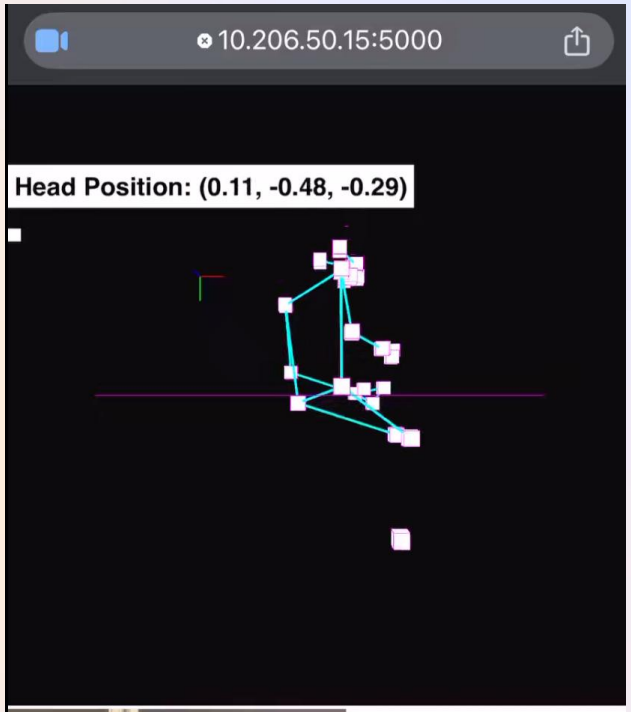
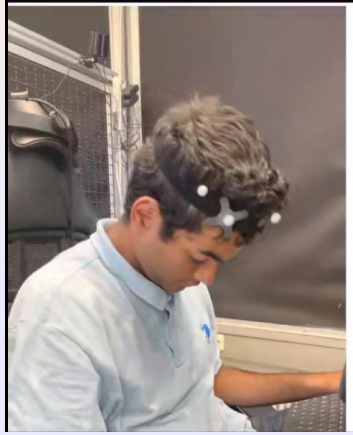
08-07-2025

# A Cross-Platform Mobile App to Detect Dangerous Driver Behaviour in Real-Time

Jai Bellare

Supervised by: Jörg Fehr

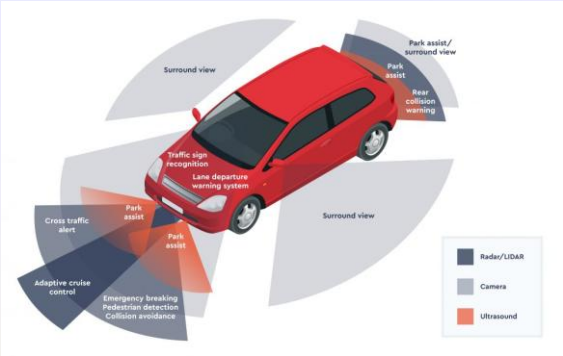


Institute of Engineering and
Computational Mechanics
University of Stuttgart, Germany
Profs. Eberhard/Fehr/Hanss

# 1. Why is Driver Monitoring Needed?

- **Driver error**, including distraction and sleep, is the critical reason for 94% of vehicle crashes [1].
- EU has mandated advanced driver assistance systems (**ADAS**) like drowsiness detection, for new vehicles from 2024 [2].
- But older vehicles rarely have ADAS. When they do, it usually monitors surrounding cars (collision detection), but not the driver




- **2-wheelers** are 25x less safe than cars (per km travelled)
- ADAS migration to 2-wheelers is difficult due to the lack of passenger restraints

- **Partial autonomy is more unsafe**: drivers assume full autonomy and stop controlling the car (they may be distracted, under the influence, or fall asleep) [3]

