Vision Paper: Towards Software-Defined Video Analytics with Cross-Camera Collaboration

Juheon Yi*
Seoul National University, Korea
johnyi0606@snu.ac.kr

Chulhong Min
Nokia Bell Labs, Cambridge, UK
chulhong.min@nokia-bell-labs.com

Fahim Kawsar
Nokia Bell Labs, Cambridge, UK
fahim.kawsar@nokia-bell-labs.com

ABSTRACT

Video cameras are becoming ubiquitous in our daily lives. With the recent advancement of Artificial Intelligence (AI), live video analytics are enabling various useful services, including traffic monitoring and campus surveillance. However, current video analytics systems are highly limited in leveraging the enormous opportunities of the deployed cameras due to (i) centralized processing architecture (i.e., cameras are treated as dumb streaming-only sensors), (ii) hard-coded analytics capabilities from tightly coupled hardware and software, (iii) isolated and fragmented camera deployment from different service providers, and (iv) independent processing of camera streams without any collaboration. In this paper, we envision a full-fledged system for *software-defined video analytics with cross-camera collaboration* that overcomes the aforementioned limitations. We illustrate its detailed system architecture, carefully analyze the key system requirements with representative app scenarios, and derive potential research issues along with a summary of the status quo of existing works.

CCS CONCEPTS

• **Computing methodologies** → Distributed artificial intelligence.

KEYWORDS

Software-Defined Video Analytics, Cross-Camera Collaboration

ACM Reference Format:
Juheon Yi, Chulhong Min, and Fahim Kawsar. 2021. Vision Paper: Towards Software-Defined Video Analytics with Cross-Camera Collaboration. In *The 3rd International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things (AIChallengeIoT 21), November 15–17, 2021, Coimbra, Portugal.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3485730.3493453

1 INTRODUCTION

In recent years, video cameras are pervasively deployed at scale in our daily lives. Organizations are increasingly deploying camera systems to analyze live video streams for various purposes, including traffic monitoring, campus surveillance, and criminal chasing. With the recent advancement in AI and computer vision, live video analytics has enabled the automation of real-time monitoring tasks and is becoming a game changer everywhere.

Limitations of Existing Solutions. While extensive efforts have been recently made to facilitate the deployment of video analytics, the capabilities of existing solutions are still highly limited. First, they mostly treat cameras as dumb video streamers and perform the whole vision processing tasks in central edge/cloud servers [8, 16, 21], thereby causing severe resource waste, raising massive privacy concerns, and limiting the scalability. Second, existing systems have hard-coded analytics capabilities due to tightly coupled hardware (deployed cameras) and software (AI models and processing pipelines). Once an analytics service is deployed, it is difficult to dynamically adapt its capability (e.g., replacing AI models and service logics) or change the functionality of the whole system (e.g., adding new analytics). Third, analytics solutions and systems are closed, meaning that they are exclusively available only to their stakeholders. We can observe different service providers deploying their cameras separately even in the same place without any collaboration, causing a huge cost waste as a society. Fourth, although multiple cameras are deployed in the same place for the same analytics, each camera stream is mostly processed independently without collaboration among them [19]. Thus the analytics often misses opportunities to benefit from spatio-temporal redundancies between proximate cameras [7].

Our Vision. We envision *software-defined video analytics with cross-camera collaboration* with the following key features:

• **Support for Software-Defined Video Analytics.** By decoupling analytics logics (including AI models) from deployed cameras, it aims at supporting dynamic composition and execution of any analytics service on demand, without modifying the hardware. Such flexibility also enables camera networks to simultaneously run multiple analytics services from different service providers.

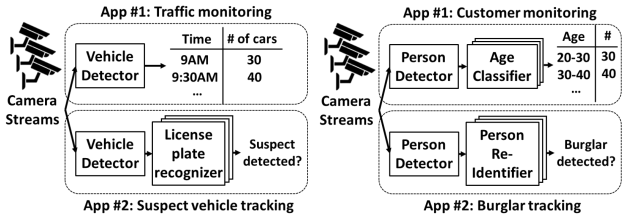
• **Abstraction for Cross-Camera Collaboration.** Even with the enormous advances made in computer vision, the analytics capability of a single camera is inherently limited due to complex scene contents. In such a case, it is expected to have a higher quality of service by having the collaboration of proximate cameras. Considering an unprecedented increase in camera deployment, the workload for collaborative analytics will be prevalent. The system aims at providing the abstraction for cross-camera collaboration, thereby facilitating the development of various video analytics services. Such an abstraction also gives the system visibility and fine-grained control for resource management.

