AegisEye

Intelligent Pan-Tilt Face Tracking Camera

Jesse Barkley and Christian Shimp

Objective, Motivation, & Target Users

Objective: Our goal was to deploy a computer vision model on a lightweight, low energy, edge device to detect human faces.

Motivation: Being able to accurately identify human faces and track with low power devices has a wide range of applications from security to personal home robotics.

Target Users: Security applications, home robotics that require face identification.

Embedded ML Rationale (BLERP)

B – Bandwidth: Processes video locally to avoid network overload from streaming frames to the cloud.

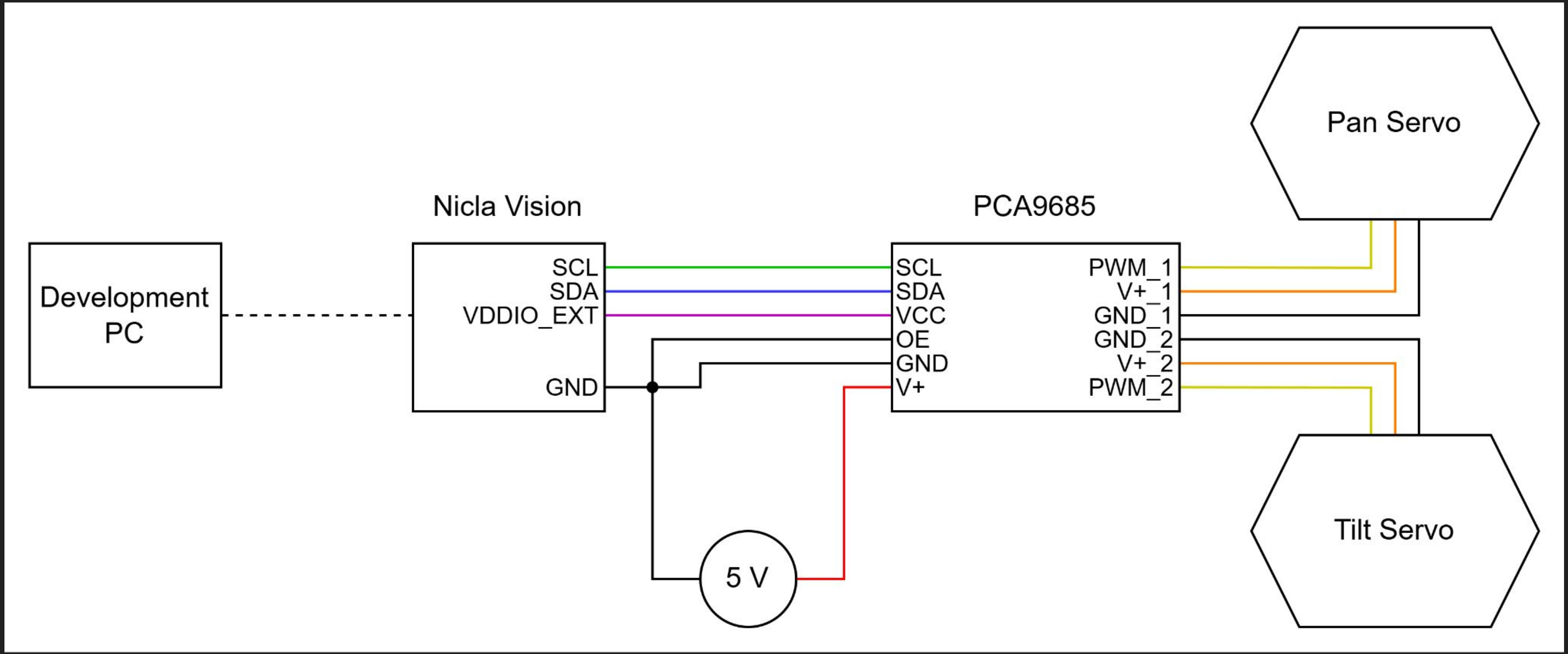
L – Latency: On-device inference provides instant face-tracking response, crucial for real-time movement.

E – Economics: Once built, it avoids recurring cloud compute costs — local processing saves money long-term.

