Published in IEEE Transactions on Circuits and Systems for Video Technology, Volume: 31, Issue: 10, October 2021.

Experimental Analysis of Optimal Bandwidth Allocation in Computer Vision Systems

Sina Davani and Nabil J. Sarhan

***Abstract*—This paper considers computer vision (CV) systems in which a central monitoring station receives and analyzes the video streams captured and delivered wirelessly by multiple cameras. It addresses how the bandwidth can be allocated to various cameras by presenting a cross-layer solution that optimizes the overall detection or recognition accuracy. In further contrast with prior work, it presents and develops a real CV system and subsequently provides a detailed experimental analysis of cross-layer optimization. Other unique features of the developed solution include employing the popular HTTP streaming approach, utilizing homogeneous cameras as well as heterogeneous ones with varying capabilities and limitations, and including a new algorithm for estimating the effective medium airtime. The results show that the proposed solution significantly improves the CV accuracy.**

***Index Terms*—Automated video surveillance, bandwidth allocation, cross-layer optimization, computer vision systems, effective airtime estimation, face recognition, many-to-one video streaming, WLAN.**

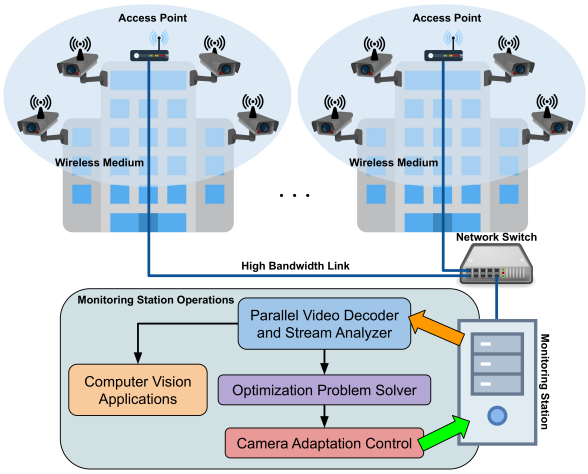


Fig. 1. An Illustration of the Considered Computer Vision System

I. Introduction

COMPUTER Vision (CV) systems, including automated video surveillance, enable the real-time detection of threats by running CV algorithms as opposed to human observation. Most research on CV has focused primarily on developing CV algorithms [1], [2], [3], [4] (and references within), but much less work has considered the design of CV systems.

This paper addresses the bandwidth allocation problem in *many-to-one* CV systems. As illustrated in Figure 1, the considered system consists of multiple video cameras capturing and delivering live video streams to a central monitoring station over a single-hop Wi-Fi LAN. Multiple such cells can be utilized to construct a larger system. The monitoring station runs CV algorithms to detect potential threats in the monitored site. This station is generally connected to the access point with a high-bandwidth wired link that is not deemed as a bottleneck. The main challenge in the considered system is the limited available network bandwidth, which should be estimated accurately and then distributed efficiently among various cameras.

Bandwidth allocation in general many-to-one video streaming systems has been addressed by only a few studies through cross-layer optimization [5], [6], [7], but all these studies are simulation-based. The objective in [5], [6], [7] was to optimize video distortion. Although distortion may be appropriate for certain systems, the detection/recognition accuracy is the most important metric in CV systems. Study [6] considered the accuracy, but only for the simple face detection tasks. Besides, the aforementioned studies assumed Real Time Transfer Protocol (RTP) streaming instead of HTTP streaming and none used H.264 encoding; they assumed MJPEG [6], [7] or abstract video data [5].

In contrast with prior work, we build a real CV system for automated video surveillance and run actual experiments. We also consider both face detection and face recognition applications. In further contrast to prior studies, the system employs HTTP streaming and employs homogeneous cameras as well as heterogeneous ones with varying capabilities (including resolutions and frame rates) to mimic varying deployment scenarios. We primarily use H.264 but consider the coexistence of other encoders. We develop the system utilizing a variety of open-source libraries, including FFmpeg, Simple DirectMedia Layer (SDL), Snappy, FaceNet, TensorFlow, and Curl. We also develop a customized video player to enable full control of the video decoding process and provide the statistics required by the optimization solution.

Moreover, we propose a cross-layer optimization solution for allocating bandwidth to various cameras in a manner that optimizes the overall detection or recognition accuracy. The solution dynamically manages the application rates and transmission opportunities of various cameras based on the current network conditions, considering the capabilities and limitations of these cameras. Furthermore, the solution includes a new algorithm for determining effective airtime of the network based on a novel method for estimating data dropping by using smoothing calculations of the stream bitrates received by the monitoring station.

We provide detailed experimental results, using three different setups. The first experimental setup utilizes real video surveillance dataset in campus, office, store, and street envi-

ronments, collected from publicly available videos, whereas the second is based on a live laboratory environment, and the third utilizes the IARPA Janus Benchmark-B Face Challenge (IJB-B) dataset [8]. The results demonstrate that the proposed solution significantly improves the CV accuracy.

This paper significantly extends our preliminary conference paper [9] by developing a solution for optimizing the face recognition accuracy and using the IJB-B dataset.

The rest of this paper is organized as follows. Section II provides background information and discusses the related work. Subsequently, Section III presents the developed system and Section IV presents the proposed cross-layer optimization solution. Section V discusses the performance evaluation methodology. Finally, Section VI presents and analyzes the main results.

II. Background Information and Related Work

A. Cross-Layer Optimization

The *Enhanced Distributed Channel Access* (EDCA) mode of the 802.11e standard enables provisioning different quality-of-service levels among different access categories within the same station by adjusting parameters, such as the *Transmission Opportunity Time* (TXOP) [10].

