Critical Video Quality for Distributed Automated Video Surveillance

Pavel Korshunov
School of Computing
National University of Singapore
pavelkor@comp.nus.edu.sg

Wei Tsang Ooi
School of Computing
National University of Singapore
ooiwt@comp.nus.edu.sg

ABSTRACT

Large-scale distributed video surveillance systems pose new scalability challenges. Due to the large number of video sources in such systems, the amount of bandwidth required to transmit video streams for monitoring often strains the capability of the network. On the other hand, large-scale surveillance systems often rely on computer vision algorithms to automate surveillance tasks. We observe that these surveillance tasks present an opportunity for trade-off between the accuracy of the tasks and the bit rate of the video being sent. This paper shows that there exists a sweet spot, which we term critical video quality that can be used to reduce video bit rate without significantly affecting the accuracy of the surveillance tasks. We demonstrate this point by running extensive experiments on standard face detection and face tracking algorithms. Our experiments show that face detection works equally well even if the quality of compression is significantly reduced, and face tracking still works even if the frame rate is reduced to 6 frames per second. We further develop a prototype video surveillance system to demonstrate this idea. Our evaluation shows that we can achieve up to 29 times reduction in video bit rate when detecting faces and 16 times reduction when tracking faces. This paper also proposes a formal rate-accuracy optimization framework which can be used to determine appropriate encoding parameters in distributed video surveillance systems that are subjected to either bandwidth constraints or accuracy constraints.

Categories and Subject Descriptors

I.2.10 [**Vision and Scene Understanding**]: Video Analysis; C.2.4 [**Distributed Systems**]: Distributed Applications

General Terms

Experimentation, Measurement, Performance

MM'05, November 6–12, 2005, Singapore.

Keywords

Video Surveillance, Video Quality Adaptation, Rate-Accuracy Function

1. INTRODUCTION

A large-scale distributed video surveillance system typically consists of many video sources distributed over a wide area, transmitting live video streams to a central location for processing and monitoring. Examples of such systems include J-Eyes and EMAS (which stands for Expressway Monitoring and Advisory System), implemented by the Land Transport Authority of Singapore, where about 1000 cameras are installed at strategic locations to monitor traffic conditions at busy junctions and expressways.

Recent advances in video sensors and the increasing availability of networked digital video cameras have enabled the deployment of such large-scale surveillance systems on top of existing IP-network infrastructure. Many commercial companies now offer IP-based surveillance solutions. Nevertheless, implementing an intelligent, scalable and massively distributed video surveillance system remains a research problem. Some of the open research issues in building such systems have been raised by Feng *et al.* [5].

Researchers in video surveillance systems mainly concentrate on issues related to the autonomy and intelligence of the systems. They focus on the problem of content understanding, such as the detection, tracking and classification of objects [3, 7, 21], and the detection and classification of unusual events [12, 13]. Researchers have paid little attention to the scalability of video surveillance systems. They generally use a centralized architecture and assume availability of all required system resources, such as computational power and network bandwidth.

In this paper, we focus on *distributed video surveillance systems*. Such systems typically consist of *video sources*, *processing proxies* and *monitoring stations* (or *monitors* for short). Video sources are typically devices with minimal computing power, such as sensors and networked cameras. These sources continuously transmit video streams to proxies for processing and filtering. Not every video stream that is sent to a proxy for processing is shown to the end user at the monitor. The proxy analyzes the video and only alerts the user if it detects a suspicious event. Since suspicious events rarely occur compared to benign events [20], most of the time, video streams transmitted from network cameras are meant for analysis by computer vision algorithms running on the proxy rather than for human viewers.

We observe that *commonly used computer vision algorithms in a video surveillance environment can perform just as well with video of low SNR and temporal quality as with video of higher quality.* This observation allows us to reduce bandwidth usage in such systems. Encoding and transmitting lower quality video also means that less computing power is required at video sources, leading to lower power consumption.

To understand the issue deeper, we consider two related questions: (i) How much bandwidth can we save? (ii) How will the accuracy of vision algorithms be affected? There is an obvious trade-off between accuracy of the algorithms and quality of the video. Our goal is to identify the sweet spot in the trade-off curve, i.e., a point to which the quality of the video can be reduced, without affecting the accuracy of the surveillance task, and below which the accuracy of the task will be reduced significantly. We call this sweet spot *critical quality.* A general framework can be developed, similar to rate-distortion optimization and utility-based adaptation, to find the desired video encoding parameter given the tolerance in errors. We introduce a mathematical framework called *rate-accuracy optimization* in this paper.

To study how the framework can be realized in practice, we choose to focus on two fundamental operations in automated video surveillance – face detection and face tracking, and find their critical video quality. We perform extensive experiments on a standard test data set and data captured from a real surveillance environment, using the Viola-Jones face detection algorithm [19] and the CAMSHIFT face tracker [2]. The experiments show that the Viola-Jones face detection algorithm has a critical SNR quality of 20 for MJPEG encoding and the CAMSHIFT face tracker has a critical temporal quality of 6 frames per second (fps). Using the knowledge of these critical quality values, our prototype surveillance system can achieve up to 29 times and 16 times reduction of the transmitted video bit rate for face detection and face tracking respectively.

The following are the contributions of this paper:

- We identify an opportunity for significant reduction in bandwidth and resource usage when a video is transmitted for analysis by vision algorithms and not for human viewers.
- We study two algorithms used in video surveillance, namely, face detection and tracking, and show through extensive experiments that a sweet spot exists for trade-off between accuracy of the algorithms and resource usage.
- We present a general mathematical framework, termed rate-accuracy optimization, that allows minimization of bit rate under accuracy constraints, or maximization of accuracy under bit rate constraints.
- We implement a prototype for video surveillance that incorporates adaptation based on critical video quality. Based on the prototype, we report actual performance improvement in a practical surveillance environment.

