Event Crowd Flow & Wayfinding — Objective & Evaluation (Hacka

Adaptive edge-only routing to reduce hallway congestion using a Snapdragon PC + Android phones

Problem We're Solving

Indoor events (campuses, conferences, fairs) suffer chokepoints where people bunch up. Existing fixes—static signage, ushers, or cloud analytics—are slow, not adaptive, or privacy-unfriendly. We need a real-time, privacy-preserving method to keep people moving smoothly without sending video to the cloud.

Objective (MVP)

Continuously estimate corridor congestion from a single camera on a Snapdragon PC and, in real time, route users on their phones along less crowded paths—edge-only and low-latency—so the average travel time decreases versus a static shortest path.

System Overview

Inputs: Overhead camera (PC only), hand-made building graph (nodes/intersections; edges/segments), optional phone localization (QR/node buttons). Processing (PC): on-edge person detection + tracking → per-segment density/flow → congestion score in [0,1]. Output: Phones subscribe over LAN to receive live, congestion-aware routes and optional stagger haptics. Privacy: no raw frames off device; only counts, scores, and segment IDs shared.

Routing Cost (what we optimize)

At time t, choose path P minimizing: $\text{Cost}(P,t) = \Sigma_{e \in P} [\ \text{length\_m}(e) \times (1 + k \times \text{score}(e,t))\ ]$ with $k \approx 2.0$ (tunable).

Success Criteria (measured in 36 hours)

- Latency: camera → updated route on phone < 300 ms median.
- Accuracy: corridor count MAE ≤ 2 people over 30–60 s windows.
- Impact: median travel time ↓ ≥ 10–15% vs. static shortest path in a staged crowd demo.
- Privacy/Bandwidth: 0 video off-device; < 10 KB/s per client.
- Stability: run 5–10 minutes without FPS/thermal collapse.

MVP Scope

In scope: one camera, a few corridor segments; on-edge detection+tracking; phone client for destination/route/haptics.
Out of scope (hackathon): full indoor positioning/SLAM, multi-camera calibration, identity tracking, cloud video analytics.

Baseline & Proof You Solved It

Baseline A: everyone follows static shortest path.
Your system: routes adapt to congestion spikes (you provoke a mini-jam). Report side-by-side med/95p travel time, reroute latency, bandwidth, and a short privacy note.

Metrics & Logging (CSV schema)

Primary: density-error (MAE/MAPE), reroute latency, user detour time.
Secondary: bandwidth, energy/thermals.
Schema examples:
ts,op,seg_id,count,density,score,lambda_in,lambda_out,fps,cpu,temp,bw
ts,op,route_recompute,from,to,chosen_path,alt_gap,latency_ms
ts,op,user_arrive,dest,route,travel_time_s

Risks & Mitigations

• Indoor GPS unreliable → use QR/node buttons to localize.
• Detector too slow → smaller input, INT8, frame skip; fallback to line-crossing counts for inflow/outflow.
• Noisy counts → EMA smoothing + combine speed/flow, not only raw counts.

2-Minute Demo Script

1) Show camera view + live heatmap on PC.
2) Trigger a micro-jam (3–5 people crowd a segment).
3) Segment score spikes; phone recomputes route to alternate corridor.
4) Walk with phone cues; feel stagger haptic if surge persists.
5) Toggle privacy view showing only segment scores (no video off-device).

Implementation Snapshot (for reference)

PC: ONNX Runtime + DirectML (YOLOv8n/SSD-MobileNet INT8), tracker (BYTE/IOU+Kalman), 1 Hz segment stats + EMA smoothing, WebSocket server.
Phone: WebSocket client, Dijkstra over tiny graph, edge weight = length × (1 + k × score), haptics for turn/stagger.