Prior studies on cross-layer optimization in wireless video streaming considered (i) systems in which only one station streams a video at a time [11] (and references within), (ii) systems in which a central video server streams to multiple stations [12] (and references within), and (iii) systems in which multiple stations deliver video streams to a central station [13], [5], [7], [6]. The latter are the most related to this paper. In such many-to-one video streaming systems, the main objective was to minimize the sum of video distortion in all received video streams [7], [5], instead of the accuracy error, which is the main concern in CV systems. Study [6] explored optimizing only face detection, which is among the simplest CV tasks. In addition, the problem formulation did not include the limitation of the cameras in sending the encoded video streams. The aforementioned studies were simulation-based, used RTP streaming, and assumed MJPEG or abstract data streams. Video streaming in wireless ad-hoc networks was considered in [14].

As will be discussed in Subsection IV-A, the optimization solution requires an accurate estimation of the effective airtime, which can be defined as the fraction of the network time that is used for delivering useful data. Study [5] developed an analytical model for the effective airtime, whereas study [7] developed an online estimation algorithm, addressing the shortcomings of that analytical model. In this paper, we enhance the estimation algorithm by incorporating a novel method for estimating data dropping via smoothing calculations.

B. CV Algorithms

We experiment with both face detection and face recognition. Face recognition is a major CV algorithm used in many applications, including authentication systems, personal photo enhancement, automated video surveillance, and photo search engines. Recent studies on face recognition employ deep learning using convolutional neural networks [15], with major algorithms including FaceNet [2] and ArcFace [3].

III. Developed Computer Vision System

We build a real many-to-one CV system and analyze the effectiveness of cross-layer optimization by providing the results of actual experiments. As illustrated in Figure 1, the system consists of a monitoring station and a variety of video cameras, including PTZ cameras, all of which are connected by a Wi-Fi network. The cameras include four Pan/Tilt/Zoom (PTZ) surveillance cameras (IPCam 7210W), one wide-angle camera (VivoTek IP7139), and two webcams (HP Truevision HD and Labtec PRO Webcam). The system employs HTTP streaming for delivering videos from the cameras to the monitoring station. To capture realistic deployment scenarios, we use both homogeneous and heterogeneous cameras. We primarily use H.264 but consider the co-existence of other encoders.

Figure 2 illustrates the overall system design, including built-in modules for performance evaluation. In the extensive system development process, we use the following libraries for developing various system aspects: FFmpeg, Simple Direct-Media Layer (SDL), Snappy, TensorFlow, and Curl. To turn the two webcams into functional IP cameras, we employ the VLC media streaming tool and develop a program in Python to act as a *virtual interface* for these cameras. Hence, we refer to these cameras as *virtualized IP cameras*. The virtual interface enables these two webcams to adjust their encoding bitrates according to the received control messages from the monitoring station, thereby allowing their treatment as any regular IP camera.

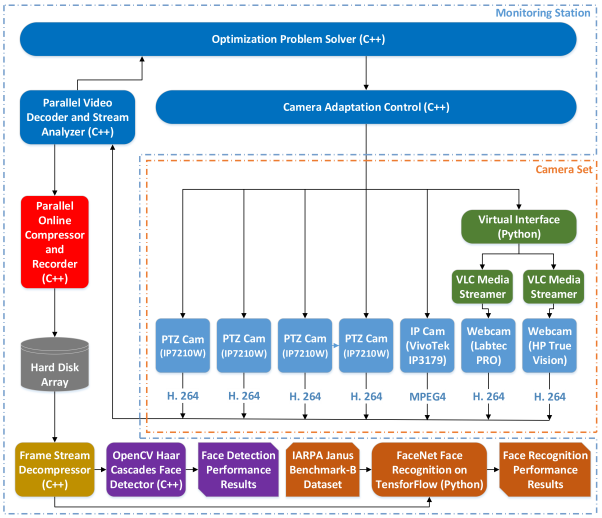


Fig. 2. An Illustration of the Overall System Design Including Built-in Modules for Enabling Performance Evaluation

The monitoring station has the following main units.

- *Parallel Video Decoder and Stream Analyzer* – This unit receives the video streams from IP cameras and decodes them. It also analyzes the streams to determine accurately all the system parameters involved in the effective airtime

estimation, bitrate smoothing, and optimization solution. We develop a customized multi-threaded video player in C++ using FFmpeg and SDL libraries to provide full control over the video decoding process and offer all the statistics required by the optimization solution. As SDL offers special multi-threading tools for media-rich applications, we incorporate it to handle parallelism and thread creation and to enforce mutual exclusion for different resources.

- *Optimization Problem Solver* uses the estimated system parameters to determine the optimal distribution of the effective airtime among various cameras and then sends the allocations to the next unit.
- *Camera Adaptation Control* – This unit receives the optimal portion of the effective airtime for each camera, generates a properly-formatted HTTP control message, and delivers the message to each camera. We utilize the HTTP message transfer function of Curl, a client-side URL data-transfer library supporting various protocols, including HTTP.
- *Parallel Online Compressor and Recorder* – We devise this unit in C++ to implement an off-line approach for (i) analyzing the received video stream by applying CV algorithms and (ii) facilitating the comparative performance evaluation of various bandwidth allocation solutions. As expected, without the use of a specialized distributed processing system, the simultaneous analysis of all video streams is not achieved in real-time, even when using a workstation with 8-core AMD Ryzen 7 running at 4 GHz with 32 GB of DDR4 RAM due to the aggregate computational complexity of CV algorithms. The devised off-line approach helps bypass this challenge. Specifically, it records the video streams from various cameras for further analysis, without any video encoding or transcoding, but this raises another challenge, that is no cost-effective storage device can provide the necessary capacity and performance. Hence, we develop a recording process that performs fast lossless compression on the received data and enables the simultaneous handling of writing the video streams on multiple hard disk drives using SDL and Snappy libraries. Snappy provides the toolkit for fast online compression. Note that the recording process and the offline examination approach are only for performance evaluation purposes and are not imposed by the proposed optimization solution. Real deployments of CV systems can address the real-time detection and recognition by utilizing a distributed processing system and/or powerful GPUs.
- *Frame Stream Decompressor* – We develop this unit in C++ to uncompress and analyze the compressed recorded video streams. The uncompression task utilizes the Snappy library.
- *Face Detection* – We develop this unit in C++ using OpenCV to run the face detection function using Haar feature-based cascade classifiers.
- *Face Recognition* – We develop this unit in Python using FaceNet on TensorFlow to run the face recognition tasks. Specifically, we utilize the facenet 1.0.3 Python package, an open-source TensorFlow implementation of the face recognizer described in [2]. We use the pre-trained model named 20180402-114759, which is trained on the VGGFace2 dataset and has the Inception ResNet v1 architecture.

