DIRECTSHOW APPROACH TO LOW-COST MULTIMEDIA SECURITY SURVEILLANCE SYSTEM

Wu Xiao, B.S.

Thesis Prepared for the Degree of

MASTER OF SCIENCE

UNIVERSITY OF NORTH TEXAS

December 2002

APPROVED:

Paul Fisher, Major Professor
Robert Renka, Committee Member
Armin Mikler, Committee Member
Krishna Kavi, Chair of the Department of Computer Science
C. Neal Tate, Dean of the Robert B. Toulouse School of Graduate Studies

Xiao, Wu, DirectShow Approach to Low-Cost Multimedia Security Surveillance System. Master of Science (Computer Science), December 2002, 54 pp., 1 table, 28 illustrations, references, 35 titles.

In response to the recent intensive needs for civilian security surveillance, both full and compact versions of a Multimedia Security Surveillance (MSS) system have been built up. The new Microsoft DirectShow technology was applied in implementing the multimedia stream-processing module. Through Microsoft Windows Driver Model interface, the chosen IEEE1394 enabled Fire-i cameras as external sensors are integrated with PC based continuous storage unit. The MSS application also allows multimedia broadcasting and remote controls. Cost analysis is included.

ACKNOWLEDGMENTS

I thank MultiMedia Information Technology Group (MMiTG) for their technical support and Development environment. I thank University of North Texas for financial support.

TABLE OF CONTENTS

Page

LIST OF TABLES AND FIGURES

Page

Table

Figures

CHAPTER 1

INTRODUCTION OF MULTIMEDIA SURVEILLANCE SYSTEM

Security surveillance, which appeared at the same time when the human society infrastructures formed 3000 years ago, is one of the oldest professions in history. Before the boost of modern industry, security surveillance was usually performed by human forces named sentries or security personnel. They protected sensitive areas against any intruders with their eyes and ears for hundreds of years.

Since the 1950's, following the advances of video/audio, network and computer technologies, a revolution on security surveillance has started with a goal to step by step replace human involvement in this process. Especially after the Terrorists' Attacks on September 11, 2001, the increasing requirements of modern society in the direction of public or private safety and security make the MSS application an intensive focus for both research and industry.

Multimedia Surveillance System (MSS) is an interdisciplinary application field emerging research and the industrial areas of multimedia capturing, signal processing, networking and computer vision. Figure 1 shows the overall architecture of a generic MSS. Different from that of human sentries, MSS applications capture the multimedia source through external sensors, and then the transmission layer carries the media stream to the control center in which raw data is processed and responses to certain events are made. Resulting Data is then saved onto storage media.

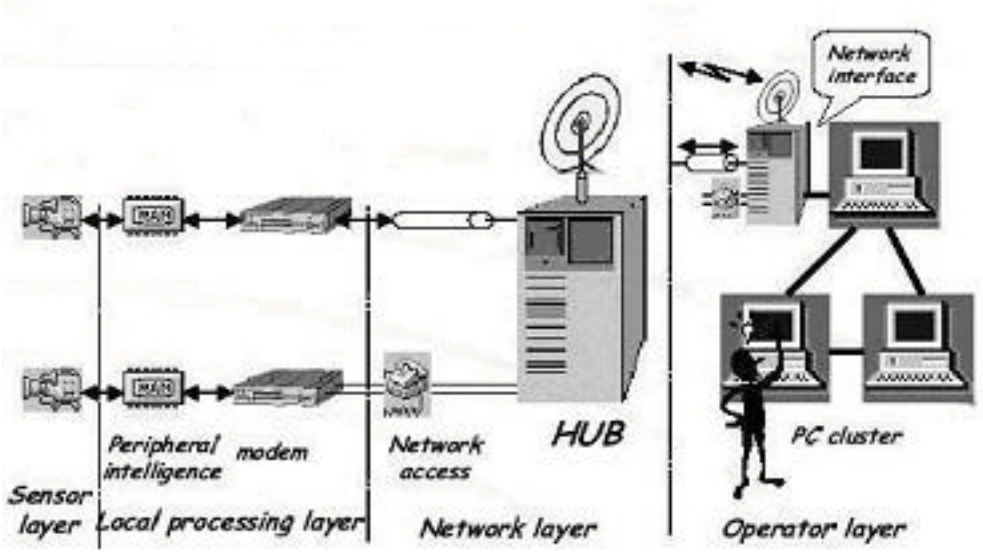

Figure 1: Architecture of Generic MSS

Since the first commercial MSS application was introduced in the 1960's [1], the potential benefits of the substitution of manpower in security surveillance with MSS strongly stimulated the quick growth of MSS applications in not only governmental but also civilian environments.

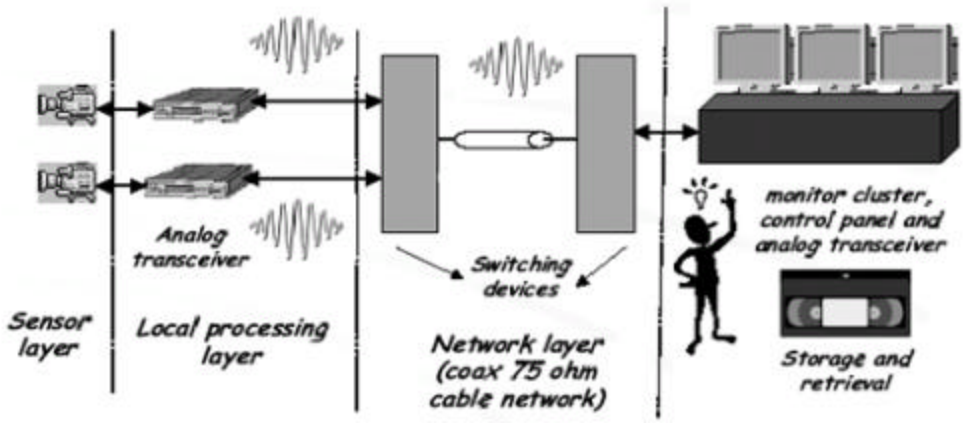

Figure 2: First Generation of MSS

The first generation MSS (Figure 2), also called Guide to Closed Circuit Television (CCTV), was built up on analogue sensors, and the sensors simply transfer signals to the storage without any processing on the signals. The system was wired into a cyclic multiplexing complex so that multiple sensors could bring what they sensed into a central storage that was an array of tape recorders. In this scenario, though the signal capture and storage processes were automatic, the surveillance operation had to be done by human operators in the control center. Also maintenance such as switching tapes and special operations such as backup, scan and search still required significant human work. But it was more efficient and cheaper than a collection of outside sentries.

The second generation (1980-2000) came with an advanced centralized monitoring system and digital sensors [2]. The most important change was the computer application in the system that possessed strong computation power and image processing abilities. This

change made the MSS "smart" enough to identify specific objects, detect motion and broadcast alerts. Wide-brand networks with either coax or wireless connections carried multimedia signals in data transmissions. But some constant human assistance including handling an alert was still necessary.

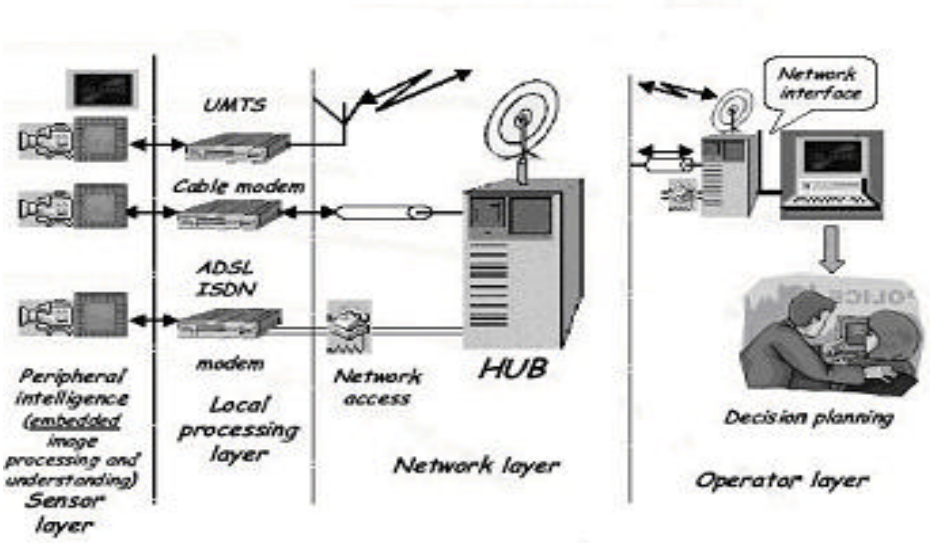

Figure 3: MSS Since 2000

The fully digitized MSS appeared in 2000 (Figure 3) [3]. Based on the most recent techniques and engineering, this new generation will be equipped with intelligent sensors. An artificial intelligent processing center will collect, selectively store and analyze the multimedia information delivered from the sensor layer automatically and in critical situations, it not only will be able to send alerts in different ways, but also would be equipped to affect defending operations. This complete automation will need minimum human supervision.

My research focused on the approach to low-cost MSS applications with the latest computer technologies.

CHAPTER 2

REQUIREMENTS OF A MULTIMEDIA SURVEILLANCE SYSTEM

Though implementation of a Multimedia Security Surveillance (MSS) that is adaptive to all kinds of environments is not absolute impossible, usual MSS applications are designed to function in certain environments. Built upon a complex of hardware and software components and their integration, MSS applications provides mission-critical services to customers for monitoring and guarding secured areas, which should never stop, crash or fail in any circumstances. So besides the ability to work in the different environments where they are installed, MSS applications must share some of the following characteristics, which have highest priorities in their design, implementation and engineering.

Robustness

Commercial MSS application must be engineered to be robust in relation to different facets and completely adaptive to the environment where installed. The designer has to consider how to guarantee the survival of the application in handling different environmental conditions and changes. For example, MSS applications on automobiles must be able to adapt to different path altitude changes (flat or hilly paths), road changes (highways or urban ones), illumination changes (day, night, sunset, or sunrise), and weather changes (sun, fog, rain or even snow). Also, unexpected or sudden changes of these conditions must be expected and handled [4].

Since the end-user expects no breakdown for an MSS application as its mission-critical property, the possibility of any faults must be decreased to zero to ensure the trust of

customers. This robustness requirement is for not only the software algorithm but also the hardware system (sensors, connections, computers and networking). The whole system must be able to withstand mechanical, physical and even chemical stresses, such as vibration or high temperature as well as software breakdown.

