**Spencer T. Parkin**

Utah, USA

(801) 970-4578

[spencer.parkin@proton.me](mailto:spencer.parkin@gmail.com)

<https://github.com/spencerparkin>

**Employment History & Experience**

**Lead Software Engineer** 2019 – Present

*L3 Driver Training Solutions, Salt Lake City, Utah*

* Developed, single-handedly, a new graphics solution using Unreal Engine 5 to replace their old outdated solution which used Havok Vision (no longer supported.) This involved writing all necessary C++ code and porting assets from the old graphics system to the new one.
* Wrote Python scripts to aid in asset porting, both in and outside of 3Ds Max.
* Wrote a lot of C++ that ran in editor to aid the porting process as well.
* Wrote a Python script to handle our Unreal building of code and assets, as well as the creation of our Unreal build installer used for distribution.
* Helped others get setup with Unreal Editor and/or a new development environment involving Unreal Engine.
* Ported the entire code-base from VS2010 to VS2019, detangling DLL hell in the process and removing all build warnings (of which there were thousands upon thousands.)
* Developed a bridge between the simulation software and Unreal using WinSock with TCP/IP and multicast-IP/UDP.
* Developed a snow-mesh plowing system using Unreal’s dynamic mesh rendering capabilities. A CPU-intensive process, I performed all mesh cutting on a dedicated thread so as to prevent any negative impact on the framerate.
* Made many bug fixes and enhancements to the simulation software.
* Developed a day/night cycle Blueprint script to be used in each scene.

**Senior Software Engineer** 2016 – 2019

*3M Health Information Systems, Murray, Utah*

* Developed and maintained, in Python, a Selenium-based test-automation process that ran as part of our continuous integration cycle. Tests exercised the coding & reimbursement software developed by the company.
* Wrote back-end processes (in Python) to carry out tests and update results in MongoDB.
* Wrote a front-end PyQt-based application for seeing the status of all tests, allowing you to edit and/or create new tests, and kick-off tests to run on the server.
* Helped others learn the test automation system and take ownership of tests.

**Programmer** 2012 – 2016

*Avalanche Software, Salt Lake City, Utah*

* Developed, using C++, a particle authoring tool that artists could use to create particle effects for us in game. The tool was based on a custom windowing framework called Snipe, which was based on OpenGL and MFC.
* Contributed to the Snipe framework/API by creating new types of controls or widgets.
* Developed a live-authoring capability for the particle authoring tool so that artists could tweak emitters in the tool while immediately seeing the results in game as the game engine rendered.
* Helped artists get oriented with the particle authoring tool and how to use it effectively. Also took their feedback to heart to make enhancements to the tool.
* Developed features, using C++, in our game engine’s particle system run-time. For example, I developed a lens-flare effect using the particle system’s framework. These could be authored in the particle editing tool as a natural extension to the UI layout.
* Made several contributions to the vertex and fragment shader code used by the particle system. This required knowledge of HLSL and GLSL, as well as, of course, linear algebra, vector calculus, and 3D viewing techniques.
* Developed a Snipe-based tool for authoring character reaction graphs. This was an editor that designers or game-play engineers could use to wire-up scripts, of a sort, that could run on a character in game. (An equivalent to this in Unreal would be Blueprint, but not as extensive.)
* Was fully present for and contributed to the entire development life-cycles of the titles Toy Story 3 and Disney Infinity.
* Frequently worked with testers to reproduce and trap software defects, understanding where the code was going wrong and then coming up with and implementing a solution to the problem.

**Associate Programmer** 2007 – 2012

*Avalanche Software, Salt Lake City, Utah*

* Made many contributes to the core math library of the game engine to support vector and matrix operations, as well as making them compatible with an additional math library based on geometric algebra.
* Wrote optimizations in the math library for various operations using SIMD intrinsics. For example, finding the inverse of a 4x4 matrix.
* Developed the second iteration of the hog file system format used by our game engine. These were essentially archives of files and folders that could be mounted by the engine, making the contents available to various sub-systems for loading assets. Hog files could also nest hog files.
* Developed a tool for generating and viewing hog files using C++ and wxWidgets.
* Wrote optimizations for the particle system using SIMD intrinsics.
* Was fully present for and contributed to the entire development life-cycles of the titles Bolt and Cars 2.
* Developed a tool (using C++ and wxWidgets) that could be used to merge scene files to resolve merge conflicts.