The rest of this paper is organized as follows. Section 2 gives an overview of related work in distributed video surveillance. Section 3 presents the architecture of the video surveillance system used in this paper. Section 4 describes our experiments to determine critical video quality for face detection and face tracking algorithms. Section 5 proposes a general rate-accuracy optimization framework, capturing the trade-off between video bit rate and accuracy of a computer vision algorithm. Section 6 presents the working prototype of a real video surveillance application. We conclude in Section 7.

2. RELATED WORK

This section presents some related work in the literature, focusing on similar work from the video surveillance and video streaming community.

Most research in video surveillance aims at developing robust computer vision and classification algorithms [8, 12, 7, 3]. Other proposals for distributed surveillance systems do not consider system issues such as efficiency and scalability in depth. For instance, video surveillance systems KNIGHT [6] and SfinX [16] transmit video with fixed encoding parameters to a central server for processing. Commercial system DETER [15] uses a dedicated network for streaming high quality video. Yuan *et al.* [21] and Nair *et al.* [12] presented systems which avoid using excessive network bandwidth by sending still images from a video source to the end user periodically. VSAM [3] deals with bandwidth constraint by sending only one low quality video at a time, and relies on workstations attached directly to video sources for the detection, tracking and classification of events.

Apart from research in video surveillance, our work is similar to many techniques proposed to adapt video transmission rate to meet the bandwidth constraints of wide area networks. One of the first suggested methods, presented by Eleftheriadis and Anastassiou [4], uses a rate-distortion function to find minimal distortion. Based on the bandwidth capacity predicted via monitoring the current state of the network, the video is dynamically reshaped by being encoded with different quantization values. Extending this idea, Kim and Altunbasak [10] suggested a technique to reshape video by scaling its spatial, temporal and SNR properties. This technique was later generalized into a utility-based framework by Kim *et al.* [9]. Our work in this paper is analogous, but we focus on the case where the video is to be consumed by automated surveillance operations rather than by human viewers.

Region of interest (ROI) is another technique to reduce video transmission rate. This technique transmits only important regions in video frames at high quality [18, 17]. For instance, video sources can stream those regions with faces in higher quality to proxies. Implementation of ROI in a video surveillance system, however, requires a significant level of intelligence and more computing power at video sources.

A recent work that is most related to ours that of Boyle on the effects of capture conditions on the CAMSHIFT face tracker [1]. That study aims to recommend how to set up a low-end web-cam for face tracking on desktop computers. Similar to our experiments, the study examines the effects of frame size, frame rate and compression quality on the CAMSHIFT face tracker. Its experiments, however, focus on a few values for each of these quality only, and do not explicitly address the issue of critical video quality.

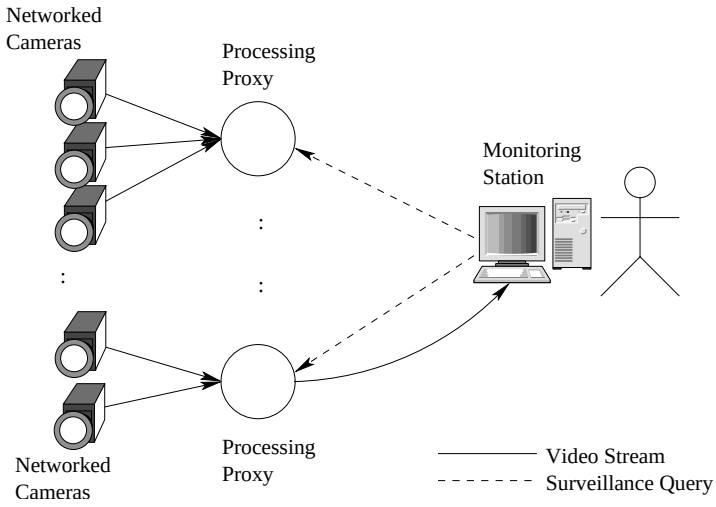

Figure 1: Architecture of Distributed Video Surveillance System.

3. SYSTEM ARCHITECTURE

Before we describe our main results on critical video quality, it is important to understand the system architecture that we are interested in. Our work focuses on distributed video surveillance systems, which consist of a number of video sources, processing proxies, and monitoring stations, connected via a wide area network. Video sources can be either networked cameras or video sensors. These sources capture, encode and transmit video streams to processing proxies. Processing proxies are computers dedicated to the processing and filtering of incoming video streams, and if needed, relaying them to monitoring stations. The need to relay depends on the queries specified by users. For instance, a user may request to see a certain video if suspicious events are detected. A sample query is "Show me the video of secured room X if someone is detected in the room." A video source sends surveillance video from room X to a remote proxy. The proxy then runs a motion detection algorithm on the surveillance video. The proxy relays the video to the monitor only if motion is detected in the room. Figure 1 shows the architecture of such a distributed surveillance system.

Using such a distributed architecture for video surveillance has several advantages. First, it allows flexibility in adding and removing cameras. Second, since video processing is done at the proxies, cheap networked cameras or video sensors can be used as video sources. Finally, by filtering uninteresting video at the proxies, the number of streams to be sent to the monitoring station is kept small, thus increasing the scalability of the video surveillance system. Due to these advantages, this type of architecture is becoming common in commercial video surveillance systems (e.g., ObjectVideo[1], MOXA[2]).

Note that our architecture does not consider archiving full quality video from video sources. While such archives would be useful for forensic video analysis, performing continuous archiving of full quality video from large number of video sources does not scale. Our architecture, however, does not preclude archiving videos at monitors.

[1]http://www.objectvideo.com
[2]http://www.moxa.com

4. CRITICAL VIDEO QUALITY

The architecture described in Section 3 opens the opportunity to reduce network bandwidth usage at the links between proxies and video sources. The basic idea is that a surveillance task, driven by some computer vision algorithms running on a proxy, may only require low quality video to achieve its goal. This low quality requirement contrasts with the high quality required for viewing by the end user at the monitor. Due to the rarity of suspicious events, most of the time, the system does not need to send video to the monitor. Therefore, video sources only need to encode and transmit low quality video streams onto the links between the sources and the proxies. However, if we reduce the quality too much, the accuracy of the vision algorithms might be affected. In this paper, we study how low the bit rate of a video stream can be, before it affects the accuracy of the vision algorithms. In other words, we are interested in finding the critical video quality, which achieves the best trade-off between the accuracy of the surveillance task and the bit rate of the video. The value of the critical video quality, however, depends on the algorithms. In this paper, we choose to study two commonly used surveillance operations, object detection and object tracking, with faces as target objects. For face detection, we are interested in critical SNR quality of the input. For face tracking, we study both critical temporal quality and critical SNR quality.

