# Johnathan J. Flaggs

LinkedIn: www.linkedin.com/in/johnathanflaggs
Email: work.johnathanflaggs@gmail.com
Website: resume.johnathanflaggs.com
Phone: +1.949.414.9545

Robotics Software Engineer & Architect with a focus on backend system development for robotics applications. Experienced within numerous industries, languages, and technologies. Strong aptitude for mechanical design and has a successful record of leading projects from proof of concept into production.

## Education

**University of California, Riverside & Davis** **Sep. 2010-Jun. 2014**

- UCR: BSME with concentration in Control Theory under Department of Mechanical Engineering (BCOE)
- UCD: Control Theory Concentration under Department of Mechanical and Aerospace Engineering

## Technical Toolset

**Development Languages & Environments:**

| | | | | |
|---|---|---|---|---|
| ▪ Visual Studio | ▪ Codeblocks | ▪ SVN, GIT, Jira | ▪ Ignition SCADA | ▪ Node.js |
| ▪ Atmel Studio | ▪ Embedded C, C++ | ▪ C# .NET | ▪ MATLAB, Python | ▪ HTML, CSS, JS |
| ▪ RSLogix/FactoryTalk | ▪ Beckhoff/TwinCAT | ▪ Festo, MagneMotion | ▪ Fanuc TP, Karel | ▪ Cognex, iRVision |
| ▪ RS-232/485, CANbus | ▪ Allen Bradley | ▪ IEC 61131-3 | ▪ gRPC, JSON, SQL | ▪ EtherCAT, EtherNet/IP |

## Professional Experience

**Sr. Robotics Software Engineer at CarbonCapture** **Jun. 2023-Present**

Designed and implemented system level software architecture and automation solutions for modular Direct Air Capture (DAC) reactors. Owned the system architecture and core control software stack. I focused on ensuring robustness, scalability, and long-term maintainability across current and future product deployments. Worked cross-functionally with mechanical, process, and data teams to translate physical system requirements into reliable, production-ready software.

- **Architected** macro and micro level software systems defining the core DAC control platform ( CCI Software Suite)
- **Developed** and maintained high-reliability automation software in C#, C++, and TwinCAT
- **Refactored** legacy codebase using object oriented architectures–significantly improving scalability and maintainability
- **Reviewed** source code at the commit level to enforce correctness, consistency, and engineering best practices
- **Proactively** evolved the software to minimize the impact of future hardware or system changes
- **Collaborated** with mechanical, process, and data science engineers to align software layers with project requirements
- **Integrated** third party hardware, control devices, and externally developed software into the primary control stack
- **Mentored** engineers through technical problem-solving, code reviews, code school, and architecture discussions
- **Reduced** cost by eliminating the need for expensive absolute encoders–resulting in a phase 1 savings of over $100K
- **Proposed** alternative solution for door actuation which replaces motors with pneumatics to reduce cost and complexity
- **Contributed** heavily to company culture by building up infrastructures such as git, jira, code reviews, and code school

# Professional Experience

**Sr. Controls Engineer at Essentium 3D** **Jun. 2021-Jun. 2023**

Leading robotics software architecture and development for high-speed industrial 3D printers. Ensuring that our team builds a software core that is sufficiently robust, scalable, and maintainable to support the product line. I refactored the existing code base into OOP scalable modules/layers. This allowed us to quickly change hardware and ultimately extend from plastics into metal printing.

- **Primarily** used Clearcore Teknic and Agito controllers for real-time control and C# for the application and UI layers
- **Lead** the design and architecture of our core software which allowed us to expand from plastic to metal printing
- **Mentor** Jr. Engineers on approaching complex problems and upkeeping best coding standards
- **OOP** and heavy emphasis on robustness, scalability, maintainability, and patterns in C# and C++
- **Anticipate** and mitigate the impact of future design changes on the software layers
- **Design** of macro and micro software architectures which define the core product
- **Manage** and review source control on a per-commit basis
- **TI & STM32** used for embedded controls (C/C++)
- **RTOS** used for multitasking on STM32

**Sr. Robotics Software Engineer at Seagate Technology** **Mar. 2017-Jun. 2021**

Developing software for cutting edge processes in digital storage technology. I focused on developing complex motion control routines for pico-scale positioning, calibration routines, state-machine process synchronization, and machine vision integration using the Congnex SDK.

- **Primarily** used Beckhoff and Delta Tau for real-time control and C#/C++ for vision, application, and UI layers
- **Motion Control** Kinematics, pick-n-place, multi-axis coordination, Quantum HSM framework
- **OOP** Heavy emphasis on encapsulation, inheritance, polymorphism, and robust design patterns
- **User Interface** Allowing users to fluidly interact with the multi-threaded application
- **User Interface** Allowing users to fluidly interact with the multi-threaded application
- **Machine-Vision Calibration** to establish precision robot coordinate system
- **Version Control** Using SVN, TFS and Agile/Scrum using Jira
- **Support** core vision libraries in C/C++
- **Cognex** VisionPro API integration

**Controls Engineer at Sorenson Engineering Inc.** **Sep. 2014-Mar. 2017**

Leading controls software development for high-volume manufacturing. I developed core software and worked closely with Mechanical Engineers to build proof of concept products (R&D environment).

- **Primarily** used Beckhoff, Festo, Maxxon, and Jenny Science for real-time control with TwinCAT for HMI/UI
- **Motion Control** – Kinematics, motor sizing, pick-n-place, multi-axis synchronization, and axis coupling
- **Increased** machine PPM by decreasing rotor inertia ratio by 140% for tighter position control (VSIII)
- **Fieldbus Integration** – Integrating multiple OEM device nodes into the EtherCAT controls network
- **Vision Inspection** – Driving digital cameras (Cognex, Baumer/VeriSens) via native C++ SDKs
- **Eliminated** need for expensive camming software license and a measurement sensor