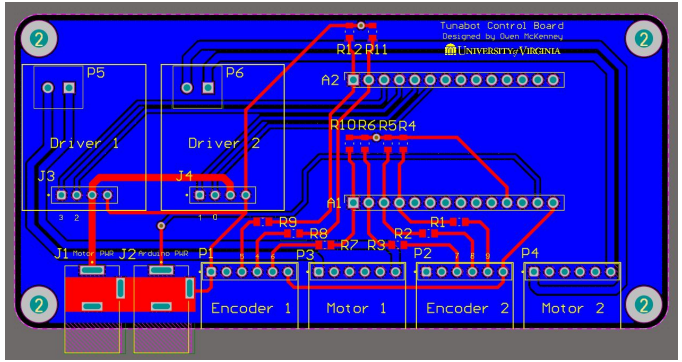
Selected Technical Work Portfolio

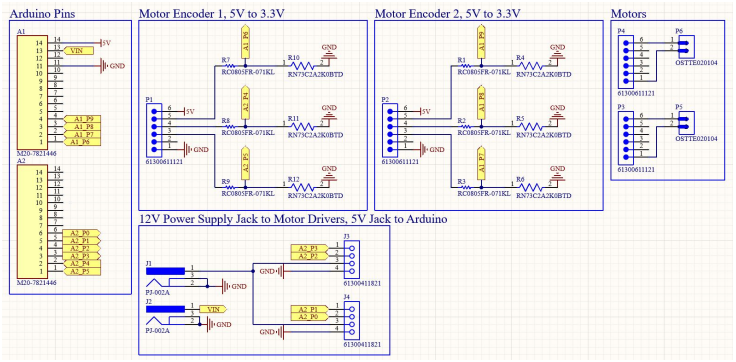
Owen McKenney

Tunabot Control Printed Circuit Board (PCB)

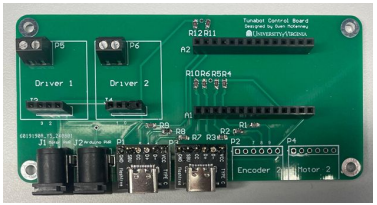
Skills: *Altium Designer, Circuit Design, Soldering, SolidWorks*