Reliability

Since MSS applications must be required to be a safety-critical complex of different devices from different vendors, a strong degree of reliability is mandatory. This means in any development or implementation phase, extensive analysis, testing and validation have to be done to ensure the correctness of the system. Unsurprisingly, the careful selection of needed hardware and software components is indispensable to ensure the accomplishment of an MSS application with high quality.

Costs

This issue presents no problems for high-cost facilities for which an expensive safety system is considered as an investment for protecting the facilities and the people involved. On the contrary, most of the security market needs are very much cost sensitive. This issue has been considered to be the key parameter to the commercial success of an MSS application development. A cost analysis on each design or engineering phase is required to aim at the reduction of the market price through a low-cost implementation and installation of the complete platform. Not only for competition reasons, it also has been estimated that such an MSS should cost no more than ordinary facilities except for some extreme cases where ultra-secure facilities are required.

In an administration point of view, the assessment of the total costs of an MSS system comes from the combination of both installation costs and the operative costs. The former ones are usually the payments to operative personnel, extensive storage and maintenance. The costs to integrate the MSS to the other services of the expected facility, for example connecting the MSS to the intranet or electricity wires, is also on the agenda of developers.

Performance

Although overall cost of the system is one of my major concerns, it does not mean to ignore the performance of the resulting system. Because the multimedia processing demands strong computation power, a point of balance with the price and performance must be located so that satisfactory multimedia capture and processing quality would be achieved at a reasonable price. This balance challenged MSS developers before 1998 while multimedia capable high-end Central Processing Units (CPU) were almost unaffordable for low-cost MSS applications, but recent price drops of CPUs release the expense pressure that bothered the MSS designers for a long time. For high-end MSS products, Pentium 4 CPUs are definitely qualified; middle class products may choose Pentium® III Processors (Intel Corporation, www.intel.com) with high execution frequency. Even Celeron® Processors (Intel Corporation, www.intel.com) are able to deal with 1 or 2 video capture sources.

Size and Installation

Not all target facilities allow lots of space to install an MSS application. Based on different uses of MSS such as office and automobiles, the size of the control center and installation of sensors should not affect or be affected by any other features of the hosting

environment. For example, the MSS installation on family cars must match the specific electric standard so that it would not noticeably increase the fuel consumption, the system needs to be compact in size and sensors need to be installed in the positions that give best observation without any interference to the environment [5]. Special measurements must be taken in special cases such as the storage component of MSS on airplanes, the "black box" which has a very hard shell to protect it from being destroyed in accidents [6]. Usually the storage components are intentionally hidden.

User Interface

A friendly user interface (UI) may help reduce faults of human operations, enhance the efficiency of the system and reduce the operation cost. In the design phase the UI issue has to be addressed to match the needs of customers. It can be just several buttons on the panel to playback or completely independent professional software that allows full control of the whole MSS application.

Security

Because the data stored by MSS applications can be used as evidence in the court, the integrity, correctness and validity of storage of original data captured by the sensors are very critital. The first aspect of this issue is to secure the transmission over the network. Some approaches [7] and [8] with signal watermarking and data hiding techniques have been studied to protect the digitized signals. In the storage module, before the original data can be saved on disk, cyclic redundancy check bits [9] may be inserted into the data stream and then the encrypted backup may hide the real content of the file [10].

Optional Enhancements

MSS implies that a distributed acquisition of information from multisensory detectors must be transmitted to the remote control center. Those requirements of MSS are different with respect to those of other multimedia applications as some properties of the surveillance problems are very peculiar. Though assigned similar missions, MSS applications may differ in their enhancements according to the installation environments and users.

After receiving and processing the information, the control center is also assigned to broadcast or send the results to the users through network connection (e.g. residential access to internet or intranet). Especially with an Internet connection in the control center, the broadcasting or remote access features are much desired since they allow the customers to monitor the system at almost anywhere in the world.

From the multimedia processing area, the requirements of MSS application are based on a simple fact that the cost of the machine is always lower than man power and the machine is always more efficient. The traditional monitoring by human security guards has been slowly replaced by parallel software-watching with at least some extent of artificial intelligence. Raw signals go through the preprocessing and filtering to remove noise generated in environmental radiation and transmission. Then depending on the changeability of the scene conditions, different scene description and recognition methods may be applied. Automatic learning capabilities are an emerging issue in MSS. The capability of automatically developing models of scenes to be recognized as potentially dangerous event from a training set of presented examples will soon be a key issue for improving end-user acceptance of MSS applications.

Real-time and low-cost can be the two factors that evaluate acceptance of the processing modules of MSS.

The research on signal processing is separated into several directions. Motion detection has been widely implemented in MSS, but the algorithms used in low-cost systems are capable of sensing video changes but lack the ability to identify the moving objects [11]. On another side, the current shape detection techniques [12] may be able to identify an object in a still image. In order to perform tracking and understanding of human motion, current research [13] effects try to combine the two fields and have focused on no-rigid object tracking which uses dynamic models to describe moving objects.

CHAPTER 3

MULTIMEDIA DATA STREAM CONTROL

Overall Architecture

Figure 4 shows my proposed Multimedia Security Surveillance (MSS) system at an architectural level. Three different modules can be developed independently. Data collection module controls the sensors hardware, gathers data from sensors and delivers the data to down-stream modules. If the data came from an analogue sensor, it would have to digitize the data before delivery. The multiplexer is an optional device if the combination of multiple streams into one is required in the installation. The interface between the Data Collection module and the Data Processing module has to be an open standard so that any change on either side will not directly affect the implementation of another. The Processing Center or Control Center will function as the "brain" of the whole system. It processes the data from the up-stream, saves the data to the Storage Center and controls all other modules and user interfaces. The Broadcasting Center acts as a portal to broadcast multimedia stream to the external network and/or allows remote control of the system.

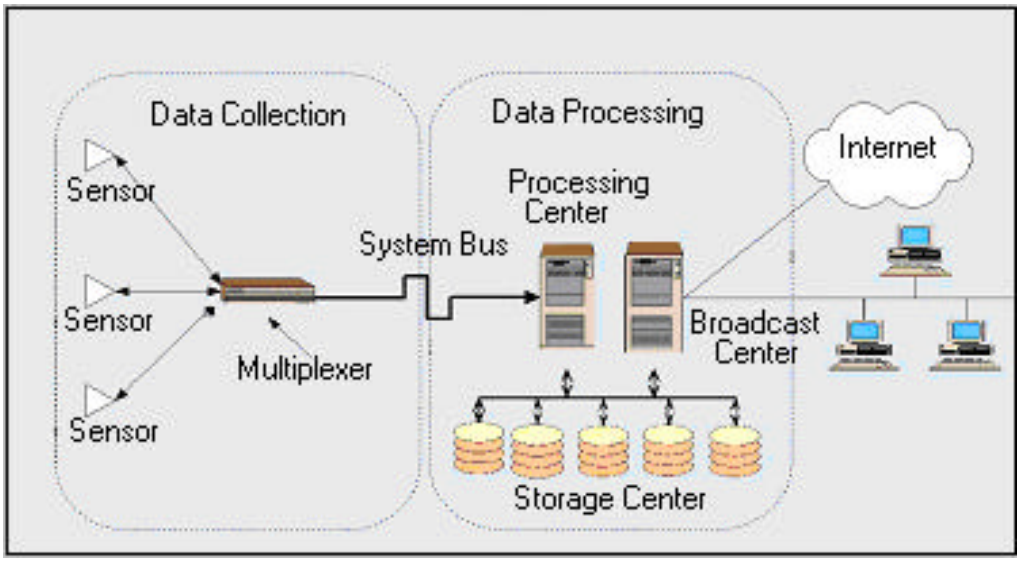

Figure 4: System Architecture

Monitor Devices

Introducing the multimedia monitor devices or capture devices onto the computer is an issue of connecting different electronic entities. The capture devices are electronic equipments that monitor the real world from different aspects and then convert what ever they "feel" into electronic signals. Most of these devices may be thought of as human sense organs but some of them are capable of sensing areas beyond the human senses. Depending on their sensing mechanisms, the classes of sensors are composed of active ones that acquire information about environment through emitting detection signals and measuring the reflections or alterations, or inactive ones that simply receive the environmental information directly. Usual multimedia capture devices are separated into several categories based on their sense areas. The most common ones are tactile, acoustic, laser, radar, and vision sensors.

Acoustic Sensors

They can be either active or inactive. They usually convert the receiving signal into sound (microphone) directly or vision (i.e. Acoustic Radar) for human understanding. They are low cost but have a limited detection range.

Laser-Based Sensors

The kind of active sensors are able to detect the distance and movements of objects through laser reflection signals. But their relative slow scanning speed and low resolution limit their popular adoption on applications.

Radar-Based Sensors

They deploy similar detection mechanisms as Laser sensors, but with more robustness to harsh weather such as rain or fog. Unfortunately drawbacks like high price and low resolution place them on the same level as Laser sensors in the MSS market.

Vision-Based Sensors

These passive sensors are very popular due to their advantage of acquiring visual information directly. They can also be integrated into most applications without changes in the current facility infrastructures. Their shortcomings come from the complex processing computations and low adaptation to environmental changes such as night, fog or direct sunshine [14].

Data Transmission Buses

Before raw signals from capture devices can be understood by computer, digitization is a unavoidable step. The content of this electronic engineering is over the range of this paper [15]. After that, digitized signals are "piped" into the computer through different standard external computer buses.

Universal Series Bus (USB) has been popularly used as a standard hot Plug and Play (PnP) external bus for many hardware devices such as keyboard, mouse and USB hard drives. The obvious benefit of hot PnP comes from the availability of the hardware device in system as soon as the device is physically connected onto the computer without a software setup, system reconfiguration or system reboot. USB 1.0 may carry the data stream at a rate up to 12 Mbps. Different Windows® Operating Systems (Microsoft Corporation, www.microsoft.com) natively support USB though some special USB devices may still need to install drivers [16]. The USB connection can be implemented cheaply because of its simplicity and readily available sensors. Upgraded USB 2.0 [17] can carry much larger data stream but it is far from a widely accepted standard [18].

IEEE (Institute of Electrical and Electronics Engineers) 1394 high-speed bus, also known as FireWire® Serial Bus (Apple Computer, Inc, www.apple.com), was designed to offer high-bandwidth transfer rate for multimedia computer devices, such as digital camcorders, cameras, and videodisc players. Also supporting PnP, the IEEE 1394 bus is capable of delivering data at an amazingly high speed up to 400 Mbps in a asynchronous mode. Some limits such as cable length and extra hubs needed are about to be solved in IEEE 1394 1.2 version.

