AN IP-BASED REAL-TIME VIDEO SURVEILLANCE SYSTEM

Michael Jiang, Weijian Liu, Bo Jing, and Xiaolin Kang
VTRON Technologies
Hi-Tech Industrial Development Zone, Science City, Guangzhou, China
{michael, liuweijian, jingbo, kangxiaolin}@vtron.com

ABSTRACT
Distributed multimedia-based surveillance has been commonly used for the monitoring and control aspects of many industry applications. These systems enable various multimedia sources to be transported, processed, and visualized with the requirement of real-time performance and synchronized visualization. In this paper, we presented an IP-based architecture for implementing large-scale and distributed video surveillance systems with real-time and synchronization requirements. The main contribution of our work lies in the distributed end-to-end flow control on top of an inherently unreliable and best-effort IP network to achieve real-time performance and the synchronization of media streams. Results from our studies showed that our distributed real-time systems were able to handle the processing of large number of multimedia sources and meet performance constraints.

KEY WORDS
Distributed systems, communication networks, real-time systems, multimedia, video surveillance.

1. Introduction

Video-based surveillance has been widely used for the control and monitoring of systems of various size and complexity, ranging from building security, product line monitoring, to large scale operation and control of large subway networks, airports, and other large systems. For a large control and monitoring system, it is often required to process large number of diverse multimedia inputs and display results with strict time constraints and performance requirements.

The rapid growth of distributed video-based surveillance applications calls for the adoption of readily available IP network. With IP technologies as the communication infrastructure, diverse products and technologies required by a surveillance system can be easily interconnected and the exchange of information among the related components and subsystems can be accomplished with Internet communication protocols.

The key requirements for our large-scale monitoring systems are the visualization of large amount of information with real-time requirements. Multiple video streams, for example, have to be processed, routed through the network processing elements, and displayed within required time frames. Among the video streams, some of them may need to be synchronized to ensure proper visualization of co-located or co-related information. Such requirements further complicate the design and implementation of the system.

Our work focuses on the design and implementation of real-time video-based surveillance systems on top of Internet architecture. While the Internet architecture provides universal connectivity and simplifies our design of the system, this infrastructure is inherently unreliable. The Internet protocol provides best-effort routing, with little or no deterministic behavior when it comes to packet delivery. Packets can also be lost along the way and it is up to higher level protocols or applications to implement their management and control mechanism. The key challenge in our system is how to provide real-time services with the best-effort communication infrastructure.

In this paper, we proposed a distributed end-to-end flow control mechanism to design and implement real-time video surveillance services on top of the IP technologies. Status information about dynamic end-to-end communication of the distributed system is collected and analyzed for system-wide optimization. Our design strategy is to minimize the occurrences of real-time violations. We combine the dynamic control of end-to-end flow of media streams and buffering techniques to balance the delays of media streams within the distributed system. In addition to optimizing the system for real-time performance, our mechanism also provides the capabilities to synchronize the visualization of media streams. To validate the feasibility of our approach, we conducted several experiments and pilot projects.

The rest of the paper is organized as follows. Section 2 describes the IP-based architecture of the distributed multimedia monitoring system. Section 3 describes the distributed flow control mechanism for real-time performance and synchronized media visualization. Section 4 describes our experiments and their results. Related work is presented in section 5 and section 6 concludes the paper.

2. Architecture for Distributed Video Surveillance

The video-based surveillance system follows a distributed architecture in which key subsystems and components
collaborate in a network environment. We built the signaling and data communication mechanism of the distributed video surveillance system based on the common Internet Protocols (IP). This section gives an overview of the system requirements and architecture supporting distributed video surveillance applications.

### 2.1 System Overview

The high-level architecture view of the real-time distributed surveillance system is shown in Figure 1. The system is built on top of the general Internet infrastructure. The video surveillance system consists of a number of video recording devices capturing video streams. The media aggregators route the media streams to the media distribution subsystem through the IP-based network. Video streams received by the distribution subsystem are forwarded to the display subsystem consisting of a group of display devices for visualization. In this distributed system architecture, any video streams can be configured to be routed to any display devices for visualization.

