Low Latency Streaming System – Design Considerations

Bhavani GK
Ittiam Systems (P) Ltd.
Consulate 1,
Richmond road
Bangalore, India
Tel: +91-80-66601020
bhavani.gk@ittiam.com

Shishir Birmiwal
Ittiam Systems (P) Ltd.
Consulate -1
Richmond road,
Bangalore, India
Tel: +91-80-66601076
shishir.birmiwal@ittiam.com

Vikal Kumar Jain
Ittiam Systems (P) Ltd.
Consulate 1,
Richmond road,
Bangalore, India
Tel: +91-80-66601147
vikalkumar.jain@ittiam.com

Abstract: Media Streaming is a key technology that is enabling anytime/anywhere access to multimedia content. The primary challenge of media streaming is to enable rich multimedia experience in the presence of random variations due to packet transmission delays and losses over communication channels. Live interactive applications such as Video telephony, Video Surveillance (involving responsive control operations) and Internet gaming additionally impose the requirement of a low end-to-end delay. The latency requirements range from 100-150ms in video conferencing applications to less than 30ms in Responsive Video Surveillance systems.

This paper focuses on the considerations in designing a low latency streaming system. It considers the factors that introduce latency at various stages in a typical streaming system. Further it examines the typical methods used to reduce end-to-end latency in a streaming system and the impact of each of these methods on the other parameters that define the performance of a streaming system. It examines the requirements and priorities of typical classes of interactive streaming applications and suggests trade-offs that can be considered to minimize the latency while achieving acceptable levels of performance with respect to the remaining parameters. We conclude with indicative values of latency achievable for various combinations of algorithms.

Index Terms: Streaming, Latency, constant bit rate (CBR), Best Effort Networks, Network Jitter, Slice Encoding, Arbitrary Slice Ordering (ASO), MPEG-2, MPEG4, H.264, IPTV, Video Surveillance

I. INTRODUCTION

With the explosive growth of multimedia applications on various IP networks, there is an increasing demand to enable efficient and real-time delivery of multimedia content over IP. Media streaming is a key technology that holds the potential of realizing real-time interactive multimedia applications over IP networks. Typical classes of interactive multimedia applications enabled by streaming are

- Media content distribution applications such as video-on-demand, In-flight and In-car entertainment systems
- IPTV, Live webcasts over the internet
- Distributed interactive applications such as virtual classrooms for distance learning, internet gaming
- Video Conferencing
- Video Surveillance in Commercial establishments such as shopping malls, banks, hospitals, airports as well as in homes

Each of the above applications enables a different level of interactivity for the users. Content distribution systems allow the user to select the content, carry out VCR like operations (Pause, Fast Forward, Fast Rewind). Video conferencing and virtual classrooms are continuously interactive. Video surveillance also requires intermittent interactivity of the servers with the user - for instance, in order to command a PTZ operation for a camera. Ground navigation systems for unmanned aerial surveillance vehicles also require the user to constantly interact with navigation controls based on the streamed video. Typical latency requirements for the examples mentioned above are tabulated below.

Application	End-to-end Latency Requirements
Content Distribution systems - Content selection	5 – 10 s
Content Distribution systems - VCR like operations	1 - 2 s
Video Conferencing	150 - 200 ms
Video Surveillance	150 - 200 ms
Ground navigation for unmanned aerial surveillance	60 - 100 ms

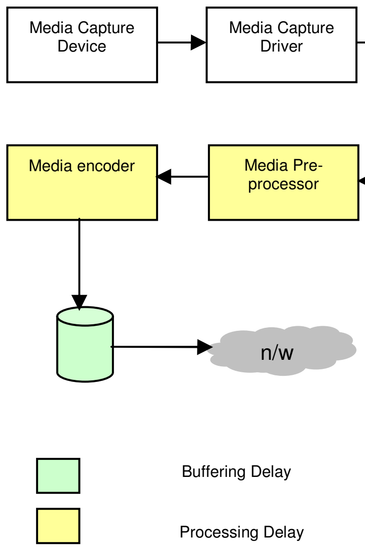
While a low latency is a requirement for all of the above, each class of applications has a larger set of performance parameters and constraints introduced by their respective operational environments. For Media content distribution systems and video-on-demand applications, media quality is of paramount importance. Scalability is a key requirement for any application that operates for a geographically spread out audience such as in virtual classrooms. Error robustness and the capability to degrade gracefully in the presence of packet losses, bit errors and variable packet arrival times is another requirement for applications such as webcasts and video conferencing that stream on heterogeneous best effort networks like the internet. Applications also have to operate within constraints such as a limit on the encoding or decoding complexities, available bandwidth for media transmission, limited or no control on the server or the clients.

