Failure-Aware, Open-Loop, Adaptive Video Streaming With Packet-Level Optimized Redundancy

Yiannis Andreopoulos, *Member, IEEE*, Ram Keralapura, *Student Member, IEEE*, Mihaela van der Schaar, *Senior Member, IEEE*, and Chen-Nee Chuah, *Senior Member, IEEE*

***Abstract*—A plethora of coding and streaming mechanisms have been proposed for real-time multimedia transmission over the Internet. However, most proposed mechanisms rely only on global (e.g. based on end-to-end measurements), delayed (at least by the round-trip-time), or statistical (often based on simplistic network models) information available about the network state. Based on recently-proposed state-of-the-art open-loop video coding schemes, we propose a new integrated streaming and routing framework for robust and efficient video transmission over networks exhibiting path failures. Our approach explicitly takes into account the network dynamics, path diversity, and the modeled video distortion at the receiver side to optimize the packet redundancy and scheduling. In the derived framework, multimedia streams can be adapted dynamically at the video server based on instantaneous routing-layer information or failure-modeling statistics. The performance of our integrated application and network-layer method is simulated against equivalent approaches that are not optimized based on routing-layer feedback and distortion modeling, and the obtained gains in video quality are quantified.**

***Index Terms*—Network-layer feedback, open-loop scalable video coding, predicting network availability, routing-layer adaptation, video transmission under path failures.**

I. Introduction

THERE are two major causes of network performance degradation when streaming video over the Internet: network congestion and routing instability due to link/node failures. Past literature on video streaming over the Internet has been focusing on coping with bandwidth variation and losses [1]–[3] that are caused by network congestion. However, recent studies have found that the level of congestion in the core IP backbone is always negligible and has relatively small impact on latency sensitive applications [4]. On the other hand, link/node failures have been observed to be fairly common in the day-to-day operation of a network [5] due to fiber cuts, faulty equipments, or router misconfigurations. Rerouting after a link/node failure can take tens of seconds within a single domain [5], while inter-domain route changes through the border gateway protocol (BGP) can take up to minutes to converge [6]. During this transient period of route convergence, packets can be dropped (because of invalid paths) or caught in routing loops leading to additional delays. These routing instabilities result in service disruptions at the application layer and can adversely affect the quality of real-time video streaming. Notice that in the case of content delivery networks, such as Akamai [46], multiple client requests for the same video stream are grouped together, which means that a failure on a backbone link could affect a number of customers.

Several coding and rate-adaptation mechanisms have been proposed to cope with bandwidth variations, losses, and/or delay jitter at either the sender or the receiver [7]. However, most proposed mechanisms rely on either delayed (at least round-trip-time) end-to-end observations of the network state [3], [9], [14], [16], [17] or simplistic statistical channel models. While some models, e.g., Gilbert model [2], [19], may be suitable for bursty loss patterns of wireless channels or congestion-induced losses, they fail to capture routing dynamics in the presence of failures, which occur frequently in the IP-networks [8]. For all these cases, the lack of feedback on instantaneous network conditions can lead to sub-optimal performance for delay-sensitive video streaming. Moreover, end-to-end or application-level measurements can only detect the packet losses, but cannot easily establish their cause, e.g., changes at the underlying network topology. For example, the paths between a receiver and multiple sources may share the same physical link. In this case, there is no true redundancy at the routing-layer to protect against failure of that critical link.

In this paper, we propose a new integrated compression, streaming, and routing framework for robust and efficient streaming of multimedia content over wide-area Internet in the presence of failures. In Section II, we propose to deploy routing proxies to detect link or node failures directly. Upon failure detection, the proxy sends explicit notification to the multimedia server to report the *starting time* and *location* (path)

Manuscript received March 28, 2005; revised February 10, 2006. The work of Y. Andreopoulos and M. van der Schaar was supported by the National Science Foundation under CAREER Award CCF-0541867 and grants from Intel IT Research and UC Micro. The work of R. Keralapura and C.-N. Chuah was supported by the National Science Foundation under CAREER Award ANI-0238348. Portions of this paper were presented at the IEEE International Conference on Image Processing, September 2003, the IEEE International Workshop on Quality of Service, June 2004, and the IEEE International Conference on Communications, June 2004. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ching-Yung Lin.

Y. Andreopoulos was with the Department of Electrical Engineering, University of California, Los Angeles, CA 90095-1594 USA. He is now with the Department of Electronic Engineering, Queen Mary University of London, London E1 4NS, U.K. (e-mail: yiannis.a@elec.qmul.ac.uk)

M. van der Schaar is with the Department of Electrical Engineering University of California, Los Angeles, CA 90095-1594 USA (e-mail: mihaela@ee.ucla.edu).

R. Keralapura and C.-N. Chuah are with the Department of Electrical and Computer Engineering, University of California, Davis, CA 95616, USA (e-mail: rkeralapura@ucdavis.edu; chuah@ece.ucdavis.edu).

Digital Object Identifier 10.1109/TMM.2006.884607

of a failure event. Similarly, the proxy notifies the multimedia server when the path is restored. We develop a novel technique to estimate the service disruption time after a failure event by taking into account transient routing dynamics during the convergence period of Internal Gateway Protocols (IGP) and operational conditions, such as BGP table size and traffic distributions. When the explicit failure notification is not available, the proposed technique estimates the *expected service disruption* based on a *realistic failure model* that captures the failure patterns observed in operational IP-backbone networks [8].

Based on the derived failure-detection and modeling framework, we propose novel multipath open-loop (scalable) video transmission schemes that enable on-the-fly redundancy adaptation, as described in Section III. A key element of the proposed adaptation mechanism is the deployment of a new distortion-estimation model (Section III-B) that infers the expected frame-decoding error as well as the temporal error propagation, based on content characteristics accumulated at encoding time. This model provides accurate predictions for the decoder objective quality in real-time, and under various transmission conditions. Finally, for the multipath failure-aware transmission framework of this paper, novel packet scheduling algorithms are proposed in Sections III-C and III-D. When network feedback is available, the algorithm leverages the explicit failure notification to schedule the application-layer packets on all working paths according to the expected distortion reduction offered by the packets of each path. Alternatively, when channel feedback is not available, the algorithm establishes the redundancy level for packet scheduling based on distortion-reduction estimates for each path provided by the combination of the proposed video distortion model with the estimated failure probability of each path.

