A Modular Distributed Video Surveillance System Over IP

Drew Ostheimer Sébastien Lemay Dennis Mayisela Pierre F. Dagba
Mohammed Ghazal Aishy Amer

Electrical and Computer Engineering, Concordia University
Montréal, Québec, Canada
Correspondence: amer@ece.concordia.ca

Abstract

We present an automated and distributed real-time video surveillance system which can be used for the detection of objects and events in a wide range of applications. Video feeds are captured from multiple sources, processed and streamed over the Internet for viewing and analysis. Components of the system can be interconnected in several manners, thus forming flexible systems. The experimental results show a system that handles multiple video feeds, running on standard computers and yielding fluid video. Several interconnected clients can view multiple feeds simultaneously, as well as the event listing.

Keywords *— Video, Surveillance, Internet Protocol, Multicast, RTSP.*

1 Introduction

In response to the increase in security concerns, automated real-time surveillance applications have received significant attention from the research and industrial communities. The deployment of such applications has become a necessity in airports, subways, offices and even homes. Currently deployed automated surveillance systems suffer from non-scalability or low frame rates due to computationally expensive algorithms. In general, a video surveillance system should have 1) affordable hardware requirements, 2) real-time environment adaptation, 3) low bandwidth consumption, 4) access control procedures and 5) efficient alerting mechanisms.

Recent surveillance systems can be categorized into specialized and generalized systems. The systems in [3–5] are specialized in train or railway surveillance [4], traffic or highway surveillance [3], and elevator surveillance [5]. These systems are tuned for specific application, therefore can not be deployed in different situations.

Generalized systems can be further categorized into distributed and non-distributed systems. Non-distributed or centralized systems deliver less performance due to the increased hardware requirements. On the contrary, distributed systems [2, 6] utilize communication protocols to divide the work amongst a network of less powerful computers, thus making automated surveillance systems available for residential or commercial use.

To address the requirements of modern surveillance, this paper proposes a system with: 1) modular and extensible design, 2) distributed computing allowing the use of standard computers, 3) MPEG-4 compression for reduced disk and network consumption, 4) real-time delivery of video over multicast, and 5) real-time event notification.

The remainder of the paper is organized as follows: Section 2 gives an overview to the proposed system; Section 3 presents the proposed system in detail. Section 4 discusses experimental results and Section 5 concludes the paper.

2 System Overview

The proposed system consists of three modules: Video Workstations, Control Workstations and a Server. Video Workstations are responsible for capturing the raw video from sources such as cameras, network streams or disks. Video Workstations are then responsible for processing the captured video to extract events from the scene. The proposed system uses the methods in [1] to extract surveillance events. The video signal is then compressed and sent using multicast to Control Workstations via the Real Time Streaming Protocol (RTSP).

A Video Workstation registers with the Server when it goes online. All communication, excluding surveillance video, between Video and Control Workstations is logged. The Server maintains security information about the access privileges of all connected Control Workstations and distinguishes between different levels of access. The Server also transfers the recorded videos from the Video Workstations at a scheduled time. It also archives and stores the received video and provides the means to access them.

Control Workstations, in a similar manner to the Video Workstations, register with the Server. A Control Workstation can view the feed from or give command to one or more Video Workstations. The proposed system utilizes embedded media players to view the surveillance video streams. Commands are created from the graphical user interface (GUI) events on Control Workstations and are sent to Video Workstations, through the Server. Control Workstations are programmed to respond to events using alert mechanisms. In the proposed system, visual and text messaging alerts are used as an example.

3 Proposed System

The proposed system consists of a set of modules (see Fig. 1): Video Workstations $VW = \{v_i\},\ i \in \{1, 2, ..., N_{VW}\}$, where N_{VW} is the number of Video Workstations, Control Workstations $CW = \{c_j\},\ j \in \{1, 2, ..., N_{CW}\}$, where N_{CW} is the number of Control Workstations and a centralized Server S. Each module is built from a set of components, which fall under two main categories. The first, generic components, denoted by $\hat{C}$ are reusable compo-

