A Privacy-preserving Video Processing Pipeline

Gábor György Gulyás, Gergely Erdődi
Vitarex Stúdió Ltd
Budapest, Hungary
emails: gabor@gulyas.info, erdodi.gergely@vitarex.hu

***Abstract*—We present a modular and scalable video analytics system designed for object detection, face recognition, and multi-camera tracking, with minimal bandwidth consumption and full compatibility with existing video surveillance infrastructure. The architecture emphasizes cost efficiency and regulatory compliance, operating primarily on on-premise deployments to align with constraints imposed by the Artificial Intelligence Act of the European Union and General Data Protection Regulation. After benchmarking a range of object detection, face analysis, and tracking models, we selected the most performant and efficient solutions and orchestrated them using Apache Airflow. The system executes a graph-based processing pipeline that supports parallel, per-camera analytics including people counting, path tracking, heatmap generation, and geofencing. Results are visualized through Apache Superset dashboards, enabling interactive, building-wide situational awareness. By leveraging open source components and a containerized, Kubernetes-compatible deployment model, the solution provides real-time, bandwidth-aware analytics with strong adaptability to diverse operational environments, supporting data-driven decision-making across sectors, such as retail, logistics, and smart infrastructure.**

Keywords-CCTV; video processing; face recognition; architecture.

I. Introduction

The proliferation of video surveillance systems in public and private domains has led to an unprecedented volume of visual data being generated every day. Yet, much of this data remains underutilized, as traditional Closed-Circuit Television (CCTV) infrastructures are designed primarily for passive recording rather than intelligent, real-time interpretation. In response, there is a growing demand across sectors—from retail, logistics and critical infrastructure—for plug-and-play analytics capabilities that can extract actionable insights from video streams without overhauling existing systems for which an example is provided in Figure 1: such systems can log customer activites and map them over the floorplan of the store for providing analytics.

Scalable and bandwidth-efficient video analytics is especially critical in environments where network infrastructure is limited or distributed across multiple physical sites. For organizations managing tens or hundreds of camera feeds, the ability to perform on-device or near-edge inference significantly reduces the load on central servers and minimizes data transfer costs. However, implementing such systems presents numerous challenges.

First, many deployments rely on legacy CCTV hardware that lacks the compute resources necessary for running modern deep learning models. Second, network constraints often prevent continuous high-resolution streaming, which complicates even somewhat real-time inference and analytics. Third, different use cases — such as people counting, heatmap generation, path tracking, geofencing, and facial recognition — require distinct models and processing pipelines, all of which must coexist within the same system. Finally, strict privacy and security regulations, such as the General Data Protection Regulation (GDPR) [1] and the European Union's Artificial Intelligence Act (AI Act) [2], impose legal constraints on how biometric and behavioral data can be processed, stored, and transmitted.

Figure 1. An example how we could use CCTV to map real-world activities into analytics.

To address these challenges, we present a modular and scalable analytics pipeline designed to operate on CPU (Central Processing Unit)- or GPU (Central Processing Unit)-based systems with minimal impact on existing CCTV infrastructure. The system supports multiple analytics tasks concurrently, including object detection, face recognition, activity monitoring, and crowd flow analysis, while maintaining compliance with data protection laws. Our architecture emphasizes deployment flexibility, bandwidth-aware processing, and robust orchestration.

This paper is structured as follows. Section II details the system architecture and deployment design. Then, in Section III, we provide details on the implemented analytics tasks and visualization methods. In Section IV, we discuss our implementation on multi-camera multi-object tracking. Finally, Section V concludes our work.

II. Architecture Details

Designing the architecture of our video analytics system required balancing multiple constraints, most notably cost-efficiency and regulatory compliance. GPU-enabled cloud infrastructures offer high performance but come with substantial operational costs, making them unsuitable for continuous, large-scale video processing. As such, we prioritized solutions that could run efficiently on CPU-only setups, both to reduce cost

and to allow greater deployment flexibility (even though we also used GPU).

