ClimaSense

Prototype presentation

ClimaSense idea

Zero Distractions

Fully automatic climate control

Peaceful and Safe Travel

Active mitigation of driver stress and drowsiness

High Quality of Travel

Turn every journey into an enjoyable experience

Efficiency and Personalization

Optimal temperature and humidity for each individual passenger

Wellness for All

Personalized comfort even for those who cannot ask for it (infants, people with disabilities)

Prototype focus

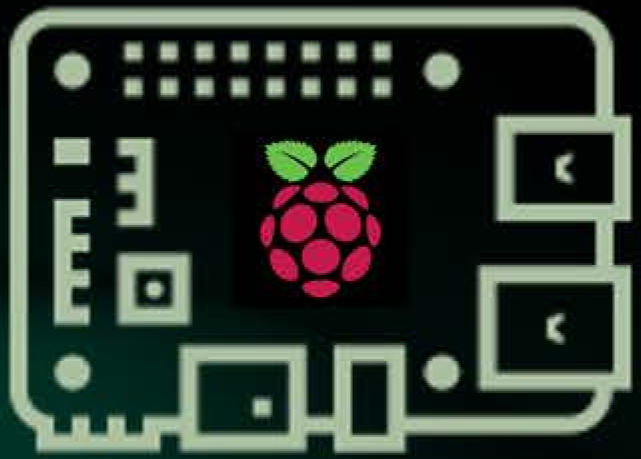
Peaceful and Safe Travel

Active detection of driver drowsiness

- Real-time video capturing and rapid face and eye analysis
- Continuous calculation of the Eye Aspect Ratio (EAR)
- ClimaSense LSTM model processes sequences of EAR values over time, specifically designed to understand dynamic changes in eye behavior

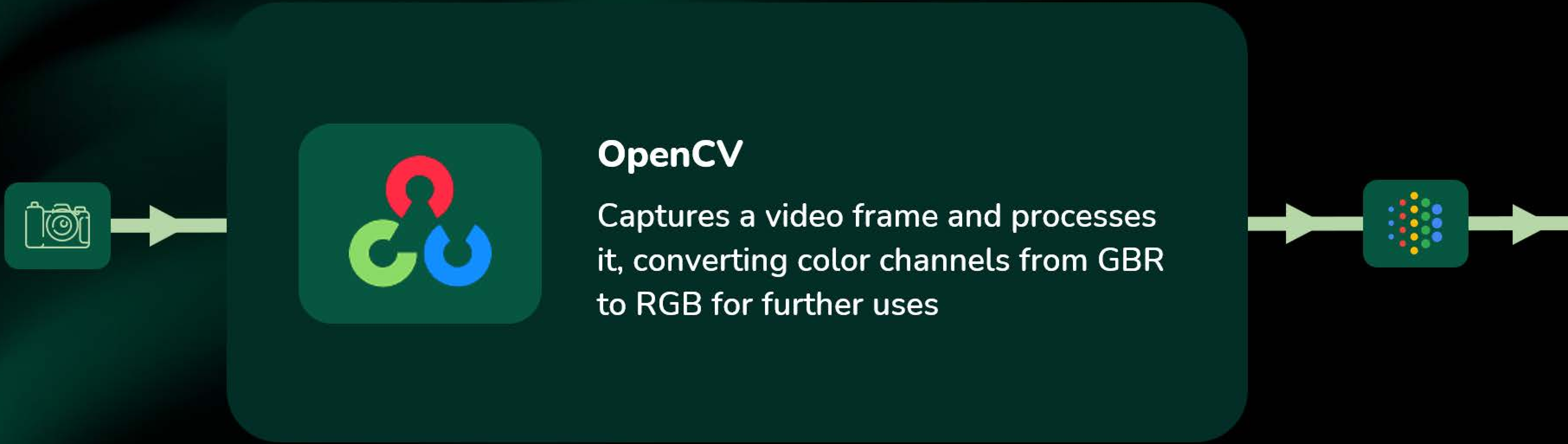
This powerful AI pipeline gains a deeper insight into fatigue levels than simple thresholding or PERCLOSE method, allowing for immediate and reliable detection of drowsiness based on temporal patterns, ensuring quick alerts to enhance safety.

