






MAHIR MAHOTA

 [linkedin.com/in/mahir-mahota](https://www.linkedin.com/in/mahir-mahota)  github.com/mahir-mahota  mahirmahota.com
 mmahota@uwaterloo.ca  +1 226-899-6987

EDUCATION

University of Waterloo – Candidate for BAsC in Mechatronics Engineering Sept. 2022 – Present
GPA: 4.0 (94.27% Cumulative) – Dean's Honours List and Academic Representative
Relevant courses – MTE 262 (Digital Logic), MTE 140 (Data Structures and Algorithms), MTE 120 (Circuits)

SKILLS

Software: C, C++, Python, Linux, Bash, Git, GDB, CMake, JavaScript, HTML, (S)CSS

Tools and Technologies: STM32, ESP32, Arduino, TensorFlow, OpenCV, Pandas, NumPy, Jupyter Notebook, Matplotlib, Visual Studio, SolidWorks, AutoCAD

Electrical: Altium, Soldering, Oscilloscope, DMM, Reflow and Hot Air Station

Protocols: I2C, SPI, CAN, UART, SMBus/PMBus, USB

EXPERIENCE

Firmware Developer onsemi May. 2023 – Aug. 2023

- Developed firmware in **C/C++** for a multi-phase voltage controller, collaborating in an **Agile** environment
- Implemented a shared **SMBus access layer** in **C** using circular buffers, eliminating global variable reliance
- Multithreaded in **C++** to test driver functionality requiring parallelism, using fundamental **RTOS** concepts
- Verified I2C state machine transitions, using bit masking and bitwise operations to check register values
- Wrote unit tests for **15+ drivers** and the PMBus library with **94% coverage**, validating expected behaviour
- Edited **CMake** and **JSON** files to include tests so they could be built and then output results successfully

Embedded Flight Software Sub-Team Member Sept. 2022 – Present
Waterloo Aerial Robotics Group

- Programmed a motor tester in **C** with an **STM32** Nucleo-F401RE for driving a servo at a range of speeds defined by a potentiometer value received through an **ADC**
- Configured the onboard timer for PWM and implemented **SPI** communication with the peripherals
- Generated a **PWM** duty cycle of 5-10% according to the digital data received using the STM32 **HAL**

PROJECTS

Eye Controlled Trolley

- Programmed an **ESP32** to drive 4 DC motors using L298N H-bridges, enabling speed control over Wi-Fi
- Interacted with **AdHawk** eye tracking glasses and their **API** in **Python** to detect blinks and monitor gaze coordinates with **80% accuracy**, allowing the distance between points around the user to be calculated
- Processed MPU6050 **gyroscope** data to drive trolley with commands received from socket interfacing

OpenCV Robotic Arm

- Implemented **C++** drivers for **UART** communication with the 6DOF arm through an HC-05 BT module
- Interfaced between **Python** program and **Arduino** using serial transmission to allow sending characters
- Detected hand gestures with 21 landmarks generated using **OpenCV** and Google's MediaPipe library

Movie Reviews Discord Bot

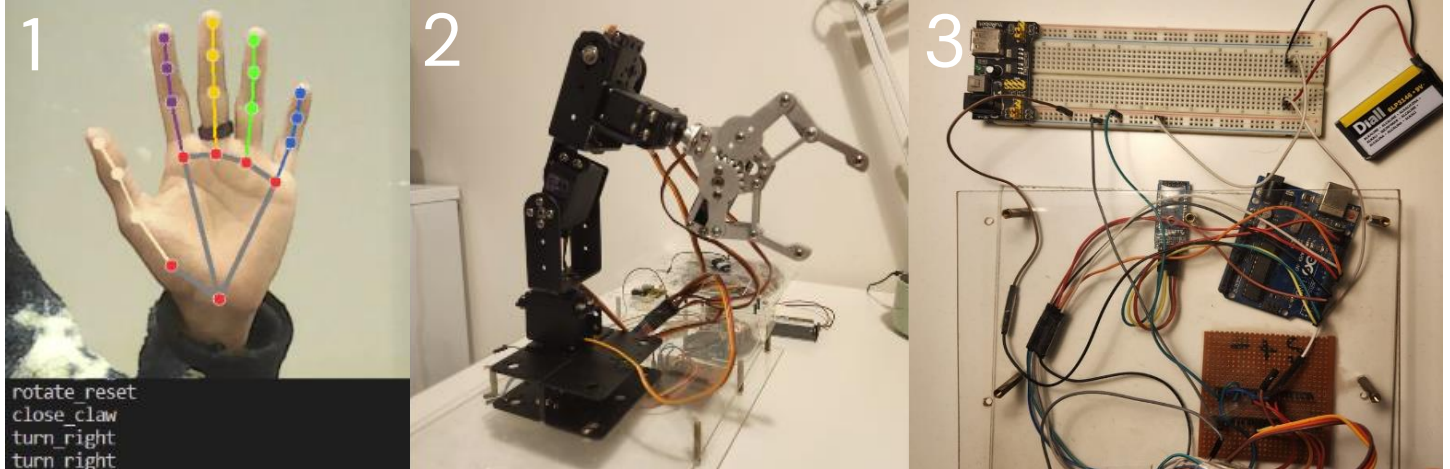
- Designed, trained, and tested a neural network for **NLP** with **TensorFlow**, using text vectorisation to do sentiment analysis of movie reviews with **98% accuracy**
- Created an interactive bot in **Python** using the **Discord API**, taking in a movie name and passing the content of web-scraped articles into the ML model to decide whether reviews are positive

