Demo: Real-Time Video Analytics for Urban Safety, Deployment over Edge and End Devices

Mahshid Ghasemi[1], Yongjie Fu[2], Xinyu Ouyang[1], Peiran Wang[1], Mehmet Kerem Turkcan[1], Jhonatan Tavori[3], Sofia Kleisarchaki[4], Thomas Calmant[4], Levent Gürgen[4], Zoran Kostic[1], Xuan Sharon Di[2], Gil Zussman[1], Javad Ghaderi[1]

[1]Electrical Engineering, Columbia University, [2]Civil Engineering, Columbia University, [3]Computer Science, Tel Aviv University, [4]Kentyou

{mahshid.ghasemi,yf2578,xo2122,peiran.wang,mkt2126,zk2172,sharon.di,gil.zussman,jg3465}@columbia.edu, jhonatant@mail.tau.ac.il,{sofia.kleisarchaki,thomas.calmant,levent}@kentyou.com

Abstract

We showcase the workflow of PAVE (Pedestrian Awareness Via Edge analytics), a scalable system for real-time video analytics that leverages street cameras to improve pedestrians' safety while maintaining their privacy. PAVE distributes computation across edge servers and end-user mobile devices. Cameras' live streams are processed at the edge to forecast vehicles' trajectories and detect danger zones. Pedestrians' mobile devices then locally determine if the user is inside a danger zone and trigger timely alerts via a custom iOS app. In addition, anonymized metadata, such as pedestrian and vehicle positions, speeds, and directions, are aggregated and displayed on a public map for broader situational awareness. We evaluated PAVE's performance through implementation on the NSF COSMOS testbed's edge server while processing real-time video stream from cameras in diverse urban environments. Live field tests at an intersection in New York City show that PAVE can alert at-risk pedestrians about 0.9 s before a vehicle reaches them. With low-latency cameras, this lead time extends to around 1.6 s which is within the 1-2 s window pedestrians typically need to react.

CCS Concepts

• **Computing methodologies → Distributed computing methodologies**; • **Computer systems organization → Real-time system architecture**.

SEC '25, Arlington, VA, USA

ACM ISBN 979-8-4007-2238-7/2025/12
https://doi.org/10.1145/3769102.3774387

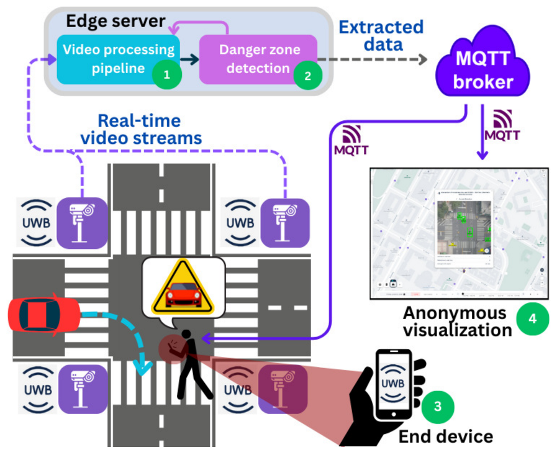

Figure 1: PAVE workflow.

Keywords

Video analytics, edge computing, testbed deployment.

ACM Reference Format:
Mahshid Ghasemi[1], Yongjie Fu[2], Xinyu Ouyang[1], Peiran Wang[1], Mehmet Kerem Turkcan[1], Jhonatan Tavori[3], Sofia Kleisarchaki[4], Thomas Calmant[4], Levent Gürgen[4], Zoran Kostic[1], Xuan Sharon Di[2], Gil Zussman[1], Javad Ghaderi[1] . 2025. Demo: Real-Time Video Analytics for Urban Safety, Deployment over Edge and End Devices. In *The Tenth ACM/IEEE Symposium on Edge Computing (SEC '25), December 3–6, 2025, Arlington, VA, USA*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3769102.3774387

1 Introduction

In this work, we study how real-time video analytics can be used to enhance road users' safety while preserving their privacy. We present PAVE (**P**edestrian **A**wareness **V**ia **E**dge analytics), which is based on our recent paper [5]. PAVE is a scalable video analytics system (illustrated in Fig. 1) distributed across edge servers and end-users' devices (e.g., iPhones). It processes live video streams from traffic cameras to determine if there are any pedestrians in a dangerous area where a vehicle is approaching and triggers timely alerts on their mobile devices. Additionally, extracted traffic/crowd-related metadata are aggregated and visualized anonymously through an interactive map developed for both real-time and historical monitoring. Some of the **challenges** encountered during the development of such a system for time-sensitive and safety-critical applications include:

Components integration complexities: A video analytics pipeline consists of several components through which data flows continuously (illustrated in Fig. 2). Each component contributes to the end-to-end latency of the pipeline and therefore should be designed and configured carefully to balance accuracy and latency. These components must be integrated in a way that minimizes data transfer overheads between processing stages (e.g., CPU-GPU memory copies, and cross-network transfers in distributed systems) and queuing delays from mismatched component throughputs. This is necessary to ensure low-latency processing.