Traces Layout

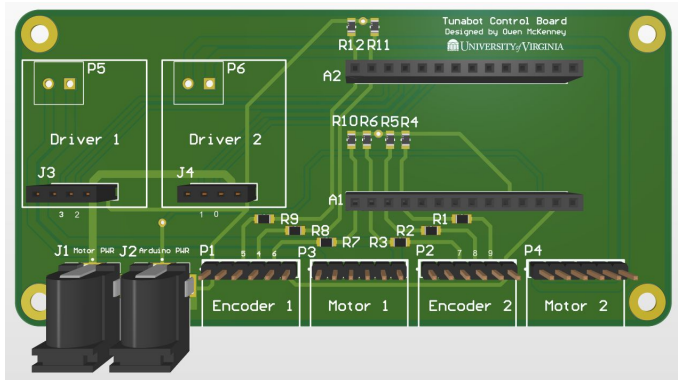


Circuit Schematic



Soldered PCB

PCB Packaging

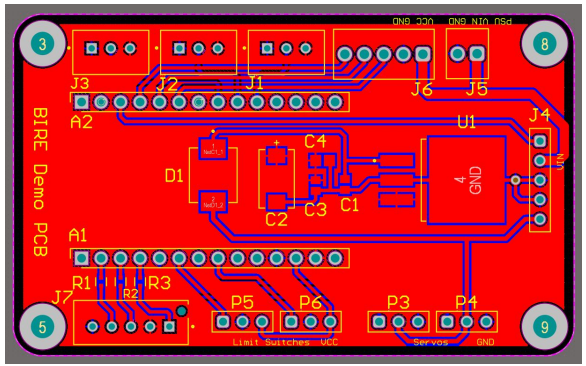


3D Model

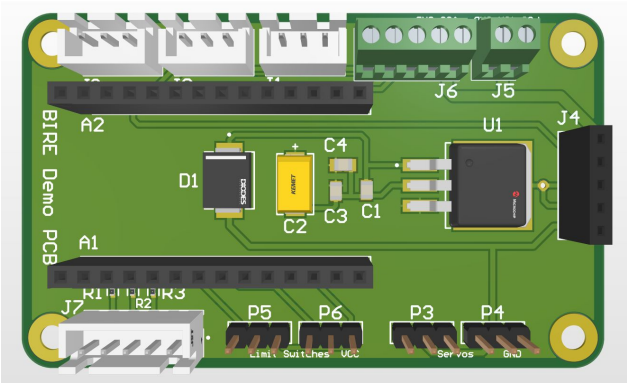
I designed a control board for BIER Lab's Tunabot. The 2-layer **PCB was designed using Altium Designer** and allows for integration with an Arduino MKR Zero, 2 motor drivers, and USB-C cables to allow for external control of up to two Tunabot robots. **I soldered components** and made multiple copies of the PCB to be used by graduate students for use in schooling experiments. I also **designed a 3D printed box** in SolidWorks to package the PCB. I also used this control board in my turning maneuverability study from slide 2.

Hydrokinetic Energy Demonstrator Control PCB

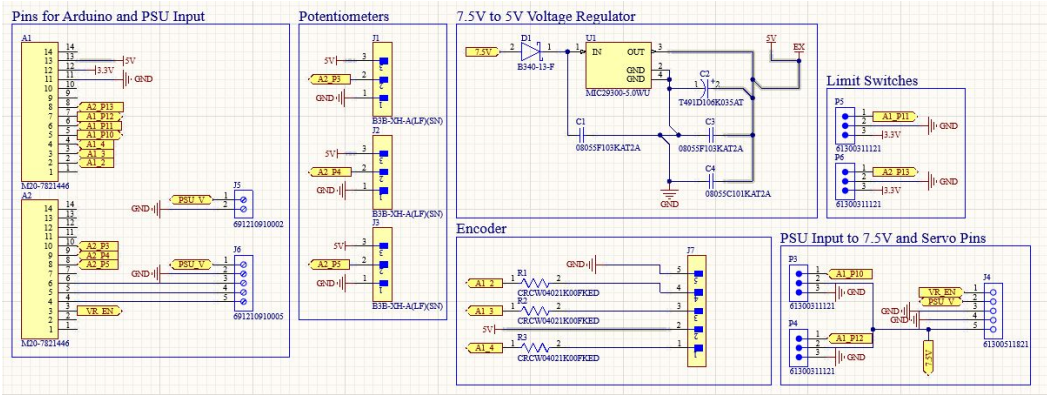
Skills: *Altium Designer, Circuit Design, Soldering*



Traces Layout



3D Model

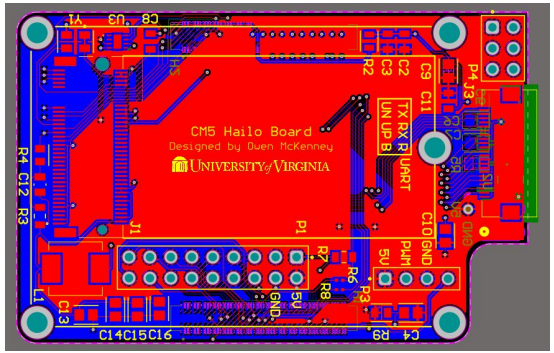


Circuit Schematic

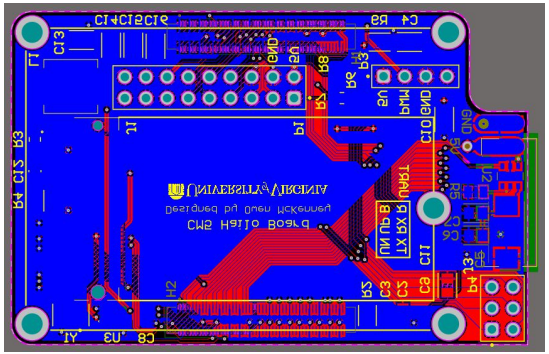
I designed a control board for BIER Lab's Tunabot. The 2-layer **PCB was designed using Altium Designer** and allows for integration with an Arduino MKR Zero, stepper motor, servos, limit switches, and potentiometers. I **soldered the board by hand**. The control board was **used to control the sinusoidal motion of a hydrokinetic riverine energy generator** demonstration system (see slide 11).

