CERIAS Tech Report 2004-83

Streaming video and rate scalable compression: what are the challenges for watermarking

by Eugene T. Lin and Christine I. Podilchuk and Ton Kalker and Edward J. Delp

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

Streaming video and rate scalable compression: what are the challenges for watermarking?

Eugene T. Lin[*], Christine I. Podilchuk[**], Ton Kalker[***], Edward J. Delp[+**],

[*]Video and Image Processing Laboratory
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47906

[**]Bell Laboratories
Lucent Technologies
Murray Hill, NJ 07974

[***] Philips Research
NL-5656-AA
Eindhoven, The Netherlands

ABSTRACT

Video streaming, or the real-time delivery of video over a data network, is the underlying technology behind many applications including video conferencing, video-on-demand, and the delivery of educational and entertainment content. In many applications, particularly ones involving entertainment content, security issues, such as conditional access and copy protection must be addressed. To resolve these security issues, techniques that include encryption and watermarking need to be developed. Since the video sequences will often be compressed using a scalable compression technique and transported over a lossy packet network using the Internet Protocol (IP), the security techniques must be compatible with the compression method and data transport and be robust to errors. In this paper, we address the issues involved in the watermarking of rate-scalable video streams delivered using a practical network. Watermarking is the embedding of a signal (the watermark) into a video stream that is imperceptible when the stream is viewed but can be detected by a watermark detector. Many watermarking techniques have been proposed for digital images and video, but the issues of streaming have not been fully investigated. A review of streaming video is presented, including scalable video compression and network transport, followed by a brief review of video watermarking and the discussion of watermarking streaming video.

1. INTRODUCTION

There are many open research problems in video streaming. These include compression, network design and transport, error correction, error concealment, caching, and security issues. For example, current streaming video systems suffer from occasional "glitches" such as temporary loss of video and artifacts arising from network congestion and transmission error. These problems in many cases render the video unacceptable and not viewable. In addition to the quality issues, the security of the video stream is critical to protect the interests of the owner and the users in an environment where perfect copies of digital video sequence can be made.

Securing digital multimedia is a very challenging problem. The owner, who may have spent considerable time and effort producing or acquiring the digital content, desires to ensure that all access to the content is authorized under the rules of a license (conditional access), unauthorized reproductions cannot be easily made (copy protection), and any illegal copies that are created can be detected and traced (content tracking) [1, 2]. Content owners and producers are also terrified that the Internet can facilitate piracy on a large scale, with users distributing unauthorized copies via peer-to-peer file sharing

[+] This research was partially supported by grants to EJD from the Purdue Center for Education and Research in Information Assurance and Security and from the Indiana 21st Century Research and Technology Fund. Address all correspondence to E. J. Delp, ace@ecn.purdue.edu.

software such as Napster and Gnutella. Despite the difficulties, we believe that solutions can be found which benefit both owners and users [3].

In this paper, we will focus on the challenges and issues involved in the watermarking of streaming video. Watermarking [4, 5, 6] is the embedding of a signal (the watermark) into a video stream that is imperceptible when the stream is viewed but can be detected by a watermark detector. The embedded watermark can be the identity of the video owner, the identification of a licensed user, copy-protection and access control information, authentication information, or some other data of interest. For many security applications, the embedded watermark must be detectable even when the watermarked video is altered.

We feel that while there has been a great deal of work in video watermarking, there has been very little reported for the unique characteristics of streaming video. In this overview, we will describe streaming video systems and discuss the role of multicasting and rate-scalable video compression in the watermarking problem.

2. OVERVIEW OF STREAMING VIDEO

A streaming video system is one in which a source encodes video content and transmits the encoded video stream over a data network (wired or wireless) where one or more receivers can access, decode, and display the video to users in real-time[1]. The presence of the network, which allows the source to be physically distant from the receivers, differentiates streaming video from pre-recorded video used in consumer electronic devices such as DVD players.

Given that uncompressed video has very large bandwidth demands, the need for efficient video compression is paramount. A unique problem that streaming presents for the compression technique is that in many applications the network cannot guarantee the bandwidth that is available. This has lead to the development of scalable compression techniques [7, 8, 9]. Much of the work in streaming research has been motivated in finding ways to overcome the limitations of the network [10]. Some of the techniques that improve video streaming systems include error control and mitigation, and scalable video compression.

2.1 The Network

An ideal data network is capable of transferring any amount of information without delay or loss; unfortunately practical networks do not possess such characteristics. In fact, if an ideal network (or an approximation thereof) were to exist, video streaming would be a trivial problem. Practical networks can introduce errors (bit errors), deletions (packet or bit loss) and insertions (cross-talk) [11] have delay and latency, and finite bandwidth. These problems can also vary temporally. Any traffic in the network, whether it is a video stream or some other data, is subject to the constraints of the network.

The performance issues of a network [10, 12] are:

1. Bandwidth - Bandwidth is the amount of data that can traverse through the network or a part of the network at any given time. Network bandwidth is a shared, limited resource and will vary with time. Networks may carry multiple video streams simultaneously or carry non-video data. A network may not be able to guarantee that the required bandwidth for transporting video will be available.

2. Delay, Jitter, and Latency - Streaming video is subject to delay constraints since the video must be decoded and displayed in real-time. If video data spends too much time in the network, it is useless even if it arrives at the receiver. Buffering can reduce the effect of delay and jitter (timing errors). Latency can also be an issue when two-way communication is necessary.