Another key architectural decision concerned the mode of operation: whether to process video streams remotely in the cloud or locally on-premise. In addition to cost considerations, legal and ethical concerns — particularly those stemming from the EU AI Act — played a decisive role. Since the Act prohibits biometric identification (such as face recognition) in many public settings unless explicitly authorized, we opted for on-premise pre-processing to ensure compliance and to maintain full control over sensitive data.

Our first step was researching different models and frameworks available for object detection, tracking, and face detection/recognition. We tested a variety of approaches, comparing their accuracy and performance to determine the best fit for our needs. This evaluation included both traditional machine learning techniques and modern deep learning-based models.

Once we identified the most suitable models, we focused on integrating them into a functional pipeline. To efficiently manage workflows, we chose Apache Airflow [3] as our orchestration tool. Airflow allowed us to automate and schedule the processing steps, ensuring seamless data flow and model execution across the system.

A. Model Selection

To select the most suitable components for our video analytics pipeline, we conducted a thorough benchmarking process across four key tasks: object detection, face detection, face recognition, and multi-object tracking. Our evaluation focused on four primary criteria: detection accuracy, inference speed and ease of integration. Additionally, we prioritized models released under permissive licenses such as MIT (created at the Massachusetts Institute of Technology) to ensure freedom for modification, commercial use, and to avoid potential legal or financial restrictions.

Object Detection. We evaluated several state-of-the-art object detectors, including YOLOv8 [4] (YOLO in general stands for You Look Only Once), YOLOX [5], EfficientDet [6], and Detectron2 [7]. These models were tested using benchmark datasets, such as COCO (Common Objects in Context) and custom video streams relevant to our target use cases. YOLOv8 provided high accuracy and very fast inference, especially when optimized with TensorRT, with moderate integration effort and an Apache 2.0 license. YOLOX offered similar accuracy, slightly lower inference speed, easier integration, and the same license, making it the preferred choice**.**

Face Detection. For face detection, we compared MediaPipe Face Detection [8], YuNet (a lightweight detector based on NPU-ready backbones, where NPU stands for Neural Processing Units) [9], and MTCNN [10]. MediaPipe was the most resource-efficient and easy to deploy on CPU-based systems. YuNet offered a compelling balance of accuracy and performance, with good hardware compatibility. It was also the fastest and most lightweight, and its compatibility with CUDA hardware acceleration made it the best choice for our use case.

Face Recognition. In the face recognition domain, we benchmarked InsightFace [11], SFace [12], and FaceNet [13]. InsightFace, based on ArcFace embeddings, demonstrated superior robustness and accuracy in identity verification tasks, supported by comprehensive pre-trained models and broad platform compatibility. However, due to its restrictive licensing, which does not include an MIT-equivalent license, FaceNet was selected as the preferred alternative. FaceNet offers comparable performance with greater configurability and is distributed under an MIT license, making it more suitable for integration within the system.

Tracking. For multi-object tracking, we tested DeepSORT [14] and ByteTrack [15]. While DeepSORT has been widely adopted in academic and commercial applications, ByteTrack showed superior performance in crowded scenes due to its effective association of low-score detections, resulting in fewer identity switches and more stable trajectories. The Kalman filter employed by ByteTrack also provided better speed than DeepSORT.

Inference Optimization. To achieve low-latency processing, we integrated NVIDIA TensorRT [16] for inference acceleration. This significantly reduced the runtime of deep models, particularly for object detection and face recognition tasks, enabling high performance even on edge devices with limited resources.

B. Architectural Setup

Our architecture is illustrated in Figure 2, and we discuss its details in the following paragraphs. There are two main components: one is the on-premise preprocessing unit (dealing with computational heavy tasks), and the other is to display results and statistics.