The traditional Peripheral Component Interconnect (PCI) local bus still stands as a major multimedia interface to computer [19]. Since PCI bus is directly connected to computer system bus, the PCI bus provides high performance on data transfer (350 Mbyte/second). Inconvenience with the PCI bus exists because of the necessary PCI interface card and the relative high price for the parts.

Windows Multimedia Application Programming Interfaces and Driver Models

The introduction of digitized multimedia stream to the computer still doesn't mean the computer may start any intermediate processing. At this moment the data is stored as packets under the operating system kernel, which is beyond the control of the user applications. These data packets are handled by drivers, which are kernel level software pieces executed as interfaces between hardware components and the operating system. Even with the same transmission protocol, drivers for different sensors are definitely different. After the installation of drivers, the functionalities of sensors are completed exposed to the user applications.

Usually the multimedia applications have to handle large amount of data in a very limited time since the audience does care about the continuousness of the multimedia stream, and high-quality digitized multimedia streams flow very fast. The multimedia means the computer will process not only one data stream but commonly at least two data streams (video and audio). The playback procedure has to synchronize the data streams so that the data streams start and stop at the same time and play at the same rate. This process may have to introduce various sources such as local media files, computer networks, terrestrial broadcasts, video cameras, and etc. Another difficult issue is the streams stand on a variety of formats

including Audio-Video Interleaved (AVI), Advanced Streaming Format (ASF), Motion Picture Experts Group (MPEG) format, Digital Video (DV), and Motion JPEG (MJPEG). Multimedia applications usually have very poor portability as they have no way to know the hardware configuration on the end-user's computers and even the most commonly used multimedia devices can be in the hundreds. It can be predicted that these challenges will make the development of the traditional multimedia applications extremely tedious.

Microsoft Video For Window (VFW) was the first Window API for helping developers to process capture, manipulate and store multimedia data. Since it was released for Windows 3.1 in the end of 1992, video capture technology has improved dramatically not only the capture rate but also the capture quality. Although Microsoft improved VFW and tried hard to fit newer releases into the needs of multimedia processing, the self-deficiency of VFW architecture such as 16 bit basis, OS API and hardware driver dependence makes the overall architecture dated to current multimedia needs. Furthermore, if a new architectural layer were built on the top of VFW interface, AVICap, would have to copy buffers very inefficiently from the operating system kernel to the user mode [20].

DirectShow® Application Programming Interface (Microsoft Corporation, www.microsoft.com) is the Microsoft solution to deal with current industrial requirements of high-quality capture, manipulation and playback of multimedia streams. Based on the new driver architecture, Window Driver Model (WDM), DirectShow serves backward compatibility to VFW applications without its shortcomings. WDM video capture aims to provide additional support for the following: USB conferencing cameras, IEEE 1394 devices, desktop cameras, TV viewing and multiple video streams. This support is provided through kernel-based streaming [21].

The DirectShow API is designed to simplify the development of multimedia Windows applications. It tries to separate issues like the hardware differences, data moving and synchronization from the application implementations. DirectDraw® and DirectSound® Application Programming Interfaces (Microsoft Corporation, www.microsoft.com), the two basic components of DirectShow, are designed to take advantage of hardware sound and graphic cards to render the video and audio stream in a high efficiency. The multimedia data is packaged into time-stamped media samples to ensure synchronization [22]. Instead of the direct use of the Windows system API, like VFW, all this DirectShow functions are constructed on Microsoft Component Object Model (COM) technology [23], which is popularly referred in the most developments of Microsoft Windows applications. In DirectShow, the bolts and nuts in this architecture are called Filters, COM components specifically implemented for manipulating multimedia data streams.

Compared to the VFW, the DirectShow/WDM architecture is considerably more efficient. The WDM Stream Class Driver, an important component in WDM architecture, processes multimedia streams and is responsible for invoking the minidrivers, which are dynamic-link libraries (DLL) associated with hardware to support device controls. The minidriver and the WDM Stream Class Driver work together to complete the low-level services for the multimedia stream. Shown in Figure 5, this WDM Kernel-based streaming makes fewer transmissions between the user mode and the kernel mode. During the streaming processing in kernel mode, the DirectShow Filters have no control mechanisms to direct the streams. And DirectShow Filters provide the high-level services for the user applications. This greatly reduces the overhead associated with numerous transmissions between operating system kernel and user modes [21].

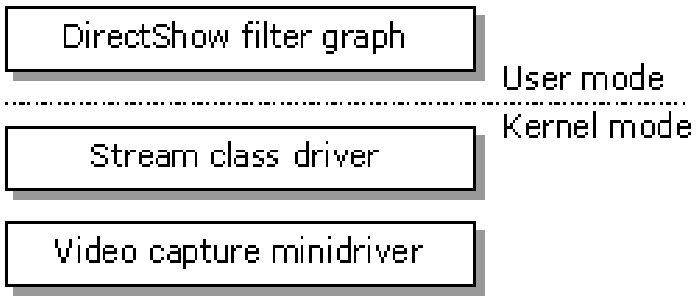

Figure 5: WDM Architecture

The Figure 6 shows the overall modular architecture of DirectShow applications. The DirectShow API completely separates the application from access to multimedia data stream. Even the API itself has no direct privileges to affect the behavior of the data stream as WDM wraps the data in the kernel mode. In another word, the API only "directs" the data stream through the WDM drivers and these kernel mode components accept the commands from DirectShow and take care of the rest of the work. In this way the API stands as a central position in the multimedia processing and connects with all multimedia related software and hardware components or resources, including local files, Internet broadcasting, WDM capture devices, VFW capture devices, sound cards and graphic cards etc., into a single entity that exposes simplified high-level functionalities to the user applications. This architecture isolates a multimedia application from many of the complexities of hardware and software issues associated with multimedia processing. Also this architecture allows the application developers to focus only on the implementation of application logic without the need to consider multimedia data processing details. Furthermore, any software and hardware components can be easily introduced to the DirectShow architecture by adding the WDM standard driver interface. Obviously, the work of hardware driver developers and multimedia processing

developers is greatly simplified as they now have an open standard for the interface of their components.

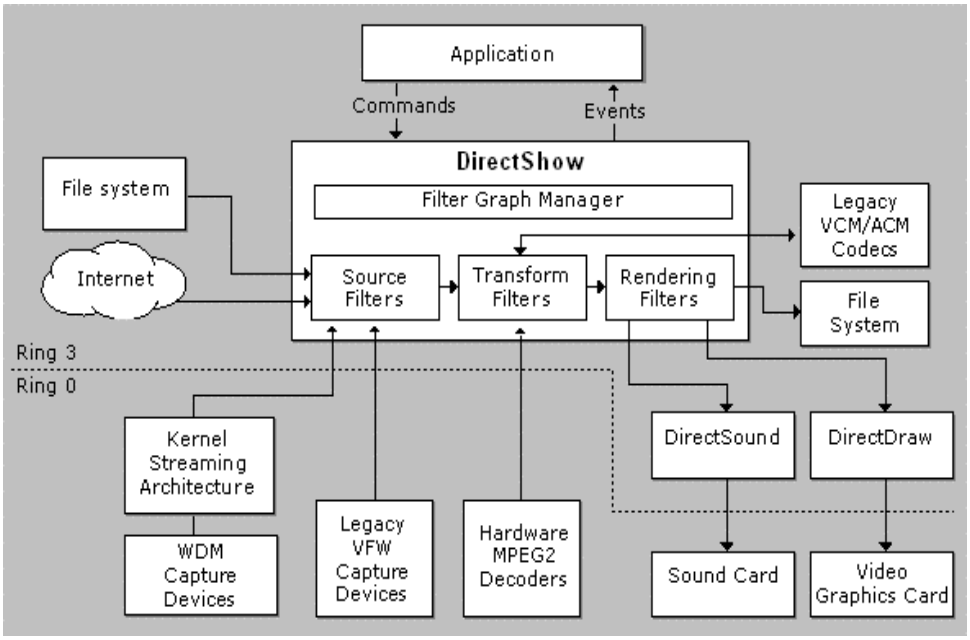

Figure 6: DirectShow System Diagram

Whenever an application needs to conduct multimedia information, through DirectShow API commands, it simply creates a DirectShow Filter Graph, which is the container of any other DirectShow objects, and a Filter Graph Manger, which exposes the interface to control following inserted DirectShow Filters. These Filters, or WDM standard COM components, are where the logic of the multimedia stream processing is implemented.

Filters may be also considered to be modules for controlling and processing the multimedia stream. To achieve desired results from the original stream, usually the Filter

Graph will contain several connected Filters and each one of them evokes a specific manipulation of the stream. According to their functions, Filters are categorized into three different kinds: Source, Transform and rendering. A Source Filter captures original multimedia data from a hardware source such as a capture card and FireWire port or a software source like a file. Then the Source Filter may frame the stream into the packets with time stamps and pass the packets to a Transform Filter. Transform Filters are implemented with multimedia transform functionalities. This transform may be one of many different functions to fit the user's needs. The multimedia compression-related Transform Filters are a particularly useful type of Filters that is capable of compressing a multimedia stream or decompressing a compressed one. In an MSS system, Compression and Decompression Filters are indispensable as the storage media may not be able to save the extremely large amount of data from a multimedia stream without compression. Render Filters represent the final sink of the multimedia stream which would be any output devices including the hard disk, screen and Internet.

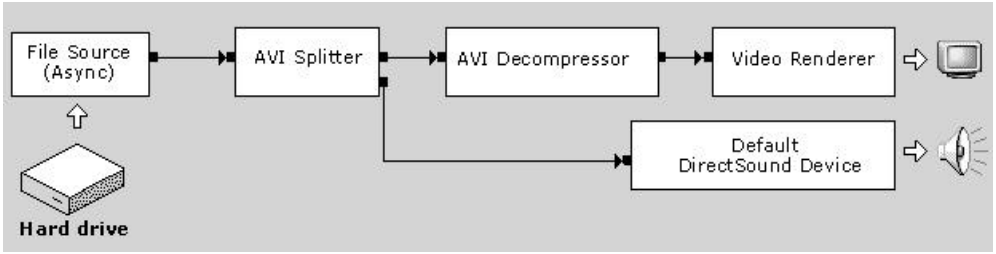

Figure 7: Simple DirectShow Playback Graph

Figure 7 shows a simple DirectShow Graph. In the graph, the source Filter gets the multimedia data from a compressed AVI file and passes it to the AVI Splitter Filter. AVI

Splitter Filter gets the data and analyses the time-stamped media samples in the data. After this, AVI Splitter passes video samples to the AVI Decompressor, which decodes the video frames and passes frames to the video Renderer Filter. The video Renderer Filter plays the video on the monitor. The AVI Splitter directly passes uncompressed audio samples to the Renderer for the audio output device. If the audio stream also were compressed, then an audio decoder Filter between the AVI Splitter and the Render Filter would also be needed.