• **System-Wide Holistic Orchestration.** When multiple analytics services are running concurrently, resource contention and performance degradation are inevitable. This problem becomes even more severe when the services are performed at edges (smart

*Work conducted during internship at Nokia Bell Labs, Cambridge, UK.

AIChallengeIoT 21, November 15–17, 2021, Coimbra, Portugal

ACM ISBN 978-1-4503-9097-2/21/11…$15.00
https://doi.org/10.1145/3485730.3493453



(a) Traffic surveillance. (b) Customer monitoring.

Figure 1: Example workloads for multi-app video analytics.

cameras or nearby edge servers) with limited processing capabilities. To this end, we aim at providing system-wide holistic orchestration that exploits the full resource capacity of the deployment environment and offers the maximum quality of service to users.

The rest of the paper is organized as follows. In Section 2, we first detail the overall system architecture for software-defined video analytics (Section 2.1), illustrate representative app scenarios (Section 2.2), and analyze the key system requirements (Section 2.3). Based on our analysis, we enumerate research issues and status quo of existing works in Section 3. Finally, we conclude the paper in Section 4 with our future research plans.

2 SOFTWARE-DEFINED VIDEO ANALYTICS WITH CROSS-CAMERA COLLABORATION

We envision the architecture of emerging software-defined video analytics with cross-camera collaboration, depict representative app[1] scenarios, and analyze the core system requirements.

2.1 System Components

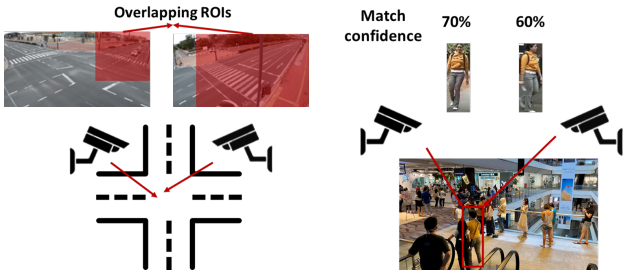
Emerging software-defined video analytics system will be comprised of the following components.

Smart Cameras. The target environment (e.g., campus, shopping mall, traffic intersections) will be covered by densely deployed *smart* cameras, each with its own processing capabilities ranging from embedded CPUs (e.g., Raspberry Pi) to GPU (e.g., NVIDIA Jetson TX2) and NPU/TPUs (e.g., Google Coral Edge TPU). The cameras may also be dynamically configurable (e.g., viewing angle can be rotated [9], zoomed in or out) so as to collaborate with each other (detailed collaboration scenarios in Section 2.2).

Edge/Cloud Server. An edge/cloud server resides between app developers and smart cameras. It plays two key roles in our system. First, it receives multiple app request queries from app developers and dynamically controls cameras for video capture, streaming, and processing. Second, it intelligently schedules concurrent workloads across smart cameras and its GPU resources (e.g., 4–8 NVIDIA Titan X GPUs), and returns the processed results back to users.

App Developers. The app developers who want to utilize smart cameras deliver app request queries to the edge/cloud server. Each query includes the following: (i) target camera(s), (ii) workload (i.e., processing pipelines and specific AI models), and (iii) processing interval and application Service Level Objectives (SLOs) (e.g., input video resolution, target latency and accuracy).

[1]Here, an app refers to a video analytics service on camera networks.



(a) Redundant ROI filtering. (b) Multi-camera fusion.

Figure 2: Examples of cross-camera collaboration.

2.2 Example Scenarios

We depict the following two example scenarios that motivate software-defined video analytics with cross-camera collaboration.

Traffic Surveillance in Urban City. A city planner wants to monitor the traffic flow in the city to develop a road construction plan. He requests a query to run the car detector (e.g., [15]) on each camera every minute, and obtains the aggregated results from the server. Meanwhile, a police officer who wants to track a suspect's vehicle also requests a query to identify the target vehicle (e.g., by vehicle detector and license plate recognizer). Upon receiving the request, the system locates the target vehicle. As multiple cameras can capture the target vehicle simultaneously (e.g., multiple cameras at the intersection), the system identifies and filters out the overlapping ROIs to minimize overall processing latency.

User Monitoring Shopping Mall. A restaurant owner in a shopping mall wants to analyze the customer demographics (e.g., age and gender) to determine which menu to sell. Meanwhile, a police officer who wants to detect and track any theft event in the mall requests a query to run a person detection and activity recognition pipeline. In case the theft is detected, the system runs the person re-identifier to track the suspect. As the burglar may not be accurately detected from a single camera (e.g., due to viewing angles, occlusion, or motion blur), the system fuses the analysis results of multiple neighboring camera streams to improve accuracy.

