

# Tim Straubinger

## Curriculum Vitae

[straubinger.tim@gmail.com](mailto:straubinger.tim@gmail.com)

[timstr.github.io](https://timstr.github.io)

### EXPERIENCE

---

**Full-time Software Developer** - [Vital Mechanics Research Inc](#) (Jan 2019 - present)

- Geometry processing and physical simulation in C++
- Various duties related to code migration, build system and continuous integration, developing novel algorithms and data structures, designing interfaces and file formats, and designing and testing new and alternative architectures.

**Undergraduate Research Assistant** - University of British Columbia  
(Jan 2018 - April 2018, Sep 2018 - Dec 2018)

- Researched efficient techniques for communicating 3D triangle mesh data between C++ server and JavaScript client

**Software Development Intern** - [Vital Mechanics Research Inc](#) (May 2018 - Aug 2018)

- Designed and implemented a user interface for viewing and customizing physical properties of digitally simulated cloth materials
- Developed inter-process communication between JavaScript front-end and C++ back-end using [JSON-RPC](#)

**Teaching Assistant** - University of British Columbia (Sep 2017 - Jan 2018)  
CPSC 121 - Models of Computation

- Taught multiple weekly labs, helping students implement, debug, and reason about digital circuits built using electronics and simulation software
- Taught students during tutorial sessions and provided detailed guidance with problem solving

**Student Assistant, Borrower Services** - Walter C. Koerner Library (Sep 2014 - Apr 2017)

- Processed, sorted and re-shelved books returned to the library
- Sorted and straightened book stacks

### EDUCATION

---

**BSc., Computer Science** - University of British Columbia (2014 - 2019)

- Graduated with distinction
- Dean's Honour List (2017 Winter Session)

## SKILLS

---

Programming Languages		Frameworks and Libraries	Tools and Environments
Proficient in	Familiar with		
<ul style="list-style-type: none"><li>• C++</li><li>• C</li><li>• JavaScript</li><li>• TypeScript</li></ul>	<ul style="list-style-type: none"><li>• C#</li><li>• CUDA C</li><li>• Erlang</li><li>• GLSL</li><li>• Haskell</li><li>• Java</li><li>• Julia</li><li>• Prolog</li></ul>	<ul style="list-style-type: none"><li>• <a href="#">Boost</a></li><li>• <a href="#">React</a></li><li>• <a href="#">SFML</a></li><li>• <a href="#">SDL</a></li><li>• <a href="#">three.js</a></li></ul>	<ul style="list-style-type: none"><li>• git</li><li>• CMake</li><li>• conan</li><li>• Docker</li><li>• Visual Studio</li><li>• Visual Studio Code</li></ul>

---

## PERSONAL WORK

More information on each project, as well as examples and results, can be found at [timstr.github.io](https://timstr.github.io)

**Flosion** – A visual programming language for synthesizing and modifying streams of sound

- Allows streams of sound to be modified and combined by constructing a visual flow graph where nodes are various DSP units and edges define dependencies
- Users can build extremely customizable and flexible synthesizers, melodies, and effects
- Sound processing units can be queried many times in parallel and can have multiple concurrent states
- Stateful information from processing units can be used to parameterize the behavior of dependencies
- Results can be listened to interactively in real-time
- Written in C++ using SFML

**Rigid Body Physics Engine** – For 2D platformer video game

- Implements collision detection and resolution of boxes and circles
- Collisions are resolved using impulses, while bodies can be manipulated at the level of forces, impulses, and positions, both linear and angular.
- Runs in real-time with hundreds of shapes colliding
- Written in C++, rendered using SFML

**Fractals** – In two and three dimensions

- Mandelbrot, mandelbox, buddhabrot, and various custom hybridized fractals in 2D, rendered using the escape-time algorithm and smoothing techniques, as well as texture mapping
- Mandelbox fractals and voxel shapes in 3D, rendered using CPU and GPU, implemented in C++ and GLSL, using SFML and SDL
- Images are anti-aliased at locations of high contrast using supersampling
- CPU ray-tracer allows for basic direct illumination, depth of field effects, and fog

---

## OTHER EXPERIENCES

**SIGGRAPH 2018, Vancouver** – Attended research presentations, technical demonstrations, trade show, and screenings of animated short films