UsAR Mirror

Team A4: Steven Lee, Anna Paek, Shengxi Wu
18-500 Capstone Design, Spring 2025
Electrical and Computer Engineering Department
Carnegie Mellon University

https://course.ece.cmu.edu/~ece500/projects/s25-teama4/

Product Pitch

Augmented reality mirrors are transforming retail by enabling customers to virtually try on accessories, driving engagement and foot traffic—while operating 24/7 without traditional fitting rooms. Yet today's solutions suffer from alignment errors and slow rendering, plus steep hardware and maintenance costs. Our more **affordable**, **high-performance** AR mirror overcomes these barriers by combining **off-the-shelf components** (cameras, displays, and consumer-grade compute) for **multi-viewpoint immersion**; **real-time rendering** on specialized GPU hardware; a **library of AR filters**; paired with an **intuitive gesture-based UI**.

Our solution achieves a real-time rendering **frame rate of 160 FPS**, end-to-end input **latency of 55 ms, input recognition accuracy of 95%,** and a **range of view of 90 degrees.**

System Architecture

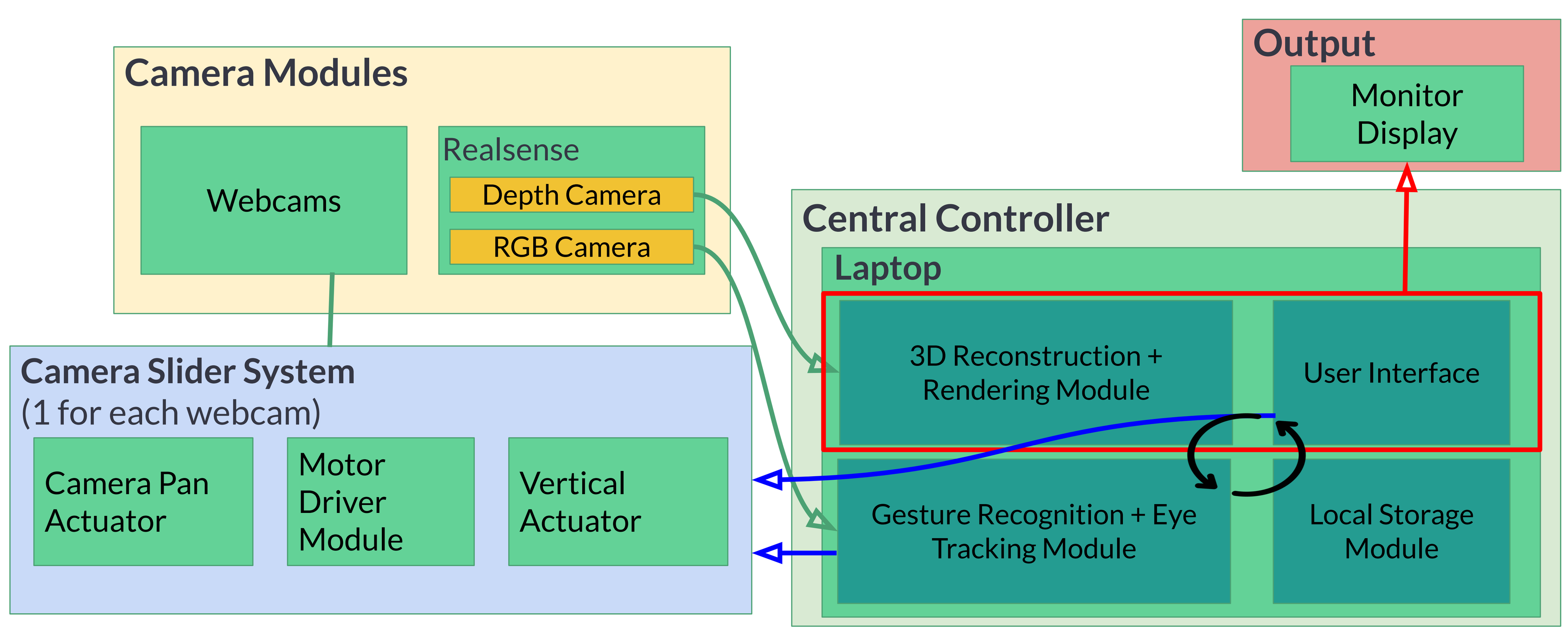

Figure 1: Overall System Architecture

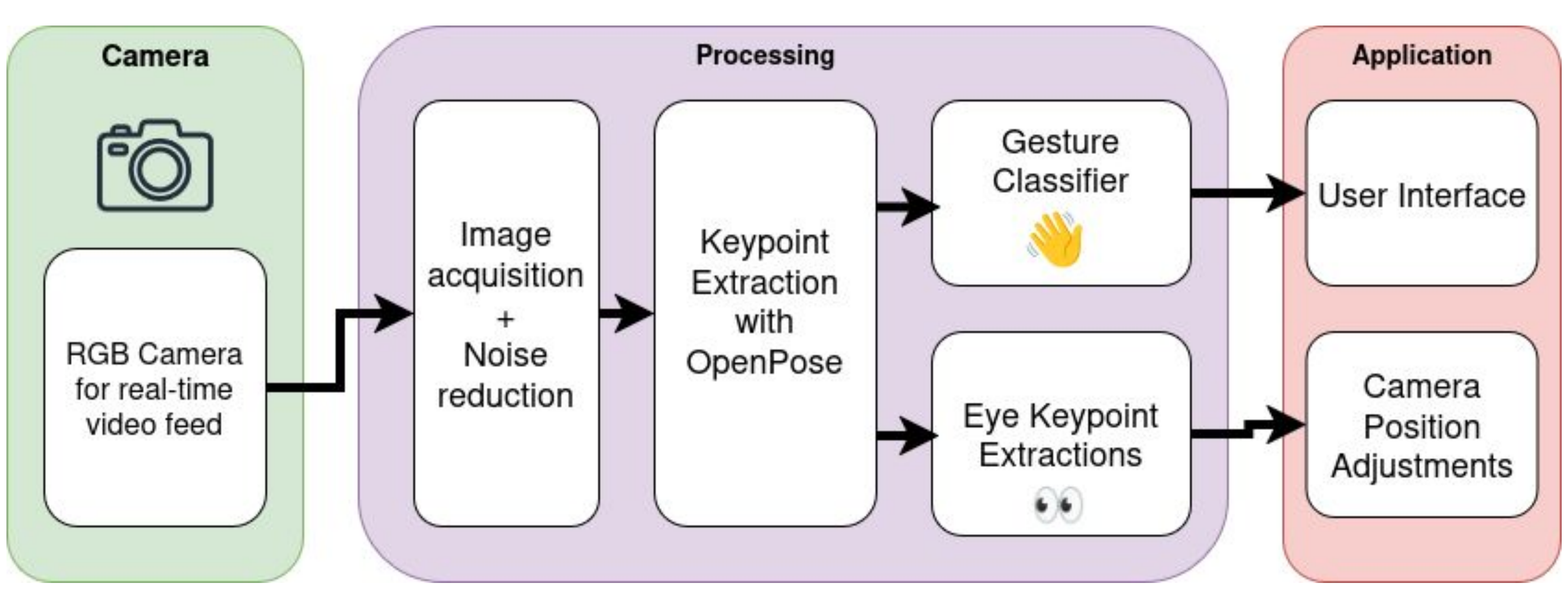

Figure 2: Gesture Recognition Pipeline

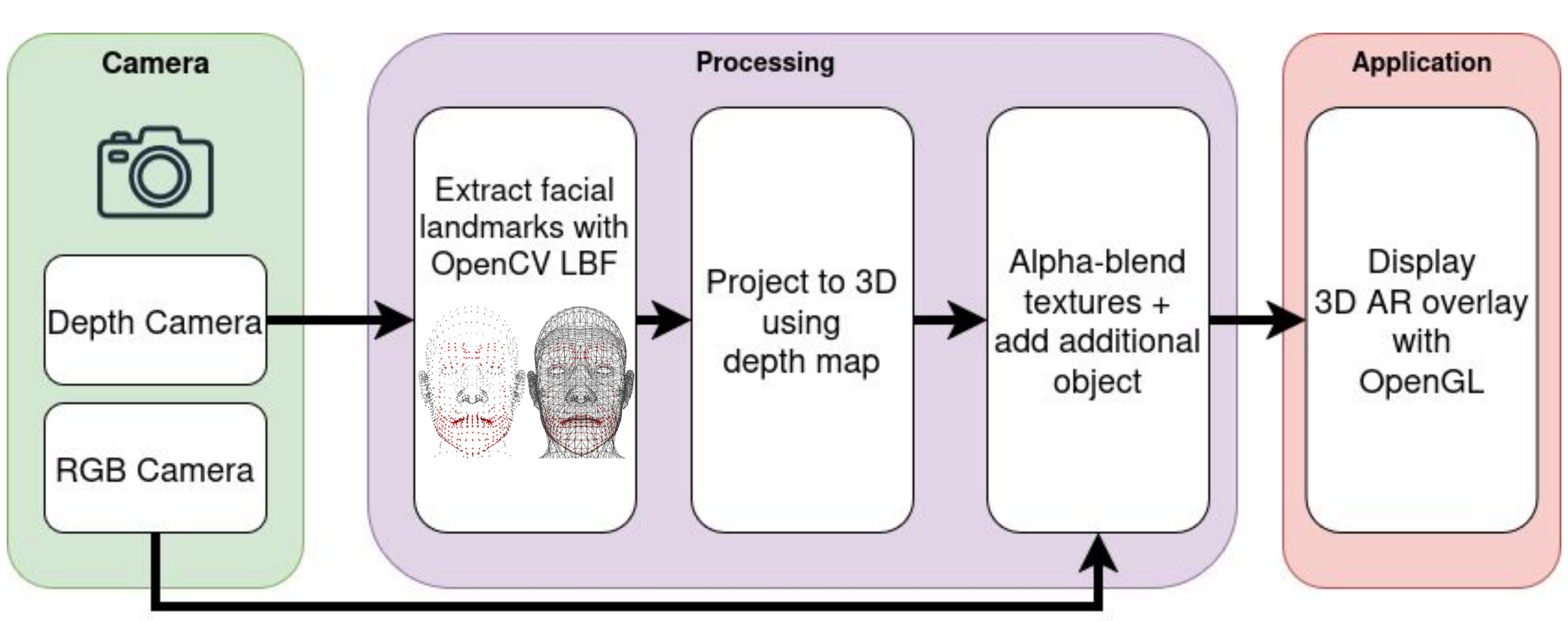

Figure 3: AR Overlay Pipeline

Conclusions & Additional Information

Overall, the UsAR Mirror realized our vision of a cost-effective, multi-view AR system—achieving sub-second 3D reconstruction, smooth overlay rendering, and robust gesture control. Working as a team taught us to test all components together early, keep each part's interfaces clear, and check performance across the whole setup. In the future, others could add smarter face-and-motion tracking, split heavy processing between device and cloud, support multiple users at once, and bake in on-device privacy safeguards to make this mirror even more useful for retail, video calls, or personal use.