We note here that we are not concerned with studying the performance of a particular algorithm on full quality video. Such studies should be investigated by the designers of the algorithms. We are concerned with how *different* the algorithms behave when we degrade the video quality. We use the term *accuracy* here to represent consistency in the performance of an algorithm when we run it on a video with decreased quality, compared to its performance when we run it on the same video with full quality. For instance, even if algorithm A can perform a task only 60% of the time on a full quality video, as long as A can perform the same task 60% of the time on a degraded quality video, we consider the algorithm to be 100% "accurate".

4.1 Face Detection

To study the trade-off between the accuracy of face detection and SNR quality of the video, we pick the Viola-Jones face detection algorithm [19] implemented in open source library OpenCV[3]. The Viola-Jones algorithm is an object detection algorithm that uses a cascade of classifiers based on Haar-like features, which include edges, lines and center-surround features. Therefore, this face detection algorithm can perform accurately with highly compressed images, as long as the compressed image contains such features. In the rest of this subsection, we present extensive experimental results on two data sets to support this point.

4.1.1 Experiments on MIT/CMU data set

We choose to study face detection because standard sets of test images with ground truth (i.e., manually tagged location of faces) are available. For our experiments, we use the image data set provided by MIT/CMU. We compress each image in the data set with different JPEG quality, ranging from 1 to 100, using the standard JPEG library from Independent JPEG Group, and run the Viola-Jones algorithm on the resulting images. We record the number of faces detected in each case. Using the provided ground truth, we obtain the number of correctly detected faces and divide it by the total number of faces to get the *detection index*. We also note the number of faces that are wrongly detected by the algorithm. Dividing this number by the total number of faces, we obtain the *false positive index*. The experimental results for a total of 507 images in the set are shown in Figure 2.

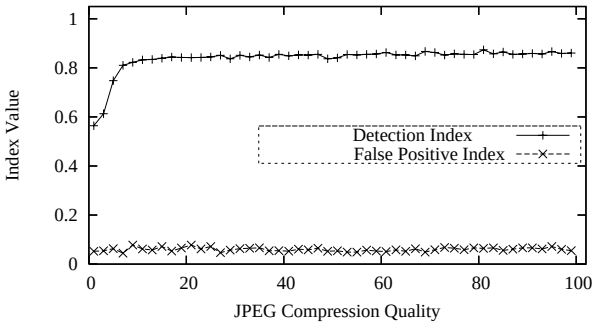

Figure 2: Accuracy of Face Detection Algorithm vs. JPEG Compression Quality.

Figure 2 plots both the detection index and the false positive index of the face detection algorithm against different compression quality. It shows that the average accuracy of the face detection algorithm does not change significantly for JPEG compression quality value ranging from 100 to 7. At the same time, the false positive index does not show any significant increase for all values of the quality. Reduction to compression quality below 7, however, shows a sharp decrease in the detection index. This observation suggests that we can send video of compression quality 7 to achieve similar detection results as with video of compression quality 100. To be conservative, we choose 20 as the critical video quality. The average file size of JPEG images in the MIT/CMU data set is 135.6 kb for compression quality 100 and 15.8 kb for compression quality 20. These numbers translate to 9 times reduction in the bandwidth, if we transmitted these images. This reduction, however, does not directly apply to video since video encoders typically use motion estimation between frames to achieve higher compression. We will provide the corresponding numbers for video in Section 6.

Figure 2 does not reflect fluctuations in the detection pattern of the algorithm. The Viola-Jones face detection algorithm is sensitive to factors such as face size, lighting, background conditions, etc. These factors can cause the algorithm to oscillate between detecting and not detecting faces as we vary the compression quality. To analyze this type of behavior, we compute the cumulative distribution function (CDF) on the maximum JPEG quality which causes the face detection algorithm to fail. The resulting CDF is shown in Figure 3.

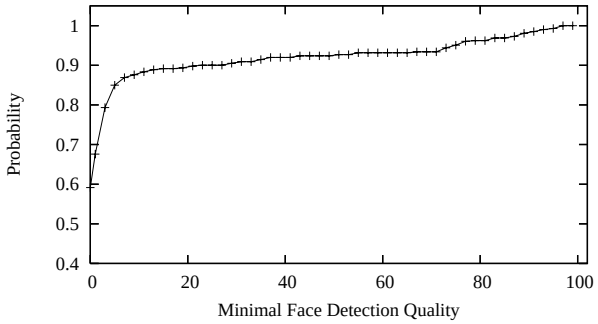

Figure 3: CDF for Minimal Face Detection Quality.

The CDF curve in Figure 3 is computed as follows: For each face, we find a compression quality q such that the face is not detected when compressed with quality q, but is detected when compressed with any quality larger than q. We call this compression quality the *minimal face detection quality*, and regard it as a random variable for Figure 3.

The CDF function shows relatively large decline (about 9% or 45 faces) when compression quality decreases from 100 to 20. This figure demonstrates inconsistent detection results for 9% of the faces. Combining this result with observation from Figure 2, we can deduce that in this subset of faces, the faces are constantly changing from being detected to not being detected as we vary the quality. For instance, one particular face is detected for compression quality 77, 76, 74 and 73, but is not detected for value 75. These faces are found to be of a smaller size. We re-plot Figure 3 for different face size, measured as the maximum of distances between left eye and right eye, and between eye-line and mouth. The new plot, Figure 4, shows that the algorithm tends to maintain more consistent detection behavior for faces that are larger than 30 pixels.