2.3 System Requirements

From the above example scenarios, we extract the following key system requirements that need to be considered to realize our vision.

Multi-App Concurrent Execution. The system needs to process multiple concurrent app requests of which number and combination change over time. The system should continuously monitor the concurrent requests and flexibly schedule their execution in a holistic manner to satisfy the SLOs specified by the users.

Camera Selection and Pipeline Adaptation. Each video analytics app workload is composed of multi-stage AI models as shown in Figure 1 (e.g., person detection on the entire frame, activity recognition for each person detected). Naive execution of the full pipeline on the entire video streams incurs significant latency and resource wastage. The system needs to efficiently identify which camera(s) to process (e.g., ones that contain the objects of interest) and adapt the processing pipelines (e.g., duty cycle, DNN complexity) depending on the scene content (e.g., object size and pose) and available resources to optimize resource-accuracy tradeoff.

Table 1: Summary of research issues and status quo of prior video analytics work.

Research issue	Keyword	Related work	Multi-camera?	Multi-app?	Collaboration?
Query abstraction	Abstraction for activity recognition	Caesar [13]	O	X	X
Fine-granule multi-app resource sharing	Computation caching and reuse	Starfish [11]	X	O	X
	Shared backbone batching	Nexus [16]	O	O	X
Multi-camera pipeline joint adaptation	Pipeline adaptation	DIVA [17], Reducto [10], EagleEye [18]	X	X	X
	Single camera duty cycling	MARLIN [1], DeepMon [6]	X	X	X
	Cross-camera duty cycling	Spatula [7], Pasandi et al. [14]	O	X	O
Multi-app cross-camera collaboration scheduling	Cross-camera redundancy optimization	CrossROI [4]	O	X	O
	Multi-camera view adaptation/selection/fusion	MoVi [2], CMCAOT [9]	O	X	O
Workload-adaptive resource scheduling	Video quality adaptation	DDS [3], Liu et al. [12], AWStream [20]	X	X	X
	Distributed edge scheduling	Distream [19], VideoEdge [5]	O	X	X
	Server cluster scheduling	VideoStorm [21], Chameleon [8], Nexus [16]	O	O	X

Workload Scheduling on Heterogeneous Edges. To maximize the system-wide performance, the system should fully exploit the computing resources of smart cameras and an edge/cloud server. To this end, it is important to balance multi-app workloads across cameras and servers flexibly and dynamically, considering their heterogeneous computing capabilities. Furthermore, the scheduling overhead (e.g., network latency and energy consumption for streaming high-resolution videos) under dynamic resource availability (e.g., network bandwidth fluctuation) needs to be considered.

Cross-Camera Collaboration. In dense deployment settings, multiple cameras can have overlapping coverage. The system should accurately identify and support cross-camera collaboration to maximize system performance. Figure 2 shows the examples of cross-camera collaboration: filtering out redundant ROIs to reduce processing latency in traffic surveillance scenario (Figure 2(a)), fusing multiple video stream processing results to improve the detection accuracy of a burglar in a shopping mall user monitoring scenario (Figure 2(b)). Cross-camera collaboration becomes more important when the cameras' viewing angle and zoom factor can be dynamically configured for different app purposes (e.g., jointly rotating the viewing angles to accurately track the target at different views [9]).

3 VISION, CHALLENGES AND STATUS QUO

We envision a software-defined video analytics platform (Figure 3), and enumerate research issues and summarize prior works (Table 1).

3.1 Query Abstraction and Translation

Software-defined video analytics platform should support abstractions for developers to specify diverse app requests. Most of the existing works, however, have been focused on single-shot object detection tasks [5, 8, 19] and lacks considerations for such abstraction. Caesar [13] proposed an initial design on abstraction for multi-camera activity recognition apps. We aim at designing abstractions for more diverse video analytics queries (e.g., multi-object tracking) and environments (e.g., multiple cameras with overlapping views).

