Flynn(Yinmingren) Fu

Mobile: +1 (669)210-2896 | Email: ymrenfu@gmail.com | LinkedIn | Github | Website

Education

Santa Clara University — Sep. 2024 – Expect Jun. 2026
Master of Science, Computer Science and Engineering — *San Jose, CA, United States*

South China University of Technology — Sep. 2020 – Jun. 2024
Bachelor of Science, Information and Computing Science — *Guangzhou, Guangdong, China*

Technical Skills

Programming Language: C/C++, C#, Java, Python, Golang, JavaScript, HTML, SQL, Shell/Bash, CUDA
Platforms & Framework: Spring Boot, Zookeeper, Kafka, gRPC, Muduo, Nginx, Netty, .Net Framework, WPF
Database: MySQL, SQLServer, Redis, PostgreSQL, MongoDB
Tools: Git, Docker, Kubernetes, AWS, gdb, cmake, TortoiseGit, Mercurial, Github Actions

Work Experience

C++ Development Engineer — Jun. 2023 – Oct. 2023
ZWSOFT — *Guangzhou, Guangdong, China*

- Designed and implemented a **thread-safe memory pool** (in C++17/20) for 3D engineering software by using **Boost** and **Singleton pattern**, leveraged **locks** and **atomic** variables for safe multi-threaded memory management.
- Implemented memory pool **pre-allocation** and **recycling** operations, enabling efficient reuse of reserved memory across multiple processes. Designed dedicated memory pool for different variable types, ensured **memory alignment**, reduced memory usage by **15%**, and improved 3D geometry import and rendering performance by **50%**.
- Enhanced 3D engineering software export module to support structured and unstructured mesh export using **VTK4.0 format**. Upgraded the export algorithm by integrating with **memory pool operations**, accelerated the algorithm and minimized redundant memory allocation overhead.
- Replaced **new/delete/malloc/free** with **allocate/deallocate** operations in memory pool, improved allocation efficiency for 3D object vertices, lines, and polygons, enhanced overall system performance and stability.

Project Experience

Distributed KV database based on Raft consensus algorithm — Dec 2024 – Present
Framework: C++, Boost, STL, Muduo, protobuf

- Implemented log replication and leader election for the Raft consensus algorithm, leveraging **threadpool** to manage heartbeat and election tasks, ensuring the maintenance of the cluster's log commit state.
- Developed an **RPC** communication framework using **Protobuf** to facilitate efficient remote procedure calls and data transmission between Raft nodes.
- Built a **skiplist-based Key-Value database** to provide data storage and retrieval.

MathMind: AI-Powered Math Scan & Solution APP — Dec. 2023 – May. 2024
Framework: Java, JavaScript, Python, Kotlin, SQL, Shell/Bash, Docker, Pytorch

- Designed and implemented a **multi-threaded** service orchestrator leveraging **Flask**, managing two **decoupled microservices** for image scan and math solution. Established server-client communication through **WebSockets** with a standardized custom **JSON** message format.
- Architected a scalable math solution microservice integrating locally deployed **TensorRT-optimized models** (**ToRA-7b/Llama3**) fine-tuned on algebra/calculus datasets with remote LLM APIs (**GPT, Gemini**).
- Architected an image scan microservice utilizing **Base64** for image decoding from client side, **OpenCV** and **Pickle** for image preprocessing, and Transformer-based OCR models for accurate LaTeX code and text extraction.
- Containerized microservices using **Docker** with **Miniconda** for portability and environment isolation, and developed **Shell/Bash scripts** to automate deployment, compilation, and service management. Designed and exposed **RESTful APIs** to enable Android client access to both image scan and math solution microservices.
- Developed an Android app, leveraging **Kotlin Coroutines** for asynchronous processing, **Handlers** for UI thread management, and push notifications for real-time updates, following **MVCC architecture**, enabling clients to remote call for image scan and math solution modules.

A High-Concurrency C++ Server Library Based on Muduo Library — Apr. 2023 – Jul. 2024
Framework: C++, Muduo, Boost, STL

- Implemented a high concurrency server using non-blocking, I/O multiplexing, and the **Reactor model**, referencing the **Muduo** and **Boost**.
- Developed with a multi-threading pattern, implementing classes like **EventLoop**, **Poller**, and **Channel** to enable loop listening, request dispatching, and event handling.