Machine Learning for Human Biometrics

SDDEC22-14: Ritvik Maripally, Nathanael Morris, Ron Mei Hang Teoh, Yee Shen Teoh, Zi-Jan Wong

Website: https://sddec22-14.sd.ece.iastate.edu/

Advisor: Dr. Akhilesh Tyagi
Client: JR Spidell

Problem Statement

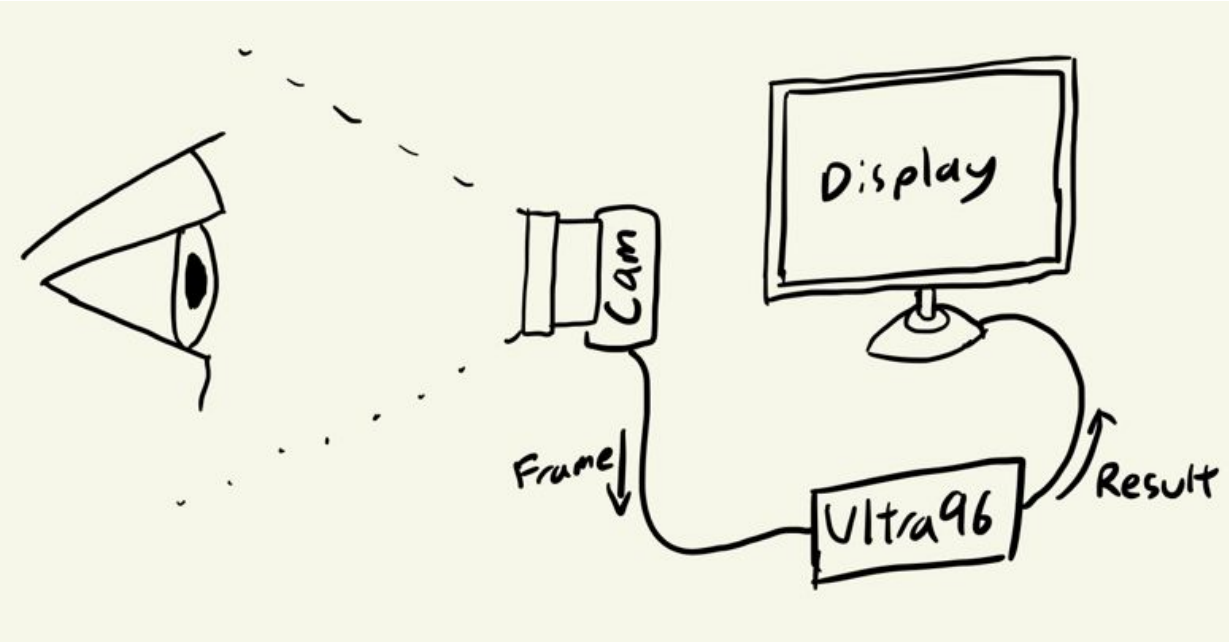
- Truck drivers are required to constantly focus on the road for long hours while driving.
- If a truck driver becomes tired or stressed while driving, this could be hazardous towards other road users and the truck driver itself.
- Sometimes when we are too focused on the task at hand, we tend to not pay attention to our surroundings and even our own physical and mental condition.

Solution

- To develop a device that monitors the truck driver’s eye movements in real time, and then predict if the driver is still suitable to keep on driving.

- Provide suitable suggestions to ensure that the truck driver is able to continue driving safely on the road.

Conceptual Sketch



Users and Use Cases

Prospective Users:

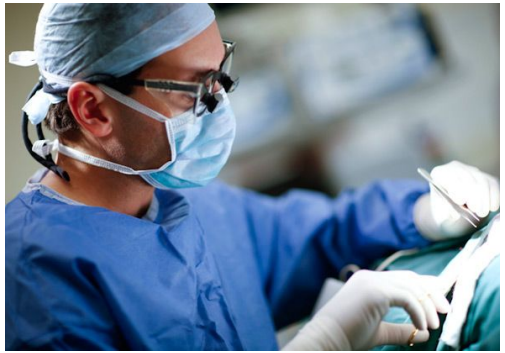
- Truck Drivers

Potential Users:

- Pilots
- Surgeons
- Any job that requires high focus and concentration
- Any job that requires work in high stress environments

Uses:

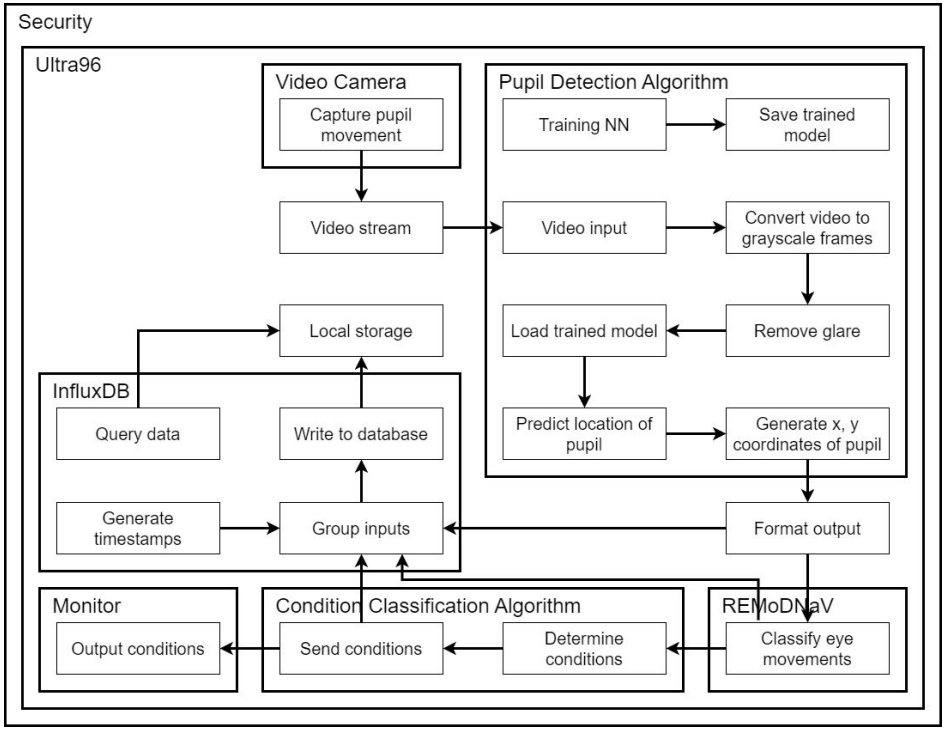
- To predict if a person is in distress or fatigued



System Design

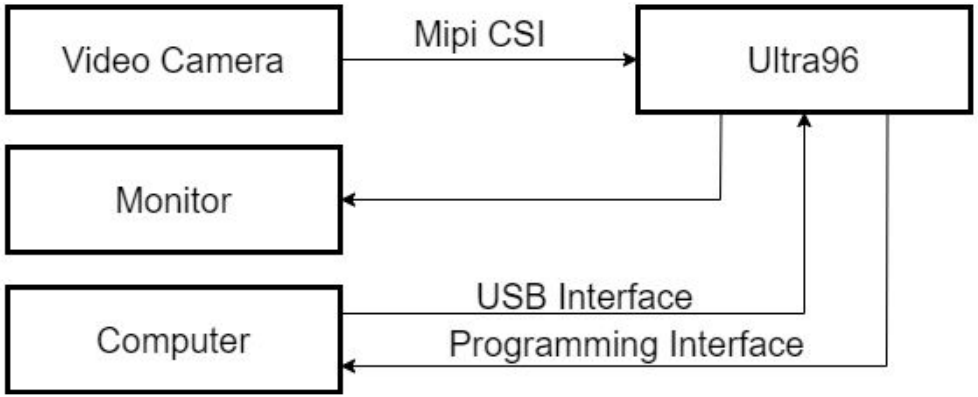
Design Approach

Detailed Program Flow



Design Approach

Detailed Hardware Flow



Design Requirements

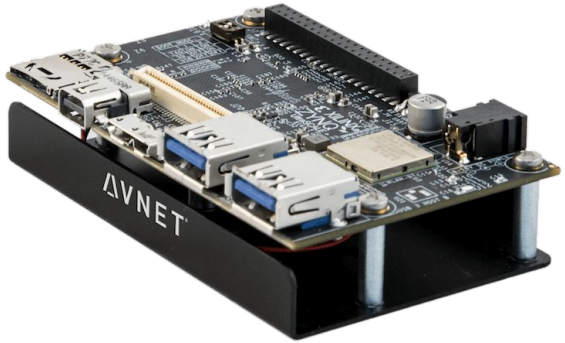
- The camera shall receive video in real time for the algorithm to process.
- The pupil detection algorithm shall process 10-bit grayscale images
- The eye movement classifier shall require a sampling rate of 30 hertz.
- The eye movement classifier shall determine if a frame is a saccade, fixation, smooth pursuit, or unknown.
- The eye movement classifier shall look for patterns in the data to figure out if a user is stressed or fatigued.
- Our project shall not violate any HIPAA laws
- The display monitor shall output the result to inform user if they are stressed or fatigued.
- Strong encryption of data when working with biometrics

Constraints / Considerations

- These are some of the constraints we had to consider:
  - If the video camera is stationary
  - Lighting condition of the room
  - Glares on the pupil from light or from wearing glasses

Functional Decomposition

- We have decomposed our project into five components:
  - Hardware
  - Pupil detection algorithm
  - Eye-movement classification algorithm
  - Database
  - Security



Ultra96-V2

Current Project Status

Hardware

Hardware

We tried using two different image for the Ultra96, both are based on Linux:

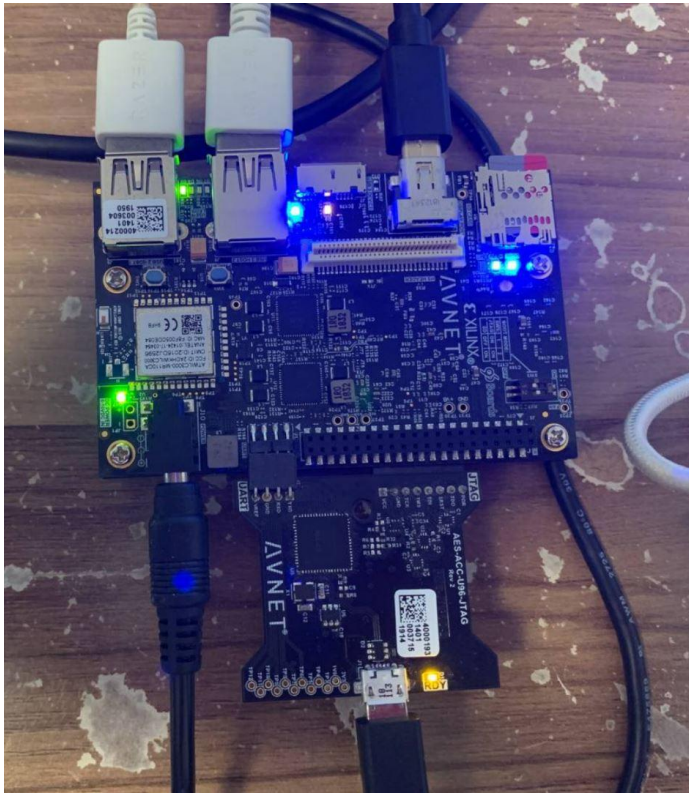
- Factory Image
- PYNQ Image

The images is saved to a 16GB microSD card.

Hardware

Factory Image

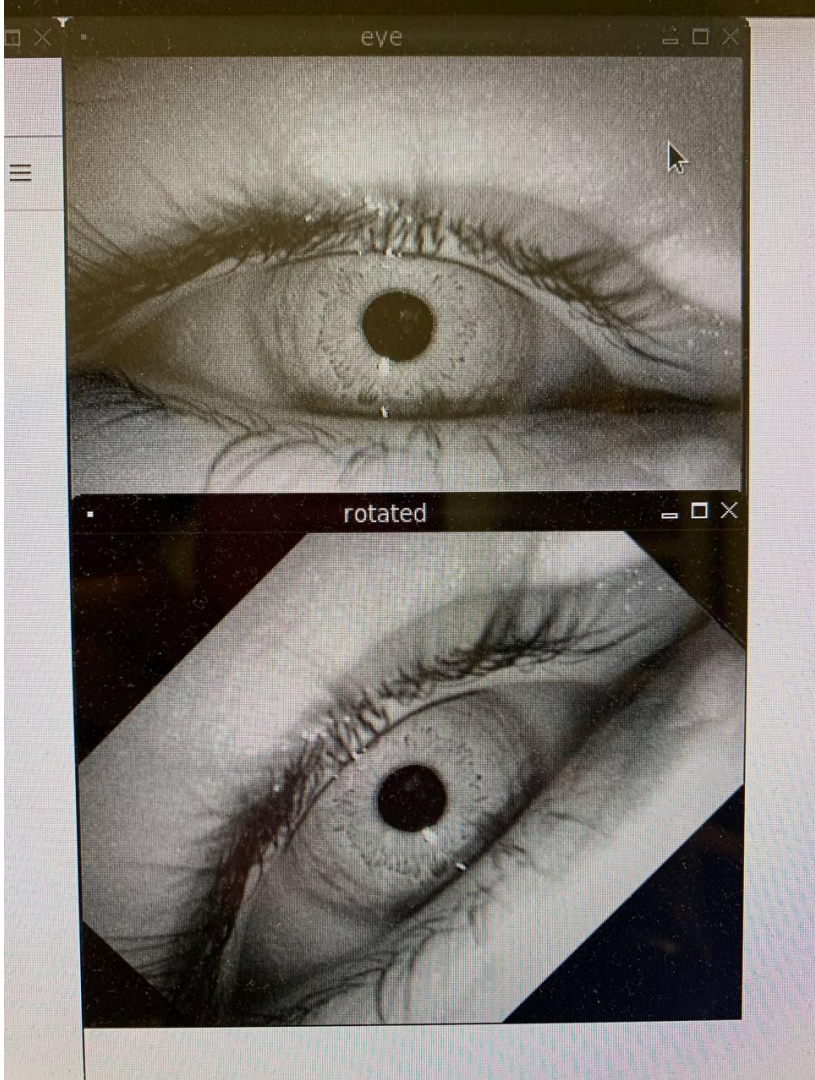
- The image environment is displayed on a displayport monitor.
- There is no application downloaded.
- Tried using apt-get at first, but found out it wasn’t on the board.
- Tried using python pip, but also resulted in errors.
- Moved on to using PYNQ image.



Hardware

PYNQ Image

- Able to display and manipulate image with python using openCV.
- Able to display video with python using openCV
- Problem downloading tensorflow for the algorithm.
- Successfully integrated server side of database, but having issue with client side.