3. “Errors” - The network may cause errors, loss or deletions, and insertions. Data can be lost in the network for a variety of reasons, including congestion, rejection due to excessive delay, and network fault. It is obvious that the streaming video system cannot ignore the possibility of data errors or loss during network transmission. These problems can manifest themselves as unnatural artifacts in the video, such as missing frames, lines, or blocks. Severe loss, such as from heavy network congestion, can cause the video playback to be stopped until the receiver

[1] It should be noted that some applications do not require real-time delivery. An example is a system that pre-loads video into a cache (a hard drive) for later viewing.

can resynchronize. The effective loss experienced by a user can be greater than the actual loss by the network. For example, if the first packet of data corresponding to a video frame is lost or corrupt, the data in all subsequent packets of that video frame may not be decodeable, even if the packets were successfully delivered by the network. Furthermore, errors arising from lost data can affect multiple video frames by *temporal error propagation*.

The above are the primary quality-of-service (QoS) issues for any network. Most networks do not have QoS control, or mechanisms to prioritize network resources for streams that carry high-priority, time-sensitive data. Networks that have QoS control can maintain "guaranteed" resources by rejecting new users if QoS restrictions are violated using "smart" routing and scheduling algorithms, or by using a reservation method for network resources [13, 14]. Typical QoS parameters include minimum available bandwidth, maximum end-to-end delay, maximum bit or packet loss rate, and jitter. Because QoS control is not available for most networks, streaming video systems are usually *end system-based*, which implies that the network is not expected to provide any support for ensuring reliable transmission. (The alternative is a *network-centric* system, which uses QoS control and other network features to assist streaming video.)[2]

In addition to the QoS issues, two other network issues affect the delivery of streaming video: *heterogeneity* and *time-variance*. A heterogeneous network is a network whose parts (sub-networks) may have vastly unequal resources. For example, some parts of a heterogeneous network may have abundant bandwidth and excellent congestion control while other parts of the network are overloaded and congested by overuse or by a lack of physical network resources. Different receivers on a heterogeneous network can experience different performance characteristics. When streaming video over a heterogeneous network, the video stream should be decodeable at optimal quality for users with a good network connection, and at useable quality for users with a poor connection. (This property is addressed below through the use of *scalable* compression.) Time-variance implies bandwidth, delay, loss, or other network characteristics can vary significantly over time, sometimes changing drastically in a matter of seconds. When streaming over a network with time-variance, the steaming video source should be able to adjust its parameters to changing network conditions (*adaptability*.)

The Internet[3] is often the target network for streaming video. It is certainly not the only network that could be used for streaming (for example, cellular and wireless networks.) The Internet is a difficult network for transporting real-time data, it is an example of a heterogeneous, time-varying, network with no QoS control.

2.2 Network Transport Using IP

In the previous section we concentrated on the aspects of the network that concerned errors and quality of service. The video data must be formatted in a way that it can be transported by and routed through the network. Most streaming applications assume a packet switched network that uses the Internet Protocol (IP) [15]. Hence streaming is sometimes referred to as "video over IP." Data is usually sent through the network using TCP/IP. This transport is very well suited to non real-time text-based data and provides reliable communication through the use of retransmission.

Retransmission is almost never possible for video traffic and hence "connectionless" protocols such as UDP are used. Other protocols, such as RTP [16], can be used with UDP for real-time applications. When multiple receivers wish to access common video content this can be accomplished by sending each receiver a separate set of packets known as a unicast stream. In unicast the network delivers an independent copy of the stream from the source to each receiver. This obviously can overwhelm the network if thousands of users are requesting content.

Multicast is a IP protocol that has broadcast-like capability and could be used to transport video [17, 18]. Multicast can significantly reduce the bandwidth required to simultaneously deliver a video stream to multiple receivers and is very useful for video conferencing and streaming of content to a large audience. Multicast is not as useful for video-on-demand applications, where different receivers are viewing different streams or at different points along a common stream. In multicast, the video is delivered from the source in such a way that additional copies of the video stream are created only when necessary. The routers in a multicast network accomplish this feat by establishing a *multicast tree* from the source to all receivers. Only a single copy of the video stream is sent along each link of the tree, even if the link is common in the paths from the source to two or more receivers. Multicast routing is not a trivial problem and the methods by which the tree is constructed and maintained are amongst the current research areas in multicast [14]. Other multicasting issues include

[2] Some providers of high quality entertainment video have proposed deploying "private entertainment networks" where complete control is possible.

[3] By "Internet" we mean the commercial Internet using IP. This does not include intranets, private networks, or Internet2.

adding quality of service control and resilience against network failure. It should be noted that when using multicast each receiver does *not* receive a unique stream, this has interesting implications with respect to security.