The derived framework facilitates flexible tradeoffs between throughput, redundancy and complexity, that are not possible with conventional error-tolerant video coding schemes such as multiple-description (MD) coding [22], [24], forward-error-correction (FEC) multiple-description coding [25], [32], or multiple-description via multiple-state encoding [23]. In particular, our solution adjusts the bitstream redundancy at the packet-level during transmission time, based on the transient network behavior. Unlike previous MD coding mechanisms for streaming under path failures [38], we use an open-loop video coding system that permits seamless bitrate adaptation via packet scheduling, and propose a mechanism for on-the-fly, optimized packet redundancy for the given transmission paths. Moreover, unlike FEC-based MD coding where the redundancy level is determined statically based on statistical information gathered over a large time interval [25], [32], in our work the redundancy level can be adjusted on-the-fly based on the instantaneous information of the channel condition *and* the expected distortion at the receiver. In order to quantify the effectiveness of the proposed algorithms, experimental results are presented in Section IV, and our conclusions are drawn in Section V.

II. Modeling Network Failures and Routing Reconvergence Dynamics

Previous empirical studies have shown that congestion losses can be modeled with low-order Markov chains, and the number of lost packets in a loss period is approximately geometric [2]. However, the use of a similar model for loss patterns during routing instabilities caused by link/node failures has not been validated. When a link or node fails, it is often followed by a transient routing instability (or route reconvergence) period during which all the routers in the network are notified of the failure, recompute their routing tables, and update their forwarding paths. There are two distinct stages during the convergence period.

- **Black-out stage**: All packets traversing the failed link are dropped initially due to invalid forwarding path.
- **Routing-loop stage**: Subsequently, some of these packets may be caught in routing loops because of inconsistent forwarding tables at various routers. The packets caught in the loop will traverse a random number of extra hops and hence experience extra delay before being successfully delivered to their final destination. The additional time-to-live (TTL) of a packet caught in a routing loop can be used to estimate the increase in end-to-end delay. Routing loops from which packets do not escape contribute to an increase packet loss rate but have no effect on the delay performance.

Once the routing protocol converges, the traffic will be forwarded on the backup path, which could be longer than the original path, resulting in an increased end-to-end delay. After the link or node recovers, traffic forwarding will resume on the original path.

A. Empirical Models for Network Failures

The first step towards measuring the impact of failures on multimedia streaming performance is to develop a detailed understanding of how often failures occur in a network and how long they last. The distribution of the frequency and duration of link failures observed in a Tier-1 ISP backbone are reported in our previous work [5], [8], [21]. Fig. 1 shows the distribution of independent link failure events that occurred across more than 600 links within a Tier-1 backbone network of an ISP over a seven-month period (April–October 2002).[1] Note that, as shown in our previous study [21], failures are fairly well spread out across weeks, days, and even over the course of a single day. Clearly, they need to be taken into account as part of every day operations. In our simulation studies and analysis, we model the network failures based on the following observations [8].

- The majority (70%) of the unplanned failure events are isolated, i.e., only affect a single link at a time, and hence can be modeled as independent link failures.
- Links are highly heterogeneous: some links fail significantly more often than others. This motivates us to classify the links into two categories: “high-frequency” and “low-frequency” links and model them separately. Within each class, the number of failures, $n(l)$, for link l roughly follows a power-law: $n(l) \propto l^{-k}$, where the exponent is found to be $k = -0.73$ for high-frequency links and $k = -1.35$ for low-frequency links.

[1]For proprietary reasons we are unable to provide absolute numbers but only the normalized number of failures to show the difference in the order of magnitude between the number of failures experienced by different links.

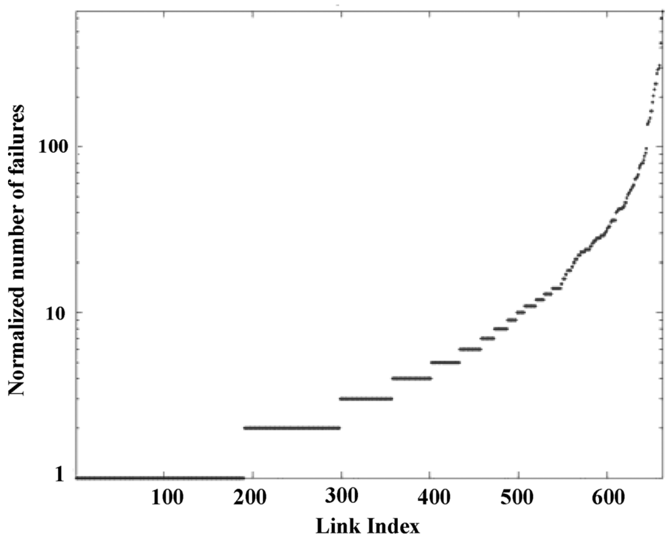


Fig. 1. Nonuniform distribution of independent failures across over 600 links within a Tier-1 ISP backbone.

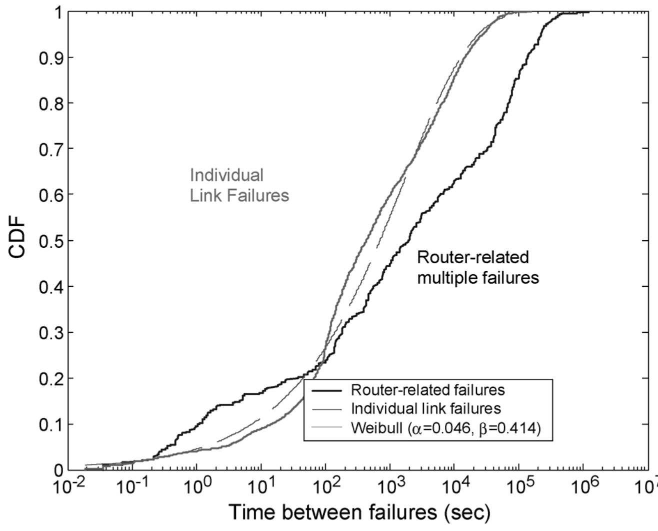


Fig. 2. Network-wide time between failures for router-related multiple failures, low-failure links, and the approximation by the Weibull distribution.

