HUANGYU ZHANG

(213) 994-7810 | huangyuz@usc.edu | DavidZhang0710.github.io | github.com/DavidZhang0710

EDUCATION

University of Southern California **Aug 2024 – May 2026**
Master of Science in Computer Science *Los Angeles, USA*

Xi'an Jiaotong University **Sep 2020 – Jun 2024**
Bachelor of Engineering in Computer Science, GPA: 90/100, Top 5% *Xi'an, China*

TECHNICAL SKILLS

Programming Languages: *Java, C++, Python*
Backend Frameworks: *Spring Boot, Flask, Node.js*
Databases: *MySQL, Redis, Mybatis*
Web Development: *HTML, CSS, Vue.js*
Data Analysis: *scikit-learn, pandas, NumPy, Matplotlib*
Tools and Others: *Git, Visual Studio Code, IntelliJ IDEA, Postman*

EXPERIENCE

Bytedance, Inc **Jun 2023 – Oct 2023**
C++ Software Development Internship *Shanghai, China*

- Designed a non-intrusive parameter serialization component based on C++ **macros** and **Templates**, and added pre-validation to ensure parameter integrity. Implemented auto-expanding serializing with only a **7KB** increase in package size and decreased API crash rates by **0.2%** through robust parameter validation
- Built a tool to automatically generate API documentation from annotated header files. Utilized **Doxygen** and **RapidXML** parser to parse definitions and comments from header files. Converted parsed information into a custom JSON format to support C++ overloading and multi-platform compatibility
- Developed a Message Bus module only based on C++ STL and **multi-threading** libraries, implemented synchronous and asynchronous sending methods, and involved queues to optimize and simplify the asynchronous sending method, reducing response time of multi-stream APIs by **6.9%**

PROJECTS

EasyRPC | *Java, Maven, Zookeeper, Proxy, Netty*

- Designed and built a Java-based RPC (Remote Procedure Call) component, and introduced **timeout retry** and request **filters** to enhance reliability and high availability
- Utilized Zookeeper as a service registry, running listener threads to dynamically update service changes and handle service registration in real-time
- Implemented client module to **asynchronously** send RPC requests and listen for responses, and server module to handle requests **asynchronously** using IO threads, with main thread processing and returning results upon completion
- Conducted stress tests with consecutive requests ranging from **100** to **10,000**, and observed no degradation in response time or accuracy

Traffic Flow Detection System based on Computer Vision | *SpringBoot, Mybatis, MySQL, Redis, Flask, Vue*

- Developed a web-based system for detecting traffic congestion in images, employing a front-end back-end separated development model
- Implemented features such as user authentication, user management, batch analysis and analysis history viewing
- Back-end was based on Spring Boot, with Redis caching, resolved potential cache penetration issues
- Front-end was developed based on a Vue starter, visualized results through lists and charts
- Trained a visual model based on **YOLOv5** on the UA-DETRAC dataset, achieving a $mAP_{0.5:0.95}$ of **0.948**, and deployed it via Flask

Socket Chat Application | *C++, Socket, Voice call*

- Designed and developed a server and client application for a **multi-user** chat room. with support for up to **100** concurrent users, file transfer and voice call
- Recorded file sizes and SHA-256 value to enable **Breakpoint Continuation** and **Offline transfer**
- Utilized Windows Multimedia API with **multiple buffers** to implement smooth voice call, achieving a latency of **0.1s** excluding network delays