1-4244-0038-4 2006 IEEE
CCECE/CCGEI, Ottawa, May 2006

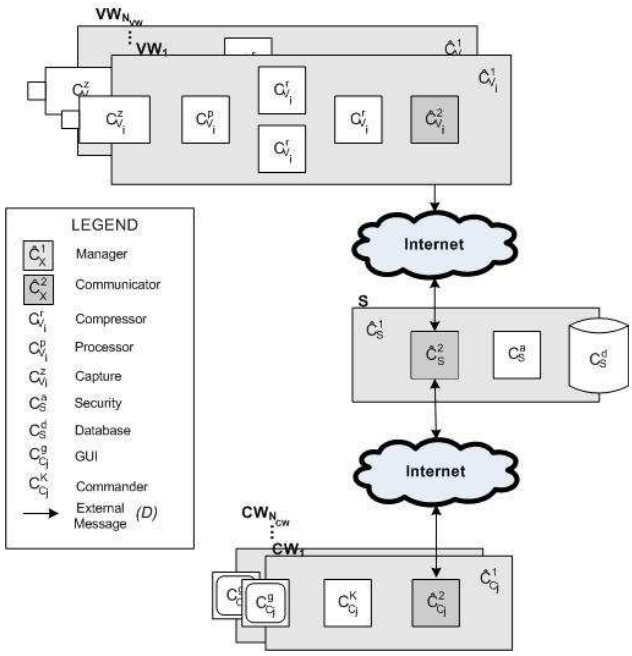

Figure 1: Proposed system modules and components.

nents, found in the CW, VW and S. The second, module specific components, denoted by C are deployed within only one module. A module v_1 consists of N_{v_1} components, labelled $v_1 = \{\hat{C}^1_{v_1}, \hat{C}^2_{v_1}, ..., C^{N_{v_1}}_{v_1}\}$. In the proposed system, $\hat{C}^1_X$ denotes the manager component, which is responsible for starting and maintaining all other components of the module. The communicator component, denoted by $\hat{C}^2_X$ is responsible for processing and delivering protocol messages across modules. Module specific components will have the indices: capturing components z, processing components p, compression components r, streaming components s, authentication components a, logging components l, and GUI components g.

3.1 System Reliability

The proposed system is designed to accomodate failure of one or more modules or components. For examples, should the communication channel between the Control Workstation and the Video Workstation fail, local archives of surveillance feeds are maintained.

Components have generic interfaces which allows for real-time component switching in the case of component failure or in response to operator requests. For example, if the MPEG-4 compression component fails, the manager component will halt the system until a replacement MPEG-4 or MPEG-2 compression component is integrated. The same example holds for the streaming component. If the Control Workstation is not able to decode RTSP streams, the manager will integrate an HTTP streaming component. Surveillance feeds and events are buffered during the halt time and will be resent upon system recovery.

3.2 Surveillance Protocol

A surveillance protocol message K is designed to carry a command D, or a surveillance event E from a Control Workstation to a Video Workstation. The communication takes place between communication components $\hat{C}^2_{v_i}$, $\hat{C}^2_{c_j}$ and $\hat{C}^2_S$.

The surveillance protocol message K consists of four sections. The first section contains the source identifier $K.source$, which will be used to acknowledge the reception of the message. The second section holds the next destination $K.next$, which is updated at each hop the message takes. The third section holds the final destination of the message $K.dest$. The last section contains the message $K.message$. For example, should c_1 wish to send the $D = requestFeed$ command, used to request the video feed, to v_1 through S, K changes as shown in Table I.

Hop	K.source	K.next	K.dest	K.message
Command Passes Authentication				
1	c_1	S	v_1	requestFeed
2	c_1	v_1	$\hat{C}^2_{v1}$	requestFeed
Command Fails Authentication				
1	c_1	S	v_1	requestFeed
2	S	c_1	$\hat{C}^2_{c1}$	authorizationFailed

TABLE I
SAMPLE SURVEILLANCE PROTOCOL PACKET.