When the cameras receive the HTTP control messages from the monitoring station, they act accordingly to adjust their video capturing and encoding parameters.

IV. Proposed Cross-Layer Optimization Solution

We develop an enhanced cross-layer optimization solution, which dynamically distributes and allocates the wireless network bandwidth among various cameras in the considered many-to-one CV system, illustrated in Figure 1, with the objective of optimizing either the overall detection or recognition accuracy. The system includes S cameras and each one streams a different video at rate r_s over a bandwidth-limited WiFi medium to the access point, which in turn delivers the stream to the monitoring station through typically a high bandwidth link. Different video sources may have dissimilar physical rates. The CV system can be expanded by including and interconnecting multiple such cells.

A. Cross-Layer Optimization Problem Formulation

As in [6], the optimization problem can be formulated as the minimization of the sum of the accuracy error ($\mathcal{E}$) of all video streams received by the central monitoring station. Since the wireless medium is shared by all S cameras, the problem can be formulated as to how to allocate the optimal portion of the airtime f_s to each camera s. The set of allocations is given by $F^* = \{f_s^* | s = 1, 2, 3, ..., S\}$. The application-layer transfer rate of camera s can be given by $r_s = f_s \times Y$, where Y is the total medium bandwidth. Assuming A_{eff} is the effective airtime of the medium, the optimization problem can be formulated as follows:

$$F^* = \arg\min_{F} \sum_{s=1}^{S} \mathcal{E}(r_s), \tag{1a}$$

Subject to

$$\sum_{s=1}^{S} f_s = A_{eff}, \tag{1b}$$

$$0 \leq f_s \leq 1, \tag{1c}$$

$$r_s = f_s \times Y, \tag{1d}$$

$$f_s \leq \frac{y_s}{Y}, \quad \text{and} \tag{1e}$$

$$s = 1, 2, 3, ..., S. \tag{1f}$$

We enhance the formulation in [6] by adjusting Condition (1d) to consider the total available physical rate of the access point and introducing Condition (1e) to consider the physical limitations of the cameras in sending the encoded video streams. In contrast, r_s is set to $f_s \times y_s$ in [6], where y_s is the perceived physical rate by camera s, thereby utilizing only a portion of the physical capacity of the medium.

B. Effective Airtime Estimation

As the formulated optimization problem requires an accurate value for the effective airtime of the medium, we propose an enhanced estimation algorithm, employing a novel method for assessing data dropping via smoothing calculations of the bitrates of the streams received by the monitoring station. Figure 3 shows the simplified algorithm. First, with each camera sending at a rate that is equal to the medium bandwidth divided by the number of cameras, the algorithm determines the effective throughput t_s for the video stream for each camera s as received by the application layer of the monitoring station. The algorithm then uses t_s to provide the initial value of the effective airtime (A_{eff}) as $A_{eff} = \sum_{s=1}^{S} t_s/Y$. Subsequently, during an estimation period, the algorithm assesses the overall data dropping and corruption (i.e., error) rate d_s at the monitoring station while receiving the video stream from video camera s, and then adjusts the estimated effective airtime accordingly. The algorithm determines d_s as the measured corrupted data rate for the stream plus a value capturing the difference between the nominal announced frame rate from the cameras and the actual received frame rate. Specifically, we develop and employ the following equation to assess d_s:

$$d_S = [CorruptData_s + (SSDR_s \times SumFrameDelayVar_s) \times DW]/EP, \quad (2)$$

where $CorruptData_s$ is the total size of the received packets from source s that are corrupted, $SSDR_s$ is the smoothed stream bitrate for source s, $SumFrameDelayVar_s$ is the sum of variations between the frames produced by source s and the corresponding one that is received by the monitoring station, DW is the delay weight constant designating the importance of the second term, and EP is the estimation period. In $SumFrameDelayVar_s$, the variation is measured in terms of the delays between consecutive frames.

Through a *smoothing operation* over the evaluation time, rather than just using the momentary bitrate value, SSDR can be estimated more accurately as follows:

$$SSDR = SC \times PSSDR + (1 - SC) \times MSDR, \quad (3)$$

where SC, $PSSDR$, and $MSDR$ are the smoothing constant, previously estimated smoothed stream data rate, and momentary stream data rate, respectively.

The algorithm determines the overall average dropping ratio perceived by the monitoring station as follows: $A_\Delta = \sum_{s=1}^{S} d_s/Y$. This value is used to adapt the current value of A_{eff} at the end of the current estimation period. If A_Δ is 0, the algorithm increases A_{eff} by $C \times A_{thresh}$. In contrast, if A_Δ is greater than some threshold A_{thresh}, it reduces A_{eff} by $\acute{C} \times (A_\Delta - A_{thresh})$, where A_{thresh} controls the allowable data dropping in the network and $\acute{C}$ and C are network related constants. Otherwise, it increases A_{eff} by $\tilde{C} \times (A_{thresh} - A_\Delta)$, where $\tilde{C}$ is also a constant value. To ensure better convergence and stability, we set C, $\acute{C}$, and $\tilde{C}$, respectively, to 20, 0.8, and 16 based on extensive experiments.