The reason for the fluctuations in detection lies in the reliance of the algorithm on different threshold values. These values affect the detection sensitivity of the algorithm for the faces in the input images. Slight changes in the pixel values of an image due to compression can unpredictably affect the decision of the algorithm on faces that are near the threshold. Further experiments support this explanation: We increase two main threshold parameters: the pruning value P for face candidates and a threshold T that is used inside the cascade classifier. We change these values from the default

[3]http://www.intel.com/research/mrl/research/opencv/

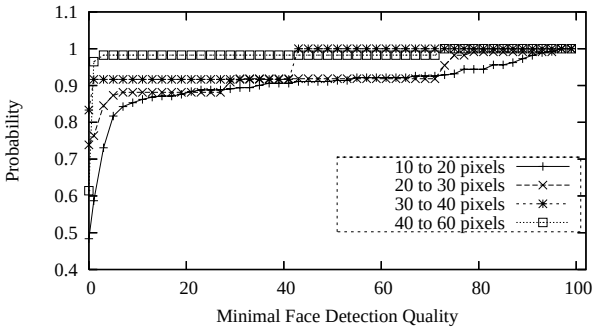

Figure 4: CDF for Minimal Face Detection Quality for Different Face Size. P=3, T=-0.0001.

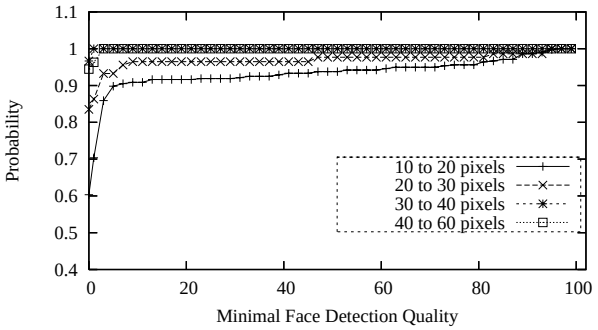

Figure 5: CDF for Minimal Face Detection Quality for Different Face Size. P=4, T=-1.0.

values of P=3 and T=-0.0001 to P=4 and T=-1.0. Figure 5 shows the new results. The consistency of detection across different compression quality values improves. The parameters, however, increase the sensitivity of the algorithm, causing the number of false positives to increase by five times. Interestingly, using the new parameters, the number of false positives is not significantly affected by compression quality.

4.1.2 Experiments on Surveillance of Lab

The MIT/CMU data set consists of images taken from magazines, posters, television, etc., and does not fully reflect the types of images expected in a real surveillance environment. To strengthen our results, we install a video camera pointed to the door of our research lab. We record video of people entering and exiting the door at 5 fps. This environment has a typical office background. Among the 22,000 frames recorded, we find 237 faces, consisting of 138 frontal faces and 99 profile faces. Note that the Viola-Jones algorithm is designed for frontal faces only. Therefore, it may not detect profile faces. Examples of the recorded frames are shown in Figure 6. We run the Viola-Jones face detection algorithm on these faces, compressed to two different quality values, 90 and 20, using two sets of threshold parameters $P = 3, T = -0.0001$ and $P = 4, T = -1.0$.

From the experiments, we find that the Viola-Jones algorithm exhibits unpredictable behavior temporally, returning false positive results periodically. We exclude such false positives by considering a face as detected only if it is detected consecutively for three frames. This method is reasonable because faces are usually present in a video in a consecutive sequence of frames (assuming sufficiently high frame rate). Table 1 presents the detection and false positive indexes for the algorithm with the two sets of parameters. This result verifies that both the detection and false positive indexes do not change significantly when compression quality is reduced to 20. The results obtained for real surveillance are consistent with our findings on the MIT/CMU data set.

	P=**3**,T=**-.0001**	P=**4**,T=**-1.0**
detection index		
quality 90	0.63	0.77
quality 20	0.61	0.76
false positive index		
quality 90	0.004	0.09
quality 20	0.01	0.11

Table 1: Experiments with Face Detection Algorithm and Actual Surveillance Image Set of 237 Faces.

Figure 6: Examples of Images Captured in Surveillance of Lab.

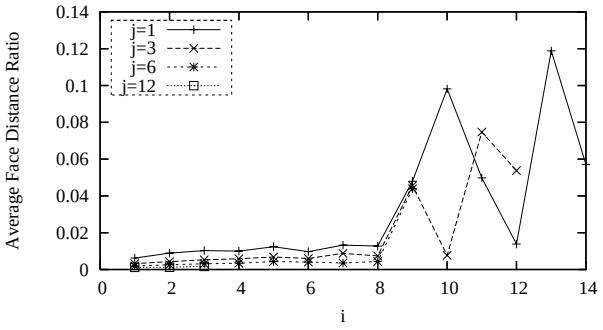

Figure 7: Average Face Distance Ratio for Different Drop Patterns (Compression Quality 100).

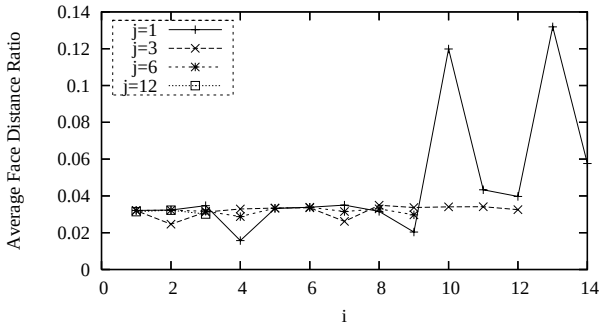

Figure 8: Average Face Distance Ratio for Different Drop Patterns (Compression Quality 50).

4.2 Face Tracking

In this section, we study the trade-off between accuracy of the face tracking algorithm, and the temporal and SNR qualities of the video. We choose to use the CAMSHIFT object tracking algorithm [2] implemented in OpenCV library.