Pipeline Design and Orchestration. The system employs Apache Airflow as the central orchestration engine, organizing the entire video analytics pipeline into modular, interdependent tasks through Directed Acyclic Graphs (DAGs). The primary DAG coordinates seven parallel processing tasks including heatmap generation, path tracking, people counting, geofencing, floor plan transformation, activity detection, and face analysis. Each task operates independently, but shares data through Airflow's XCom mechanism, enabling efficient parallel processing while maintaining data consistency across the pipeline. The entire Airflow service runs within a dedicated Docker container [17], while the scheduler operates separately in its own container (within the on-premise device). To support scaling requirements, the system is compatible with Kubernetes and Helm charts, allowing flexible deployment and management in cloud or cluster environments. This setup ensures environment consistency and simplifies deployment, allowing individual components to be modified, scaled, or replaced without impacting the overall system architecture.

Video Processing and Inference Pipeline. The video processing architecture employs multithreaded frame sampling with configurable frequency to balance processing speed with tracking accuracy. Each video undergoes systematic frame extraction, where frames are distributed across consumer

threads for parallel inference using YOLOX object detection. To optimize performance further, we used OpenCV [18] built with NVIDIA VIDEO CODEC support for hardware-accelerated decoding, which significantly reduces CPU load and improves frame reading speed. The system also maintains separate processing queues with bounded capacity to prevent memory overflow during high-throughput scenarios. Feature extraction operates concurrently with detection, utilizing specialized models for face detection (YuNet), face recognition (FaceNet), and activity estimation when enabled, with results aggregated into tracking histories.

Camera Integration and Data Sources. The system operates across multiple network tiers with PostgreSQL [19] serving as the central data repository, while the Airflow scheduler manages task distribution. Video data flows from an ISAPI (Internet Server Application Programming Interface) enabled [20] NVR (Network Video Recorder) through a dedicated scheduler service that continuously monitors recording queues and triggers Airflow processing workflows.

The scheduler service has zone-specific configuration parameters including boundary lines, transformation matrices, and processing frequencies. Video downloads are managed through authenticated sessions with automatic retry mechanisms and status tracking in the database, ensuring reliable data acquisition even under network instability or camera downtime conditions. This system ensures robust error handling and monitoring through transactional safeguards, graceful thread shutdown, automated recovery, and task-level visibility via Airflow, which also provides built-in retry and alert mechanisms.

C. System Evaluation

The current deployment runs on a single machine equipped with an NVIDIA RTX 4060 GPU, 16 GB of RAM, and an Intel Core i5-13400F CPU. Benchmark tests indicate that one hour of Full HD video can be processed in approximately 100 seconds using the GPU, whereas CPU-only processing requires 3.8 hours. At this rate, the system can process up to 72 camera streams per day with GPU acceleration—assuming each camera records 12 hours of footage—compared to 0.5 streams per day using the CPU alone. This throughput is achieved under continuous operation without parallel GPU saturation, providing a reliable baseline for scalability. Further gains can be realized by distributing workloads across multiple GPUs or nodes via the existing Kubernetes-compatible architecture.

Existing open-source tools provide only partial overlaps with this functionality. Kerberos.io is a lightweight, Docker-deployable platform focused primarily on motion detection, with limited AI-based analytics achievable through external integrations. ZoneMinder is designed for recording and basic motion detection, with optional analytics via plugins. In contrast, the proposed pipeline natively integrates object detection, multi-target tracking, optional face recognition, people counting, and heatmap generation (both per-camera and layout-based), while offering a modern analytics-focused web interface and scalable deployment through Docker or Kubernetes.

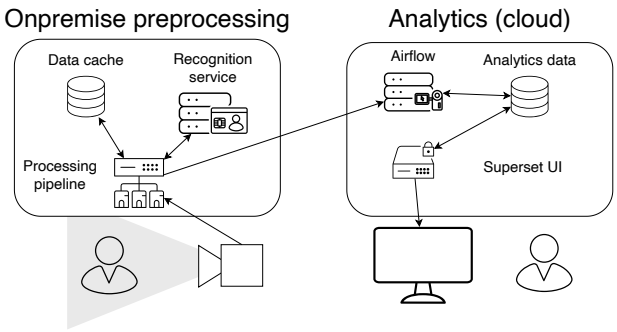

Figure 2. Our architecture setup.

