

PHILIP WESLEY FACKLER

Contact: 2213 Maplewood Drive, Knoxville, TN 37920 • (270) 871-4091 • philip.fackler@gmail.com

EDUCATION	Doctor of Philosophy (PhD) in Computational Engineering December 2017 The University of Tennessee at Chattanooga, Chattanooga, TN Thesis: A Physics-Based Adaptive Point Distribution Method for Computational Domain Discretization Advisor: Dr. Christopher Bruce Hilbert Cumulative GPA: 4.0
	Master of Science (MS) in Computational Engineering December 2013 The University of Tennessee at Chattanooga, Chattanooga, TN Thesis: Physics-Based Point Placement by Particle Dynamics Simulation Advisor: Dr. Steve L. Karman, Jr. Cumulative GPA: 4.0
	Bachelor of Arts (BA) in Mathematics, cum laude May 2008 Asbury College, Wilmore, KY Advisor: Dr. David Coulliette Cumulative GPA: 3.73
EXPERIENCE	Oak Ridge National Laboratory , Oak Ridge, TN April, 2019 – present Position: Research Software Engineer
	Branch Technology , Chattanooga, TN July, 2017 – March, 2019 Position: Algorithms Engineering Primary developer of the company software framework. This includes a data/operator hierarchy which blends object-oriented and generic programming styles; structured and unstructured meshing data structures, including a generic multi-block mesh data set; a logging system; container adapters; low-overhead static mixins; signals and slots; a polymorphic, object-oriented geometry interface; spatial subdivision trees; geometric and applied math tools, including a dual-number implementation; and various utilities, including a unit testing framework. Significantly refactored and expanded the company's collection of components (written in C#) for use in the Grasshopper Rhino plugin, which are used to generate robotic path plans for Branch's large-scale free-form 3D printing method. This has involved extending the Grasshopper API with specialized base classes which provide our components a consistent means of access to our custom data sets. Established a CMake build system for all of our projects. Established software quality assurance practices including integrated software evaluation targets (using CTest and our own C++ testing framework) and detailed documentation using Doxygen. Established consistent version control practice using Git. Authored company coding guidelines for both style and quality standards (<i>i.e.</i> , best practice) and Git commit and branching guidelines. Administer issue-tracking projects in Jira and continuous integration in Bamboo. Make use of open-source libraries such as VTK, Eigen, and boost. Work in an interdisciplinary team of developers, architects, and engineers, balancing research goals and production deadlines with two-week, six-week, and quarterly meetings. Meet daily in software team to maintain priorities for incremental development as well as digital production.
	Motion View Software , Chattanooga, TN January, 2016 – June, 2017 Position: Research and Development Software Engineer Worked on several algorithms for locating anatomical landmarks on 3D models of faces and teeth. Began work on generating vertical support structures for 3D printed geometries. This involved writing custom VTK pipeline algorithms. Worked with customer support team to prioritize software requirements issues based on customer feedback.

Pointwise, Inc., Fort Worth, TX

May – July, 2015

Position: Product Development Intern

Wrote code for reading and writing meshes in various file formats. Researched methods for propagating curvature in higher-order meshes, namely radial basis functions and mean value coordinates.

University of Tennessee at Chattanooga, Chattanooga, TN August 2011 – October, 2017

Position: Graduate Research Assistant

Conducted independent research. Met weekly with advisor and group of peers to discuss progress and provide mutual feedback.

PROGRAMMING

- C++17 (advanced, 10 yr)
Modern object-oriented and generic programming styles, RAII, template metaprogramming
Libraries (in order of experience level):
 - [C++17](#) (advanced)
 - [Kokkos](#) (intermediate)
 - [VTK](#) (intermediate)
 - [Eigen](#) (intermediate)
 - [OpenMP](#) (intermediate)
 - [MPI](#) (intermediate)
 - [boost](#) (intermediate)
 - [CGAL](#) (elementary)
 - [Qt](#) (elementary)
 - [OPT++](#) (elementary)
 - [hdf5](#) (elementary)
- C# (intermediate, 2 yr)
Libraries:
 - [RhinoCommon](#)
 - [Grasshopper](#)
- [CMake](#) (advanced, 4 yr)
- Other**
 - C (intermediate)
 - [Tcl/Tk](#) (intermediate)
 - [Glyph](#)
 - [Julia](#) (elementary)
 - [Python](#) (elementary)
 - bash

OTHER TECHNICAL SKILLS

- Operating Systems**
 - Linux
 - Windows
- Development Tools**
 - [Git](#)
 - [GitExtensions](#)
 - [CMake](#)
 - CTest (unit testing)
 - CPack with [NSIS](#) (packaging)
 - [Doxygen](#) (code documentation)
 - Gitlab
 - Microsoft Visual Studio
 - [Atlassian](#):
 - Bitbucket (repository management, wiki)
 - Jira (bug tracking and kanban)
 - Bamboo (continuous integration)
 - [LeanKit](#)
 - [Vim](#)
 - [gdb](#), [valgrind](#)
 - [make](#)
 - Markdown
- Compilers**
 - MSVC
 - gcc
 - clang
- Productivity Tools**
 - [LyX](#)
 - [LibreOffice](#)
 - [Slack](#)
 - Google Drive and Docs
- Meshing and Geometry**
 - [Pointwise](#)
 - [Rhino](#)
- Visualization and Mathematics**
 - [VisIt](#)
 - [ParaView](#)
 - [Gnuplot](#)
 - Matlab/[Octave](#)
 - Maple

**PROFESSIONAL
INTERESTS**

- Modern C++ programming
- C++ standardization
- Software architecture
- Open source software
- Spatial subdivision and meshing data structures
- Computational geometry
- Scientific computing, concurrency and parallelism
- Performance portability
- Numerical mathematics
- Software engineering, best practices, and quality
- Agile team software development

**GRADUATE
COURSES**

- Linear Algebra and Matrix Theory
- Partial Differential Equations
- Numerical Analysis (I, II)
- Numerical Solution of Partial Differential Equations
- Techniques of Applied Mathematics
- Advanced Programming for Physical Simulation
- Introduction to Parallel Algorithms
- Parallel Scientific Supercomputing
- Viscous Flow Theory
- Viscous Flow Computation
- Grid Generation (I, II)
- Computational Fluid Dynamics (I, II)
- Computational Design
- Introduction to the Finite Element Method
- Advanced Topics in the Finite Element Method