ALYZEN JERAJ

✉ alyzenjeraj@gmail.com ☎ 587-432-8665 in in/alyzenjeraj alyzenjeraj

EDUCATION

University of Waterloo | *Bachelor of Applied Science, Honours in Mechatronics Engineering* **2021-2026 Expected**

TECHNICAL SKILLS

Languages: C++, Python, C, CUDA, XML (URDF/xacro), JavaScript, TypeScript

Embedded: ARM, STM32, TI, Raspberry Pi, Arduino, NVIDIA Jetson, CAN, SPI, UART, XCP, UDS, PWM, I2C

Developer Tools: Linux, Git, ROS2, Rviz2, Gazebo, Foxglove, GDB, FreeRTOS, OpenCV, Docker

WORK EXPERIENCE

Robotics Software Engineering Intern | Ocado Intelligent Automation **May 2024 - Present**

- Developed a teleoperation framework using **ROS2** and **Docker** to allow virtual reality (VR) controllers to operate a **UR10e** robot arm and a multi-modal end effector for data collection to train imitation models
- Integrated **cartesian pose estimation** and **cartesian admittance/impedance controllers** into teleoperation movement stack, serializing data over **gRPC** with **Protobufs** to control arm with ROS2 in **Python** and **C++**
- Transformed twists from the camera frame to the robot base frame, allowing complete remote teleoperation
- Visualized robot movement in **Rviz2** and analyzed recorded movements in **Foxglove** for validation
- Implemented **Linux kernel patches** to enable alternative VR headset connectivity for teleoperation system
- Developed a data retrieval, processing, and analysis pipeline in **Python**, enabling engineers to extract and visualize **ROS** topic data from millions of **MCAPs** from robots deployed at scale to view sensor data trends

Firmware Engineering Intern | Tesla **Sept 2023 – Dec 2023**

- Configured an **MPPT driver** to utilize **high-resolution pulse width modulation (HRPWM)** written in **C** on a Texas Instruments Sitara chip, **increasing resolution by 98%** and reducing electrical noise within the system
- Validated ePWM peripheral changes on bench with an **oscilloscope**, checked against schematics on **Altium**, verified registers on **TI CCS** and further confirmed signals with a **Typhoon HIL** scope and schematic editor
- Built and flashed firmware using the **UDS** protocol on a **CAN bus**, observing changes on **PCAN-Explorer 5** and debugged firmware using **JTAG** and **GDB** to set breakpoints, see the backtrace, and check registers
- Developed and optimized a data buffering library, **increasing packing efficiency by 300%** through type changes, scaling factors, and bitwise manipulation for efficient value storage in **C** while retaining precision
- Engineered and coded innovative solutions to seamlessly migrate and update features from retiring chips, including feature setting algorithms and adding **XCP** to allow buffer logging, and updated bootloaders
- Created a pipeline to generate **C firmware code** directly from **Simulink** electrical schematic diagrams using **Python**, eliminating development time and removing human translation errors with **lexical analysis**

Autonomous Robotics Software Engineering Intern | Martinrea Alfield **Jan 2023 – Apr 2023**

- Developed an algorithm with **visual odometry** to map moving objects and predict their trajectories in 3D space for an autonomous mobile robot using **C++**, **CUDA**, **OpenCV**, **NVIDIA's VPI**, and **PCL**, relying on only camera systems in a high-activity factory environment to ensure user safety when the robot is operating
- Integrated with the **ORB-SLAM3** mapping algorithm to localize the vehicle in a memorized map, dynamically masking moving objects, and to control the vehicle to respond to predicted movements of local objects
- Engineered image processing techniques for computer vision using **CUDA** on **NVIDIA Jetson Orin** hardware to increase analysis speed using **parallel computing architecture**
- Worked with the **Clang Abstract Syntax Tree** and **LLVM** in C++ to analyze source code and automatically generate equivalent code in **Python** and **Java** to integrate with the autonomous robot

TECHNICAL PROJECTS

F1TENTH Autonomous Race Car | *Robotic Software Development, ROS2, SLAM, RRT* *(In Progress)*

