EyeCoD: Eye Tracking System Acceleration via FlatCam-based Algorithm & Accelerator Co-Design

H. You[1], C. Wan[1], Y. Zhao[1], Z. Yu[1], Y. Fu[1], J. Yuan[2], S. Wu[2], S. Zhang[2], Y. Zhang[1], C. Li[1], V. Boominathan[2], A. Veeraraghavan[2], Z. Li[3], and Y. Lin[1]

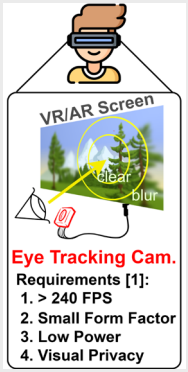
[1] Georgia Institute of Technology, SCS [2] Rice University, ECE [3] Meta

Georgia Tech College of Computing
Center for Research into Novel Computing Hierarchies

ISCA'22
IEEE Micro Top Picks'23

Background & Motivation

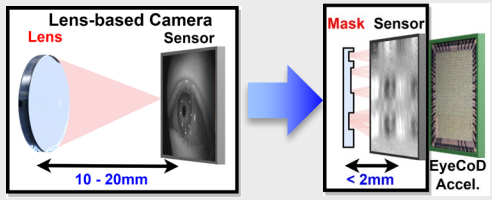
- **Eye tracking** is an essential human-machine interface modality in AR/VR [1]



- **Challenges for eye tracking in AR/VR** [1]:
  - >240 FPS
  - Small form factor
  - Power consumption in mW
  - Visual privacy
- **Existing works** [2,3]:
  - ☹ One order of magnitude slower
  - ☹ Large form factor and low visual privacy due to the adopted lens-based cameras

Unexplored Opportunities for Eye Tracking?

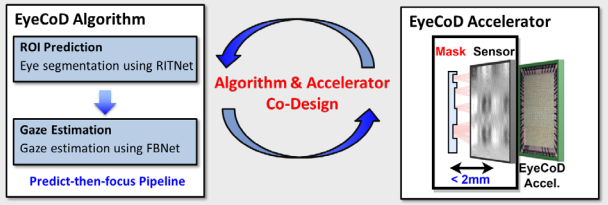
- **Can we build a lensless eye tracking system?**
  - A Lensless camera, i.e., FlatCam [4]
    - ☺ Small form factor, i.e., 5×~10× thinner
  - A dedicated accelerator featuring algorithm and accelerator co-design
    - ☺ >240 FPS
    - ☺ mW power consumption



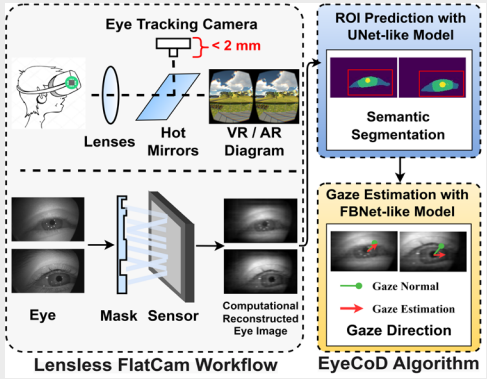
[1] C. Liu, et. al., IDEM'21 [2] Y. Feng, et. al., IEEE VR'22 [3] K Bong, et. al., VLSI'15
[4] M. Asif, et. al., TCI'17 [5] P. Crisiina, et. al., Sensors'21 [6] EdgeCPU (Raspberry Pi)
[7] CPU (AMD EPYC 7742) [8] EdgeGPU (Nvidia Jetson TX2) [9] GPU (Nvidia 2080Ti)

Proposed EyeCoD System

- **FlatCam-based algorithm & accelerator co-designed system (EyeCoD),** incorporating:
  - ☺ Sensing-processing interface
  - ☺ Predict-then-focus alg. pipeline
  - ☺ Dedicated AI accelerator

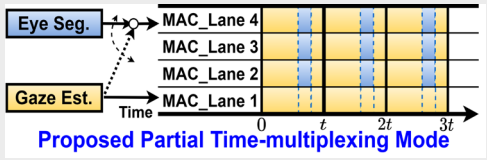


EyeCoD Algorithm



- **Predict-then-focus pipeline:**
  - Stage 1: Image reconstruction
    - ☺ Sensing-processing interface replaces FlatCam sensing & model first layer with coded masks
  - Stage 2: Region-of-interest (ROI) prediction
    - ☺ Predict and crop the most informative core eye area, i.e., ROI
    - ☺ Once per 50 frames
  - Stage 3: Gaze estimation
    - ☺ Estimate the gaze direction based on ROIs
    - ☺ Processed for each frame

EyeCoD Accelerator

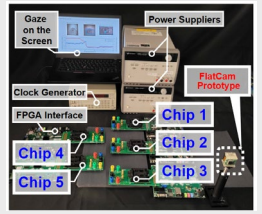


- ☺ Balance the diff. execution frequencies of ROI prediction and gaze estimation
- ☺ 2.31× speed up over time-multiplexing mode; 1.6× higher energy eff. over concurrent mode

- **EyeCoD accelerator features:**
  - Partial time-multiplexing mode for workload orchestration
  - Intra-channel reuse for depth-wise convolutional layers (DW)
    - ☺ Reduce 71% of DW's latency
  - Activation partition and memory access parallelism
    - ☺ Save 36% activation memory
    - ☺ Save 50~60% activation bandwidth

Evaluation Results

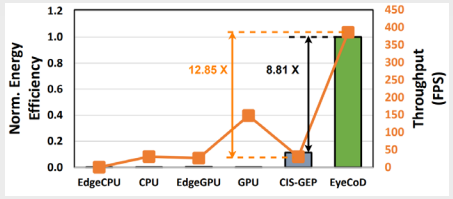
- **Evaluation setups**
  - Datasets
    - Industry-standard datasets: OpenEDS'19 & OpenEDS'20
  - Metrics
    - Gaze estimation accuracy, Throughput, Energy efficiency



  - EyeCoD accelerator
    - Silicon prototype in a 28nm technology

Act GB0/GB1	Weight Buffer0/1	Weight GB	Index SRAM	Instr. SRAM
512KB * 2	64KB * 2	512KB	20KB	4KB
MAC Lanes	**MACs/MAC Lane**	**Area**	**Clock frequency**	**Power**
128	8	8 mm^2	370MHz	335mW

- **EyeCoD evaluation results**
  - EyeCoD algorithm's accuracy matching the winner in the industry-led challenge [5]
  - EyeCoD accelerator over CPU/GPU platforms
    - 2966×, 12.7×, 14.8×, and 2.61× throughput improvements over EdgeCPU [6], CPU [7], EdgeGPU [8], and GPU [9]
  - EyeCoD over state-of-the-art eye tracking accelerators
    - 12.8× throughput improvement and 8.1× higher energy efficiency over CIS-GEP[4], respectively



Georgia Tech