2.3 Scalable Video Compression

When one compresses a video sequence the following parameters must be determined: *frame size*, *frame rate*, *data rate*, and *de-compressed quality*. One of the problems with many video compression methods is that these parameters are fixed at the encoding time and cannot be easily changed. For example, suppose one encodes a standard definition video sequence with MPEG-2 [19] at 6 Mb/s and stores it on a network video server that will stream it over a network. At a later time if a receiver requests this video information but does not want it at 6 Mb/s, but at 4 Mb/s, the server would have to transcode the sequence to meet the new rate requirement, which is very computationally intense. Another way to address this problem is to store multiple copies of the compressed sequence at the video server that is then able to satisfy different users' needs; this is very expensive. Similarly a viewer may want the sequence at a different spatial resolution (i.e., different frame size) or at a different frame rate. This is an extremely important issue for a media producer who may have to provide content to be delivered at different resolution (temporal, spatial and/or rate) levels depending on the receivers' capability as well as the users' choice. A *scalable* compression technique is one that allows compressing the video data once and then decompressing it at multiple data rates, frames rates, spatial resolutions, and/or video quality (SNR). Such a compression technique would be very desirable from a networking viewpoint as it allows differentiated quality and bit rates depending on the kind of service chosen by the user. An important question relative to scalability that impacts its use in video streaming is how often and fast can the compression parameters be changed. We shall say that a compression scheme is *dynamically scalable* if the compression parameters can be changed many times during the transmission process, i.e., during the streaming of the sequence. The network could then use this when congestion occurs to change the compression parameters on the fly and hence reduce the bandwidth being used. In this paper we will emphasize the use of rate-scalable compression.

Rate-scalable compression encodes the video such that the compressed video stream can be decoded at multiple rates, with increasing quality as the data rate increases. The video provider requires only one rate scalable video stream to support all the users in a heterogeneous network; the users with high bandwidth receive and decode the entire compressed video stream while those with lower bandwidth may only receive and decode a portion of the same video stream. Rate-scalable compression also allows the system to handle time-variance in the network, as each receiver receives the video stream at a rate that the network will allow at a given moment. A disadvantage for rate-scalable compression is compression efficiency.

There are several strategies for rate-scalability, including layered scalability, embedded coding, and hybrid layer/embedded coding. The scalability modes of most compression standards use layered scalability; this includes MPEG-4 and H.263+ [20, 21]. In layered scalability, the compressed video stream describes a *base layer* and one or more *enhancement layers*. The base layer is encoded at the minimum rate necessary to decode the video stream, and its decoding results in the lowest quality version of the video. Successive enhancement layers improve the quality of the video beyond that of the base layer. For example, a video source encoded using three layers, 64 kb/s base layer, 128 kb/s enhancement layer, and an additional 128 kb/s second enhancement layer can be decoded at 64 kb/s (base layer only, lowest quality), 192 kb/s (base + first enhancement, intermediate quality), and 320 kb/s (base + full enhancement, best quality.) To decode at a given enhancement level, the receiver requires access the base layer and all enhancements layers up to the given level.

An alternative to using layer scalability is the use of an embedded coder [22, 23]. In an embedded coding scheme, the compressed video data is coded as a single unit without the use of distinct layers. For each frame, the decoder receives the compressed video stream from the picture header up till the available data rate, at which point the decode process stops and the video quality corresponding to that rate is achieved. To achieve the best performance, the most important information about the picture is placed at the beginning of the compressed video stream, followed by information of decreasing importance. A hybrid layer/embedded scalability technique combines layered and embedded coding, such as the Fine Grain Scalability (FGS) mode of MPEG-4 [8, 24]. Similar to layer scalability, the compressed video stream consists a base layer and enhancement layer. However, instead of using multiple enhancement layers to produce decodeable versions of intermediate quality, a single enhancement layer is coded using an embedded coding strategy. In MPEG-4 FGS, the MPEG-4 base layer is enhanced using bit-plane coding of the DCT coefficients of the residual.

2.4 Error Control, Resilience, and Concealment

Error control provides layers of defense that serve to increase the tolerance of the video stream to errors during transmission. The first mechanism is the use of forward error correction or (rarely) retransmission. As mentioned above, retransmission is typically not suited for video traffic because the latency involved in sending the retransmit request and reply is too great. In addition, the additional communication involved in retransmission is itself subject to error and loss. Forward error correction (FEC) [12, 25, 26] is the use of channel coding or joint source/channel coding to add redundancy to the compressed bit stream so that errors can be corrected at the receiver without interaction with the source or network. The disadvantage of FEC is that additional bandwidth must be consumed for transmitting the error correction information and that overhead can be significant.

The goal of error resilience [13, 27, 28] is to exploit redundancy in the bit stream syntax to minimize the effect of any damage that is not corrected by FEC, such as by isolating the errors, recovering data that may have been lost after de-synchronization, and error concealment. In H.263+, some of the mechanisms for error resilience [29] include the use of slices (Annex K) and segments (Annex R) to provide points for resynchronization and the ability to use pictures other than the most recently decoded picture for prediction (Annex N). The error resilience features in MPEG-4 [30] include reversible variable-length codes, "video packets" (similar to H.263+ slices) and data partitioning for resynchronization, and the ability to add redundant copies of header information (to avoid dropping entire frames when headers are lost or corrupted.) Finally, error concealment [25, 31] can be used to minimize the visual impact arising from data loss when the video is displayed to the user by predicting the contents of missing or corrupted areas in the video.

2.5 Summary

From the previous discussions, it is clear that the network issues present a daunting challenge for streaming video. If a network such as the Internet is used to deliver streaming video, additional design elements beyond an encoder, network interface, and decoder can be introduced to overcome the network delivery issues. A schematic view of a video streaming system is shown on Figure 1. The video and audio sources are compressed and multiplexed into a single binary stream, adding synchronization and control information necessary for demultiplexing and playback. (For example, this may be MPEG video and audio streams multiplexed by the MPEG system layer, or H.263 video and compressed audio multiplexed with H.223 [32].) The binary stream is then packetized. During the packetization, forward error correction can be added into the binary stream, as well as framing and timing information.

