Remote Video Surveillance System Based on DirectShow and Image Compression

Shangfu Gong, Ligang Wu, Yanjun Wang
College of Computer Science and Technology, Xi'an University of Science and Technology
Xi'an, 710054, China
E-mail: gongsf@ xust.edu.cn

***Abstract*—Video surveillance, convenience and rich information, has been widely used in security, protection, monitoring and other occasions, and has already been one of the most important precautionary measures in commercial, residential and transportation areas. However, considering the massive data transmission needs and higher real-time requirements for video surveillance, a remote video surveillance plan has been put forward on the analysis basis of Microsoft DirectShow and Image Compression. This plan adopts the C/S structure, adapts to the requirements of real-time video transmission, with a better fluency. In addition, the picture clarity satisfies the application requirements.**

Keywords-directShow technology; image compression; video surveillance; filter technology; video transmission

I. INTRODUCTION

Video surveillance, due to its convenience, intuitive and rich content, has been widely used in security, monitoring, and other occasions, which is an important prevention means in the areas of commerce, residence, and transportation, etc. In recent years, with the more rapid development and universal application of communication technology, control technology and computer network technology, video surveillance via network is possible [1]. Remote video surveillance is mainly used to show the real-time information, such as video and image, on the remote devices through the network transmission, so that people can immediately know what is happening in the area being monitored, to avoid unnecessary losses. Using VC++ programming techniques, combined with DirectShow and Image Compression, a complete solution for remote video surveillance has been put forward, which meets the requirements of real-time picture quality of video surveillance.

II. DIRECTSHOW OVERVIEW

A. Basic Theory

DirectShow is a multimedia development package based on Windows platform by Microsoft, for simple and efficient processing of stream media, which is of great support to the capture and playback of stream media [2]. DirectShow is a development framework based on modules. Each functional module is a Filter, using the Component Object Model (COM). DirectShow adopts a model named Filter Graph to handle the entire data streams. Each filter in Filter Graph forms an "assembly line" to work together in certain order [3]. The system is shown in Figure 1. Filter can be divided into three categories according to its functions.

- Source Filters： Mainly used to obtain data. The data sources can be files, video cameras. And then transmission will be done;
- Transform Filters: Responsible for the data format conversion, and then transmitting the data;
- Rendering Filters: Responsible for the final destination of the data, the data delivery to the graphics card, and sound card for multimedia presentations, or output to a file for storage.

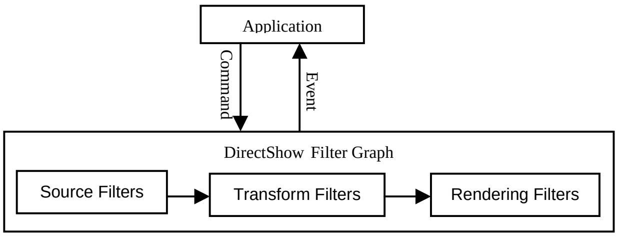


Figure 1. DirectShow architecture

B. DirectShow Development Process

Generally, the application programming of Direct Show has three steps:

- Create a Filter Graph Manager.
- Create a complete Filter link according to the actual application.
- Call the interface method on the Filter Graph Manager to control and complete the event interaction between Filter Graph Manager and the application.

III. IMAGE COMPRESSION TECHNOLOGY

Image compression is a technology to represent the original matrix of pixels with relatively little or no loss, which is also known as image encoding. The original image data of this system is in BMP format, without any compression. Therefore, the amount of data is very large via Internet transmission. As this system needs to transfer large amount of image data to achieve real-time remote video playback, the image compression technology should compress the images into JPEG. JPEG is a lossy image format proposed by the ISO and the International Telephone and Telegraph Board, which is widely used because of the small space occupation and highly image quality. But as an image compression, the quality of the image can also be affected as the image data is compressed. JPEG compression process is generally divided into the following steps [4]:

- Color Space Conversion. As JPEG exclusively supports the color model in YUV, except for the RGB. Therefore, the RGB image data must be converted into YUV.
- DCT Conversion. DCT conversion is a process to convert the image signal in the frequency domain to separate the high and low frequency information, then compress the high-frequency part of the image to achieve the purpose of image data compression.
- Quantization. Integer is used in the encoding process, so the converted frequency must be quantified so as to convert it into an integer.
- Encoding. It is used to encode the DCT coefficient after quantization, to minimize its entropy.

IV. SYSTEM DESIGN

The video surveillance system of this paper adopts client/server model (C/S), and the design is based on two aspects, the remote monitoring and the client. The advantages of the C/S model are as follows: it can exert the processing capacity of client PC as many works can be done by the client to submit to the server; the C/S model has a good security performance, and shows cariousness at user interface, satisfying personalized needs of the users [5].

