Improving Efficiency and Quality of Service Control ds(IEQOSC) in MPEG-4 Adaptive Video Streaming

B.Shaji[†], Dr.A.Vimala Juliet[††], P.R.Jasmine Jeni[†††]

[†, †††]Research scholar, SRM University, Chennai
[††]Professor, SRM University, Chennai

Abstract
Mobile video surveillance represents a new paradigm that encompasses, on the one side, ubiquitous video acquisition and, on the other side, ubiquitous video processing and viewing, addressing both computer-based and human-based surveillance. There are many parameters that affect video quality but their combined effect is not well identified and
Understood when video is transmitted over mobile/ wireless networks. In addition, video content has an impact on video quality under same network conditions. The main aim of this paper is the prediction of video quality combining the application and network level parameters for all content types. Firstly, video sequences are classified into groups
Representing different content types using cluster analysis. The classification of contents is based on the temporal (movement) and spatial (edges, brightness) feature extraction. Second, to study and analyze the behavior of video quality for wide range variations of a set of selected parameters. We use a daily video request model and a video popularity model to determine the adaptive streaming speed and dynamic patching window. Numerical results show that, when the available network bandwidth is reduced below the required level due to background traffic, the efficient algorithms can considerably reduce the average user waiting time and the number of waiting requests. The video itself is streamed on top of the Real-time Protocol (RTP) and the parameters are negotiated with the Real-time Streaming Protocol (RTSP) before the streaming commences. The research problem of this is to provide easy solution for QoS measurements in networks that are closed nature.

Keywords:
Quality of service Control – QOSC, NS2, MPEG video, video adaptation, error correction.

Introduction

One of the most important traffic types in future packet-switch networks is high-bandwidth; variable-bit rate (VBR) compressed video. Compressed video is inherently bursty and will become more so with the advance of compression technology and VLSI chip design. Compressed video sources can adapt its rate dynamically by adjusting compression parameters. As compressed video is a delay-sensitive media, the network must allocate resources to maintain QoS guarantees on delay, delay jitter, loss, and throughput to video connection. Some feedback-based control schemes proposed for the rate-adaptive video traffic. In, an additive-increase, multiplicative decrease algorithm is employed to adapt sending rate of real-time streams based on the feedback of packet loss from end-receivers. The schemes proposed mainly network bandwidth sharing based on the feedback of Resource Management (RM) celli's initially proposed for the ATM Available Bit Rate (ABR) traffic control. Cerla et. a1 proposed a feedback-based algorithm in which network layer provides explicit information of available bandwidth and propagation delay. The protocol enhances the communication between the end nodes anti "the network', and avoids to put stress on routers and more importantly, avoids per-queue, per connection state in routers. The major contribution of this paper is that we use delay as the decision parameter and "network" sensor to monitor the network congestion, and adjust sending rate.

Overview of QoS protocol

The QCP is implemented at the source and the destination only. A QoS packet is employed in QCP to communicate QoS information. There are 6 fields in a QoS packet. Table 1 shows the fields within the QoS packet structure. The QoS packet is inserted after 3 video frames at the source, and is sent from the source to the destination. When the destination receives the QoS packet, it calculates the jitter, delay, loss and throughput. These QoS parameters are written into the appropriate field of the QoS packet and sent back to the source. When the source receives a backward QoS packet from the destination, it adjusts its rate to a new rate based on delay. Delay is employed as a network sensor due to the fact that the large switch queue length leads to large delays while the small queue length leads to small delays. QoS parameters used in our scheme are delay, jitter, loss and throughput. Throughput, jitter, and loss are calculated per QoS period, and delay is calculated per cell. Let t i be the i th QoS period, the time duration from the arriving time of i-l th QoS packet to the arriving time of I th QoS packet. These parameters are defined as follows.
Delay:

Manuscript received October 5, 2010
Manuscript revised October 20, 2010

Delayi = arrival-time - sending-time;
Where Delay, is the delay of the i th cell,
Arrival-time; and sending-time are the arrival and sending timestamps of the i th cell respectively.
Jitter i= Max-delay - Min-delay,
Jitter:
Where Jitter, is the jitter of the i th video frame, Max-delay; and Min delay, are the maximum and minimum delay for the cells within the i th QoS period respectively.
Sum of the lost cells within the ith QoS period.
Throughputi = ∑P/ t i,
Where Throughputi, is the throughput (bit&) during the ith video frame interval, ∑P, is the total bits of all received cells within the ith QoS period. The QCP provide QoS control by doing the following:
The source starts sending the video at its peak cell rate (PCR), Y, = PCR. For every 3 transmitted video frames, the source sends a QoS packet (ID= 1, DIR= 0, DELAY, JITTER, LOSS, THROUGHPUT); upon the receipt of a forward QoS packet, the destination calculates the jitter, loss and throughput, and applies these values to each QoS field as appropriate.
DIR= 1;
DELAY = delay;
JITTER = Max-delay - Min-delay;
Max-delay = Min-delay = delay;
THROUGHPUT = (cells received within
Loss = lost cells within this QoS period;
this QoS period)/(the time interval);
The destination send the QoS packet (ID= 1, DIR= 1, DELAY, JITTER, LOSS, THROUGHPUT) back to the source;
Upon the receipt of a backward QoS packet, the source adjusts its sending rate due to DELAY within QoS packet.
If (DELAY > delay-threshold)
then r, = throughput;
then Y, = Y~ (1+ a), where p is a fraction
Which could be any value from 0.1 to 0.9, ~1 is a parameter set by the user with a default value 1/2. The source can set delay-threshold, a and p. It is observed that choosing the smaller C (could improve the fairness for competing connections. Also delay-threshold should set to be bigger than the propagation delay along the connection.
If the delay is bigger than delay-threshold, that means the switch queue length is big as well, then the algorithm decreases the source rate to the throughput contained in the QoS packet from the destination. In this way, the sending rate is adjusted to the value that the network can transmit. This efficiently reduces the switch queue length, while reduces the delay.

Related to other Protocols

The QoS adaptation of GuenkovaLuy et al.
Occurs in three phases. The first phase is the pre negotiation phase, in which the terminals exchange QoS parameters, addresses and ports, codec's and RTP packetization. The second phase, negotiation phase, is about exchanging parameters relating to the configuration and the establishment of a specific multimedia session. The exchanged parameters define the media streams created for the specific multimedia session. The third phase is the renegotiation phase. The renegotiation phase is about taking into use or discarding specific QoS contracts and context, and indicating adaptation conditions. The renegotiation phase is triggered, if resources change on some level of the system, which leads to a violation of the QoS contract. The scaling of the media stream is done by changing the frame rate, frame size, color quality range and overall quality range.

The QoS framework in video streaming

The adaptation of Vandal ore et al. is based on monitoring the QoS specification that is done in a flexible manner by defining the minimum and maximum values for bandwidth, delay and jitter. The minimum must always be provided. The network conditions are charted by periodic sending of feedback packets containing information from the components in the network at different levels of the architecture. The adaptation, depending on the monitored conditions, in application level affects on the encoding parameters of the video stream. Since the video is layered, layers are either added to the video or removed from the video. The base layer must always be provided, since it contains time critical information and essential image data. Additional layers are optional increasing the video quality when added. In the network layer, the QoS path is renegotiated, if a violation is detected, for example if transmission power falls below defined limit.

Real-time video encoding versus pre-encoded (stored) video system

Video may be captured and encoded for real-time communication, or it may be pre-encoded and stored for later viewing. Interactive applications are one example of applications which require real-time encoding, e.g. videophone, video conferencing, or interactive games. However real-time encoding may also be required in applications that are not interactive, e.g. the live broadcast of a sporting event. In many applications video

content is pre-encoded and stored for later viewing. The video may be stored locally or remotely. Examples of local storage include DVD and Video CD, and examples of remote storage include video-on-demand (VOD), and video streaming over the Internet (e.g. as provided by Real Networks and Microsoft). Pre-encoded video has the advantage that it does not require a real-time encoding constraint. This can enable more efficient encoding such as the multi-pass encoding that is typically performed for DVD content. On the other hand, it provides limited flexibility as, for example, the pre-encoded video cannot be significantly adapted to channels that support different bit rates or to clients that support different display capabilities than that used in the original encoding.