Compact Control PCB for Robot Computer Vision

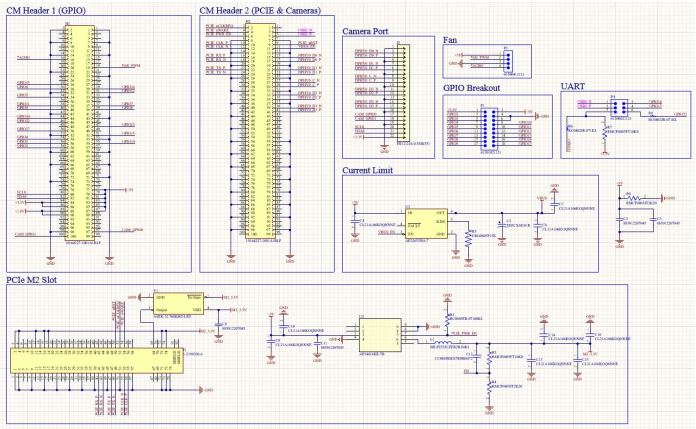
Skills: *Altium Designer, Circuit Design*



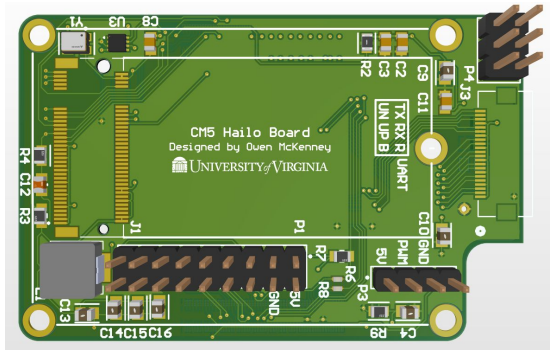
Traces Layout (Top Layer)



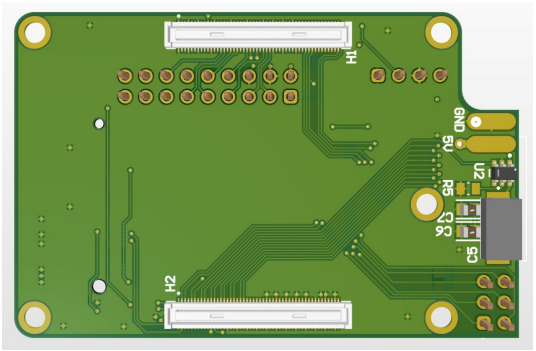
Traces Layout (Bottom Layer)



Circuit Schematic



3D Model (Top View)

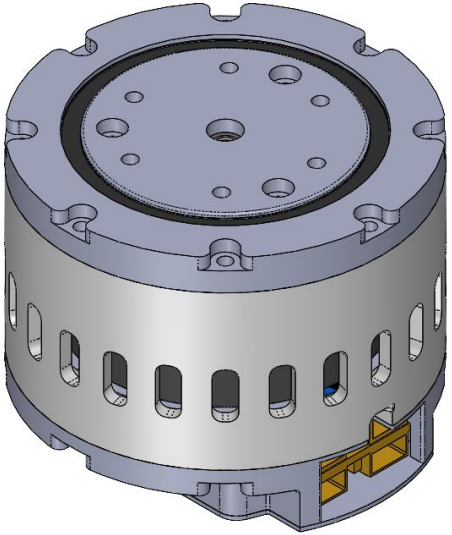


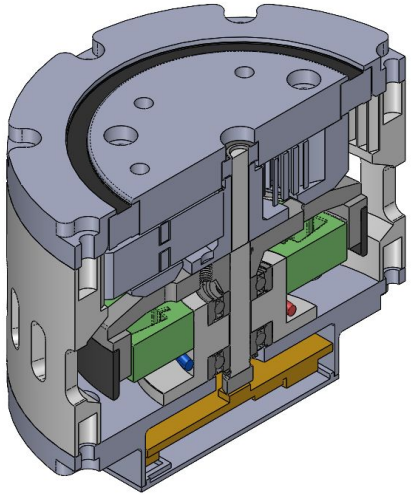
3D Model (Bottom View)