- The empirical cumulative distribution function (CDF) for time between any two failures can be approximated by a Weibull distribution as found in our previous study [8].

For example, Fig. 2 shows the empirical CDF for the network-wide time between failures for low-frequency links. The Weibull parameters can be derived for each set of empirical data based on maximum-likelihood estimation, e.g., in this case, $\alpha = 0.046$, and $\beta = 0.414$. The cumulative distribution of the duration of failures observed over the same period shows that most failures are transient (i.e., short-lived): 46% last less than a minute and 85% last less than ten minutes [21]. However, as discussed earlier, the actual failure duration is much less important since the packets will be forwarded correctly on the recomputed (alternate) paths once the routing protocol converges. Hence, we only concentrate on modeling the loss behavior during the reconvergence period following a failure, but not the failure duration itself. The experiments conducted in the same study [21] indicate that this instability period can last between 2 to 6.6 s. This is in agreement with another finding that the duration of routing loops is mostly under 10 s [20]. We will describe the routing convergence behavior in details in the next section.

B. Network Dynamics During Routing Convergence

Within a single network domain or an autonomous system (AS), an IGP like IS–IS [12] and OSPF [13], is used to exchange connectivity information and compute the shortest path between different source-destination pairs. These protocols are *linkstate* protocols, where each node has complete knowledge of the network topology including all the links present in the network. When a node detects a change in the network (due to link/node failures or a configuration changes), it is responsible for disseminating the new topology description to all its neighbors, recomputing all-pair shortest paths, and updating its own routing and forwarding tables.

From the time of the topology change to the time all nodes have been informed of the change and have updated their routing/forwarding tables, traffic disruptions (i.e., packet drops, routing loops) are possible as the nodes may have an inconsistent view of the network. We define "network convergence time" as the time it takes for all nodes to be notified of the change and update their forwarding tables. We can also define "convergence time," on a per node basis, where the time for network convergence is the maximum among all per-node convergence times. The convergence time can be summarized as a combination of three components [40].

- *Detection time*: The time required by a node in the network to identify if neighboring nodes are not reachable. Today's IP routers provide several mechanisms to perform this function [15] but all of them are based only on local information exchanged between the two nodes. For this reason, detection time represents a fixed price that is independent of the network topology or configuration.
- *Notification time*: The time taken by the routing message update to propagate across the network. In link state protocols, messages are flooded throughout the network. Hence, the notification time strongly depends on the network diameter (maximum hop distance between two nodes). Each node processes the message update and forwards it to its neighbors introducing a delay in the propagation of the information.
- *Route update time*: The time spent by each node in updating its routing information. The update of this information consists of two steps. First, each node recomputes its routing tree (i.e., the shortest path to every other node) and then applies the changes to all the network prefixes that have been learnt via the BGP inter-domain protocol. The result of this computation is a forwarding table where each prefix is associated with a neighboring node (next hop).The route update time of a node is proportional to the number of prefixes for which the next hop information needs to be changed. In turn, the number of prefixes to be updated depends on the location of the topology change and on the distribution of prefixes to egress nodes. Indeed, the closer

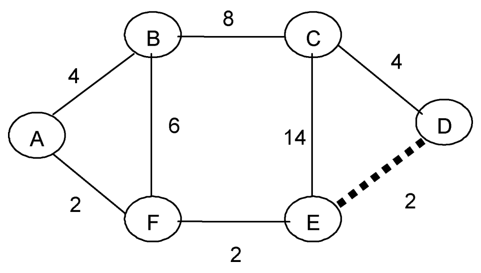


Fig. 3. Example of convergence time and service disruption.

TABLE I
SUMMARY OF ROUTING EVENTS (WITH A ROUTE UPDATE TIME OF 400 ms)

Time	Event	Forwarding Path (A-D)	Service (A to D)
0.0 sec	Failure of link E-D	A-F-E-D	NO
1.0 sec	D, E: failure detection	A-F-E-D	NO
1.8 sec	D, E: route update	A-F-E-C-D	YES
2.0 sec	C, F: notified of failure	A-F-E-C-D	YES
2.2 sec	C: route update	A-F-E-C-D	YES
2.4 sec	B: notified of failure	A-F-E-C-D	YES
2.8 sec	F: route update	A-F-B-F-... (Routing loop B-F)	NO
3.0 sec	A: notified of failure	A-F-B-F-... (Routing loop B-F)	NO
3.2 sec	B: route update	A-F-B-C-D	YES
3.8 sec	A: route update	A-B-C-D (Network Convergence)	YES

the topology change occurs to a node, the larger will be the number of prefixes affected. Similarly, if a large number of prefixes share the same egress node, a change in the topology close to that egress node will result in long route update time for all nodes in the network.

The network convergence time only provides a rough upper bound on service availability. The service is not available when the packets cannot reach their destinations due to the lack of forwarding information. Given that traffic forwarding may resume even if all the nodes in the network have not updated their routing table, there may be a significant difference between network convergence and service availability.

Consider the network illustrated in Fig. 3. Assume that the nodes and the links have similar characteristics with a detection time of 1000 ms, a notification time between adjacent nodes of 200 ms, and a node route update time of 800 ms (except for node C for which we assume this is 200 ms given that it does not need to change its route to reach D). The number on each link indicates the IGP weight. Consider the disruption observed for the traffic sent from node A to node D due to the failure of link E–D at time $t_o = 0$. Table I shows the routing events, changes in the forwarding path and service availability from node A to node D. In this example, we assume that a node notifies neighboring nodes only after it has completed the update of its own routing table. However, it is easy to verify that if the updates are sent before updating the routing table, the example yields similar results.