The example demonstrates how to simply playback AVI files. But it covers the procedures to implement a multimedia-processing module with DirectShow. It is a safe assumption that the more complicated the tasks, the more Filters will be inserted into the Filter Graph. The massive management of the added Filters consists of connecting all Filters, synchronizing data streams and controlling the Graph state changes. Fortunately, DirectShow provides a series of Graph Managers to help with this issue. For a simple program like the example above, all the programmer needs to do is to provide the source Filter and render the Filter. Even in professional multimedia processing, complete manual connection of Filters may still be superfluous.

In MSS applications, the multimedia stream has to be intensively processed before it gets to its final output destination. Even with the assistance of Graph Manager, developers still have to know the Filter connection mechanisms in detail since in a complex Graph, the Graph Manager may not be able to complete the connection or it might result in an unexpected Graph. Filters are connected through their “pins”, distinct COM components associated with Filters. Input pins accept the data stream for a Filter from the output pins of another Filter with a negotiable agreement on the multimedia data format referred to as “Media Type” in DirectShow. Each pin has a property interface to show what kinds of Media

Type that it may accept. Some pins accept a wide range of Media Types, while others are limited. Once the pins are connected, the upstream Filter packs media data as packages and sends them to the downstream Filter through the pin connection. Each package is a COM object named "Media Sample". It contains not only the actual data, but also the time stamps, which is used to pledge synchronization.

Multimedia Data Storage

Storage is an essential element for an MSS application. In a computer-based system, after processing and analysis, information from sensors has to be stored for future use. Following the price drop of hard drive, the storage component is no longer the most expensive part of the system as it was. But even though it is possible to store the raw data directly to the disk, this will surely be expensive and inefficient. Due to the progress of data compression techniques and improvement of PC computation power, compression procedures are found in most MSS even though it is still considered a time-consuming step.

Many video compression algorithms have been developed in the last decade [24]. Motion Picture Expert Group (MPEG) has established a series of compression standard: MPEG-1, MPEG-2, MPEG-4 for medium quality and medium bit rate video and audio compression. The encoded data rate of MPEG-1 is targeted at 1.5Mb/s as this is a reasonable transfer rate of a double-speed CD-ROM player (rate includes audio and video) so that VHS-quality playback is promised by this level of compression. The MPEG-2 standard provides much better Video and Audio playback quality. The MPEG-4 standard, introduced in 1999 and still under development, is aiming at multimedia interactive applications.

The DirectShow Filter framework allows the direct implementation of compression methods. Also commercial and free compression Filters are available.

Since multimedia data requires itself to be transferred and processed in high speed, multimedia data consumes enormous space and bandwidth relative to program files or "text" documents when the control center saves it onto a hard drive. Due to the high transmission rate, the storage process of multimedia data demands far more system resources than other kinds of file storage. A multimedia file system must reconcile the deficiencies of conventional storage subsystems [25][26][27]. For low-cost systems, the modification of file system costs too much in implementation than installation of a hard drive with a higher capacity. Instead, a relatively simpler module for the continuous strorage of data onto disk at background may be implemented.

Multimedia Broadcasting

Although remote control is usually an optional component for an MSS application, when considering the cost of employing a human operative, it has recently become an indispensable asset for lowering the system cost. MSS remote control has three basic functions: remote access, remote administration and broadcasting. Remote access may allow the end-user to access the current multimedia data or saved data and sometimes permits monitoring the status of the system through the intranet or internet. More complicated, the remote administration requires the system to accept the commands from remote users and take corresponding actions such as adjusting the focus and direction of vision-based sensors. Simple reference of current client-server techniques [28] may help the implementation and integration of remote access and administration onto an MSS application with Point-to-Point

access protocol. Realization of the network connection causes a serious concern as security issues involved are very critical, especially in the situation where the connection is through the internet. Therefore common internet security measures including user authentication [29] and transmission encription mechnisms [7][8] should be implemented, especially when the system requires remote administration.

A simple broadcasting implemetation for remote access has been agreed upon on many low-cost systems. But directly implementing a multimedia-broadcasting server can be very costly. Microsoft Media Encoder technology just fits into the criteria. Even with the capability of capture and compression of digital multimedia information, the Media Encoder technology is not a competitor to DirectShow for multimedia information processing due to lack of features associated with the control and edit of the media stream. However, its network-broadcasting feature makes it the best choice [30] for implementing a low-cost multimedia-broadcasting server.

Also based on COM technology and part of Microsoft Media® Technologies (Microsoft Corporation, www.microsoft.com), Media Encoder provides a tool for the Internet or intranet to deliver multimedia information. Though the base implementation of Media Encoder is complicated and hidden from the end-user, the Media Encoder simplifies the whole process of developing a multimedia web server into several steps in a wizard; for the user, just going through the application wizard and filling up some options in the wizard may finish an implementation of a multimedia broadcasting server. The wizard usually explores a few property pages including capture source, compression codec and broadcasting profile. The source for media broadcasting can be a multimedia file (normally an ASF file), hardware capture source or simply the screen of the computer. Since broadcasting the raw data of a

multimedia stream is untolerably slow, compression before sending the data onto the network is indispensable and this can be easily done by selecting a compression codec in the codec profile page. Then in the profile page a port for broadcasting can be declared. After clicking the Broadcasting button, a broadcasting server starts to execute. To integrate the Media Encoder into a program is a similiar task: create an Encoder COM instance, change the property of the instance and run it. On the client side, Microsoft Media Player may be able to receive the multimedia data stream online and render it to the user.

CHAPTER 4

SYSTEM DESIGN AND IMPLEMENTATION

In order to fit the variety of market needs, my Multimedia Security Surveillance (MSS) system was developed on both a compact and standard hardware architectures. It was required to be able to function on both platforms. Both computers have been integrated with direct Universal Serial Bus (USB) and FireWire supports. Please see appendix I for detail specifics.

Development Environment

Cameras with both USB and FireWire connections from a few venders are chosen and tested. I noticed even though many manufacturers provide USB 1.0 capable cameras; only few of them come with DirectShow support and fewer provide a software driver in the new Window Driver Model (WDM) standard. Another option is an analogue-to-digital converter such as LifeView's USB CapView which provides an interface to convert analogue video signal into digital stream with a little compression [31]. This device makes the camera selection much easier as many analogue cameras may be put on the waiting list. I did not find functional USB 2.0 cameras on the market during the research.

Only two to three providers had FireWire capable cameras. iBOT™ (Figure 8) camera (Orange Micro, Inc, www.orangemicro.com) which shipped from Orange Micro is one of them.

Figure 8: iBOT Camera

But the final winner is the Fire-i™ Color Digital Camera (Figure 9) (Unibrain S.A., www.unibrain.com). The Fire-i Digital Camera is considered to be the best home/office IEEE1394 Digital Camera available in the market. Its superior performance clearly distinguishes it from other 1394 or USB cameras. With crystal clear 640x480 resolution and 400Mbps transfer rate, the Fire-i Digital Camera is an ideal device for video conferencing, video capturing, surveillance and monitoring applications. The most impressive feature of Unibrain's Fire-i camera comes from the two FireWire ports on each camera. Via Unibrain's Fire-i Software, several cameras can be connected to one PC and viewed simultaneously. The bundled software allows the user to start monitoring one or all cameras at the touch of a single button, as well as change and adjust all camera settings for each and every camera connected. Appendix III shows the details of the iBOT and Fire-i cameras.

Figure 9: Fire-i camera

Then a simple intranet interface was installed on several office with EtherFast® Cable/DSL Routers (Linksys Group Inc, www.linksys.com), and the Internet access was a Digital Subscriber Line (DSL) service offered by TXU Communication.

Windows® 2000 Service Pack (SP) 3 was installed on the development computer and also the drivers for all necessary hardware. Although Visual Basic® 6.0 Development System (Microsoft Corporation, www.microsoft.com) is able to directly access the DirectShow Application Programming Interface (API), some system level limits and possibly low performance of Visual Basic programming which is a script language running over an interpreter [32] make the Visual C++® 6.0 Development System (Microsoft Corporation, www.microsoft.com) the only choice for the multimedia processing requirement. Both Flash 5.0™ Development System (Macromedia, Inc., www.macromedia.com) and Visual Basic 6.0 were used for the graphic user interface development. Microsoft DirectX® 8.0 Application Programming Interface (Microsoft Corporation, www.microsoft.com) and Media Encoder 7.1 are integrated with Visual C++ 6.0.

Multimedia Processing Module

In DirectX 8.0, Microsoft made more promises on API integrations and System Performance. Furthermore, DirectShow is fully associated into the API. In DirectX 7, DirectShow was an optional component. These changes make the DirectX 8.0 a complete API for multimedia application development. It is capable of not only handling drawing and sound but also gaining access to the video stream.

With the DirectX 8.0 API, theoretically any development environment for Microsoft Windows system is supported to build multimedia applications. But choosing Microsoft Visual C++ is a very natural thing as it is thought to be the best development tool for Windows applications. Before packing DirectX 8.0 with Visual C++, the whole API library was compiled and built under Visual C++. This makes the API accessible within the Visual Studio Integrated Development Environment (IDE).

As mentioned in the last chapter, when an application requires a complex Filter Graph, the manual Filter management becomes unavoidable since the Graph Manager would be unqualified and can even make semantically wrong connections. Bundled with DirectShow, Filter Graph Editor is a powerful visual tool to test the Graph connection design before the implementation starts. As shown in the example below (Figure 10), a simple playback DirectShow Graph is being edited in the Graph Editor.

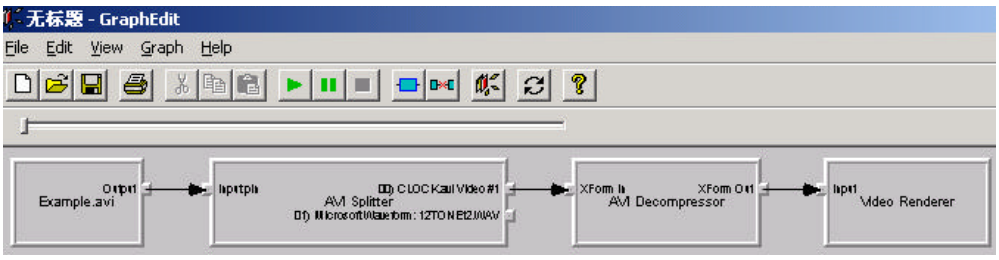

Figure 10 Filter Graph Editor

The source Filter is a simple DirectShow file source Filter that reads an available media file and converts it into a video stream. If the user left-clicks the pin on the file source Filter and choose the render option, Graph Editor will automatically add necessary Filters and connect them to form a complete playback graph. Here source Filter reads an Audio-Video Interleaved (AVI) file, and then passes the data stream to an AVI splitter Filter that separates the original data stream, a mixture of video and audio data into two individual streams. In the Graph Editor, simply right click the connection pins, a new window about the data interface will pop-up and give all information about the delivering data stream format, the result of the connection negotiation of the two Filters. Since the video stream is usually compressed in some way, the AVI Decompressor is added. This Microsoft wrapper Filter contains popular Depression codecs. As AVI file format supports all kinds of compression, developers may have to implement or install a decompression Filter for their own needs. An AVI multiplexer (used to combine video and audio streams) before the File Writer is necessary.