Meeting the latency requirement of the application thus translates into making a choice of the right strategies at various stages of streaming given the remaining performance parameters and constraints of the system. In this paper we study the sources of latency in a typical streaming system and the common methods used to reduce the latency at different stages in a streaming system. We examine trade-offs that each of the methods brings in considering the performance parameters of typical classes of applications.

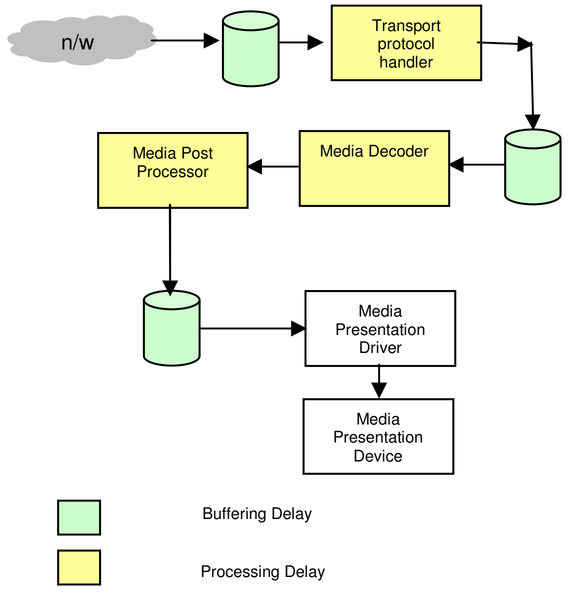
II. Causes of Latency

End-to-end latency in media streaming systems can be attributed to

- Multiple stages of buffering
- Processing delays
- Network transmission delays
- Buffering delays at Media capture and presentation device drivers
- Latency of scheduling each of these activities in a multi-threaded implementation of a streaming server ora client



Server Pipeline


n/w
Transport protocol handler
Media Decoder
Media Post Processor
Media Presentation Driver
Media Presentation Device
Buffering Delay
Processing Delay

Client Pipeline

The diagrams depict various delays in a typical server and client processing pipeline.

Buffering is introduced at multiple stages in a streaming system in order to overcome variations introduced by different components of the system. Video encoders spread out the impact of the deficit or excess of bit consumption due to sudden variation in the complexity of the input video over several frames. This require live media servers to start transmission of data only after a few frames are encoded to prevent starving the channel of data or overflowing the data buffers. It also requires the decoders being served over constant bitrate channels to buffer up several frames before starting to decode to prevent data starvation or buffer overflows during the decoding session. Buffering is also introduced to absorb the random variations in packet transmission times over multi-path networks.

Processing delays include execution time of media processing algorithms such as compression, decompression, any pre-compression video enhancements, post-decompression enhancements.

Scheduling variations can introduce buffering delays at the media capture and presentation drivers. In addition, delays can be introduced by media device due to any processing and video scanning delays.

Scheduling latencies in multi-threaded systems can be especially pronounced in RTOS based systems where pre-emption is the primary means of scheduling.

In the coming sections, we examine techniques that can be used to reduce the latencies listed above. Each technique targets one or more of the above causes. We examine the affects of each technique on the parameters such as scalability, error robustness of the system and complexity.

