Max Freeman

max_freeman@berkeley.edu | (607) 327-7804 | linkedin.com/in/maxfreeman | Mountain View, CA

Education

University of California, Berkeley **August 2024 - May 2025**
Master of Engineering in Mechanical Engineering, *Concentration in Controls & Robotics* GPA: 3.98

Cornell University **September 2020 - May 2024**
Bachelor of Science in Mechanical Engineering GPA: 3.77 | *Magna Cum Laude*

Skills

Embedded Systems: Sensors & Actuators | Communication Protocols (CAN, I2C, UART) | Oscilloscope Debugging
Prototyping & Assembly: CAD (SolidWorks) | 3D Printing & Rapid Prototyping | Manual Mill & Lathe
Programming & Tools: Python | C++ | ROS2 | MATLAB | Simulink | Git | Linux | Bash | PCAN-View | Jira
Robotics & Controls: Feedback Control (PID, LQR, MPC) | State Estimation & Sensor Fusion | Dynamics & Kinematics

Highlighted Experience and Projects

System Integration Intern, *Toyota Research Institute* **July 2025 - Present**

- Support cross-functional integration of mechanical, electrical, and software systems to enable rapid prototyping and testing of next-generation research vehicles.
- Integrate a steering wheel encoder into a prototype vehicle, taking ownership of CAN communication setup, wiring and connector fabrication, and the design of a custom 3D-printed mount using SolidWorks.
- Develop a ROS2 node in Python to convert raw steering angle data into vehicle yaw rate estimates using a kinematic bicycle model, publishing the signal to the localization stack to improve pose estimation.
- Reverse-engineer undocumented CAN signals by operating vehicle components while scanning logs in PCAN-View, isolating message patterns to identify key control signals.

Lead Robotics Engineer, *Multimodal Autonomous Platform Design (MEng Capstone)* **September 2024 - May 2025**

- Led a team of 4 engineers in developing a hybrid ground-aerial robot, owning hardware bring-up, control system implementation, and testing strategy across both driving and flight modes.
- Built and deployed a ROS2-based flight controller in Python using a cascaded PID architecture with integrated safety checks; validated stability and responsiveness through iterative flight tests.
- Designed and executed a staged flight validation plan, leading 10+ tethered flight tests to verify subsystem functionality, capture debug logs, and identify failure modes while minimizing hardware risk.
- Developed custom Python-based logging and visualization tools to support hardware testing of both control modes, enabling rapid debugging and performance validation.
- Analyzed test logs to isolate root causes of instability and control issues, iteratively refining controller behavior to improve system robustness and tracking accuracy.

Control of Autonomous Flight Project, *University of California, Berkeley* **September 2024 - December 2024**

- Developed a quadcopter flight controller in C++, achieving precise attitude, altitude, and position control.
- Implemented sensor fusion models in C++, utilizing data from optical flow, IMU, and Time-of-Flight sensors to improve sensor data accuracy and provide precise feedback for control.
- Diagnosed and resolved stability issues through targeted subsystem testing, sensor debugging, and PID controller tuning to validate fixes and improve system response.

Hardware Test & Integration Intern, *Lit Motors* **June 2024 - July 2024**

- Contributed to test infrastructure development and subsystem design refinement at a fast-paced startup building a two-wheeled self-balancing vehicle actuated by Control Moment Gyroscopes (CMGs).
- Led development of a dual-plane dynamic balancing test rig for CMG evaluation, integrating structural hardware, sensors, and a belt-drive system to identify weight imbalances and reduce vibrations.

Embedded Autonomous Vehicle Project, *Cornell University* **January 2024 - May 2024**

- Managed end-to-end development of a small-scale autonomous ground vehicle, integrating sensors and motor drivers on an Arduino Nano and implementing control logic in C++ and Python.
- Interfaced with Time-of-Flight and IMU sensors over I2C, developing sensor fusion algorithms and software-based filters in C++ to minimize sensor output noise by over 50%, enhancing sensor accuracy.
- Implemented and tuned distance-based PID controllers to improve tracking, reducing settling time by 40%.
- Diagnosed hardware and sensing issues using oscilloscope traces and telemetry data, resolving integration faults, and improving system performance.