Interestingly, as the message update propagates across the network, the forwarding path from node A to node D changes four times. Some of these intermediate paths are valid thus restoring service between A and D, while some are not, causing packet drops ("traffic black-hole") and routing loops. For example, packets are dropped until node E has computed a new forwarding path to reach D and routing loops occur when nodes B and F have conflicting forwarding information. In the next subsection, we introduce an algorithm to compute the cumulative time for which service is not available or affected during the routing convergence period.

C. Routing Introspection and Feedback Layer

One can assume that multimedia clients can send acknowledgment packets (ACKs) or statistical feedbacks in the form of Real-time Control Protocol (RTCP) reports to inform the sender/proxy of the loss rate and round trip time (RTT) observed at the application layer. While such feedbacks are useful, they do not provide sufficient information to determine the cause of the performance degradation. We propose the following two mechanisms to explicitly detect link/node failures and estimate the associated service disruption time.

1) Proxy-Assisted Failure Detection: We propose to deploy routing proxies in the network to detect link or node failures directly. The proxy contains a route listener, such as Python Routing Toolkit (PyRT) [10] or Zebra [11], that allows it to receive routing messages from an adjacent network router. Since most network domains run link-state routing protocols, such as IS–IS [12] or OSPF [13], any changes in the routing topology (link/node addition or deletion) are flooded throughout the network, e.g., via Link-State Announcements (LSAs) in OSPF.

The routing proxy associated with a multimedia server keeps track of the end-to-end paths used for video streaming to its various clients, e.g., by running `traceroute` from the video server to its client. When the proxy receives an LSA announcing a link or node failure that affects the path of the video streaming, it notifies the video server of the failure. This marks the start time of a failure and routing instability period, which can take up to tens of seconds [5] before the network convergences to the backup routing paths. The passive route listeners can be placed at strategic locations to monitor routing dynamics of both IGPs within an AS and external gateway protocols between different ASes. When the multimedia server receives an explicit failure notification from the proxy, the coding and streaming schemes are adapted, as described in Section V.

2) Estimating Service Disruption Time: To capture the effect of a link failure and its subsequent impact on traffic forwarding and perceived streaming quality, we define a metric called service disruption time (SDT) [40]. SDT is the amount of time for which the service between a source and destination is disrupted due to a link failure. In essence, it is the upper bound of the time for which connectivity between a particular video client and the server experiences a degradation in performance. In Table II, we present an algorithm to compute SDT.

D. Multipath Routing Architecture

Since one of the requirements of our system is for the multimedia server to find multiple paths from the source to the destination, we conclude this section by identifying two ways to accomplish this.

TABLE II
ALGORITHM TO ESTIMATE SERVICE DISRUPTION TIME (SDT) DUE TO SINGLE-LINK FAILURES

Step 1:	Initialize the *service disruption time*, $f_l(x,y)$ for the path from x to y due to the failure of the link l to zero i.e. $f_l(x,y) = 0$. If the original path from x to y does not contain link l then QUIT.
Step 2:	Find the convergence time (CT) for each node and list the nodes in the increasing order of convergence time. Let the convergence time of the 1st node in the list be CT^1, 2nd node be CT^2 and so on. In general, the convergence time for the n^{th} node in the list is CT^n. Set the *current node*, k, (i.e. the node number on the sorted list) to zero.
Step 3:	Increment k to 1. Set $f_l(x,y) = CT^1$. At the time instant CT after the failure event, find the path that a packet from the source node, x, follows to reach the destination node, y, taking into account that the first node in the list has converged and others have not. If the path has a routing loop or black-hole, then set *previousDisruption = true* else set *previousDisruption = false*.
Step 4:	Increment k by 1. At the time instant CT^k after the failure event, find the path that a packet from the source node, x, follows to reach the destination node, y, taking into account that the intermediate nodes might have converged or not.
Step 5:	If the path that the packet follows does not contain the failed link l or a routing loop, then set *previousDisruption = false*. Go to Step 7. Else go to Step 6.
Step 6:	If the path that the packet follows contains the failed link l or has a routing loop, then the path from x to y is still disrupted. If *previousDisruption = false*, then do not update the *service disruption time* but set *previousDisruption = true* Else if *previousDisruption = true*, then update the *service disruption time* in the k-th iteration as, $f_l(x,y) = f_l(x,y) + CT^k - CT^{k-1}$
Step 7:	If there are more nodes in the list then go to Step 4. Else QUIT.

1) Collaboration with different ISPs to exploit Equal Cost Multi-Path (ECMP) routing. ECMP provides the multimedia server with multiple paths between a source and destination node.
2) Construction of an overlay network or leveraging of the mechanisms offered by existing overlay networks like Resilient Overlay Network (RON) [42], Detour [43], SplitStream [44], OverCast [45], etc. Overlay networks are application layer networks with multiple overlay nodes that collaborate with each other at the application layer to provide features (such as multipath routing) that are not readily supported by IP layer routing services. For example, RON [42] and Detour [43] demonstrate that end-to-end route selection often finds multiple alternative paths by relaying traffic among overlay nodes.

Even though there are multiple ways to accomplish multipath routing, in this work, we propose to use the second approach based on overlay networks not only because it is simple but also because, in practice, ISPs are very reluctant to cooperate on routing issues with their customers. In the remainder of the paper we assume that the multimedia server has the ability to discover and utilize multiple paths through an ISP backbone network, and focus mainly on its streaming technique.

III. FAILURE-AWARE STREAMING WITH OPTIMIZED REDUNDANCY

We present a novel framework for optimized redundancy in video transmission over multiple paths through the use of (open-loop) motion compensated temporal filtering (MCTF) as a decorrelating transform [29]. Our scheme retains the advantages of low-complexity since the redundancy generation is applied post-encoding [29], i.e., during the bitstream-extraction and packetization stage. In this way, our system takes into account the bitstream packetization aspects and consequently achieves a better synergy between the application and network layers. A certain percentage of the important visual information can be easily reproduced via all transmission paths so that quality-of-service (QoS) requirements are facilitated. This is achieved in a very efficient manner by exploiting the rate-distortion metrics established during the multitrack hinting stage. Section III-A briefly reviews MCTF-based video coding. Section III-B presents our proposed distortion-modeling strategy at the packet level. Section III-C presents the proposed multipath video transmission framework and finally Section III-D presents two scenarios for packet scheduling and redundancy that depend on the available network feedback mechanisms and on the proposed distortion-modeling strategy.