We run the algorithm on a video with different frame dropping patterns and different compression quality. Due to a lack of standard test data sets, we capture our own face using a web-cam. The captured video is 20 seconds in length, and has a resolution of 352x288 and a frame rate of 30 fps. We also tested face tracking on some movie clips. For each test video, we change the video frame rate from 2 fps to 30 fps by dropping frames from the original video using the following pattern: "Drop i consecutive frames for every $i+j$ frames." We vary i and j from 1 to 14. The value i represents the gap between frames. For example, if we drop every third frame, the value of i is 1; when three consecutive frames out of nine frames are dropped, i is 3. Note that while these two patterns give the same average frame rate, the accuracy of the tracking algorithm can be different.

During the experiments, we record the mean distance between the center of the tracked face in a video with reduced frame rate, and the center of the tracked face in the original 30 fps video. We use the ratio of the mean distance and half the average diagonal of the tracking rectangle as a metric of accuracy of the tracking algorithm. We call this metric the *average face distance ratio*. The smaller the average face distance ratio, the larger the accuracy of the face tracking algorithm.

Figure 7 shows the average face distance ratio for one of the test videos for patterns with i varying from 1 to 14 and j equal to 1, 3, 6 and 12. The figure shows that i plays a more important role in the accuracy of the tracking algorithm compared to j.

We can see from Figure 7 that accuracy decreases slowly with increase in i value. For each value of i, accuracy decreases slowly as j decreases. The figure also shows that when i is more than 8, the algorithm behaves unpredictably. This behavior is caused by a large number of consecutively dropped frames, which can lead to one of the following two situations: (i) the tracked face moves far away from the original position, leading to errors in the algorithms; (ii) the face moves back to the position of the previous frame, providing for an accurate result. Therefore, accuracy oscillates. We expect the oscillation to be highly dependent on the content of the video.

The above observations point out the significance of value i for the accuracy of the tracking algorithm. In the video used for Figure 7, the gap i should be bounded by 8 for tracking to be consistent. Therefore, the algorithm can achieve reasonable accuracy (within 2 pixels) using the pattern: "Drop 8 frames out of 9 frames." In other words, the video source only needs to send at 1/9 the original frame rate.

Next, we study the effect of SNR quality on accuracy of face tracking. We compress the video used above using the Microsoft Video 1 codec included with Microsoft Windows, with compression quality 85, 75, 65, 50, 45, 35, 25 and 10, and repeat the experiments. The results for video with compression quality 50 are shown in Figure 8. We can see that accuracy is lower on average for video of higher compression ratio. An increase in compression ratio leads to an increase in average face distance ratio since highly compressed video has fewer details, making the border of a tracked face less distinct.

The results reported above come from experiments on a single video, captured using a web-cam in a normal office environment. We repeat the experiments for different videos with different contents. The results are summarized below.

We first repeat the experiments on a movie clip showing a person talking, moving his hands occasionally. Not surprisingly, this scene only shows 5% errors even when compressed with quality 10 and sent 1 out of 15 frames. On the other hand, for a movie clip showing a character moving his head constantly in a fast and jumpy motion, the critical temporal quality is found to be 6 fps (sending 1 out of 5 frames). This critical temporal quality is consistent for all compression quality. We further repeat this experiment on video captured from a web-cam in different lighting conditions. For each of these videos, we find the *critical drop gap*, i.e., the value of i (for dropping i out of $i+1$ frames), in which further increase will cause the tracking to become unpredictable. The results are shown in Figure 9. From this figure we can see that compression quality does not affect accuracy of the tracking algorithm significantly. The performance of the algorithm depends mainly on type of face motions and lighting conditions. Considering that jumpy motions are not likely to occur in an office environment, we conservatively suggest using $i = 4$ as critical temporal quality, leading to a frame rate of 6 fps.

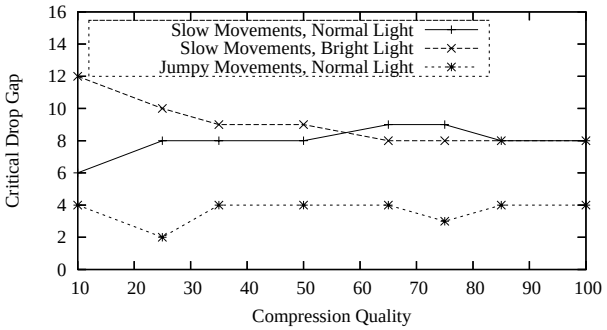

Figure 9: Critical Drop Gap vs. Compression Quality.

5. RATE-ACCURACY FUNCTION

Our experiments on the Viola-Jones face detection algorithm and the CAMSHIFT face tracker show that critical video quality exists in these two algorithms, thus allowing significant reduction in video quality without affecting the accuracy of the algorithms. In this section, we generalize our experimental findings by introducing the rate-accuracy function for a given computer vision algorithm.

The rate-accuracy function of a vision algorithm, under an environmental condition, gives the minimum rate of a video stream that satisfies the required accuracy of the algorithm. Deriving this function is non-trivial, as the rate of the stream depends on three different quality dimensions: SNR, temporal and spatial, each having different effects on different computer vision algorithms. For instance, the accuracy of a face detection algorithm mainly depends on the SNR quality of video while the accuracy of an object tracker is affected by the temporal quality as well.

Let the accuracy a of a computer vision algorithm $\mathcal{A}$ under an environmental condition e be a function g of the quality of the video, which consists of temporal quality τ, compression quality γ, and spatial quality σ. The environmental conditions can be discrete values chosen from a pre-defined set, such as dark, bright, outdoor, etc. We can express the accuracy function as:

$$a = g_{\mathcal{A},e}(\tau, \gamma, \sigma)$$

We can obtain the function g through off-line profiling of the algorithm.