Surveillance protocol messages are relayed through the Server using the authentication tables T_{comm} and T_{view} (see Eqn. 1). The message originating from $\hat{C}^2_X$ received by S, is forwarded as shown in Eq. 2. If the surveillance protocol message fails authentication, an authorization failed event $E = authorizationFailed$ is triggered to alert the user, and replaces $K.message$.

$$\begin{aligned} T_{comm} &= \{(v_i, c_j) \mid c_j \ commands \ v_i\} \\ T_{view} &= \{(v_i, c_j) \mid c_j \ views \ v_i\}. \end{aligned} \tag{1}$$

$$K.next = \begin{cases} K.dest & : ((K.message == view) \wedge \\ & ((K.source, K.dest) \in T_{view})) \vee \\ & (K.source, K.dest) \in T_{comm} \\ K.source & : otherwise \end{cases} \tag{2}$$

The protocol design allows for expandability, as new events and controls, require no modification to the message structure. The message also allows for authentication at a centralized server and later delivery to the source.

3.3 Video Workstation

Video Workstation components register with the manager, $\hat{C}^1_{v_i}$. This internal registry facilitates component

maintainace. Acquisition of the video signal from capture component $C_{v_i}^{z_j}$ and is delivered to consecutive components. The video signal is then processed by $C_{v_i}^{p}$ using the algorithm in [1] to detect surveillance events such as object occluding or approaching restricted sites.

The proposed system is independent of the video processing algorithm in [1], however these methods are used for their real-time performance with [1]. The video signal is transformed to label video objects and assign different overlaid colours to make them easy for viewing. The events are annotated on the video, (see Fig. 2), and are sent to Control Workstations through S using $\hat{C}_{v_i}^{2}$.

To improve the system frame rate, the proposed system utilizes a frame control algorithm. A threshold TH_{fra} is set by $\hat{C}_{v_i}^{1}$ in regards to the current system load, shown in Eq. 3, where $FPS.cur$ is the current frame rate, and $FPS.desired$ is the desired frame rate. The act of dropping a frame is controled by

$$TH_{fra} = \begin{cases} TH_{fra} - 1 : FPS.cur > FPS.desired \\ TH_{fra} + 1 : FPS.cur < FPS.desired \end{cases} \quad (3)$$

$$f[n] = f[n \times TH_{fra}]. \quad (4)$$

Compressed and non compressed surveillance video can be recorded to disk. The selection criteria is based on a threshold variable TH_{rec}, which is the maximum duration of time between two successive events to continue recording. Let $t_1 = time(E_1)$ and $t_2 = time(E_2)$ be the creation times of two events, E_1 and E_2, surveillance video is recorded using

$$record(t_1, t_2) = \begin{cases} 1 & : |t_2 - t_1| < TH_{rec} \\ 0 & : otherwise \end{cases}. \quad (5)$$

After compression, the video can be streamed out over the network by $C_{v_i}^{s}$ to the different Control Workstations. It is streamed using multicast addressing over RTSP to reduce network usage. Multicast networks allow for data messages to be sent once from the source, in this case $\hat{C}_{v_i}^{s}$, and be delivered to Control Workstations. Packet duplication is done at the router level, therefore average network usage is increased, despite lower load on the originating machine. This allows theoretically an unlimited number of clients to view the processed streams. The only limitation on the use of multicast is that imposed on the router to support the redirection of data on multicast addresses.

3.4 Server

The server is responsible for maintaining two tables of available c_i and v_i, which are denoted by T_{online}^{vw} and T_{online}^{cw} (see Eqn. 6). The system distinguishes between two types of access to v_i controlled by T_{comm} and T_{view} (see Eqn. 1) in C_S^a. The first type of access allows $C_{c_j}^{g}$ to view the video feed from $C_{v_i}^{s}$. The second type allows $C_{c_j}^{g}$ to issue commands to any $C_{v_i}^{X}$. Based on these two access levels, the proposed system can grant personalized functionality to end-users.

$$\begin{aligned} T_{online}^{vw} &= \{v_i \mid isOnline(v_i)\} \\ T_{online}^{cw} &= \{c_j \mid isOnline(c_j)\} \end{aligned} \quad (6)$$

The server is also responsible for polling Video Workstation at a specific interval and transfer the archived data which was previously recorded. The data will remain on the server, being overwritten as space requirements demand.

3.5 Control Workstation

The Control Workstation provides the main form of user interaction with the proposed system. The three main components of the Control Workstation are: the manager $\hat{C}_{c_j}^{1}$, the communicator $\hat{C}_{c_j}^{2}$, and the GUI $C_{c_j}^{g}$.