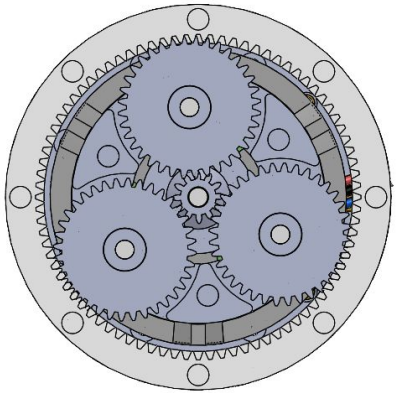
I designed a compact (62mm x 40mm) control board **using Altium Designer** for BIER Lab's Tunabot and Mantabot robots. This PCB interfaces with a RPi5 Compute Module and a Hailo 8L Accelerator Module to enable real-time computer vision inference at 30 FPS. **PCB will be used to enable tracking of other robots underwater, enabling self-organization.** PCB includes breakout pins to interface with onboard cameras and sensors.

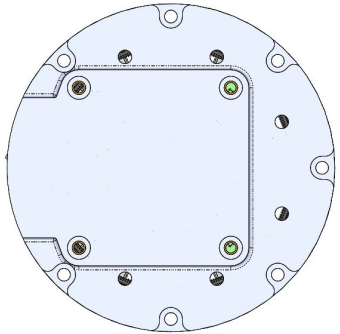
3D Printed Planetary Actuator for Robotics

Skills: *SolidWorks.* See more: https://github.com/owenmckenney/Robotic_Actuator







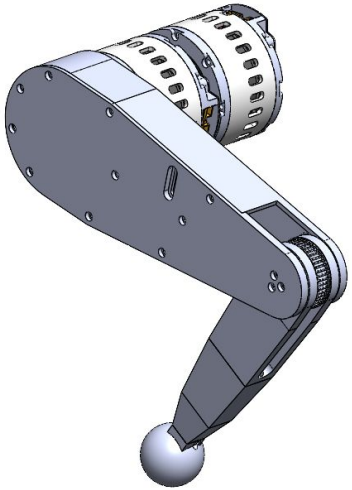


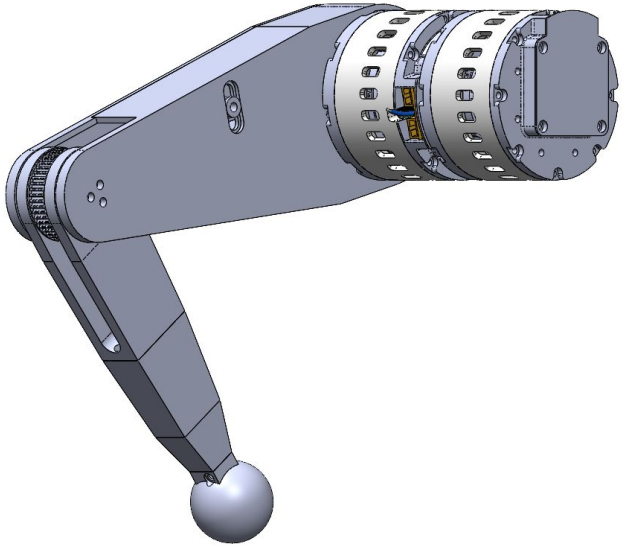
I designed an open-source 3D printed, backdrivable planetary actuator for robotic applications. The CAD assembly has over 15 custom components designed to interface with mjbots motors and encoders, bearings, and other off-the-shelf motion components. The planetary gearbox has a reduction of 8:1 and was developed to be driven by a Raspberry Pi.

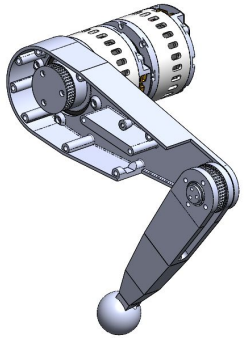
CAD Video: https://www.youtube.com/watch?v=ny0xnwEDxxo

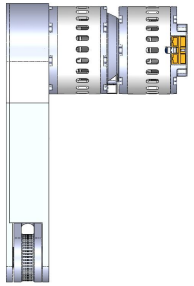
Robotic Quadruped Leg

Skills: *SolidWorks, Python.* See more: https://github.com/owenmckenney/Robotic_Leg