III. Analytics and Visualization

After setting up Airflow, we needed a way to visualize and interact with the results. For this, we chose Apache Superset, an open source business intelligence tool. Superset enabled us to create interactive dashboards, providing valuable insights from our data and model outputs.

The platform implements a suite of analytic functions (running in the cloud) that transform raw surveillance data into actionable intelligence. **Density analysis** employs Gaussian accumulator matrices [21] with adaptive kernel parameters to generate movement heatmaps that reveal high-traffic zones and pedestrian flow patterns across the monitored environment. **Trajectory analysis** utilizes perspective transformation to create unified coordinate systems that enable cross-camera path tracking, with Bézier curve interpolation providing smooth trajectory visualization that facilitates pattern recognition and anomaly detection.

The system also delivers behavioral analytics through **geofencing** that monitors zone-specific activity and transition events, and **activity recognition** to identify task-specific behaviors, such as object manipulation and stationary activities. (**Demographic analysis** uses facial recognition models to provide age, gender, and ethnicity distribution insights with statistical confidence metrics.)

Visualization outputs encompass multiple analytical modalities including images like **density heatmaps** and **trajectory overlays** with color-coded pathways representing individual movement patterns, and **statistical dashboards** displaying temporal trends through bar charts, line graphs, and occupancy histograms (see example in Figure 3). The platform generates **cumulative analytics** with configurable temporal windows, supporting multi-scale analysis from minute-by-minute monitoring to long-term behavioral pattern identification. Integration with Apache Superset enables interactive dashboard creation with drill-down capabilities, cross-filtering, and automated report generation, providing data-driven insights for operational optimization and security enhancement.

IV. Multi-Camera Multi-Object Tracking

To further enhance evaluation of tracking data, we incorporated Multi-Camera Multi-Object Tracking (MCMOT),

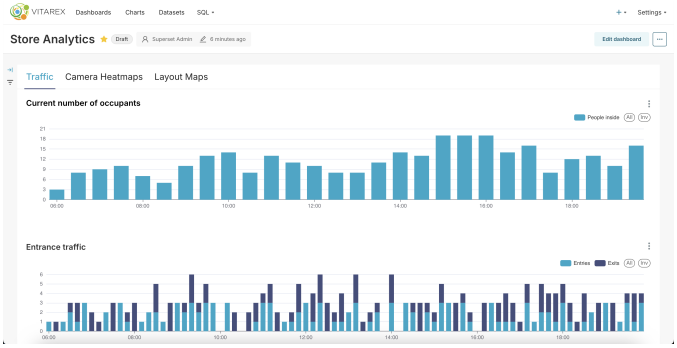

Figure 3. An example from our Apache Superset dashboard.

allowing us to follow individuals across multiple camera feeds. Additionally, we developed a unified layout map that extends per-camera analytics to an entire building. Using projective geometry, we map detections from different cameras onto a real-world floor plan, enabling heatmapping, path tracking, and people counting at a global level. This holistic approach provides a comprehensive view of movement patterns and occupancy trends, further improving surveillance and analytics capabilities.

The tracking system employs **BYTETracker** as the foundation for single-camera object tracking, which maintains temporal consistency through association of detections across consecutive frames. The multi-camera matching system transforms single-camera tracks into trajectory segments that represent a person's movement through the camera's field of view.

Homography-based mapping to real-world layout. The system uses perspective transformation matrices to map camera coordinates to a unified layout coordinate system. For each camera, we manually define correspondence points between the camera view and the real-world floor plan. Then, the bottom-center point of each person's bounding box is transformed using the homography matrix [22], providing real-world positioning on the monitored building's floor plan. Then, the tracklet association is done using approximation algorithms.

Constraint Validation. The tracking pipeline integrates validation layers to ensure data quality and spatial consistency. Floor mask validation ensures all detections remain within walkable areas, with the help of panoptic segmentation [23] to create a binary mask of the walkable areas and then relocates invalid points to the nearest valid floor position.

