Austin Yang

(706)-393-5318 | awtyang21@gmail.com | wentao.blog

EDUCATION

Columbia University – School of Engineering & Applied Science New York, NY

B.S. Computer Science *Anticipated May 2028*

- Relevant Coursework: Data Structures and Algorithms (Java), Advanced Programming (Linux, C, UNIX), Introduction to Databases (SQL), Competitive Programming (C), C++ Deep Dive for C Programmers (C++) , Advanced Systems Programming (C), Operating Systems (C), Design Using C++, Programming Languages and Translators (C)

EXPERIENCE

Omen Trade New York, NY

Software Developer Intern *October 2025 - Present*

- Built Grafana dashboards to monitor Kubernetes cluster health, AI agent performance, and workloads with **~5,000 requests per second** for a fintech trading startup.
- Designed a full-stack creator analytics platform in **Next.js** and **React** backed by **Neon** serverless **PostgreSQL**, integrating platform APIs via **RapidAPI** to track real-time view metrics and detect sponsored content across **50+** creators via caption and bio parsing, accumulating **1M+** views.

ACHIEVEMENTS

WatrFall - "Most Popular", Top 1 of 100+ competitors (Columbia DevFest 2025) *February 2025*

NeuroTalent - "Best Use of Auth0", Top 1 of 60+ competitors (Columbia DivHacks 2024) *September 2024*

PROJECTS

DouDiZhu (斗地主) Game Engine with MCTS Bot *January 2026*

- Developed a full DouDiZhu rules engine in **C++20** including move parsing, legality checks, and logic.
- Designed a **Monte Carlo Tree Search** decision engine using **UCT selection, expansion, rollout simulation, and backpropagation** to choose the statistically strongest moves.
- Simulated engine performance through **thousands** of games to achieve an average **47%** winrate, on par with many professional players with a **~2%** margin of error, nearing performance of trained RL agents.

Linux Container Runtime *May 2026*

- Implemented a rootless Linux container runtime in **C** using **clone()** with multiple namespaces, **cgroups v2** for resource limits, **pivot_root()** for filesystem isolation, and **libcap**/**seccomp** for capability dropping and syscall filtering without root access.

Dynamic Memory Allocator *February 2026*

- Implemented a 64-bit dynamic memory allocator in C from scratch using an explicit doubly-linked free list with first-fit placement, boundary tag coalescing across all 4 adjacency cases, block splitting, and page-aligned heap extension via a custom **mmap-backed sbrk** model.
- Encoded block metadata in **64-bit bitfield headers** (60-bit size, allocated flag) with matching footer tags enabling O(1) coalescing, 16-byte doubleword alignment enforcement, and prologue/epilogue sentinel blocks for simplified boundary checks.

Linux Debugger *March 2026*

- Built a Linux debugger in C using **ptrace()** supporting single-step execution, software breakpoints via **INT3** (0xcc) injection with saved-byte restoration and **RIP** rollback, and byte-level memory read/write with word-alignment handling; implemented **ELF** stack backtracing by parsing section headers to locate **symtab/strtab** within the ELF file and walking saved RIP chains to resolve function names up to main().

C++ Build System *November 2025*

- Implemented a C++ build system clone with regex-based Makefile parsing, DAG dependency resolution with cycle detection, stat()-driven staleness checks, and a custom generic binary serialization library used to cache parsed rules across runs.

SKILLS & INTERESTS

Languages: C, C++, Python, Java, JavaScript, TypeScript, Go, SQL, Bash, x86-64 Assembly

Tools: Linux, UNIX, Docker, Kubernetes, Vulkan, Git, PostgreSQL, Neon, Grafana, Loki, Node.js, Next.js, React