☎ *(651) 261–2862*
✉ *john.r.harwell@gmail.com*
https://jharwell.github.io
Google Scholar
Github

John Harwell

Summary

- Experienced software architect with 10+ years of experience across domains.
- Skilled multi-agent systems researcher and proven interdisciplinary collaborator, leader, mentor, and problem-solver.

Education

2016–2022	**Ph.D. in Computer Science**, *University of Minnesota*, Twin Cities.
2016–2018	**M.S. in Computer Science**, *University of Minnesota*, Twin Cities.

Areas of Expertise

Theory	**Modeling**: Bio-inspired modeling, stochastic processes, differential equations, graph theory, queueing theory **Algorithms**:Parallel, greedy, bio-inspired, graphical, task allocation
Embedded Systems	**OS**: Petalinux, FreeRTOS, RTEMS, bare-metal **Architectures**: ARM Cortex-M7, SPARC LEON2 **Middleware**: QEMU **Design**: Hardware/software trade-offs, hotfix debugging
Multi-agent Systems	**OS**: Linux (ubuntu, debian, raspbian) **Platforms**: ARGoS, Gazebo, ROS1, ROS2, Turtlebot3 **Behavior Design**: Vector fields, bio-inspired modeling, decentralized task allocation **Analysis**: Differential equations, cooperative algorithms, metric design, imperfect sensor/actuator compensation
High Performance Computing	**Platforms**: SLURM, PBS **Optimization**: Profiling,architectural/memory/cache analysis, algorithm analysis

Technical Skills

Languages	**Expert**: C: embedded, systems programming C++: 11/14/17/20 with templates, metaprogramming **Proficient**: C: kernel programming, python **Familiar**: Fortran, bash, fish, MATLAB
Software Development	**Architecture**: Design patterns, OOP, polymorphism **Devops**: GitHub/Gitlab CI/CD, Ansible, Docker **Toolchains**: LLVM (clang-*), Intel (icx, VTune), GNU (gcc-*) **Tools**: cmake, Bazel, git, gdb, valgrind, OpenOCD, oscilloscope, JTAG, Conan, Black Magic Debug **Data Structures**: Graphs, trees, R-trees, Poisson queues, heaps, maps
Protocols	UART, I2C, SPI, NMEA
Libraries	STL, Boost, OpenMP, MPI, CMSIS, pandas

Experience

2024–present **Senior Algorithm Architect**, EpiSci, Minneapolis, MN.

- Led algorithmic development effort for UAV systems in GPS-denied environments using task allocation and Mobile Ahdoc Wireless Networks (MANETs).

2023–2024 **Embedded Development Lead**, Satelles, Minneapolis, MN.

- Demonstrated technical leadership by developing software engineering guidelines and methodologies for large software framework to support overall business goals.
- Facilitated meetings with key leaders to ensure timely decision-making and communication between stakeholders and assisted in technical personnel management in small teams.
- Design, implementation, and maintenance of a custom QEMU plugin to reduce risk in commercializing custom Position, Navigation, Timing (PNT) ASIC.
- Ported large software framework for embedded PNT receivers to custom ASIC.

2022–2023 **Postdoctoral Researcher**, SIFT, Minneapolis, MN.

- Developed models of flocking behaviors to extract control policies and parameters automatically from trajectory data to estimate physical properties and limits of military vehicles.
- Reduced debugging time by enhancing in-house tooling for efficient visualization of multivariate spatio-temporal data of large-scale multi-agent systems.
- Contributed to business development through market research and proposal writing.

2016–2022 **Researcher**, University of Minnesota, Minneapolis, MN.

- Achieved publication of 9 papers at top conferences and journals, including 6 first author papers, through strong writing and organization skills, and collaboration with other researchers.
- Derived cuboid structure model using graph theory to develop simple algorithms to provably manipulate graphs (structures) from one state to another.
- Demonstrated robust predictions of steady-state collective foraging behaviors up to practical engineering limits using differential equation modeling.
- Showed that the origin of collective intelligence in task allocating swarms lies in self-organized learning task relationships, rather than costs.
- Reduced development cycles and increased utility of automated design methods through better measurements for design principles of multi-agent systems.

2016–2022 **Research Group Leader**, University of Minnesota, Minneapolis, MN.

- Mentored highs school and undergraduate students interested in AI, robotics, and academic research to apply for grants, publish original research, and present at workshops.
- Managed parallel undergraduate research projects through weekly meetings, check-ins. Helped students to develop as independent researchers: fostered excitement in research through freedom of topic choice and technical approach, and clarity in student goals through project scoping.

2013–2016 **Research Engineer**, Southwest Research Institute, San Antonio, TX.

- Led flight software development on NASA subcontract for Cyclone Global Navigation Satellite System (CYGNSS) in collaboration with the University of Michigan.
- Developed prototype NASA cFS-compatible file system with configurable memory footprint and increased robustness for flash-based media.

Open-Source Projects

2016-present **Author**, Core Swarm Library, .

- Middleware-esque C++ library providing a transparent, zero-cost API to different robotics platforms (ROS1, ARGoS, etc.), for both real and simulated robot types.
- Computationally optimized for efficient execution with systems of over 10,000 robots on supercomputing clusters and on real systems of Raspberry PI-powered TurtleBot3 robots.

2016-present **Author**, C/C++ Development Core, C , C++ .

- Focused on reusability to kickstart development on any C/C++ project.
- C++ modules: metric collection, logging, spatial reasoning, data structures.
- C++ generic design patterns: decorator, factor, FSM, prototype, singleton, visitor.
- C modules: data structures, minimal stdlib, publisher/subscriber bus, logging mechanisms for embedded applications.

2017-present **Author**, SIERRA: Scientific Method Automation, .

- Given a user query of an independent variable over a range, generate experimental inputs, run experiments, process results, and generate visualizations.
- Plugin-based python framework supports any agent type, platform (e.g., simulator, ROS1), or execution environment (e.g., supercomputing cluster, real robot).