Prototype



Prototype setup

Frame processing steps



OpenCV

Captures a video frame and processes it, converting color channels from GBR to RGB for further uses

Frame processing steps

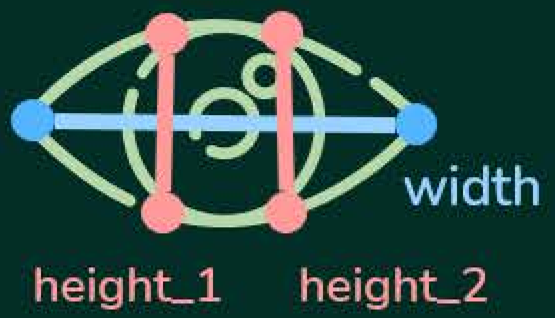
Google FaceMesh

Extracts face landmarks from the frame if a face is detected. A face landmark set consists of 478 points, and 10 of them are relative to the eye shape.
FaceMesh is a model designed by Mediapipe (Google) for real-time systems relying on edge devices.

Frame processing steps

EAR Computation

Eye Aspect Ratio is calculated as the ratio between an eye height and width. The final EAR is the average of the EAR of the right and left eyes.

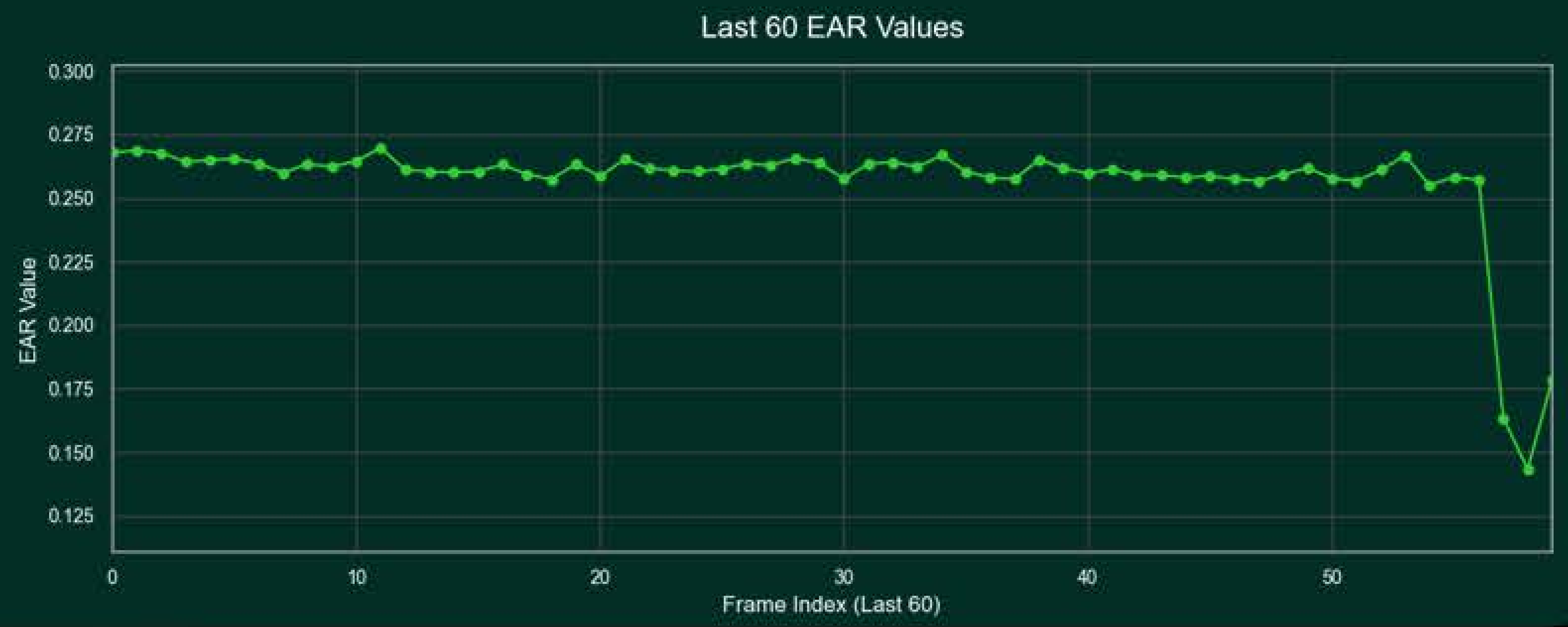


$$\mathrm{EAR} = \frac{height\_1 + height\_2}{2 \times width}$$

