EyeMatic

–

Eye Anatomy Recognition ECE Capstone

Winter 2024 Design Review

Overview

1. Problem
2. EyeMatic
3. Team
4. Diagrams
5. Demo Video
6. Components
7. Machine Learning
8. Completed Tasks
9. Model Progress
10. Timeline
11. Acknowledgements
12. Conclusion + Q&A

Problem

- Cataract surgery is very common
    - ⅙ Americans develop cataracts by age 40
    - More than half of Americans develop cataracts by 80
- Patients' eyes usually rotate or move around when patients lie down during surgery
    - Makes surgery confusing and complicated at times
    - Surgeon’s attention primarily focused on camera display

What is EyeMatic? - The Solution

- EyeMatic is a camera system that utilizes machine learning in order to detect eye anatomy on a patient's eye.
- Differentiate different anatomy of eye anterior + detect eye rotation and movement → makes system smarter
  - Guide other surgical devices to do eye alignment + registration
- Allows surgeons to focus more on the surgery at hand
  - Rotation is important for surgeons to accurately perform cataract surgery

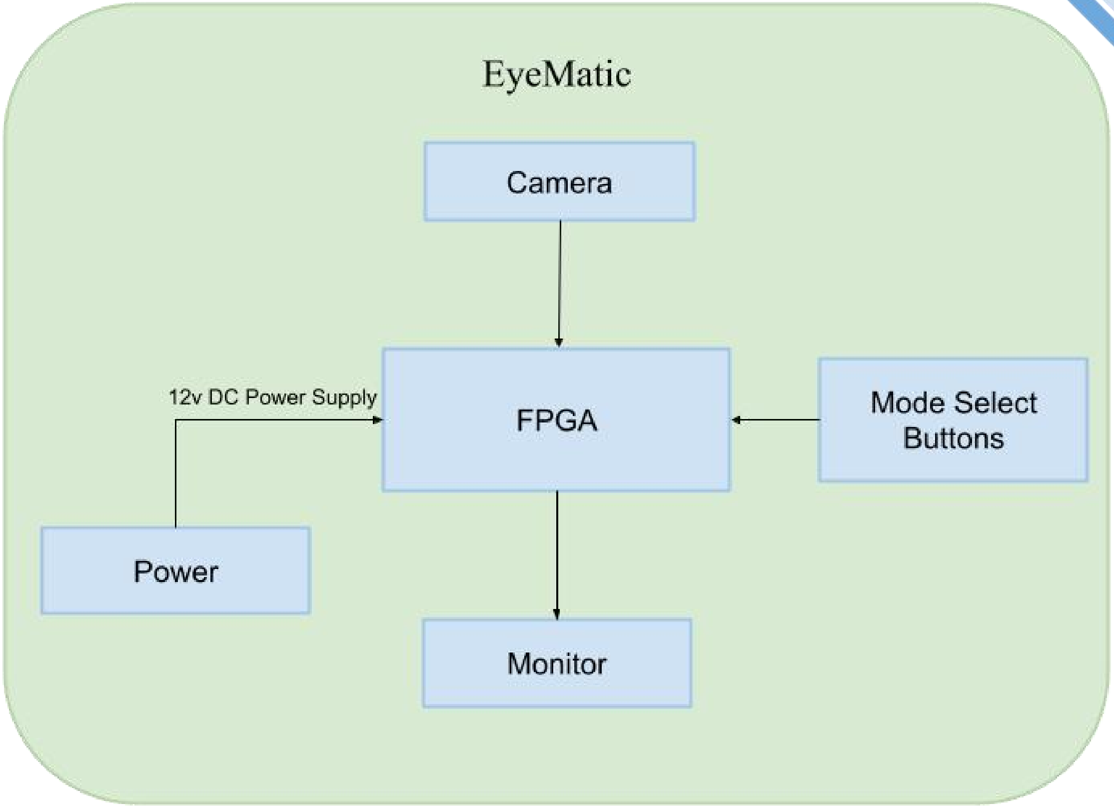
Team

- Kenya Aridomi: Model
- Andrew Chen: Training
- Michelle Ly: Project Lead, Presentation/Documentation, Training
- Ethan Nguyen: Training
- Marco Wong: Model