3.2 Multi-App-Aware Pipeline Adaptation

Fine-Granule Multi-App Resource Sharing. Video analytics pipelines commonly start by detecting and recognizing objects of

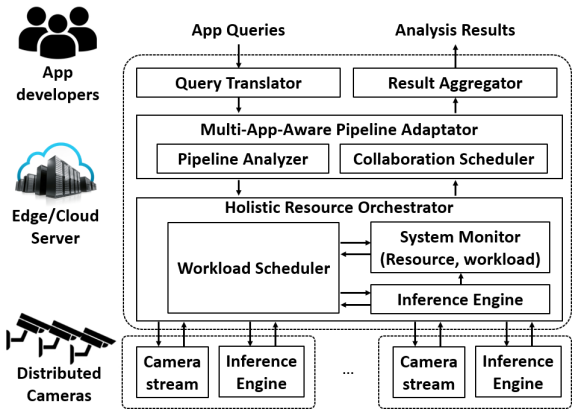


Figure 3: Software-defined video analytics architecture.

interest in the scene. Therefore, it will be highly likely that multiple app requests share common processing blocks. For example, in the customer monitoring scenario in Section 2.2, both a customer demographics analyzing app and a burglar detection app require a person detection stage. To improve the overall performance by resource sharing, the system should closely analyze the pipeline of each video analytics app query, identify redundancy at a fine-grained operation level, and share computing resources (e.g., via caching or batching). For example, Starfish [11] considers caching and reusing common image processing blocks across multiple continuous vision apps running on a single mobile camera. We can consider extending the idea to multi-camera settings with overlapping ROIs. Nexus [16] batches the DNN inference queries at the server with a common backbone network to maximize throughput [16]. We can also consider extending the idea to schedule batching at distributed smart cameras considering the networking overhead.

Multi-Camera Pipeline Joint Adaptation. In processing multiple apps, the system should efficiently identify the target camera streams and corresponding pipelines (e.g., duty cycle and DNN model) in a joint manner. Existing works, however, have mostly been designed for and limited to single-camera, single-app environments. MARLIN [1] tracks the scene content changes to adapt the duty cycle or reuse cached results. EagleEye [18] adapts the face identification pipeline complexity depending on the captured face content (i.e., pose and resolution) and expected resource usage.

DIVA [17] and Reducto [10] take the hierarchical filtering approach to select the necessary frames to process using lightweight filters (e.g., edge detection). Recently, Spatula [7] leverages the spatio-temporal correlation of multiple neighboring cameras to jointly adapt their duty cycle, but only assumes a single target tracking scenario. We believe new research efforts should be made to realize the joint adaptation of concurrent multi-camera pipelines.

Cross-Camera Collaboration Scheduling Finally, the system should flexibly schedule the cross-camera collaboration policy to maximize the SLOs of multiple apps. Several works have proposed various cross-camera collaboration policies, but are limited to single-app scenarios with static policies (e.g., filtering out overlapping ROIs to minimize resource usage [4], choosing the camera that best captures the target to maximize accuracy [2]). Depending on the latency/accuracy target and resource availability, the system should dynamically adapt the collaboration policies to satisfy the SLOs of multiple app requests. Furthermore, in case that cameras are dynamically configurable, the system should also handle conflicting requests (e.g., two different queries may request the camera to rotate in opposite angles) across multiple app queries.

3.3 Workload-Adaptive Resource Scheduling

The workload complexity of video analytics apps highly fluctuates over time and camera location depending on the scene content [19] (e.g., the workload complexity of customer demographics analysis heavily depends on the number of people in the current frame). The system should continuously monitor and flexibly balance the workloads across heterogeneous smart cameras and servers. However, most of the existing video analytics systems [8, 16, 21] have assumed the centralized processing architecture (i.e., it receives video streams from cameras and processes all processing operations on the server-side) and focused on server cluster computing resource scheduling. Recently, Distream [19] has aimed at workload-adaptive resource scheduling in distributed smart camera environments, but lacks considerations for network transmission overhead (both between cameras and between cameras and a server) as well as concurrent multi-app workload scheduling. A number of works [3, 12, 20] focused on video stream quality adaptation for network bandwidth optimization, but are limited to single-app, single-camera scenarios. Multi-app workload-adaptive, holistic resource scheduling on heterogeneous edges remains an unsolved challenge.

4 CONCLUDING REMARKS

We envisioned a full-fledged system for software-defined video analytics with cross-camera collaboration to overcome the limitations of current camera deployment and video analytics systems, analyzed its core system requirements, and enumerated research issues in supporting them. For future work, we aim at designing abstractions for the users to dynamically compose and execute various collaborative analytics apps. Furthermore, we plan on designing an end-to-end holistic resource orchestrator that utilizes the computing capabilities of smart cameras to support concurrent multi-app video analytics apps.

REFERENCES

[1] Kittipat Apicharttrisorn, Xukan Ran, Jiasi Chen, Srikanth V Krishnamurthy, and Amit K Roy-Chowdhury. 2019. Frugal following: Power thrifty object detection and tracking for mobile augmented reality. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 96–109.

