1

Challenges in Distributed Video Management and Delivery

Rainer Lienhart[†], Igor Kozintsev[†], Yen-Kuang Chen[†], Matthew Holliman[†], Minerva Yeung[†], Andre Zaccarin[†1], and Rohit Puri[‡2]

[†] *Intel Corporation, Santa Clara, California, USA*

[‡] *EECS Dept, UC Berkeley, USA.*

Emails: *{Rainer.Lienhart, Igor.v.Kozintsev, Matthew.Holliman, Yen-Kuang.Chen, Minerva.Yeung}@intel.com, Andre.Zaccarin@gel.ulaval.ca, rpuri@eecs.berkeley.edu.*

1. INTRODUCTION

In recent years, we have witnessed the evolution of computing paradigms with the advent of the Internet as a dominant application, content and service delivery platform as well as with the gradual migration of some applications and services from client-server, to edge services, and then to peer-to-peer (P2P) and distributed computing systems. P2P computing systems are capable of accommodating a vast number of users – as publishers as well as consumers. Distributed video storage, distributed video search & retrieval, distributed video processing, and distributed video distribution, among others, bear many opportunities but along also many technical challenges.

The computing platforms and devices connected to a P2P network, however, can differ significantly in their computing capabilities as well as in the speeds and types of network connectivity. This poses serious practical challenges especially for video [16][24]. In future, the diversity of these capabilities will only increase as the infrastructure must accommodate wireless connections and devices like personal digital assistants (PDAs), tablets, compact media players, and even cell phones, as illustrated in Figure 1. Combined digital devices like mobile phones with attached cameras are reality aggressively pushed by mobile services providers such as NTT DoCoMo

[1] Present affiliation: ECE Dept, Université Laval, Quebec City, Quebec, Canada

[2] Dr. R. Puri's research was supported in part by the grant from Intel Corporation during years 2000 to 2002. The authors would also like to acknowledge Professor Kannan Ramchandran and Abhik Majumdar of University of California at Berkeley for their insightful comments and discussions, as well as for their help in improving the quality of this document.

and others. Together with the expected emergence of low-power surveillance and multimedia sensor networks, the days of media transmission as a "downlink" experience (e.g., TV broadcast) only will soon be over.

Many interesting questions arise in distributed video management, storage and retrieval, and delivery. Basically, the key question is centred on: **How should video data be stored, managed and delivered to accommodate for the connectivity and bandwidth dynamics as well as variances of the networks, computing capabilities of the clients, and battery life of multiple, and mostly mobile, devices?**

There are many technical challenges, specifically:

- Given that popular content is replicated in multiple locations in a P2P network, can we design efficient coding and transmission algorithms that operate in a distributed manner enabling efficient downloads?
- How should the P2P system deal with the limited storage capacity since more and more video is added every day?
- How can the videos be indexed to enable efficient search in distributed video databases?
- How can we accommodate the varying bandwidths and provide reliable delivery to distributed devices?
- Where and how should we incorporate "transcoding"? Where in the multi-device system should computations such as video transcoding or channel coding for robust wireless transmission be performed? How can video transcoding be done efficiently?
- How can robust delivery be achieved over wireless LANs? How can real-time streaming media resources be handled in unicast and multicast environments?
- How can we monitor the quality of service (QoS) of the video streams delivered? Can we quantify the perceived visual quality or client experiences especially in the event of a paid service?

We will address most of the above questions in the subsequent sections, present some current solutions, and highlight future directions. We first present a general architecture for multi-device distributed computing system, and propose a P2P computing and networking framework, the design principles and system architecture of a video-adaptive and aware peer service platform, in Section 2. Section 3 is on distributed video storage and data management. We focus on three important aspects: multi-rate differential encoding (Section 3.1), distributed video coding (Section 3.2), and adaptive storage and caching management (Section 3.3). Automatic indexing and support for distributed search are addressed in Section 4. We show in Section 5 two important video delivery mechanisms: optimal delivery with dynamic transcoding (Section 5.1) to adapt content to network connectivity and client resources, and robust video unicast and multicast over wireless LANs (Section 5.2). We will touch upon the quality of service monitoring in Section 6. The sequence of the presentation is intended to give readers an overview of the layers and aspects of technical challenges lying ahead, as well as some promising directions of research to provide some answers to the aforementioned questions.

2. SYSTEM ARCHITECTURE

Figure 1 shows a potential scenario of a distributed video storage, management and delivery system. We shall use this to illustrate some interesting aspects of distributed video production, distributed video coding, distributed video processing and distributed video distribution. In this base model, every user/client can be a video producer/provider, intermediate service provider (e.g., such as of video transcoding), and video consumer.

To further illustrate the distributed video database scenario depicted in Figure 1, we use the following example: When a user with a handheld device requests a media file (or a life stream) from the distributed video database system by specifying desired attributed, the underlying P2P infrastructure (MAPS - Media Accelerated Peer Services) issues a distributed query, and returns a list of possible downloads together with their associated transmission and local decoding costs as well as their compression format description such as standard MPEG-1, high-quality MPEG-2, and low bitrate MPEG-4. The local P2P client aggregates the return list with respect to video quality. It also extends the list by possible lower video formats, which could be achieved by transcoding, and which may be more adequate for the client device. From this list the user selects the desired (and feasible) quality of the video. If the version of the video the user selects already exists in the network, the file is streamed to its destination; if not, the P2P infrastructure transcodes the content. For instance, suppose the user requests an MPEG-4 version of a video clip that exists only in MPEG-2 format, possibly at a different rate and resolution. After performing the search, the P2P system discovers the MPEG-2 version of the requested clip and returns the cost associated with streaming it to the client and decoding it there. In this scenario, it is, firstly, costly to transfer the MPEG-2 sequence over the wireless link and, secondly, the handheld device does not have the computational power to decode and render the sequence in real time. Moreover the device has limited battery life and should not perform transcoding itself even if it is possible. Hence, the P2P infrastructure automatically assigns a peer to perform the transcoding based on a cost evaluation, so that only a low bitrate MPEG-4 version of the video is transferred over the wireless link to the user. This example demonstrates that the system not only takes into account storage capacity and network bandwidth, but also CPU power and battery capacity when needed.

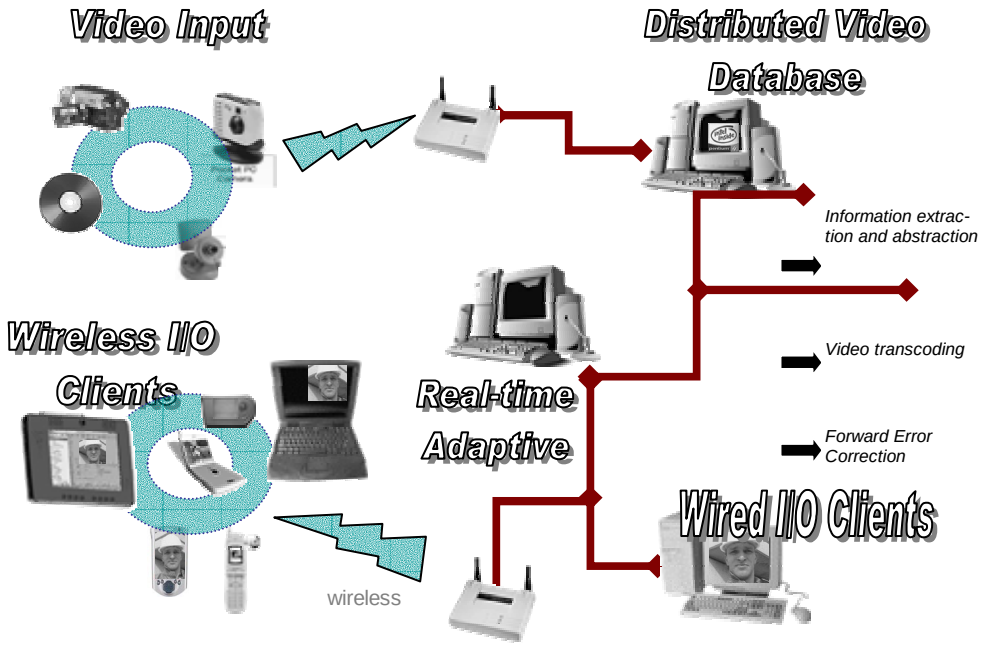


Figure 1: A distributed video management and delivery system. Every user can be a video producer/provider, intermediate service provider (e.g., such as of transcoding), and video consumer.

Our proposed system architecture to enable scenarios as described above is depicted in Figure 2. Existing P2P infrastructures are extended with several modules. These modules are collated into the **Media Accelerating Peer Services** (**MAPS**) [14]. In detail, the software layers in the system can be classified as follows, from bottom to top:

1. The traditional underlying operating system and related services, viz. sockets, RPC, DCOM, etc.
2. The P2P service layer providing basic functionality such as peer discovery, user and group management, name resolution, and delivery primitives. Examples of systems providing some or all of these services are Gnutella, FreeNet [4], and JXTA [9]. MAPS extends this layer with a plug-in architecture for customization of the basic operations of search and delivery. MAPS currently provides two extensions: a module enabling real-time media streaming (via RTP over UDP) and a module for enhanced search capabilities (via distributed SQL).
3. The MAPS video support modules, which use the underlying service layer for network and resource access and which provide transformation and computation services on data obtained from/sent to the layer below. Examples include a support module for MPEG video that provides transcoding support, and a module that automatically analyzes images, videos and music files on each peer to provide better search capabilities.
4. The application layer.

Similar to other P2P environments, a key characteristic of the architecture is the transparency of the physical location of a resource. That is, the location of data, services, and other resources need not be determined explicitly by applications using the provided services. Instead, data and services can

exist anywhere in the network, and the system handles their discovery and delivery to the requesting application.