A. Motion Compensated Temporal Filtering

Recent state-of-the-art scalable video coding schemes that have been adopted for the generation of adaptive packet redundancy [29] are based on motion compensated temporal filtering. During MCTF, the original video frames are filtered temporally in the direction of motion, prior to performing the spatial transformation and coding, Video frames are filtered into L (low-frequency or average) and H (high-frequency of difference) frames, as shown in Fig. 4. For example, for two consecutive video frames A and B, an instantiation of MCTF can be written using the lifting formulation [26], which, for each pixel (x, y) of a $X \times Y$ video frame, is written as

$$H[x,y] = \frac{1}{\sqrt{2}}\left(B[x,y] - p_{\mathcal{F}}[x+v_x, y+v_y] \cdot A[x+v_x, y+v_y]\right) \quad (1)$$

$$L[x+v_x, y+v_y] = \sqrt{2}A[x,y] + u_{\mathcal{B}}[x,y] \cdot H[x,y] \quad (2)$$

where (v_x, v_y) is the motion vector associated with pixel (x, y) and $p_{\mathcal{F}}[\bullet,\bullet]$, $u_{\mathcal{B}}[\bullet,\bullet]$ are pixel weights chosen by an optimization mechanism that normalizes the information during the forward $(\mathcal{F})$ motion-compensated prediction of (1) and the backward $(\mathcal{B})$ inversion of the motion information during the update step of (2). We refer to recent work [30], [33], [34] for a variety of algorithms that estimate these factors for various temporal

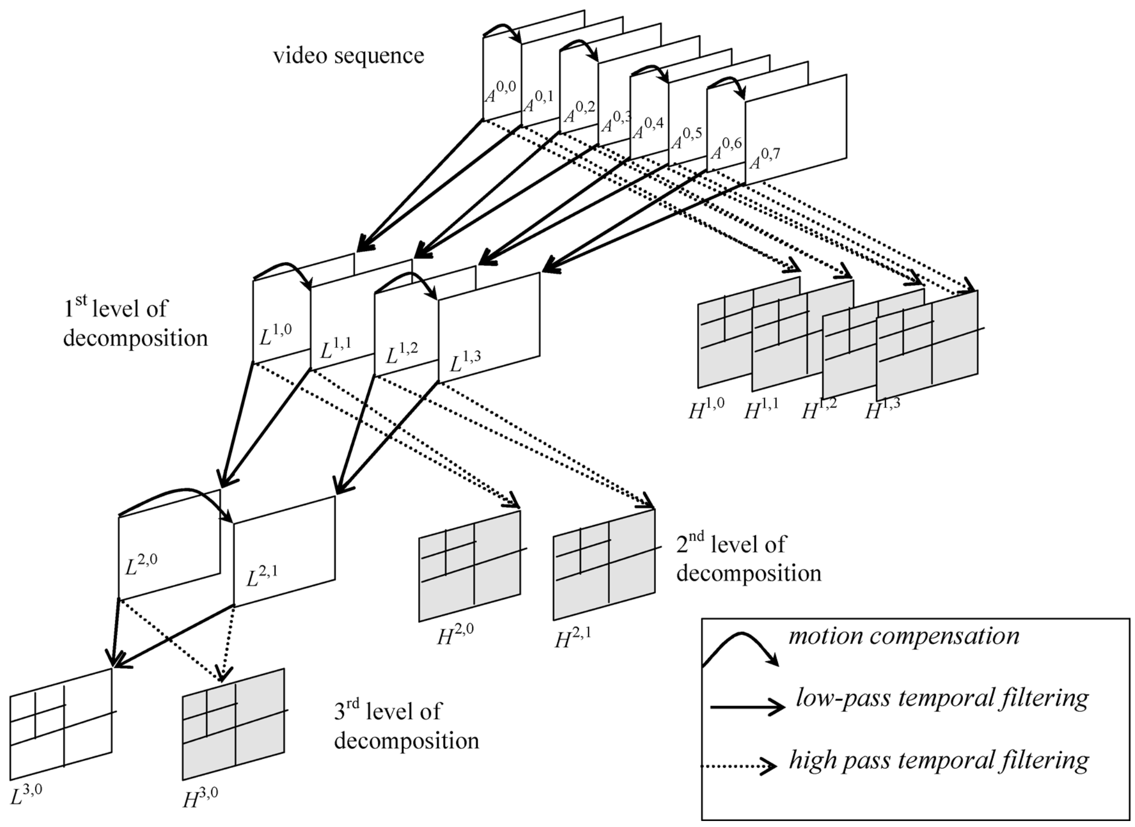

video sequence
$A^{0,0}$ $A^{0,1}$ $A^{0,2}$ $A^{0,3}$ $A^{0,4}$ $A^{0,5}$ $A^{0,6}$ $A^{0,7}$
1st level of decomposition
$L^{1,0}$ $L^{1,1}$ $L^{1,2}$ $L^{1,3}$
$H^{1,0}$ $H^{1,1}$ $H^{1,2}$ $H^{1,3}$
2nd level of decomposition
$L^{2,0}$ $L^{2,1}$
$H^{2,0}$ $H^{2,1}$
3rd level of decomposition
$L^{3,0}$ $H^{3,0}$
motion compensation
low-pass temporal filtering
high pass temporal filtering

Fig. 4. MCTF decomposition.

decompositions in order for the filtering process to approximate an orthonormal temporal transform. The process is applied initially in a group of pictures (GOP) and also to all the subsequently-produced L frames thereby forming a total of T temporal levels, as shown in Fig. 4.