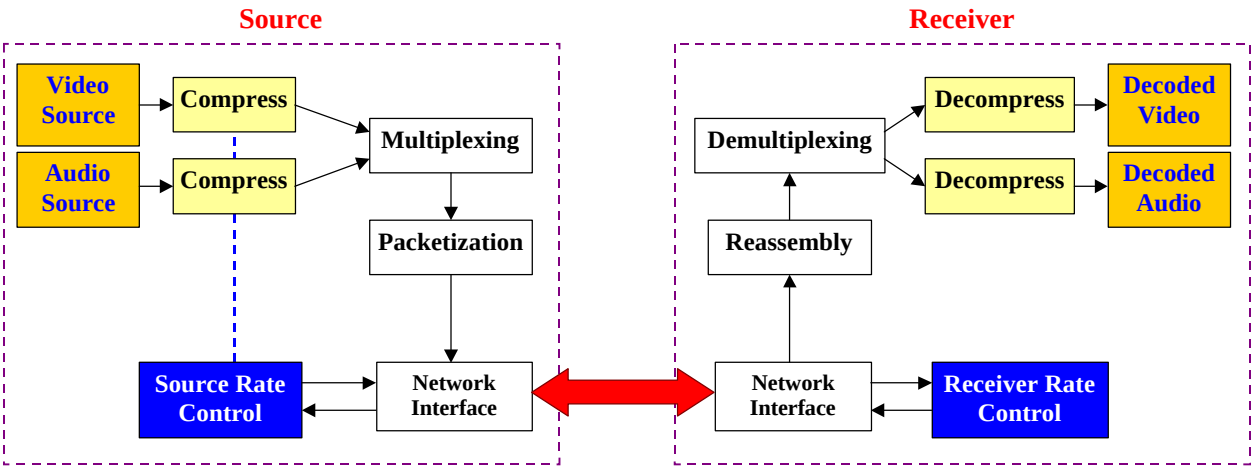

Figure 1. A Video Streaming System

3. OVERVIEW OF VIDEO WATERMARKING

Early video watermarking techniques were proposed as still image watermarking techniques extended to video by watermarking each frame independently. While video-specific properties were not considered, many concepts in image watermarking such as spread-spectrum [33], non-invertibility [34] and signal-adaptivity [35], visual models [4, 36], synchronization [37, 38], and attack [39, 40] can be applicable to video watermarking. While many methods for watermark design, embedding, detection, and attack have been proposed and debated, a general consensus is that embedded watermarks should be *invisible*, *robust* (for non-authentication watermarks), and have high *capacity* [5]. Invisibility is the degree that an embedded watermark remains unnoticeable when a user views the watermarked contents. Robustness is the resilience of an embedded watermark against removal by signal processing, such as by lossy compression, re-scaling, rotation, cropping, warping, addition of random noise, blurring, or re-sampling (D/A, A/D conversion.) Capacity is the amount of information that can be expressed by an embedded watermark. Theoretical capacity of embedded watermarks has been examined using information-theoretic concepts [41, 42].

Digital video is not merely a sequence of images displayed at regular time intervals. Some of the video characteristics that impact watermarking include:

- High spatial correlation between successive frames. In most cases, successive frames of video are not independent and have a high degree of similarity. If independent watermarks are embedded on each frame, an attacker could perform frame averaging to remove significant portions of the embedded watermark.
- Embedding the same watermark in all frames may be insecure, as the attacker would have a lot of information about the structure of the watermark for estimation and removal.
- Some applications, such as watermarking live content, embedding copy-control information, and embedding fingerprints for content tracking, require real-time watermark embedding.
- Other applications, such as detecting copy protection information and on-the-fly video authentication, require real-time watermark detection.
- Embedding a watermark must not significantly increase the data rate of the watermarked video stream, especially for streaming video applications where bandwidth is scarce.

The real-time requirements for some applications imply that complex feature extraction, full-frame search with sliding correlators (such as for resynchronization), and other computationally expensive tasks cannot be performed.

3.1 Video Watermarking

Watermark embedding in the compressed domain (inserting the watermark directly into the compressed bit stream) has the advantage in that the watermark embedder can process the compressed video stream, which has a much lower data rate than uncompressed video. Re-compression is also not necessary after the watermark has been embedded, which is a significant savings for a real-time watermark embedding (the motion estimation procedure used by many compression schemes is computationally intensive.) To embed in the compressed domain, the watermark embedder must parse the video stream to identify structures suitable for watermark embedding and then modify the compressed stream to embed the watermark. Watermarking streams which have been compressed by block motion-compensated video compression techniques has been addressed, partially motivated by the fact that most modern video compression standards (including H.261, H.263+, MPEG-1, MPEG-2, and (parts of) MPEG-4) use block motion-compensation to take advantage of spatial and temporal redundancy in the video. Common features in these compression techniques include the use of the discrete cosine transform (DCT), encoding in units of blocks and macroblocks, and motion vectors. Also, some pictures are intra-coded (coded without using temporal prediction) while other pictures are coded using a previously decoded picture as a predictor.

Langelaar [43] describes a technique for embedding a watermark into the intra-coded pictures of a video sequence. The 8x8 blocks of the original intra-picture are randomly grouped into sets of a fixed size. Each set of blocks encodes a single bit of the watermark, which is determined by comparing the total energy contained the higher-frequency DCT coefficients of half the blocks of the set with the energy contained in corresponding coefficients of the other half of the set. If necessary, DCT coefficients are zeroed in some blocks to express the desired watermark bit. Zeroing DCT coefficients also guarantees that the video data rate does not increase, as it is more efficient to code zero coefficients. The watermark is vulnerable to transcoding, especially if a different GOP (group of picture) structure was used (with a different set of intra-pictures.)

