Research Statement

HUO Yintong
School of Computing and Information Systems, Singapore Management University
Tel: (65) 6828-0416; Email: ythuo@smu.edu.sg
19 (Day) 12 (Month) 2025 (Year)

Background

Modern software systems are undergoing a fundamental paradigm shift from static tools to **interactive software systems**, powered by AI agents, adaptive interfaces, and autonomous operations. While these advancements promise efficiency, they introduce new dimensions of **reliability risks**. A hallucinating coding assistant introduces subtle bugs; a misaligned UI generator creates functional dead-ends; and opaque logs in complex systems hide critical failures from operators.

My research lies at the intersection of *software reliability* and *intelligent interaction*. I argue that to build reliable systems in the AI era, we must secure the *interactions* between humans, agents, and system internals. My long-term goal is to pioneer reliability assurance for interactive systems, developing techniques that ensure AI agents generate correct software artifacts, observability tools provide accurate diagnoses, and automated front-end engineering yields functionally robust interfaces. I aim to transform "interaction" from a source of ambiguity into a mechanism for resilience.

Research Areas

1) Reliability in context-aware code intelligence

The integration of Large Language Models (LLMs) into software development is inevitable, but their lack of reliability prevents widespread adoption in critical engineering tasks. LLMs often hallucinate APIs, violate syntax rules, or generate logically flawed code. My research addresses the correctness and robustness of AI-generated code. I focus on grounding probabilistic models in deterministic software engineering realities.

- Static Context Integration: To reduce generation errors, we developed techniques that enforce "static constraints" (such as call graphs and type information) during the generation process. This ensures that the AI's output respects the structural rules of the codebase [1, 3].
- Automated Repair & Recovery: Recognizing that generation will never be perfect, we focus on resilience that automatically repairing buggy code. By clustering failure patterns and utilizing template-based fixes, we guide models to recover from errors reliably [2].
- Knowledge-Grounded Accuracy: We improved the accuracy of API usage by developing tools that automatically resolve and link unformatted text to official documentation, reducing "hallucinated" libraries [4].

Future directions: Trustworthy Human-Agent Collaboration**.**

As we transition from code generation tools to autonomous software agents, the primary challenge shifts from simple syntax correctness to architectural reliability. My future work aims to establish a framework for *Trustworthy Human-Agent Collaboration*, where AI systems can reason about the broader implications of their code changes. I am interested in exploring how agents can move beyond completion to verification, understanding the safety properties of a system and proactively communicating potential risks to human developers. The ultimate goal is to bridge the cognitive gap between human intent and machine execution, ensuring that autonomous agents act as reliable partners in the engineering lifecycle.

2) Observability and reliability engineering (AIOps)

Reliability depends on the speed and accuracy with which we can detect and diagnose failures. In large-scale interactive systems, logs are the ground truth of system health. However, the volume and complexity of logs often obscure the root cause of failures. My research enhances system resilience by transforming chaotic data into actionable reliability insights. My work focuses on the semantics and adaptability of log analysis to prevent "observability failure", where operators miss critical signals due to parsing errors.

- Semantic Consistency: Existing parsers often break when log formats change. We developed SemParser [9] to extract semantic meaning rather than just syntax, ensuring that reliability monitoring continues uninterrupted even as software evolves.
- Adaptive Anomaly Detection: Reliability tools must evolve with the system. We proposed EvLog [10], which use feedback loops to adapt anomaly detection models to software updates, preventing false alarms and missed failures.
- Efficient Diagnosis with LLMs: We leveraged the reasoning capabilities of LLMs to parse and interpret logs [5, 8, 9], demonstrating that post-learning strategy can be a powerful tool for diagnosing failures in systems where training data is scarce.

Future Directions: From detection to causal understanding and prevention
Current AIOps solutions largely focus on the detection of anomalies: telling engineers that something is wrong. My future research aims to shift this paradigm toward deep understanding and proactive prevention. I plan to investigate methods that can derive causal relationships from multimodal operational data, enabling systems to not only pinpoint the root cause of a failure but also explain why it happened in natural language. Furthermore, I envision a closed-loop reliability framework where operational insights are used to predict future risks, allowing systems to "learn" from past incidents and suggest preventative architectural improvements before failures reoccur.

3) Functional Reliability in Multimodal UI Engineering

The User Interface (UI) is the functional contract between the user and the system. If the UI is broken, the system is unreliable, regardless of backend stability. As we automate UI engineering using Multimodal LLMs (MLLMs), ensuring the functional integrity and correctness of generated interfaces is a major reliability challenge. My research treats UI generation as a rigorous engineering problem, not just a design task.