Portioning streaming data

An important observation is that bits which closely follow a resync marker are more likely to be accurately decoded than those further away. This motivates the idea of placing the most important information immediately after each resync (e.g. motion vectors, DC DCT coefficients, and shape information for MPEG-4) and placing the less important information later (AC DCT coefficients). This approach is referred to as data partitioning in MPEG-4. Note that this approach is in contrast with the conventional
Approach used in MPEG-1/2 and H.261/3, where the data is ordered in the bit stream in a consecutive macro block by macro block manner, and without accounting for the importance of the different types of data.

Typical video compression

Consecutive frames in a video sequence exhibit temporal redundancy since they typically contain the same objects, perhaps undergoing some movement between frames. Within a single frame there is spatial redundancy as the amplitudes of nearby pixels are often correlated. Similarly, the Red, Green, and Blue color components of a given pixel are often correlated. Another goal of video compression is to reduce the irrelevancy in the video signal, which is to only code video features that are perceptually important and not to waste valuable bits of information that is not perceptually important or irrelevant. Identifying and reducing the redundancy in a video signal is relatively straightforward, however identifying what is perceptually relevant and what is not is very difficult and therefore irrelevancy is difficult to exploit. To begin, we consider image compression, such as the JPEG standard, which is designed to exploit the spatial and color redundancy that exists in a single still image. Neighboring pixels in an image are often highly similar, and natural images often have most of their energies concentrated in the low frequencies. JPEG exploits these features by partitioning an image into 8x8 pixel blocks and computing the 2-D Discrete Cosine Transform (DCT) for each block. The motivation for splitting an image into small blocks is that the pixels within a small block are generally more similar to each other than the pixels within a larger block. The DCT compacts most of the signal energy in the block into only a small fraction of the DCT coefficients, where this small fraction of the coefficients are sufficient to reconstruct an accurate version of the image. Each 8x8 block of DCT coefficients is then quantized and processed using a number of techniques known as zigzag scanning, run length coding, and Huffman coding to produce a compressed bit stream. In the case of a color image, a color space conversion is first applied to convert the RGB image into a luminance/chrominance color space where the different human visual perception for the luminance (intensity) and chrominance characteristics of the image can be better exploited.

The sequence constraints in normal video streaming

A significant amount of insight can be obtained by expressing the problem of video streaming as a sequence of constraints. Consider the time interval between displayed frames to be denoted by Δ, e.g. Δ is 33 ms for 30 frames/s video and 100 ms for 10 frames/s video. Each frame must be delivered and decoded by its playback time; therefore the sequence of frames has an associated sequence of deliver/decode/display deadlines:
1. Frame N must be delivered and decoded by time TN
2. Frame N+1 must be delivered and decoded by time TN + Δ
3. Frame N+2 must be delivered and decoded by time TN + 2Δ
Any data that is lost in transmission cannot be used at the receiver. Furthermore, any data that arrives late is also useless. Specifically, any data that arrives after its decoding and display deadline is too late to be displayed. (Note that certain data may still be useful even if it arrives after its display time, for example if subsequent data depends on this “late” data.).

Related approaches in video streaming:

Rate control at end-hosts avoids congestion by dynamically adapting the transmission rate. Alternatively, congestion can also be avoided by providing unchanging amount of resources to each flow, but instead limiting the addition of new flows. This is similar to the

telephone system that provides Performance guarantees although with a possibility for call blocking. With all the difficulties facing streaming media systems in the Internet, there has been work towards providing some Quality of Service (QoS) support in the Internet. The Integrated Services (IntServ) model of the Internet, for Instance, is an attempt to provide end-to-end QoS guarantees in terms of bandwidth, packet loss rate, and delay, on a per-flow basis. QoS guarantees are established using explicit resource allocation based on the Resource Reservation Protocol (RSVP). The guarantees in terms of bandwidth and packet loss rate would have greatly simplified streaming media systems. Nevertheless, this is only at the expense of additional complexity in the network. The high complexity and cost of deployment of the RSVP-based service architecture eventually led the IETF to consider other QoS mechanisms. The Differentiated Services (DiffServ) model, in particular, is specifically designed to achieve low complexity and easy deployment at the cost of less stringent QoS guarantees than IntServ. Under DiffServ, service differentiation is no longer provided on a per-flow basis. Instead, it is based on the code-point or tag in each packet. Thus, packets having the same tags are given the same treatment under DiffServ regardless of where they originate. The cost of easy deployment for DiffServ compared to IntServ is the reduced level of QoS support.

Error control resilient video coding

Compressed video is highly vulnerable to errors. The goal of error-resilient video coding is to design the video compression algorithm and the compressed bit stream so that it is resilient to specific types of errors. This section provides an overview of error-resilient video compression. It begins by identifying the basic problems introduced by errors and then discusses the approaches developed to overcome these problems. In addition, we focus on which problems are most relevant for video streaming and also which approaches to overcome these problems are most successful and when. In
Addition, scalable video coding and multiple description video coding are examined as possible approaches for providing error resilient video coding.

Problems in video streaming by errors

Most video compression systems possess a similar architecture based on motion-compensated (MC) prediction between frames, Block-DCT (or other spatial transform) of the prediction error, followed by entropy coding (e.g. run length and Huffman coding) of the parameters. The two basic error induced problems that afflict a system based on this architecture are:
1) Loss of bit stream synchronization
2) Incorrect state and error propagation
The first class of problems, loss of bit stream synchronization, refers to the case when an error can cause the decoder to become confused and lose synchronization with the bit stream, i.e. the decoder may lose track of what bits correspond to what parameters. The second class of problems, incorrect state and error propagation refers to what happens when a loss afflicts a system that uses predictive coding.

Loss of Bit stream video Synchronization

Loss of bit stream synchronization corresponds to the case when an error causes the decoder to lose track of what bits correspond to what parameters. For example, consider what happens when a bit error afflicts a Huffman codeword or other variable length codeword (VLC). Not only would the codeword be incorrectly decoded by the decoder, but because of the variable
Length nature of the code words it is highly probably that the codeword would be incorrectly decoded to a codeword of a different length, and thereby all the subsequent bits in the bit stream (until the next resync) will be misinterpreted. Even a single bit error can lead to significant subsequent loss of information. It is interesting to note that fixed length codes (FLC) do not have this problem, since the beginning and ending locations of each codeword are known, and therefore losses are limited to a single codeword. However,
FLC's do not provide good compression. VLC's provide significantly better compression and therefore are widely used.
The key to overcoming the problem of loss of bit stream synchronization is to provide mechanisms that enable the decoder to quickly isolate the problem and resynchronize to the bit stream after an error has occurred.

Simulation and Experimental Results

Our implementation of the video streaming system consists of a video server, a proxy server, and mobile clients. We assume that all communication between the server and the mobile client is routed through a proxy server typically located in proximity to the client. The video server is responsible for streaming compressed video to the client. The proxy server transcodes the received stream, adds the appropriate

control information, and relays the newly formed stream to the mobile client. In our initial implementation, for the sake of simplicity and without loss of generality, we use the proxy server to also double up as our video server. The Entire Setup is being simulated in NS2 Simulator Environment

