International Journal of Advanced Smart Convergence Vol.12 No.3 25-31 (2023)
http://dx.doi.org/10.7236/IJASC.2023.12.3.25

IJASC 23-9-4

Design of Real-Time Video Play System Using Web Camera

Seung Ju Jang

Professor, Department of Computer Engineering, Dong-Eui University, Busan, 47340 Korea
sjjang@deu.ac.kr

Abstract

This paper designs a real-time video playback system using a web camera in the RTSP server. It designs a function to play the video data of the web camera in the client in real time using the web camera in the server and using the RTSP protocol. It consists of a server module function that produces real-time video information using a web camera and a client module function that plays video received from the server in real time. The experiment was conducted by establishing an environment for designing a real-time video playback system using a web camera. As a result of the experiment, it was confirmed that real-time video playback from the server's web camera worked well.

Key words: *web camera, RTSP, video playback, real-time, server module, client module*

1. Introduction

Recently, due to the spread of the Internet and smart phones, demand for a system for reproducing an image captured at a remote location in real time is increasing. Accordingly, a real-time video playback system using a web camera in a remote location has become an important technology. A system that captures a video using a web camera in a remote location and plays the video in real time at another location can be usefully used in various fields. In the field of security-related technology, it is important to build a security system using CCTV cameras. At this time, if security monitoring can be performed in real time using a remote web camera, a more efficient security system can be built.

In this paper, we propose a real-time video playback system using a remote web camera. Through this, it is possible to implement a real-time video playback system using a remote web camera in various fields such as security systems, travel photography, and education. In this paper, we propose a way to capture and transmit a web camera video in a remote location, and to implement a server and client module. Through this, the technical aspects of the real-time video playback system using a remote web camera are covered, and the system operation process is shown through the actual system construction. This paper makes it possible to build a real-time video streaming system at low cost using commonly used web cameras.

Manuscript Received: June. 12, 2023 / Revised: June. 19, 2023 / Accepted: June. 25, 2023
Corresponding Author: sjjang@deu.ac.kr
Tel: +82-51-890-1710, Fax: +82-51-890-2629
Professor, Dept. of Computer Engineering, Dongeui University, Korea

This paper proposes the following design method. First, we select and design the RTSP communication protocol to transmit the video taken from the remote web camera. At this time, a module for efficient data transmission is designed. Next, we implement the client module and server module. The client module provides an interface through which users can play and control video in real time, and the server module receives video data from a remote web camera and transmits it to the client module in real time using the RTSP protocol. Experiments were conducted on the design contents proposed in this paper. As a result of the experiment, it was confirmed that it worked normally.

Chapter 2 of the paper explains related research, Chapter 3 is the contents of the system design proposed in the paper, Chapter 4 is experiments, and Chapter 5 is conclusions.

2. Related research

Recently, various protocols for moving video streaming have been developed, and representative examples are a Real-time Transport Protocol (RTP) and a Real-Time Streaming Protocol (RTSP). RTP guarantees the stability of data transmitted in real time, and RTSP is a protocol for controlling video streaming using RTP. In addition, WebRTC (Web Real-Time Communication) technology has been widely used recently, which is a technology that transmits video and audio in real time using web technology in browser [1, 2, 3].

Recently, along with the development of Internet of Things (IoT) technology, interest in a real-time video playback system using a remote camera has increased. Accordingly, technical development for stably transmitting an image of a remote camera is being actively conducted. Video data compression technologies such as H.264/AVC and H.265/HEVC have been developed as a technology for transmitting images from remote cameras, and more efficient video streaming is possible using these technologies. With the development of IoT technology, interest in a real-time video playback system using a remote camera is increasing.

In addition, with the development of cloud technology, a cloud-based remote camera system that processes image data on the server side and receives and reproduces the data on the client side is also being developed. Since such a cloud-based system does not require high-end equipment or infrastructure for processing image data, it has the advantage of being able to process image data at low cost. In addition, with the development of cloud technology, cloud-based remote camera systems that process image data on the server side and receive and play data on the client side are also being developed [4, 5, 6, 7].