The command generation process starts with $C_{c_j}^{g}$ creating an internal protocol message in response to a GUI event that is captured by the manager $\hat{C}_{c_j}^{1}$ in the next processing cycle. D_{int} has information about which v_i the message should be delivered to. The manager delivers the message to the the communicator component $\hat{C}_{c_j}^{2}$ queue for processing and delivery. $\hat{C}_{c_j}^{2}$ delivers the message to $\hat{C}_S^2$.

The response to the command generation process, received from $\hat{C}_S^2$ is displayed in the $C_{c_j}^{g}$ component. Using a generic embedded MPEG-4, RTSP compliant player, the video stream is rendered to the end-user, with annotated events. The $C_{c_j}^{g}$ component also stores $T_{event} = \{E_i | E_i \in E\}$, a table of events.

The $C_{c_j}^{g}$ component is scalable such that rich clients, for example, personal computers, can display both video feeds and T_{event}. Portable mobile devices, for instance cell phones, will only display T_{event}, as video feed rendering is not supported by these devices.

4 Results

The proposed system has been tested on a standard Intel Pentium 4 computer with two concurrent surveillance feeds. With the addition of another machine the system can achieve four simultaneous processed video feeds. Each feed is processed and delivered to the Control Workstations on average at 21 frames per second. Using a SCSI hard disk in the Video Workstations it is possible to record both feeds in a raw format.

The Control Workstation, as shown in Fig. 2, holds a list of currently available Video Workstations. This list allows the Control Workstation to see or operate multiple video feeds. Should the operator select more than four feeds, the view will expand downwards creating room for another two feeds. Commands can be sent to the Video Workstation by means of the command panel, either above or below the video feed, and asynchronous javascript and XML. Incoming events from the Server are displayed in a scrollable table which allows the operator to see an event history.

Figure 2: Web-based Surveillance System.

As the system uses multicast RTSP to send data the network overhead has been noted to be no more than 100KB/s per stream, independent of the number of Control Workstations.

The proposed system has also been used without streaming to record a live video stream to disk.This configuration demonstrates the component interchangeability and expandability of the system as a whole.

5 Conclusions

We have proposed a system which is created by the interconnection of modules and components, allowing the system to be deployed on standard readily available computers. Using frame thresholds, the system can scale to multiple sources, depending on the CPU, producing an encoded stream in real-time. Overcoming the network constraints, in relation to multiple destinations, multicast real-time packets are delivered using the real-time streaming protocol. Experiments under normal and heavy loads have yielded visually acceptable frame rates on multiple destinations. Several capture devices, video processing algorithms, and notification methods have demonstrated that the proposed system can be used in different environments such as, commercial and residential.

References

[1] A. Amer, E. Dubois, and A. Mitiche, *"Rule-based real-time detection of context-independent events in video shots"*, *Elsevier Journal for Real-Time Imaging*, vol. 11, no. 3, pp. 244-256, 2005.

[2] Christopher Jaynes, Stephen Webb, R. Matt Steele and Quanren Xiong, *"An open development environment for evaluation of video surveillance systems," IEEE workshop on performance analysis of video surveillance and tracking*, June 2002.

[3] Shunsuke Kamijo, Howard Koo, Xiaolu Liu, Kenji Fujihira and Masao Sakauchi, *"Development and evaluation of real-time video surveillance system on highway based on semantic hierarchy and decision surface," IEEE conference on systems, man and cybernetics*, vol. 1, pp. 840-846, October 2005.

[4] Claudio Sacchi and Carlo S. Regazzoni, *"A distributed surveillance system for detection of abandoned objects in unmanned railway environments," IEEE transactions on vehicular technology*, vol. 49, no. 5, pp. 2013-2026, September 2000.

[5] Hui Shao, Liyuan Li, Ping Xiao and Maylor K.H. Leung, *"ELEVIEW: An active elevator video surveillance system," IEEE workshop on human motion proceedings*, pp. 67-72, December 2000.

[6] Xiaojing Yuan, Zehang Sun, Yaakov Varol and George Bebis, *"A distributed visual surveillance system," IEEE international conference on advanced video and signal based surveillance proceedings*, pp. 199-204, July 2003.