Raspberry Pi Based Live Streaming For Amphibious Surveillance Robot

Devashish Panchal¹, Vaibhav Paradkar², Shashank Parkar³, Akshar Parmar⁴, Dr. Bhavin Shah⁵

¹²³⁴⁵(Department of Electronics and Telecommunication, Atharva College of Engineering, India)

Abstract: Surveillance has a paramount importance in today’s fast changing world. Real-time surveillance is an indispensable component in any environment where there is an immense need of security for both personal and commercial resources. Technology today is used in various ways in order to provide us such surveillance. This feature is extremely useful for real-time surveillance and monitoring. The optimum amount of sensing devices must be integrated into the system for efficient real-time surveillance. There is a rise in demand for a compact and robust surveillance system that can function both on land and water, which requires a proper real-time surveillance set-up, for effective supervision of surrounding environments. In this model live streaming is achieved by using netcat streamer. Raspberry Pi 3 is used for live streaming. The client-server model is TCP/IP. This real-time streaming model is a part of a semi-autonomous land and water surveillance system.

Keywords-Surveillance, Net-cat, Mplayer, G-Streamer, Raspberry Pi, Local network

I Introduction

There is a growing need for security and surveillance models which are highly effective. Video surveillance has become one of the major requirements recently. This application has wide-ranging purposes like traffic monitoring, understanding human activity. Different kinds of cameras are used for surveillance like fixed cameras, and pan and tilt cameras. These kinds of cameras are generally utilized for indoor security. For indoor security system, multiple cameras are mounted on a wall with myriad angels for tracking objects. These types of systems require a computer or a laptop for monitoring. Nowadays, most of the systems use a mobile robot with a camera for surveillance. These types of robots are more flexible than the fixed cameras. Most of these surveillance robots are wheeled robots. The wheel-based robots are more suitable for a terrestrial platform. Due to rapid strides wireless technology, we can stream the real-time video data with methods such as netcat-mplayer, MJPEG, etc. and it can be seen on any device such as a laptop or a computer.

With the continuous development of wireless communication and internet, security systems are rapidly improving. This has resulted in the rise of efficient surveillance systems. Two such types of systems are available which are Analog and IP based video Surveillance systems. The analog systems are comparatively less expensive and simple to operate than the IP-based systems. However, it has some limitations, such as for covering a greater area we need to deploy more number of cameras and once the cameras are set up at particular locations it is very tedious as well as complicated to shift them to another location since the system has wired connectivity.

There are various methods to implement a video surveillance system for autonomous remotely operated robotic systems. In this proposed design we have used Raspberry Pi, since it is a very powerful and effective platform for implementing robotic systems. The IP-based system can be easily implemented using a Raspberry Pi 3. It is a highly effective and expected to run smoothly even when high resource software is used. Ethernet, audio and video processing, bigger sized RAM and a larger amount of storage space, makes it an effective mini-computer. It consists of a complete operating system loaded on an SD card, audio out, HDMI and RCA video output and an Ethernet port.

II Challenges Faced

The camera module used must be compatible with the Raspberry pi module. For streaming the video using raspberry camera module certain methods are employed. However, some problems are encountered while making this design. The challenges faced by this conceptual model are as follows:

1. At the listener end, video is played only when the cache memory is completely filled. This proves to be a vexing problem if the system is expected to monitor and transmit real-time data.

2. If read rate is greater than the receiving rate of cache memory then the video starts to lag which means a delay of 4-5 seconds takes place in streaming. Such a delay might seem small, but it makes a huge amount of
difference for real-time surveillance systems. A delay of even a fraction of a second can prove costly for monitoring models. Hence this delay or lag must be reduced and made as minimal as possible.

3. Unless the number of frames per second is specified, mplayer will face difficulty playing an H264 video stream. If the number of frames of a receiver is less than that of the sender machine, there is a lag of 3-4 seconds. This delay must be eliminated.