The rate of the streaming video, R, directly depends directly on the video quality based on some function r, i.e.:

$$R = r(\tau, \gamma, \sigma)$$

There are two different problems that may be of interest in the context of video surveillance. The first is to minimize the transmitting video bit rate for the given accuracy a' of an algorithm $\mathcal{A}$. Let $S(a')$ be the set of solution triples $(\tau', \gamma', \sigma')$ that satisfies the equation:

$$a' = g_{\mathcal{A},e}(\tau, \gamma, \sigma)$$

Each triple, or combination of encoding parameters, leads to different possible video bit rates, among which we should find one that minimizes the bit rate. Denoting $R_{\mathcal{A}}(a')$ as the minimal bit rate of a video satisfying the accuracy a' of algorithm $\mathcal{A}$, we can express:

$$R_{\mathcal{A}}(a') = \min_{(\tau',\gamma',\sigma') \in S(a')} (r(\tau', \gamma', \sigma')) \quad (1)$$

As an example, let algorithm $\mathcal{A}$ be the CAMSHIFT face tracker, and accuracy a (measured as average face distance ratio) be 0.3. Based on the experimental findings partly presented in Figure 7 and Figure 8 in Section 4.2, the accuracy, subject to environmental conditions, can be achieved with videos encoded using the parameters shown in Table 2. For example, the first row of Table 2 shows that we can achieve the accuracy of 0.3 with SNR quality of 100, and frame rates between 3.3 fps and 30 fps. This set of encoding parameters yield a resulting bit rate ranging from 0.2 Mbps to 2 Mbps. Note that videos of compression quality below 50 cannot satisfy the required accuracy of 0.3.

From the above set of videos satisfying the specified accuracy requirement, we can find the minimum bit rate according to Equation 1. This minimum rate is achieved using video of compression quality 50 and temporal quality 3.75 fps, resulting in a minimum bit rate of 12 kbps.

Besides minimizing the bit rate, the set of equations above also gives us a framework for dynamic video rate adaptation, with the goal of maximizing the accuracy of a given algorithm. Such formulation is similar to the utility-based adaptation framework presented by Kim *et al.* [9]. In their framework, the rate is constrained, and the goal is to find the maximum quality of the video based on human perceptions. In our context, instead of maximizing the video quality, we need to maximize accuracy of the algorithm. The problem becomes the following: Find a combination of encoding parameters to maximize the accuracy of the algorithm when the available bandwidth is less than B:

$$a_{max} = \max_{\tau,\gamma,\sigma}(g_{\mathcal{A},e}(\tau, \gamma, \sigma))$$

$$\text{subject to } r(\tau, \gamma, \sigma) \leq B$$

In the case where several computer vision algorithms need to be performed on the same video source, the resulting video should have the quality to satisfy the most quality-sensitive algorithm among them. The maximum of temporal, spatial and SNR qualities among all values that meet the accuracy of all these algorithms should be used. If there is a constraint on network bandwidth, priority can be assigned to each operation. Taking the priorities into account, the resulting video rate can be adjusted by solving the max-min problem with varying SNR, temporal and spatial qualities. Further exposition of this theoretical framework is outside the scope of this paper.

6. PROTOTYPE OF THE VIDEO SURVEILLANCE SYSTEM

Once the rate-accuracy function of a particular operation is known (through off-line profiling), a video surveillance system can dynamically adjust the rate of streaming video. When the result of the surveillance task does not meet the query criteria (e.g., no face is detected), the video source can stream low quality video from the video source to the processing proxy. In this case, the proxy would be in "observe" mode, continuously running video analysis algorithms on low quality video without relaying it to the monitor. Once an algorithm detects something in the video, the proxy requests the video source to raise the quality of the video and

Compression Quality	Min FPS	Max FPS	Min Bit Rate	Max Bit Rate
100	3.3	30	0.2 Mbps	2 Mbps
75	3.3	30	70 kbps	0.6 Mbps
50	3.75	30	12 kbps	100 kbps

Table 2: Profiles of Video Matching Required for Face Tracking Accuracy of 0.3.

relay it to the monitor, thus alerting the end user. In this scenario, the proxy would be in "alert" mode. In observe mode, usage of network bandwidth between proxy and video source is minimized while in alert mode, full quality video is transmitted from video source to monitor.

To demonstrate the feasibility of our experimental findings in a real environment, we build a prototype video surveillance system that can operate in the two modes above. The prototype has one video source: a Canon VCC4 camera connected to an LML33 capture card, one computer as a processing proxy, and another computer serving as a monitoring station. To transmit and display video, we use the OpenMash[4] framework. We use an OpenMash extension called Indiva [14] to remotely control the compression quality and frame rate of the video captured from the camera. At the proxy, we run the Viola-Jones face detection algorithm on incoming video from the camera.

The experiments on the prototype system are carried out in an office-like environment. We use video of size 352x288. Faces appearing in a video generally have eyes, nose, and mouth within a 20x20 pixels square. We run our system in several scenarios for both the MJPEG and H.261 video encoders, the two main encoders available in OpenMash. Nevertheless, our experiments can be carried out in a similar way on other modern codecs such as MPEG and H.263.

To verify our experimental findings presented in Section 4.1, we run our system with changing compression quality every three seconds, ranging from 90 to 1 and decreasing by 2 every time. We use scenarios where one person is sitting in front of the camera, moving her head and talking. The sample shots are shown in Figure 10(c) and Figure 10(d). We run the system in such scenarios eight times each, using the MJPEG and H.261 encoders. For faces that have eyes, nose and mouth within a square of 10x10 pixels size (e.g., Figure 10(d)), the detection index demonstrates unpredictable fluctuations. Faces that are bigger in size (e.g., Figure 10(c)) are correctly detected at least until compression quality is reduced to 15. These observations are consistent with our experimental results on images from both the MIT/CMU data set and our own lab surveillance.

Our prototype system can dynamically adapt the bit rate for surveillance video according to the current result of the face detection algorithm. When no face is detected, the system runs in observe mode, using only a small amount of bandwidth. Video is compressed with quality equal to 20, and the proxy does not relay it to the monitor. Once a face is detected, the system automatically switches to alert mode by changing compression quality to 90, and relays the video to the monitor to alert the user. The system switches back to observe mode when no face is detected. We run the prototype on a video scene with a person walking in and out of the camera's view. The sample shots of the video used are shown in Figure 10(a) and Figure 10(b). The system successfully detects faces and changes to alert mode in accordance with our experimental findings.

