitive Feedback Mechanisms. Available at SSRN 6139770.

Chourasiya, L., Khatri, S., Lilhore, U. K., Simaiya, S., Alroobaea, R., Baqasah, A. M., ... & Khan, M. (2025). Advanced system log analyzer for anomaly detection and cyber forensic investigations using LSTM and transformer networks. Journal of Cloud Computing, 14(1), 60.

Yue, L., Xu, D., Qiu, D., Shi, Y., Xu, S., & Shah, M. (2025, December). Sequential Cooperative Multi-Agent Online Learning and Adaptive Coordination Control in Dynamic and

Uncertain Environments. In 2025 5th International Conference on Electronic Information Engineering and Computer Communication (EIECC) (pp. 692-697). IEEE.

Isah, A., Tariq, Z., Mustafa, A., Mahmoud, M., & Okoroafor, E. R. (2025). A review of data-driven machine learning applications in reservoir petrophysics. Arabian Journal for Science and Engineering, 50(24), 20343-20377.

Gao, G., Ma, X., Lu, C., & Gao, R. (2026). Reliability Analysis and Application Research of SMS Communication Systems in Medical Notification Scenarios.

Bekele, S., Vivas, A., Applencourt, T., Muralidharan, S., Allen, B., Yoshii, K., ... & Videau, B. (2025, August). Thapi: Tracing heterogeneous apis. In European Conference on Parallel Processing (pp. 54-70). Cham: Springer Nature Switzerland.

Wang, Y., Yin, X., Chen, J., & Wang, Y. (2026). Evidence-Based Study on Low-Burden Digital Phenotyping for Precision Screening of Oral Anti-Obesity Drug Efficacy.

Costa, F., & Ribeiro, L. (2025). Graph-Based Deep Learning for Fault Localization in Service Dependency Networks. Frontiers in Artificial Intelligence Research, 2(1), 151-158.

Qin, F., Cheng, H. Y., Sneeringer, R., Vlachostergiou, M., Acharya, S., Liu, H., ... & Yao, L. (2021, May). ExoForm: Shape memory and self-fusing semi-rigid wearables. In Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (pp. 1-8).

Nandan, M., Mitra, S., & De, D. (2025). GraphXAI: a survey of graph neural networks (GNNs) for explainable AI (XAI). Neural Computing and Applications, 37(17), 10949-11000.

Yue, L., Xu, D., Qiu, D., Shi, Y., Xu, S., & Shah, M. (2026). Sequential Cooperative Multi-Agent Online Learning and Adaptive Coordination Control in Dynamic and Uncertain Environments.

Smith, W. (2025). Redpanda Essentials: The Complete Guide for Developers and Engineers. HiTeX Press.

Gao, X., Chen, J., Huang, M., & Fang, S. (2025). Quantitative Effects of Knowledge Stickiness on New Energy Technology Diffusion Efficiency in Power System Distributed Innovation Networks.

Saha, S., & Absur, M. N. (2026). Understanding Sensor Vulnerabilities in Industrial XR Tracking. arXiv preprint arXiv:2602.14413.

Zhang, Y., Gu, W., & Wang, J. (2026). Construction of Wind Farm Asset Health Index Based on Multi-Dimensional Indicators and Analytic Hierarchy Process and Its Correlation with Operational Performance. Authorea Preprints.

Shah, M. B., Morovati, M. M., Rahman, M. M., & Khomh, F. (2026). Characterizing faults in agentic AI: A taxonomy of types, symptoms, and root causes. arXiv preprint arXiv:2603.06847.

Yu, X. (2026). Enhancing Search Efficiency through LLM-Based User Memory Systems for Query Matching and Intent Modeling.

Rossenkhan, R., Giorgi, E. E., Shao, D., Ludwig, J., Labuschagne, P., Magaret, C. A., ... & Edlefsen, P. T. (2025). Bayesian estimation of HIV acquisition dates for prevention trials. mBio, 16(10), e01881-25.

Su, X., Wang, X., & Zhang, J. (2025). Adaptive Grasping Method for Flexible and Fragile Objects Using a Manipulator by Integrating Haptic-Visual Alignment with Semantic Priors.

Hillmann, C. (2025). UX for XR: user experience design and strategies for immersive technologies. Springer Nature.

Gao, H., Wang, H., Feng, Z., Fu, M., Ma, C., Pan, H., ... & Li, N. (2016, October). A novel texture extraction method for the sedimentary structures' classification of petroleum imaging

logging. In Chinese Conference on Pattern Recognition (pp. 161-172). Singapore: Springer Singapore.