Hardware

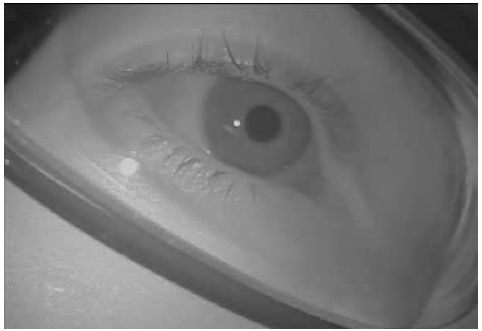
Technical Challenges

- Haven’t really work with hardware before.
- Do not have a ton of experience with linux.
- First time using Ultra96.
- Error while downloading different libraries.

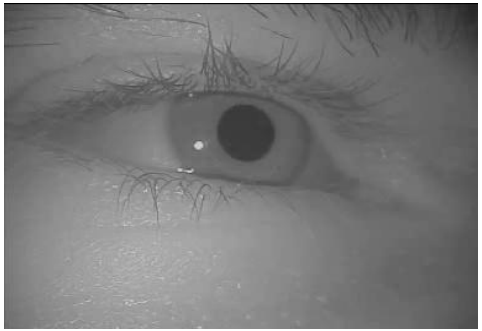
Pupil Detection Algorithm

Pupil Detection Algorithm

- Training the neural network using several datasets
- Save the model for future uses



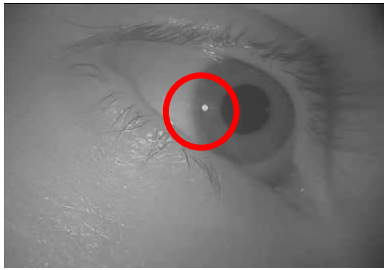
Dataset with glasses



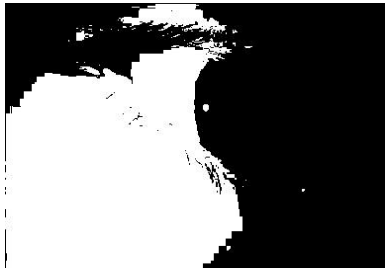
Dataset without glasses

Pupil Detection Algorithm

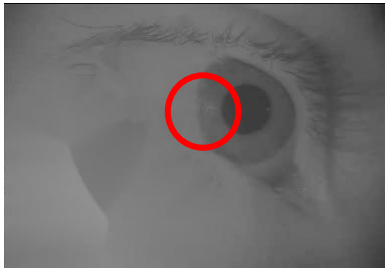
- Take in video
- Break down into frames
- Convert to grayscale and remove glare
- Predict x-y coordinates of the pupil