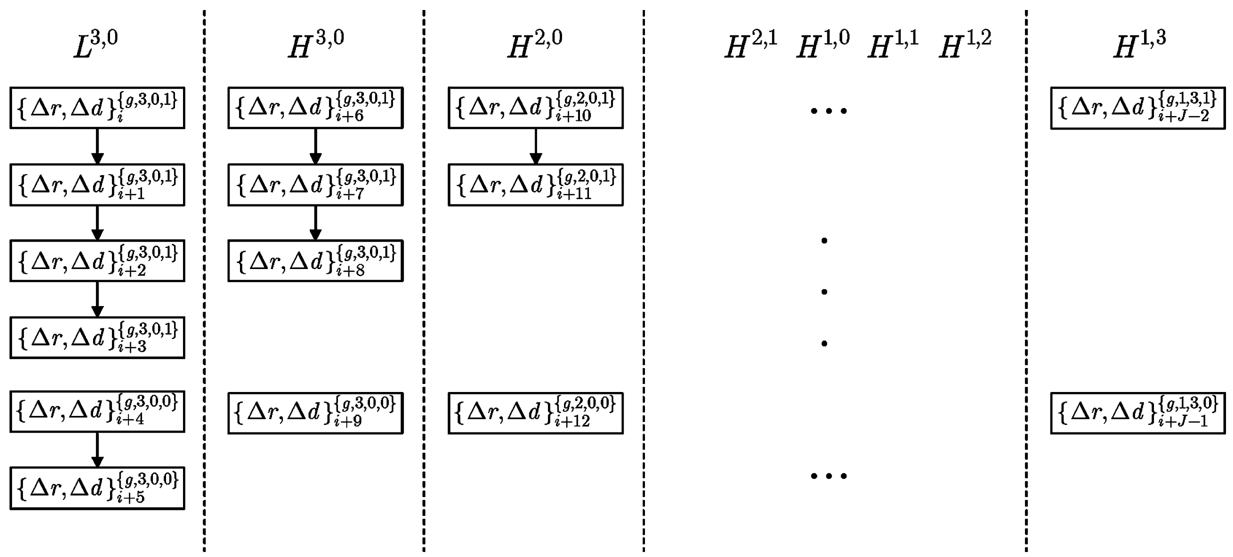
We use the notation $H^{t,k}$ to indicate the k-th H frame of temporal level t, where $1 \leq t \leq T$ and $0 \leq k < 2^{T-t}$. Equivalently the notation $L^{T,0}$ is used to indicate the remaining L frame at the last level after the completion of the temporal decomposition in the GOP.

B. Packetization and Distortion-Modeling Aspects of MCTF-Based Video

For each GOP, each $H^{t,k}$ frame and the remaining $L^{T,0}$ frame are compressed using embedded wavelet image coding [27], [33], [34]. Alternatively, embedded coding based on H.264/AVC [36] can be applied [37]. The vast majority of state-of-the-art embedded coding engines for still-images applies bitplane encoding, where, if the compressed bitstream is truncated at an intermediate point, one can estimate the expected transform-domain distortion [27].

Once the expected transform-domain distortion is established at different bitstream truncation points of each decoded-frame, we can associate this information with different video packets generated for that frame. An example is given in Fig. 5, where different frames of the temporal decomposition of one GOP are represented by a series of video packets. Each video packet is characterized by its corresponding rate increment and distortion decrement. In particular, for each video packet vp_{i+j}^{g}, where i is the first packet number of the current GOP and $0 \leq j < J$ is the in-GOP packet number (assuming a total of J packets for the current GOP), we denote these as $\Delta r[\mathrm{vp}_{i+j}^{g}]$, $\Delta d[\mathrm{vp}_{i+j}^{g}]$ (respectively), with $g = \{g, t, k, r\}$ a spatio–temporal coordinates vector indicating: the current-GOP number (g), the temporal level (t), the frame index within the temporal level (k) and the spatial decomposition level (r).

As indicated by the arrows of Fig. 5, in order to create video packets with bounded maximum size (where the bound is set by the size of an application-layer packet), the compressed information of a certain spatial resolution level r may be divided into several packets, which are dependent on previous packets of the same resolution. These dependencies vary based on the utilized compressed scheme; in the example of Fig. 5, we assume that the entire resolution level of each video frame is compressed into one bitstream, which is the case in the utilized codec [30]. Under any loss pattern $\mathbf{j} = \{j_1, j_2, \ldots, j_N\}$ for the video packets, if the packets of a frame at a certain spatio–temporal resolution are indicated by vp_{i+l}^{g}, with l bounded as $l_{\mathrm{start}} \leq l < l_{\mathrm{end}}$, we can define the expected transform-domain distortion reduction for this frame at the decoder side as shown in (3) at the bottom of the next page. Since the spatio–temporal transform decomposition is based on biorthogonal filter-pairs, the transform-domain distortion-reduction estimates based on distortion-modeling of embedded wavelet coding [27] in conjunction with (3) will not represent the distortion-reduction from the inverse MCTF process that utilized the certain amount of packets. Concerning lack of orthonormality in space, one can weight the transform-domain



$L^{3,0}$
$H^{3,0}$
$H^{2,0}$
$H^{2,1}$ $H^{1,0}$ $H^{1,1}$ $H^{1,2}$
$H^{1,3}$
$\{\Delta r, \Delta d\}_{i}^{\{g,3,0,1\}}$
$\{\Delta r, \Delta d\}_{i+1}^{\{g,3,0,1\}}$
$\{\Delta r, \Delta d\}_{i+2}^{\{g,3,0,1\}}$
$\{\Delta r, \Delta d\}_{i+3}^{\{g,3,0,1\}}$
$\{\Delta r, \Delta d\}_{i+4}^{\{g,3,0,0\}}$
$\{\Delta r, \Delta d\}_{i+5}^{\{g,3,0,0\}}$
$\{\Delta r, \Delta d\}_{i+6}^{\{g,3,0,1\}}$
$\{\Delta r, \Delta d\}_{i+7}^{\{g,3,0,1\}}$
$\{\Delta r, \Delta d\}_{i+8}^{\{g,3,0,1\}}$
$\{\Delta r, \Delta d\}_{i+9}^{\{g,3,0,0\}}$
$\{\Delta r, \Delta d\}_{i+10}^{\{g,2,0,1\}}$
$\{\Delta r, \Delta d\}_{i+11}^{\{g,2,0,1\}}$
$\{\Delta r, \Delta d\}_{i+12}^{\{g,2,0,0\}}$
$\{\Delta r, \Delta d\}_{i+J-2}^{\{g,1,3,1\}}$
$\{\Delta r, \Delta d\}_{i+J-1}^{\{g,1,3,0\}}$