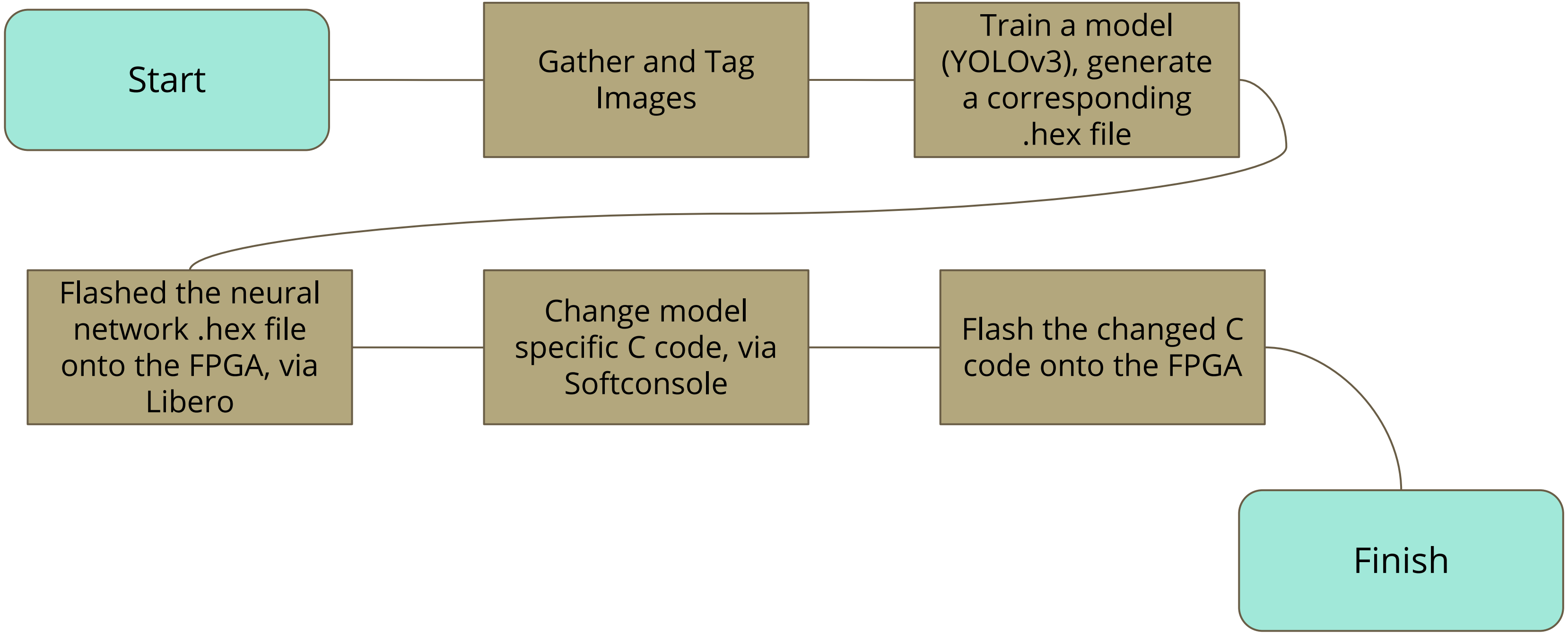




Block Diagram

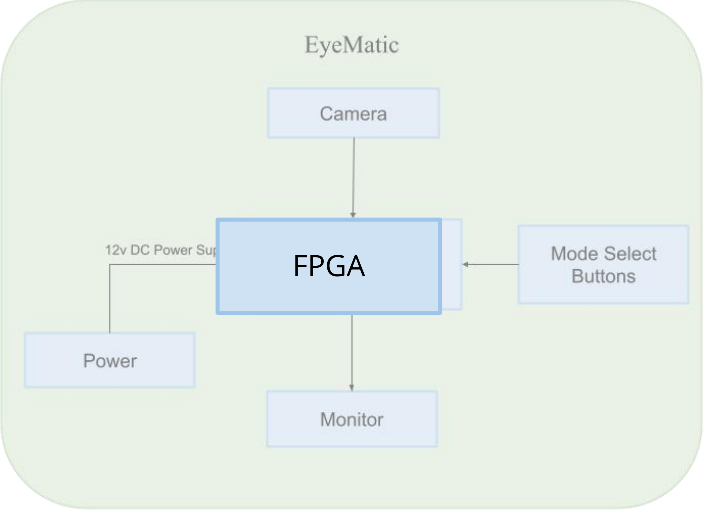


Workflow


Start
Gather and Tag Images
Train a model (YOLOv3), generate a corresponding .hex file
Flashed the neural network .hex file onto the FPGA, via Libero
Change model specific C code, via Softconsole
Flash the changed C code onto the FPGA
Finish