In addition, with the recent commercialization of 5G technology, a technology for transmitting image data in real time using a network with faster speed and stability is also being developed. As these technologies develop, a real-time video playback system using a remote camera is further expected to develop [8, 9, 10].

Recently, demand for services to provide audio and video data in real time in various environments at low cost is increasing. A technology that satisfies these requirements is system development using the RTSP protocol. As a system application field, it is used in various fields such as provision of disaster information system or provision of traffic information [11, 12, 13, 14, 15, 16].

3. Real-time video playback system design

The system configuration for designing a real-time video playback system using a web camera is as follows. The hardware configuration for designing a real-time video playback system using a web camera is as follows.

Web camera: We use a web camera that can record the desired environment. (IPC HD1 WEBCAM)
Computer: We use a system with a Windows 11 operating system capable of processing videos.

A real-time video playback system uses a web camera generally used. It uses the computer using the Windows 11 operating system capable of processing video. Windows 11 uses libraries, SDKs, and GPU acceleration features to speed up processing. In general, it uses a computer with an i5 or better CPU, 32GB or more RAM, and an Nvidia GeForce. In addition, SSDs are used to speed up data processing. Because SSDs have faster data processing speeds than conventional hard disks, it can increase recognition and processing speeds. If you select a web camera and a computer considering these hardware components, you can build a system more accurately and quickly.

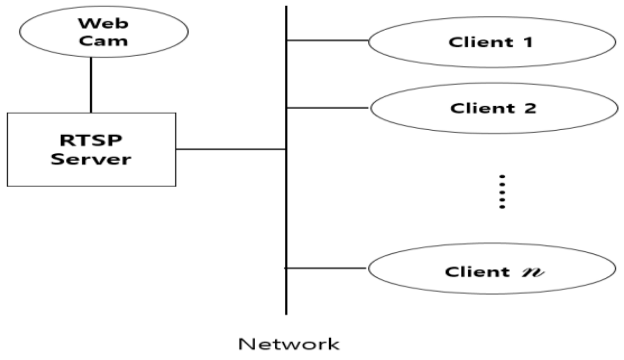


Figure 1. System architecture

Figure 1 above shows the structure of a real-time video playback system using a web camera. A real-time video playback system using a web camera can build a real-time streaming system at low cost. First of all, RTSP (Real-Time Streaming Protocol) server system should be built using a web camera. The RTSP server system construction environment for the web camera is as follows. The RTSP server system uses the simple-RTSP server module. It builds an environment that can transmit web camera videos to clients using the simple-RTSP server module.

The client system is built using the Python programming language. The client system makes it possible to build a system so that you can easily view the real-time video streaming system sent from the RTSP server anywhere. By constructing the system in this way, a real-time data streaming system can be built and used at low cost. A real-time video playback system using a web camera is required in various fields. In the conventional method, there are many limitations in use because it uses sometimes an expensive camera. Therefore, the low-cost video playback system is expected to be applied and used in many fields.

Functions operated in the RTSP server system are as follows. The real-time video stream captured by the web camera is transmitted to the client using the RTSP protocol. The user can connect to the RTSP server using the client module and play the video captured by the web camera in real time.

A server can handle connections from multiple clients at the same time. The client can connect to the server and play the video using not only the RTSP client but also other RTSP clients such as VLC. Through these functions, it is intended to implement a real-time video playback system at a low cost and to be utilized in various fields.

The client system can make multiple accesses to the web camera server system anywhere. The client system connects to the RTSP server system and provides a function to play real-time video. The client system plays a role of receiving and outputting real-time video streaming from the RTSP server system.

The client system can access the real-time video stream generated by the web camera RTSP server system. This stream is transmitted to the client using the Real Time Streaming Protocol (RTSP) protocol. The client's Python module plays the role of receiving real-time video streaming from the RTSP server system, decoding it, and outputting it to the user.

The client Python module connects to the RTSP server system and provides a function to play real-time video. In addition, the client system is capable of multiple access. It allows multiple users to access simultaneously and watch video streaming together with other users. These functions can be usefully utilized in fields such as distance education, meetings, and lectures.