Glare in iris



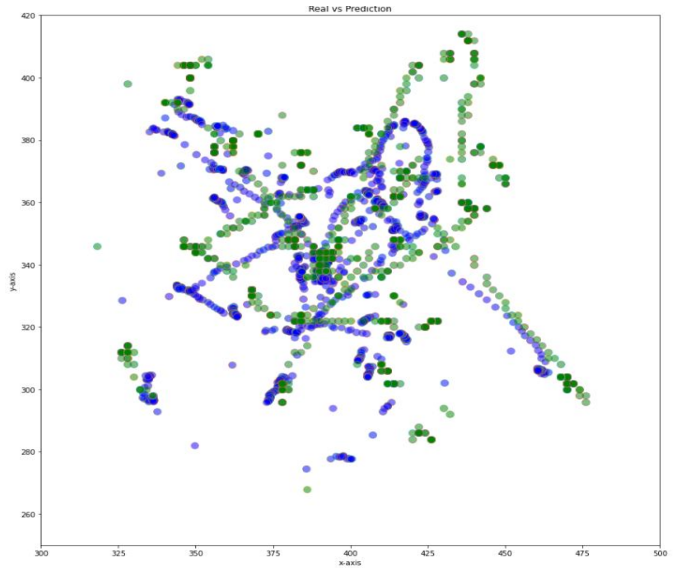
Mask



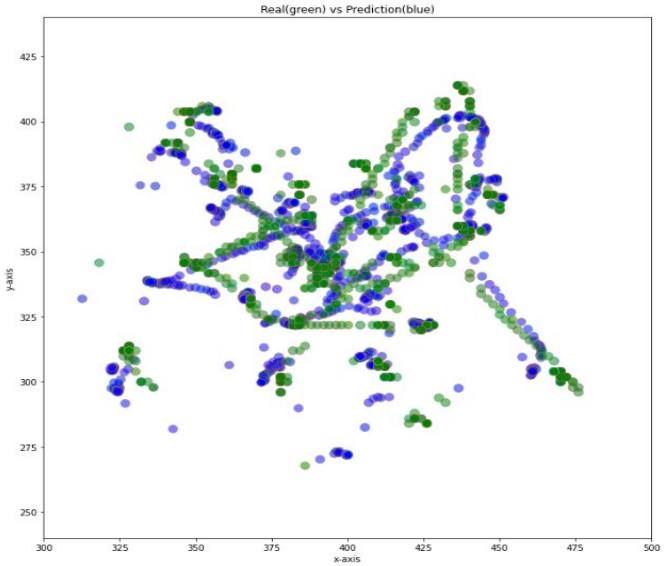
Glare removed

Pupil Detection Algorithm

Comparison between **real coordinates** & **predicted coordinates**



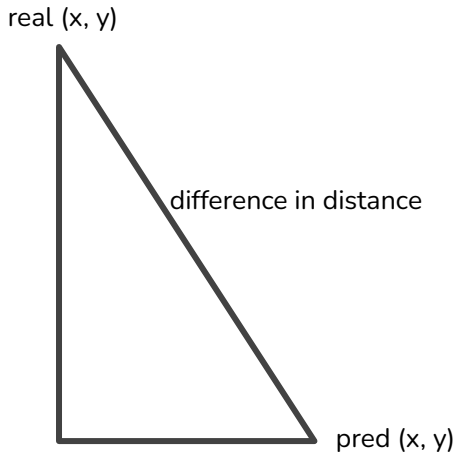
Glares in pupil



Glares removed

Pupil Detection Algorithm

- Error calculation strategy:
  - Pythagoras theorem



Pythagoras Theorem

Pupil Detection Algorithm

Technical Challenges

- First exposure with machine learning
- First exposure with Tensorflow
- First project using Python & Jupyter Notebook
- Convert model to Tensorflow lite so that compatible with Ultra96
- Glare in pupil & iris

Eye-Movement Classification Algorithm

Eye-movement Classification Algorithm

- To make eye-movement classifications, we used an algorithm called REMoDNaV (Robust Eye Movement Detection for Natural Viewing)
- Types of eye-movements classifications that can be made:
  - Fixations (FIXA) – Eye focusing on an object
  - Saccades (SACC) – Eye moving from one place to another
  - Smooth pursuit events (PURS) – Eye following a visual target
  - Post saccadic oscillations – Small movements after a saccade
  - Peak, median, and average eye velocity

Eye-movement Classification Algorithm

- Text file from pupil detection algorithm is then parsed and formatted into a new file that can be read by REMoDNaV algorithm
- REMoDNaV takes input as a tab delimited .tsv file with two columns containing the x and y coordinates respectively
- REMoDNaV then makes predictions on the type of eye movement that occured during certain time stamps

onset	duration	label	start_x	start_y	end_x	end_y	amp	peak_vel	med_vel	avg_vel
0.000	0.058	FIXA	149.7	161.0	159.1	164.8	0.188	420.772	139.792	153.212
0.058	0.373	PURS	156.8	165.5	232.9	147.3	1.453	240.922	56.791	70.079
0.431	0.064	FIXA	236.4	140.8	234.0	170.1	0.547	137.411	40.674	45.269
0.495	0.704	PURS	234.6	170.2	298.8	137.7	1.336	200.755	42.995	51.410
1.199	0.010	SACC	287.5	135.6	153.6	134.3	2.484	365.977	284.370	271.017
1.209	0.042	FIXA	150.9	138.1	228.3	179.0	1.624	154.946	72.676	69.857
1.251	0.489	PURS	228.1	178.5	250.6	131.0	0.976	536.865	51.054	66.812
1.740	0.013	SACC	237.3	131.4	159.9	153.1	1.492	247.349	112.964	136.520

Output of REMoDNaV algorithm

Eye-movement Classification Algorithm

- There are many different parameters when using the REMoDNaV algorithm which can be adjusted
- Our goal was to find the parameters that would result in the smallest error between the predicted classifications and real classifications
- We had a dataset from our client with the real eye-movement classifications, and we calculated the error between the real data and predicted data
- The parameters to REMoDNaV were adjusted to give us the lowest error percentage

Eye-movement Classification Algorithm

Research

- In a study titled Eye Movement Characteristics Reflected Fatigue Development in Both Young and Elderly Individuals (Ramtin Zargari Marandi et al.), they found that blink duration/frequency increased and peak velocity and saccade duration decreased over time as individuals became more fatigued

- In a paper called Number of Saccades and Fixation Duration as Indicators of Pilot Workload (Iveta Škvareková et al.), they found that when pilots were had longer fixation periods on instrument panels (less saccades), is an indication that they are using higher levels of focus, which could lead to cognitive overload and stress

Database

Database

InfluxDB

- Time-series database (TSDB)
  - Open-source
  - User-friendly UI
  - Optimized for time-stamped or time series data

- Purpose of having a database
  - Comparison purposes (Old data vs New data)
  - Use collected data to train existing model
  - Post-event analysis (analyze what went wrong)

Database

Comparison

InfluxDB	MongoDB	PostgreSQL
Open-source time-series Database	Open-source document-oriented database	Open-source object-relational database system
Written in Go	Written in C and C++	Mostly written in C
Perfect for custom monitoring and metrics collection, real-time analytics, plus IoT and sensor data workloads	Perfect for mobile app development, real-time analytics, IoT	Perfect for application development, data integrity, and building resilient and secure environments.
Optimized for time-series data	Optimized for structured or unstructured data	Optimized for security and redundancy

Database

Technical Challenges

- Spent too much time setting up the database client and server on the Ultra96 board
  - Inexperience in Linux environment
  - Ultra96 was lacking several important libraries.
  - Faced several errors while installing the necessary libraries.