In this test, manually adding and removing certain Filters are allowed. In the Graph command of the Editor, the user may open the list of current registered Filters in the system registry and select a Filter for the Graph test (Figure 11).

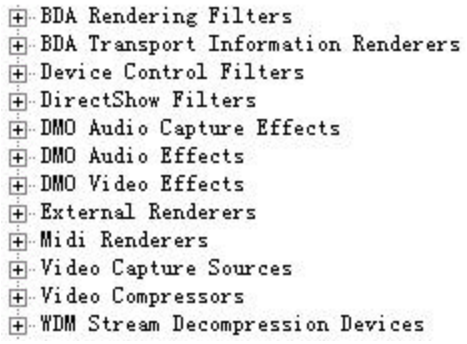

Figure 11: Filter List in the System Registry

For testing the correctness of the Graph, the user may click the run button on the tool bar and execute the Graph. In most cases, the implementation is just to follow the procedures of the above test. The following code gives an outline of implementing this simplest Graph in C++.

Step 1: Create an instance of the playback Filter Graph manager by CoCreateInstance function, which is usually used to initialize a Component Object Model (COM) entity.

```
IGraphBuilder * pGraph ;    // Pointer of a graph manager object interface
CoCreateInstance (CLSID_FilterGraph, NULL, CLSCTX_INPROC,
                  IID_IGraphBuilder, (void **) &pGraph);
```

Step 2: The method RenderFile automatically selects, adds and connects the right Filters usually including File Reader, Decompressor and Renderer to render the AVI file.

```
pGraph->RenderFile(L"C:\\EMAIL1.avi", NULL);
```

Step 3: Through the Graph Manager, the ImediaControl interface can be queried. It controls the media stream actions such as run and stop.

```
pGraph->QueryInterface(IID_IMediaControl,(void**)&pControl);
```

```
pControl->Run();
```

The intelligent completion of the Filter Graph through RenderFile function is literally called “Automatic Connection” or “Smart Connection”. Although it saves a lot of efforts in many multimedia processing cases, “Semiautomatic” and “manual” connections would help to solve a few critical technical issues when applications like MSS gets into the details of controlling the data stream.

When the playback of audio is needed, the audio playback Filter may be inserted into the Graph and give the modification as Figure 12.

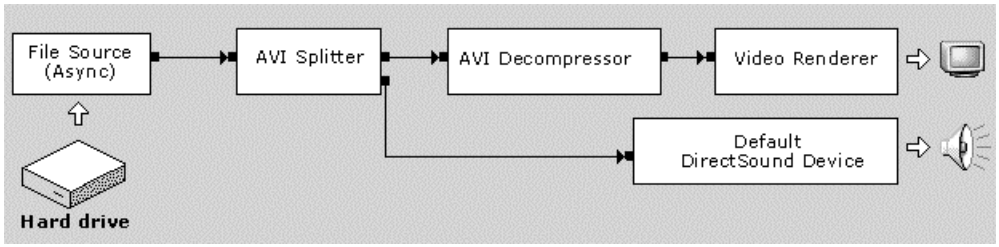

Figure 12: Graph with Audio Playback

In my MSS, the media data stream will come from the multimedia capture devices. But like reading files through a File Source Filter, there can be more than one available source, for example installation of multiple cameras. Additional customized Compression Filters will be introduced in the Graph; plus a Grabber Filter was to be implemented and tested. Because a simple Filter Manager may not be able to deal with all these complexities, the most powerful Manager, CaptureGraphBuilder2, was chosen to conduct the MSS Filter Graph as the following:

```
ICaptureGraphBuilder2 * g_pCapture;
```

```
CoCreateInstance (CLSID_CaptureGraphBuilder2, NULL, CLSCTX_INPROC,
IID_ICaptureGraphBuilder2, (void **) &g_pCapture);
```

A search of the Capture Source Filters gives an enumeration of all the capture sources in the system (Figure 13). Once the desired source is found or is selected by the user, a Moniker object is bound to it to expose its functionality. Then the addition of the source filter forms a Graph with a single Filter.

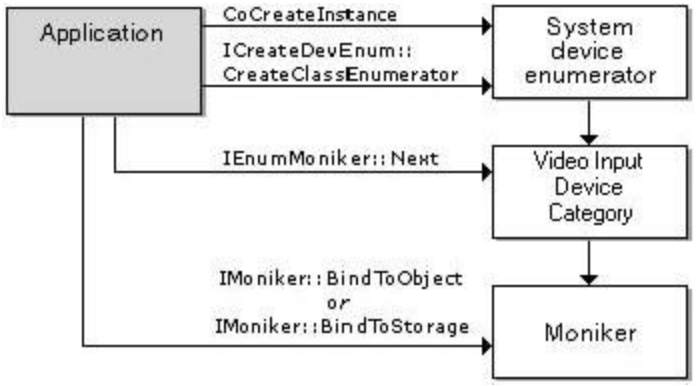

Figure 13: Search the Capture Source Filter

Through

```
IBaseFilter * g_pRender;
IFileSinkFilter * g_pSink;
g_pCapture->SetOutputFileName(&MEDIASUBTYPE_Avi, Filename, &g_pRender, g_pSink);
```

another two important filters, Multiplex Filter and File Writer Filter, are inserted into the Graph. The Graph now looks like Figure 14.

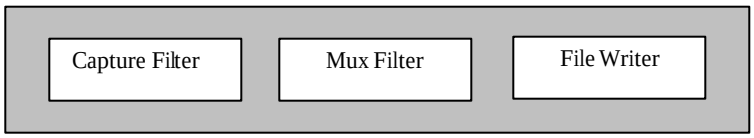

Figure 14: Three Unconnected Filters in Graph

Like the RenderFile funtion, the RenderStream function of the GraphBuilder2 interface will connect the Filters with a “Smart Connection” and will construct a complete Graph as Figure 15.

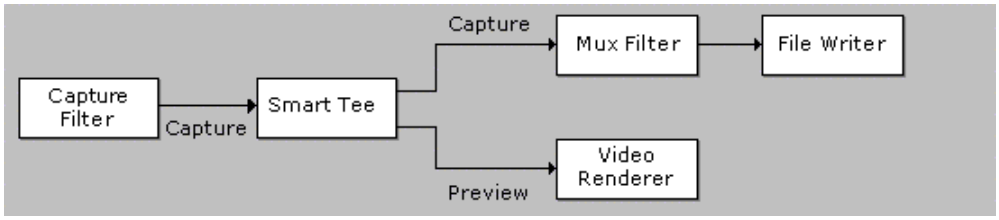

Figure 15: Completed Graph

The Manager inserts the Smart Tee Filter and Video Render Filter. The Smart Tee Filter is capable of splitting a single multimedia stream into two identical streams; the Video Render Filter will playback the duplicate stream on screen. The execution of the above Graph will save the original captured multimedia stream on the disk. Even with 320*280 size, 64K color and 30 frame/second standard, it will consume a very large amount of disk space shortly (about 8Mbyte/second), which is unpractical and unacceptable.

Though an AVI compressor is shipped with DirectShow. Microsoft mentioned it wraps almost all popular compression codecs and the functions to select the best compression codec

in the Graph depending on the input data format. Unfortunately the compression part of a DirectShow Graph turns out to be far more complicated than simply inserting the AVI Compressor into the Graph and connecting it with other Filters. The complete procedure is a similar enumeration (Figure 16) as finding the Capture Source Filter above, but the COM category becomes Compressor this time.

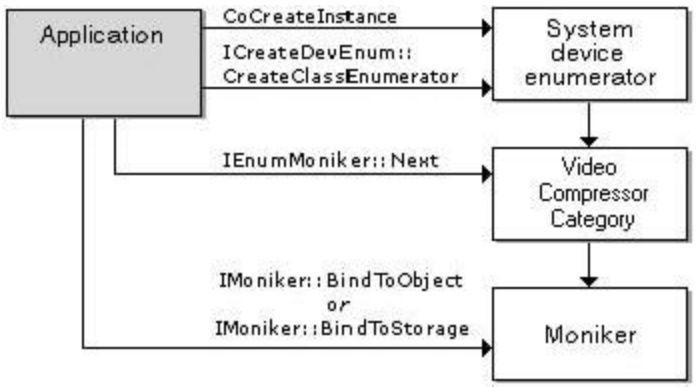

Figure 16: Search the Compression Filter

In Graph Editor, a list of available Compression Filters is retrieved from the system registry (Figure 17). Some of them were installed with DirectX or other multimedia application installations. When a third party Filter or customer Filter is needed the Filter executable files have to be saved on disk as well as the registration of the Filter to the system must occur.

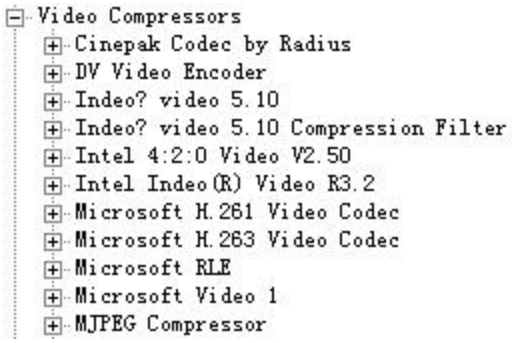

Figure 17: Registered Compression Filter in the System Registry

Frequently, the end user wishes to watch the uncompressed stream from a terminal (because the quality of the video will be perfect) and save the compressed stream into disk to save space. Therefore, the best position for the Compressor Filter is between Smart Tee Filter and the Multiplex Filter that is just before the File Writer. At this moment the Graph has been fully connected, so a manual disconnection of Smart Tee and Multiplexer must be done before the insertion and then connections of selected Compressor Filter and the two disconnected ones afterwards. This finishes the construction of the following Graph (Figure 18).