Input: $\{t_1, .., t_S, PSSDR_1, .., PSSDR_S, MSDR_1, .., MSDR_S, CorruptData_1, .., CorruptData_S, sumFrameDelayVar_1, .., sumFrameDelayVar_S\}$
Output: $\{A_eff\}$
if this is the first time to run the algorithm
  $A_{eff} = \sum_{s=1}^{S} t_s/Y$;
At the end of each estimation period{
  For each source s = 1 to S\{
    $SSDR_s = SC \times PSSDR_s + (1 - SC) \times MSDR_s$;
    $d_S = [CorruptData_s + (SSDR_s \times SumFrameDelayVar_s) \times DW]/EP$; } // For
  $A_\Delta = \sum_{s=1}^{S} d_s/Y$;
  if ($A_\Delta == 0$) // Increase A_{eff} by $C \times A_{thresh}$
    $A_{eff} = A_{eff} + C \times A_{thresh}$;
  else if ($A_\Delta > A_{thresh}$) // Reduce A_{eff} by $\acute{C} \times (A_\Delta - A_{thresh})$
    $A_{eff} = A_{eff} - \acute{C} \times (A_\Delta - A_{thresh})$;
  else // Increase A_{eff} till first decrement
    $A_{eff} = A_{eff} + \tilde{C} \times (A_{thresh} - A_\Delta)$; }// At

Fig. 3. Simplified Algorithm for Dynamically Estimating the Effective Airtime

C. Cross-Layer Optimization Solution

1) Face Detection: According to [6], the accuracy error for face detection can analytically be modeled as $\mathcal{E}(r_s) = a \times r_s^b + c$, where a, b, and c are constants. Hence, it can be shown that the optimization problem (Equation (1)) is a budget-constrained convex problem and subsequently can be solved by Lagrangian Relaxation. Assuming realistically that all cameras have the same b_s value, the solution can be given by

$$f_s^* = \left(\frac{-\lambda}{a_s Y^{b_s} b_s}\right)^{(1/(b_s-1))}, \quad (4)$$

where

$$\lambda = \left(\frac{A_{eff}}{\sum_{s=1}^{S} \left(\frac{-1}{a_s Y^{b_s} b_s y_s}\right)^{(1/(b_s-1))}}\right)^{(b_s-1)}. \quad (5)$$

We devise the following method to ensure that Condition (1e) is met: if f_s^* is larger than y_s/Y, we restart this solving process after setting f_s to y_s/Y, subtracting y_s/Y from A_{eff}, and then eliminating that source from the problem domain.

2) Face Recognition: According to [16], the accuracy error for face recognition can be modeled as $\mathcal{E}(r_s) = a \times e^{b \times r_s} + c \times e^{d \times r_s}$, where a, b, c, and d are constants. Assuming $b = d$, the model can be simplified as $\mathcal{E}(r_s) = a \times e^{b \times r_s}$, where a and b are constants. Our experiments demonstrate the accuracy of the simplified model. For the original model, the SSE, R-Square, Adjusted R-Square, and RMSE values are 0.006267, 0.995, 0.9931, and 0.02799, respectively. In contrast, the values with the simplified model are 0.007374, 0.9941, 0.9935, and 0.02715, respectively.

Using the simplified model, the optimization problem in Equation (1) becomes a budget-constrained convex problem and can also be solved by Lagrangian Relaxation, similar to the case of face detection. Again, assuming realistically that all b_s values are equal, the solution can be given by

$$f_s^* = \frac{ln(\frac{-\lambda}{a_s b_s})}{b_s}, \quad (6)$$

where

$$\lambda = -(e^{\frac{A_{eff}}{b_s}} \prod_{s=1}^{s} a_s b_s)^{\frac{1}{S}}. \quad (7)$$

D. Proposed Method for Determining the Constant Values of the Accuracy Error Models

The optimization solution requires knowing a priori all the constant values of the analytical error models for the detection and recognition accuracy, but these constants depend on the actual deployed system and monitored site, and their values may change with time. Hence, we present the following method for determining the constant values of the analytical accuracy error model, namely a and b. During initial calibration or subsequent re-calibration, these values are determined offline by first recording a video of the actual environment during typical operation at the highest supported resolution and bitrate by a selected surveillance camera in the deployed system. Later on, the video is transcoded to different combinations of the resolutions and bitrates. Subsequently, the proportions of the detected/recognized faces relative to the original video are computed to find the accuracy error ($\mathcal{E}$) values. Finally, the values of the model constants (a and b) are estimated based on the analytical models of the accuracy error. Although the computational complexity of the transcoding can be high considering the combination of encoded parameters, the system needs to recompute the constants only in the presence of significant changes in the system or environment. Besides, the readjustment of these constant values is a relaxed requirement and thus the values can be recomputed in a background process while the system proceeds to operate normally. Specifically, the allocation value set F^* can be determined dynamically by applying Equation (4) or (6), during the normal operation of the CV system, depending on the desired detection/recognition optimization performance.

V. Performance Evaluation Methodology

Table I summarizes the main parameters. Extensive analysis indicates that setting A_{thresh} and *Estimation Period* to 0.0075 and 5 seconds, respectively, improves performance in terms of both stability and convergence.

We conduct experiments using three different setups, as summarized in Table II and detailed later in this section. All experiments are performed using a TP-LINK TL-WR841N wireless router as the access point and a workstation with 8-core AMD Ryzen 7 running at 4 GHz with 32 GB of DDR4 RAM as the monitoring station. H.264 is used in all cameras except for VivoTek IP7139, which supports MPEG-4 instead.