Hartung [44] proposed a spread-spectrum watermarking technique, where the watermark is inserted into the non-zero DCT coefficients of the original video sequence. To construct the embedded watermark, the watermark bits are replicated to add redundancy (the authors use the term *spreading*) and modulated by a pseudo-random noise signal before being added to a DCT coefficient of the original picture. The amount of replication performed for each watermark bit is a robustness and capacity tradeoff, with more replication yielding more robustness but smaller capacity. Only non-zero DCT coefficients in the original picture are watermarked, and then, only if the embedding does not increase the data rate of the video. (It was noted that typically 10-20% of the DCT coefficients were altered to embed a watermark, depending on the video sequence, bit rate, and GOP structure.) Synchronization is a significant problem in this technique, particularly for real-time watermark detection in the presence of temporal (frame insertion, deletion, swapping, re-sampling) attacks. Some robustness to transcoding is claimed. It is possible to embed the watermark in other parts of the compressed video stream, such as the motion vector, GOP, or other information structures in the bit stream. However, such watermarks have not been very robust when the video stream is re-compressed (i.e. transcoded.)

Other video watermarking techniques do not embed directly into compressed streams. The advantage of these techniques is that the watermark is not dependent on the structure and intricacies of a video compression method (for example, compression that is not based on block motion-compensation.)

Swanson [45] proposed a watermark based on a temporal wavelet transform. The video sequence is segmented to scenes, and for each scene a temporal wavelet transform is performed over all the frames of that scene. The temporal wavelet transform ensures that parts of the watermark will vary from picture to picture while other parts remain constant, providing robustness against inter-frame collusion (such as frame-by-frame pixel averaging.) It was mentioned that the detection does not require knowledge of the location of the frame within the scene (i.e. the frame index.) The technique is robust, but the disadvantage is that a large amount of buffering and computation is required to embed the watermark.

Kalker [46] describes a watermark that is embedded in the spatial domain. A pseudo-random Gaussian watermark is embedded into the pixel levels of the original frame. A Laplacian high-pass filter is used to control the amplitude of the watermark, minimizing watermark visibility in smooth areas of the video. By using a small (128x128 pixel) watermark that is tiled to fill the size of the video frame, spatial shifts in the watermark can be detected and overcome without exhaustive search. The detection process is performed in the FFT domain to efficiently synchronize the watermark in the presence of a spatial shifting attack, and it is noted that the best detection results occur if only the phase information of the FFT is used.

3.2 Authentication Watermarking

Authentication is the process of verifying that a video stream originated from an alleged source and that the stream has not been altered or tampered. Fragile and semi-fragile watermarks have been proposed to authenticate digital images [47, 48, 49, 50], but comparatively little work has been done for authenticating video. One method for authenticating video is to embed an image authentication watermark for each frame. Such a technique could detect frame dropping and insertions, but would not authenticate the synchronization between the audio and corresponding video pictures [51]. There is also little work in semi-fragile watermarks for scalable digital video.

4. WATERMARKING STREAMING VIDEO

To summarize the previous discussions, the network is the most significant component of a streaming video system. A network may have many limitations, including limited bandwidth, lack of QoS control, heterogeneity, and time-variance. To overcome those limitations, streaming video systems can employ a variety of techniques such as error control, and rate-scalable video compression. Multicasting is a technique by which a source can efficiently deliver a video stream to many receivers. The primary effect of the network is that the video stream transmitted by the source may be lost or corrupted before reaching a receiver. Network loss and errors can damage embedded watermarks, removing them from the video or reducing their robustness. Authenticating digital video streams over the network is also a considerable challenge because of network errors.

Scalable video compression affects watermarking in two ways. First, scalable compression allows receivers to decode the video stream at different rates. Some receivers, because of limited bandwidth or network heterogeneity, may not receive the entire video stream (and hence, the complete watermark.) Second, the interaction between the enhanced and non-enhanced versions of the video stream must be considered in the design of a watermark for scalable video. Multicast transport

presents a challenge for content tracking and fingerprinting applications, where the goal is to deliver a different video stream to each receiver but the multicast framework allows only a single version of the stream to be distributed.

4.1 Research Issues

Watermark robustness and the network. Many applications require a watermark to be embedded at the source and detected at a receiver, such as the insertion of copyright and conditional access information. For these applications, the watermark must survive network transmission and still possess sufficient robustness for the level of security required by the application. Unlike the typical watermark attack scenario, the network has no constraint to preserve the quality of the video. Is it possible to design watermarks that are more resistant to network loss? Can the error control and resilience features in the video stream be exploited or modified to protect embedded watermarks more effectively?

Where to embed the watermark? A video stream could be watermarked at the source, the network, or a receiver. Embedding watermarks at the receiver would side step the robustness issues for transmitting a watermark through the network, but can be problematic in applications where the security of a receiver cannot be guaranteed. For example, a receiver could be compromised to reveal watermarking keys or embed forged watermarks. If the watermarking keys are not stored at the receiver, a secure method of sharing keys is necessary. Watermark embedding in the network has not been practical because these watermarking techniques require support from the network components. It remains to be seen whether network providers are willing to provide such support, especially if video streams are a relatively small fraction of their traffic. Another alternative is to have the source embed part of a watermark, and the receivers embed another part.