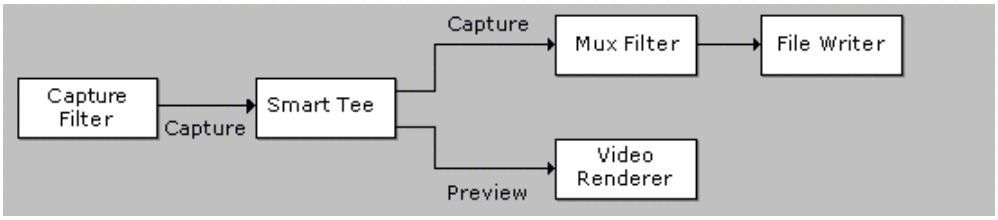

Figure 18: Completed Graph

The last step, a customized Grabber Filter is constructed and added to the Graph. The Grabber Filter is a Transform Filter that is capable of grasping a still image from a moving video stream. It is very useful in adding customer features, for example, a motion detection

interface. DirectShow provides an incomplete Grabber Filter that simply outlines how to grab samples in a media stream and leaves the detailed implementation for the application developers. Based on the example, I built a functioning Grabber Filter, which was able to capture a single image from a video stream, and added it into the Graph above (Figure 19).

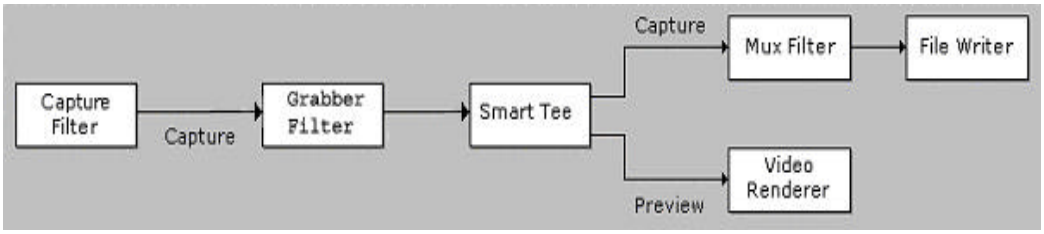

Figure 19: Completed Graph with a Grabber Filter

Graphic User Interface

The user interface (UI) of the control center is the client for the video processing as well as the server side for the multimedia broadcasting. Any programming tool that is able to implement the UI is considered to be a potential candidate. But the UI of MSS applications requires not only the friendliness and the conveniences of the UI presentations but also minimum consumption of system resources. At the same time, it must be a thin piece of the whole development processing. Similar UIs were implemented and tested for my MSS application with several different tools such as Flash, Visual C++ and Visual Basic. It turns out that Flash UI gives the best visual effect but takes the most memory and Central Processing Unit (CPU) cycles. The Visual C++ UI is the most efficient but the least visually attractive; furthermore, implementing good Visual C++ UI is time consuming. My current UI

is developed with Visual Basic as shown in Figure 20. This Visual Basic module directly controls the behavior of the multimedia-processing module by passing WM_COPYDATA system event to the DirectShow C++ application. The DirectShow Control Center responds to this event and takes the necessary actions. They are separate processes in the MSS application.

Figure 20: User Interface

Also, direct controls may go through the video preview window. Each item in the right-click popup menu exposes an associated property page with the Filter objects [33] in the Filter Graph. So on the UI, user can easily track the status of the Capture Filter, Compression Filter and Graph Manager etc. and change their behavior in the allowed ranges as well.

Storage Module

In addition to a high-capacity hard drive (40 Gbyte), a background module associated with the DirectShow processing module monitors the status of the hard drive. Whenever the free space on the disk is less than 500 MByte, it enables its searching function to find a set of the oldest archives and erase them from the disk.

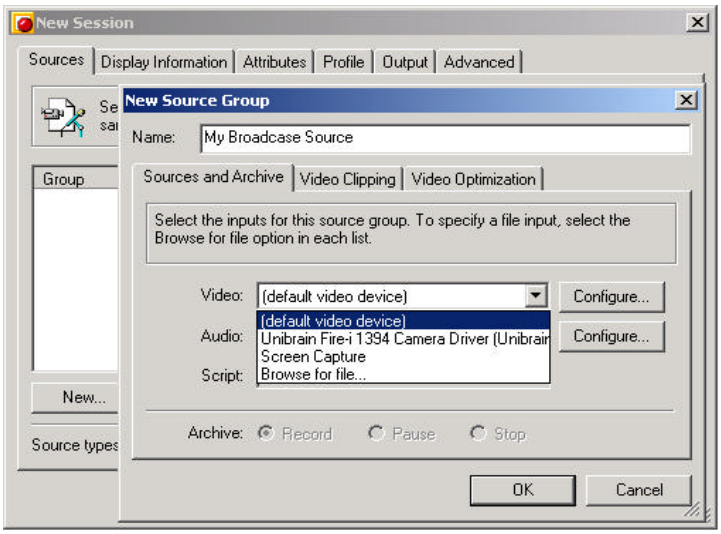

Figure 21: Encoder Capture Source Configuration Dialog

Broadcasting Center

The implementation of the Media Encoder broadcasting module starts from the dialog box shown as Figure 21. When the DirectShow processing module is running, the capture hardware is locked by the DirectShow Graph, and other programs have no access to the multimedia data stream. In order to share the hardware resource, some implementation options are possible such as using a “bridge” Filter to access the multimedia stream in the Graph. Directly capturing the screen turns out to be a very convenient solution particularly

when multi capture sources are enabled. Since the Encoder is reading the screen memory block, there is little performance loss and the sacrifice of some video quality is irrelevant to the unavoidable suffering found in the network transmission.

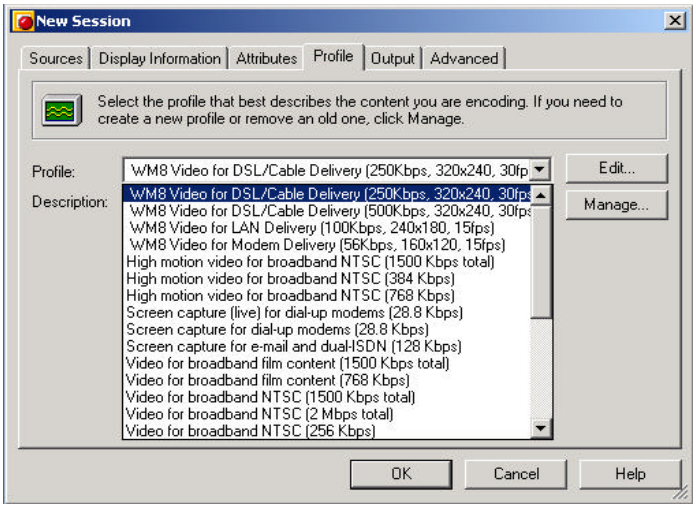

Figure 22: Encoder Profile

The Encoder profile allows the selection of the ways the users want the multimedia stream to be sent onto the network (Figure 22). Although it is defined in the profile, the actual data delivery depends on the server's computation power, network capacity and traffic [34]. After claiming the output port and network settings as in Figure 23, the broadcasting server is controlled from the main panel (Figure 24).

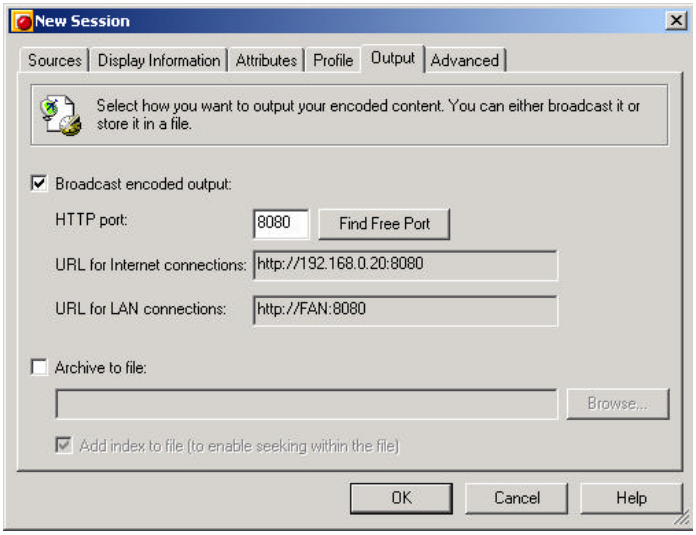

Figure 23: Encoder Output Configuration

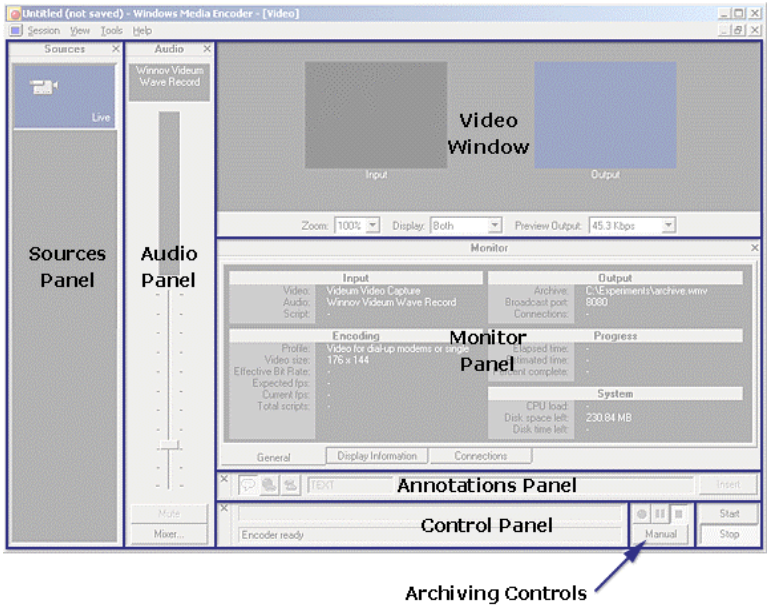

Figure 24: Encoder Control Panel

CHAPTER 5

ANALYSIS AND RESULTS

Test Environment

The full version of the Multimedia Security Surveillance (MSS) application on the hardware mentioned in chapter 4 was analyzed. And the stability of the compact version of the MSS application was also tested.

System Stability

The value of software releases for the application provides a controlled addition of the functionalities, while at the same time providing increased reliability. That differs from other kinds of software in the MSS application development, the two aspects of system stability, are system durability and performance robustness and require more concern than any other stability issues. System durability measures how the system maintains working status without human backup. A steady system performance during a long time frame is equated to robustness.

The durability of the developed MSS system is tested by running the application for a very long time and simultaneously tracking the system conditions. The MSS application showed outstanding stability in the tests. In a time frame of 30 days, the running MSS with 4 video captures did not cause any critical system failure such as system hang-up, capture pause or memory leakage. Both memory usage and Central Processing Unit (CPU) cycles were in a 3% range of vacillations.