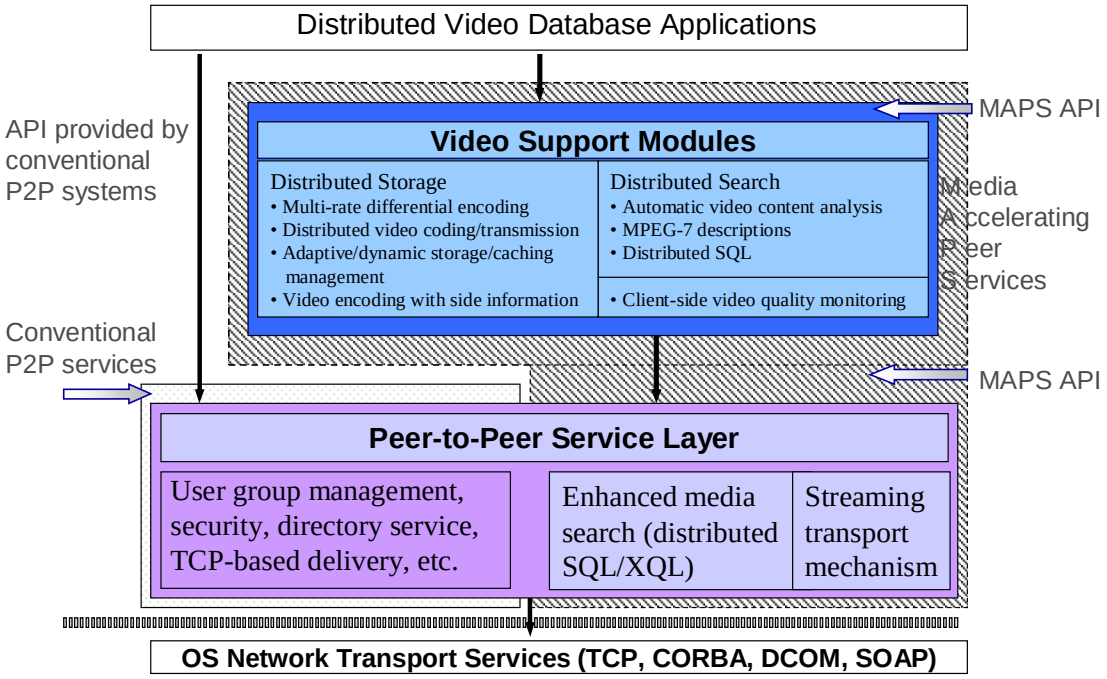


Figure 2: System architecture.

3. DISTRIBUTED STORAGE

The computing devices in a P2P network differ significantly in their computing power and link bandwidth capabilities. This heterogeneity must be adequately considered by advanced media archiving techniques, so that the content can be seamlessly served to the end-clients. Section 3.1 explains how multiple versions of the same video at different bitrates can be created and stored efficiently. It is followed by the VISDOM framework [18] – a fully distributed video coding and transmission scheme. VISDOM exploits that content can be replicated in disparate locations in order to provide load balancing of servers and fault tolerance. By nature, its transmission scheme falls semantically under Section 5; however, it is presented here together with the coding scheme for clarity. Finally, Section 3.3 addresses how individual peers can deal with their limited storage capacity in a distributed fashion.

3.1 MULTI-RATE DIFFERENTIAL ENCODING

In the P2P scenario of a distributed video database, video data has to be encoded and stored to accommodate for the different connectivity and bandwidth of the network, and computing capabilities of the clients. As described in the previous section, transcoding is the principal means through which video content is adapted to the client specifications. By default, our P2P system keeps the transcoded content for future retrieval since transcoding is a very expensive operation; however, storage is also a precious resource. Therefore, Section 3.1.2 describes our approach of saving storage space by differentially encoding the transcoded bit stream of the

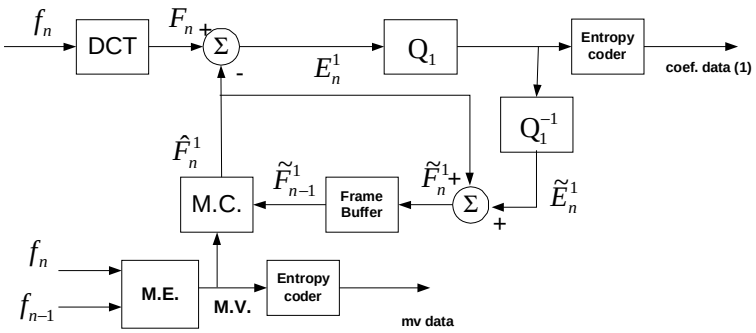
video clip with respect to the compressed bit stream of the same clip, but at a different bit rate.

This computational trade-off can also be done at encoding time. Instead of encoding a clip at a single rate and transcoding it at other rates at later times, it is possible to simultaneously encode it at different rates that are likely to be targeted rates by the users. Compared to transcoding, this encoding is better from a rate-distortion point of view since it avoids the successive re-quantization done in transcoding. Although the amount of storage gets increased, it is possible to combine the multi-rate encoder with the differential encoder mentioned above to compensate for this increase.

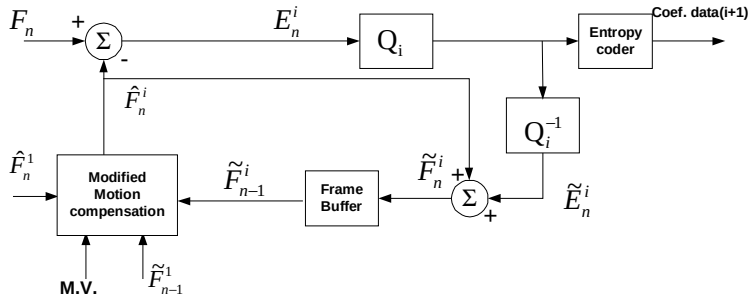
3.1.1 MULTI-RATE ENCODING OF VIDEO

Figure 3 shows the block diagrams of the multi-rate encoder we have developed [27]. The encoder is used to generate N streams at different rates from the same video sequence. For each frame f_n, the encoder computes the compressed data for all streams. The first stream is the reference stream, and the others are dependent streams. There are three main characteristics to this encoder. First, most of the processing is done in the DCT domain. This avoids going back and forth from the image space to the DCT space and therefore can significantly reduce memory operations. Second, motion estimation is done once. As this is the most computationally heavy block of the encoder, this significantly reduces the computational load. Finally, with this architecture, motion compensation has to be performed in the DCT domain.

Fast algorithms for motion compensation in the DCT domain (MC-DCT) can be used within our encoder. Also, since we are sequentially encoding a video sequence at different rates, we have the possibility to re-use data that was computed for the reference stream to reduce computation, and also determine how much loss we incur if no motion compensation or only part of it is performed on the dependent stream. The adaptation of the motion compensation is done on a macro block level and three modes can be used. The first of these modes is the usual mode for which the previously reconstructed frame of the same stream is motion compensated to generate the current predicted frame. In the second mode, the motion compensation is computed on the difference between the $(n-1)^{\text{th}}$ reconstructed frames of the current dependent stream and the reference stream. Typically, this difference will be small which, in the DCT domain, translates by a higher number of small or zero coefficients. An appropriate implementation of the MC-DCT can then take advantage of this by not computing the motion compensated DCT coefficients where we expect them to be zero or small. This can be determined by using the quantized residuals of the reference stream ($\tilde{E}_n^1$). Of course, if a DCT coefficient of the motion compensated frame is not correctly computed at the encoder, it will contribute to a drift (mismatch) error at the decoding time. Finally, in the third mode, we simply use the data from the motion compensated frame of the reference stream. This is more appropriate for B frames since the mismatch errors that are introduced do not propagate to other frames.



A) Encoder for the reference stream



B) Encoder for the dependent streams

Figure 3 Block diagram of multi-rate encoder.

3.1.2 DIFFERENTIAL ENCODING OF DCT COEFFICIENTS

Figure 4 shows a high level diagram of the differential encoder. We assume that a video sequence is simultaneously encoded at multiple rates or quality levels to generate *n* independent standard compliant (e.g., MPEG-2) video streams. This can be done using the encoder described in Section 3.1.1. One of the video streams is chosen to be the reference stream, in this example, the i^{th} stream, and the other single layer streams are encoded **without loss** with respect to that reference stream to form differentially encoded streams. The output of the differential video encoder is therefore a reference stream, which is a standard compliant video stream, and a number of differentially encoded streams.

As illustrated in the bottom of Figure 4, those differential bit streams can then be used to reconstruct without loss the single layer video streams when the reference stream is available. We emphasize that the video stream generated by the differential decoder is one of the streams generated by the multi-rate encoder, and therefore, it has the rate-distortion characteristics of the single layer encoder used to generate it. This attribute differentiates this approach from scalable codecs and transcoders.

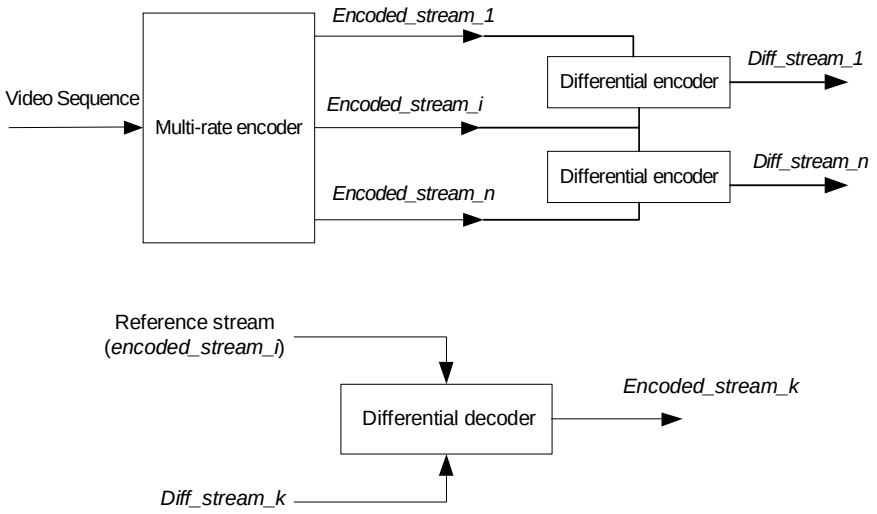


Figure 4: High-level representation of the differential video encoder (top) and decoder (bottom).