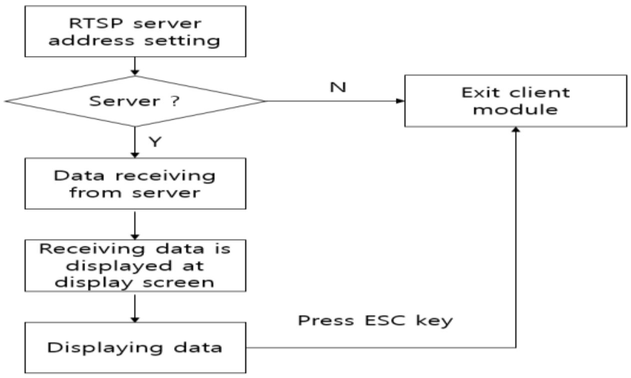

RTSP server address setting
Server ?
N
Exit client module
Y
Data receiving from server
Receiving data is displayed at display screen
Press ESC key
Displaying data

Figure 2. System operation process

Figure 2 shows the operating process of the video playback system using the RTSP server. First, it sets the RTSP server address. It searches the RTSP server address, and if the RTSP server address is found, real-time video data is received from the server. The client module outputs the video data received from the server to the client display window. The real-time video data received from the server is continuously displayed on the client display window. If the user wants to stop displaying, he presses the ESC key. When the ESC key is pressed, the client module is terminated.The client module outputs real-time video data received from the server. The functions performed by the client module are shown in Figure 3.

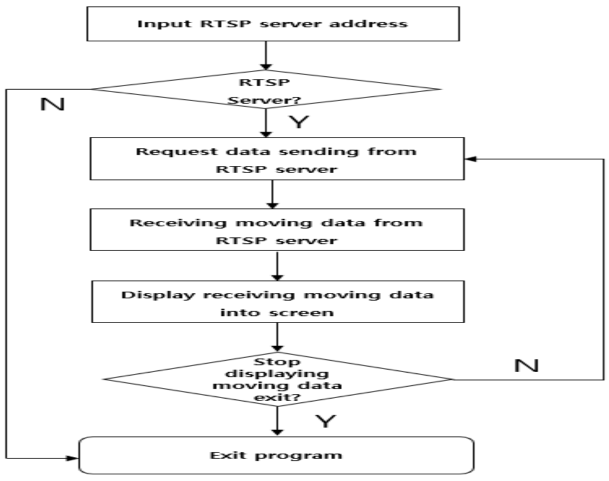

Input RTSP server address
RTSP Server?
N
Y
Request data sending from RTSP server
Receiving moving data from RTSP server
Display receiving moving data into screen
Stop displaying moving data exit?
N
Y
Exit program

Figure 3. Client module operation process

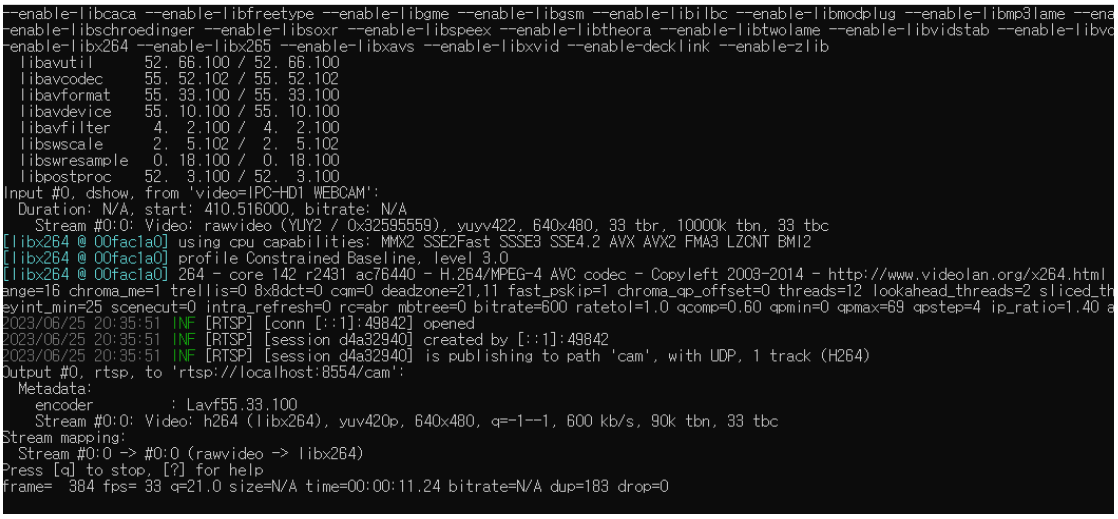
Figure 3 shows the client module operation process. If the user wants to connect to the server from the client module, he must enter the RTSP server address. If there is no RTSP server, the program terminates. If there is a desired RTSP server, the server is requested to transmit video data. When data is normally transmitted from the server to the client, the client module receives the data. The client module outputs the received data to the output window. If he wants continuous real-time video data output from the client module, the previous process is repeated. Otherwise, if real-time video playback is to be terminated in the client module, the program is terminated.