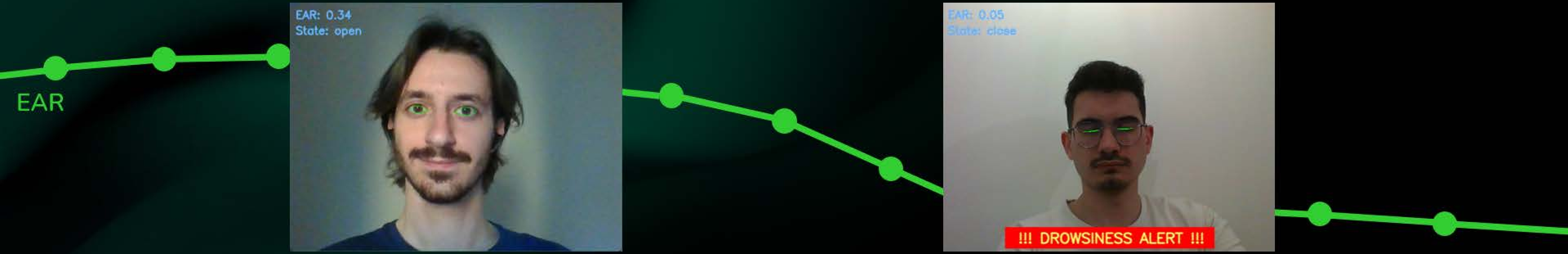
Frame processing steps

Our LSTM

We developed an LSTM that takes in input the 60 most recent EAR values and outputs the state of the eyes, directly linked to the drowsiness of the driver



Classification process



Awake state

When the eyes are in an `open`, `opening` or `closing` state, the frame (or better, the user) is classified as “awake”

Drowsy state

When the eyes are in the `close` state, the frame (or better, the user) is classified as “drowsy”

Classifications and drowsiness state are identified according to criteria learned by the LSTM when training on the DMD dataset, for which the “`close`” state refers to drowsiness conditions, specifically: microsleep and sleepy driving. The QR code is a link to the dataset annotation criteria page.

