

---

# 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

---

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

## EXPERIENCE

---

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

- Developed modular and configurable C++ libraries, utilizing modern OO design patterns, within a logically clocked, microservices-driven AI training pipeline, ensuring deterministic synchronization of actions, observations, and rewards between a reinforcement learning algorithm and a high-fidelity simulation environment.
- Designed and extended a plugin-based architecture for reinforcement learning components (actions, observations, rewards), utilizing dynamic linking and polymorphism to enable quick experiment iteration without recompilation.
- Implemented a domain-specific query language using lexer/parser generation tools and grammar, 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 forecasting 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 groups of services for use by vectorized environments, utilizing docker networks and containerization 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, event-driven recording to capture serialized data for debugging, performance profiling, and offline analysis.

- Improved build and deployment pipelines by maintaining complex CMake-based build systems, integrating numerous third-party libraries, and supporting containerized and Linux-based environments, accelerating development cycles and ensuring reliable, efficient builds across diverse platforms.
- Contributed bug fixes and enhancements to a division-wide C++ microservices toolbox, strengthening the reliability, maintainability, and performance of shared distributed infrastructure leveraged across multiple projects.
- Implemented custom C++ plugins within AFSIM to create “shadow sensors,” enabling the simulation of alternative sensor capabilities during legitimate simulation runs without impacting baseline simulation state and fidelity, thus expanding analytical depth for AI-driven training and experimentation.

**Software Engineering Intern**, Torch Technologies – Huntsville, AL

05/2020 – 08/2022

(Summers/Winters)

- Developed Python/C++ data generation/processing pipelines to run, record, and analyze thousands of simulation scenarios, applying machine learning techniques (Gradient Boosting, KNN, DNN) to train predictive classification models for real-time threat assessment and decision-making.
- 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 visualization projects.
- Extended Unreal Engine visualization C++ front-end with additional DIS packet handling, enabling real-time visualization of projectiles, detonations, and scenario events in distributed simulations.

**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

---

### • Core Software Architecture & Design

- Advanced proficiency in C++17/20 with strong OO design principles (Factory, Strategy, Singleton) and extensive UML modeling
- Expertise in modular, layered, microservices, and service-oriented architectures
- Experience applying MVC and related design patterns for clean and maintainable presentation layers

### • Operating Systems & Systems Programming

- In-depth understanding of kernel architectures, CPU schedulers, memory managers, and the MMU’s role in virtual memory management
- Strong knowledge of POSIX APIs, IPC mechanisms, and deterministic vs. non-deterministic operating system behaviors
- Skilled in concurrency (multithreading, synchronization primitives), low-level memory management, and event-driven design

### • Distributed & Parallel Computing

- Familiarity with messaging and serialization protocols (gRPC, RabbitMQ, Protocol Buffers, FlatBuffers)
- Experience with containerization (Docker) and virtualization (QEMU/KVM)
- Understanding of parallelization frameworks (MPI, OpenMP) and performance optimization techniques

- **Machine Learning Integration**

- Practical experience integrating ML/DL frameworks (ONNX, PyTorch, TensorFlow) into simulation pipelines
- Working knowledge of deep learning architectures (GANs, Transformers, LSTMs) and core ML concepts (backpropagation, activation functions)

- **Collaboration & Methodology**

- Proficient in Git-based workflows, code review processes, and industry-standard coding guidelines (Google Style, C++ Core Guidelines)
- Experience with Agile methodologies (Scrum, Kanban), unit testing (gtest, Catch2), and profiling tools (gdb, Valgrind)
- Committed to continuous improvement through static analysis (clang-tidy) and iterative development practices

## **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 user input to the vehicle's control system.

### **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.