Authentication of video streaming content. A good semi-fragile watermark for authenticating streaming video is still an research problem. Unlike still images, semi-fragile watermarks for video must consider temporal effects such as frame dropping, insertion, and transposition. One possible method for detecting frame manipulation is to embed the frame index in the watermark for each frame, but this is not sufficient for a good semi-fragile video watermark. Temporal error propagation must be tracked if using motion-compensated video compression. (For example, any blocks that refer to a tampered block should also be identified "in doubt.") The semi-fragile watermark should also protect the accompanying audio and video-audio synchronization data and not be removed if the video is transcoded.

Attacking watermarks with error-concealment techniques. Error control and resilience are generally thought of as beneficial to watermarks, as the methods that protect the video data also protect an embedded watermark. However, error concealment techniques, which attempt to estimate the contents of damaged or missing sections of the video stream by using spatial and temporal estimation, can be used in attacking watermarks. An attacker could introduce intentional loss in the video stream in hopes of destroying a watermark, using error resilience to "restore" the visual quality of the video after the attack. An error resilience technique could also be modified to construct an approximation of the video that does not contain the watermark. What are the necessary conditions for video watermarks to resist such attacks?

Synchronization. The spatial and temporal synchronization of the embedded watermark is the Archilles' heel of many blind watermarking techniques, including spread-spectrum, as watermark detection becomes impossible once synchronization is lost. The synchronization mechanism is often remarkably fragile despite the purported "robustness" of a watermark, and hence is often the target of attack. Synchronization attacks themselves do not destroy an embedded watermark, but render it undetectable. (If synchronization could be restored, the watermark would be readily detected.) Spatial synchronization attacks include rescaling, cropping, rotation, shifting, and warping of video pictures. Temporal synchronization attacks include frame dropping, frame insertion, frame swapping, and temporal re-sampling. In streaming video, network congestion alone can cause many consecutive frames to be dropped, leading to lost synchronization.

The vulnerability of watermarks to synchronization attacks had been recognized in still image watermarking, and much work was focused on designing watermarks more resilient to such attacks or recovery of synchronization after an attack. Unfortunately, many of the resynchronization techniques are computationally expensive and are not feasible for real-time watermark detection. Streaming video can also have initial synchronization problems, where a user begins viewing a video stream from an arbitrary temporal location and the watermark detector must synchronize to that location. (For example, the user could begin watching a live streaming video broadcast of an ongoing event.)

Some methods for establishing and re-establishing synchronization are mentioned in [5], including the embedding of special synchronization markers within the watermarked content. The resynchronization markers must be detectable in real-time and yet cannot be predicted and removed by an attacker. The necessary search required for temporal re-synchronization can be reduced using a periodic or cyclic encoding for the watermark, similar to Kalker's [46] technique for reducing the necessary search for resisting spatial shifts. Another possibility is to use synchronization markers that are embedded in the video stream, however the watermark must not be vulnerable to a replacement attack (where the attacker removes the markers and inserts new ones, perhaps adding a small amount of shifting or warping.) Other methods for fast resynchronization should be investigated.

Fingerprinting under Multicast. Some applications require different watermarks to be embedded to the stream delivered to each receiver. Multicast is a very efficient way of delivering the video, but all receivers receive the identical video stream (in the absence of network loss.) Is there a way to take advantage of multicast but be able to embed a different watermark to the video for each recipient? One proposed method [52] is to have each network router or node embed part of the watermark as they relay the video stream. An advantage of this method is that a trace of the route that the video stream traversed the network is embedded in the watermark. The embedded watermarks could also be more resilient to collusion between recipients that share a common path from the source. A major disadvantage, however, is that this technique requires support from the network routers and as mentioned before, such support may not be forthcoming by network providers. Watermarking by the recipient is a possible solution, accompanied by encryption of the stream by the source. (Encryption is necessary to prevent an unwatermarked video stream from being intercepted "in the clear.")

Watermarking and rate-scalable compression. The challenge for watermarking rate-scalable compressed video is that not all receivers will have access to the entire (watermarked) video stream. The embedded watermark must be detectable when only the base layer is decoded (for layered and hybrid layered/embedded methods) or for a low rate version of the video stream (for embedded methods.) However, the enhancement information adds value to the video stream and should not be left unprotected by a watermark. Ideally, there should be a uniform improvement in the detectability of an embedded watermark as the decoded rate increases.

One method for watermarking rate-scalable video streams is to embed a watermark in the base layer and a separate watermark in the enhancement layers. For temporal scalability, this is an effective method for watermarking as the enhancement information does not alter the frames encoded in the base layer. However, for other forms of scalability, care must be taken so that the multiple watermarks do not interfere with each other once the decoder merges the base and enhancement information. The watermarks could interfere in visibility, where the distortions introduced by adding all watermarks is unacceptable, or detectability, where the presence of all the watermarks impair the ability to detect each watermark individually. The ability to detect each embedded watermark individually (before the enhancement and base information are merged) is not sufficient for a robust watermark, as such a system would be vulnerable to a collusion attack between the non-enhanced and enhanced versions of the video.

For embedded scalability modes, one could design a watermark analogous to an embedded coding scheme, where the most significant structures of the watermark are placed near the beginning of the video stream, followed by structures of lesser significance. In the context of watermarking, "significance" implies detectability, as certain portions of the watermark may be more important to its detectability more than others. In such a watermark, increasing the video data rate also increases the watermark detectability, preventing collusion between a low-rate and a high-rate version of the video stream.