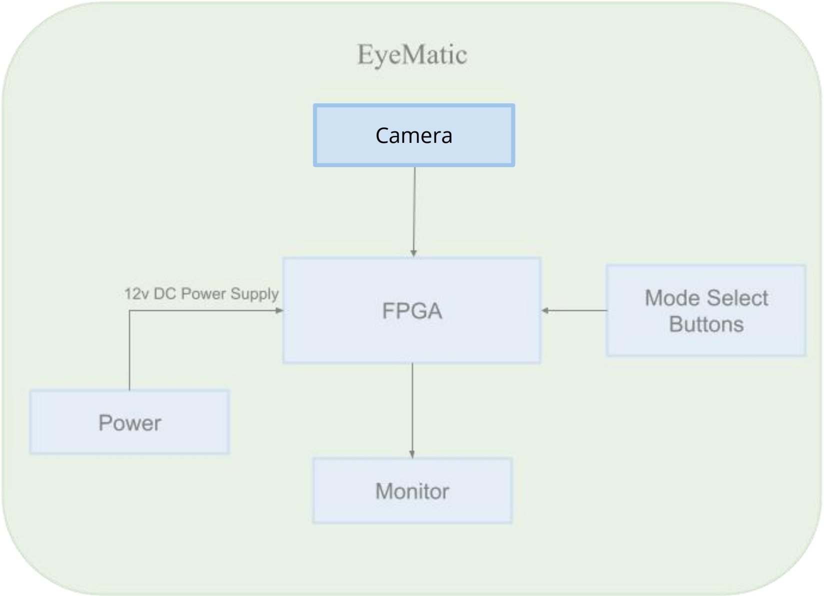
Components - FPGA

- PolarFire FPGA MPF300-VIDEO-KIT - manufactured by Microchip Technology
- 4GB DDR4 x32 RAM
- Embedded programming and debugging using SPI and JTAG
- 300K Logic Elements
- 1x 1Gb SPI Flash Memory
- USB to UART interface
- HDMI 2.0 RX and TX
- HDMI 1.4 TX



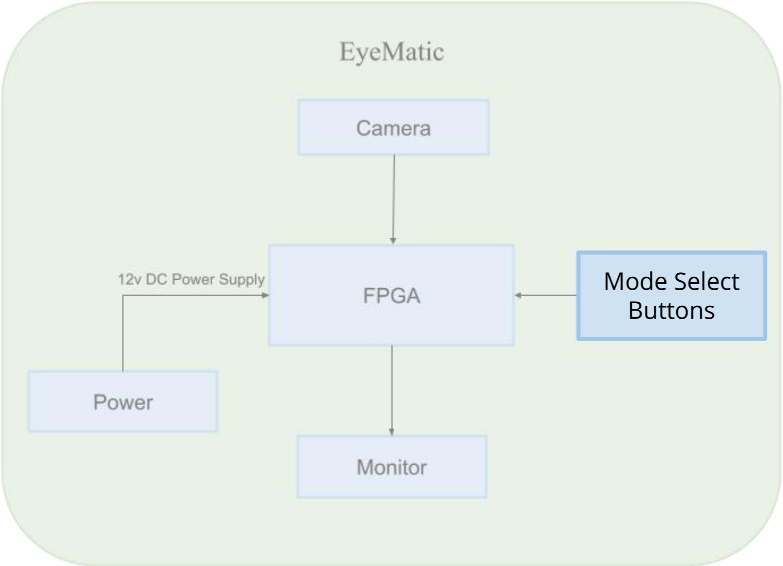
Components - Camera

- Sony Dual Camera Sensor (IMX334) over Amphenol FCI connector (CSI-2 RX)
- 60FPS RGB
- 8.42M pixels resolution