### III Proposed System

In this project, video capturing module is 5mp raspberry pi camera, which gives HD image/video quality. There are two protocols are which can be used for streaming process. TCP/IP has advantages over UDP such as transmission of data in user specified order and guarantee of intactness of data at the receiver end.\(^9\)Since the data rate of video is high it uses extra bandwidth hence we need video compression it is sent.\(^9\)When video streaming is initiated using TCP/IP various processes are takes place,\(^9\) Hence in this project, the TCP/IP based real-time video streaming method is used with netcat (also known as ‘nc’ or ‘Swiss Army knife’). The idea behind using this methodology is to combat packet delays caused by TCP/IP that are essentially interpreted as errors by the streaming application. ‘Real-time video streaming experiments has shown an improved result with TCP/IP-based streaming mechanisms.\(^10\) Netcat is a tool which is used for reading and writing connections. Netcat is structured in such a way that it creates various types of connection which can be used directly or easily driven by other programs and scripts. In order to establish a network connection between sender and receiver, netcat has to be installed in both client (sender) and server (receiver) machines. While receiving data, netcat simply detects an incoming connection and does not attempt to initiate any kind of authentication procedure, so it is essential to take care whenever netcat is used to encode video stream.\(^12\)

On the server (receiver) end open media player software is used known as ‘mplayer’. MPlayer supports seems including Microsoft Windows, Linux, macOS, and UNIX. MPlayer supports a wide range of media formats, all format which is supported by FFmpeg libraries, and can save all streamed data to a specified file in local machine.

Another program, called MEncoder, can take an input stream, file or a sequence of picture files, and encode it into several different output formats, optionally applying various transforms while processing them. Sufficient amount of time must be provided to fill the cache memory before a streamed video is played.

Standard syntax used in client and server machines is as follows:

**Client (sender end):** raspivid -t 999999 -o -w 512 -h 512 -fps 15 | nc [IP address of receiver machine]

[port number of receiver machine]

We are using VNC viewer to control raspberry pi, which works by entering IP address of raspberry pi. With the help of VNC viewer we can use control raspberry pi from any corner.\(^13\) IP address of receiver machine and port number of receiver machine must be specified in the command. ‘t’ indicates the time duration for the video to be captured. Here ‘w’ indicates width of the display window and ‘h’, indicates height. Dimensions of the display window can be adjusted as per requirements.\(^14\) Fps indicates the frames per second. This completes the sender part of the process.

**Server (receiver end):** [netcat executable file path] -L -p [port number of receiver machine] | [mplayer executable path] -vo direct3d -fps 24 -cache 512 –

The netcat file path must be copied in the command. Port number of the receiver machine and mplayer executable path is copied in the command line.\(^15\) Similar to the sender end we need to specify the number of frames. This number must be greater than the sender frames.\(^16\)

Raspivid: Raspivid is the command line tool for capturing the video using the raspberry Pi camera module. When video utility ‘raspivid’ is initiated, host machine sends packets to the listener machine. Command line is as follows:

raspivid -o vid.h264

For specifying the video length pass the –t flag in the raspivid command line. The time in this flag is given in milliseconds.

Raspistill: It is the command line used for capturing still images using the raspberry Pi camera module. Command line is as follows:

raspistill -o cam.jpg

In the following figure we have clicked the photo using the raspberry camera using the command raspistill. The motion capture is done using the above mentioned Pi commands.
The images captured have resolutions of 2592 x 1944 pixels. –vf or –hf flags can be used for vertical or horizontal flipping of the image or the video. The resolution of this camera is 5 megapixels.

IV  Representative Design of the Proposed System

Fig 3: Block diagram of live video streaming using netcat and mplayer

The raspberry pi camera module is used as the data sensor. It can be easily programmed as per user requirements. The camera in this design is used for real time video streaming as well as for motion capture. Raspberry Pi 3 is the host machine used in this design; it collects the data from the camera module to transmit it to the listening device via the TCP protocol. The listener machine can be as per the user requirement. For capturing still image, appropriate command must be sent to the host machine and it will provide the output.
mplyer terminal can be used for viewing the live streaming or the still image. Due to the implementation of the delay has been minimized.[18]

V Conclusion

In this paper, we have seen how the proposed design effectively tackles the problem of delay in the streaming process and improves the overall system performance. This design with additional modifications can be efficiently used for marine surveillance purposes. Continuous monitoring can be implemented based on this design. For the amphibious robot, the streaming should continue whenever the device moves from land to water or vice versa. This is easily possible due to this design. For higher, complex applications this design can be upgraded as per the requirements such as real time attendance cam, in which algorithms can be used to detect human faces and identify them.[19][20]

References

Journal Papers:


International Conference on Innovative and Advanced Technologies in Engineering (March-2018) 50