Efficient memory management: Processing live video streams demands robust memory (de)allocation strategies. Without careful management, memory fragmentation [8] and garbage collection [2, 9] overheads can cause unpredictable delays and degrade performance.

In this work, we mainly focused on designing system architecture and optimizations that enable existing vision models to be deployed effectively in real-world latency-critical applications. PAVE integrates ubiquitous street cameras, Message Queuing Telemetry Transport (MQTT) communication [12, 13], and Ultra-Wideband (UWB)-based localization to alert at-risk pedestrians before danger reaches them. PAVE preserves pedestrians' privacy by only sending danger zones' data to their phone (if the app is installed) and each phone determines locally—without sharing any data with the edge server—whether the user is entering a danger zone and triggers an alert if necessary. To reduce computational and network overhead and improve scalability, PAVE processes video streams at low resolution to detect vehicles (large objects). Only when potential danger zones are detected, it performs selective Region of Interest (ROI) processing to determine if a pedestrian (small objects) is inside those zones. This targeted ROI processing boosts mean Average Precision (mAP) by ~**50%** without the high GPU and bandwidth costs of continuous full high-resolution processing.

Prior work has applied real-time video analytics to urban safety and smart cities. For instance, [6, 7] process live feeds for traffic insights, while [4] detects collisions at intersections. GUTS [1] tracks road users in world coordinates, StreetNav [11] guides low-vision users, and UTS [3] tracks vehicles using a Kalman Filter (KF) based motion model. SafeCross [17] monitors blind zones in left turns to issue warnings. Compared to previous studies, this work is focused on operational feasibility and scalability of the end-to-end system with a specific emphasis on latency. Each component is specifically designed and configured to achieve reasonable accuracy while maintaining low latency and deployment cost. We demonstrate the workflow of the end-to-end alert system and the interactive map.

2 System Workflow and Field Test

As illustrated in Figs. 1 and 2, PAVE's real-time workflow consists of the following steps:

(1) Video Processing Pipeline (light blue box): Real-Time Streaming Protocol (RTSP) video streams are transmitted to edge servers, decoded into raw frames, and processed by object detection (e.g., YOLOv8 [16]) and tracking (e.g., NVIDIA DCF) models. The pipeline outputs object bounding boxes, labels (pedestrian, car, truck), and tracking IDs, which are converted to a unified top-view to estimate vehicle and pedestrian speeds and directions.

(2) Danger Zone Detection (light purple): Vehicles' future trajectories are predicted using a KF-based model to identify potential danger zones near crosswalks where pedestrians might be at risk. Identified danger zones trigger a targeted ROI analysis within the video processing pipeline to detect pedestrian presence with high precision. If pedestrians are detected, danger zone information is communicated through MQTT protocol to mobile devices via the iOS app.

(3) End Device (iOS App): The mobile app determines whether the pedestrian is (going) inside an identified danger zone using UWB-based localization, subsequently issuing timely alerts, if necessary.

(4) Visualization Dashboard: Extracted metadata, including pedestrian/vehicle locations, speeds, and directions, are continuously sent to a visualization map via the MQTT protocol for anonymized display. This map, which is publicly accessible, presents only moving bounding boxes and associated metadata, without exposing identifiable images of pedestrians or vehicles.

Field test. We conducted a field test on the COSMOS testbed [15] using two street-level cameras viewing an intersection in New York City (NYC), as shown in Fig. 3[1]. We measured how fast PAVE can alert pedestrians at risk before vehicles reach the danger zone. For safety, we programmed an iPhone to locate itself inside a danger zone while held on the sidewalk. We then recorded the time gap between when the iPhone received an alert and when the vehicle reached the identified danger zone (i.e., time to react). We repeated this test **24** times, and the results show that reaction time ranges from **0.2 s** to **0.94 s**, with an average of ~**0.6 s**. Since our cameras add ~**0.7 s** latency according to our measurements, using low-latency cameras would enable PAVE to

[1]Our use of COSMOS' live and recorded video streams was designated IRB-exempt by Columbia University.

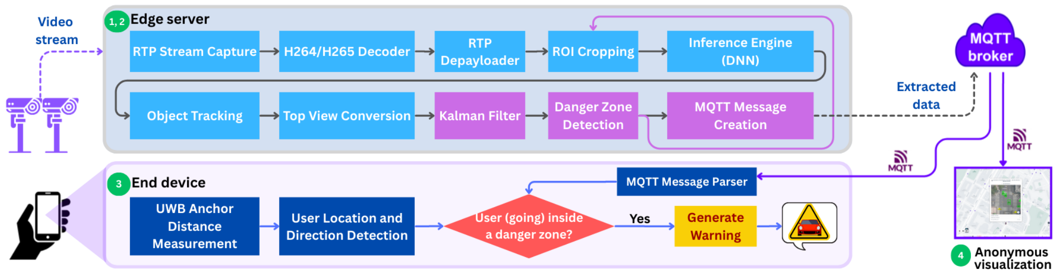

Figure 2: PAVE's end-to-end components deployed across the edge and end devices and its anonymized public map.

Figure 3: Cameras deployed on 1st and 2nd floor of a building and their view of an intersection in NYC.

achieve the **1 to 2 seconds** of reaction time required for pedestrians to move to safety, as reported by [10, 14].