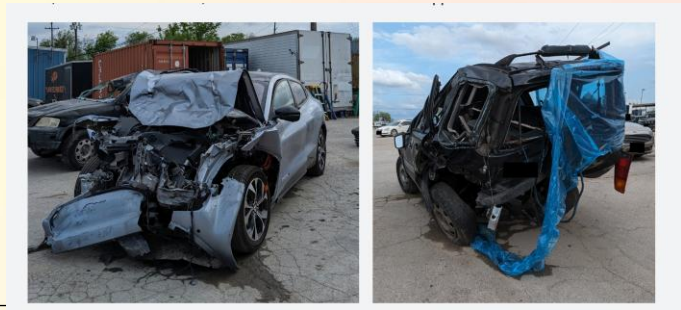


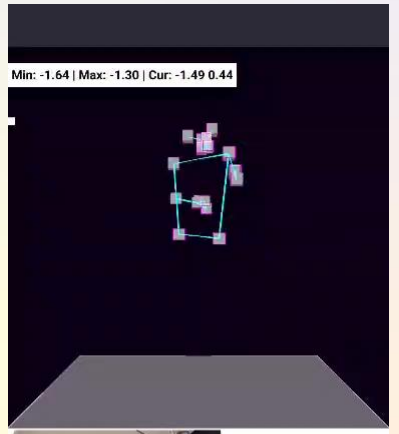
**Figure**. Ford Mustang Mach-E with extensive front damage (left) and Honda CR-V with extensive rear damage (right) photographed postcrash at the tow yard.

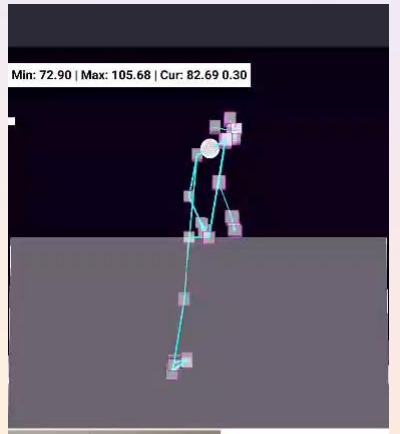
# 2. Our App

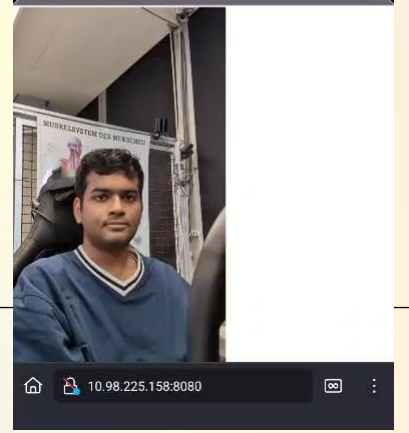
- Javascript-based so runs in the browser. Try it now: https://jjbel.github.io/itm-driver-behavior/
- So targets any device with a browser – Android and iOS phones, Raspberry Pi
- Once the app is downloaded, it can run offline (local inference)

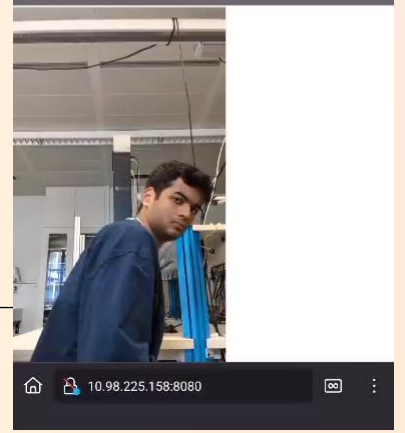
- The app uses pretrained computer-vision models from Google Tensorflow: blazepose, facemesh
- The model takes in the selfie camera video feed and estimates the 3D position of the keypoints of the human body skeleton (eyes, shoulders, wrists…)

- MIT-licensed – don't restrict safety software behind closed-source/paywalls



# 3. Prior Work

1. **Survey of mobile apps for road safety [4]**

- Out of 657 apps, only one driver monitoring app – restricted to drowsiness detection

**2. Study closest to our work:** Real-time monitoring of driver drowsiness on mobile platforms using 3D neural networks. Neural Comput & Applic 32, 9731–9743 (2020). https://doi.org/10.1007/s00521-019-04506-0

- Again limited to drowsiness detection.
- Neither app nor source code released
- Created a custom dataset of drowsy driver images, and trained a Tensorflow model on it.
  - Motivates **training our own dataset** using OptiTrack and the driving simulator

**Thus there is no available app for complete driver behavior monitoring.**

- In our app, once the body pose, face mesh, or eye gaze is estimated, multiple unsafe behaviors can be analyzed, without change to the hardware setup.