We collect the bit rate for the MJPEG and H.261 encoders during a period of 100 seconds. The collected data is shown in Figure 11 and Figure 12. The figures show that when there are no faces detected, i.e., the compression quality is reduced to 20, the bandwidth on average is reduced up to 94% for the H.261 encoder and up to 72% for the MJPEG encoder. The H.261 encoder demonstrates higher reduction in bandwidth for videos with static background due to its conditional replenishment algorithm [11]. The important thing to note is that the frame rate remains at 30 fps throughout the experiment. Since the frame rate of the video is less important for face detection, we can further reduce the frame rate to 5 fps in observe mode. By doing so, we obtain bandwidth reduction of up to 35 times for the H.261 encoder and up to 29 times for the MJPEG encoder. The above experiments are conducted on a video scene with static background. In our experiments on video scene with intensive background motions, the effect of motion on bandwidth reduction is significantly reduced, showing mainly the effect caused by a decrease in compression quality. With these conditions, we can still obtain up to six times bandwidth reduction for the H.261 encoder. For MJPEG, there is no significant differences in the bandwidth measurement since the MJPEG format is not motion compensated.

In similar experiments on the CAMSHIFT face tracker, the way the tracking algorithm was used in our prototype is different. Usually, the tracking algorithm is used to support higher level tasks such as detecting suspicious behavior, identifying a running or falling person, group tracking, etc. Therefore, the decision whether to stream video to the user or not would be made by those algorithms. We do not implement such high level algorithms. Therefore, instead of

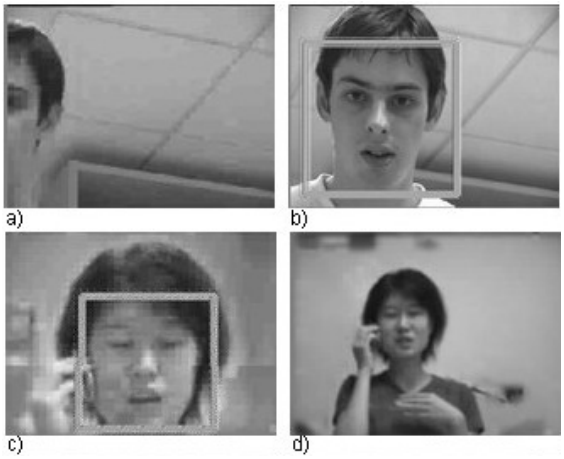

Figure 10: Sample Video Shots used in Experiments on the Prototype Video Surveillance System.

[4]www.openmash.org

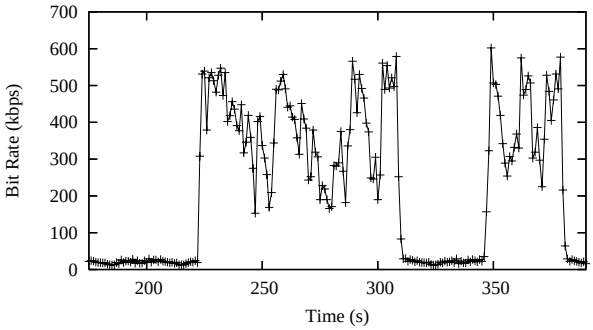

Figure 11: Video Bit Rate when a Face Comes In and Out of the Camera's View (H.261).

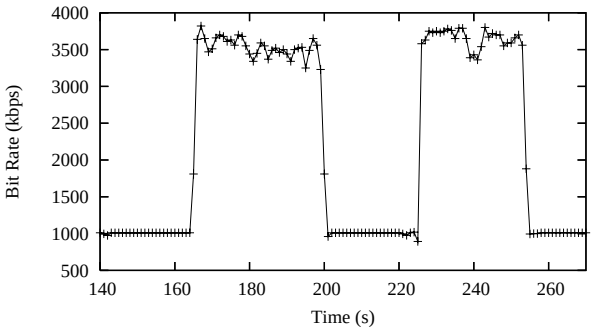

Figure 12: Video Bit Rate when a Face Comes In and Out of the Camera's View (MJPEG).

switching between observe mode and alert mode, we simply run the tracking algorithm on the video with the suggested critical video quality of compression 50 and frame rate of 6 fps. Such settings lead to an MJPEG bit rate of 175 kbps on average, giving us 16 times reduction in the bandwidth.

A possible concern is the latency caused by switching from observe to alert mode. Such latencies might cause high quality video frames of suspicious events to be lost. To address this concern, we measure the latency between when a face is detected, and when high quality video is received at the monitor in our prototype. This delay is found to be at most 100 ms. A caveat is that our prototype system runs over a local area network. This latency might increase if the system is deployed over a wide-area network.

7. CONCLUSION

Our extensive experiments on face detection and face tracking have shown that sweet spots for trade-off between accuracy of these algorithms and video bit rate exist. In this paper, we have proposed using such sweet spots, termed critical video quality, to significantly reduce the bit rate of video between video sources and processing proxy in a distributed video surveillance system. Such reduction allows the system to efficiently use its network, computing and power resources without degrading the accuracy of vision algorithms significantly. We have further proposed a generic framework for the rate-accuracy function, which allows the system to automatically decide the minimal rate at which to send video, given the required tolerable inaccuracy. The framework also allows the system to find encoding parameters to achieve the highest accuracy if the bandwidth is constrained. Finally, we have presented a prototype video surveillance system implementing the idea, and shown that we can reduce bandwidth usage up to 29 times when detecting faces, and up to 16 times when tracking faces.

In this paper, we have concentrated on two specific algorithms for face detection and face tracking. We believe that critical quality exists for other video surveillance algorithms, such as those for motion detection, face recognition, gait analysis, etc. Identifying the critical video quality for these algorithms remains an open problem.