5. CONCLUSIONS

We have reviewed digital video streaming and watermarking techniques, and discussed some of the issues that need to be addressed for the watermarking of streaming video content. Some of the problems may not have workable solutions or have impractical solutions that require major changes in network infrastructure. Currently, video streaming systems are a sort of novelty, as the technical issues have not been solved to enable high quality video streaming. One could even argue that none of the content that is currently being streamed is worth protecting with watermarks. However, we believe that the technical limitations for the delivery of high quality video streams will be overcome in the near future, and when that time comes a good security solution must be in place to protect the rights of all users.

REFERENCES

1 A. Eskicioglu and E. Delp, "An overview of multimedia content protection in consumer electronics devices," *Signal Processing: Image Communication,* vol. 16, pp. 681- 699, 2001.

2 J. Bloom, I. Cox, T. Kalker, J. Linnartz, M. Miller, and C. Traw, "Copy protection for DVD video," vol. 87, no. 7, pp. 1267-1276, July 1999.

3 E. Lin, G. Cook, P. Salama, and E. Delp, "An overview of security issues in streaming video," *Proceedings of the International Conference on Information Technology: Coding and Computing*, April 2-4, 2001, Las Vegas, pp. 345-348.

4 R. Wolfgang, C. Podilchuk, and E. Delp, "Perceptual watermarks for digital images and video," *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1108-112, July 1999.

5 F. Hartung, M. Kutter, “Multimedia watermarking techniques,” *Proceedings of the IEEE*, vol. 87, no. 7, pp. 1079-1107, July 1999.

6 M. Swanson, M. Kobayashi, and A. Tewfik, “Multimedia data-embedding and watermarking technologies,” *Proceedings of the IEEE*, vol. 86, no. 6, pp. 1064-1086, June 1998.

7 E. J. Delp, P. Salama, E. Asbun, M. Saenz, and K. Shen, “Rate scalable image and video compression techniques” *Proceedings of the 42nd Midwest Symposium on Circuits and Systems*, August 8-11, 1999, Las Cruces, New Mexico.

8 W. Li, “Overview of fine granularity scalability in MPEG-4 standard,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 3, pp. 301-317, March 2001.

9 H. Lou, C. Podilchuk, J. Choi, “Progressive video streaming over 2G and 3G wireless systems,” *Proceedings of the 11th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC 2000)*, September 18-21, 2000, vol. 2, pp. 1550-1554.

10 J. Lu, “Reactive and proactive approaches to media streaming: from scalable coding to content delivery networks,” *Proceedings of the International Conference on Information Technology: Coding and Computing*, April 2-4, 2001, Las Vegas, pp. 5-9.

11 M. Davey and D. MacKay, “Reliable communication over channels with insertions, deletions, and substitutions,” *IEEE Transactions on Information Theory*, vol. 47, no. 2, pp. 687-696, February 2001.

12 D. Wu, Y. Hou, Y. Zhang, “Transporting real-time video over the Internet: Challenges and approaches,” *Proceedings of the IEEE*, vol. 88, no. 12, pp. 1855-1875, December 2000.

13 W. Tan, A. Zakhor, “Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol,” *IEEE Transactions on Multimedia*, vol. 1, no. 2, pp. 172-186, June 1999.

14 B. Wang and J. Hou, “Multicast routing and its QoS extension: Problems, algorithms, and protocols,” *IEEE Network*, pp. 22-36, January/February, 2000.

15 W. Stevens, *TCP/IP Illustrated, Volume 1*, Addison-Wesley, Reading, Massachusetts, 1994.

16 RFC 1889, RTP: A Transport Protocol for Real-Time Applications

17 C. Miller, *Multicast networking and applications*, Addison-Wesley, Reading, Massachusetts, 1999.

18 K. Savetz, N. Randall, and Y. Lepage, *MBONE Multicasting Tomorrow’s Internet*, IDG Books, Foster City, California, 1996.

19 B. G. Haskell, A. Puri and A. N. Netravali, *Digital Video: An Introduction to MPEG-2*, Chapman and Hall, 1997.

20 T. Ebrahimi and C. Horne, “MPEG-4 natural video coding - an overview,” *Signal Processing: Image Communication*, vol. 15, no. 4, pp. 365-385, 2000.

21 G. Cote, B. Erol, M. Gallant, F. Kossentini, “H.263+: Video coding at low bit rates,” *IEEE Transactions On Circuits And Systems For Video Technology*, vol. 8, no. 7, pp. 849-866, November 1998

22 K. Shen and E. J. Delp, "Wavelet Based Rate Scalable Video Compression," *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 9, No. 1, February 1999, pp. 109-122.

23 B. Kim, Z. Xiong, W. Pearlman, “Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)”, *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 10, no. 8, pp. 1374-1387, December 2000.

24 H. Radha, “The MPEG-4 fine-grained scalability video coding method for multimedia streaming over IP, *IEEE Transactions on Multimedia*, vol. 3, no. 1, March 2001.

25 Y. Wang, Q. Zhu, “Error control and concealment for video communication: A review,” *Proceedings of the IEEE*, vol. 86, no. 5, pp. 974-997, May 1998.

26 B. Girod, K. Stuhlmüller, M. Link, U. Horn, “Packet loss resilient internet video streaming,” *Proceedings of the SPIE Visual Communications and Image Processing 1999*, vol. 3653, part 2, pp. 833-844, San Jose, California, January 25-27, 1999.