# 4. First Iteration using Apple ARKit

**Initial app by Jonas -**

- Used iPhone Pro's LiDAR camera – true 3D estimation
- Results are much more accurate than Tensorflow's model, due to the additional 3D data
- However, LiDAR access is only through ARKit
- ARKit requires **full-body visibility – not possible in vehicle cabin**

Current Tensorflow model was not trained on LiDAR data. Again, something we can do in future.

**Common theme – poor software support, security restrictions**

1. ARKit needs full-body visibility, no Android support
2. OpenPose – no mobile support. Does not run in real-time even on NVIDIA RTX 4090 GPU
3. Tensorflow – no LiDAR support
4. Epic Live Link Face – very accurate mobile face tracking, but no body tracking
5. Captury – very accurate body tracking, but not mobile
6. Our app – can't run in background due to Android,iOS security restrictions

# 5. Testing with OptiTrack, MATLAB, BeamNG

Test Setup:

- Driving simulator occupant drives around in BeamNG (the vehicle physics sim)
- Using BeamNG's Python scripting API – trigger a head-on collision with a vehicle controlled by us, at a random time
- Measure head movement using **optitrack** (the marker-based tracker) and our app **model**
- Both send data in real-time to a MATLAB script running on the PC
- The script plots the xyz coordinates of the head, for example to detect whiplash
- We compare the model with optitrack on the basis of framerate, latency, and positional accuracy

# 6. Results

| Parameter | OptiTrack | Model |
|---|---|---|
| Framerate | 120Hz | ~10Hz |
| Latency | | 500ms (rel) |
| Accuracy (for head turn) | | ±5° (rel) |

The model is much slower than OptiTrack: compute limitation (**future: try GPU speedup**)
Tensorflow tested the model to run ~30fps. Need to debug why ours doesn't

The model is less accurate: not important if detecting impulses or large-scale motion (eg whiplash)
However won't detect pre-crash muscle contractions – need Electromyography

Due to lack of LiDAR:

1. model cannot detect absolute depth. Hip is always at origin of coordinate system.
    - Not an issue: hip is usually stationary w.r.t seat in a vehicle anyways
    - Perhaps can be fixed by estimating perspective using dimensions of human

2. Relative z depth data is noisy compared to x,y

# 7. Future Work

## Short Term:

1. **Collect more pre-crash data:**
    1. Track the entire upper body, not just the head (the model already supports this. Just have to wear OptiTrack trackers)
    2. Run more elaborate crash scenarios using BeamNG and the motion platform. For example: partial frontal collision
    3. Send a simple acceleration spike to the motion platform – it is more repeatable than BeamNG

2. **Analyze human body motion pre-crash**
    1. are there any useful indicators of a crash? (eg bracing for impact)
    2. are there any indicators which can be detected within the latency bounds of the app (~500ms)

3. **Make detection more robust:**
    1. Currently, behavior checks are hard-coded as if-statements: there are fixed position thresholds, chosen empirically, beyond which a warning is triggered
    2. Instead, use a neural network to adapt the warning for different human body dimensions

# 7. Future Work

## Long Term:

1. **Create a dataset to train a computer-vision model**
    1. Tensorflow was trained on generic images of humans dancing, doing yoga. It was not automobile-oriented
    2. We have valuable combo of driving simulator for realistic motion and OptiTrack for accurately labelling the data
    3. Create a dataset of images, videos of driving simulator occupants in various behaviors: distracted, sleeping
    4. Use different phone mount positions: behind steering wheel, windshield phone holder, AC vent phone holder. Limited visibility dataset will greatly improve model accuracy
    5. Use different lighting conditions: detection at night is much harder

2. Use a **Human Body Model to estimate the risk** of injury in different positions and communicate the risk to the driver

3. Use the **rear LiDAR sensor to do mobile collision detection**, lane-change warning: cost-effective implementation of ADAS systems for old cars lacking them

# 7. Thank you

Prof. Fehr for constant guidance
Jonas for the initial app and helping at steps
Vincent for the entire sim setup & with BeamNG crash scenarios
Abhi for showing how the auto industry works
The entire ITM for amazing help!