- Programming a 1/10th scale simulation race car with planning and control algorithms to competitively race
- Implemented **Rapidly-exploring Random Tree (RRT)** and **Model Predictive Control (MPC)** for autonomous navigation using **ROS2**, **Python**, and **C++**, testing algorithms on an **Rviz** simulation track with **LiDAR** data

Custom RTOS | *RTOS, ARM, OS Development, C, Firmware*

- Developed a custom **RTOS** from scratch in **C** on an **ARM-based** STM32 microcontroller, implemented a round robin scheduler, interrupt service routines (ISRs), thread control blocks (TCBs), and timing management
- Implemented concurrent multithreading on a single core CPU, saving register values for context switching

INTERESTS

Cycling FPV Drones Cooking Hiking Photography

PORTFOLIO

ALYZEN JERAJ

OVERVIEW OF PROJECTS

*All employment related projects have been approved for external showcase

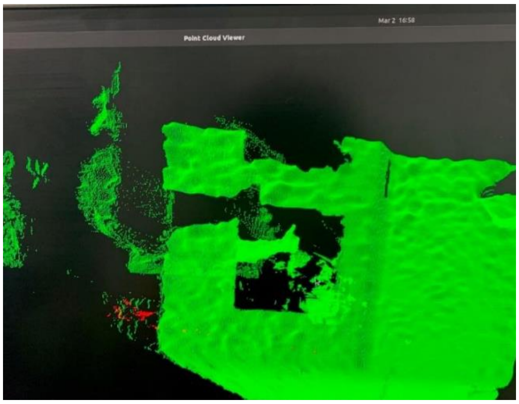
MOVING OBJECT DETECTION & TRAJECTORY PREDICTION

01

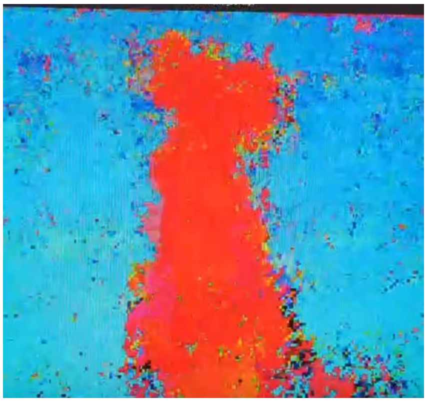
Martinrea Industries

- Designed an algorithm to detect and track moving objects and then predict their future trajectories in 3D space for motion planning using **C++**, ensuring safety in a high traffic factory environment
- Used a **Dense Optical Flow** algorithm and **OpenCV** to find pixel translation between continuous frames, mapping these results to a depth image to get instantaneous 3D positioning and velocity
- Biasing the current heading and speed of the robot against the relative pixel translation allows the algorithm to determine the absolute speeds of everything in its environment using their distances from the camera