Considering the possible installation of the MSS on vehicles, necessary vibration tests were also performed on the MSS application and similar results were obtained on a bobbing machine as well as with direct installation on vehicles.

Performance

The setup of the full version of the MSS application with relatively strong computation power may give us clear comparisons of system status with different execution situations. Instead if a less powerful or normal system were used, the tests would push the machine to its limits before the most demanding tests could be carried out. Figure 25 and Figure 26 show the CPU occupation in percentages and memory usage when only previews with different numbers of cameras were turned on without any capture, compression and broadcasting. As expected, both parameters increase in a linear mode as the number of cameras used. When previewing the 4 video sources, the CPU is 30% occupied and memory usage is at 95 Mbytes. A noticeable jump of use in the system resource appears when the capture option is on; for the CPU occupation, capture option drives a distinguishing gap of about 30% over preview and also takes 18 more Mbytes of memory for every additional camera. Even with dual Pentium III CPUs, when the fourth camera is being captured, the CPUs are running at their full capacity (Figure 27 and Figure 28). It turns out that turning on the broadcasting module pushes the system to its limits. Therefore MMiTG company recommended a system with a Pentium IV processor and at lest 1.5 Gbyte memory for executing the optional broadcasting module over the standard MSS application.

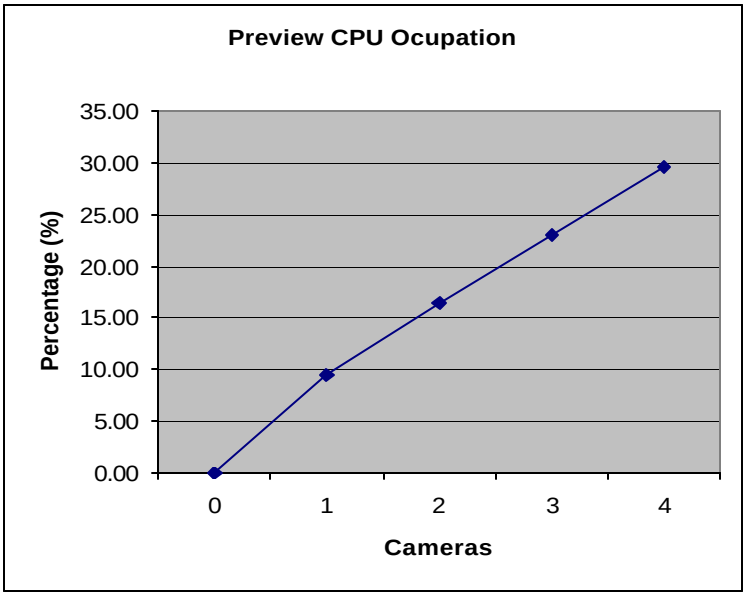

Figure 25: Preview CPU Occupation

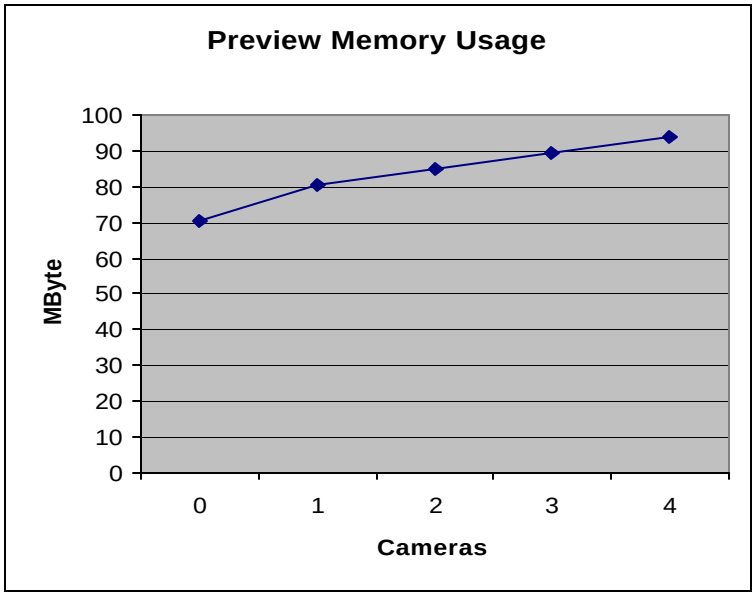

Figure 26: Preview Memory Usage

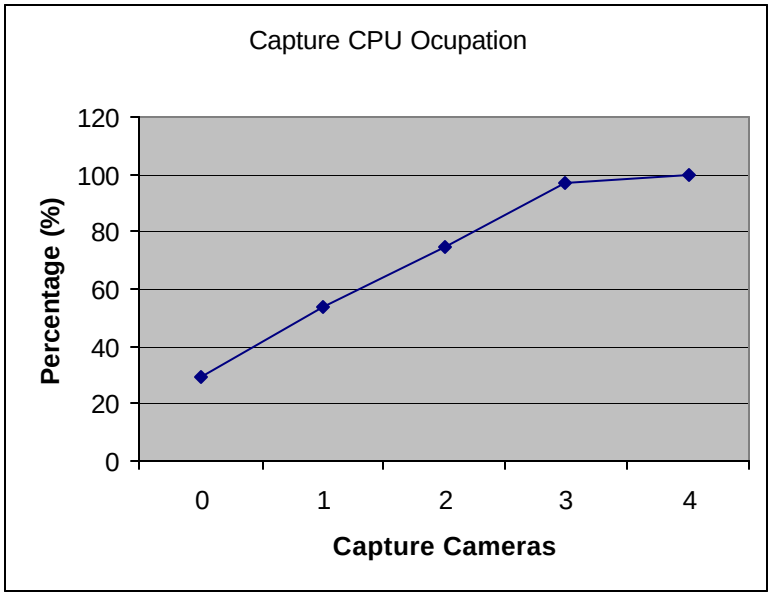

Figure 27: Capture CPU Occupation

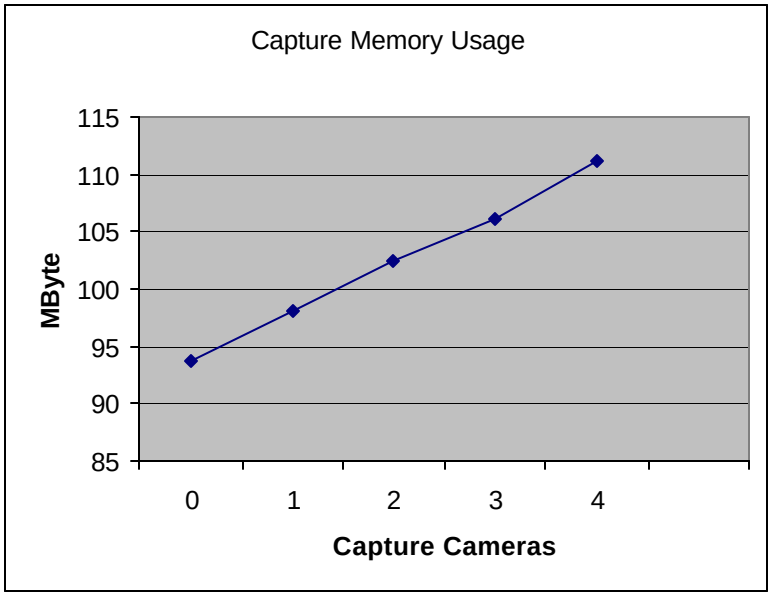

Figure 28: Capture Memory Usage

In Figure 11, a list of current registered compression codecs or Filters are enumerated from the system. I tested not only every codec in this list but also many other codecs on the market. The conclusion was the same as many compression research results. Motion Picture Experts Group (MPEG) 4 type compression codecs give the best quality at a high compression ratio [35]. Table 1 shows a simple comparison of the most popular MPEG4 codecs. The MSS application is independent of the compression codecs and the user may choose whatever codec they want. The default codec was set to Divx MPEG4 Low Motion because of its higher compression ratio than the Microsoft counterpart at the same quality. Additionally Divx codecs can be freely downloaded from the Internet [36] and even the source code as well.

Codec	**3.75f/s**	**7.5f/s**	**15f/s**	**30f/s**
Divx Mpeg4 Fast Motion	219.4KB 3.6142Kb/Second	384.4KB 6.4152Kb/Second	636.3KB 10.843Kb/Second	1122.2KB 18.650Kb/Second
Divx Mpeg4 Low Motion	395.8KB 6.52Kb/Second	716.2KB 12.111Kb/Sec	1287.4KB 21.472Kb/Sec	2283.0KB 38.139Kb/Sec
Microsoft Mpeg4 Codec V1	571.2KB 9.51Kb/Second	1112.6KB 18.609Kb/Sec	1876.6KB 31.303Kb/Sec	3603.0KB 60.154Kb/Sec
Microsoft Mpeg4 Codec V2	523.0KB 8.63Kb/Second	944.8KB 16.147Kb/Sec	1760.0KB 29.396Kb/Sec	2978.2KB 49.69Kb/Sec

Note:
(1) Test Environment: PIII550, 256M memory, Unibrain FireWire Camera,
Windows 2000 Server, DirectX 8.1
(2) The length of original video clip is 5 minute; the video size is 320 * 240, and the video format is Y422.

Table 1

Though Fast Motion MPEG4 codecs compress more due to their dynamical compression algorithms, the trade-off is in the quality of the resulting video stream. They

usually damage some of the details in the original video stream, and this becomes very important if the video files are to be used as evidence on court.

Cost Analysis

As the result of the terrorist attack of September 11, 2001 in the United States, security needs are increased dramatically. Since the aim of the MSS system is the security application market of civilian facilities, a correct price position of the MSS system is critical to the success of the application on the market. However, in order to provide security solutions to a wide range of customers, necessary scalability and flexibility were considered during the development and engineering of the system along with the target pricing level. The possible applications of the MSS system include the following locations:

- Personal offices.
- Automobiles such as school buses.
- Public areas with limited size such as middle size restaurants.
- Sensitive cashier areas.

Based on the market investigation [37] in the United States, depending on different features and options, more than 90% of residential users in the above categories are willing to invest from $2000 to $5000 for a single MSS system in their facilities.

The basic MSS configuration is listed below:

- Compact Personal Computer (PIII 1GHz, 1 Gbyte RAM, 40 Gbyte Hard Drive).
- Camera (up to 4 for a single system).
- Installed software (UI, DirectShow Processing Center, Broadcasting Server etc.).