4. Experiment

In this paper, experiments were conducted on the design of a real-time video playback system using a web camera. The system environment for conducting the experiment is as follows.

Web camera system: A commonly used web camera (IPC HD1 WEBCAM)
RTSP server system: Server system using simple-RTSP server module
Client module: Playback system module using Python program

An actual system was built and tested for real-time video playback using a web camera. To conduct the experiment, it set up the RTSP server environment that can transmit data from the web camera to the client. The RTSP server system environment is built using the simple-RTSP module. The server system environment built with the simple-RTSP module is shown in Figure 4.



```
--enable-libcaca --enable-libfreetype --enable-libgme --enable-libgsm --enable-libilbc --enable-libmodplug --enable-libmp3lame --ena
-enable-libschroedinger --enable-libsoxr --enable-libspeex --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvo
-enable-libx264 --enable-libx265 --enable-libxavs --enable-libxvid --enable-decklink --enable-zlib
  libavutil      52. 66.100 / 52. 66.100
  libavcodec     55. 52.102 / 55. 52.102
  libavformat    55. 33.100 / 55. 33.100
  libavdevice    55. 10.100 / 55. 10.100
  libavfilter     4.  2.100 /  4.  2.100
  libswscale      2.  5.102 /  2.  5.102
  libswresample   0. 18.100 /  0. 18.100
  libpostproc    52.  3.100 / 52.  3.100
Input #0, dshow, from 'video=IPC-HD1 WEBCAM':
  Duration: N/A, start: 410.516000, bitrate: N/A
    Stream #0:0: Video: rawvideo (YUY2 / 0x32595559), yuyv422, 640x480, 33 tbr, 10000k tbn, 33 tbc
[libx264 @ 00fac1a0] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2 AVX AVX2 FMA3 LZCNT BMI2
[libx264 @ 00fac1a0] profile Constrained Baseline, level 3.0
[libx264 @ 00fac1a0] 264 - core 142 r2431 ac76440 - H.264/MPEG-4 AVC codec - Copyleft 2003-2014 - http://www.videolan.org/x264.html
ange=16 chroma_me=1 trellis=0 8x8dct=0 cqm=0 deadzone=21,11 fast_pskip=1 chroma_qp_offset=0 threads=12 lookahead_threads=2 sliced_th
eyint_min=25 scenecut=0 intra_refresh=0 rc=abr mbtree=0 bitrate=600 ratetol=1.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 a
2023/06/25 20:35:51 INF [RTSP] [conn [::1]:49842] opened
2023/06/25 20:35:51 INF [RTSP] [session d4a32940] created by [::1]:49842
2023/06/25 20:35:51 INF [RTSP] [session d4a32940] is publishing to path 'cam', with UDP, 1 track (H264)
Output #0, rtsp, to 'rtsp://localhost:8554/cam':
  Metadata:
    encoder         : Lavf55.33.100
    Stream #0:0: Video: h264 (libx264), yuv420p, 640x480, q=-1--1, 600 kb/s, 90k tbn, 33 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (rawvideo -> libx264)
Press [q] to stop, [?] for help
frame=  384 fps= 33 q=21.0 size=N/A time=00:00:11.24 bitrate=N/A dup=183 drop=0
```