3 Demonstration

We showcase the workflow of the end-to-end alert system as well as the interactive map using COSMOS testbed's street cameras in NYC. This demonstration highlights how live video streams are processed on the edge server, how alerts are generated and delivered to mobile devices, and how traffic and pedestrian activity are visualized anonymously through the interactive map.

Acknowledgments

This work was supported in part by NSF and Center for Smart Streetscapes (CS3) under NSF Cooperative Agreement EEC-2133516, NSF grant CNS-2038984 and corresponding support from the Federal Highway Administration (FHWA), NSF grant CNS-2148128 and by funds from federal agency and industry partners as specified in the Resilient & Intelligent NextG Systems (RINGS) program, NSF grant CNS-2450567, and by computing resources provided by the NVIDIA Academic Grant Program and the Empire AI Consortium.

References

[1] Martin Ahrnbom, Ivar Persson, Håkan Ardö, and Mikael Nilsson. 2022. Generalized urban traffic surveillance (GUTS): World-coordinate tracking for traffic safety applications. In *Proc. IEEE ITSC*.

[2] Emery D Berger, Sam Stern, and Juan Altmayer Pizzorno. 2023. Triangulating python performance issues with SCALENE. In *USENIX OSDI*.

[3] Henry Bradler, Adrian Kretz, and Rudolf Mester. 2021. Urban Traffic Surveillance (UTS): A fully probabilistic 3D tracking approach based on 2D detections. In *Proc. IEEE IV*.

[4] Hadi Ghahremannezhad, Hang Shi, and Chengjun Liu. 2022. Real-time accident detection in traffic surveillance using deep learning. In *Proc. IEEE IST*.

[5] Mahshid Ghasemi, Y. Fu, X. Ouyang, P. Wang, M. Turkcan, J. Tavori, S. Kleisarchaki, T. Calmant, L. Gürgen, Z. Kostic, X. Di, Gil Zussman, and J. Ghaderi. 2025. Real-time video analytics for urban safety: Deployment over edge and end devices. In *Proc. ACM/IEEE SEC*.

[6] Mahshid Ghasemi, Sofia Kleisarchaki, Thomas Calmant, Levent Gürgen, Javad Ghaderi, Zoran Kostic, and Gil Zussman. 2022. Real-time camera analytics for enhancing traffic intersection safety. In *Proc. ACM MobiSys*.

[7] Mahshid Ghasemi, Sofia Kleisarchaki, Thomas Calmant, Jiawei Lu, Shivam Ojha, Zoran Kostic, Levent Gürgen, Gil Zussman, and Javad Ghaderi. 2023. Real-time multi-camera analytics for traffic information extraction and visualization. In *Proc. IEEE PerCom Workshops*.

[8] Jörg Herter. 2014. *Timing-predictable memory allocation in hard real-time systems*.

[9] Matthew Hertz and Emery D Berger. 2005. Quantifying the performance of garbage collection vs. explicit memory management. In *Proc. ACM SIGPLAN OOPSLA*.

[10] Alireza Jafari and Yen-Chen Liu. 2024. Pedestrians' safety using projected time-to-collision to electric scooters. *Nat. commun.* 15, 1 (2024), 5701.

[11] Gaurav Jain, Basel Hindi, Zihao Zhang, Koushik Srinivasula, Mingyu Xie, Mahshid Ghasemi, Daniel Weiner, Sophie Ana Paris, Xin Yi Therese Xu, Michael Malcolm, et al. 2024. StreetNav: Leveraging street cameras to support precise outdoor navigation for blind pedestrians. In *Proc. ACM UIST*.

[12] R. A. Light. 2017. Mosquitto: Server and client implementation of the MQTT protocol. *J. Open Source Software* 2, 13 (2017), 265.

[13] Biswajeeban Mishra and Attila Kertesz. 2020. The use of MQTT in M2M and IoT systems: A survey. *IEEE Access* 8 (2020), 201071–201086.

[14] Bingbing Nie, Quan Li, Shun Gan, Bobin Xing, Yuan Huang, and Shengbo Eben Li. 2021. Safety envelope of pedestrians upon motor vehicle conflicts identified via active avoidance behaviour. *Sci. Rep.* 11, 1 (2021), 3996.

[15] D. Raychaudhuri, I. Seskar, G. Zussman, T. Korakis, D. Kilper, T. Chen, J. Kolodziejski, M. Sherman, Z. Kostic, X. Gu, H. Krishnaswamy, S. Maheshwari, P. Skrimponis, and C. Gutterman. 2020. Challenge: COSMOS: A city-scale programmable testbed for experimentation with advanced wireless. In *Proc. ACM MobiCom*.

[16] Juan Terven and Diana Cordova-Esparza. 2023. A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond. *arXiv preprint arXiv:2304.00501* (2023).

[17] Baofu Wu, Yuankai He, Zheng Dong, Jian Wan, Jilin Zhang, and Weisong Shi. 2022. To turn or not to turn, SafeCross is the answer. In *Proc. IEEE ICDCS*.