The total cost lies in a range of $1500-$2500 for the different selection of the hardware and $1000-$2000 for the software. Optional features can be added at an extra cost. One possible big change on the current architecture would modify the Window Media broadcasting engine into a Point-To-Point download server or a secured broadcasting service to reduce the network traffic. The integration of special options for video processing including motion detection, object identification and event alert will take advantage of the DirectShow Filter technology just like the implementation of the Grabber Filter. However the selection of different algorithms and possible software licenses from other companies very much depend on the user requirements and the cost consideration.

Customized Enhancements

I did not implement any special function that is not on the requirement list of video processing after the Grabber Filter became a part of the Filter graph. The options are open to the customers. Diversity of possible add-on features as mentioned in Chapter 2 and some features are possibly requested by specific customers were other reasons that MMiTG chose to end development at this point. Extracted from customers' feedback on the current version, further development on frequently requested optional features is under investigation and some enhancements have be scheduled to add to the current application.

Because the DVD-like video stream goes through the entire video processing system from one Filter to another, high-quality sampling can be directly added on the DirectShow Graph as an interface on a Filter so that more processing on the captured sample in either still image or video segment is allowed. The Grabber Filter is considered to be a sampling Filter since it captures a single image from video stream and saves data in RGB format. From this

interface, implementation of video processing extensions that need a continuous sampling is straightforward.

A good example is a simple motion detection algorithm which takes the RGB structured images and compares the difference between two consecutive images in a narrow time frame.

Motion detection is the most common request from customers. Issues around this feature include action before, during and after motion is detected. In chapter 2, a few algorithms are listed for detecting motion in a video stream and selection of the best one for customers depends upon the monitoring environment. Like the Grabber Filter, an implementation of a Filter to wrap the chosen algorithm is believed to be the most efficient development path though other approaches exist.

If there is no motion, the MSS should be able to reduce the storage requirement so that unnecessary saving of unchanged video can be avoided. This class of video stream (without any changes) happens more than 90% of the capture time in most cases. The process will be back online just the moment after a moving object is discovered through the detection algorithm. Also an alert can be broadcasted either to the administer through a wireless phone message or to the police office. In order to obtain an even better video, a sharpening Filter may be applied to the area where the objects are active.

Shape distinction is also a widely requested feature such as detecting gun-like objects in the check-in site of airport. Algorithms to resolve certain shapes or distinguish predefined objects in a video stream or in a still image vary in many ways for a variety of MSS applications [38]. Although the same solution as motion detection is one approach for this add-on, some requirements may need special modifications to the original video stream, for example,

filtering to distinguish the object from the background [39]. In some cases, necessary database backup is indispensable such as license-plate recognition for traffic control [40].

According to the specific requirements, some other enhancements on video processing, media broadcasting component and remote administration may plug into the MSS application. Though a single Filter graph can only have one capture source, an application may manage several Filter graphs so that extraction of the compression Filters from the Filter graphs can be more efficient. The combination of current multiple compression processes into one process may certainly increase the complexity of the program, but its obvious benefits make it a unavoidable step in improving the performance of the MSS system.

In this case where a customer would need a guarantee of the integrity of the saved multimedia files, a watermark or cryptography algorithm would be applied on the multimedia stream [41].

WORKS CITED

1. Herman Kruegle, *CCTV Surveillance: Video Practices and Technology*, St. Louis, MO: Butterworth-Heinemann, 1996, pp.11-40

2. Elia T. Zureik and David Lyon, *Computers, Surveillance, and Privacy*, Minneapolis, MN: University of Minnesota Press, 1996.

3. Jon C. Leachtenauer and Ronald G. Driggers, *Surveillance and Reconnaissance Imaging Systems: Modeling and Performance Prediction*, Norwood, MA: Artech House, Incorporated, 2001, pp.165-178

4. Gian Luca Foresti, Carlo S. Regazzoni and Petri Mahonen, *Multimedia Video-Based Surveillance Systems Requirements, Issues and Solutions*, New York, NY: Kluwer Academic Publishers, 2000, pg.24

5. Philip Walker, *Electric Security Systems*, Third Edition, St. Louis, MO: Butterworth-Heinemann, 1998, pp.248-249

6. G. L. Friedman, "The trustworthy digital camera: restoring credibility to the photographic image", *IEEE Trans. Consum. Electron.*, vol. 39, 1993, pp.905-910

7. F. Mintzer, G. W. Braudaway, and M. M. Yeung, "Effective and ineffective digital watermarks", *in Proc. ICIP'97, IEEE Int. Conf. on Image Processing, Santa Barbara, CA*, vol. III, 1997, pp.223-226

8. F.Hartung and M.Kutter, "Multimedia Watermarking Techniques", *Proceedings of the IEEE*, vol. 87, no. 7, 1999, pp.1079-1107

9. James F. Kurose and Keith W. Ross, *Computer Networking, A Top-Down Approach Featuring the Internet*, Boston, MA: Addison-Wesley Publishing, 2001, pp.389-391

10. Remagnino, P., Maybank, S., Fraile, R., and Baker, K., *Advanced Video Surveillance System, Automatic visual surveillance of vehicles and people*, New York, NY: Kluwer Academic Publisher, 1995, pp.95-105

11. Paolo Remagnino, Carlo S. Regazzoni, Graeme A. Jones, and Nikos Paragios, *Video-Based Surveillance Systems: Computer Vision and Distributed Processing*, Part II, New York, NY: Kluwer Academic Publishers, 2001.

12. Karmann, K. P., Brandt, A., and Gerl, R., "Moving object segmentation based on adaptive reference images", In Proc. 5th European Signal Processing Conf., Barcelona, 1992, pp.951-954

13. Gerard Honey, *Electronic Protection & Security Systems*, Louis, MO: Butterworth-Heinemann, 1996, pp.1-19.
Gian Luca Foresti, Carlo S. Regazzoni and Petri Mahonen, *Multimedia Video-Based Surveillance Systems Requirements, Issues and Solutions*, New York, NY: Kluwer Academic Publishers, 2000, pg.14

14. Gian Luca Foresti, Carlo S. Regazzoni and Petri Mahonen, *Multimedia Video-Based Surveillance Systems Requirements, Issues and Solutions*, New York, NY: Kluwer Academic Publishers, 2000, pp.25-26

15. Richard J. Hartley and Keshab K. Parhi, *Digit-serial Computation*, New York, NY: Kluwer Academic Publishers, 1995.

16. MSDN, "Universal Serial Bus Support for Windows CE 2.1", Microsoft Corporation, January 2002: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnce21/html/usb.asp>.

17. Don Anderson and Dave Dzatko, *Universal Serial Bus System Architecture*, Boston, MA: Addison Wesley Longman, Inc., 2001, pp.13-24

18. Steven McDowell and Martin D. Seyer, *USB Explained*, Upper Saddle River, NJ: Prentice Hall PTR, 1998, pp.1-10

19. Tom Shanley and Don Anderson, *PCI System Architecture*, Boston, MA: Addison Wesley Longman, Inc. 1999, pp.7-14

20. MSDN, "WDM Video Capture Overview", Microsoft Corporation, January 2002: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndevice/html/vidcap.asp>.

21. MSDN, "About WDM Video Capture", Microsoft Corporation, January 2002: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dx8_c/directx_cpp/htm/aboutwdmvideocapture.asp>.

22. MSDN, "DirectShow System Overview", Microsoft Corporation, January 2002: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wcegmm/htm/_dxce_dshow_directshow_system_overview.asp>.

23. Dale Rogerson, *Inside COM: Microsoft's Component Object Model with Cdrom*, Redmond, WA: Microsoft Press, 1996.

24. Al (Ed.) C. Bovik, Jerry D. Gibson and Al Bovik, *Handbook of Image and Video Processing*; San Diego, CA: Academic Press, Inc., 2000, pp.1-20

25. Bitton, D., "Disk Shadowing", *Proc. 14th Intl. VLDB Conf.*, Los Angeles, CA, 1988, pp.331-338

26. Gemmell, J. and S. Christodoulakis, "Principles of Delay-Sensitive Multimedia Data Storage and Retrieval", *ACM Trans. of Information Systems,* Vol. 10, No. 1, 1992, pp.51-90

27. P. Venkat Rangan and Harrick M. Vin, "Efficient Storage Techniques for Digital Continuous Multimedia", *IEEE Transactions on Knowledge and Data Engineering*, 1993.

28. Douglas E. Comer and David L. Stevens; *Internetworking with TCP/IP: Windows Sockets Version -- Client-Server Programming and Applications*, Vol. 3, Upper Saddle River, NJ: Prentice Hall PTR, 1997, pp.123-130

29. Richard E. Smith, *Authentication: From Passwords to Public Keys*; Boston, MA: Addison Wesley Longman, Inc., 2001, pp.313-450

30. MSDN. "Overview of the Windows Media Encoder SDK", Microsoft Corporation, January 2002: < http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wmencode/htm/sdkoverview.asp >.

31. <www.lifeview.com.tw>.

32. MSDN. "Compiled vs. Interpreted Applications", Microsoft Corporation, January 2002: < http://msdn.microsoft.com/library/default.asp?url=/library/en-us/vbcon98/html/vbconcompiledvsinterpretedapplications.asp >.

33. MSDN. "Building COM Property Pages with the Active Template Library", Microsoft Corporation, January 2002: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dncomg/html/comwatl.asp >.

34. MSDN. "An Introduction to Windows Media 8 Encoding Utility", Microsoft Corporation, January 2002: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwmt/html/wmencodutil.asp >.

35. Peter D. Symes, *Video Compression*, New York, NY: McGraw-Hill, 1998, pp.1-16

36. <www.divx.com>.

37. *U.S. Closed-Circuit TV & Video Surveillance Market Report (2001)*, Newtown, CT: J.P. Freeman Co., 2001.

38. Z. Li and H. Wang, "Real-time 3-D motion tracking with known geometric models", *Real Time Imaging*, Vol. 5, 1999, pp.167-187
P.J.L. Van Beek, A.M. Tekalp, N. Zhuang, I. Celasun and M. Xia, "Hierarchical 2-D mesh representation, tracking and compression for object-based video", *IEEE Transaction on Circuits and Systems for Video Technology*, Vol.9, 1999, pp.617-634

39. S. Bouchafa, "Contrast invariant motion detection – Application to abnormal crowd behavior detection in subway corridors." PhD Dissertation, *Univ. Paris VI/Inrets*, 1998.

40. Barroso J., A. Rafael, E. L. Dagless, and J. Bulas-Cruz. "Number Plate Reading Using Computer Vision." *In Procs IEEE Int. Symposium on Industrial Electronics*, 1997

41. Reference 7 and 8.
B. Schneier, *Applied Cryptography*, New York, NY: John Wiley and Sons, 1999.