Matthew Hamilton

matthewhamilton3141@gmail.com | linkedin.com/in/matthewhamilton3141 | github.com/matthewhamilton3141 | matthewhamilton.dev

Education

University of Waterloo — Waterloo, ON

Bachelor of Applied Science in Systems Design Engineering — *Sep. 2026 – May 2031 (Expected)*

- **Scholarships**: Alumni @ Microsoft Engineering Scholarship (1 recipient/year, facultywide)
- **Relevant Coursework**: Digital Computation (SYDE 121, C++), Linear Algebra (MATH 115)

Projects

gsplat-rt | *Python, TensorRT, CUDA, PyTorch, OpenUSD, Isaac Sim* — 2026

- Sustained 82.7 FPS end-to-end (12.1 ms/frame) on an NVIDIA A10G by architecting a multithreaded, lock-free real-time monocular pipeline converting live RGB video into a 3D Gaussian Splat scene and physics-ready collision mesh (OpenUSD export for Isaac Sim / Omniverse), run in NGC containers (`--gpus all` via the NVIDIA Container Toolkit) with host-driver/container-runtime debugging
- Cut depth latency to 6.3 ms/frame — 2.24× faster than TF32 at >0.9999 output correlation — by compiling a strongly typed FP16 TensorRT engine for Depth Anything V2 with zero per-frame GPU allocations
- Reduced trajectory error to 3.5 cm ATE on TUM RGB-D (vs. 5.7 cm for classical ORB) by deploying a SuperPoint + LightGlue learned SLAM front-end as an FP16 TensorRT engine at 7.4 ms/frame (18× over ONNX Runtime's CUDA EP)
- Held TSDF fusion to 0.06 ms/frame — under 0.5% of the pipeline's 12.1 ms frame budget — with a custom CUDA kernel feeding marching-cubes mesh extraction and PhysX collision export
- Drove a PPO navigation policy to 0 collisions in evaluation while beating a hand-tuned baseline on path efficiency (verified by a physics drop-test PASS in Isaac Sim 5.1) by training a one-step-lookahead safety shield on the reconstructed scenes (Gymnasium / stable-baselines3)

Retermina | *Tauri v2, Rust, React, TypeScript* — 2026 – Present

- Delivered a local-first, Claude-centric desktop terminal workspace with no cloud backend, token limits, or subscription by building on Tauri v2 (Rust + React/TypeScript), where a Rust backend spawns and multiplexes native PTY sessions over Tauri IPC with tools docked in a Cursor-style drawer
- Kept terminals — and the dev servers inside them — alive across app quit, crash, or auto-update by architecting a live-session layer that hosts every PTY in a separate re-exec'd process, reconnecting over a Unix-domain socket and replaying scrollback on relaunch, with a grace-TTL reaper reaping lingering daemons
- Turned it into a supervision layer for coding agents: embedded Claude Code with a live context-window gauge (parses session JSONL to chart context fill against the model window), per-project token and cost accounting, and cross-restart session resume that reconnects the agent and rehydrates its transcript
- Shipped universal (Apple Silicon + Intel) macOS releases via a GitHub Actions pipeline and a minisign-signed in-app auto-updater that verifies each download against a `latest.json` manifest before applying

Sketchstack | *Next.js, React, TypeScript, React Flow, Supabase, Tailwind CSS* — 2026

- Built a visual systems-design canvas that compiles a hand-drawn architecture diagram into a structured, mode-aware prompt for AI coding agents; guest-first — everything but cloud save works without an account
- Engineered an interactive graph editor on React Flow (`@xyflow/react`) — 50+ typed components with per-type icons, connect/reconnect edges, multi-select align/distribute with snapping, undo/redo, and a command palette
- Wrote a prompt generator that traverses the graph into a grouped spec (merging bidirectional connections), with swappable node packs and templates across four diagram modes: System Design, App/UI Flow, Database Schema, and Task Planning
- Built a Supabase backend — GitHub OAuth auth and row-level-secured Postgres for cloud-saved diagrams plus one-click read-only share links with rich previews; deployed on Vercel

Technical Skills

Languages: TypeScript, Python, Rust, C++, JavaScript, HTML/CSS
Frameworks & Libraries: React, Next.js, Node.js, Tauri, Tailwind CSS, PyTorch, OpenCV, stable-baselines3, Gymnasium, PyBullet
Tools & Platforms: Git, Linux, CUDA, TensorRT, ONNX, Docker, NVIDIA Container Toolkit, NGC, OpenUSD, NVIDIA Isaac Sim, Supabase, Vercel
AI / ML: 3D Gaussian Splatting, Depth Anything V2, reinforcement learning (PPO), visual SLAM, TensorRT engine optimization