27 J. Villasenor, Y. Zhang, and J. Wen, “Robust video coding algorithms and systems,” *Proceedings of the IEEE*, vol. 87, no. 10, pp. 1724-1733, October 1999.

28 J. Apostolopoulos, “Error-resilient video compression,” *Proceedings of the SPIE International Conference on Multimedia Systems and Applications II*, vol. 3845, pp. 180-191, September 20 - 22, 2000, Boston, Massachusetts.

29 S. Wenger, G. Knorr, J. Ott, F. Kossentini, “Error resilience support in H.263+,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 7, pp. 867-877, November 1998.

30 R. Talluri, “Error resilient video coding in the ISO MPEG-4 standard,” *IEEE Communications Magazine*, vol. 36, no. 6, pp. 112-119, June 1998.

31 P. Salama, N. B. Shroff, and E. J. Delp, "Error concealment in encoded video," *IEEE Journal on Selected Areas in Commuincations*, Vol. 18, No. 6, pp. 1129 –1144, June 2000.

32 "Multiplexing protocol for low bit rate multimedia communication, ITU Standard H.223, March 1996.
33 I. Cox, J. Kilian, T. Leighton, and T. Shamoon, "Secure spread spectrum watermarking for multimedia," *IEEE Transactions on Image Processing*, vol. 6, no. 12, pp. 1673-1687, December 1997.
34 S. Craver, N. Memon, B. Yeo, and M. Yeung, "Resolving rightful ownership with invisible watermarking techniques: limitations, attacks, and implications," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 573-586, May 1998.
35 J. Su and B. Girod, "Power-spectrum condition for energy-efficient watermarking," *Proceedings of the IEEE International Conference on Image Processing*, Kobe, Japan, October 1999.
36 C. Podilchuk and Z. Wenjun, "Image-adaptive watermarking using visual models," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 4, pp. 525-539, May 1998.
37 M. Kutter, "Watermarking resisting to translation, rotation, and scaling," *Proceedings of the SPIE Multimedia Systems and Applications*, vol. 3528, pp. 423-431, Boston, Massachusetts, November 2-4, 1998.
38 M. Alghoniemy and A. Tewfik, "Geometric distortions correction in image watermarking," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents II*, vol. 3971, pp. 82-89, San Jose, California, January 24-26, 2000.
39 F. Petitcolas, R. Anderson, "Weaknesses of copyright marking systems," *Proceedings of the Multimedia and Security Workshop at ACM Multimedia '98*, pp. 55-62, Bristol, United Kingdom, September 1998.
40 F. Hartung, J. Su, and B. Girod, "Spread spectrum watermarking: Malicious attacks and counterattacks," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents*, vol. 3657, pp. 147-158, San Jose, California, January 25-27, 1999.
41 P. Moulin and J. O'Sullivan, "Information-theoretic analysis of watermarking," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing '00*, vol. 6, pp. 3630-3633, Istanbul, Turkey, June 5-9, 2000.
42 B. Chen and G. W. Wornell, "Quantization index modulation: A class of provably good methods for digital watermarking and information embedding," to appear in *IEEE Transactions on Information Theory*.
43 G. Langelaar, R. Lagendijk, and J. Biemond, "Watermarking by DCT coefficient removal: A statistical approach to optimal parameter settings," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents*, vol. 3657, pp. 2-13, San Jose, California, January 25-27, 1999.
44 F. Hartung and B. Girod, "Watermarking of uncompressed and compressed video," *Signal Processing*, vol. 66, no. 3, pp. 283-301, May 1998.
45 M. Swanson, B. Zhu, B. Chau, and A. Tewfik, "Multiresolution video watermarking using perceptual models and scene segmentation," *Proceedings of the IEEE International Conference on Image Processing*, vol. 2, pp. 558-561, Santa Barbara, California, October 1997.
46 T. Kalker, G. Depovere, J. Haitsma, and M. Maes, "A video watermarking system for broadcast monitoring," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents*, vol. 3657, pp. 103-112, San Jose, California, January 25-27, 1999.
47 J. Fridrich, "Image watermarking for tamper detection," *Proceedings of the IEEE International Conference on Image Processing*, vol. 2, pp. 404-408, Chicago, Illinois, October 1998.
48 P. Wong, "A watermark for image integrity and ownership verification," *Final Program and Proceedings of the IS&T PICS 99*, pp. 374-379, Savannah, Georgia, April 1999.
49 E. Lin, C. Podilchuk, and E. Delp, "Detection of image alterations using semi-fragile watermarks," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents II*, vol. 3971, pp. 152-163, San Jose, California, January 24-26, 2000.
50 D. Kundur, D. Hatzinakos, "Towards a telltale watermarking technique for tamper-proofing," *Proceedings of the IEEE International Conference on Image Processing*, vol. 2, pp. 409-413, Chicago, Illinois, October 1998.
51 J. Dittmann, M. Steinebach, I. Rimac, S. Fischer, R. Steinmetz, "Combined video and audio watermarking: Embedding content information in multimedia data," *Proceedings of the SPIE Security and Watermarking of Multimedia Contents II*, vol. 3971, pp. 455-464, San Jose, California, January 24-26, 2000.
52 I. Brown, C. Perkins, J. Crowcroft, "Watercasting: Distributed watermarking of multicast media," *Networked Group Communication '99*, pp. 286--300, Pisa, Italy, November 1999.