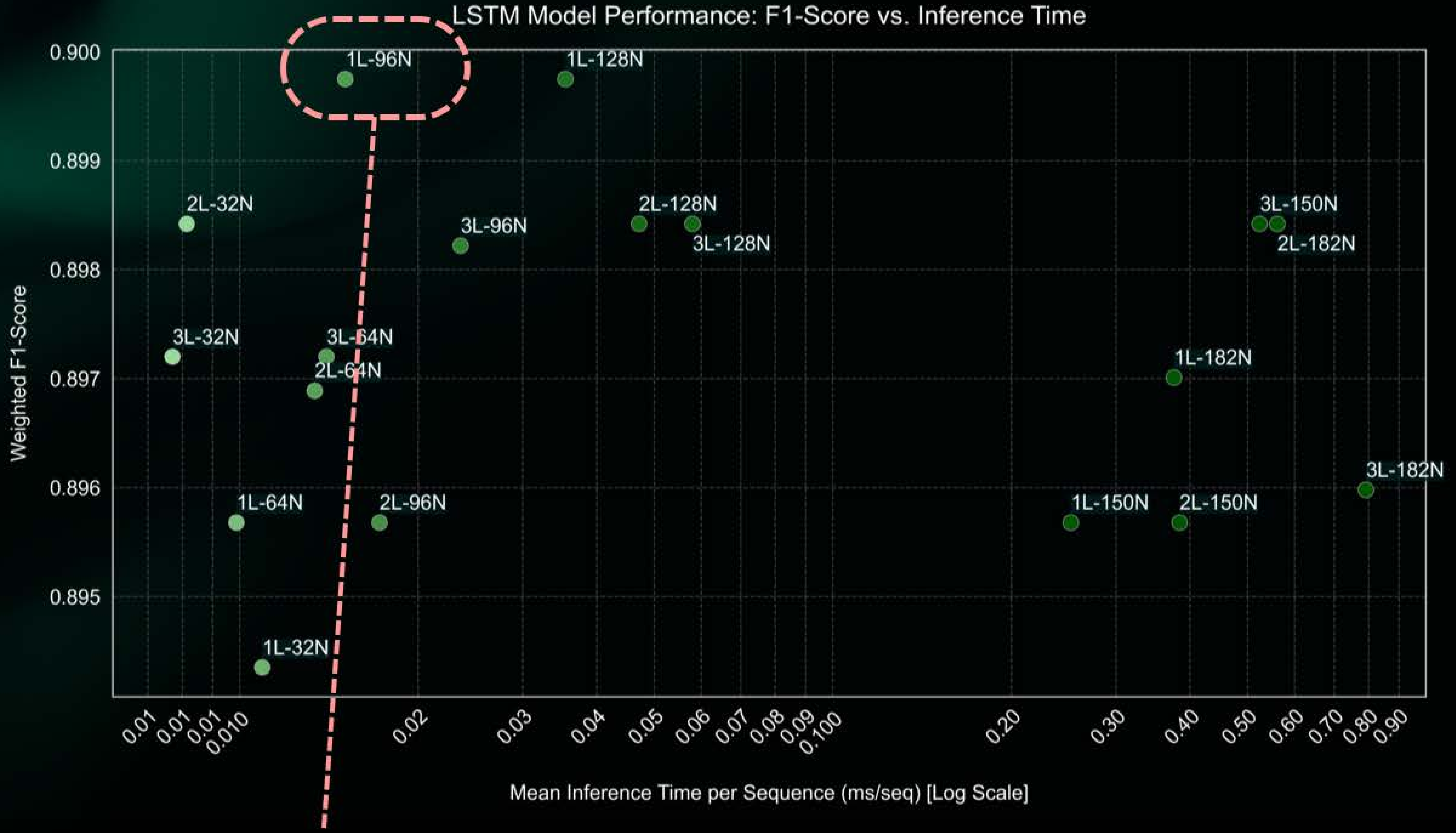
Design process for our model

We chose the LSTM architecture for its ability to capture complex temporal relations between data.

We developed several LSTMs, varying their hidden layer size and the number of layers (stacked LSTMs). The LSTMs' outputs were fed to a fully connected layer, acting as a classification head.

The best network architecture was chosen following these requirements:

- **Minimize the inference time**
- **Maximize the weighted F1-score**, giving more weight to the "drowsy" class
- **Minimize resources usage** (memory, CPU and GPU)



The model that addresses the trade-off in the best way is the one with 1 LSTM layer and 96 hidden states

Test environments

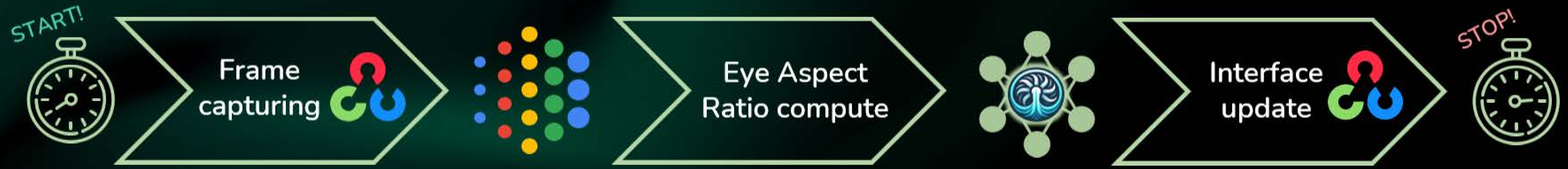
Laptop

12th Gen Intel(R) Core(TM) i7-1280P
NVIDIA RTX 3050 (CUDA 12.8)
16 GB RAM
OS: Windows 11