- Unable to run InfluxDB on the Ultra96 board
  - Suspect something wrong with file path access permissions

```
ron@ron-VirtualBox:~/Desktop$ wget https://dl.influxdata.com/influxdb/releases/influxdb2-2.4
.0-xxx.deb
sudo dpkg -i influxdb2-2.4.0-xxx.deb
--2022-10-19 17:38:18--  https://dl.influxdata.com/influxdb/releases/influxdb2-2.4.0-xxx.deb
Resolving dl.influxdata.com (dl.influxdata.com)... 108.156.184.60, 108.156.184.121, 108.156.
184.124, ...
Connecting to dl.influxdata.com (dl.influxdata.com)|108.156.184.60|:443... connected.
HTTP request sent, awaiting response... 404 Not Found
2022-10-19 17:38:18 ERROR 404: Not Found.
```

Security

Security

System Security

- Anti-Malware Executable
- Whitelist
- Firewall

Authentication Methods

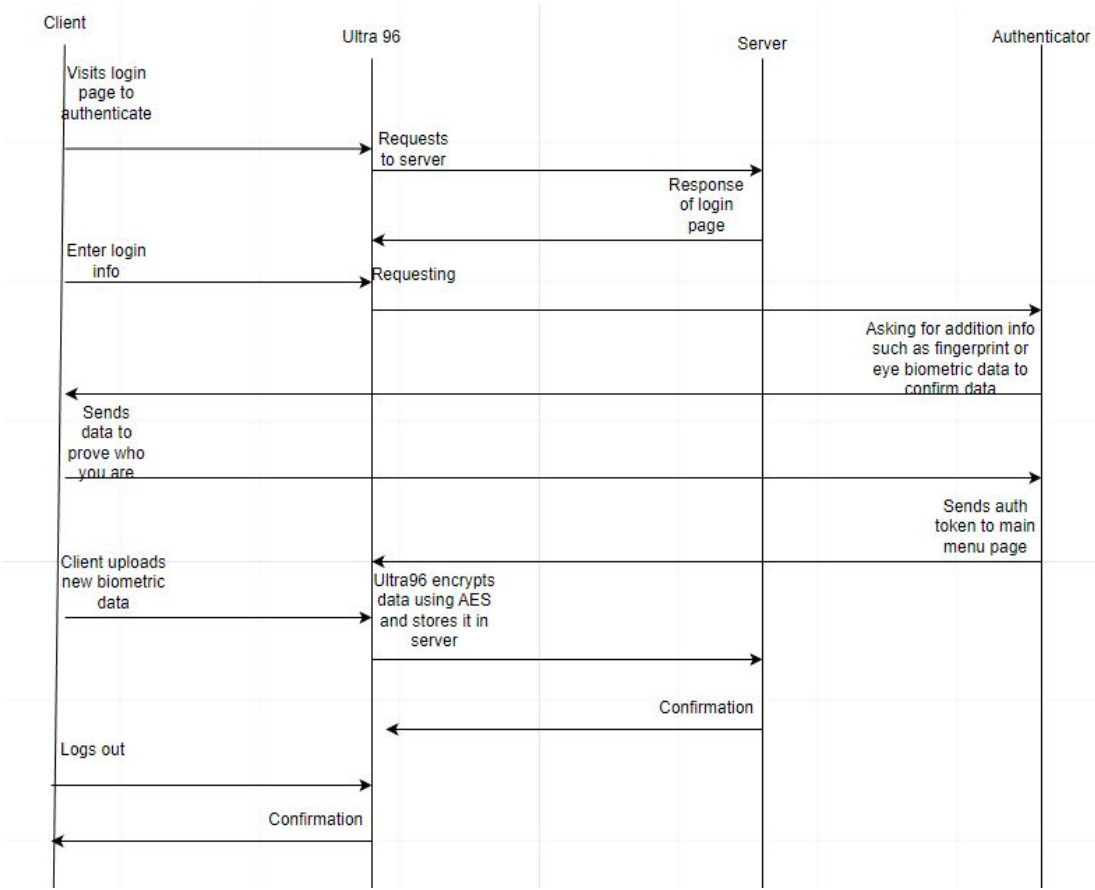
- Physical Token Based Authentication(Key Fobs, Keycards, etc)
- MFA(Universal 2nd Factor)
- Biometric(Fingerprint or Eyes scans)

Encryption Methods

- PKI(Public Key Infrastructure)
- OpenPGP
- AES(Advanced Encryption Standard)

Security

- In this scenario client will be updating/uploading new biometric data to the server.



Security

Technical Challenges

- Time Constraints
- Adaptation to threats and updating mitigations
- Implementation of security to the actual application
- New to authentication and encryption methods with little to no real testing

Demo

Demo

Pupil Detection & Eye-movement Classification live capture demo

Ultra96 - Display Image demo

Ultra96 - Display Video demo

Thank You

Questions?

Backup Slides

Future Plans

Future Plans

- Our project client informed us that this project would be passed to another senior design group to continue working on it. These are some of the features that future teams can implement
    - Adding an algorithm that uses the eye-movement classifications to predict if the user is fatigued or under cognitive overload
    - Real time graphing
    - Finding better images for hardware, or a method to download/ use tensorflow.