The remote monitoring implements the image capture and local playback via DirectShow technology, and compresses the captured images which can be transmitted through network. When the client receives the transmitted image data, firstly it will decompress and process; and displays on the screen at client, completing real-time monitoring. The system design refers to figure 2:

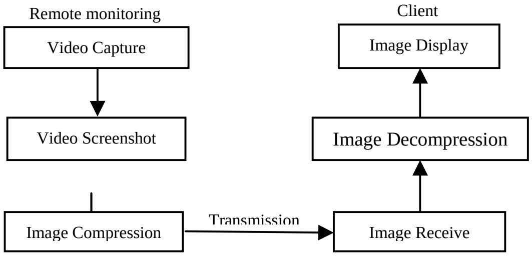


Figure 2. system design

- Video Capture and Playback: The DirectShow technology is adopted to implement real-time capture and local playback. Based on the advantage of COM technology, this module is very convenient for video local playback, with clear and stable image display.
- Video Screenshot: Through the Sample Grabber Filter of DirectShow Technology, the real-time screenshot can be done.
- Image Compression and Decompression: The images captured through DirectShow Technology are saved in BMP format, which is of heavy information and not good for transmission through network. Therefore, the compression is needed for transmission. At client, the compressed images should be decompressed and displayed.

V. SYSTEM IMPLEMENTATION

For the DirectShow application, the first step is the development environment configuration. And then to add the streams' header and link library file like strmbasd.lib, uuid.lib, winmm.lib, Quartz.lib. DirectShow application is a COM application. Before using the COM Function, we need to initialize the COM library by calling Co Initialize. And finally, we need to anti-initialization the COM library by calling Co initialize.

A. Implementation of Real-time Video Capture

This function is implemented via DirectShow Technology. The Filter link is shown in Figure 3:

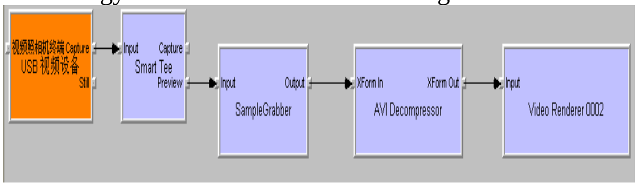


Figure 3. Filter Link Route

Connect each Filter in order. Then graph shown above is completed. And next, the application sends commands to the graph. Finally the entire video capture is implemented. Before capturing video, we must obtain the capture device of the system, and then establish the Video Capture Filter by the device identification. Based on the device identification, the corresponding Video Capture Filters can be built up. And then, the entire graph can be built up based on the Filter Link Route shown above. Finally, video capture and playback can be implemented.

1) Create Filter Graph Manager Module and video capture filter, and its core code is shown as follows:

```
    hr=CoCreateInstance(CLSID_FilterGraph,NULL,CLSC
TX_INPROC_SERVER,IID_IGraphBuider,(void**)&
m_pGB);
    hr=CoCreateInstance(CLSID_CaptureGraphBuilder2,N
ULL,CLSCTX_INPROC,IID_ICaptureGraphBuilder2,
(void**)&m_pCapture) ;
    m_pCapture->SetFiltergraph(m_pGB);
```

2) Through the enumeration of the system devices, the filters of video capture device can be obtained and the core code is as the follows:

```
    ICreateDevEnum *pCreateDevEnum ;
    IEnumMoniker *pEm;
    hr=CoCreateInstance(CLSID_SystemDeviceEnum,
NULL,CLSCTX_INPROC_SERVER,
IID_ICreateDevEnum, (void**)&pCreateDevEnum);
    hr=pCreateDevEnum->
    CreateClassEnumerator(
    CLSID_VideoInputDeviceCategory,&pEm,0);
    pEm->Reset();
    ULONG cFetched;
    IMoniker *pM;
    int index = 0 ;
    while( hr = pEm->Next(1,&pM,&cFetched),
hr == S_OK , index <= iDeviceID ){
    IPropertyBag *pBag;
    hr =
    pM>BindToStorage(0,0,IID_IPropertyBag,(void**)&pB
ag);}
    hr = m_pGB->AddFilter(m_pBF,L"Capture Filter");
```

B. *Implementation of Real-time Screenshot and Image Transmission*

1) The core code for video playback and real-time screenshot is as follows:

```
    hr = m_pGB->AddFilter(pGrabberFilter,
L"Sample Grabber");
    pGrabberFilter->
QueryInterface(IID_ISampleGrabber,(void**)
&pGrabber);
    m_pCapture->
RenderStream(&PIN_CATEGORY_PREVIEW,&MEDIAT
YPE_Video,m_pBF,pGrabberFilter, NULL);
    BITMAPINFO BitmapInfo;
    ZeroMemory(&BitmapInfo, sizeof(BitmapInfo));
    CopyMemory(&BitmapInfo.bmiHeader,
&(pVih->bmiHeader),sizeof(BITMAPINFOHEADER));
```

2) Compression, Transmission and Decompression of Images

As the image data from step B in BMP format, which is of large data, the client will have jitter problems to be transmitted over the current network conditions. Therefore, the image data needs to be compressed. For this system, the libjpeg is used to the BMP image compression.