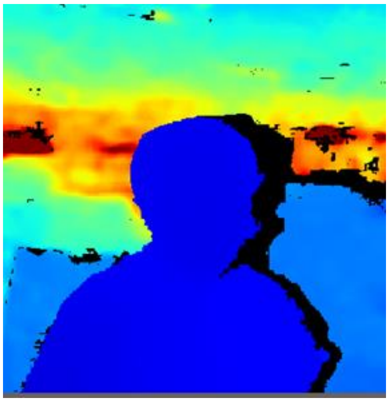
Autonomous Mobile Robot



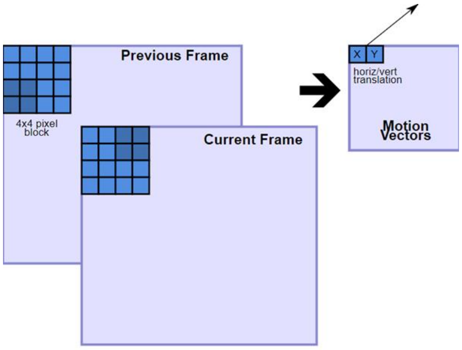
ORB-SLAM3 shown in PCL



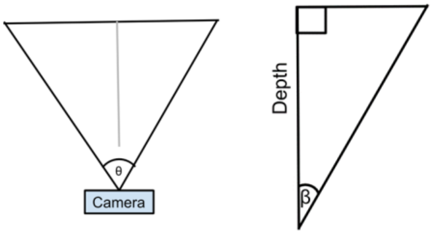
Dense Optical Flow Pixel Translations



Intel RealSense Depth Camera



Dense Optical Flow Transform



Trigonometry for Depth

- Parallelized computation using **CUDA** to allow pixel analysis to run simultaneously on the **GPU** of an **NVIDIA Jetson Orin** for 6 camera streams
- Integrated the algorithm with the **Point Cloud Library (PCL)** to prevent moving objects being saved in the **ORB-SLAM3** map, and used **CMAKE** build tool
- Since the DOF algorithm returns the translation of 4x4 pixel blocks, this data needs to be re-mapped to the colour and depth images respectively
- Using the camera's field of view (FOV), and the depth value of each pixel, the physical translation distance of each pixel could be determined across the span of the frame

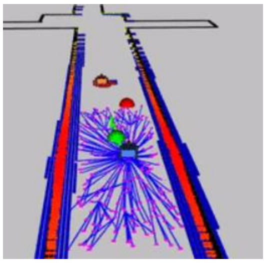
F1TENTH Autonomous Race Car

02

Lab Oriented Course Offered by UPenn

- Designed and implemented advanced path planning and control algorithms such as Rapidly-exploring Random Tree (**RRT**), Model Predictive Control (**MPC**), and Proportional-Integral-Derivative (**PID**) loops for autonomous navigation in simulated environments, ensuring optimal performance and safety during high-speed maneuvers, pulling data from a simulated LiDAR
- Configured and utilized **ROS2** and **Gazebo** for developing and testing algorithms in a realistic, high-fidelity simulation environment. Employed tools like **RViz2** for visualization and debugging within the simulation
- Development and studying still in progress – Future plans to apply **Reinforcement Learning** to control the car