The differential lossless encoder we have developed works at the level of the quantized DCT coefficients. Basically, for a given block of 8x8 pixels, the values of the DCT coefficients of the reference stream are used to predict the values of the DCT coefficients of the target stream. The difference between the predicted and actual values of the target stream DCT coefficients is **losslessly** encoded using an entropy encoder. Coefficients from inter and intra coded blocks are treated differently as we briefly explain next. For intra coded blocks, the predicted coefficient values of the target stream are simply the quantized coefficient values of the reference stream which are re-quantized with the quantization parameters used for the target stream. The same simple prediction mode could be used for inter coded blocks. For inter coded blocks, however, the DCT coefficients encode the difference between the video frame and the motion compensated prediction. As the motion compensated data is not the same at different bit rates, the motion compensated difference will vary with different bit streams. The inter coefficient predictor we use takes into account the mismatch between the motion compensated frame different of both streams. It adds to the quantized coefficient values of the reference stream the difference between the motion compensated coefficients of the reference and target streams.

Using the multi-rate encoder described previously, we encoded video sequences without rate control at multiple bit rates. Table 1 shows the results for the sequence Football (CIF, 352x240, 30fps) encoded at 5 different rates from 750Kbits/sec to 2Mbits/sec using a quantization parameter value ranging from 7 to 17. The results show that the differential encoding of the coefficients allows saving 30% of the storage space when compared to independently storing the five bit streams at a cost of slightly increasing the CPU load (required to perform differential decoding). The results also show that the performance is independent of the choice of the reference stream. In this specific case, the reference stream was alternatively chosen to be the stream encoded at 1.5 Mbits/sec, 1.2 Mbits/sec and 1 Mbits/sec with very small difference in performance.

Quant. scale value	Bit rate of differential and reference streams (bits/sec)			Bit rate of independently coded streams
	Ref Q=9	**Ref Q=11**	**Ref Q=13**	
7	1,035,357	1,200,314	1,336,536	1,957,853
9	1,492,394	785,261	926,537	1,492,394
11	670,705	1,119,814	625,793	1,193,814
13	626,852	536,158	992,703	992,703
17	435,082	586,105	481,483	741,834
Total bit rate	**4,260,390**	**4,301,652**	**4,363,052**	**6,378,599**
Savings	**33.2%**	**32.6%**	**31.6%**	

Table 1: Bitrate of differentially encoded streams and storage savings with respect to independent coding of the streams for the CIF sequence Football.

3.2 DISTRIBUTED VIDEO CODING AND TRANSMISSION

Storage of popular content at a single node in a distributed video database can lead to fragility in the system due to the "single point of failure" phenomenon. Situations when a large number of clients attempt to access data from this single popular storage, can lead to excessive loading and consequently content node failure.

A simple and reasonable, but not necessary efficient and always sufficient approach to combat the above problem and to ensure reliability is to replicate or mirror the data at multiple locations, as it usually happens in P2P networks. Therefore, we have developed the VISDOM framework [18], which uses algorithms for efficient coding and transmission of video content that are distributed in nature. VISDOM enables download of real-time content in parallel from multiple sources, with no need for any communication between them, by leveraging the latest advances in the fields of video compression, distributed source coding theory and networking. Although, VISDOM is distributed, there is no loss in performance with respect to an equivalent centralized system. It also leads to load balancing of servers and fault tolerance. That is, if there is a link/server outage, there will be graceful quality degradation at the client end (the decoded quality decreases smoothly with the reduction in the number of received packets).

The VISDOM approach seeks to deliver a nearly *constant perceptual quality* to the client. Given that the receiver is connected to multiple sources, VISDOM accomplishes this task while *minimizing the net aggregate bandwidth* used from all the sources put together. The striking aspect of VISDOM approach is that the net aggregate bandwidth used in VISDOM is the same as that would be used if content were to be streamed from a single source. In this sense, there is no loss in performance with respect to a centralized system even though the VISDOM system is distributed in nature. In fact, the natural diversity effect that arises because of streaming from multiple sources provides load balancing and fault tolerance automatically. The three main aspects of VISDOM namely video compression, distributed coding and networking are operating in synergism leading to a satisfying end user experience. A block diagram of the system is shown in Figure 5.

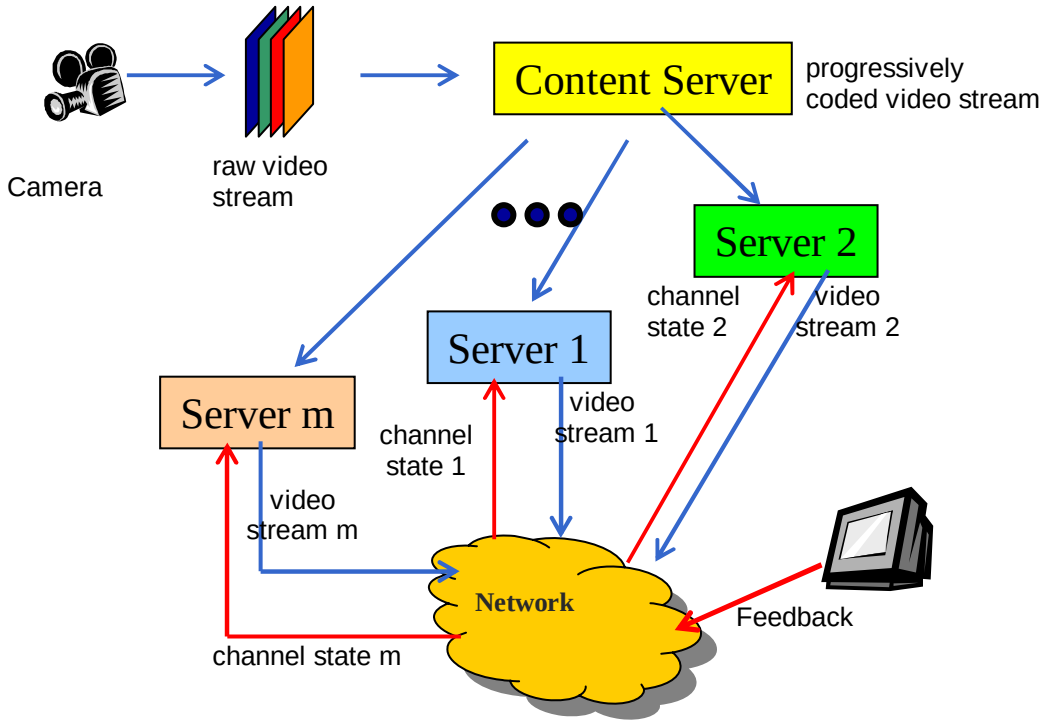


Figure 5: VISDOM System Block Diagram.

Video Compression: We use a multi-resolution encoded video bit-stream, such as H.263+ or MPEG4-FGS [6]. The advantage of using such a bit-stream is that a single stream can be served over a dynamic network with varying bandwidths and/or cater to diverse receivers.

Robust (Distributed) Coding: A multi-resolution bit stream, as described above, is sensitive to the position of packet loss, i.e., a loss of more "important" bits early on in the bit-stream can render the remaining bits useless from the point of view of decoding. Such a prioritized bit stream is inherently mismatched to the Internet, which offers equal treatment to all packets. So, a transcoding mechanism to transform a prioritized bit stream into an un-prioritized one, in which the decoded quality is only a function of the number of packets received, is required. The MDFEC (Multiple Descriptions through Forward Error Correction codes) algorithm offers a computationally efficient solution to this problem by trading off bandwidth for quality [21]. The MDFEC algorithm is discussed in more details in Section 5.2. Essentially, the algorithm uses Forward Error Correction (FEC)[3] codes of varying strengths to differentially protect (the more important parts of the bit-stream are protected more than the less important parts) the video bit-stream. The parameters for encoding are decided based on both the state of the transmission channel as well as the rate-distortion characteristics of the source content. Given these two factors, these parameters are optimized so as to give the best end-to-end performance using a computationally efficient, Lagrangian approach based algorithm. The computational ease of the approach enables dynamic adaptation to changes in network/content characteristics.

[3] FECs introduce redundancy at the encoder by converting **k** message symbols into **n>k** code symbols. This enables the decoder to recover the input **k** message symbols correctly even if there are some losses in transmission and only some **m** $\geq$ **k** code symbols are received.

Figure 6 illustrates the operation of the MDFEC framework. The source content that has been encoded in the multi-resolution format in N layers is unequally protected using channel codes of varying strength. The first layer is protected using an n=N, k = 1 channel code, the second layer is protected using n=N, k =2 code and so on. Thus, with the reception of any one packet, quality commensurate with the first multi-resolution layer is decoded and with the reception of any two packets quality commensurate with the first two layers is obtained and so on. Unlike the multi-resolution bit-stream where the decoded quality is sensitive to the position of packet loss, in the MDFEC stream the decoded quality is dependent only on the number of losses.