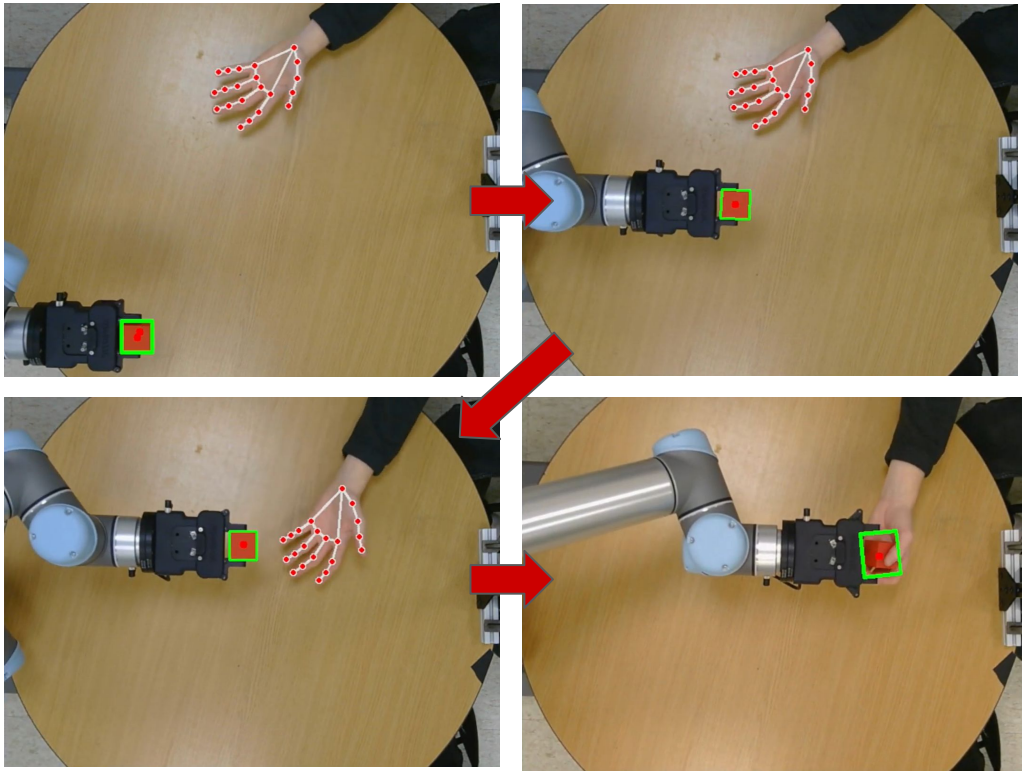


I designed an open-source 3D printed robotic quadruped leg (full extension ~385mm) using the robotic actuators I developed. I wrote Python scripts to calculate the inverse kinematics of the foot and was able to achieve basic gait control.

CAD Video: https://www.youtube.com/watch?v=-uwau5FXpTE

Vision-Based Human-Robot Collaboration

Skills: *Python, UR5 Manipulator, OpenCV, PID.* See more: https://github.com/owenmckenney/Vision_Based_UR5_Interaction



Using a single webcam and a Universal Robots' UR5 manipulator, I wrote python scripts to track an object and a user's hand, allowing for an **object to be delivered to the user's hand** safely. Hand and object coordinates were transformed from camera-frame to robot-frame, and the object was guided to the hand using **PID controllers**. The manipulator was able to respond effectively to a dynamically moving hand (see video). **Over 400 lines of Python.**

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}$$

Coordinate frame transformation matrix
(drastically oversimplified)

$$u = K_p e(t) + K_i \int e(t)dt + K_d \cdot \frac{de}{dt}$$

Proportional-Integral-Derivative (PID) Control

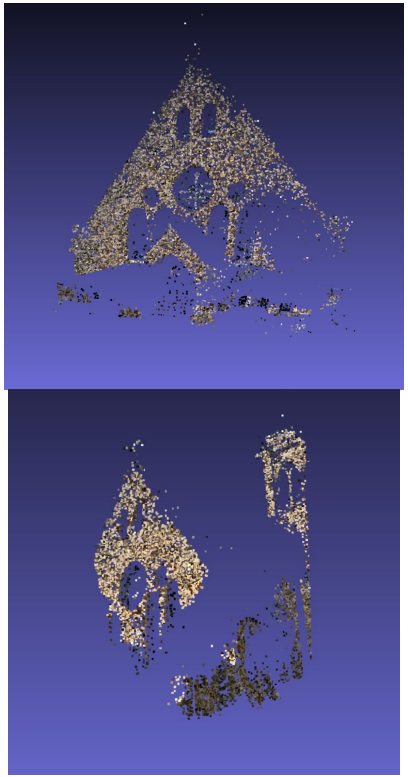
Demonstration Video: https://youtube.com/watch?v=DS-w8ZLICBk