Another problem that remains open is the study of critical spatial quality for vision algorithms. We have shown that a face detection algorithm can behave unpredictably with small faces. The size of the faces, however, depends on the scene captured, not on the video encoding. We plan to study how we can exploit the pan, tilt and zoom features of cameras to automatically adjust the spatial quality of faces to increase the accuracy of computer vision algorithms.

We hope that the findings from this work can inspire researchers in the areas of multimedia systems, video coding and video analysis in the following ways: In the area of distributed multimedia systems, we have shown that when a video stream is meant to be processed by software and not viewed by humans, the strategy and design space for resource management can be different. In the area of video coding, our results raise the question of whether one can design a new video compression algorithm specifically optimized for vision operations. Such a new algorithm may give better performance than the existing video compression schemes which are designed and optimized for human perception. In the area of video analysis, our work shows that it is important to design a video analysis algorithm that works well on low quality video. We believe that such quality-tolerance algorithms play a crucial role in building scalable and efficient distributed video surveillance systems.

8. ACKNOWLEDGMENTS

We would like to thank our anonymous reviewers, our shepherd Eckehard Steinbach, our copy editor Alexia Leong and members of NeMeSys Research Group for their thoughtful comments and suggestions to improve the paper. We are also thankful to Lu Liming, Liza Marchenko and members of Multimedia Information Systems Lab for their kind participation in our experiments.

9. REFERENCES

[1] M. Boyle. The effects of capture conditions on the CAMSHIFT face tracker. Technical Report 2001-691-14, Department of Computer Science, University of Calgary, Alberta, Canada, 2001.

[2] G. R. Bradski. Computer vision face tracking as a component of a perceptual user interface. In *Proceedings of the Forth IEEE Workshop on Applications of Computer Vision, WACV'98*, pages 214–219, Princeton, NJ, January 1998.

[3] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, , and O. Hasegawa. A system for video surveillance and monitoring. Technical Report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, May 2000.

[4] A. Eleftheriadis and D. Anastassiou. Constrained and general dynamic rate shaping of compressed digital video. In *Proceedings of the IEEE International Conference on Image Processing, ICIP'95*, pages 396–399, Washington, DC, USA, October 1995.

[5] W. Feng, J. Walpole, W. Feng, and C. Pu. Moving towards massively scalable video-based sensor networks. In *Proceedings of the Workshop on New Visions for Large-Scale Networks: Research and Applications*, pages 12–14, Washington, DC, USA, March 2001.

[6] O. Javed, Z. Rasheed, O. Alatas, and M. Shah. KNIGHT^M: A real-time surveillance system for multiple overlapping and non-overlapping cameras. In *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME'03*, pages 649–652, Baltimore, Maryland, July 2003.

[7] O. Javed and M. Shah. Tracking and object classification for automated surveillance. In *Proceedings of the 7th European Conference on Computer Vision, ECCV'02*, pages 343–357, Copenhagen, Denmark, May 2002.

[8] V. Kettnaker and R. Zabih. Bayesian multi-camera surveillance. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR'99*, volume 2, pages 253–259, Fort Collins, Colorado, USA, June 1999.

[9] J. Kim, Y. Wang, and S. Chang. Content-adaptive utility-based video adaptation. In *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME'03*, volume 3, pages 281–284, Baltimore, Maryland, July 2003.

[10] M. Kim and Y. Altunbasak. Optimal dynamic rate shaping for compressed video streaming. In *Proceedings of the International Conference on Networking, ICN'01*, pages 786–794, Colmar, France, July 2001.

[11] S. McCanne and V. Jacobson. vic: A flexible framework for packet video. In *Proceedings of the Third ACM International Conference on Multimedia, ACMMM'95*, pages 511–522, San Francisco, CA, November 1995.

[12] V. Nair and J. J. Clark. Automated visual surveillance using hidden markov models. In *Proceedings of the 15th International Conference on Vision Interface, VI'02*, pages 88–92, Calgary, May 2002.

[13] W. Niu, J. Long, D. Han, and Y. Wang. Human activity detection and recognition for video surveillance. In *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME'04*, volume 1, pages 719–722, Taipei, Taiwan, June 2004.

[14] W. T. Ooi, P. Pletcher, and L. Rowe. Indiva: A middleware for managing distributed media environment. In *Proceedings of the SPIE Conference on Multimedia Computing and Networking, MMCN'04*, pages 211–224, Santa Clara, CA, January 2004.

[15] I. Pavlidis, V. Morellas, P. Tsiamyrtzis, and S. Harp. Urban surveillance systems: From the laboratory to the commercial world. *IEEE Proceedings*, 89(10):1478–1497, October 2001.

[16] R. Rangaswami, Z. Dimitrijevi, K. Kakligian, E. Chang, and Y. Wang. The SfinX video surveillance system. In *Proceedings of the IEEE International Conference on Multimedia and Expo, ICME'04*, Taipei, Taiwan, June 2004.

[17] V. Sanchez, A. Basu, and M. Mandal. Prioritized region of interest coding in JPEG2000. In *Proceedings of the 17th International Conference on Pattern Recognition, ICPR'04*, volume 2, pages 799–802, Melbourne, Australia, August 2004.

[18] R. Schumeyer, E. A. Heredia, and K. E. Barner. Region of interest priority coding for sign language videoconferencing. In *Proceedings of the First IEEE Workshop on Multimedia Signal Processing, MMSP'05*, pages 531–536, Princeton, NJ, June 1997.

[19] P. Viola and M. Jones. Robust real-time face detection. In *Proceedings of the ICCV 2001 Workshop on Statistical and Computation Theories of Vision, ICCV'01*, volume 2, page 747, Vancouver, Canada, July 2001.

[20] Y. Wu, L. Jiao, G. Wu, E. Chang, and Y. Wang. Invariant feature extraction and biased statistical inference for video surveillance. In *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS'03*, pages 284–289, Miami, FL, July 2003.

[21] X. Yuan, Z. Sun, Y. Varol, and G. Bebis. A distributed visual surveillance system. In *Proceedings of the IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS'03*, pages 199–204, Miami, FL, July 2003.