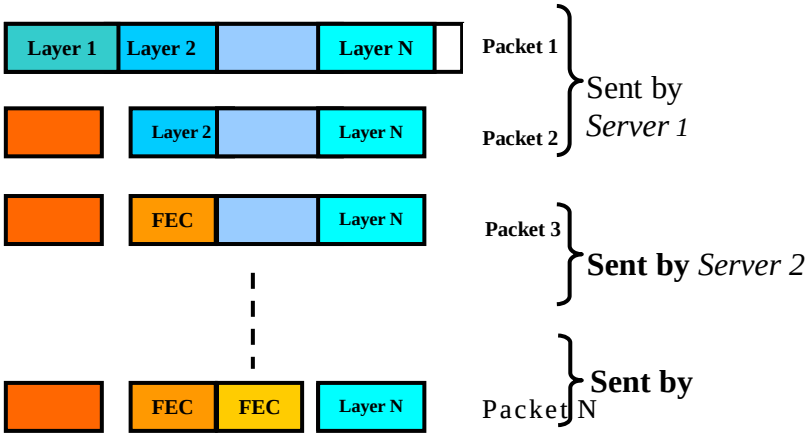


Figure 6: MDFEC: The arrangement of FECs and video layers enables successful decoding of layer 1 with reception of any one packet and layers 1 and 2 with the reception of any two packets and so on.

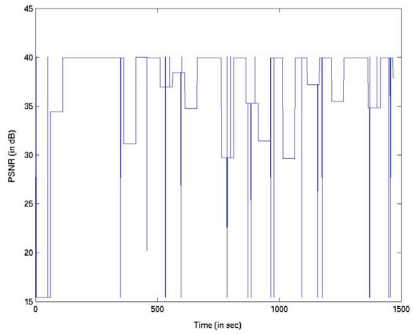
While MDFEC was originally designed for a single channel between the sender and the receiver, an interesting property of this algorithm is that it is inherently distributed. There is a notion of an "aggregate" channel between the client and all the servers, and in this case the MDFEC algorithm is applied to this aggregate channel. How this is achieved in a manner that the servers do not need to communicate with each other is explained in the next paragraph.

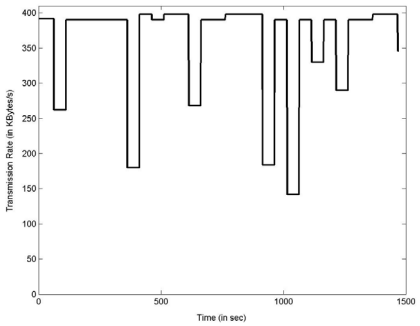
Networking: Streaming real-time content from multiple senders to the client in a scalable way without any communication between the various remote servers is accomplished through the client, which acts as the synchronization "master" coordinating all the servers. Thus the client is the controlling agent in this framework ensuring that all the servers operate in harmony.

Further, the same idea is used in ensuring the distributed MDFEC encoding at the servers. The client having access to the aggregate channel state conveys that to all the servers. The individual servers use this information along with the content rate-distortion characteristics to run the MDFEC algorithm independently and hence come up with identical encoding strategies. Then based on a pre-decided allocation, which is also conveyed by the client to all the servers, each of the servers send the pre-assigned portion of

the MDFEC bit-stream. See **Figure 6** for a sample allocation of packets. The dynamic, adaptive nature of MDFEC allows us to run the VISDOM algorithm over coarse time scales to adapt to the varying traffic conditions in the network. The system is fault-tolerant since if one of the servers goes down, the graceful quality degradation of MDFEC with the number of packets as well as the dynamic nature of the adaptation absorbs these variations lead to a nearly imperceptible end-user experience.

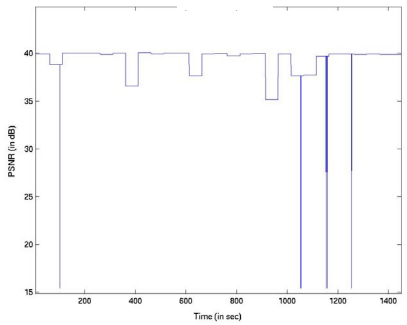
Simulation Results: The validity of the VISDOM framework was confirmed in an Internet setting where a content client was placed at University of California, Berkeley and three content servers at University of Illinois, Urbana-Champaign, Rice University, Houston, Texas and EPFL (Ecole Polytechnique Federale de Lausanne) Lausanne, Switzerland respectively. The Football video sequence was streamed to the client in three settings: single server, two servers and three servers. Figure 7 (a),(c),(e) show the delivered picture quality measured in PSNR (dB) as a function of time for the single, two and three server case respectively. Figure 7 (b), (d), (f) show the total transmission rate measured in Kbytes/sec as a function of time for the single, two and three server case. We notice from these plots that for a nearly the same transmission rate, the delivered quality becomes more and more smooth as the number of servers is increased. This highlights the diversity effect of using multiple servers.

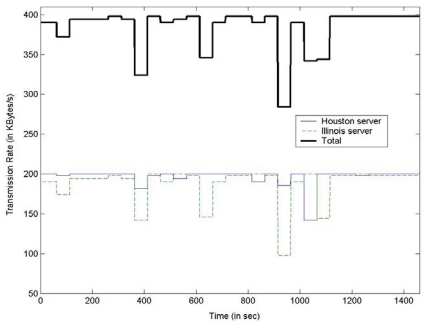




(c)

(d)





(e) (f)

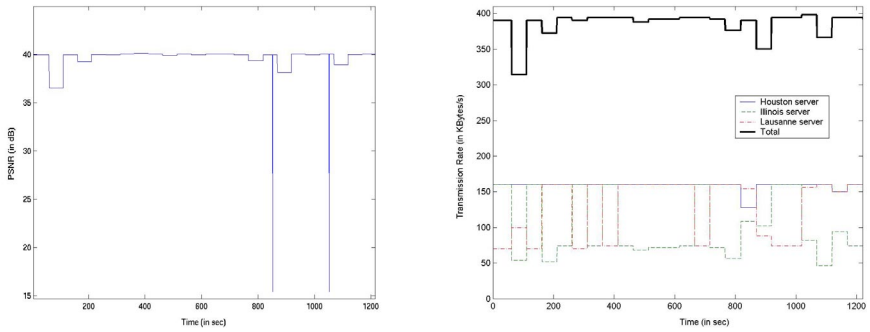


Figure 7: Plots (a),(c),(e) show the delivered quality for the Football video sequence (measured in PSNR (dB)) as a function of time for one, two and three servers respectively. Plots (b),(d),(f) show the total transmission rate as a function of time for one, two and three servers respectively. Even though the transmission rate is nearly the same for the three cases, the delivered quality becomes more and more smooth as the number of content servers is increased.

3.3 ADAPTIVE/DYNAMIC STORAGE AND CACHING MANAGEMENT

The storage or caching operations in current database systems—centralized or distributed—are performed on a binary level; either some data is stored/cached or it is deleted. There is no concept of trading in gradual degradation for storage efficiency. For multimedia data such as video "gradual" states of data storage are much more suitable. Media data is bulky but possesses the unique property that it is *malleable*.

The MAPS Enhanced Media Storage module offers a novel cache management strategy for media data by going beyond binary caching decisions. It exploits transcoding of media data to achieve a higher utilization of cache space. Less frequently accessed media data is cached at a lower quality level in order to consume less cache space. Consequently, more data will fit into the cache, which in turn will lead to a higher cache hit rate.

For every cache entry, a recall value is maintained indicating its popularity. An entry's recall value is calculated based on its access pattern, using heuristics with the following properties:

- The more often a media cache entry is accessed, the larger its recall value.
- The recall value of a cache entry that is never accessed approaches zero over time, and will thus be deleted if the cache runs out of space.
- Often-recalled data will have a higher recall value than newly-accessed data.

The cache management system uses entries' recall values to rank the media data in descending order (see Figure 8) and to assign class numbers to them according to a local mapping table (see Table 2). The class number of a data entry determines its maximum allowed bitrate. If the compression ratio of a cached entry is higher than its allowed bit rate, the entry is added to a background transcoding queue.

Class	From /To
0	$[0, N_0]$
1	$]N_0, N_1]$
2	$]N_1, N_2]$
…	…
N	$]N_{N-1}, N_N]$
N+1	$] N_N, N_{N+1}]$

Table 2: Mapping data to data classes based on its position in the cache. The position is determined by the data's recall value.

Table 3 gives an example of a possible mapping from class numbers to target compression qualities.

Class	**Video Target Quality**	**Audio Target Quality**
0	Unchanged original quality	Unchanged original quality
1	MPEG4 320x240 at 1.5 Mbits/s	MP3 128Kbits/s
2	MPEG4 320x240 at 700 Kbits /s	MP3 96Kbits/s
…	…	…
N	MPEG4 160x120 at 100 Kbits/s	MP3 mono 11.2 KHz at 16 Kbits/s

Table 3: Mapping data class numbers to best allowed compression quality.

The amount of transcoding possible at each cache node in a distributed storage system depends on the computational capabilities of the node as well as its current load. Thus, different transcoding tables and cache-partitioning tables should be available at each node for different workloads as well as for nodes with different performance. For instance, a very slow node should not use any transcoding, but instead support only a binary caching decision. Consequently, there would be only one data class assigned to the whole cache. A powerful node, on the other hand, can use a very fine granularity scale transcoding scheme.

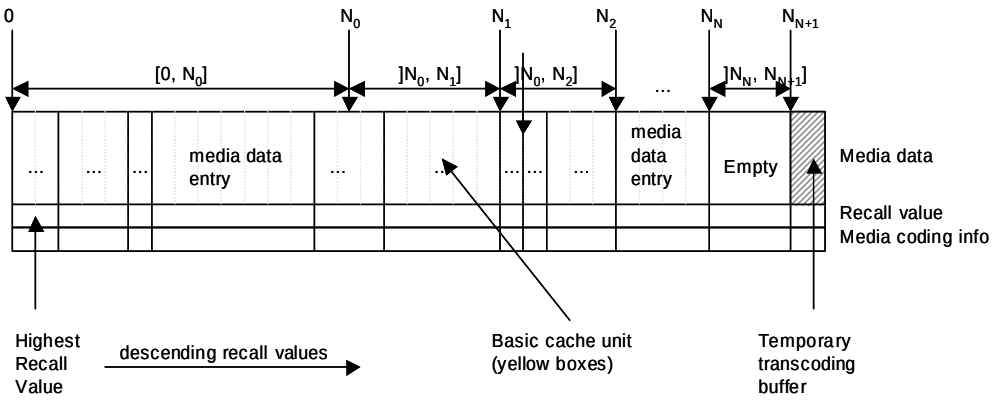


Figure 8: Media data ordering in the adaptive/dynamic storage and caching management system based on recall values. The total cache size is N_{N+1}.