Figure 4. Streaming server system settings

Figure 4 is a screen where the RTSP server system construction. It was successfully performed to provide the video captured from the web camera to the client in real time. If the RTSP server is built normally, the information that is normally operated is output as shown in Figure 4. In this way, the RTSP server must be properly established to enable access when a client user wants to play a real-time video. After the RTSP server environment is established, the server's web camera is ready to transmit real-time video to the client system. The client module provides a function to play video transmitted from the RTSP server. The Python programming language is used to implement the video playback function in the client module. The client module receiving video data from the server outputs real-time video data on the client display window. Figure 5 is a screen displaying the video data received from the RTSP server on the display window of the client module.

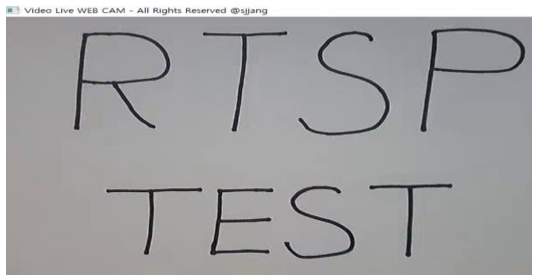


Figure 5. Real-time video receiving screen from RTSP server

Figure 5 is an output screen after receiving and outputting real-time video data after connecting to the RTSP server from the client module using the Python program. First, the client module tries to connect to the RTSP server address. When a normal connection is established by attempting to connect to the RTSP server, real-time video data is received from the web camera in the RTSP server. The video data received in real time is output to the client output screen. In Figure 5, you can see that the RTSP server's web camera data, "RTSP TEST", is received well and displayed on the client display window. As shown in the experiment in Figure 5, it was confirmed that the system proposed in this paper operates normally.

5. Conclusion

In this paper, we design a real-time video playback system using a web camera. In this paper, we design video receiving from RTSP server web camera, real-time data transmission to client, and implementation method of server and client module. Through this, the technical contents of the real-time video playback system using the RTSP server web camera are designed.

This paper presents the following design method. First, an appropriate communication protocol such as simple-RTSP is selected to transmit the video taken from the RTSP server web camera, and an RTSP server environment is built using the simple-RTSP. The RTSP server system receives video data from the web camera and transmits the video data to the client module in real time using the RTSP protocol. Next, we implement the client module. The client module provides an interface through which users can play and control videos in real time. A Python program is used to design the client module. It connects to the RTSP server through the Python program and provides the function of receiving and outputting real-time video data. For the real-time video playback system design using a web camera proposed in this paper, the system was implemented and experiments were conducted. As a result of the experiment, it was confirmed that it worked normally.

References

[1] M. I. M. Firmansyah1, N. Suharto, Y. H. Prasetyo, “RTSP and HTTP Protocol Analysis for Streaming Services on Manet Networks in State Polytechnic of Malang,” *Journal of Telecommunication Network*, Vol. 12, No.3 2022. 9. DOI: https://doi.org/10.33795/jartel.v12i3.473

[2] D. V. Nguyen, H.T.T. Tran, and T.C. Thang, “A Client-based Adaptation Framework for 360-Degree Video Streaming,”

Journal of Visual Communication and Image Representation, Vol. 59, pp.231-243, 2019. DOI:https://doi.org/10.1016/j.jvcir.2019.01.012

[3] M. Han, S. H. Lee, and S. Ok, "A Real-Time Architecture of 360-Degree Panoramic Video Streaming System," 2nd IEEE International Conference on Knowledge Innovation and Invention, pp.477-480, 2019. DOI: https://doi.org/10.1109/ICKII46306.2019.9042658

[4] S. G. IVÁN, R. G. ALEXANDRA, M. G. JEZABEL AND C. G. PINO, "Implementation and Analysis of Real-Time Streaming Protocols," *Sensors*, PP.1-17, 2017. DOI: HTTPS://DOI.ORG/10.3390/s17040846

[5] C P. YOON, C. G. HWANG, D. J. KIM, "A STUDY ON CDN(CONTENTS DELIVERY NETWORK) SYSTEM USING RTSP(REAL TIME STREAMING PROTOCOL) STREAMING SERVER," KOREA INFORMATION AND COMMUNICATION SOCIETY 2020 FALL PROCESSING, PP.277-279. 2020.