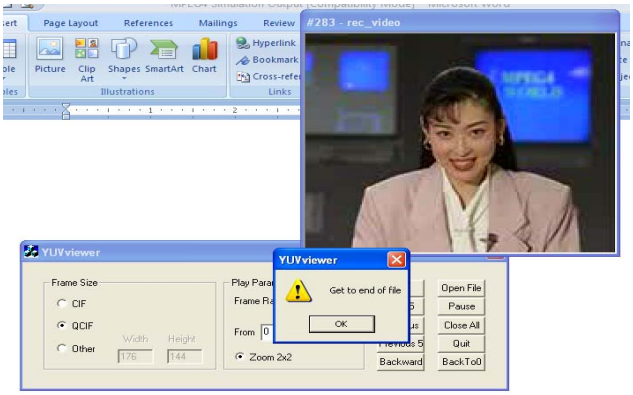

Figure I: Playing the Received Original file using YUV viewer

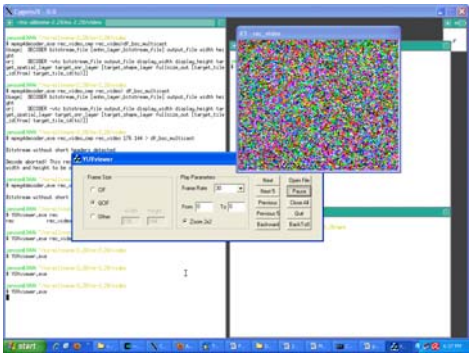

Figure II: View encoded Video packets

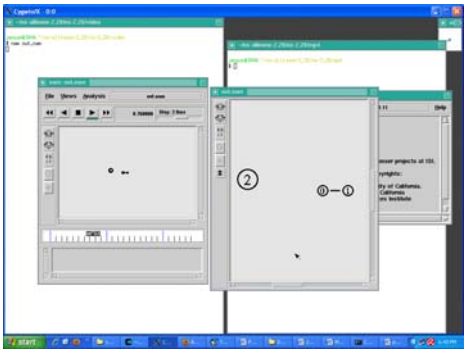

Figure III: NAM Animator Analysis

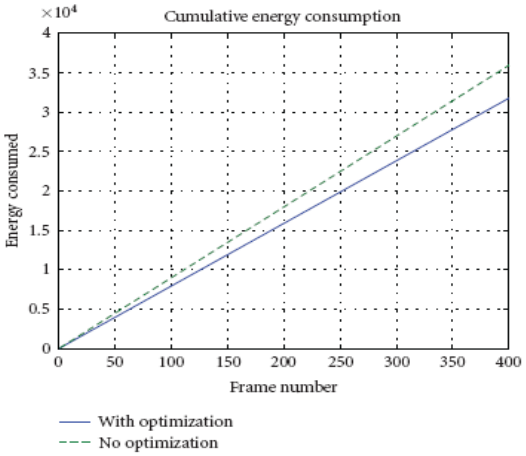

Figure IV. Energy consumption with and without optimization

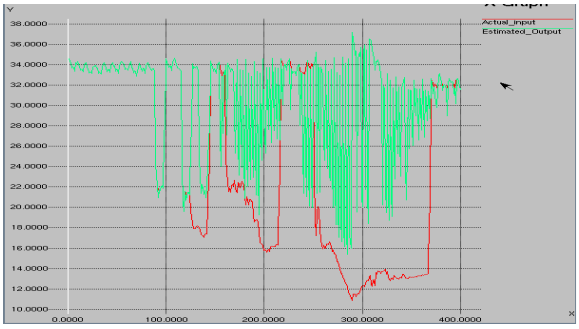

Figure V. PSNR Curve

Conclusion

In this paper, a novel methodology for reliable video streaming over wireless networks has been discussed. The technical basis of the proposed approach relies on advanced concepts like multicarrier modulations for variable-bit-rate transmissions and multi-layered scalable MPEG-4 coding that will characterize future generations of multimedia wireless networks. The application testbed considered for assessing the proposed video transmission method is related to a satellite multicast video streaming service targeted to vehicular users. Extensive tests provided the experimental proof of the robustness of VBR MC-CDM approach, achieved through a UEP mechanism implicit in the multi-carrier multiplexing. Future developments will include the investigation of the use of multi-user detection algorithms, the adaptive management of power resources attributed to each MUX channel, the integration of error resilience tools in the standard MPEG-4 encoding. Furthermore, aspects related to the actual implementation of the proposed video transmission system over HW/SW architectures (DSP) will be considered