F1TENTH Vehicle

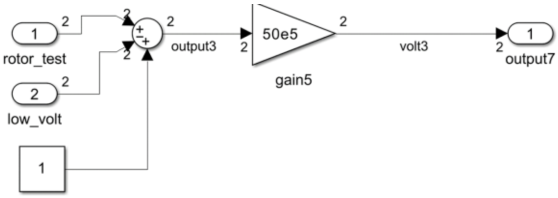


RRT Path Planning

03

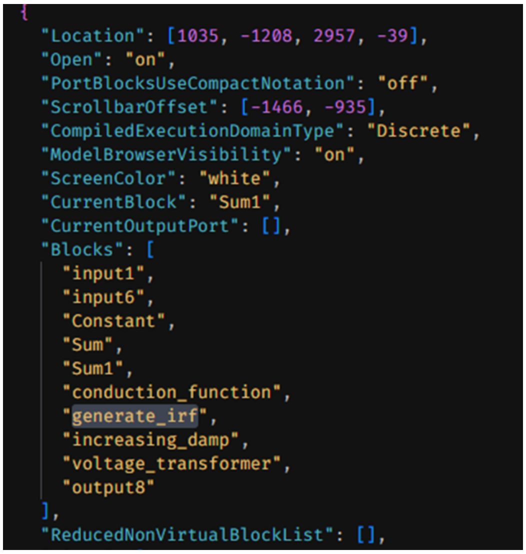
SIMULINK TO C FIRMWARE CODE GENERATION

Tesla



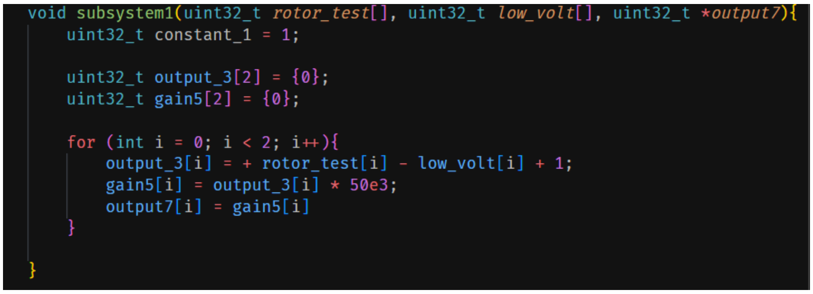
Sample Simulink System

- Created a pipeline for engineers to generate source **C firmware code** directly from their **Simulink** models using **Python**
- Parsed the **JSON data** of the model to tokenize values, variables, and functions, and combine them into functions, expressions, and statements using **lexical analysis**
- Implemented a **linked tree structure** in Python to map the direction of control and signal flow, allowing for predictable code generation and diff preservation for git commits



```
{
  "Location": [1035, -1208, 2957, -39],
  "Open": "on",
  "PortBlocksUseCompactNotation": "off",
  "ScrollbarOffset": [-1466, -935],
  "CompiledExecutionDomainType": "Discrete",
  "ModelBrowserVisibility": "on",
  "ScreenColor": "white",
  "CurrentBlock": "Sum1",
  "CurrentOutputPort": [],
  "Blocks": [
    "input1",
    "input6",
    "Constant",
    "Sum",
    "Sum1",
    "conduction_function",
    "generate_irf",
    "increasing_damp",
    "voltage_transformer",
    "output8"
  ],
  "ReducedNonVirtualBlockList": [],
```

Snippet of JSON Simulink File



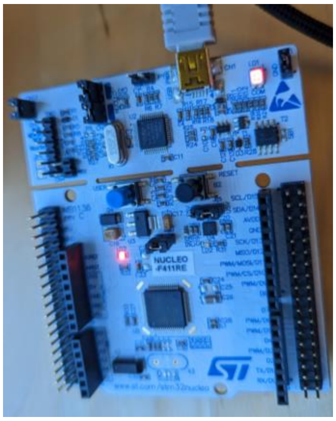
```
void subsystem1(uint32_t rotor_test[], uint32_t low_volt[], uint32_t *output7){
    uint32_t constant_1 = 1;

    uint32_t output_3[2] = {0};
    uint32_t gain5[2] = {0};

    for (int i = 0; i < 2; i++){
        output_3[i] = + rotor_test[i] - low_volt[i] + 1;
        gain5[i] = output_3[i] * 50e3;
        output7[i] = gain5[i]
    }

}
```

Auto Generated Code

CUSTOM REAL TIME OPERATING SYSTEM (RTOS) 04

Built for use with an auto planter

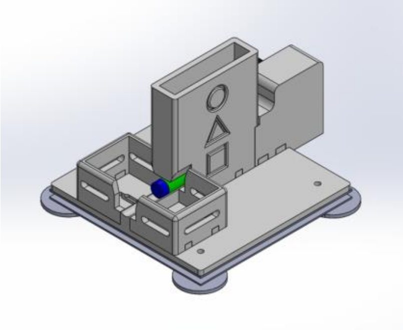
- Developed a **RTOS** from scratch on an **ARM Cortex M4** chip for use with a hydroponic garden in **C**
- Implemented concurrent multithreading on a single core CPU through use of **context switching**, thread control blocks, interrupt service routines, and timing management
- Programmed custom memory stack allocation for new threads as they are spun off, accounting for proper allocation and deallocation to reduce possibilities of memory leaks



Flashing an STM32

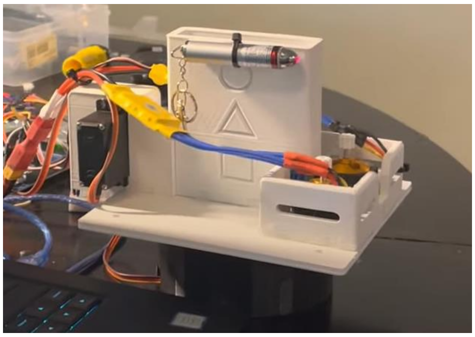
05 GAME OF SQUID TARGETING SYSTEM AND SHOOTER