Libjpeg is a Library totally programmed in C language, including the realization of the widely used JPEG decoding, encoding and other JPEG functions. When compressing the images, a variable in jpeg_compress_struct is needed to be set up, as this variable has stored the detailed information of all JPEG data, and it also stores the output details after the encoding. Then, invoking the jpeg_create_compress to initialize the ones to be encoded and decoded. And then, invoking the jpeg_stdio_dest to set the target data location of encoding. Through the settings of jpeg_compress_struct variables, the encoding parameter can be set. And finally, compressing the data of each line. At lase, the compressed data can be transmitted at the remote monitoring.

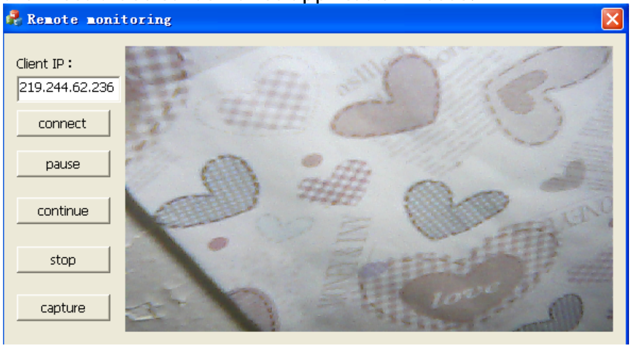
At client, after receiving the data from the remote monitoring, to ensure display quality, the image needs to be decompressed and then can be displayed. Firstly, creating the variable of jpeg_decompress_struct; secondly, invoking jpeg_create_compress to initialize the ones to be encoded and decoded; thirdly, using the jpeg_stdio_src to set the source date for decoding. We can get the JPEG header information through jpeg_read_header. We can set the decoding parameters by setting the jpeg_decompress_struct variable. At last, decoding the image data and displaying the decoded image at client port to achieve remote video surveillance.

The core code for image compression and decompression is as follows:

```
    while (jcs.next_scanline < jcs.image_height)
    {
    row_pointer[0] = & data[(jcs.image_height-
jcs.next_scanline-1) * (row_stride+nAdjust)];
     jpeg_write_scanlines(&jcs, row_pointer, 1);
    }
    while (cinfo.output_scanline < cinfo.output_height)
    {
    row_pointer[0]=&data[(cinfo.output_height-
cinfo.output_scanline-1)*(cinfo.image_width*
    cinfo.num_components+nAdjust)];
    jpeg_read_scanlines(&cinfo,row_pointer ,1);
    }
```

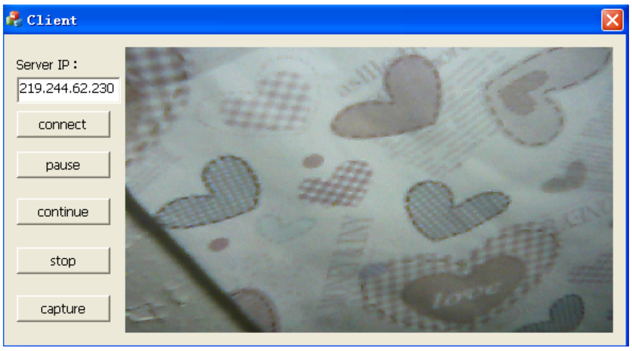
C. *System Testing*

In *the* network with a speed of 1M/S, the test shows that when the client can smoothly monitor the status of the remote monitoring, the packet loss rate of the system is about zero, *and* system delay and jitter problems are less. It can fulfill the purpose of video surveillance, basically satisfy the most video surveillance application needs.



Client IP : 219.244.62.236

Figure 4. Remote monitoring



Server IP : 219.244.62.230

Figure 5. Client

VI. CONCLUDING REMARKS

Depending on the DirectShow and Image Compression technology, this paper accomplishes the design and development of video surveillance between the remote monitoring and the client, implements the local video playback and remote surveillance functions. This system is of smooth transmission performance without real-time display jitter. Furthermore, it is of highly reusability and development, and can be applied to practical video surveillance areas.

REFERENCES

[1] Z.L. Zhang, Y.M. Liu, Z.L. Ding, “Remote Surveillance System, Based on Winsock and DirectShow”, Journal of Tianjin University of Science & Technology, 2007,22,48.

[2] X.X. Zheng and W. Wei, “Study on Video Capture and Transmission System Based on Direct Show”, Digital Technology and Application, 2011.

[3] Q.M. Lu, Guidance on DirectShow Development, Beijing Tsinghua University Press, 2011.

[4] Y. Hu, “Relation Analysis on JPEG and Image Compression Quality, Press Quality and Standardization”, Yan Yan and Qiaoliang, 2010,44

[5] Q.J. Zhang, “Design and Realizaton of H.264 Real-time Transmission System Based on RTP”, Enterprise Development Technology,2011,30.

Published by Atlantis Press, Paris, France.