Guitar Playing Robot

- Developed functions in **C** to take input from a colour and ultrasonic sensors for tracking task completion
- Processed output from onboard **motor encoders** to play songs with up to five frets with **91% accuracy**

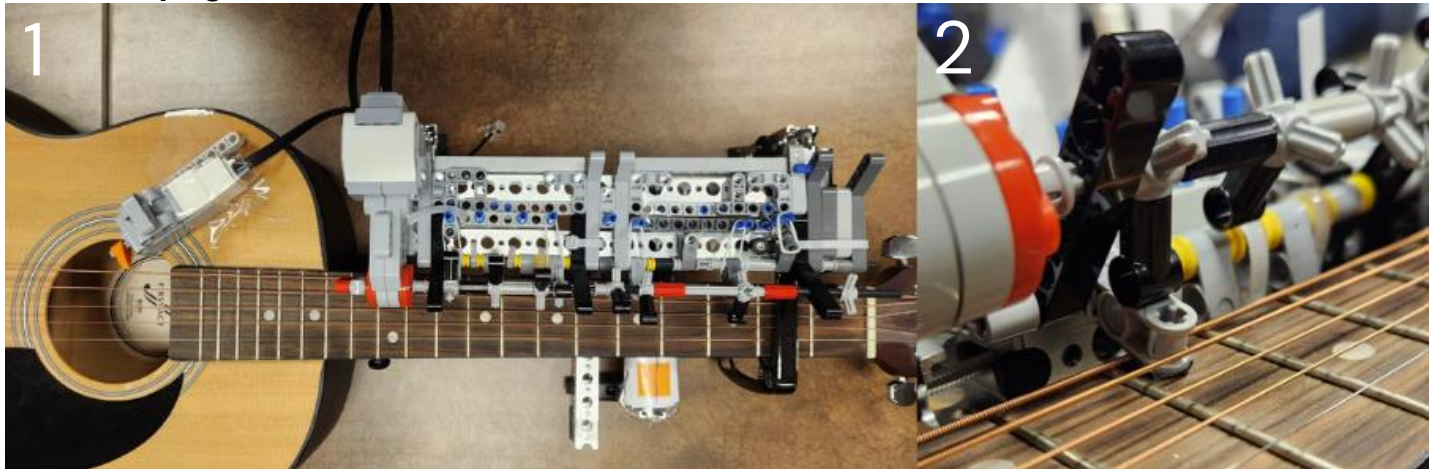
PORTFOLIO

OpenCV Robotic Arm



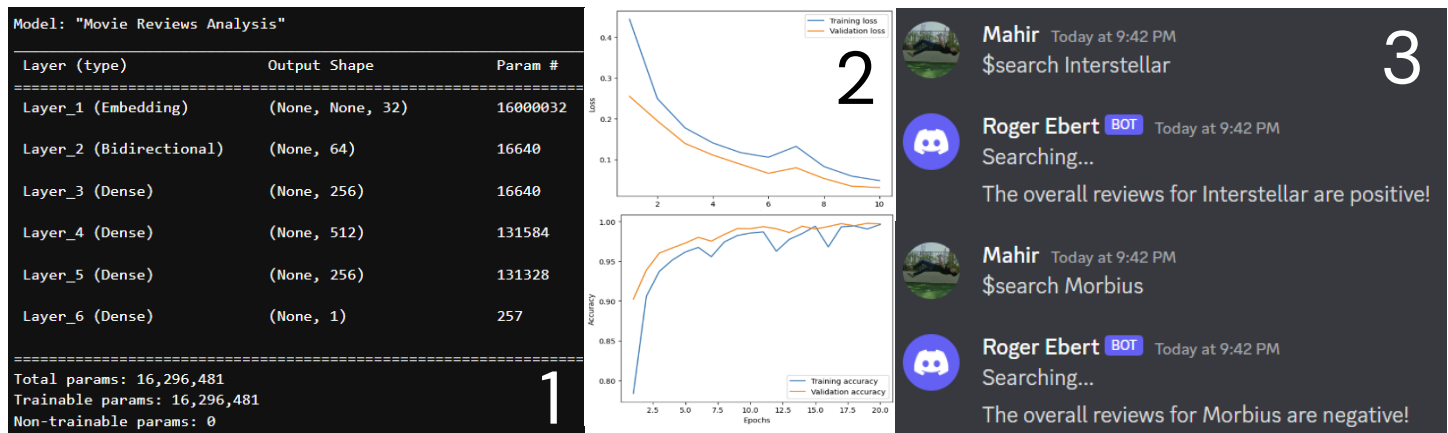
1. Transferred the hand landmark co-ordinates into an array and used those to detect when finger or hand positioning changed. Used these changes to define commands (printed in the terminal) that could be sent to the Arduino through the Bluetooth module, affecting the position of the servo motors.
2. Built the robotic arm with six degrees of freedom. Cut the acrylic base using a vertical bandsaw and drilled holes to insert screws around the border.
3. Wired the six servo motors to two external power supplies on a breadboard to properly support their stall current. Soldered the ground wires of the motors and the power supplies together with the ground on the Arduino. Connected the servo control wires and the Bluetooth module to ports on the Arduino, enabling serial communication with the Python program.