Raspberry Pi 3B+

1.4GHz 64-bitquad-core Broadcom
BCM2837B0, Cortex-A53 (ARMv8)
1GB LPDDR2 SDRAM.
OS: DietPi 9.12

Performances on Raspberry Pi 3B+



Frame capturing

Eye Aspect Ratio compute

Interface update

FaceMesh landmark detection

Last 60 EAR values used as input to the LSTM

Frame capturing:
9.55 ms (std. 13.93)

FaceMesh + EAR:
55.78 ms (std. 35.12)

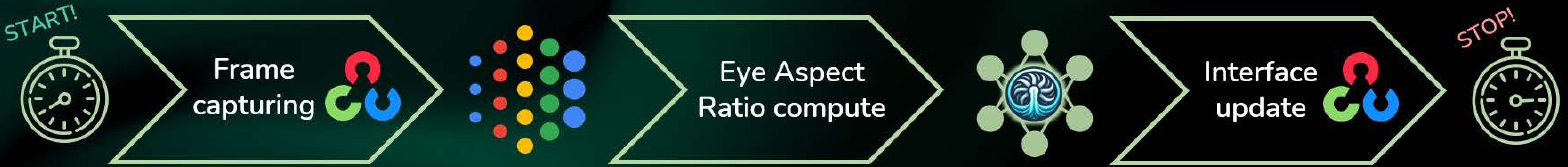
LSTM inference:
18.40 ms (std. 4.11)

Mean frame processing time (jitter): 86.51 ms (37.37)

Avg CPU Usage: 44.8% (Std: 4.5%)
Avg RAM Usage: 62.2% (Std: 0.3%)

Estimated average FPS:
11.40 FPS

Performances on Raspberry Pi 3B+ vs laptop



START!

Frame capturing

Eye Aspect Ratio compute

Interface update

STOP!

FaceMesh landmark detection

Last 60 EAR values used as input to the LSTM

Frame capturing:
9.55 ms (std. 13.93)
16.90 ms (std. 14.79)

FaceMesh + EAR:
55.78 ms (std. 35.12)
2.93 ms (std. 0.37)

LSTM inference:
18.40 ms (std. 4.11)
1.74 ms (std. 2.32)

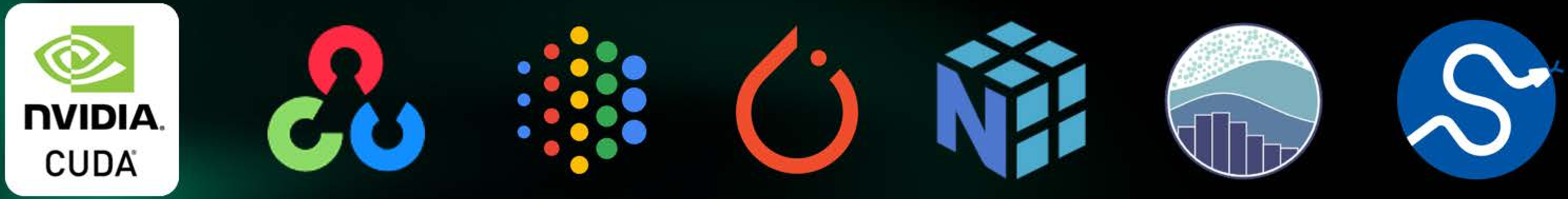
Mean frame processing time (jitter): 86.51 ms (37.37) / 23.38 ms (14.91)

Estimated average FPS:
11.40 FPS / 36.96 FPS

References

- FaceMesh documentation: https://ai.google.dev/edge/mediapipe/solutions/vision/face_landmarker?hl=it
- DMD drowsiness detection dataset home page: https://dmd.vicomtech.org
- Github DMD page – labelling criteria: https://github.com/Vicomtech/DMD-Driver-Monitoring-Dataset/wiki/DMD-drowsiness-related-action-annotation-criteria

Employed software and libraries



THANK YOU