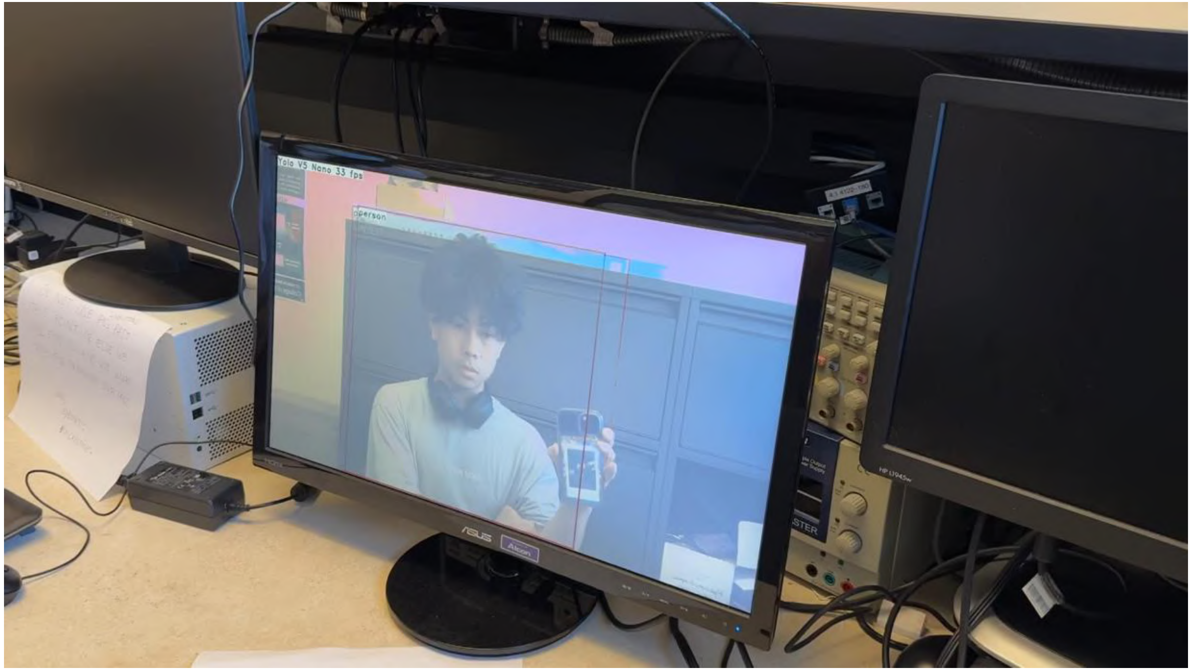


Components - Mode Select Buttons

- Allows user to pick and choose from a list of programmed models
  - Flashed and stored in SPI memory