Fig. 5. An example of the video packets generated for the spatio–temporal decomposition of Fig. 4.

distortion generated by the embedded wavelet coder according to the gain of the synthesis filter-kernel [27]. Nevertheless the lack of orthonormality in temporal decomposition cannot be corrected by a fixed weighting scheme, since advanced MCTF techniques, such as the ones used in the present paper and other recent state-of-the-art open-loop coding methods [33], [34], [37], create different dependencies among neighboring frames, depending on the motion-vectors selected for each area of every frame. In practice, this corresponds to the use of different filters in different levels of the temporal filtering, as well as in different areas of each frame.

In order to simplify our analysis we make the certain assumptions and simplifications. Similar to other studies [35], we do not analyze the case of fractional-pixel MCTF, but rather focus on the case of full-pixel MCTF. Moreover, instead of analyzing the error-propagation per pixel, we assume a uniform distribution of the decoding error among the pixels of an L or H frame [41]. Finally, we assume no correlation among the decoding error of consecutive L and H frames used for the reconstruction of consecutive output frames [41]. Under these assumptions, we can simplify the adaptive MCTF decomposition for each temporal level by expressing it as

$$H^{1,t} = \frac{1}{\sqrt{2}}\left(A^{0,2t+1} - \overline{p_{\mathcal{F}}^{2t}} \cdot A^{0,2t} - \overline{p_{\mathcal{B}}^{2t+2}} \cdot A^{0,2t+2}\right) \quad (4)$$

$$L^{1,t} = \sqrt{2}A^{0,2t} + \overline{u_{\mathcal{F}}^{t-1}} \cdot H^{1,t-1} + \overline{u_{\mathcal{B}}^{t}} \cdot H^{1,t} \quad (5)$$

where the factors $\overline{p_{\mathcal{M}}^{i}} = (1/X \cdot Y)\sum_{x=0}^{X-1}\sum_{y=0}^{Y-1} p_{\mathcal{M}}^{i}[x,y]$, $i = \{2t, 2t+2\}$, $\mathcal{M} = \{\mathcal{F}, \mathcal{B}\}$ express the average weighting of the pixels of the corresponding frames $A^{0,i}$ during the prediction step that utilizes forward ($\mathcal{F}$) and backward ($\mathcal{B}$) motion compensation, and $\overline{u_{\mathcal{M}}^{j}} = (1/X \cdot Y)\sum_{x=0}^{X-1}\sum_{y=0}^{Y-1} u_{\mathcal{M}}^{j}[x,y]$, $j = \{t-1, t\}$, express the average weighting of the pixels of the error frames $H^{1,j}$ during the (forward and backward) inverse motion compensation of the update step. These weighting factors are calculated during encoding according to the optimized temporal filtering for each frame and stored along with the compressed bitstream. Notice that, by adjusting $\overline{p_{\mathcal{M}}^{i}}$ and $\overline{u_{\mathcal{M}}^{j}}$, (4) and (5) express a generalized form of temporal filtering that can include the bidirectional Haar filter-pair with and without the update step [30], as well as the 5/3 temporal filter-pair [26] with and without the update step [30], which constitute the common choices found in the relevant literature [26], [30], [34] . For example, by setting $\overline{p_{\mathcal{B}}^{2t+2}} = 0$ and $\overline{u_{\mathcal{F}}^{t-1}} = 0$, (4) and (5) correspond to (1) and (2), i.e. the (uni-directional) Haar MCTF.

For each pair of frames $L^{1,t}, H^{1,t}$, the inversion process is found by establishing $A^{0,2t}$, $A^{0,2t+1}$ based on (4) and (5), yielding:

$$A^{0,2t} = \frac{1}{\sqrt{2}}\left(L^{1,t} - \overline{u_{\mathcal{F}}^{t-1}}H^{1,t-1} - \overline{u_{\mathcal{B}}^{t}}H^{1,t}\right) \quad (6)$$

$$A^{0,2t+1} = \sqrt{2}H^{1,t} + \overline{p_{\mathcal{F}}^{2t}}A^{0,2t} + \overline{p_{\mathcal{B}}^{2t+2}}A^{0,2t+2}. \quad (7)$$

The last equation can be expressed in function of L and H frames by replacing $A^{0,2t}$ and $A^{0,2t+2}$ based on (6), yielding:

$$A^{0,2t+1} = \left(\sqrt{2} - \overline{p_{\mathcal{F}}^{2t}} \cdot \overline{u_{\mathcal{B}}^{t}} - \overline{p_{\mathcal{B}}^{2t+2}} \cdot \overline{u_{\mathcal{F}}^{t}}\right) H^{1,t} - \overline{p_{\mathcal{B}}^{2t+2}} \cdot \overline{u_{\mathcal{B}}^{t+1}} \cdot H^{1,t+1} - \overline{p_{\mathcal{F}}^{2t}} \cdot \overline{u_{\mathcal{F}}^{t-1}} \cdot H^{1,t-1} + \overline{p_{\mathcal{F}}^{2t}} \cdot L^{1,t} + \overline{p_{\mathcal{B}}^{2t+2}} \cdot L^{1,t+1}. \quad (8)$$

