Jacob Lane Moore

Huntsville, AL 35806 ◇ jlmoors001@gmail.com ◇ (256) 701-7888

Education

MS, Georgia Institute of Technology, Computer Science — Expected 2027

- GPA: **4.0/4.0**
- **Coursework:** Hypervisors, Distributed Systems, Software Architecture, Compilers

BS, Mississippi State University, Computer Engineering — Graduated

- GPA: **4.0/4.0**
- **Coursework:** Embedded Systems, Systems Programming, Computer Architecture, Operating Systems

About Me

Languages	C++/C, Python
Development	Linux, Neovim, VSCode, Docker, CMake, Make, Zsh/Bash, Git
ML/DL	PyTorch, TensorFlow, RLlib, Scikit-Learn, Pandas, Grafana, Matplotlib
Certs	Neural Networks and Deep Learning (DeepLearning.AI, Andrew Ng), CompTIA Security+

Experience

Software Design Engineer, Torch Technologies – Huntsville, AL — 05/2023 – Current

- Developed modular C++ libraries within a logically clocked, microservices-driven training pipeline, ensuring deterministic synchronization of observations, actions, and rewards between a reinforcement learning algorithm and a high-fidelity simulation environment.
- Designed and extended a plugin-based architecture for reinforcement learning components (observations, rewards, actions), utilizing dynamic linking and polymorphism to enable experimentation without recompilation.
- Implemented a domain-specific query language using lexer/parser generation tools, enabling users to define complex observation structures and compositions through simple configuration files, thereby reducing boilerplate and enhancing flexibility.
- Integrated efficient serialization protocols (e.g., FlatBuffers) and custom UDP-based messaging to enhance inter-service communication, while supporting asynchronous, event-driven pub/sub mechanisms for responsive and scalable distributed systems.
- Developed a versatile C++ data model abstraction that dynamically loads, transforms, and batches datasets for a variety of deep learning models, with specialized support for the state-of-the-art forcasting model Temporal Fusion Transformer (TFT), and built a custom ONNX runtime abstraction to enable efficient model inference.
- Engineered a service allocation framework to enable parallel experience gathering for reinforcement learning (RL) training by dynamically isolating sets of microservices for use by vectorized environments, supporting containerized deployments to maximize hardware utilization and system throughput for large-scale RL experiments.
- Developed and maintained a versatile data recording utility library abstracted for seamless integration into any microservice, supporting both real-time distributed systems with asynchronous, time-driven recording and logical-time systems with synchronous, blocking operations to capture structured data for debugging, performance profiling, and offline analysis.

- Improved DevOps workflows by supporting environment setup and build automation, primarily maintaining the Linux environment (spanning multiple distributions over time), and aided containerized development and deployment efforts, accelerating development cycles and streamlining team operations.
- Leveraged CMake expertise to maintain, extend, and stabilize build systems across codebases, integrating numerous third-party libraries and ensuring reliable, efficient builds across diverse environments.
- Mentored teammates in software design and C++ best practices, contributing to the team's growth in developing high-quality code, debugging, and efficient development workflows.

Software Engineering Intern, Torch Technologies – Huntsville, AL 05/2020 – 08/2022 (Summers/Winters)

- Integrated internal C++ microservices with simulation pipelines and logical-time scheduling, enabling massive-scale data generation and reinforcement learning dataset collection, while ensuring strict temporal synchronization among distributed components.
- Developed Python/C++ data processing pipelines to run, record, and analyze thousands of simulation scenarios, applying machine learning techniques (Gradient Boosting, KNN, DNN) to identify the best predictive models, and visualizing results with Grafana for performance tuning.
- Created a Python-based GUI tool for interactive manipulation of high-resolution elevation and imagery data, automating the segmentation and export of complex terrain tiles into Unreal Engine—accelerating environment setup for simulation and training tasks.
- Extended Unreal Engine's C++ front-end with new DIS packet handling and deterministic network replay capabilities, enabling real-time visualization of projectiles, detonations, and scenario events in distributed simulations, and ensuring reproducible debugging sessions.
- Authored custom plugins and tooling to enhance the simulation-to-ML workflow, enabling rapid iteration and integration of new sensor models, scenario parameters, and training conditions without impacting core simulation integrity.

Undergraduate Research Assistant, Mississippi State University CSE Department 09/2020 – 05/2022

- Strengthened machine learning foundations by researching and applying advanced techniques (e.g., recurrent neural networks, GANs, transfer learning) to predictive maintenance tasks, building a solid theoretical and practical base for subsequent AI projects.
- Engineered comprehensive data acquisition pipelines that interfaced directly with specialized sensing hardware, converting raw fiber optic signals into actionable training datasets for neural network modeling.
- Explored innovative anomaly detection and time-series clustering algorithms, refining model selection and leveraging deep learning architectures (e.g., LSTMs) to enhance fault prediction accuracy and robustness.

SKILLS

- Expert in C++ OO design (factory, strategy, abstraction), data structures, memory management, and multithreading.
- Proficient in distributed systems, IPC mechanisms (ROS2 pub/sub, UDP + Flatbuffers), and synchronization techniques.
- Skilled in building and maintaining complex codebases with CMake, Python bindings, and containerized CI/CD workflows.
- Experienced in integrating ML/DL frameworks (RLlib, SB3) with simulation environments, optimizing training pipelines and inference performance.
- Adept at embedded/systems-level concepts, logical time simulation, and cross-platform development on Linux/Windows.

PROJECTS

Autonomous RC Car (Minesweeper):

- Solely engineered the entire software stack for a low-cost, disposable autonomous RC car designed to sweep for and detonate landmines.
- Architected a robust ROS 2 pub/sub service-oriented navigation system fusing raw IMU/GPS data for precise Ackermann steering control via electronic speed controllers.
- Implemented a Python-based command controller integrating base station input and RTK correctional data over a mavros radio, ensuring accurate real-time navigation.
- Developed all supporting base station software and communication interfaces, seamlessly linking computational logic with field operations.

Tiger Language Compiler:

- Designed and implemented a C++ compiler for a functional, ALGOL-like language (Tiger), producing optimized MIPS assembly.
- Introduced a TAC intermediate representation and implemented two register allocation strategies:
  - Chaitin's Naive Allocation: A simpler, less optimized approach relying on heuristic allocation.
  - Briggs Optimization: A comprehensive scheme incorporating control-flow and liveness analysis, as well as instruction cost metrics to minimize load/stores and branches.
- Achieved best-in-class performance on benchmark tasks, reducing instruction counts by a factor of 10 compared to peer solutions.

Before and After Creator:

- Built a locally hosted web application leveraging unsupervised ML and deep learning to match before/after car detailing images.
- Automated bulk image processing, generating side-by-side collages for a detailing business, improving workflow efficiency and consistency.
- Engineered a custom pairing algorithm to reliably identify corresponding images and streamline final outputs for user download.