III. Encoder/Decoder Buffer Reduction

Video is inherently a variable complexity signal. If video is compressed using a constant number of bits per unit of video (such as a frame), the quality will be function of the complexity of video. If it is compressed to maintain a constant quality, the rate of generation of bits will vary as a function of the complexity of video. These can be regarded as special cases of the behaviour of an encoder that chooses to distribute the impact of the variations in the video complexity over a given duration of time (the time being 0 in the first case and infinite in the latter).

In order to transmit video on a constant bitrate channel, this duration of time T needs to be finite. Real-time streaming over a CBR channel can be achieved by maintaining a buffer to hold T s worth of bits on the transmission and reception side. Longer the duration of time, higher is the latency, lower is the variation of quality for variation in the complexity of the input video. Typically the buffer for video conferencing like applications, T will be of the order of less than a couple of frame intervals. In content distribution applications such as live webcasts, T can be a few seconds in duration.

Applications such as live webcasts are required to provide random access to enable users to join on-going sessions. Users will need random access into streams in order to switch channels while using an IPTV service. Video conferencing users need random access in order to join an ongoing conference. Video compression involves predictive coding which requires a decoder to have access to frames decoded in the past to decode the current frame. Thus providing random access requires periodic or on-request transmission of independently encoded frames, often referred to as key frames or Intra coded frames. In IPTV application in order to minimize channel switching times, the interval between two key frames needs to be typically lesser than half a second.

Intra frames usually consume higher number of bits than the predictive coded frames (Inter Frames) owing to lower compression. Periodic insertion of Intra frames introduces a need for buffering in order to absorb the difference between the sizes of intra and inter frames. Lower the period between successive intra frames, lower is the seek interval for a user, higher is the buffering required. In applications such as IPTV, although the latency is persists throughout the session, it is perceptible to the user only as an initial latency before the client begins to play the received media. In applications such as video conferencing, the latency is perceptible throughout the session due to the two way interactive nature of the application. On-request insertion of intra frames is a feasible option for video conferencing where the number of users is limited as against the scenario for IPTV or live webcasts. Hence periodic insertion of intra frames is recommended for IPTV where as on-request intra frame insertion is preferable in video conferencing applications.

The bitrates required to encode video at entertainment quality or large resolutions is high. Each frame of video typically spans several MTU (maximum transfer unit) size on most networks. The decoder side requires to collect all the packets belonging to a given access unit (independently decodable bitstream entity) before it begins to decode. In order to minimize this buffering, the access unit size needs to be reduced. H.263, MPEG-2, MPEG-4 and H.264 allow access units at sub-frame levels called slices. Each slice can be decoded independently of the other slices of the frame. Encoding a frame as a set of independent slices rather than single frame leads to a lower compression and hence a lower video quality at a given bitrate. Slice encoding mode however has an additional advantage that any loss resulting from bit errors or packet losses are limited to the slice in which the errors occur. The remaining part of the frame can still be reconstructed accurately. This is also conducive to the use of any error concealment algorithms on the client. Slice encoding mode can be resorted to if the latency is required to be less than a frame interval such as in a class surveillance systems where the data obtained from remote surveillance is used as an input for related control systems.

Another sub-frame encoding option is to encode field pictures as supported by MPEG-2 and H.264. This reduces the buffering required at the decoder to half of that required for a frame. The bit overhead associated with field picture encoding is far lesser than that associated with slice encoding. However, field picture can lead to significant degradation with respect to media quality that can be obtained by encoding frames for progressive video content due to the overhead of transmitting additional motion vectors.

IV. Network Characteristics Management

The type of communication channel used to deliver the stream is an important consideration that affects the system design. Circuit switched networks such as ISDN deliver packets in order while introducing a constant delay and random bit errors. Packet switched networks such as LAN and internet introduce variable delay and even loss of packets. Due to the nature of multimedia data, each packet is required to arrive within a deadline for it to be useful at the receiving client. Any packet that arrives after the intended presentation time of the data that it carries will have to be discarded. Discarding video packets can lead to a sustained error due to the predictive nature of the video compression algorithms.

