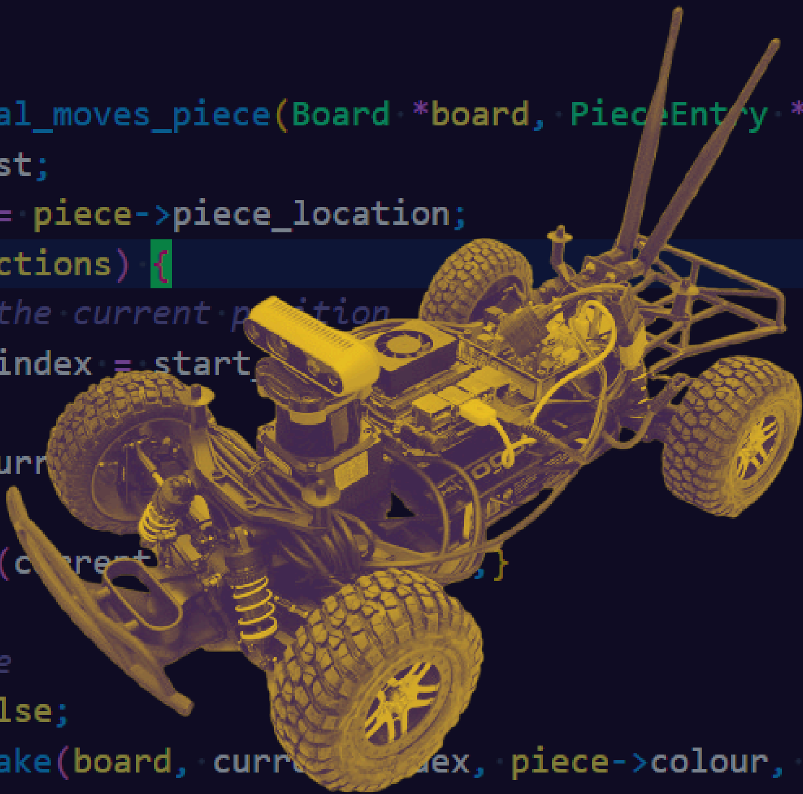
M A R K D O

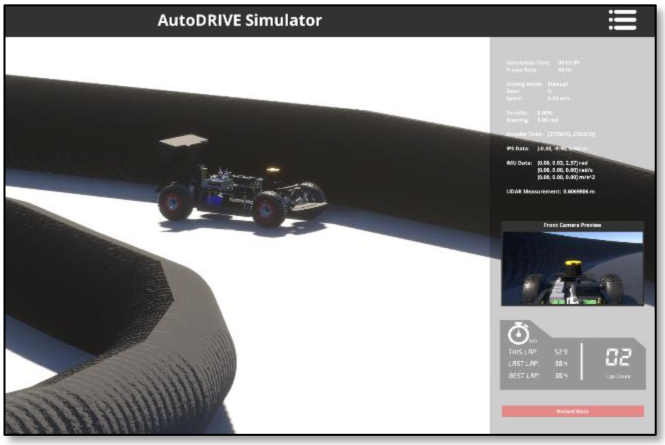


```
bool MoveGenerator::can_move_or_take(Board *board, int index, int colour, bool
    return false;
}

std::vector<Move>
MoveGenerator::get_pseudo_legal_moves_piece(Board *board, PieceEntry *piece, s
    std::   <Mo    ove_list;
    unsig    t     index = piece->piece_location;
    for (          directions) {
                   from the current position
                   rrent_index = start_
        do {
            current_index = curr
                        ounds
                  in_bounds(current
                       or move
            bool capture = false;
            if (can_move_or_take(board, curr    ex, piece->colour, capture)
                // Handle move encoding
                      itset<4> move_code(0b000u);
                      code[2] = capture; // 2nd bit is true for captures
                      _list.emplace_back(start_index, current_index, move_code);
                i     apture
                    break;
                }

                 In-bound              t capture nor move; is blocked by a frien
                b   ;

        } while (long_range)           ange pieces should break out of this lo
    }
    return move_list;
}

const bool MoveGenerator::can_capture(Board *board, unsigned int index, int co
    return board->board[index]->colour != colour;
}

const bool MoveGenerator::is_empty(Board *board, unsigned int index_8x8) {
    /**
     * Returns true if square is empty. Assumes square is in bounds.
```