Two-View Structure From Motion 3D Reconstruction

Skills: *Python, CV Algorithms, Point Clouds.* See more: https://github.com/tymisiorek/stereo_reconstruction

Feature matching between images (correspondence)

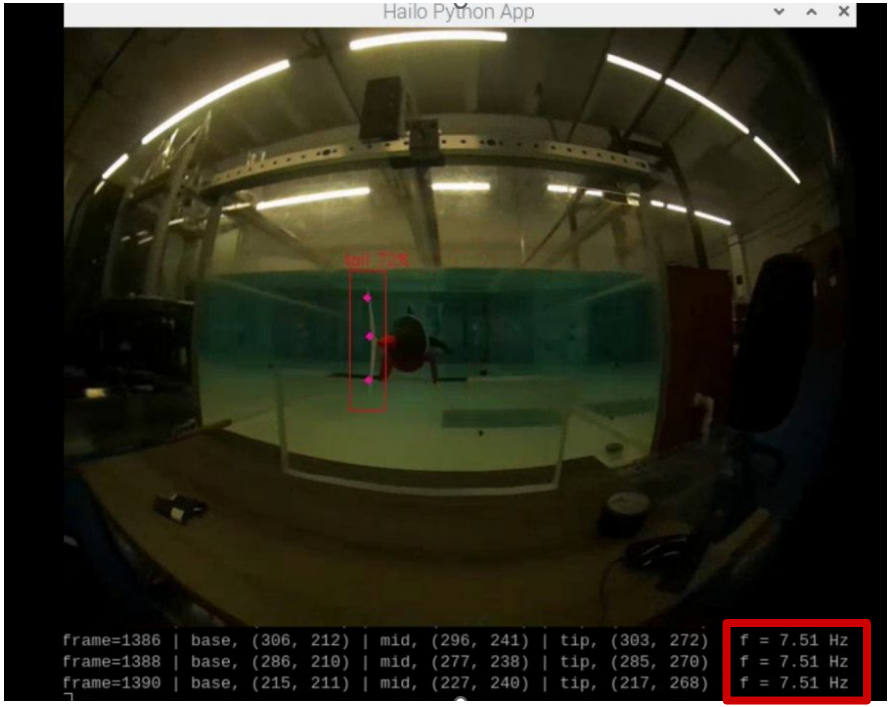
Point Cloud Reconstructions



Using fundamental techniques in computer vision, I worked with a groupmate to write a structure from motion reconstruction pipeline that **generates a 3D point cloud from two images.** Pipeline first matches features between images (SIFT + RANSAC), estimates the relative poses of the images (essential matrix), performs global pose chaining to accumulate pairwise transforms into world-frame camera extrinsics, and finally triangulates matched points to produce a sparse 3D reconstruction. Reconstruction pipeline contains **over 700 lines in Python.**

Tailbeat Pose Estimation for Bio-Inspired Robot

Skills: *Python, C++, YOLOv8, Hailo-8L, Raspberry Pi.* **Work In Progress.**



Using hardware showcased in Slide 6, I **trained a custom YOLO model** to enable **real-time pose estimation** of a Tunabot tail **at 30 FPS**. Tracking will be used to determine the relative position of a follower Tunabot with respect to a leader Tunabot for self-organization and schooling. Additionally, I developed a script to estimate the flapping frequency of the detected tail using the normalized max/min coordinates of the detected keypoints. Creating this pose estimation required a **custom-written Python script** as well as a modified post processing script in C++ to handle the required keypoint detection. Models were trained using ~500 images labeled in DeepLabCut (results suggest a need for a larger, more robust training set).

Demonstration Video: https://youtu.be/hY4j5fmpxBk