We compare the proposed solution, referred to as *New Optimization*, with the solution in [6], referred to as *Existing Optimization*. We also analyze the case when the optimization is disabled, referred to as *No Optimization*. The main analyzed metrics are *face detection accuracy* and *face recognition accuracy*, measured in terms of the overall number of detected/recognized faces. *OpenCV* is used to run the Viola-Jones algorithm on the decoded video streams in Experimental Setups I and II. In contrast, *FaceNet* is utilized to run face recognition in Experimental Setup III.

TABLE I
Summary of Experimental Characteristics and Parameters

Parameter	**Model/Value(s)**
Number of video cameras	4, 7
Recording Period (minutes)	10
Application Rate	If not optimized: Max. Access Point Rate /Number of Cameras
Video Frame Rate (fps)	Camera Dependent: 7.5, 10, 25
Physical characteristics	Extended Rate (802.11n)
Physical Data Rate (Mbps)	30, 25, 20, 15, 13, 10, 5
State Report Interval (seconds)	5
Detection Model Constants: a, b	3103, -1.309
Recognition Model Constants: a, b	1.593, -88.35×10^{-5}
Used Cameras	IP7210W (4 units), HP Truevision HD, Labtec PRO Web-cam, VivoTek IP7139
Camera Video Resolutions	1280×720, 800×600, 640×480

TABLE II
Summary of the Three Experimental Setups

Setup	**Recorded Resolution(s)**	**Cams**	**Video Content**	**Application**
I	1280×720, 800×600, 640×480	7	Surveillance videos	Detection
II	1280×720	4	Live laboratory environment	Detection
III	1280×720	4	IJB-B Dataset	Recognition

A. Experimental Setup I: Using a Real Video Surveillance Dataset

Experimental Setup I uses various types of cameras (discussed in Section III) to capture real surveillance videos rendered on separate monitors (model: Dell E2210Hc), thereby providing repeatable, realistic, and diverse scenery. Table III summarizes the main characteristics of the video surveillance dataset, which are collected from YouTube and other sources and will be publicly available. The videos have different characteristics (including resolution and frame rate) and come from different environments: *office*, *campus*, *stores*, and *busy streets*. The original videos are truncated so that each category has nearly the same total video duration. Figure 4 shows sample video frames form this dataset.

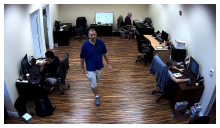
B. Experimental Setup II: Live Laboratory Environment

Experimental Setup II uses our PTZ IP-Cam 7210W cameras to capture videos from a real lab environment. As discussed in Section III, the monitoring station runs the optimization solution and sends the target bitrate for each camera, which in turn produces and transmits the adapted H.264 video stream. In this setup, the monitoring station also provides a controlled patrol movement for each camera. To allow fair comparisons among various allocation solutions, a person in the lab acts according to a predefined script in each evaluation session. The script specifies the paths that must be traversed by the acting person, the standing and walking

TABLE III
CHARACTERISTICS OF THE REAL SURVEILLANCE VIDEOS USED IN EXPERIMENTAL SETUP I

Type	Resolution	Duration (sec)	Frame Rate (fps)	Bitrate (Kbps)
Campus	1280 × 720	27	29	2704
Campus	1920 × 1080	44	23	3145
Campus	1280 × 720	77	30	2149
Office	1920 × 1080	10	30	2317
Office	480 × 360	8	30	343
Office	1280 × 720	57	30	2098
Office	1280 × 720	32	25	2088
Office	640 × 360	42	29	575
Store	480 × 360	57	6	355
Store	370 × 252	23	23	363
Store	1280 × 720	69	23	2550
Street	1280 × 720	14	30	768
Street	1920 × 1080	27	23	4215
Street	1920 × 1080	40	23	4035
Street	1280 × 720	68	29	2199

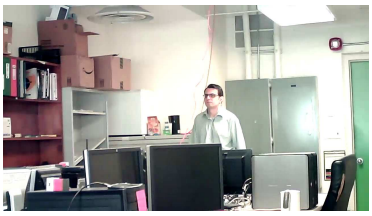
(a) Campus (b) Street (c) Store



(d) Office

Fig. 4. Sample Frames form the Videos in Experimental Setup I

directions, and the time spent on each path. Figure 5 shows sample concurrent views of two PTZ cameras.



(a) Camera 1

(b) Camera 4

Fig. 5. Sample Concurrent Views of Two Cameras in Experimental Setup II

C. Experimental Setup III: Using Videos from Janus Benchmark-B Face Challenge Dataset

Experimental Setup III is similar to Setup I, except for the use of a different dataset. As the dataset used in Experimental Setup I does not contain any ground truth concerning the present faces, we utilize an unconstrained face recognition dataset, specifically the IARPA Janus Benchmark-B Face Challenge (IJB-B) dataset [8]. As the main objective of the face recognition experiment is to compare the performance of different bandwidth allocation solutions under competition for the available effective airtime, we use the following criteria to select surveillance-like video files from the dataset: (a) presence of changing/moving backgrounds, (b) presence of multiple subjects, and (c) presence of a wide range of facial expressions/angles of the main subject in the scene. Table IV summarizes the main characteristics of the selected video files and Figure 6 shows sample frames.