Client systems connected over packet switched networks will need to employ buffering mechanisms in order to overcome jitter. If the buffer size is constant, it will need to be close to the difference between the maximum packet transmission time and the minimum packet transmission time in order to minimize the number of packets that will be declared as lost. However maintaining a constant buffer of a relatively large size introduces a persistent latency. A constant buffer cannot respond to the variations in the load of the network.

Alternately adaptive buffering mechanisms can be maintained that varies the delay with which a packet is flushed out of the buffer into the media decoding pipeline according to the prevalent load on the network. This model minimizes the average delay and is responsive to the variations in the network load. However an adaptive jitter buffer can cause a drift in the AV synchronization. In order to ensure that the media presentation remains unaffected by the variable buffer at the front-end of the client pipeline, the rate o media presentation will need to be regulated according to the current state of the jitter buffer. For video this would require holding a given frame for duration longer than one frame interval. For audio, this would require a time scale modification process. This increases the system complexity.

Another method of reducing the buffering is the usage of video compression technique called Arbitrary Slice Ordering (ASO). ASO is supported b H.263+ and H.264 video compression standards. This allows the video bitstream to carry independently decodable slices encoding video data in an order other than the raster scan order. The decoder will not expect data to arrive in a raster scan order in a stream where ASO is enabled. The decoder can start to decode data as soon as at least one slice is available. Using ASO jitter buffers can be nearly eliminated in video-only systems. ASO is also useful if the encoder is required to give preferential treatment to a user specified region of interest as required by several video surveillance systems. However H.264 employs algorithms of a higher computational complexity than MPEG-2 or MPEG-4 and may not be suitable for systems with a stringent limit on the decoding complexity. The system designer can avail the benefit of ASO even in MPEG-2 or MPEG-4 based systems using the traditional slice encoding algorithm supported by these standards if the designer has complete control of the design of both the server and client side systems. In this case the client and decoder can be designed to start decoding of slices as soon as they arrive even if they have been re-ordered by the network.

V. Pipelining

The serial nature of activities on the server and client results in adding up the latency at each stage. The overall system latency introduced by processing and scheduling delays can be reduced by pipelining various processing activities and scheduling the media sourcing (media capture, media reception over network) and sinking activities (media transmission and media presentation) such that the pipeline of activities is neither starved nor overloaded.

The media capture devices should be activated once the encoding pipeline is ready to be started. The media presentation device should be activated once displayable data is made available by the decoding pipeline. This minimizes any buffering at the driver. This ensures that the delay between the time at which media data is scheduled for presentation and the actual presentation is only caused by the scanning or processing delays in the device itself.

The transmission of data can be pipelined with the encoding process in order to prevent the transmission delay from being totally additive to the encoding delay. However this will mean that the encoding process will need to be pre-empted if the system is running on a RTOS. This introduces thread switching overheads. The optimum intervals at which transmission can be scheduled has to be determined to ensure that the system can benefit from the pipelining while keeping the overheads t acceptable levels.

When used with slice encoding algorithms, optimal pipelining and scheduling, if employed at both the server and the client ends, can reduce the end-to-end processing delays close to two frame intervals. By employing drivers which can capture and render media at sub-frame resolutions (a row of macroblocks), the end-to-end latency can be brought close to one frame interval. However, in order to maximize the benefit by pipelining the processing activities, the system designer will require control over components on both the server and the client ends which is possible in closed systems such as proprietary surveillance systems.

VI. Conclusions

We examined the causes of latency and the methods commonly used to reduce latency. We examined the affects of the methods on various requirements and performance parameters of various classes of streaming applications.

Video compression methods such as the usage of a low delay rate control algorithm, field encoding and slice encoding techniques reduce buffering requirements but also lead to degradation in media quality. Slice encoding has an additional advantage of rendering the system more robust to errors than otherwise. Buffering requirements introduced by the network jitter can be reduced by employing adaptive buffering mechanisms or ASO at the cost of additional computational complexity. Pipelining processing activities reduces the processing delays to under one frame interval. Optimal scheduling can eliminate buffering by media device drivers and reduce the overall system latency. Several design options are also dependent on whether the system designer has control over both the server and client ends.