4. DISTRIBUTED SEARCH

Video — a composition of moving pictures and audio clips — possess many more dimensions of searchable attributes than text or web documents. Consequently, a video-centric P2P system must allow users to search for video based on a large and diverse set of syntactic and semantic attributes. While costly manual annotations may exist in special — but rare — cases, a video-catering peer service must be able to extract information automatically from media content to assist search, browsing, and indexing.

MAPS Distributed Search subsystem therefore provides

- A toolbox to extract syntactic and semantic attributes automatically from media (including MPEG-7 audio and video descriptors), and
- A powerful search mechanism beyond hash keys, file names and keywords.

Enhanced search specification: For search, the MAPS Media Search subsystem allows each node to have different search capabilities. Besides standard search methods based on keyword matches, SQL queries are used. If a node does not support SQL queries, it either ignores the request or maps it automatically to a keyword-based search.

Automatic extraction of Meta-descriptions: The MAPS Distributed Search subsystem exploits two sources for acquiring Meta information:

- Meta information encoded in the media files or the media files accompanying description files,
- Information derived by automatically analyzing the media content.

The most prominent examples of standardized meta information extracted by MAPS from media files are ID3 tags [11] from audio clips, e.g., describing album, title, and lyric information, Exif (Exchangeable Image File) tags from images captured with digital cameras, e.g., describing camera make and model [8], and general information such as frame size, bitrate, duration, and, if available, the date and time of recording [2]. In future, MPEG-7 media descriptions will be parsed, too.

Furthermore, by extracting semantic information automatically from media, our MAPS Video Content Analysis module supports search beyond standard Meta information. Currently, the MAPS Video Content Analysis module incorporates several types of automatic information extraction, for example, visible text in images and video (see Figure 9) [13], faces and their positions, key frames for efficient video summarization [12][26], and colour signatures for similarity search (Figure 10). The list of media content analysis tools is not exhaustive as automatic content analysis algorithms are ongoing research topics [23]. Many of the state-of-the-art tools can be plugged into the MAPS Video Content Analysis module.

Figure 9: Example of text extraction in videos.

Figure 10: Sample results of color-based similarity search. Similar videos are showed in small icons surrounding the displaying video in the center.

5. DISTRIBUTED DELIVERY MECHANISMS

Once the requested video has been located, the P2P system optimises its delivery. In conventional file sharing systems media data is typically delivered asynchronously using standard network routing methods, and reliable network protocols such as TCP. This, however, may not be the best solution in situations when **real-time** streaming is desired and/or video transcoding is required. In the following sections we describe mechanisms that facilitate the delivery of video data over heterogeneous networks to heterogeneous peers. Section 5.1 describes our novel scheme of dynamic transcoding along a network path, while Section 5.2 focuses on robust video unicast and multicast in wireless LANs.

5.1 OPTIMAL DELIVERY WITH DYNAMIC TRANSCODING

In a network in which data is shared and proliferated, typically multiple copies of a given file will exist throughout the system. When the data of interest is multimedia, in general, and video in particular, a larger array of possibilities arises. For example, different versions of the same video are likely to exist throughout the network at multiple bit rates and resolutions. Furthermore, content can be transcoded or transformed, even on the fly, e.g., to generate versions at new bit rates and resolutions or to incorporate additional redundancy for error resiliency over noisy channels—in other words, to make the content adaptive to client device capabilities, storage capacities, and bandwidth constraints.

In such a scenario, obtaining an exact version of data from a far-away source may be a poorer choice than obtaining a different version of the data from a near-by source and transcoding it into the required version. The "best" means of getting a version of an object (e.g., a video or audio clip) on the network can be estimated by a metric that attempts to rank and prioritize available paths to an object. In this context, a path consists of both a conventional network route and an optional sequence of transformations throughout the network that are necessary to deliver the object to the destination node so as to satisfy the requester's constraints. We will define a cost function to measure the aggregate computational complexity and transmission overhead required to deliver a given object. MAPS attempts to minimize the cost function over the set of possible paths to find the most efficient sequence of operations to employ.

To support cost-function-based optimization, nodes exchange periodic link and node state information through intermittent broadcast updates and as additional information appended to query response packets. Node state information includes an indication of the node's level of activity (based on current reservations and any queued best-effort requests), its transcoding capabilities (reflective of computational power), and an advertised cost associated with each transformation type. Each node assigns its own values to transformations' costs. The cost function may therefore vary from node to node, e.g., a node may penalize computation more than storage depending on its capabilities, or vice versa. Since transcoding is potentially a lengthy operation, MAPS uses a resource-reservation type approach, so that nodes do not become overloaded with concurrent requests.

The cost of the most efficient sequence of operations to obtain a version of an object O satisfying some set of required parameter constraints P at a given node N_D can be expressed (recursively) as follows:

$$c(N_D, O, P) = \min_j \min_k \{c(N_k, O, P_j) + c_t(N_k, N_D, O, P_j) + c_x(N_D, O, P, P_j)\},$$

where $\{ N_k \}$ is the set of "neighbouring" nodes (viz. peer nodes from which the object may be delivered to N_D, including itself), $\{ P_j \}$ is the set of possible constraints for which a version of the object can be delivered to the local node N_D from its neighbouring nodes, $c_t(N_k, N_D, O, P_j)$ is the cost of transmission of the intermediate object from the neighbouring node N_k to the local node N_D, and $c_x(N_D, O, P, P_j)$ is the cost of locally converting a version of O possessing initial parameters P_j to one that satisfies the node's target constraints P.

For example, if a node N_D has a version of the object that satisfies its target constraints P, then the cost of retrieving it, $c(N_D, R, P)$, is zero; $c_t(N_D, N_D, R, P)$ is zero since no transmission is required, and $c_x(N_D, R, P, P)$ is zero since $P=P_j$ making transcoding unnecessary. On the other hand, if N_D does not have a version of the resource that satisfies the constraints P, then the cost $c(N_D, R, P)$ can include a transcoding cost at node N_D, the cost of transmission from a peer, as well as the cost of any similar operations (i.e. other transcoding and transmissions) required on its neighbours to obtain the needed object. The best way to obtain a media object can be determined by minimizing the preceding cost function.

Figure 11 shows how path selection is determined for an example where a node Peer A requests an MPEG-4 version of a stream ("example") from the network. The requesting node constructs a directed graph representing the relationships between participating nodes (viz. nodes containing the requested stream, and nodes capable of transforming the content to the needed form), and their advertised costs for carrying out any needed transformations. Transmission and transcoding costs are expressed as edge weights, while peer nodes and content transformations are mapped to vertices. While Peer B transcodes "example" from MPEG-2 to MPEG-4, a virtual vertex (Peer B) is created.

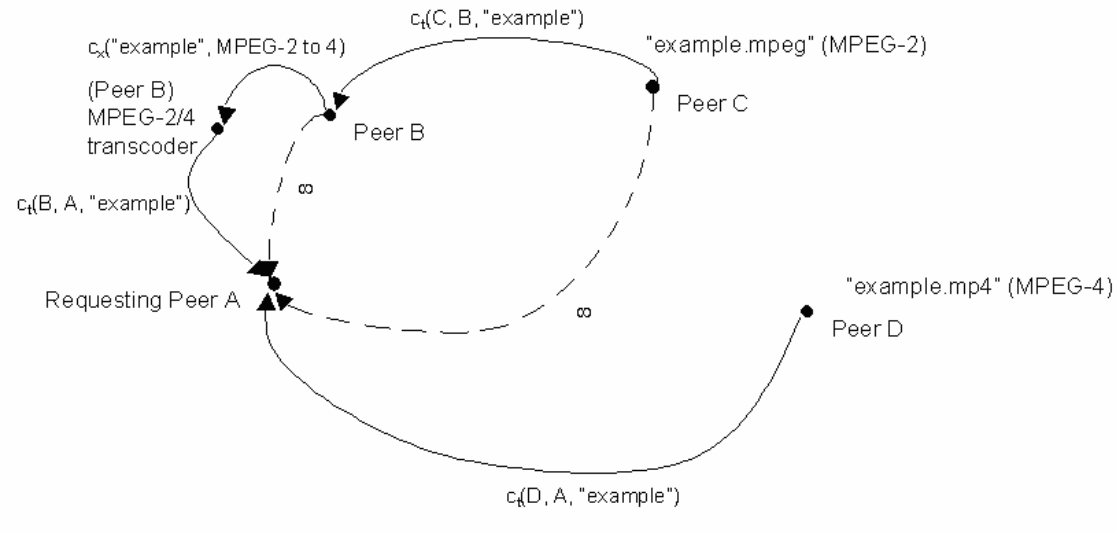


Figure 11: Example of transcoding/path selection graph construction for delivery of a video stream "example" to a client Peer A possessing only an MPEG-4 decoder. The content is available in MPEG-2 format from Peer C, or directly as an MPEG-4 sequence from Peer D. The requesting node compares the cost of obtaining the clip along several possible paths: directly from D, which involves a cost $c_t(D, A)$ of transmission and from C to B to A, which involves the cost of transmission, $c_t(C, B) + c_t(B, A)$, plus the cost of transcoding at B, c_x(*"example", MPEG-2 to MPEG-4)*. Because Peer A can only decode MPEG-4, it is useless to get an MPEG-2 version of the object to Peer A. In this case, which cannot satisfy the required constraints, some edges are marked with infinite cost.