TABLE IV
CHARACTERISTICS OF THE VIDEOS USED IN EXPERIMENTAL SETUP III

Type	File No.	Resolution	Length (sec)	Frame Rate (fps)	Bitrate (Kbps)
Street/ Crowd	626	1280 × 720	6	29.97	2620
Sports	847	1280 × 720	13	25	2607
Street/ Interview	1038	1280 × 720	20	25	3006
Politician Visit	1058	640 × 360	5	25	917
Street/ Person	1668	368 × 300	7	29.97	268

Fig. 6. Sample Frames from the Videos in Experimental Setup III

VI. RESULTS PRESENTATION AND ANALYSIS

A. Tuning System Constants

Let us first discuss how to select the values of the constants of effective airtime estimation, namely the delay weight (DW) and smoothing constant (SC). Figure 7 illustrates how these constants impact face detection accuracy. This number of detected faces increases initially with DW because of the tendency towards producing higher frame rates, leading to reducing the stream bitrates and thus reducing both the contention for the medium bandwidth as well as the data dropping and corruption rate. Ultimately, the perceived frame rate by the monitoring station is increased. After a certain

point, however, the generated high frame rates greatly reduce the stream bitrates and consequently the video quality. Likewise, the number of detected faces increases with SC up to a certain point and then starts to decrease. The increase happens because the smoothing operations enable the system to estimate the effective airtime more accurately, whereas the subsequent decrease is due to aggressive smoothing, which greatly marginalizes the impacts of the momentary values of the stream bitrates. The best values of DW and SC in the considered system configuration are 0.65 and 0.99, respectively.

B. Comparing Effective Airtime Estimation under Different Solutions

Figure 8 shows that the proposed solution produces the largest area under the effective airtime curve; its area is 125% larger than that of the best existing solution and 8% larger than that with disabled optimization. As discussed in Subsection IV-A, the existing solution causes the system to use only a portion of the available bandwidth, thereby reducing the effective airtime to even lower values than those with disabled optimization.

C. Analysis of Cross-Layer Optimization for Face Detection

Let us now analyze the effectiveness of cross-layer optimization in the terms of detection accuracy under Experimental Setup I. Figure 9 shows the number of detected faces versus bandwidth capacity for the entire video dataset and for each video category. The proposed solution achieves 19%, 10%, 10%, and 10% higher accuracy than the existing solution in campus, office, store, and street environments, respectively, and 54%, 45%, 72%, and 69% higher accuracy than disabled optimization. As expected, the number of detected faces generally increases with the medium capacity. The occasional dips in the case of the existing solution are due to the aforementioned problem in utilizing the medium bandwidth. After a certain point, the proposed solution and disabled optimization converge to similar values, when the medium capacity is large enough to accommodate the maximum supported bitrates of the cameras, thereby eliminating bandwidth contention. *In actual systems, the number of employed cameras and the supported bitrates are larger, thereby raising the medium bandwidth at which convergence occurs.* Interestingly, the existing solution performs worse than disabled optimization when the contention among cameras goes below a certain level. Figure 10 compares various solutions in detection accuracy for each camera. Cam1, Cam2, Cam3, and Cam4 refer to the IPCam 7210W wireless PTZ security cameras, whereas Cam5 and Cam6 are the HP Truevision HD and Labtec PRO webcams, which are turned into functioning IP cameras using the VLC media player, and Cam7 is the VivoTek IP7139 camera. The proposed solution consistently performs the best, whereas the existing solution performs even worse than disabled optimization with certain cameras due to its aforementioned problem.

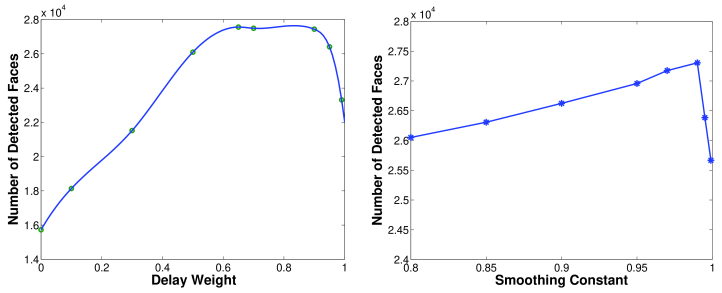
Figure 11 compares various solutions in terms of face detection accuracy under Experimental Setup II. It shows the number of detected faces by the entire CV system and by each camera. Although there is only one person in the scene, the person appears in multiple frames of the video streams. Therefore, having higher quality video streams translates to a larger number of detected faces. The proposed solution achieves 123% higher accuracy than the existing solution and 148% higher than disabled optimization.

Let us now shed light on the dynamics of the system as a result of the interplay of various factors. Figure 12 demonstrates the relationships among different system metrics, namely the average received rate by the monitoring station, effective airtime, frame rate, and the number of detected faces. To effectively display the different values of attributes together in the same chart, attribute values are normalized. The number of detected faces is the most important metric, and indeed the proposed solution continues to hold the lead in that metric. Achieving a high value in this metric depends on two main factors: the effective airtime and the average received frame rate. The proposed solution demonstrates a remarkable balance in improving these two main factors, resulting in producing the highest face detection accuracy. The existing solution becomes a viable choice only when the power consumption is of utmost significance and preferred over accuracy. The results also demonstrate the high effectiveness of cross-layer optimization. The number of detected faces with disabled optimization is considerably lower than that by the two optimization solutions. This behavior can be explained as follows. When the available medium bandwidth is limited and each camera sends at the highest rate without any governing policy, severe congestion in the network will result, thereby greatly increasing the probability of data packet loss and frame dropping and subsequently critically decreasing the received frame rate at the monitoring station. This regulation of bandwidth allocation (via cross-layer optimization) is of great importance, especially when a subset of the cameras have considerably lower physical rates than the rest, thereby causing significant variations in the number of detected faces when no cross-layer optimization is used.

D. Analysis of Cross-Layer Optimization for Face Recognition

Figure 13 compares the numbers of correctly recognized faces achieved by various solutions in Experimental Setup III. The proposed solution achieves 10% and 29% higher accuracy than the existing solution and disabled optimization, respectively. Figure 14 demonstrates the relationships among different system metrics: the average received rate by the monitoring station, effective airtime, frame rate, and the number of recognized faces. The values are normalized.