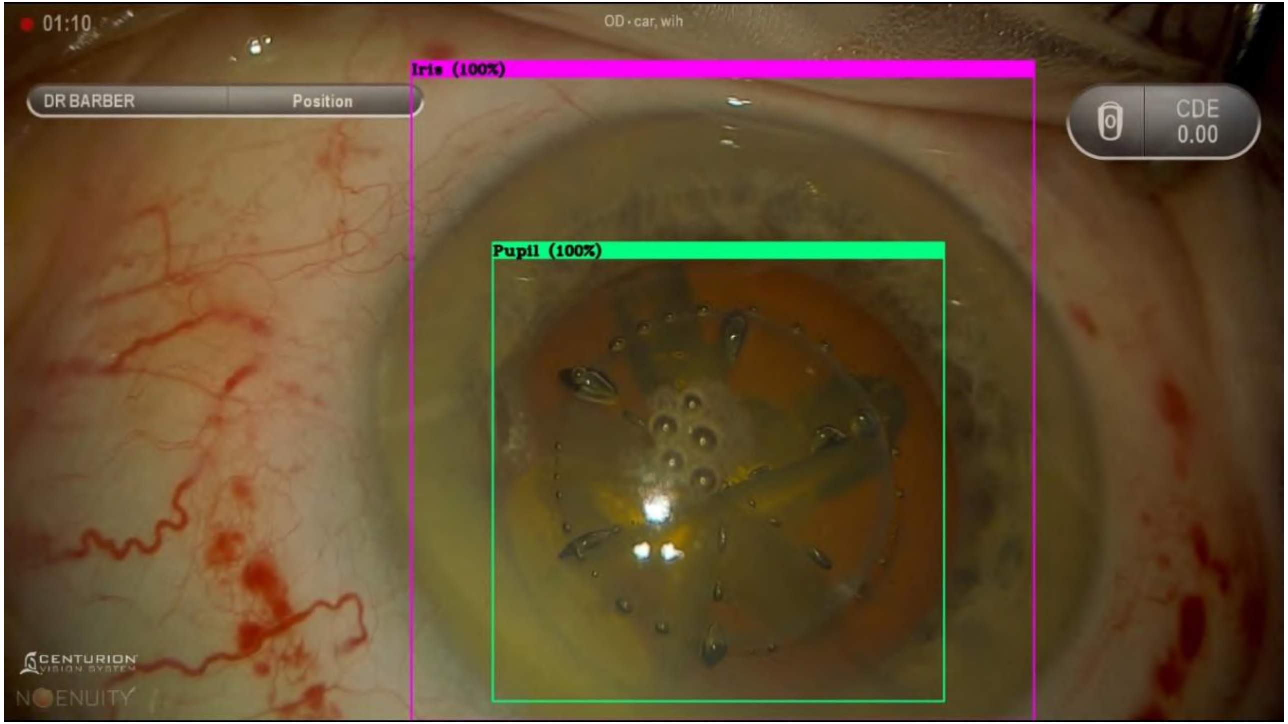


Demo Video (Flashed Models)



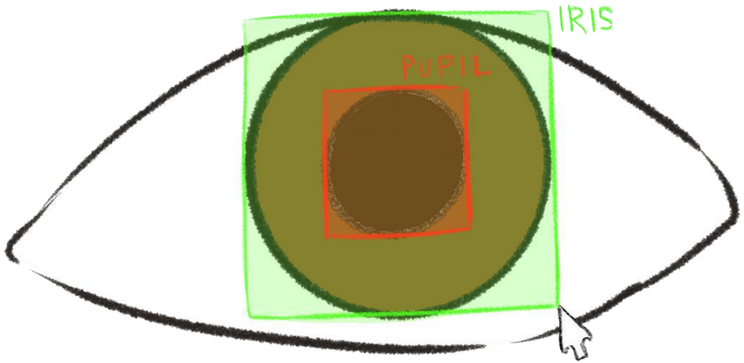
VIEWER DISCRETION ADVISED:
CLOSE UP OF EYES, SHARP OBJECTS

Demo Video (Our Trained Model)



Machine Learning

- Training YOLOV3
  - Training models
    - Eye anatomy - classification
    - Eye rotation - regression
- Tagged ~5000 images
  - Need to ensure that there's variation in data

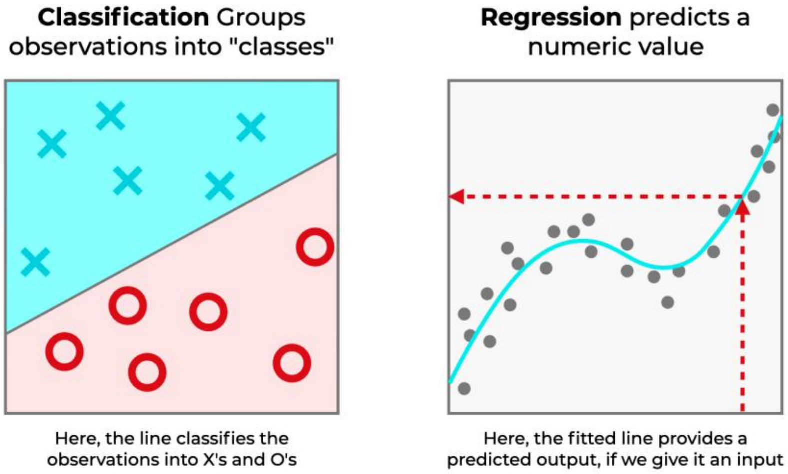


What We've Done

- Finished tagging images
- Found more images for more accurate training
- Close to finishing model training
- Uploaded model
- Began looking into creating model for eye rotation