In reality, minimizing the proceeding cost function is computationally infeasible. Each peer node in the network can potentially be used to process dif-

ferent versions of the object. The graph of the problem will become more complicated if continuous transcoding of different bitrates is allowed. In practice, we simplify the formulation by some assumptions. We trade the optimality of the cost with the computational complexity of evaluating the formulation. To find a computationally feasible solution to the problem of transcoding and routing of multimedia data we assume that the transcoding, if needed, is performed firstly in a single peer on the network, and, secondly, in only one peer. Thus, for a network consisting of n peers, there are only n possible transcoding positions. For each transcoding position N_k, we calculate the total cost of taking the route as

$$c^k(N_D, O, P) = \min_j \{c(N_k, O, P_j) + c_x(N_k, O, P, P_j)\} + c_t(N_k, N_D, O, P),$$

where the delivery cost from Node N_k to Node N_D $C_t(\ldots)$ can be found by using well-known network routing optimisation via dynamic programming. In other words, we simplify the optimization problem to two network routing problems – one from the source to the transcoder and the other one from the transcoder to the destination.

After we have the cost of transcoding at each note, we minimize the overall cost by:

$$c(N_D, O, P) = \min_k \{c^k(N_D, O, P)\},$$

i.e., after solving at most n dynamic programming problems, the optimal path can be determined. This approach is possible in the situations where N is not restrictively large. Note that most of the times only a limited number of nodes in the peer network are able to actually handle transcoding efficiently, and, therefore, the actual number of transcoding nodes is significantly smaller than N.

The path MAPS uses to deliver a video is determined by minimizing the preceding cost function. One point to note here is that although "cost" is advertised based on a local perspective by peers, the cumulative cost of an operation is inherently global in nature as it spans multiple nodes. The assumption is that peers behave "fairly," by trading individual for global benefits.

5.2 ROBUST VIDEO UNICAST AND MULTICAST IN WIRELESS LANS

With wireless networks gaining prominence and acceptance, especially the LANs based on the IEEE 802.11 standards, it is foreseeable that wireless streaming of audio/video will be a critical part of the P2P infrastructure. Therefore, we investigate in detail the specific features of real-time video streaming in wireless LANs.

There are two major challenges for video streaming over wireless LANs: (1) fluctuations in channel quality, and (2) higher bit/packet error rates compared with wired links. In addition to wireless channel-related challenges, when there are multiple clients, we have the problem of heterogeneity among receivers, since each user will have different channel conditions, power limitations, processing capabilities, etc., and only limited feedback channel capabilities. (This is the case when, for example, multiple peers request a resource that is a real-time streaming.) In the following we analyze both the single-user and multi-user cases in detail, and propose practical solutions to address the afore-mentioned challenges.

There has been substantial amount of prior work in the area of video streaming. The single user scenario has been addressed by a number of researchers (see, e.g., [1] [20] [3] and references therein). In the following we introduce a new method that combines the reliability and fixed delay advantages of Forward Error Control (FEC) coding with the bandwidth-conserving channel-adaptive properties of Automatic Repeat ReQuest (ARQ) protocol. In a multicast scenario, to tackle the problem of heterogeneity and to ensure graceful quality degradation the use of multi-resolution based scalable bit streams has been previously suggested in [19][25]. However, such a bit stream is sensitive to the position of packet loss, i.e., the received quality is a function of which packets are erased. To overcome this problem, the Priority Encoding Transmission scheme was proposed in [1], which allows for different resolution layers to be protected by different channel codes based on their importance. Substantial work has been done towards finding an algorithm for optimizing the amount of parity bits used to protect each resolution layer [20][21]. A near-optimal, O(N) (where N is the number of packets) complexity, algorithm was proposed in [21] to solve this problem.

5.2.1 802.11 wireless LAN as packet erasure channel

To address the problem of wireless video streaming we need to model the communication channel. The IEEE 802.11 Media Access Control (MAC)/Logical Link Control (LLC) and physical (PHY) layers represent two lower layers in the Open System Interconnect (OSI) reference model, i.e., the Data link and Physical layers. In real life, we do not have direct access to the Physical (or even MAC) layer. Further, most of the successful wireless networks adopt the Internet Protocol (IP) as a network layer simplifying the integration of wireless networks into the Internet networks. In this scenario, user applications see the wireless channel as an IP packet channel with erasures---much like the wired Ethernet. Therefore, in designing our algorithms (which run at the application layer) we model the wireless network channel as a packet erasure channel at the network layer level. There is a very close connection between the problem discussed in Section 3.2 and the problem we consider here. In fact having multiple unreliable servers streaming data to a client is essentially analogous to having a single server sending information over a packet erasure channel. It is not a surprise, therefore, that we use a similar MDFEC-based solution for streaming over wireless LANs.

In order to reliably communicate over packet erasure channels, it is necessary to exert some form of error control. Asynchronous communication protocols, such as ARQ, are reliable but have unbounded delay. In synchronous protocols, the data are transmitted with a bounded delay but generally not in a channel adaptive manner. To provide for some measure of reliability, Forward Error Control coding is employed (See also section 3.2). If the number of erased packets is less than the decoding threshold for the FEC code, the original data can be recovered perfectly. However, FEC techniques cannot guarantee that the receiver receives all the packets without error. Popular Reed-Solomon (RS) codes are described by two numbers **(n,k)**, where **n** is the length of the codeword and **k** is the number of data symbols in the codeword. Each symbol is drawn from a finite field of $\mathbf{2^s}$ elements, where **s** (we use **8**) is the number of bits to be represented in each symbol. The total number of words in the code equals $\mathbf{2^s - 1}$. RS codes can be used to correct errors, erasures, or both. Particularly efficient decoding algo-

rithms based on Vandermonde matrices [22] exist if only erasures are to be corrected. In this case, each parity symbol can correct any one missing data symbol. This means that we can recover the original codeword, and hence the original data, if at least **k** of the original **n** symbols are received.

5.2.2 Robust video unicast over wireless LANs

In a unicast scenario there is only one recipient of the video data. The cost function to be optimized is a reconstruction distortion subject to rate and delay constraints. The first approach to we describe is the purely FEC-based MDFEC algorithm [21], which assumes as input a scalable video bit stream. On the other hand, the Hybrid ARQ protocol [17], is a combination of FEC and ARQ techniques and does not assume a scalable video bit stream as input, and therefore is readily applicable to existing video content stored on DVDs and VCDs. The problem we address in this section is that of finding the parameters for source and channel coding schemes for a single server and a single client, to maximize the overall data quality (or, equivalently, minimize the distortion) subject to a communication delay constraint.

MDFEC

MDFEC is a transcoding mechanism to convert a prioritized multiresolution bit stream into a non-prioritized multiple description bit stream (see Figure 12) using efficient FEC codes.

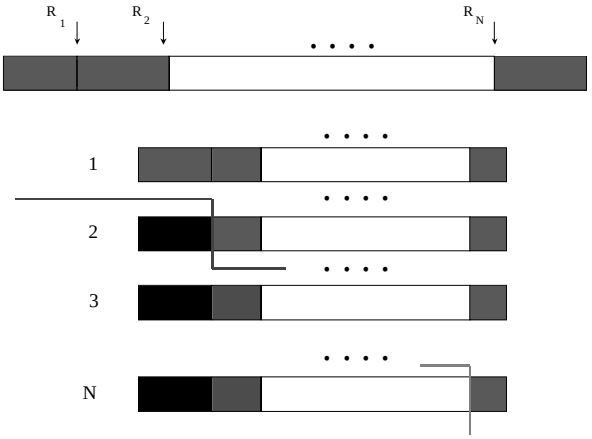


Figure 12: Mapping progressive stream to packets using MDFEC codes. Progressive stream (top) is partitioned by R_i and encoded with (N,i) codes. R_i corresponds to layer i in Figure 6.

Let **d** be an N-dimensional distortion vector (also called the distortion profile) where d_k reflects the distortion attained when k out of N packets are received. The progressive bit stream is marked at N different positions (that form N resolution layers), which correspond to achieving the distortion levels d_k as shown in Figure 12. The i^{th} resolution layer is split into i equal parts and an (N,i) RS code is applied to it to form the N packets as shown in Figure 12. Since every packet contains information from all the N resolution layers, they are of equal priority. The RS code ensures that the i^{th} resolution layer can be decoded on the reception of at least i packets. Since the distortion-rate function D(r) for a source is a one-to-one function of the rate r, finding

the N dimensional distortion vector **d** corresponds to finding the rate partition **R** = (R_1,...,R_N) of the multiresolution bit stream (see Figure 12). A fast, near-optimal algorithm, of complexity O(N) (where N is the number of packets), based on Lagrangian principles, to solve this problem is described in [21].

Hybrid ARQ

MDFEC method is an attractive solution but it requires progressive video input. Here we propose a way to combine the ARQ and FEC error control methods to improve the performance of unicast communications of single resolution video over packet erasure channels. Hybrid ARQ schemes have been extensively studied in the literature for various communication channels. We do not attempt to survey all of them (the reader is referred to [15] for a textbook treatment) but rather we propose a scheme that specifically addresses the problem of video streaming over 802.11 networks. The idea is illustrated in Figure 13. We start by splitting our multimedia data into ``packet groups,'' consisting of k packets each, and then, for each packet group, appending n-k RS parity packets to the group as in the FEC coding scheme described above. However, unlike in the pure FEC scheme, we initially send only the first k data packets to the receiver. Then transmitter starts sending parity packets until one of the following two events occur: either an acknowledgment from the receiver arrives, or the deadline for the transmission is reached. Once at least k packets are received intact, the receiver sends an acknowledgment. Once the acknowledgment is received, the transmitter continues with the next k data packets. One significant advantage of this algorithm is that it does not break down even when acknowledgments are lost. Instead, the transmitter simply assumes that more parity is needed.