System Description

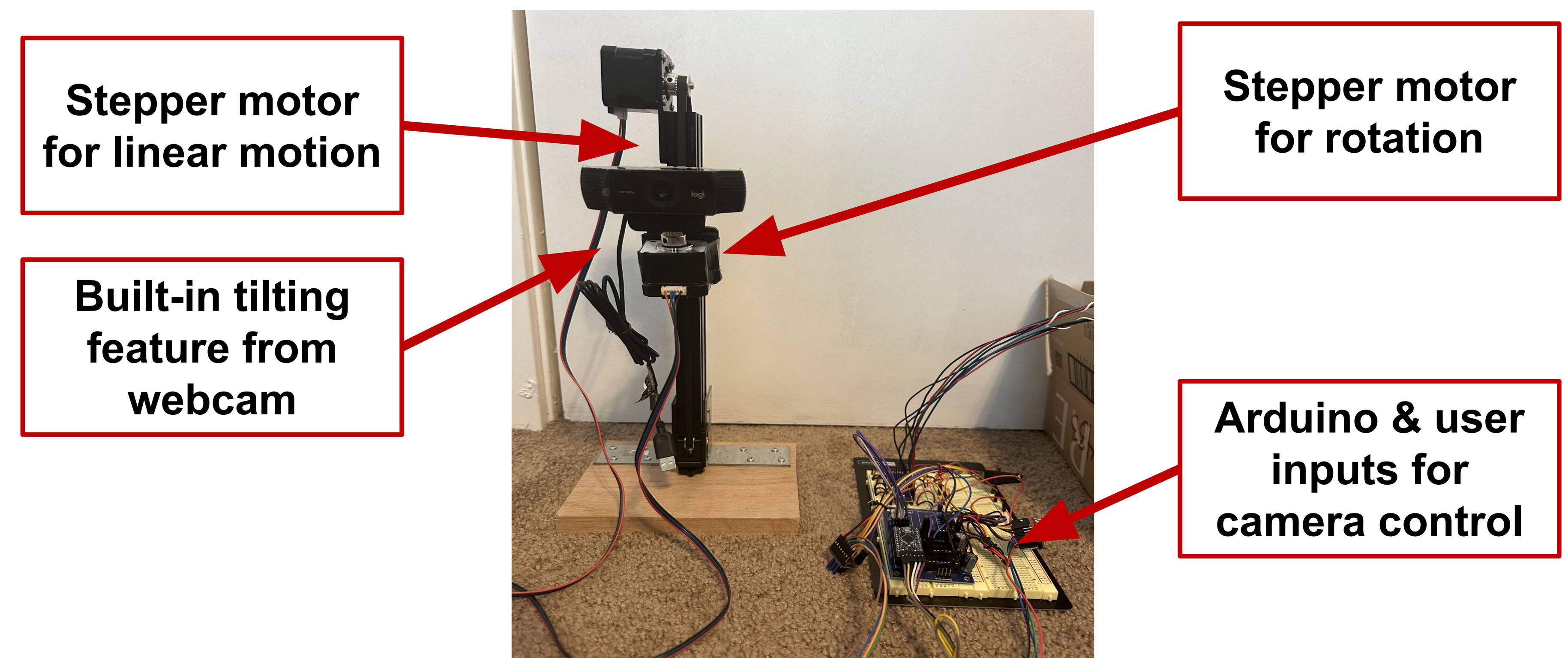

Figure 4: Camera Hardware Rig

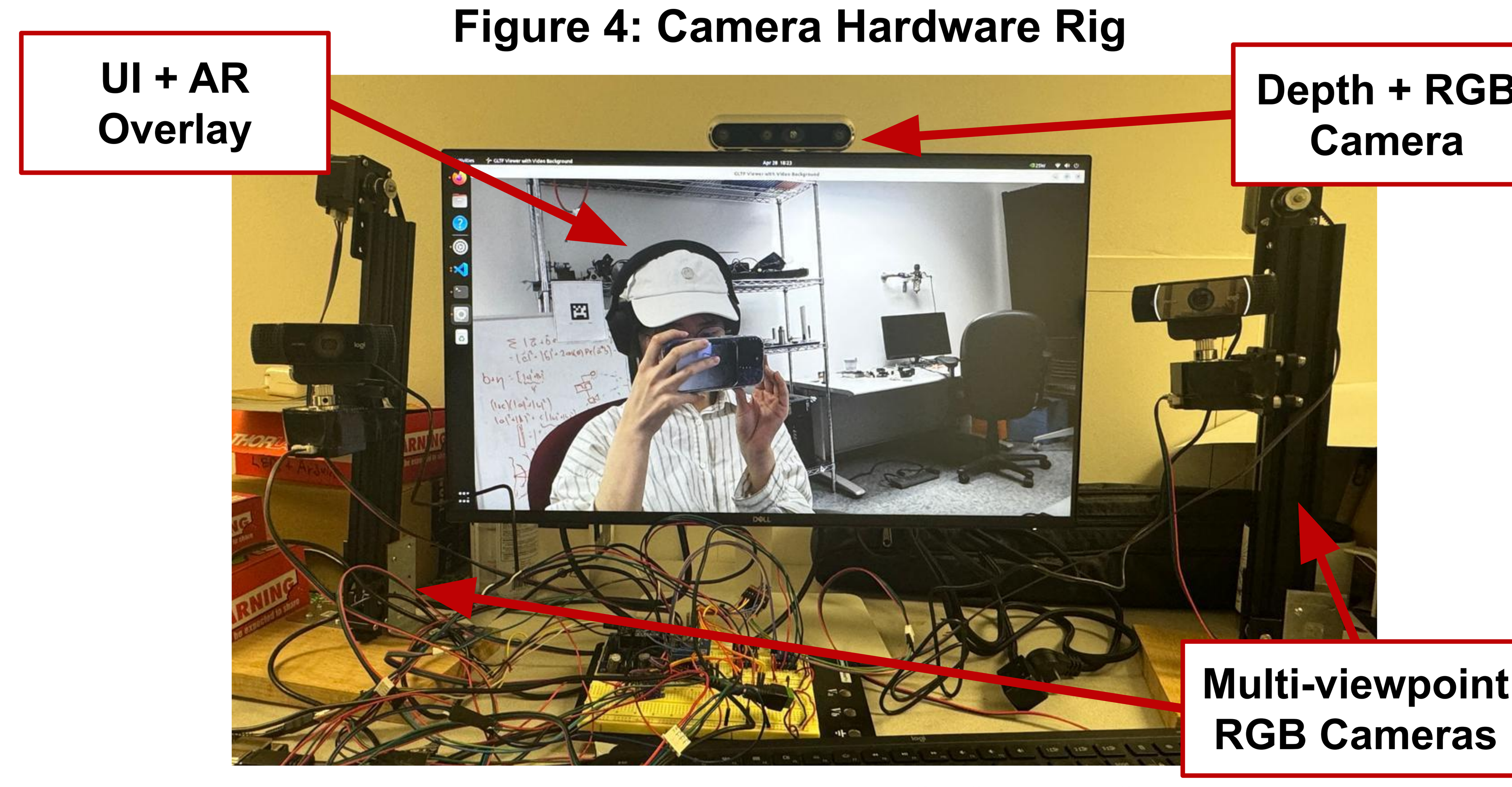

Figure 5: Full Hardware + Software Setup

System Evaluation

Use-Case Requirements:

Metric	Target	Actual
Gesture input latency	≤ 200 ms	**55 ms**
Gesture input accuracy	≥ 90% correct	**95% correct**
AR filter rendering framerate	≥ 15 FPS	**160 FPS**
AR filter drift range	≤ 5 px	**3-5 px**
Face model generation latency	≤ 50 ms	**20 ms**
Head pose estimation error	≤ 5 px	**10-15 px**
Head pose estimation latency	≤ 150 ms	**2 ms**
Camera control precision (rotation)	≤ 5 deg/step	**3.7 deg/step**
Camera control precision (translate)	≤ 0.05 in/step	**0.031 in/step**
Camera control range (rotation)	≥ 90 deg	**180 deg**
Camera control range (translate)	≈ 11.8 in	**10-11 in**

Trade-offs:

- Gesture algorithm:
  - **Position-based inputs** (hand location on screen) vs. **Velocity-based** inputs (velocity of hand).
  - Chose **position-based inputs**, for **superior accuracy, robustness, and ease of use** in a UI.
- Camera control system:
  - **Speed** (faster motor) vs. **Precision** (better accuracy).
  - Chose **precision,** for **superior accuracy, reliability of operation, lower power consumption.**
- AR overlay detection backend:
  - **dlib** vs. **OpenCV DNN + LBF** vs. **MediaPipe Face Mesh**.
  - Chose **OpenCV DNN** for balance of **ease of integration** and face model **latency/robustness.**

Electrical & Computer ENGINEERING
Carnegie Mellon