Hack the North

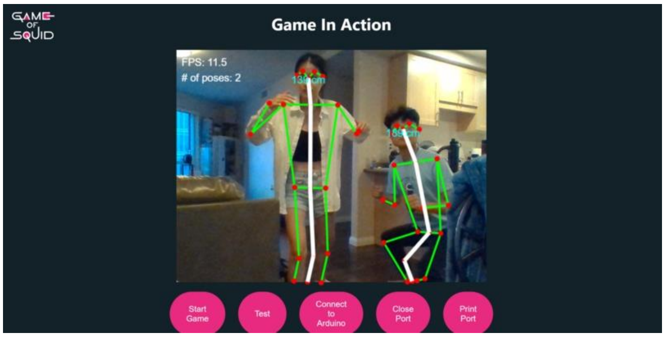


Shooter Mechanism CAD

- For the Hack the North Hackathon, my team wanted to make something from popular media into a reality. Enter our version of the “Red Light, Green Light” game from Squid Game!
- Captured user’s images and processed them through **TensorFlow’s MoveNet** Machine Learning pose tracking framework using **TypeScript** and **React.JS**
- Developed a regression analysis to estimate user distances and angles from the camera with **95% accuracy** and communicated with an **Arduino** using the **Web Serial API** for targeting purposes
- Programmed Arduino to respond to serial input and actuating the servo with the **PCA9685 PWM Driver**



Fully Assembled Shooter



Game In Action

FPS: 11.5

# of poses: 2

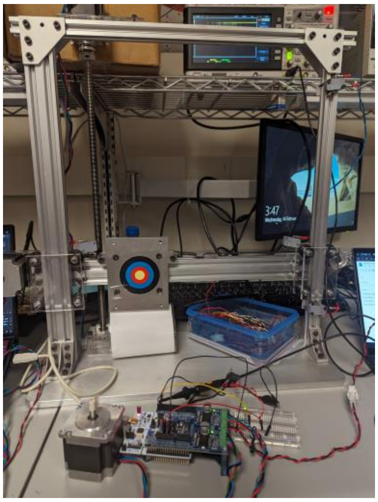
Game Interface

TWO AXIS TARGETING PLOTTER

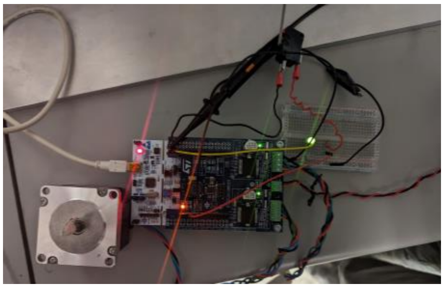
06

Firmware for positioning and motor control

- Developing **firmware** in **C** for a two-axis machine to move to specific co-ordinates in succession like a plotter
- Built using an STM32 Nucleo board, with an ARM Cortex M4 and an L6470 motor driver
- Programmed software interrupts when limit switches are triggered to halt and reverse motors for proper bounding
- Currently in Progress



Two Axis Machine Hardware



Motor Control with Limit Switches

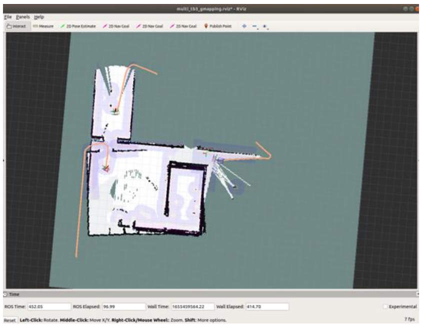
MULTI AGENT ROBOTICS RESEARCH

07

Undergraduate Researcher, Autonomous Vehicles Research and Intelligence Lab (AVRIL)

LIMO Robot
Research Vehicle

- Involved in research involving multi-agent robotics collaboration and path planning, using **ROS** on Linux and **C++** to develop algorithms
- Trialing multi agent **LiDAR SLAM** mapping, using **GMapping** and **RViz**, working on enabling co-operation between **10 LIMO** robots
- Currently exploring methods for simultaneous path-planning and obstacle avoidance, with plans for future for adaptive task allocation and co-operative mission planning based on changing environmental factors
- Still in early stages of research and development of **Multi Agent Reinforcement Learning**



ROS Time: 452.05 ROS Elapsed: 96.99 Wall Time: 1655459564.22 Wall Elapsed: 414.70 □ Experimental

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click/Mouse Wheel:**: Zoom. **Shift**: More options.

Multi Agent Mapping