**Lab Aide** 2003 – 2007

*Weber State University, Ogden, Utah*

* Helped fellow students with computer-related tasks while earning a 4-year degree (a B.S. in math.)
* Maintained lab equipment.

**Level 1 Programmer** 2001 – 2002

*Acclaim Entertainment, Sugar House, Utah*

* Developed, using C++, the front-end menu system for the game Legends of Wrestling II.
* Worked closely with artists and designers to bring their vision of the UI to fruition.
* Handled asynchronous streaming of 3D wrestler models into and out of memory as characters were selected on the character selection screen.
* Wrote a mesh deformation feature used by the character customization feature of the game. This allowed you to change the size of body-parts on your custom wrestler.

**Programmer Intern** 2000 – 2001

*Acclaim Entertainment, Sugar House, Utah*

* Worked closely with my mentor, Larry Hutcherson, to develop the particle system for the game Legends of Wrestling I.
* Under Larry’s guidance, implemented blood splatter (texel writes) on the wrestling mat. These coincided with blood particles hitting the mat.
* Learned the Perl scripting language.

**Education**

**Bachelor of Science in Mathematics** 2003 – 2007

*Weber State University, Ogden, Utah*

* While earning a 4-year degree, participated in math club, and submitted solutions to problems published in math journals.
* Earned a minor in computer science.
* Occasionally gave talks to the math club on interesting math puzzle solutions.

**High School Diploma** 1998 – 2001

*Viewmont High School, Bountiful, Utah*

* While in the first year of high school, played clarinet in the marching band.

**College Credit** 2000 – 2001

*Davis Applied Technology Collage, Kaysville, Utah*

* Spent half of the last year of high-school at a community college to earn college credit towards a computer science degree.

**Personal Programming**

In my spare time, I have written software exploring the following topics, just to name a few.

* Wrote a basic game engine using C++ with DirectX 11, and another with DirectX 12, both of which supported dynamic as well as static mesh rendering and dynamic lighting.
* Developed a means of importing rigs and their animations into my game engine, as well as being able to animate the skeletons and skin the meshes (real-time mesh deformations.)
* Developed a collision system that uses a BVH for broad phase and the GJK algorithm for narrow phase. It also supported ray-casting for picking and other purposes.
* Implemented the expanding polytope algorithm (EPA) to solve for convex hulls as well as the penetration depth between convex hulls (an extension of GJK.)
* Developed a font renderer using the free-type library that uses GPU instancing.
* Implemented real-time shadows using a depth pass to make a depth buffer used in the main render pass.
* Developed a basic PBR shader with bump-map capability.
* Wrote a framework of classes designed to handle CPU/GPU synchronization.
* Wrote classes for handling basic memory management for things like the upload heap and various descriptor heaps.
* Implemented basic rigid body dynamics that can simulate convex hull masses moving through space and colliding with one another.
* Developed my own C++ audio library that can load, save and work with audio files and sound-font files. It can analyze an audio clip to determine the fundamental pitch, play back MIDI files, and synthesize sound from a MIDI device, as well as add reverb to the produced sounds.
* Wrote my own ray-tracing renderer that can do reflections and refractions. It took advantage of multiple threads to produce the final image.
* Wrote a CPU-only renderer (not accelerated by a GPU) that can rasterize triangles. It took advantage of multiple threads to accelerate rendering.
* Developed a twisty-puzzle web page (<https://twisty-puzzle.onrender.com>) using WebGL and JavaScript.
* Developed a Cesium-based web app (<https://utahavamap.onrender.com>) that uses the Utah Avalanche Center’s API, again using JavaScript.
* Developed solitaire for the Web (<https://solitaire-8076.onrender.com>) using JavaScript and React.
* Wrote a card game that can do Klondike, Spider & Free Cell using DirectX 12 (without any engine layer.)
* Implemented the game of chess using C++ and wxWidgets. You can play against the computer which uses the Minimax algorithm (with alpha-beta pruning.) You can install it yourself, if you’d like, using the windows command: winget install SpencerSoft.Chess