The proposed solution holds the lead in the number of recognized faces, which is the most important metric. By balancing the average received frame rate, it significantly improves the effective airtime and average received rate, resulting in capturing high-quality frames and the highest face recognition accuracy. The existing solution has the second-best results, with low medium bandwidth usage. Like Experimental Setup I, the existing optimization becomes a viable choice only when the power consumption is of utmost significance



(a) Delay Weight (b) Smoothing Constant

Fig. 7. Effects of the Smoothing Constant and Delay Weight on Detection Accuracy [Experimental Setup I, 15 Mbps Medium Bandwidth]

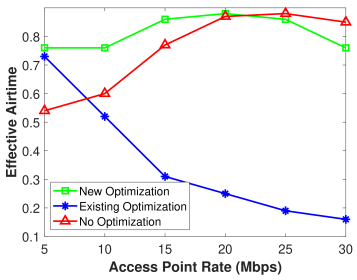
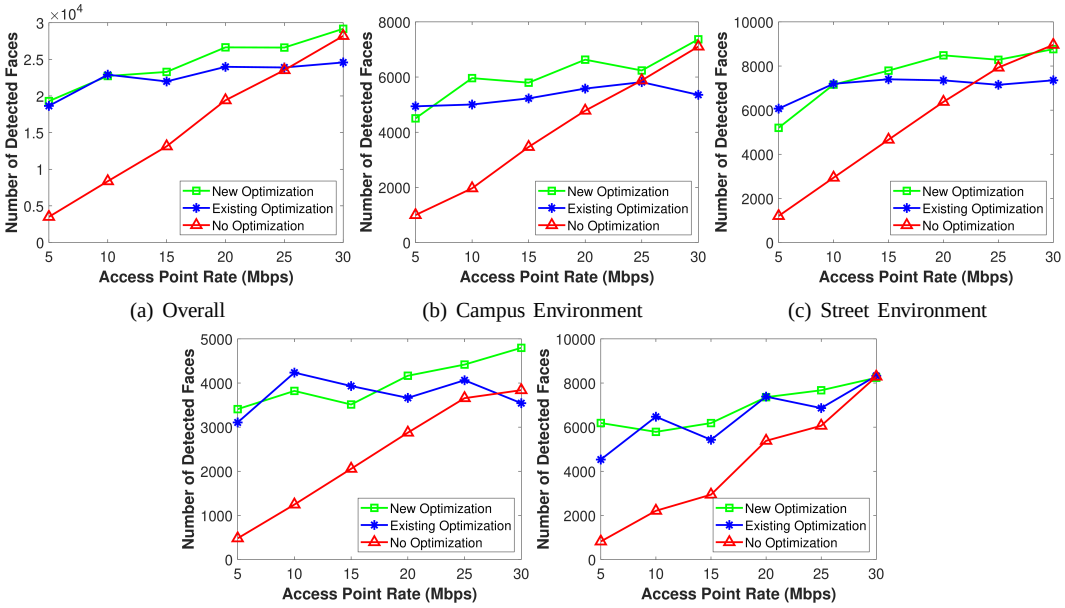


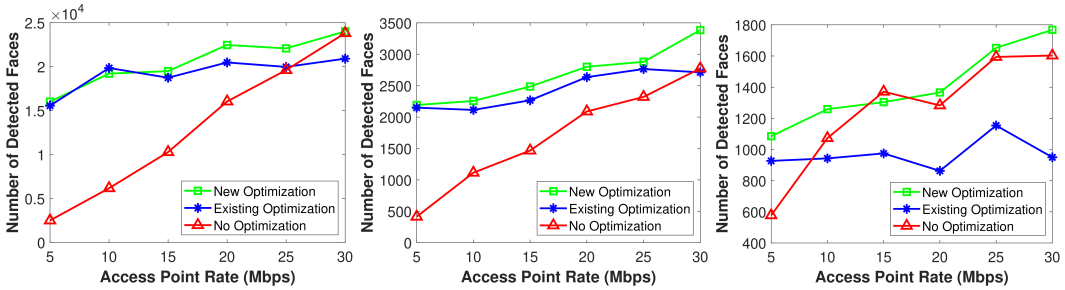
Fig. 8. Comparing Various Solutions in the Overall Effective Airtime [Experimental Setup I]



(a) Overall (b) Campus Environment (c) Street Environment

(d) Store Environment (e) Office Environment

Fig. 9. Comparing Various Solutions for All Video Categories and Each Category [Experimental Setup I]



(a) Combined Cameras 1-4 (IP7210W) (b) Combined Cameras 5 and 6 (Virtualized Web-cams) (c) Camera 7 (VivoTek IP7139)

Fig. 10. Comparing Various Solutions in Detection Accuracy Using the Entire Video Dataset [Experimental Setup I]

and preferred over accuracy. The number of recognized faces when the optimization is disabled is considerably lower than the two optimization solutions.