Sources

https://www.nature.com/articles/s41598-018-31577-1

https://pdf.sciencedirectassets.com/

Demo

Ultr96 - Display Image demo

Ultra96 - Display Video demo

Database (backup slide)

InfluxDB	MongoDB	PostgreSQL
Open-source time-series Database	Open-source document-oriented database	Open-source object-relational database system
Written in Go	Written in C and C++	Mostly written in C
Perfect for custom monitoring and metrics collection, real-time analytics, plus IoT and sensor data workloads	Perfect for mobile app development, real-time analytics, IoT	Perfect for application development, data integrity, and building resilient and secure environments.
Optimized for time-series data	Optimized for structured or unstructured data	Optimized for security and redundancy

Backup slide for database

Predict the x-y coordinates of the pupil

x	y
390.39822	342.30383
391.15472	342.66885
391.59204	343.1756
391.86282	343.48328
391.9814	344.1802
391.82986	344.2986
392.1062	344.37552
391.86453	344.4579
392.09497	344.77368
392.0745	344.8529
392.07614	345.41232
391.9408	345.00287
392.12375	345.455
392.1024	345.2744
391.8353	345.50888
392.05353	345.23242

Output of prediction (original)

x	y
390.4	342.3
391.15	342.67
391.59	343.18
391.86	343.48
391.98	344.18
391.83	344.3
392.11	344.38
391.86	344.46
392.09	344.77
392.07	344.85
392.08	345.41
391.94	345.0
392.12	345.46
392.1	345.27
391.84	345.51
392.05	345.23

Output of prediction (rounded to 2 decimal places)

Accuracy of NN model

RMSE:

```
x_rms: 35.07844243286703, y_rms: 30.34973406584388
zipped rms: 32.76448432261416
```

Average distance difference:

```
average distance difference: 6.95px
smallest distance difference: 0.2px, count: 11
largest distance difference: 45.61px, count: 6
126
```

Accuracy:

```
accuracy: 53.96%
number of acceptable x-y coordinates: 708
number of prediction x-y coordinates: 1312
```

Descriptive Statistics for Each Metric by SpO_2 Level and Mean Percent Change from Baseline

Metric	BL Mean (SD)	95% Mean (SD) ΔBL	90% Mean (SD) ΔBL	85% Mean (SD) ΔBL	80% Mean (SD) ΔBL	75% Mean (SD) ΔBL	Lowest Mean (SD) ΔBL	Post-Hyp. Mean (SD) ΔBL
Saccade Angle (deg.)	5.47 (0.88)	5.17 (0.73) -4.21%	5.21 (0.95) -3.93%	5.28 (0.77) -2.59%	4.98 (0.94) -8.13%	5.52 (0.52) -1.75%	4.91 (0.88) -9.27%	5.49 (0.96) -2.08%
Saccade Duration (ms)	36.88 (7.91)	37.08 (4.36) +2.75%	37.28 (4.63) +3.78%	40.33 (8.56) +12.14%	40.46 (10.50) +10.76%	44.33 (12.83) +18.3%	42.44 (12.83) +15.3%	36.36 (5.48) +0.80%
Saccade Rate (per min.)	58.50 (17.6)	56.45 (18.84) -4.11%	51.23 (18.25) -13.3%	54.05 (17.67) -7.44%	56.05 (20.22) -5.50%	63.05 (10.49) -4.22%	55.36 (18.71) -5.75%	49.18 (8.09) -11.85%
Saccadic Velocity (deg/sec)	168.08 (32.0)	154.83 (19.01) -6.28%	157.08 (17.65) -4.82%	149.55 (24.64) -9.20%	150.86 (23.47) -8.70%	142.48 (39.69) -12.69%	144.97 (25.55) -12.85%	164.10 (27.47) -1.21%
Fixation Duration (ms)	954.80 (263)	965.26 (324.05) +0.32%	995.19 (321.05) +3.82%	902.57 (264.97) -5.20%	854.23 (287.66) -11.74%	762.33 (288.69) -9.59%	868.03 (284.71) -9.84%	1002.72 (196.34) +10.15%
Fixation Rate (per min.)	63.10 (18.8)	63.45 (21.38) +0.001%	60.59 (20.08) -4.24%	64.24 (17.76) +2.61%	68.83 (22.66) +8.22%	69.04 (23.35) -1.50%	64.70 (22.00) +2.60%	56.59 (9.52) -6.43%
Pupil Diameter (pixels)	28.03 (3.36)	27.54 (3.30) -1.67%	26.93 (3.48) -3.98%	26.86 (3.34) -4.18%	26.09 (3.39) -6.87%	25.60 (2.15) -4.03%	27.35 (5.03) -6.78%	28.94 (5.03) +0.21%

Relative movement

Average distance difference:

```
average distance difference: 6.95px
smallest distance difference: 0.2px, count: 11
largest distance difference: 45.61px, count: 6
126
```

Descriptive Statistics for Each Metric by SpO_2 Level and Mean Percent Change from Baseline