R – Reliability: Works even when offline — ideal for mobile or remote environments.

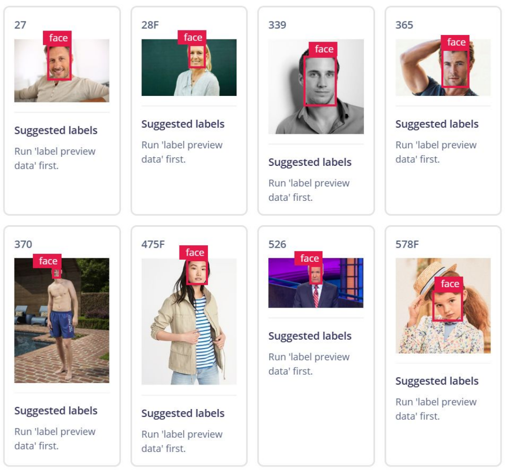
P – Privacy: Keeps all facial data on-device, preventing cloud exposure of personal information.

System Block Diagram



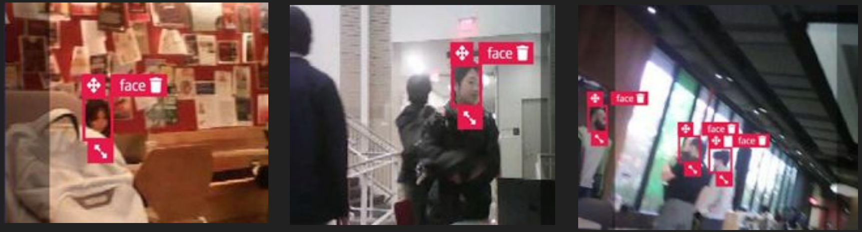
Data Collection

- Used personally collected data + Kaggle dataset
- Kaggle dataset: 833 Men + 835 Women = 1668 Faces (mixed gender and ethnicity)
- Self collected data: more than 1000 images (captured from nicla vision)
- Total = 2620 Training, 647 Testing images

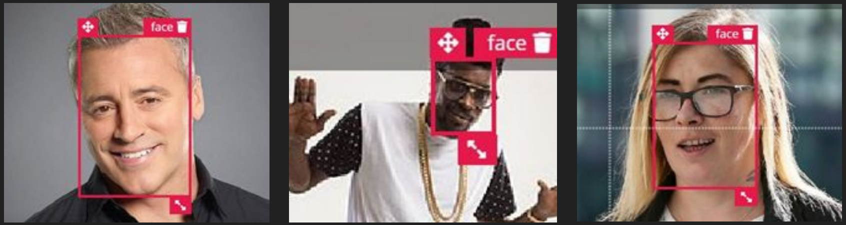


Edge Impulse AI autolabel used to streamline data annotation

Self Collected



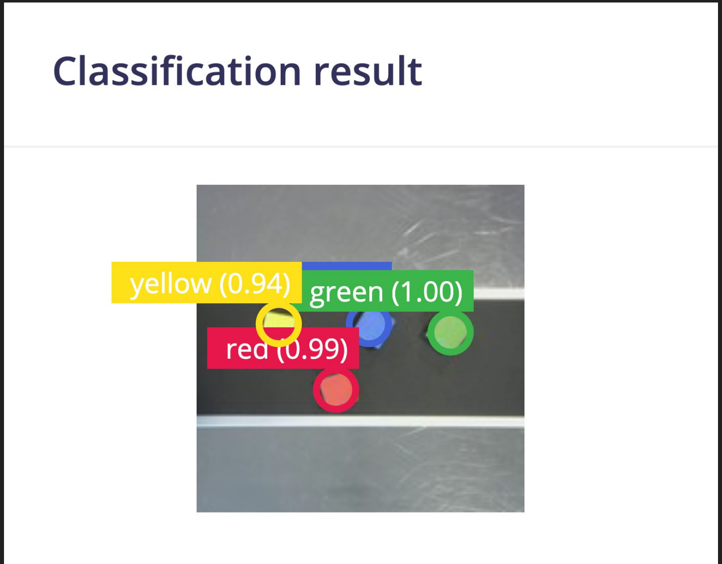
Kaggle



FOMO Object Detection

Faster Objects, More Objects (FOMO)