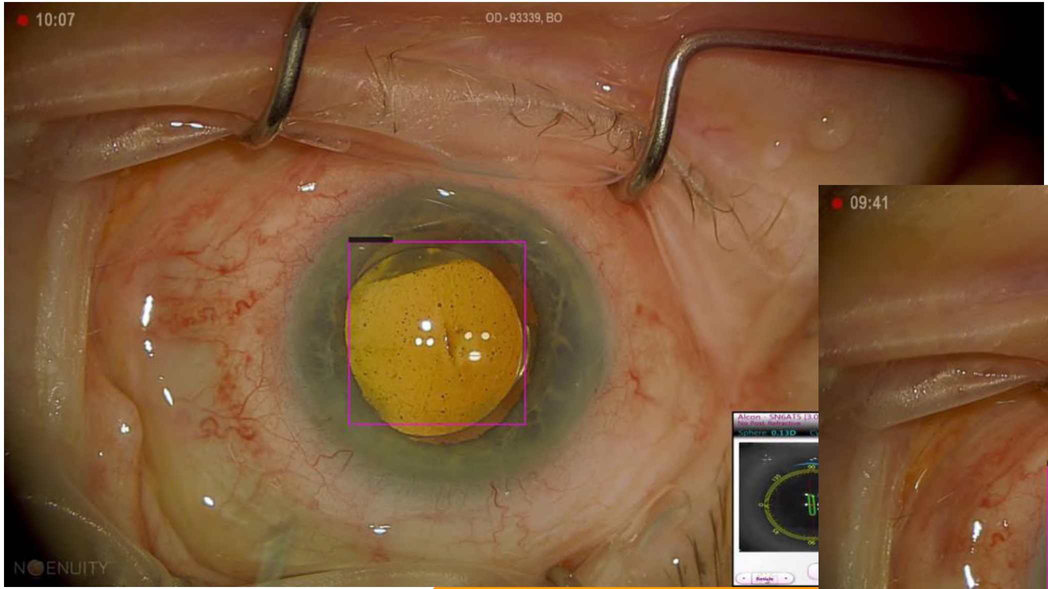
Eye Rotation Model (Possible Approach)

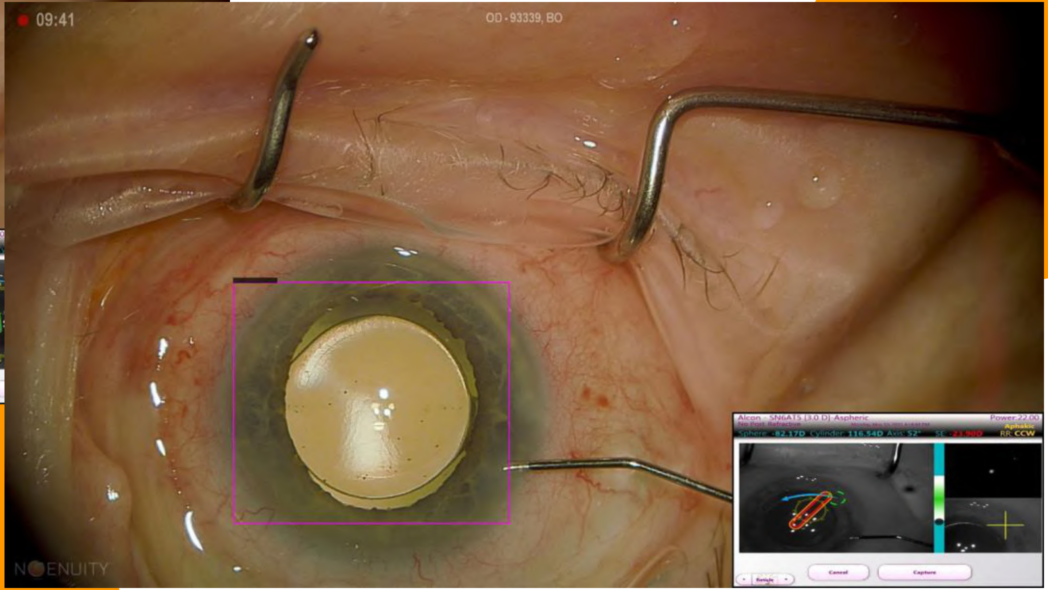
- Brute Force Method Approach
- Augment and Tag images manually for each degree
- Train YOLOv3 model as a regression task compared to a classification task