The operation, coordination, and control of the media aggregators, the media distribution subsystem, and display subsystem are under the management and control of the system controller. A media aggregator is configured to forward video streams to any display devices of the display subsystem. Statistical information with respect to delays, packet loss, and other related information is collected and analyzed by the system controller for system-wide optimization. Based on the system analysis, the system controller instructs media aggregators, the distribution subsystem, and other components to make necessary adjustments to ensure system-wide synchronization and real-time requirements are met or violations reported.

### 2.2 Real-Time Constraints for Video Surveillance System

The inherent best-effort packet routing services provided by the Internet architecture make it challenging to achieve real-time performance for the video-based surveillance systems [1] [2]. To implement real-time services, we developed the distributed system to support the end-to-end monitoring, control, and adjustment schemes for all media sources. Dynamic information is collected by the system controller for overall system analysis and optimization, with the goal to maintain the system within the bounds of real-time requirements.

Another key system requirement is the synchronized visualization of multiple video sources. This is required when a single video stream is routed to multiple display units for visualization. Since the involved display units may receive the media stream with different delays, these display units need to coordinate so that the same media frame can be visualized at the same time across the related display units. Another situation is the synchronization of multiple video streams in one or more

![Figure 1: System architecture](image-url)
display units. This may be necessary when several different video streams capture related information and video frames captured at the same time need to be visualized in synchronization. In a traffic intersection, for example, multiple cameras may be positioned from different angles to provide 3D viewing of the traffic from all directions in real-time.

To minimize network traffic and optimize real-time performance, multicasting is enabled in our network system to support the forwarding of a video stream to multiple recipients efficiently. It should be noted that we tackle the real-time constraints problem through system-level optimization and with the assumption that the network infrastructure provides reasonably sufficient resources. Through dynamical and optimal adjustment of frame rates and buffering schemes discussed later, our video-based surveillance system attempts to satisfy the real-time constraints of the overall system.

3. Distributed Flow Control and Synchronization

We designed a distributed end-to-end flow control mechanism to satisfy real-time constraints and achieve the synchronized visualization of multiple media streams. Status information about end-to-end transportation of media streams is collected and analyzed for overall system optimization, with the goal of meeting real-time constraints and minimizing the occurrences of real-time violations. Based on the congestion situation of the network, the rates of media streams can be regulated up to limits set by their corresponding real-time constraints. Our design also adopts buffering techniques to assist the implementation of both flow control and synchronized visualization of multiple media streams.

3.1 Distributed Flow Control

The lack of real-time capabilities in the IP-based distributed infrastructure calls for external control mechanism to implement real-time performance. This external control mechanism coordinates with all related components and subsystems in the distributed environment to share common network resources. As discussed earlier, the System Controller is designed to perform system-wide traffic flow analysis and instructs all other components and subsystems to regulate the use of system resources.

Our design to achieve real-time performance is to keep the delays of end-to-end flows within the bounds of time constraints specified at the system level. Since the network layer provides only best-effort packet routing services, the flow of media traffic from a media source to the display subsystem will incur unpredictable delays throughout the network, including processing time in a media aggregator, packet routing time through the network, and processing time in the media distribution subsystem. Media buffers for video streams are added in the media distribution subsystem to balance unpredictable delays and support both real-time processing and synchronized visualization of media streams.

Real-time performance is met by ensuring the accumulated delays throughout the network fall within the bounds of system constraints as shown in Figure 2. As shown in the diagram, latencies from media source to display subsystem includes the processing time in the media aggregators, packet routing delays in the IP

![Figure 2: Flow control for real-time performance](image)
network, and the processing time in the media distribution subsystem. If the total delay from a media server to the display subsystem for a media source is above the specified bounds of the system, it is viewed as real-time violation of the system and will be reported accordingly. To keep media source 1 within the time constraints, for instance, the sum of latencies \( T_{11}, T_{12}, \) and \( T_{13} \) should be kept under the specified bound for media source 1.

The time stamp in the media frames set it by the media source is used to calculate delays throughout the network. Latency is computed by using the time difference between a packet received by the media distribution subsystem and the packet sent by the input source. The latency between the distribution subsystem and the display units is minimal in our chosen architecture and is not considered. Another aspect is that latency information of intermediate packet hops through the network can provide finer grain of details on the congestion status of the network. To simply the design and implementation of our system, the current work doesn’t take this into consideration.