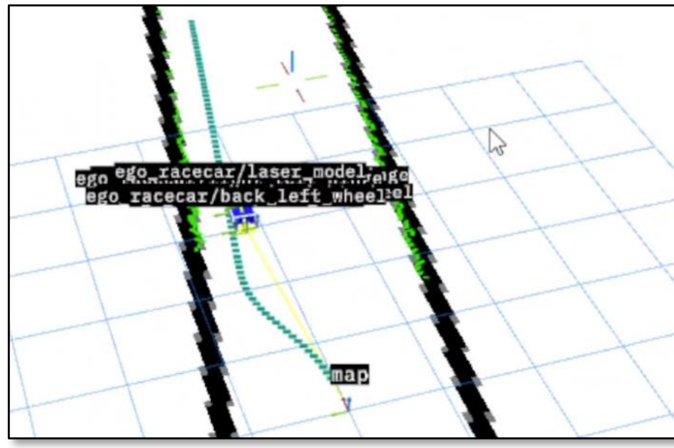
Mark Do - Portfolio

647-766-6794 | mdo@uwaterloo.com | github.com/hepromark | linkedin.com/in/markdouw/

F1Tenth Autonomous Racing Team Lead



F1Tenth Vehicle Simulator



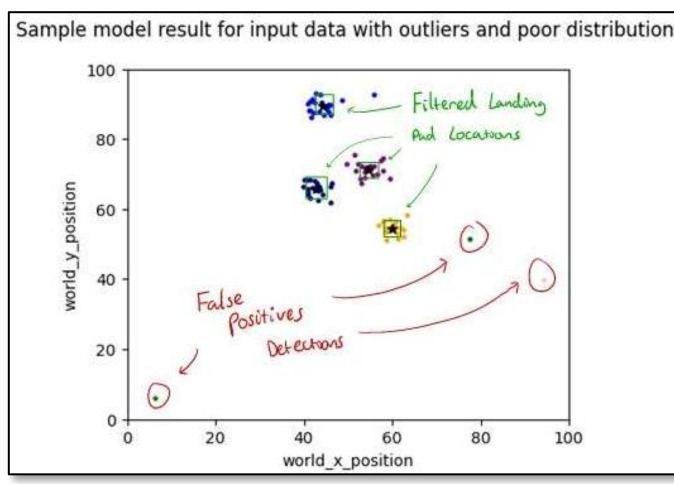
Pure-pursuit control applied to a spline generated by RRT

The F1Tenth competition is a 1/10th scale autonomous racing cup where global teams compete for the fastest autonomy algorithms.

- Leading a team of 15+ undergraduates to develop a self-driving software stack for F1Tenth Autonomous Racing Competition.
- Modelled the onboard software architecture after the See-Process-Act paradigm, prioritizing real-time control & safety.
- Led the design & implementation of pure pursuit control, lattice motion planner, offline & online SLAM algorithms.
- Built a dockerized dev environment integrating ROS2, Foxglove and AutoDRIVE physics simulator.
- Project Repo can be found here.

Landing Pad Clustering Algorithm: In-flight unsupervised learning model

WARG's 2023 Competition Drone: ICARUS



Clustering Algorithm Output: False positives are discarded from potential landing pads.

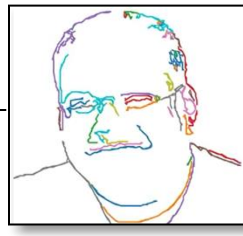
I worked on WARG's in-flight unsupervised learning model for the 2023 AEAC Competition

- Developed a clustering module for an autonomous drone to cluster landing pad detections & predict actual pad locations.
- Implemented a Variational Gaussian mixture model to consume pad locations over time and continuously output estimates.
- Created filters with dynamic thresholding to remove outlier pad locations, enabling drone to operate in poor visibility weather.
- Designed an intelligent memory system to bias towards new observations while retaining knowledge of pads only seen once.
- Code can be found here