Guitar Playing Robot



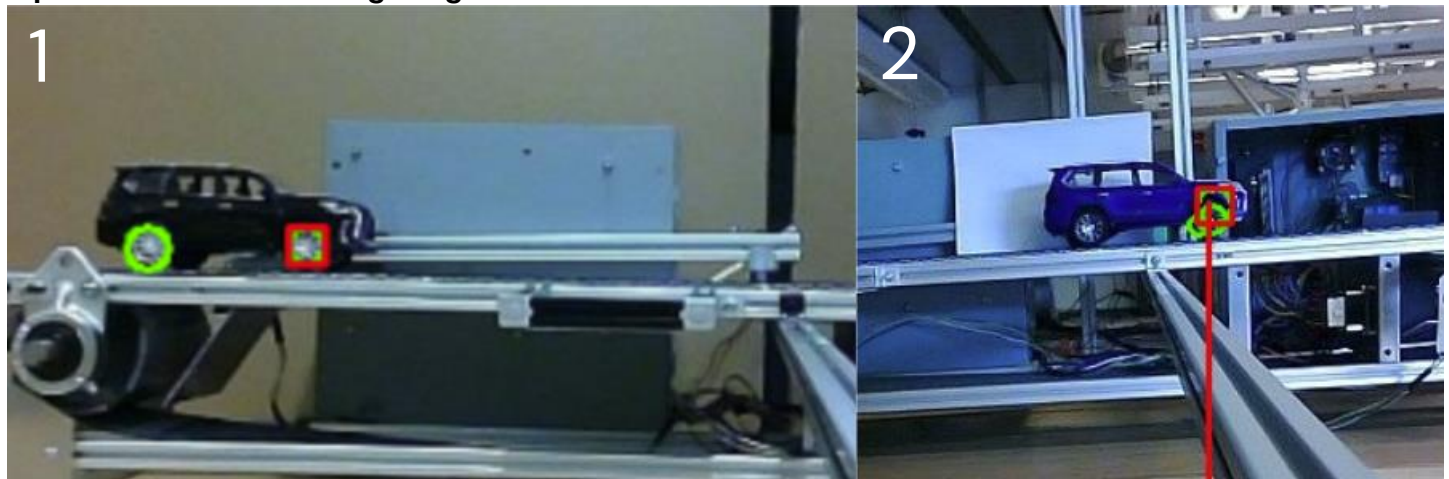
1. Designed mechanisms for colour reading, fretting, and strumming to mount onto the guitar. Used colour and ultrasonic sensors to detect the progress of tasks. Collaborated with three peers to create the robot as our first term final project.
2. Built a camshaft system that can press on the string at different frets depending on the angle a motor is rotated. The current position of the motor was tracked using an onboard motor encoder.
3. Programmed (in RobotC) and created flowcharts (in Microsoft Visio) for nine functions that ran the mechanisms and ensured that they were all in sync to play the songs properly.

Movie Reviews Discord Bot



1. Created an NLP model with TensorFlow using a dataset of 50K IMDb reviews. Pre-processed the data by vectorising it and creating an input pipeline, separating training, validation, and testing sets. Installed CUDA and cuDNN to enable the model to train using the computer GPU. Saved the trained neural network to be reused in the Python program for the bot.
2. Adjusted the neural network to account for any unexpected trends in the data after reviewing loss and accuracy graphs at various stages of training.
3. Interacted with the Discord API to get access to user messages, letting the bot communicate in real time. Used asynchronous programming to allow the bot to recognise commands while processing previous tasks in the background.
4. Programmed a web-scraping script using Python libraries that collects URLs from a Google query search and then extracts text from the main body of the websites. This enables the bot to search for and pass the content of reviews through the model.

OpenCV Wheel Tracking Program



1. Participated in the Toyota Innovation Challenge Hackathon to program a solution for taking images of cars at the right time on a conveyor belt, for quality analysis. The aim was to use wheel detection instead of the current standard of mechanical triggers, to improve accuracy.
2. Developed a successful program in under 12 hours using OpenCV code snippets, such as Hough transform and contour detection functions, working in a group of four. Both wheels are detected, and the front one is bounded using a red square. When the front wheel passes a certain point (denoted by the vertical red line), an image is taken and saved. The code was tested on a scaled down model of the factory system.
3. Got the depth camera working in Python instead of C++ using WSL, allowing easier detection of stickered holes in the car chassis and greater quality control capabilities.