Tom Wang

647-852-7705 | zxTomw.com | tom.wang3@uwaterloo.ca | linkedin.com/in/zxtomw | github.com/zxTomw

Education

University of Waterloo — Waterloo, ON
3rd Year Bachelor of Computer Science, Honours — *Sept. 2022 – May 2027 (Anticipated)*
- GPA: 3.72, Excellent Standing

Technical Skills

Languages: C/C++, TypeScript/JavaScript, Python, Java, SQL, Swift
Frameworks: NextJS, React, Node.js, Express, Flask, Django
Developer Tools: Git, Docker, CMake, Google Cloud, AWS, MS Azure, Foxglove, Turborepo, GitHub Actions
Technologies: YOLO, PostgreSQL, MongoDB, ROS, WebSocket, Supabase, Pandas, NumPy, TensorFlow

Experience

Software Engineering Research Assistant — Jan 2024 – Dec 2024
Tang Nanotechnology Lab, Faculty of Science, University of Waterloo — *Waterloo, ON*
- Developed a full-stack web app for a remote healthcare system for patient data monitoring and remote diagnosing
- Contributed to the research paper by collecting data and analysis on the application usage and clinician reviews
- Implemented real-time sensor data streaming to the web interface using **Websocket** and **Azure** IoT Hub
- Reduced streaming delay of real-time data by 15% by avoiding copying of dataset on the client side
- Utilized **NextJS** and **TypeScript** to optimize performance and user experience with server-side rendering
- Leveraged Azure Functions and CosmosDB to build and maintain a scalable **serverless** backend
- **Visualized** real-time sensor data into interactive graph components using **ChartJS** and **React**
- Built CI/CD pipelines with **GitHub Actions** and **TurboRepo** for streamlined deployment and testing

Projects

NavGoose | *Python, OpenCV, YOLO, PyTorch, Raspberry Pi, Websocket, OpenAI* — GeeseHacks 2025
- Built an internet-based hardware that helps visual-impaired individuals to avoid hazards while walking
- Trained an object detection network using **YOLOv11** models to identify 100+ types of possible hazards
- Implemented video streaming from the **Raspberry Pi** to the web server with **Websocket** and **Flask**
- Utilized **OpenAI** and **LangChain** SDK to generate and tune voice alerts for detected hazards and locations

Lidar Pathfinding Robot | *C++, CMake, Docker, ROS, Foxglove*
- Implemented message-based communications between node objects using the **ROS** Library and observer pattern
- Developed the cost map module to convert raw lidar inputs into coordinate-based values to be stored in local maps
- Leveraged **ROS tf2** for coordinate transformations to align and aggregate local maps with the global map
- Utilized the **A*** algorithm to generate the shortest paths and avoid collisions
- Reduced the computation for updating the global map by 60% by optimizing the algorithm for aggregating maps

PersonalNotes | *TypeScript, NextJS, OpenAI, Flask, Google Cloud, Postgre SQL* — Hack the North 2024
- Designed an AI learning system to aggregate and summarize documents and notes
- Lead frontend development and backend database integrations
- Employed **ShadCN** and NextJS app router to build and manage the login pages, file uploads, and dashboards
- Configured CI/CD pipelines for automated deployment to **Vercel** and **Google Cloud**
- Leveraged OpenAI **vector embeddings** to construct knowledge bases from uploaded materials
- Built and managed a database using **PostgreSQL** and Flask for user authentication and file indexing

CLI Chess | *C++, GNU, CMake*
- Implemented a command line application following the **model-view-controller** pattern
- Constructed chess classes by adopting the **RAII** paradigm, **OOP** principles (inheritances, polymorphism), and various design patterns
- Designed the algorithms of the computer players of different levels for move prediction with **alpha-beta pruning**
- **Optimized** the AI players and reduced processing time by 45% by refactoring the move ranking algorithms

Lost and Found App | *JavaScript, ExpressJS, MongoDB, NodeJS, AWS*
- Developed **RESTful APIs** for the backend server using the Express framework, supporting database operations
- Utilized **MongoDB** and **ExpressJS** to manage user and item data, and defined their shapes using schema
- Designed referencing schema between the user and item clusters to allow fast retrieval of independent data