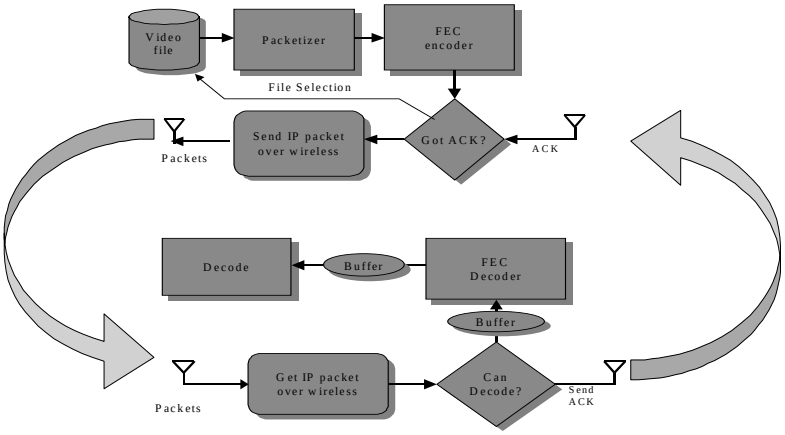


Figure 13: Hybrid ARQ algorithm data flow.

The Hybrid ARQ scheme is a general algorithm and can be adjusted to fit specific cases as is appropriate. For further details on Hybrid ARQ algorithm and its theoretical performance evaluation we refer to [17].

5.2.3 Robust Video multicast over wireless LANs

ARQ-based schemes are less appropriate for the multicast case for two reasons: ACK explosions and the requirement to retransmit different packets to all users. For significant packet loss rates, each user will require frequent packet replacement, and different users are most likely to require different packets. To respond to requests by multiple users, we may have to resend a

significant fraction of the original data even for small loss rates. However, for small multicast networks, the hybrid ARQ scheme can alleviate the problem of sending different correction packets to each user. Because each parity packet can replace any missing data packet, there is no need for each user to identify, which packet is missing. Instead, each user can simply transmit an acknowledgment when it receives enough parity to decode the transmitted data. When acknowledgment packets have been received from all known multicast users, the transmitter can move on to the next packet group.

An alternative attractive solution is based on the MDFEC approach. In order to address the multiuser setup we propose to minimize the following criterion: $\Delta(\mathbf{R}) = \max_i (E[d_i(\mathbf{R})] - E[d_i]_{min})$, where $\mathbf{R}$ is the rate partition as defined above, $E[d_i]_{min}$ is the minimum expected distortion for the i-th client, achieved by using the optimal coding scheme when it is the only client, and $E[d_i(\mathbf{R})]$ is the expected distortion for the particular coding scheme being used. Such an overall quality criterion is fair in the sense that it minimizes the maximum penalty that any client suffers. It then can be shown that the problem of finding the optimal partitions $\mathbf{R}$ is the problem of minimizing the intersection of convex surfaces [17]. For example, in a two-client case our solution is to use the single user version of MDFEC algorithm for each client, and then find the lowest intersection of two distortion-rate surfaces (convex) as the point of equal disappointment. Similarly, for more than two users we can perform the outlined optimization pair wise and select the lowest point among all possible pairs.

To actually find the rate partition that maximizes the overall quality criterion, we use a simplex optimization method although other standard techniques may be used. While the computational complexity of the simplex algorithm may be high, we do not expect that the algorithm will be run for a very large number of users. This is due to the fact that the algorithm runs at each access point in the wireless LAN network and we do not expect many users present in vicinity of any access point.

Table 4 shows the difference in the MSE penalty ($\Delta(\mathbf{R})$) for users with packet erasure channels having different probabilities p_i for the case when the parameters were optimized for the worst channel case and for the proposed minimax criterion. In the first case the user 1 suffers significant quality degradation (compared to a system optimized for user 1 only), while in the second case users 1 and 3 "share" the largest degradation in MSE.

Table 4: Comparison of penalty in distortion for three users for MDFEC designed for the worst-case channel, and for MDFEC designed for the minimax cost.

user	p_i	$E[d_i]_{min}$	Worst case MSE penalty	Minimax MSE penalty
1	0.113	137.5	32.3	7.0
2	0.156	162.9	14.9	0.3
3	0.197	193.5	0	7.0

To summarize this section, we point out that media streaming over wireless networks in a single or multiple user scenario poses specific problems to P2P delivery mechanisms that can be efficiently addressed by channel cod-

ing for packet erasures (FEC, ARQ and their combinations) and progressive source coding.

6. QUALITY-OF-SERVICE MONITORING AT CLIENT

In the previous sections we described how the delivery of media data can be efficiently and robustly delivered over distributed networks. However, to many end-users (clients) the only thing that matters is the quality-of-service (QoS) of the received video. QoS can be in the form of perceived quality of the received content, the response time, the latency/delay, and the playback experiences. For example, received content quality can vary extensively depending on a number of factors, including encoder sophistication, server load, intermediate transformations (re-encoding and transcoding), and channel characteristics (packet loss, jitter, etc).

The optimized delivery and robust wireless streaming mechanisms proposed in Sections 5.1 and 5.2 are aimed to enhance QoS from system resource optimization perspectives, taking into account the practical constraints such as unreliable and error-prone channels. However, in order to address QoS end-to-end, we need to evaluate the QoS at the client side to validate the results. This is especially important in the events of paid delivery services to guarantee the QoS at the receiving end.

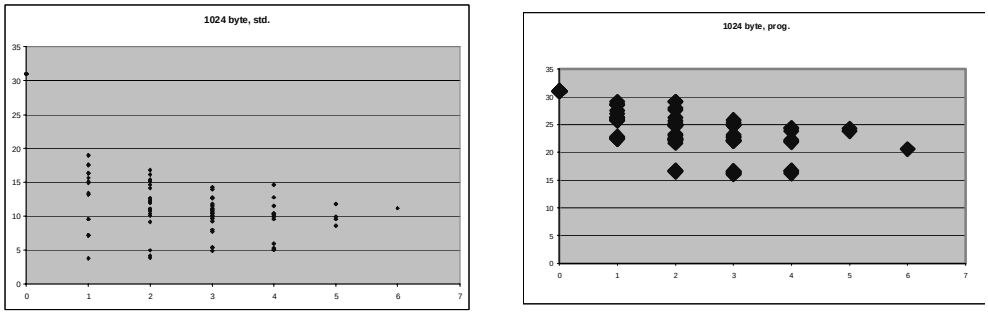
6.1 QUALITY METRICS FOR VIDEO DELIVERY

For QoS investigations, many in the networking research communities have relied on metrics such as bit and/or packet erasure characteristics. It is more realistic to investigate errors in the form of packet drops if we focus on video delivery over common TCIP/IP infrastructure. Unlike ASCII data, multimedia data, when encoded (compressed), can poses special challenges when the packets are dropped – the degradation of the quality is not uniform. The image/video quality can vary significantly based on loss packets – some packets would render a significant portion of the image fail to display, while others would only cause slight degradation to some small locally confined pixel areas. The quantitative measurements like bit/packet erasure rates and variances are not representative of perceived visual quality, and of course, QoS in general. to the end users. While scalable encoding/compression aim to solve the fluctuations of quality caused by packet losses, it does not lessen the needs for client side quality monitoring.

While QoS can embrace many different metrics to describe user perceived level of services, we will focus on image/video visual quality in this section.

Figure 14 shows the extent of quality fluctuations versus packet erasure measurements for standard encoding and also for scalar (progressive) encoding, in one example image. We use PSNR (peak-signal-to-noise-ratio) as the objective quality metrics, although other metrics can be used for perceptual quality measurements. Here we simulate the characteristics of packet erasure channels (Section 5.2). In the event which we have an average of 1 packet erasure (1024 bytes) per image (JEPG encoded), the PSNR of the received image frame can vary from under 10dB to close to 20dB – a 10dB differential! Scalable encoding improves quality despite the packet drops, but the PSNR can still show a 5 to 10dB differential for most erasure levels. (Similar characteristics have been observed for other images in our experiments.) This example demonstrates the needs for finding good quality

metrics and tools for assessing and monitoring perceived quality (visual in this case) of the received content.



(a) PSNR vs. packet erasure characteristics on test image (Standard compression, packet size = 1024 bytes)

(b) PSNR vs. packet erasure characteristics on test image (Progressive compression, packet size = 1024 bytes)

Figure 14: PSNR vs. packet erasure characteristics

In order to verify that the results of delivery (through a combination of packet loss, transcoding, or re-encoding, among other operations), we need to first focus on how to reliably assess a client's received content quality. For example, distortion assessment tools could be applied to quality-based real-time adaptation of streaming services. A streaming server could increase or decrease the bandwidth/error correction assigned to a stream based on a client's estimated perceptual quality, while given a corrupted frame, a streaming client could decide whether to attempt to render that frame, or instead repeat a previous uncorrupted frame. Similarly, encoders or transcoders could use automated quality monitoring to ensure that certain quality bounds are maintained during content transformations such as those of Section 5.1.

6.2 WATERMARKING FOR AUTOMATIC QUALITY MONITORING

For client side applications, it is often not feasible for the client devices to have access to the original content for quality comparison or assessment purposes. Thus, a general quality assessment scheme should require no use of the original uncorrupted signal. To tackle the challenge, we propose the use of *digital watermarks* as a means of quality assessment and monitoring [10].

Digital watermarking has previously been proposed as a key technology for copyright protection and content authentication. Watermarking considers the problem of embedding an imperceptible signal (the watermark) in a cover signal (most commonly image, video, or audio content) such that the embedded mark can later be recovered and retrieved, generally to make some assertion about the content, e.g., to verify copyright ownership, content authenticity, and authorization of use, among others[7][5]. Typically, a watermark is embedded by perturbing the host content by some key-dependent means, e.g. by additive noise, quantization, or a similar process.