Metric	BL Mean (SD)	95% Mean (SD) ΔBL	90% Mean (SD) ΔBL	85% Mean (SD) ΔBL	80% Mean (SD) ΔBL	75% Mean (SD) ΔBL	Lowest Mean (SD) ΔBL	Post-Hyp. Mean (SD) ΔBL
Saccade Angle (deg.)	5.47 (0.88)	5.17 (0.73) -4.21%	5.21 (0.95) -3.93%	5.28 (0.77) -2.59%	4.98 (0.94) -8.13%	5.52 (0.52) -1.75%	4.91 (0.88) -9.27%	5.49 (0.96) -2.08%
Saccade Duration (ms)	36.88 (7.91)	37.08 (4.36) +2.75%	37.28 (4.63) +3.78%	40.33 (8.56) +12.14%	40.46 (10.50) +10.76%	44.33 (12.83) +18.3%	42.44 (12.83) +15.3%	36.36 (5.48) +0.80%
Saccade Rate (per min.)	58.50 (17.6)	56.45 (18.84) -4.11%	51.23 (18.25) -13.3%	54.05 (17.67) -7.44%	56.05 (20.22) -5.50%	63.05 (10.49) -4.22%	55.36 (18.71) -5.75%	49.18 (8.09) -11.85%
Saccadic Velocity (deg/sec)	168.08 (32.0)	154.83 (19.01) -6.28%	157.08 (17.65) -4.82%	149.55 (24.64) -9.20%	150.86 (23.47) -8.70%	142.48 (39.69) -12.69%	144.97 (25.55) -12.85%	164.10 (27.47) -1.21%
Fixation Duration (ms)	954.80 (263)	965.26 (324.05) +0.32%	995.19 (321.05) +3.82%	902.57 (264.97) -5.20%	854.23 (287.66) -11.74%	762.33 (288.69) -9.59%	868.03 (284.71) -9.84%	1002.72 (196.34) +10.15%
Fixation Rate (per min.)	63.10 (18.8)	63.45 (21.38) +0.001%	60.59 (20.08) -4.24%	64.24 (17.76) +2.61%	68.83 (22.66) +8.22%	69.04 (23.35) -1.50%	64.70 (22.00) +2.60%	56.59 (9.52) -6.43%
Pupil Diameter (pixels)	28.03 (3.36)	27.54 (3.30) -1.67%	26.93 (3.48) -3.98%	26.86 (3.34) -4.18%	26.09 (3.39) -6.87%	25.60 (2.15) -4.03%	27.35 (5.03) -6.78%	28.94 (5.03) +0.21%

Glare removal with glasses

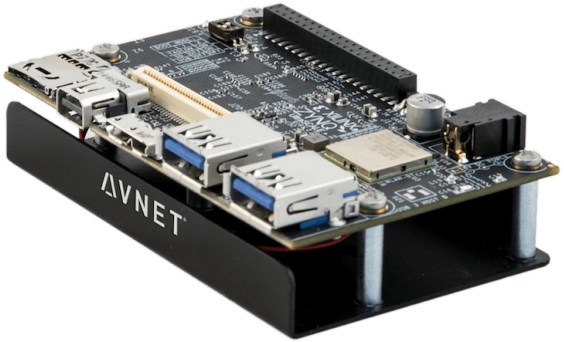
Glare in pupil & iris

Glare removed

Resource Estimate

- ArduCam for Raspberry Pi NOIR OV5647 Camera Module: MSRP $27.99.
- Ultra96-V2: MSRP $299.00.
- Total: $326.99

ArduCam Camera Module



Ultra96-V2

Backup Eye-Movement Classification Algorithm

Eye-movement Classification Algorithm

- Sampling Rate
  - Had to make sure this is equal to the fps of the video used in the pupil detection algorithm
- Px2deg
  - Used the following formula to calculate this factor: *degrees(atan2(.5 * screen_size, viewing_distance)) / (.5 * screen_resolution)* where the screen is the size of the eye video file
- Savgol-length
  - Size of Savitzky-Golay filter for noise reduction
  - Reduced the error for fixations

Eye-movement Classification Algorithm

Lessons Learned

- We learned that the eye can to determine many factors about a person’s well-being
- We learned about different types of eye-movement classifications and how they can be found through a simple input of x-y coordinates of the pupil
- We learned how important it is to have an accurate pupil detection algorithm, since it has a significant impact on the accuracy of the eye-movement classifications that are made from REMoDNaV

Eye-movement Classification Algorithm

Technical Challenges

- Figuring out how the REMoDNaV algorithm works so that I can find out parameters that should be used to reduce error
    - How different parameters will impact the output classifications and which parameters had little to no impact
- Since REMoDNaV is usually installed via pip, I had to figure out how to dissect the pip installation to use REMoDNaV within our python application instead of as a command line application

Security

Lessons Learned

- Instead of working on this alone, talk to experts and see what can be improved or added onto the methods that are in place

- Security methods are not always perfect and will be updated regularly throughout the lifecycle of the product/project

- Working with HIPAA Compliance is a very big deal and following their protocols should be taken step by step