VII. Conclusions

We have built a real computer vision system for automated video surveillance and have analyzed extensive results of actual experiments using different video datasets as well as in a live laboratory environment. The main results can be summarized as follows. (1) Cross-layer optimization in CV systems is highly effective in improving the detection/recognition accuracy. (2) By optimally distributing the available medium bandwidth and increasing the effective medium airtime, the system can successfully deliver high-quality video streams at

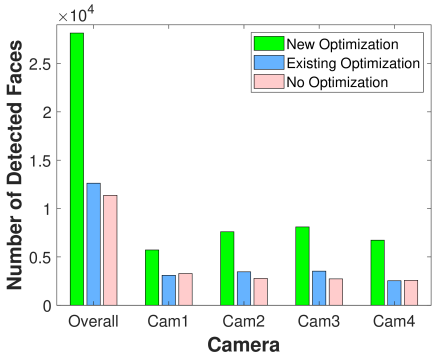


Fig. 11. Comparing Various Solutions in Detection Accuracy in the Live Lab Environment [Experimental Setup II]

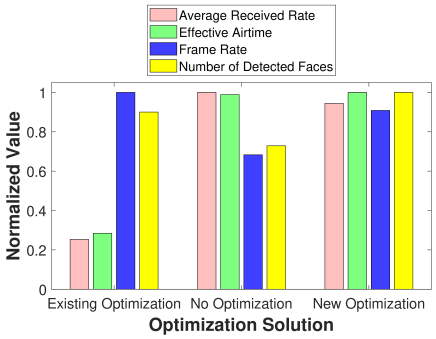


Fig. 12. Relationships Among Different System Metrics [Experimental Setup I]

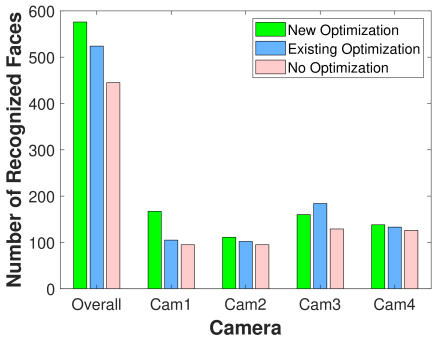


Fig. 13. Comparing Various Solutions in Recognition Accuracy [Experimental Setup III]

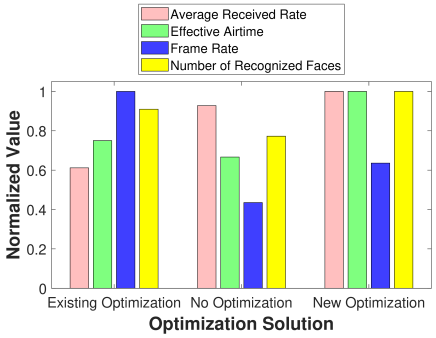


Fig. 14. Relationships Among Different System Metrics [Experimental Setup III]

high frame rates to the monitoring station. (3) The proposed optimization solution significantly enhances the face detection and face recognition accuracy. (4) By properly assessing the overall data corruption and dropping rate through bitrate smoothing, the proposed effective airtime estimation algorithm achieves high accuracy. (5) The highest detection/recognition accuracy is achieved when the packet dropping and error rate is very small. (6) A distributed processing system or one with powerful GPUs is required for the real-time detection of threats when a large number of video sources are supported.

REFERENCES

[1] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," *arXiv*, vol. 2004.10934, 2020. [Online]. Available: https://arxiv.org/abs/2004.10934v1

[2] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," *CoRR*, vol. abs/1503.03832, 2015. [Online]. Available: http://arxiv.org/abs/1503.03832

[3] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "ArcFace: Additive angular margin loss for deep face recognition," in *Proceeding of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4685–4694.

[4] M. Benetti, M. Gottardi, T. Mayr, and R. Passerone, "A low-power vision system with adaptive background subtraction and image segmentation for unusual event detection," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 11, pp. 3842–3853, 2018.

[5] C.-H. Hsu and M. Hefeeda, "A framework for cross-layer optimization of video streaming in wireless networks," *ACM Transactions on Multimedia Computing Communications and Applications*, vol. 7, pp. 5:1–5:28, February 2011.

[6] M. Alsmirat and N. J. Sarhan, "Cross-layer optimization for automated video surveillance," in *Proceeding of IEEE International Symposium on Multimedia (ISM)*, Dec 2016, pp. 243–246.

[7] M. Alsmirat and N. Sarhan, "Cross-layer optimization for many-to-one wireless video streaming systems," *Multimedia Tools Appl.*, vol. 77, no. 19, p. 24789–24811, Oct. 2018.

[8] C. Whitelam, E. Taborsky, A. Blanton, B. Maze, J. Adams, T. Miller, N. Kalka, and A. K. Jain, "IARPA Janus benchmark-b face dataset," in *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017, pp. 592–600.

[9] S. Gholamnejad Davani and N. J. Sarhan, "Experimental analysis of bandwidth allocation in automated video surveillance systems," in *Proceedings of the 25th ACM International Conference on Multimedia*, ser. MM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 1457–1464.

[10] M. H. Sanan, K. A. Alam, M. Z. Rafique, and B. Khan, "Quality of service enhancement in wireless LAN: A systematic literature review," in *Proceeding of 13th International Conference on Mathematics, Actuarial Science, Computer Science and Statistics (MACS)*, 2019, pp. 1–8.

[11] Z. He and D. Wu, "Resource allocation and performance analysis of wireless video sensors," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 5, pp. 590–599, May 2006.

[12] H. Zhang, Y. Zheng, M. A. Khojastepour, and S. Rangarajan, "Cross-layer optimization for streaming scalable video over fading wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 3, pp. 344–353, 2010.

[13] J. Huang, Z. Li, M. Chiang, and A. K. Katsaggelos, "Pricing-based rate control and joint packet scheduling for multi-user wireless uplink video streaming," in *Proceeding of 15th International Packet Video Workshop (PV2006)*, Dec 2006.

[14] J. Tian, H. Zhang, D. Wu, and D. Yuan, "Interference-aware cross-layer design for distributed video transmission in wireless networks," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 5, pp. 978–991, 2016.

[15] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, ser. AAAI'17. AAAI Press, 2017, p. 4278–4284.

[16] H. R. Hamandi and N. J. Sarhan, "Novel analytical models of face recognition accuracy in terms of video capturing and encoding parameters," in *Proceedings of the 2020 IEEE International Conference on Multimedia and Expo (ICME)*, 2020, pp. 1–6.