The media distribution subsystem provides a frame buffer for every input source to help maintain frame rates and the synchronization of multiple sources. Within the allowable maximal delays, the use of frame buffering helps minimize jitters since the arrival of packets can have different latencies and can be out of order. A buffer is maintained for each receiver of an input stream in the system. If an input stream has two receivers, each receiver maintains its own buffer. The size of a frame buffer for an input source is determined based on the frame rates and maximal delays allowed of the input source. If an input source allows longer delays, a larger buffer will be created for this input source.

3.2 Synchronized Visualization

As discussed above, synchronization is needed when a single media stream is routed to two or more display units for visualization and the media frames with identical time stamps are required to visualize at the same time. Synchronized visualization is also required when multiple media streams capture co-related contents. In certain applications, visualization of multiple media streams without proper synchronization would lead to undesirable results or incorrect system behaviors.

Synchronization of media streams is performed by using media buffers in the media distribution subsystem as shown in Figure 3. Every media frame carries its own time stamp recorded at its source. The buffers in media distribution subsystem enable that media frames with identical time stamps are visualized at the same time. If media streams 1, 2, and 3 are required for synchronized visualization, for instance, media frames from corresponding receivers 1, 2, and 3 with time stamp \( TS_1 \) can be taken out of their queues and sent to the corresponding display units for visualization. Display units that involve synchronization are under the overall coordination of the system controller.

Frames in a stream buffer may be skipped to ensure proper synchronization of two or more media streams. Our approach to synchronization of multiple media streams is based on the slowest one. The media distribution subsystem sequentially processes all the media buffers based on time stamps. For any time slot, if a frame exists in one buffer and no frames with identical time stamp are found in other buffers, this particular time slot is skipped and the related frame is discarded. This approach is to ensure the visualization of all the related media streams is achieved, although at the expenses of frame loss and viewing quality.

The adoption of buffers will inevitably introduce latencies. This happens due to the overhead of buffer processing and the abandonment of media frames. Since the overall system goal is to ensure real-time performance,
the system constraints are still met if the introduced latencies do not lead to real-time violation. Thus, the tolerance of latency is determined by the specification of time constraints at the system level.

3.3 Integrated Real-Time Handling and Media Synchronization

Our system design takes into consideration of both real-time media processing and synchronized visualization of one or more media streams. The algorithm ensures real-time requirements are met after attempting to synchronize the visualization of media streams if specified. If real-time requirements cannot be satisfied, violations will be reported.

The integrated real-time handling and media synchronization process is illustrated in Figure 4. First, time related information for all the media sources is collected from the media aggregators representing the media sources and from media distribution subsystem representing the media receivers. The collected information is used for computing latencies incurred by the related components, subsystems, and the communication network. The next step is to perform synchronized visualization of media streams as discussed in the previous subsection.

To properly compute real-time performance, latencies caused by synchronization processing are accrued, in addition to those collected from related network components and subsystems. For normal situation, the system will continue to loop forever with LP1, with the goal of maintaining synchronized visualization of media streams. After the completion of synchronization processing, the system proceeds to verify real-time performance by checking whether the latencies of any media sources have reached the constraints specified by the application. If the specified constraints for a particular media source cannot be satisfied by the system, latencies in other media sources are checked for the possibility of rate reduction without affecting their real-time constraints. Media sources with the biggest room for rate adjustment are chosen and are adjusted accordingly. The processing algorithm continues the above “checking-adjusting” cycle as indicated with loop LP2 to ensure that
the system adapts to the changing network environment and the availability of network resources. If there is no room to be found for adjustment for any media sources, the process will resort to the reporting of constraint violations and continue to look for system optimization as indicated in loop LP3. User intervention may also be initiated as part of the reporting procedures. It should be noted that the network infrastructure needs to provide reasonably adequate network resources such that the system can balance across multiple media streams and minimize the occurrence of real-time violations.

4. Experiments

Our system was piloted for multiple digital security and surveillance applications, with several deployments in city traffic and airport security surveillance. In our experiments, we specify real-time requirements as time constraints of maximal delays allowable on different media streams. Users can also specify synchronization requirements on single or multiple media streams. The applications are validated against synchronized visualization of media streams and real-time performance specified.

Figure 5 shows the system configuration of our lab experiments. In this network setup with three LAN network segments with two multi-segmented network emulators to simulate Internet infrastructure with dynamic behaviors, we drove the system testing with up to ten recorded media streams of various stream rates played by the PCs. The purposes of the network emulators are to mimic network behaviors by introducing latencies, congestions, packet loss, etc. With the above setup, we adjust the network emulators to the point that time constraints cannot be satisfied. We then turned off the flow control and optimization and observed the behaviors of the system. The above testing process was repeated with different bandwidth restrictions until violations were reported for all of the input sources.