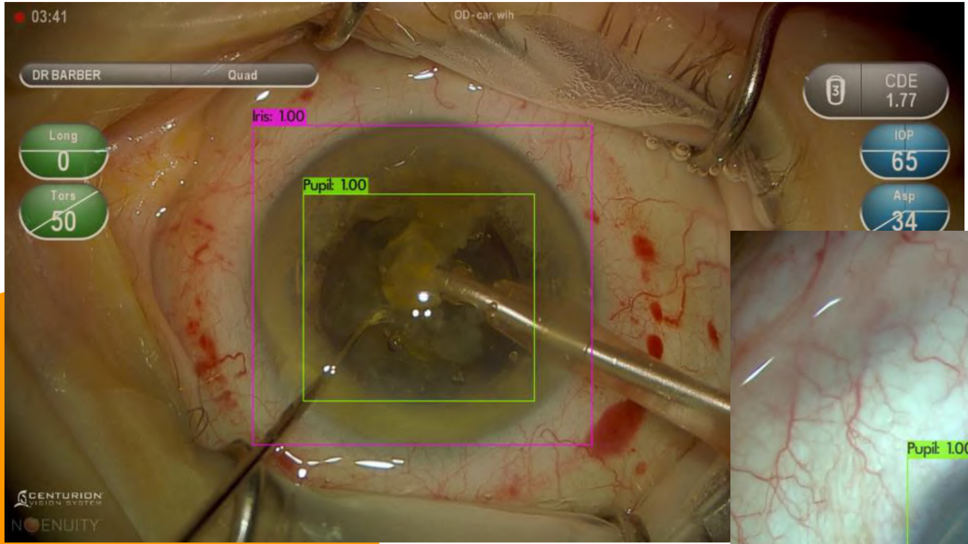
VIEWER DISCRETION ADVISED:
CLOSE UP OF EYES, SHARP OBJECTS

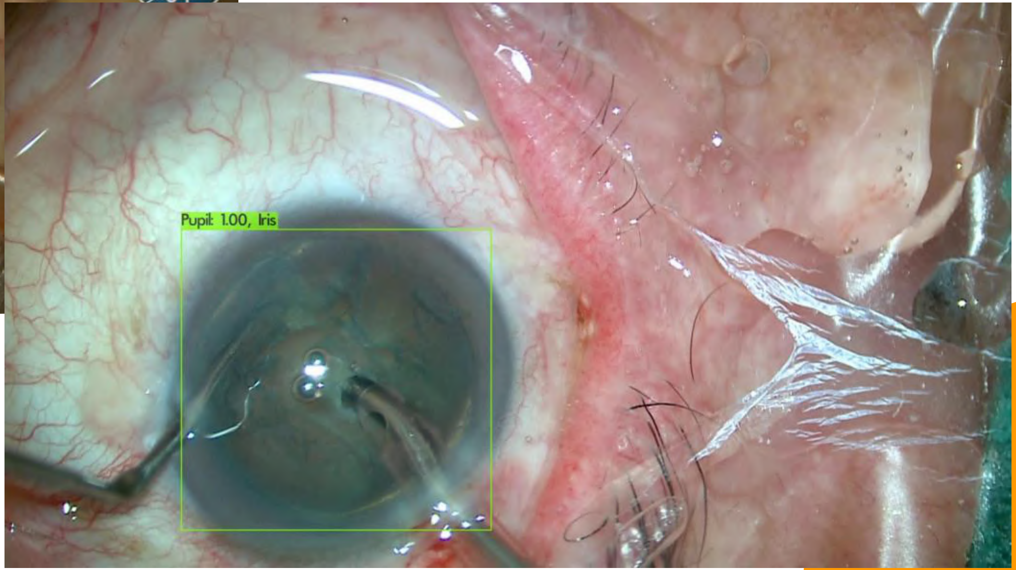
Model Progress (FALL)





Model Progress (WINTER)





Timeline

Winter
Continue tagging,
labelling, training model(s),
flash model(s) on FPGA

Spring
Train model(s),
have a working model on
FPGA
(Stretch Goal: Finding the
rotation of the eye)

Risk Analysis

- Deploying own model on the FPGA
  - Most time consuming task at the moment, the only step left before finishing the primary goal of the project
  - Meetings had with both Alcon and Microchip to debug and resolve issues
- Accuracy with model, some images where the pupil and iris look too similar have issues
  - Increased data diversity by finding more
  - Train using more tagged images, being cautious with overfitting

Acknowledgements

- Dr. Yogananda Isukapalli (Capstone Lead)
- Alex Lai (TA)
- Yuepei Hu (Alcon)
- Ky Nguyen (Alcon)
- Garo Janir (Microchip)

Thank you!

Questions?