[6] D. h. Youa, "Software-defined Radio (SDR): An Approach to Real-Time Video Data Transceiver Implementation," *Journal of the Society of Broadcasting Engineering*, vol. 28, no. 1, 2023.1. DOI:https://doi.org/10.5909/JBE.2023.28.1.149

[7] R. Delgado, S.H. Lee, B.D. Ahn and B. W. Cho, "Development of a Simultaneous Audio Broadcasting System Based on Embedded Hardware using RTSP," *Journal of Korean Institute of Intelligent Systems*, Vol. 28, No. 4, pp. 362-368, August 2018, DOI:http://dx.doi.org/10.5391/JKIIS.2018.28.4.362

[8] Liang Jianbing and Chen Shuhui, "The Design and Implementation of RTSP/RTP Multimedia Traffic Identification Algorithm," Journal of Physics: Conference Series, Conf. Series 1168 (2019), DOI: https://doi.org/10.1088/1742-6596/1168/5/052033

[9] C. Y. Woo , N. H. Park , J. Y. Beak, Y. J. Jung, M. J. Kim, "Person Selectable Transmission System in Real-Time Video Conference," Korea Information Processing Society Proceedings, pp. 184-187(4pages), vol. 27, no. 1, 2020.

[10] A. D. GOVINDASAMY, "GETTING RTSP TO WORK NATIVELY IN THE BROWSER*," Asian Journal of Advances in Research*, Volume 5, Issue 1, Page 860-867, 2022.

[11] P. Iyyanar, M. Arunachalam, A. M. Patil , N. Uke, J. D. Lal, V. R Sonawane, R. Rajagopal, "A Real-Time 3D Video Streaming System Using SRTP AND RTSP Protocol," *IJCSNS International Journal of Computer Science and Network Security*, Vol. 22 No. 6, Pp.620-626, June 2022. DOI:https://doi.org/10.22937/IJCSNS.2022.22.6.76

[12] Samy Y. Doo, Molina O. Odja, Andri P. Chandra, Almido H. Ginting, "EVALUASI PROTOKOL RTSP & HTTP DALAM TRANSMISI VIDEO PADA WLAN," *JURNAL ILMIAH FLASH*, Vol. 9 No. 1, pp.31-36, Jun 2023. DOI: https://doi.org/10.32511/flash.v9i1.1070

[13] Xinkan Zhang, Fufeng Chu, "Multimedia Real-Time Transmission Protocol and Its Application in Video Transmission System," *Computational Intelligence Neurosci*, 2022 May, DOI: https://doi.org/10.1155/2022/8654756

[14] L. Nuñez and R. M. Toasa, "Performance evaluation of RTMP, RTSP and HLS protocols for IPTV in mobile networks," *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, Seville, Spain, pp. 1-5, 2020. DOI: 10.23919/CISTI49556.2020.9140848.

[15] J. -C. Chiu, H. -Y. Tseng and Z. -Y. Lee, "Design of Multidimension-media Streaming Protocol Based on RTSP," 2020 International Computer Symposium (ICS), Tainan, Taiwan, pp. 341-347, 2020. DOI: 10.1109/ICS51289.2020.00074.

[16] HIBA K. ABDULAZEEZ1, NASSER N. KHAMISS2, SAIF M. AHMED, "OPTIMAL ADAPTATION OF INTERNET VIDEO STREAMING," Journal of Engineering Science and Technology, Vol. 17, No. 2 (2022) pp.1010 – 1027, 2022.