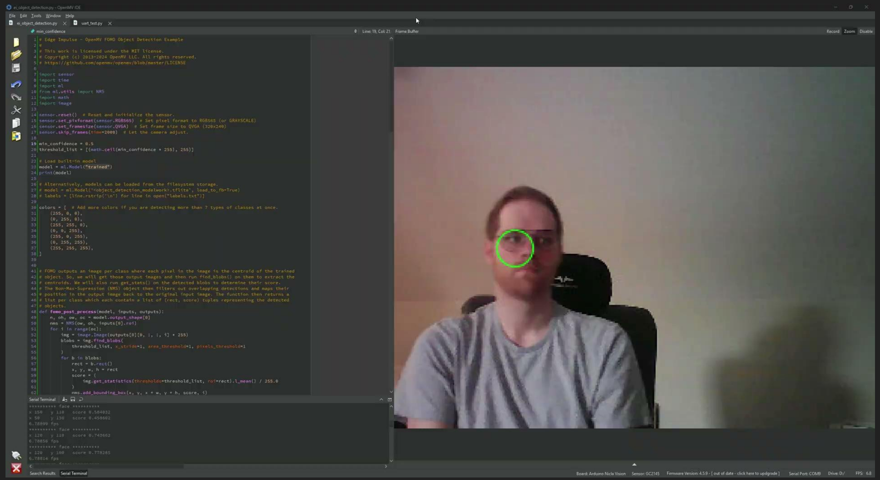
- Object detection algorithm designed for resource-constrained devices like microcontrollers
- Uses grid-based classification
- Centroid-Based Output
    - No boundary boxes



https://www.edgeimpulse.com/blog/fomo-self-attention/

Embedded Implementation

- Used OpenMV as Development Platform
  - Micropython-based
  - Integrated camera-feed for rapid development
- Modular Code Base
  - FOMO Model Component
  - Servo Control Component
  - System Logic Component
- Used a PCA9685 controller to regulate servos via I2C
  - Pan Servo controls horizontal movement
  - Tilt Servo controls vertical movement



Validation & Testing Accuracy Results

Validation Accuracy

Model — Model version: Quantized (int8)

Last training performance (validation set)

F1 SCORE
82.7%

Confusion matrix (validation set)

	BACKGROUND	FACE
BACKGROUND	100.0%	0.0%
FACE	21.4%	78.6%
F1 SCORE	1.00	0.83

Metrics (validation set)

METRIC	VALUE
Precision (non-background)	0.87
Recall (non-background)	0.79
F1 Score (non-background)	0.83

On-device performance — Engine: EON™ Compiler (RAM optimized)

INFERENCING TIME	PEAK RAM USAGE	FLASH USAGE
60 ms.	137.7K	81.3K

Testing Accuracy

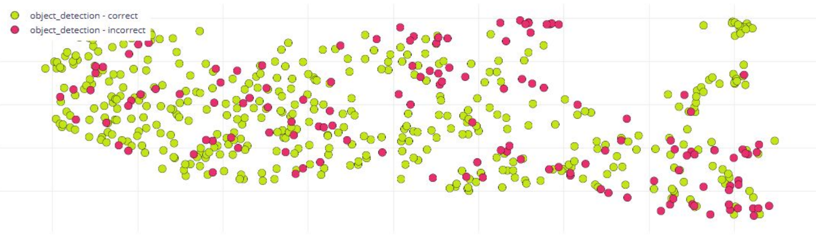
Results — Model version: Quantized (int8)

ACCURACY
75.04%

Metrics for Object detection

METRIC	VALUE
Precision (non-background)	0.87
Recall (non-background)	0.82
F1 Score (non-background)	0.85

Feature explorer



Video Example of Gimbal tracking a face (change slide title)

Challenges

- Development process of OpenMV vs. Arduino IDE
    - Implementing/Optimizing FOMO model on Arduino IDE
    - Implementing servos I2C logic on OpenMV