[2] Xuan Bao and Romit Roy Choudhury. 2010. MoVi: mobile phone based video highlights via collaborative sensing. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*. 357–370.

[3] Kuntai Du, Ahsan Pervaiz, Xin Yuan, Aakanksha Chowdhery, Qizheng Zhang, Henry Hoffmann, and Junchen Jiang. 2020. Server-driven video streaming for deep learning inference. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*. 557–570.

[4] Hongpeng Guo, Shuochao Yao, Zhe Yang, Qian Zhou, and Klara Nahrstedt. 2021. CrossRoI: Cross-camera Region of Interest Optimization for Efficient Real Time Video Analytics at Scale. *arXiv preprint arXiv:2105.06524* (2021).

[5] Chien-Chun Hung, Ganesh Ananthanarayanan, Peter Bodik, Leana Golubchik, Minlan Yu, Paramvir Bahl, and Matthai Philipose. 2018. Videoedge: Processing camera streams using hierarchical clusters. In *2018 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 115–131.

[6] Loc N Huynh, Youngki Lee, and Rajesh Krishna Balan. 2017. Deepmon: Mobile gpu-based deep learning framework for continuous vision applications. In *Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services*. 82–95.

[7] Samvit Jain, Xun Zhang, Yuhao Zhou, Ganesh Ananthanarayanan, Junchen Jiang, Yuanchao Shu, Paramvir Bahl, and Joseph Gonzalez. 2020. Spatula: Efficient cross-camera video analytics on large camera networks. In *2020 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 110–124.

[8] Junchen Jiang, Ganesh Ananthanarayanan, Peter Bodik, Siddhartha Sen, and Ion Stoica. 2018. Chameleon: scalable adaptation of video analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 253–266.

[9] Jing Li, Jing Xu, Fangwei Zhong, Xiangyu Kong, Yu Qiao, and Yizhou Wang. 2020. Pose-assisted multi-camera collaboration for active object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 34. 759–766.

[10] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, and Ravi Netravali. 2020. Reducto: On-camera filtering for resource-efficient real-time video analytics. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*. 359–376.

[11] Robert LiKamWa and Lin Zhong. 2015. Starfish: Efficient concurrency support for computer vision applications. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. 213–226.

[12] Luyang Liu, Hongyu Li, and Marco Gruteser. 2019. Edge assisted real-time object detection for mobile augmented reality. In *The 25th Annual International Conference on Mobile Computing and Networking*. 1–16.

[13] Xiaochen Liu, Pradipta Ghosh, Oytun Ulutan, BS Manjunath, Kevin Chan, and Ramesh Govindan. 2019. Caesar: cross-camera complex activity recognition. In *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*. 232–244.

[14] Hannaneh Barahouei Pasandi and Tamer Nadeem. 2019. Collaborative intelligent cross-camera video analytics at edge: Opportunities and challenges. In *Proceedings of the First International Workshop on Challenges in Artificial Intelligence and Machine Learning for Internet of Things*. 15–18.

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[16] Haichen Shen, Lequn Chen, Yuchen Jin, Liangyu Zhao, Bingyu Kong, Matthai Philipose, Arvind Krishnamurthy, and Ravi Sundaram. 2019. Nexus: a GPU cluster engine for accelerating DNN-based video analysis. In *Proceedings of the 27th ACM Symposium on Operating Systems Principles*. 322–337.

[17] Mengwei Xu, Tiantu Xu, Yunxin Liu, Felix Xiaozhu Lin, and ECE Purdue. 2021. Video Analytics with Zero-streaming Cameras. In *2021 USENIX Annual Technical Conference (USENIX ATC 21)*. 459–472.

[18] Juheon Yi, Sunghyun Choi, and Youngki Lee. 2020. EagleEye: Wearable camera-based person identification in crowded urban spaces. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–14.

[19] Xiao Zeng, Biyi Fang, Haichen Shen, and Mi Zhang. 2020. Distream: scaling live video analytics with workload-adaptive distributed edge intelligence. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*.

[20] Ben Zhang, Xin Jin, Sylvia Ratnasamy, John Wawrzynek, and Edward A Lee. 2018. AWStream: Adaptive wide-area streaming analytics. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication*. 236–252.

[21] Haoyu Zhang, Ganesh Ananthanarayanan, Peter Bodik, Matthai Philipose, Paramvir Bahl, and Michael J Freedman. 2017. Live video analytics at scale with approximation and delay-tolerance. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*. 377–392.