V. Conclusion and Future Work

We presented a scalable, bandwidth-efficient video analytics system that integrates object detection, tracking, and face recognition into existing CCTV infrastructures with minimal overhead. The system leverages open source technologies and is designed for deployment flexibility, supporting both on-premise and cloud-native environments.

Our architecture supports per-camera analytics, such as heatmaps, path tracking, people counting, and geofencing, enabling actionable insights for operational and security decisions. Apache Airflow plays a central role in orchestrating the multi-model pipeline, while model selection was guided by accuracy, performance, and hardware efficiency—particularly under privacy and legal constraints like GDPR and the AI Act.

Future work includes integrating active learning, anomaly detection, and federated training to further enhance performance and compliance across distributed deployments.

Acknowledgement

The project entitled "GDPR compliant anonymous facial recognition for analytics and improving security", with no. 2021-1.1.4-GYORSÍTÓSÁV-2022-00076 has been implemented with the support provided by the Ministry of Culture and Innovation of Hungary from the National Research, Development and Innovation Fund, financed under the 2021-1.1.4-GYORSÍTÓSÁV funding scheme.

References

[1] *Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing directive 95/46/ec (general data protection regulation)*, https://eur-lex.europa.eu/eli/reg/2016/679/oj, Accessed: 2025-04-11, 2016.

[2] *Regulation (eu) 2024/xxxx of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts*, https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206, Accessed: 2025-04-11, 2024.

[3] Apache Software Foundation, *Apache airflow: Workflow management platform*, https://airflow.apache.org, Accessed: 2025-06-30, 2023.

[4] G. Jocher et al., "Yolov8: Ultralytics next-generation object detector," *arXiv preprint arXiv:2301.04634*, 2023.

[5] Z. Ge et al., "Yolox: Exceeding yolo series in 2021," *arXiv preprint arXiv:2107.08430*, 2021.

[6] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," *CVPR*, 2020.

[7] Y. Wu et al., "Detectron2," *Facebook AI Research*, 2019, https://github.com/facebookresearch/detectron2.

[8] C. Lugaresi et al., "Mediapipe: A framework for building perception pipelines," *arXiv preprint arXiv:1906.08172*, 2019.

[9] OPPO Research, "Yunet: A fast and accurate face detector," *GitHub repository*, 2022, https://tinyurl.com/fdyunet.

[10] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multi-task cascaded convolutional networks," *IEEE Signal Processing Letters*, 2016.

[11] J. Deng et al., "Arcface: Additive angular margin loss for deep face recognition," *CVPR*, 2019.

[12] S. Zhang et al., "Sface: An efficient network for face recognition," *arXiv preprint arXiv:2105.06070*, 2021.

[13] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," *CVPR*, 2015.

[14] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," *ICIP*, 2017.

[15] Y. Zhang et al., "Bytetrack: Multi-object tracking by associating every detection box," *ECCV*, 2022.

[16] NVIDIA Corporation, *Tensorrt: Nvidia deep learning inference optimizer and runtime*, https://developer.nvidia.com/tensorrt, 2023.

[17] Docker Inc., *Docker: Empowering app development for developers*, https://www.docker.com, Accessed: 2025-06-30, 2023.

[18] OpenCV Contributors, *Opencv: Open source computer vision library*, https://opencv.org, Accessed: 2025-06-30, 2023.

[19] PostgreSQL Global Development Group, *Postgresql: The world's most advanced open source relational database*, https://www.postgresql.org, Accessed: 2025-06-30, 2023.

[20] Hikvision Digital Technology Co., Ltd, *Isapi protocol specification*, https://www.hikvision.com, Accessed: 2025-06-30, 2023.

[21] B. Zhou, Y. Tang, and X. Wang, "Understanding crowd behaviors using dynamic textures and statistical analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 895–908, 2011. DOI: 10.1109/TPAMI.2011.176.

[22] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge university press, 2004.

[23] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic segmentation," in *CVPR*, 2019, pp. 9404–9413.