- Structural Correctness: We identified that "single-pass" generation leads to structural hallucinations. Our Divide-and-Conquer (DCGen) approach [12] improves reliability by decomposing screens into verifiable sub-components, ensuring the final code faithfully implements the visual intent.
- Interactive Integrity: A static UI is useless if the buttons don't work. Our benchmark Interaction2Code [11] revealed significant reliability gaps in how current models implement dynamic logic (e.g., event handlers, state changes). We are the first to systematically quantify these "interaction failures."

Future Directions: Aligning Human-Machine Comprehension for Web Agents
A fundamental barrier to reliable UI engineering is the semantic gap between how humans perceive an interface (visually and intuitively) and how machines process it (structurally and logically). My future work focuses on aligning human-machine comprehension, creating models that can see a UI not just as pixels or DOM trees, but as a set of functional affordances and user intents. This alignment is the critical stepping stone for the next generation of autonomous web agents. By enabling machines to deeply understand the purpose and behavior of UI elements, we can move toward agents that not only generate interfaces but can also navigate, test, and interact with the web with the same reliability and intuition as a human user.

Selected Publications and Outputs

A comprehensive list of publications is available on google scholar.

Reliability in Code Intelligence & Agents
1. [TSE 2024] Yichen Li*, **Yintong Huo***, Zhihan Jiang, Renyi Zhong, Pinjia He, Yuxin Su, Lionel C. Briand, and Michael R. Lyu. "*Exploring the Effectiveness of LLMs in Automated Logging Statement Generation: An Empirical Study*." IEEE Transactions on Software Engineering.
2. [ICSE 2024] Yun Peng, Shuzheng Gao, Cuiyun Gao, **Yintong Huo**, and Michael R. Lyu. "*Domain Knowledge Matters: Improving Prompts with Fix Templates for Repairing Python Type Errors."*
3. [FSE 2024] Yichen Li, **Yintong Huo**, Renyi Zhong, Zhihan Jiang, Jinyang Liu, Junjie Huang, Jiazhen Gu, Pinjie He, and Michael R. Lyu. "*Go Static: Contextualized Logging Statement Generation."*
4. [ICSE 2022] **Yintong Huo**, Yuxin Su, Hongming Zhang, and Michael R. Lyu. "*ARCLIN: Automated API Mention Resolution for Unformatted Texts."*

Observability and reliability engineering
5. [ICSE'26] Shiwen Shan, **Yintong Huo**, Yuxin Su, Zhining Wang, Dan Li, and Zibin Zheng. *"ConfLogger: Enhance Systems' Configuration Diagnosability through Configuration Logging"*
6. [FSE 2024] Zhihan Jiang, Jinyang Liu, Zhuangbin Chen, Yichen Li, Junjie Huang, **Yintong Huo**, Pinjia He, Jiazhen Gu, and Michael R Lyu. *"Lilac: Log parsing using LLMs with adaptive parsing cache."*
7. [ICSE 2024] Junjielong Xu, Ruichun Yang, **Yintong Huo**, Chengyu Zhang, and Pinjia He. *"DivLog: Log Parsing with Prompt Enhanced In-Context Learning."*

8. [ASE 2023] **Yintong Huo***, Yichen Li*, Yuxin Su, Pinjia He, Zifan Xie, and Michael R. Lyu. *"AutoLog: A Log Sequence Synthesis Framework for Anomaly Detection."*
9. [ICSE 2023] **Yintong Huo**, Yuxin Su, Baitong Li, and Michael R. Lyu. *"SemParser: A Semantic Parser for Log Analytics."*
10. [ISSRE 2023] **Yintong Huo**, Cheryl Lee, Yuxin Su, Shiwen Shan, Jinyang Liu and Michael R. Lyu. *"EvLog: Identifying Anomalous Logs over Software Evolution."*

Functional Reliability in UI Engineering

11. [ASE 2025] Jingyu Xiao, Yuxuan Wan, **Yintong Huo**, Zhiyao Xu, Michael R Lyu. *"Interaction2Code: How Far Are We from Automatic Interactive Webpage Generation?".*
12. [FSE 2025] Yuxuan Wan, Chaozheng Wang, Yi Dong, Wenxuan Wang, Shuqing Li, **Yintong Huo**, and Michael R Lyu*. "Divide-and-Conquer: Generating UI Code from Screenshots"*.
13. [AIWare'25] Truong Hai Dang, Jingyu Xiao, and **Yintong Huo**. *“Envisioning Future Interactive Web Development: Editing Webpage with Natural Language”.*