Articus Maximus: Sketching Robot

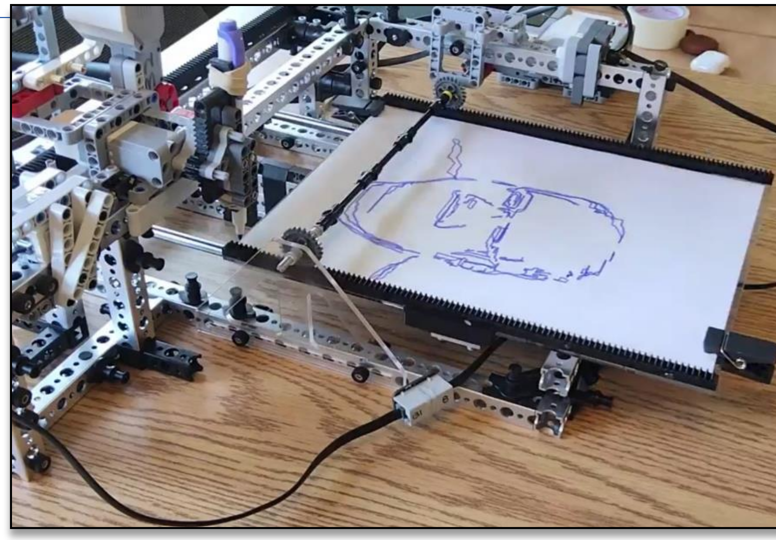
Input Image

Canny edge detection



Contour Detection

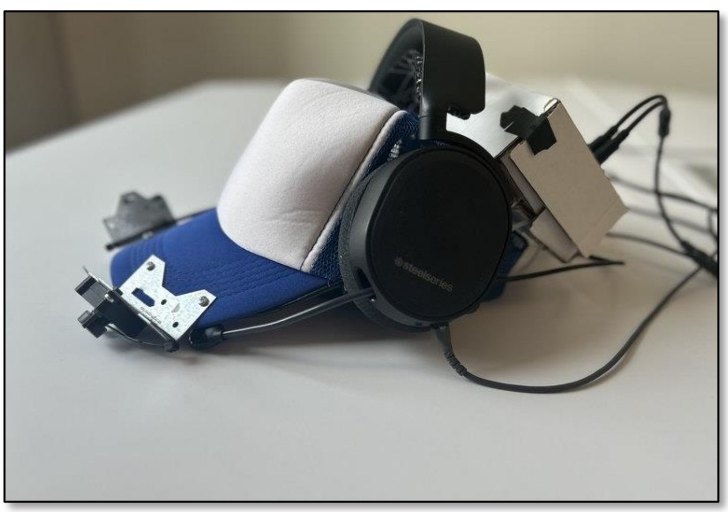
Output Drawing



Sketching Robot after completing a drawing

- Designed & manufactured a 2-axis gantry sketching robot which controls a pen to draw images on paper from digital file input.
- Developed an image pipeline that extracts drawable contours from digital images with edge detection & contour detection.
- Optimized robot drawing speed by using the Douglas Peucker algorithm and Hu Moments to reduce total contour count.
- Programmed firmware PID controller in C with anti-windup & a 1D motion profile to draw lines accurate within 2 degrees.
- Code can be found [here].

BlindWatchers: Assistive Vision Headwear

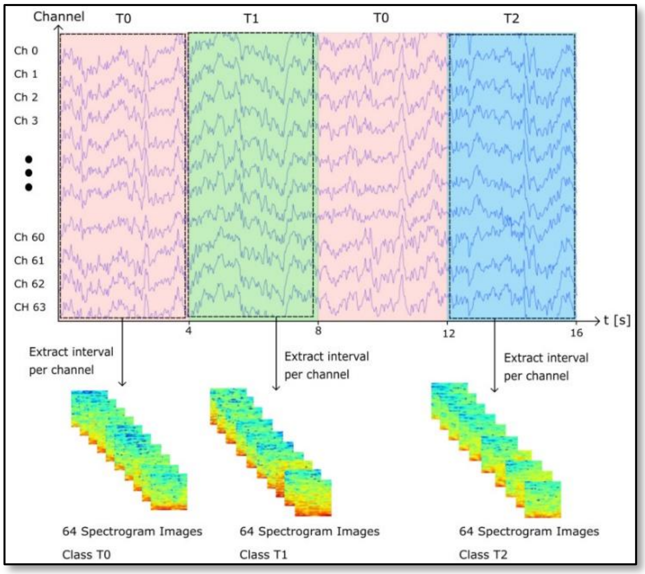


Headwear has 2 cameras, headphones for audio output, with an NVIDIA Jetson attached behind for compute.