The motivation behind the approach lies in the intuition that if a host signal is distorted or perturbed, a watermark embedded within the host signal can

be designed to degrade, and reflect the distortion by the degradation. By embedding known information (the watermark) in the content itself, a detector without access to the original material may be able to deduce the distortions to which the host content has been subjected, based on the distortions the embedded data appears to have encountered.
Figure 15 shows an overview of the general approach. If a watermark is designed well, it is possible to make a variety of assertions about the distortion to which the host signal has been subjected. In fact, the watermark can be designed to bear quantitative and qualitative correlations with perceived quality, and other QoS measurements. One approach is to embed perceptual weighting into a host signal (for example, an image) that reflects the human perception of the signals.

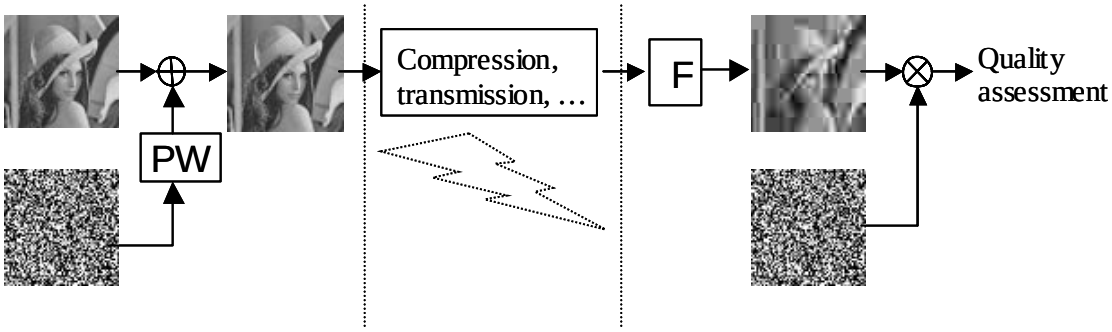


Figure 15: The System Overview (PW: Perceptual Weighting; F: Filtering)

However, there is a tradeoff. The watermark embedded has to be sufficiently robust, such that small distortions cannot render the watermark to disappear or undetectable. Yet it has to be sufficiently "fragile" to reflect the quality degradations. To achieve the goals, we propose using the class of robust detectors with bounded distortions, such that the detector response degrades with distortion magnitude, in a predictable manner. The detected results can then correlate with quality (quantitative and/or qualitative) metrics. In addition, the distortions can be localized and quantified with perceptual impacts. We call this "distortion-dependent watermarking".

There are few studies into this area and there are many interesting questions and opportunities for further research. [10] provides some basic problem formulations and investigation results. To illustrate, and as a basic formulation, we define the quality watermark signal *w* as

$$w_n = g_n(Q(X_n + r_n)),$$

where Q corresponds to a scalar quantizer with step size L, X_n is the n^{th} host signal element, r_n is chosen pseudo-randomly and uniformly over the range [0, L), and $g_n(\circ)$ corresponds to the output of the n^{th} pseudo-random number generator seeded with the specified argument, i.e. quantizer output. The quality watermark signal w and a Gaussian-distributed reference watermark w_{ref} are summed and normalized, and embedded into the content using one of the many watermarking algorithms reported in the literature. This essentially constitutes a pre-processing step that is applied before the content is manipulated and delivered. In [10], a simple spatial-domain white-noise watermark was used as the underlying embedding/detection mechanism. Several related and widely used objective quality metrics can subsequently be estimated from the distorted content, for

example, the sum of absolute differences (SAD). From our investigations the predicted distortion closely tracks the actual distortion measured from the experiments. Similar approaches can also be used to construct other perceptually weighted distortion metrics.

To our knowledge the problem of blind signal quality assessment has not been studied extensively, and many more sophisticated solutions are no doubt possible. As digital delivery of content becomes more prevalent in the coming years, we believe that such solutions will collectively become important enabling component in media delivery infrastructure.

7. SUMMARY AND DISCUSSIONS

In this chapter we have presented many important aspects of distributed video management, computing and delivery system in a peer-to-peer setting, beyond existing infrastructure. We however, believe the issues discussed here, and some of the solutions presented, are just a beginning, rather than a conclusion, of related topics. Many interesting questions are not yet answered or have been extensively studied.

Nonetheless, this is a start. We have introduced novel schemes for multirate differential encoding, distributed video coding and distributed adaptive storage and cache management, for data storage and management. We have also discussion distributed search mechanisms. For content delivery over distributed networks, we have presented new formulations and schemes for optimal dynamic transcoding as well as robust unicast and multicast over wireless LANs. In addition, we have introduced the concept of perceived quality monitoring via embedded watermarks to assess or validate the content delivery results. And more importantly, we have generalized the various components and designed an architecture on top of standard peer-to-peer service layer, and, built a prototype system MAPS to illustrate the concepts. The technology components and the integrated system architecture are aimed to enable an array of emerging applications in distributed video management and delivery.

REFERENCES

[1] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan, "Priority encoding transmission," IEEE Trans. Inf. Theory, no. 42, pp. 1737--1744, November 1996.

[2] AV/C Digital Interface Command Set VCR Subunit Specification. 1394 Trade Association Steering Committee. Version 2.0.1, Jan. 1998.

[3] P. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," submitted to IEEE Transactions on Multimedia, 2001.

[4] I. Clarke, O. Sandberg, B. Wiley, and T.W. Hong, “Freenet: A Distributed Anonymous Information Storage and Retrieval System,” in Designing Privacy Enhancing Technologies: International Workshop on Design Issues in Anonymity and Unobservability, ed. by H. Federrath. Springer: New York (2001).

[5] Communications of the ACM, Special Issue on Digital Watermarking, 1998.

[6] G. Cote, B. Erol, M. Gallant and F. Kossentini, “H.263+: Video Coding at Low Bit Rates,” IEEE Transactions on Circuits and Systems for Video Technology, vol.8, no.7, pp.849-66, November 1998.

[7] I. Cox, M. Miller, and J. Bloom, Digital Watermarking, Morgan Kaufmann, 2002.

[8] EXIF - Exchangeable Image File tag specification. http://www.pima.net/standards/it10/PIMA15740/exif.htm.

[9] L. Gong, “JXTA: A Network Programming environment,” IEEE Internet Computing, pp. 88-95, May 2001.

[10] M. Holliman and M. Yeung, “Watermarking for automatic quality monitoring”, Proceedings of SPIE: Security and Watermarking of Multimedia Contents IV, Volume 4675, pp. 458-469, 2002.

[11] ID3 tag specification. http://www.id3.org/.

[12] R. Lienhart, “Reliable Transition Detection in Videos: A Survey and Practitioner’s Guide,” International Journal of Image and Graphics (IJIG), Vol. 1, No. 3, pp. 469-486, 2001.

[13] R. Lienhart and A. Wernicke, “Localizing and Segmenting Text in Images, Videos and Web Pages,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 12, No.4, pp. 256 -268, April 2002.

[14] R. Lienhart, M. Holliman, Y.-K. Chen, I. Kozintsev, and M. Yeung, “Improving Media Services on P2P Networks,” IEEE Internet Computing, pp. 73-77, Jan./Feb. 2002.

[15] S. Lin and D. Costello, Error Control Coding: Fundamentals and Applications, Prentice-Hall, 1983.

[16] M. Macedonia, “Distributed file sharing: barbarians at the gates?” *IEEE Computer*, Vol. 33(8), pp. 99-101, Aug. 2000.

[17] A. Majumdar, D. Sachs, I. Kozintsev, K. Ramchandran and M. Yeung, "Multicast and unicast real-time video streaming over wireless LANs," IEEE Transactions on Circuits and Systems for Video Technology, Volume: 12 Issue: 6 , Jun 2002, Page(s): 524 -534.

[18] A. Majumdar, R. Puri and K. Ramchandran, “Rate-Distortion Efficient Video Transmission from Multiple Servers,” International Conference on Multimedia and Expo (ICME), Lausanne, Switzerland, August 2002.

[19] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," ACM SIGCOMM 96, 1996.

[20] A. E. Mohr, E. A. Riskin, and R. E. Ladner, "Graceful degradation over packet erasure channels through forward error correction," Proc. Data Compression Conference, Snowbird, UT, Mar. 1999.

[21] R. Puri, K. W. Lee, K. Ramchandran, and V. Bharghavan, "An Integrated Source Transcoding and Congestion Control Framework for Video Streaming in the Internet," IEEE Transactions on Multimedia, vol. 3, no. 1, pp. 18--32, March, 2001.

[22] L. Rizzo, "On the feasibility of software FEC," DEIT Technical Report LR-970131. Available at http://www.iet.unipi.it/~luigi/softfec.ps.

[23] Video Content Analysis Homepage. http://www.videoanalysis.org.

[24] S. Tilley and M. DeSouza, “Spreading Knowledge about Gnutella: A Case Study in Understanding Net-centric Applications,” in *Proceedings of the 9th International Workshop on Program Comprehension*, pp. 189 –198, 2001.

[25] D. Wu, T. Hou, and Y.-Q. Zhang, "Scalable video transport over wireless IP networks," IEEE International Symposium on Personal, Indoor and Mobile Radio Communication, Sept. 2000, pp. 1185-1191.

[26] B.-L. Yeo and B. Liu, “Rapid Scene Analysis on Compressed Video,” IEEE Transactions on Circuits and Systems for Video Technology, Vol. 5, No. 6, Dec. 1995.

[27] A. Zaccarin & B.L. Yeo, “Multi-rate encoding of a video sequence in the DCT domain”, Proceedings of IEEE Int. Symposium on Circuits and Systems, Phoenix, AZ, USA. May 2002.