<table>
<thead>
<tr>
<th>--Congestion level--</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization enabled</td>
<td>0</td>
<td>1,2</td>
</tr>
<tr>
<td>Optimization disabled</td>
<td>3</td>
<td>7,8,9,10</td>
</tr>
</tbody>
</table>

Table 1: Violations of time constraints with different congestions

Table 1 showed the number of media streams experiencing violations under different network congestions with ten streams in the above experiments. The system under test was able to perform consistently without leading to real-time violations when there was sufficient network bandwidth. The system would gradually lead to real-time violations after further reduction of network bandwidth governed by the network emulators. When there was little bandwidth to meet the minimal requirements, the performance under both situations was equally bad. Our experiments showed that real-time violations occurred more frequently than when the flow control and optimization was turned off. Up to 20% reduction in violations could be found with end-to-end flow control from the network configurations in our experiments. For those systems deployed for city traffic control, similar behaviors of the system were also observed.

Real-time violation should be better handled or its severity can be reduced if priority of traffic is taken into consideration in the processing algorithm. Traffic priority is not implemented in the current system. Although the experiments showed improvement with end-to-end flow control, our deployments are still limited and more validation needs to be done to further optimize the control algorithm.

Results from our experiments and deployments showed that IP infrastructure also needs to be carefully engineered to support real-time applications. With proper configuration of network routing and sufficient provisioning of network bandwidth, real-time performance can be maintained and synchronized visualization of media streams achieved.

5. Related Work

Much work has been done on real-time multimedia processing and applications [3] [4]. Video conferencing systems built on telecom infrastructures, for instance, provide real-time video transport on private and dedicated network with QoS (Quality of Service) guarantee on network bandwidth and real-time performance.
For Internet based distributed systems, additional challenges exist when real-time requirements are imposed onto such systems due to the best-effort delivery mechanism of the Internet routing protocols. Methods were proposed and models developed in [5] to monitor the quality of compressed video transmitted over a packet network. Authors in [6] investigated the problem of assessing the quality of video transmitted over IP networks. A model was developed to assess the impact of various network-dependent and application-specific factors on the quality of the decoded video. Authors in [7] described a middleware for video-based surveillance networks. The middleware provides support for both computational and communication aspects of automated video surveillance networks. Communication on the surveillance network is supported through the instantiation of the service oriented architecture, with publish/subscribe messaging.

An architecture based on wireless mesh network was described in [8] that can efficiently transport real-time video streams from multiple sources to a central monitoring station. Specialized wireless routers were designed to optimize the transport of real-time traffic and algorithms were developed to reduce end-to-end delays for multimedia streams. Authors in [9] presented a distributed real-time application with the design and implementation of surveillance, detection, and tracking of time-critical targets. Authors in [10] described a time-synchronized application-level MAC protocol that operates over existing 802.11 MAC protocol to provide a real-time and scalable communication infrastructure for video streaming.

The contribution of our work lies in our approach of dynamic and distributed end-to-end flow control on top of an inherently unreliable IP network to improve real-time performance of the overall system and achieve the synchronized visualization of related media streams of the system. Dynamic latencies incurred in the components and subsystems are collected and system-wide optimization is performed to minimize the occurrences of real-time violations.

6. Conclusion and Future Work

This paper presents a distributed end-to-end flow control mechanism for the design and implementation of large-scale real-time video surveillance services with the capabilities of synchronized visualization on top of the common IP network infrastructure. Latencies of dynamic communications and media processing in components and subsystems of the distributed system are collected and analyzed for system-wide adjustment and optimization. The system allows the specification of synchronized visualization of related video streams. The real-time performance of the system is achieved through a continuous adaption process taking into account of the changing network environment.

To validate the feasibility of our approach, we conducted experiments internally and deployed systems for city traffic control and monitoring applications. Results from both internal experiments and external applications indicated that real-time performance can be improved and synchronization achieved as opposed the same operations under the same network infrastructures when the the processing and optimization algorithm is disabled. Our processing algorithm stills has a lot of room for improvement and more experiments are needed for further optimization.

References