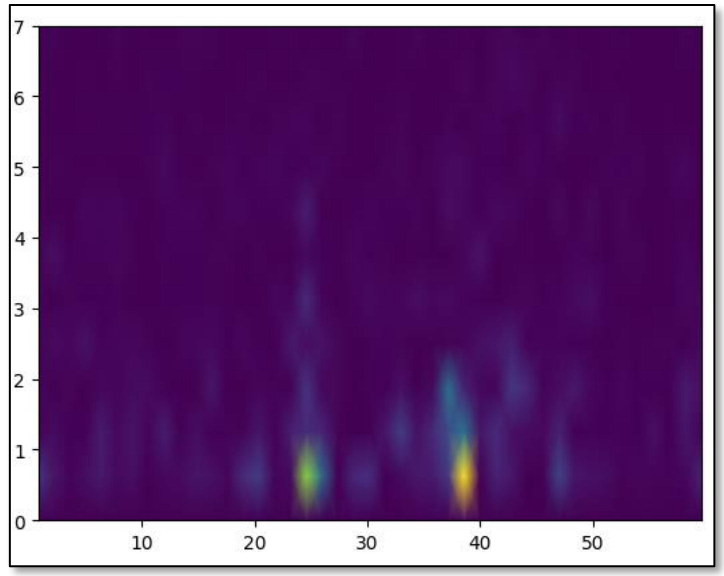
Sample YoloV8 detection bounding boxes, before conversion to spatial audio.

- Designed and developed a smart helmet for the visually impaired with an NVIDIA Jetson, integrating YOLOv8 object detection, directional audio, & Google Cloud Speech-to-Text to provide real-time auditory descriptions of nearby objects.
- Estimated object real-world 3D poses from 2D bounding boxes to provide matching directional audio for more intuitive UX.
- Implemented an asynchronous architecture with a fast-inferencing computer vision model to decrease latency between detection & audio.
- Code can be found [here]

EEG Machine Learning Model: Motor Imagery Classifier



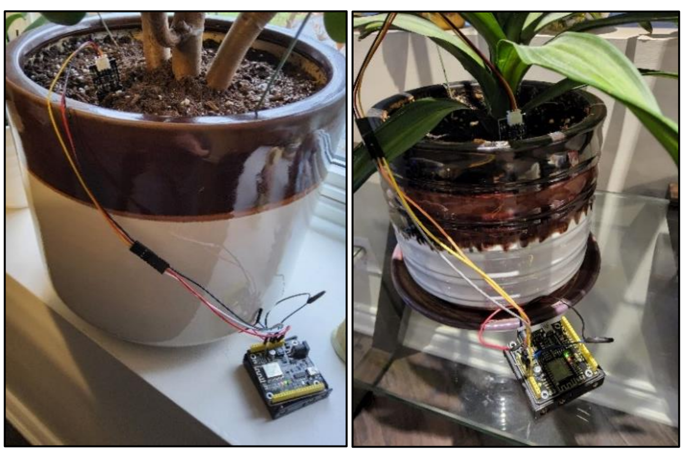
CNN Architecture takes 3D input volumes, created from stacking 64 electrodes' spectrogram images



Spectrogram image produced from a single electrode's 2-minute EEG segment

- Developing deep learning model on EEG brain data for motor imagery classification task on a 64-channel Motor Imagery dataset.
- Converted 64-channel time series data into volumes in the time-frequency domain using FFTs for use with convolutional kernels.
- Wrote custom data samplers in PyTorch with weak shuffling to optimize data read speeds for training batches from .h5 files.
- Project Repo can be found [here]

Soil Humidity IoT: Cloud-based agriculture monitoring.



Examples of soil-humidity detector ESP8266 setup

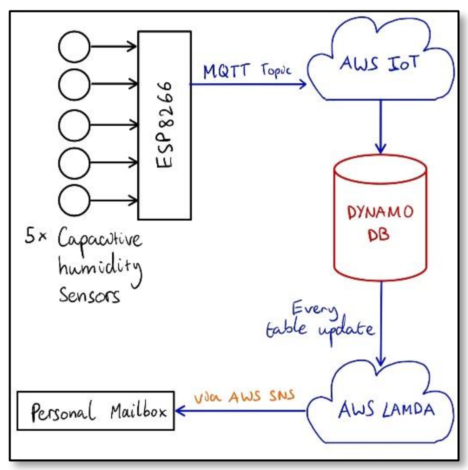


Diagram of the agriculture monitoring system.

- Designed and created a monitoring system to track houseplant soil humidity, alerting users via SMS when below set threshold
- Programmed ESP8266s to publish humidity levels via Wi-Fi to an AWS IoT MQTT topic, storing the data in Dynamo DB.
- Created an AWS Lambda function listening to database writes, to alert users using AWS SNS if readings below threshold.
- Code can be found [here]