Based on (6) and (8) we can express the expected mean square error (MSE) of the reconstructed output frames $\widetilde{A}^{0,2t}$,

$$D_{\mathrm{g}} = \begin{cases} \sum_{l=l_{\mathrm{start}}}^{l_{\mathrm{end}}-1} \Delta d\left[\mathrm{vp}_{i+l}^{\mathrm{g}}\right], & \text{if } \forall l \in [l_{\mathrm{start}}, l_{\mathrm{end}}-1] : i+l \notin \mathbf{j} \\ \sum_{l=l_{\mathrm{start}}}^{l_{\mathrm{loss}}-1} \Delta d\left[\mathrm{vp}_{i+l}^{\mathrm{g}}\right], & \text{if } \exists l_{\mathrm{loss}} \in [l_{\mathrm{start}}, l_{\mathrm{end}}-1] : [i+l_{\mathrm{loss}} \in \mathbf{j} \;\&\; (\forall l \in [l_{\mathrm{start}}, l_{\mathrm{loss}}-1] : i+l \notin \mathbf{j})] \end{cases}. \quad (3)$$

$\widetilde{A}^{0,2t+1}$ in function of the MSE of decoded (and spatially-reconstructed) frames $\widetilde{H}^{1,t}$, $\widetilde{H}^{1,t-1}$, $\widetilde{H}^{1,t+1}$, $\widetilde{L}^{1,t}$, $\widetilde{L}^{1,t+1}$. For any frame $M[x,y]$ we define this error as

$$\mathrm{E}\left\{e_A^2\right\} = \frac{1}{X \cdot Y} \sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} \mathrm{E}\left\{\left\|M[x,y] - \widetilde{M}[x,y]\right\|^2\right\} \quad (9)$$

where $\widetilde{M}[x,y]$ represents the decoded pixel of frame M at position (x,y). Based on our assumption that no correlation exists on the decoding error of adjacent transform-domain frames, we have

$$\mathrm{E}\{e_M \cdot e_N\} = 0 \quad (10)$$

for

$$M, N = \{H^{1,t}, H^{1,t-1}, H^{1,t+1}, L^{1,t}, L^{1,t+1}\} \text{ with } M \neq N. \quad (11)$$

In this way we derive the expected mean-square error after the reconstruction as:

$$\mathrm{E}\left\{e^2_{A^{0,2t}}\right\} = \frac{1}{2}\left(\mathrm{E}\left\{e^2_{L^{1,t}}\right\} + \left(\overline{u^t_{\mathcal{B}}}\right)^2 \mathrm{E}\left\{e^2_{H^{1,t}}\right\} + \left(\overline{u^{t-1}_{\mathcal{F}}}\right)^2 \mathrm{E}\left\{e^2_{H^{1,t-1}}\right\}\right) \quad (12)$$

$$\begin{aligned}\mathrm{E}\left\{e^2_{A^{0,2t+1}}\right\} = & \left(\sqrt{2} - \overline{p^{2t}_{\mathcal{F}}} \cdot \overline{u^t_{\mathcal{B}}} - \overline{p^{2t+2}_{\mathcal{B}}} \cdot \overline{u^t_{\mathcal{F}}}\right)^2 \mathrm{E}\left\{e^2_{H^{1,t}}\right\} \\ & + \left(\overline{p^{2t+2}_{\mathcal{B}}}\right)^2 \left(\overline{u^{t+1}_{\mathcal{B}}}\right)^2 \mathrm{E}\left\{e^2_{H^{1,t+1}}\right\} \\ & + \left(\overline{p^{2t}_{\mathcal{F}}}\right)^2 \left(\overline{u^{t-1}_{\mathcal{F}}}\right)^2 \mathrm{E}\left\{e^2_{H^{1,t-1}}\right\} \\ & + \left(\overline{p^{2t}_{\mathcal{F}}}\right)^2 \mathrm{E}\left\{e^2_{L^{1,t}}\right\} \\ & + \left(\overline{p^{2t+2}_{\mathcal{B}}}\right)^2 \mathrm{E}\left\{e^2_{L^{1,t+1}}\right\}. \end{aligned} \quad (13)$$

Based on (12), (13) we can estimate the reconstruction error after the inverse MCTF that utilizes a certain subset of video packets if we replace the expected MSE $\mathrm{E}\{e_M^2\}$, with M defined by (11), by the expected transform-domain reconstruction error given by (3) and appropriately weighted by the gain of the spatial-analysis filter-bank. For the example case of one temporal decomposition level and R spatial decomposition levels we have

$$\mathrm{E}\left\{e_M^2\right\} = \sum_{r=0}^{R} S_{\{g_M,1,t_M,r\}} D_{\{g_M,1,t_M,r\}} \quad (14)$$

with g_M, t_M indicate the GOP number and frame-index (within temporal-level one) of the particular frame M, respectively. In this paper, the utilized weighting for the transform-domain distortion of each spatial resolution r, $S_{\{g_M,1,t_M,r\}}$, is based on the weighting used in related work: $S_{\{g_M,1,t_M,r\}} = (\sqrt{2})^{-r}$.

In the generic case of a multilevel temporal decomposition, (3), (14), and (12)–(13) are applied hierarchically for all the frames of each temporal level in order to estimate the expected mean-square error at the output video frames of each GOP. For this purpose, in our streaming experiments the percentages of the pixels predicted and updated (forward and backward) by each frame during the temporal decomposition are kept along with the rate-distortion information in order to be used during the actual streaming process for the formulation of the expected distortion. It is important to notice that the application of (3), (14) and subsequently the calculations of (12) and (13) can be performed multiple times in real time during the streaming process as they do not demand a significant number of arithmetic operations. Hence, even if the proposed distortion-estimation scheme leads only to approximations of the actual decoded-frame distortion, it consists of a realistic framework that could be readily deployed on a streaming server. Alternatively, if nonembedded coding based on H.264/AVC [36] is used, or different forms of embedded coding inspired thereof [37], the entire process could be performed offline: for each packet-loss pattern, the distortion $D_{\{g_M,1,t_M,r\}}$ of each decoded frame could be measured based on multiple decodings, followed by the application of (12) and (13). This is similar to the concept of distortion-chain modeling [18], and could be potentially performed offline. Nevertheless, the complexity of such a framework can be significantly higher [18], since a large number of packet-loss patterns would have to be evaluated by offline decoding at the streaming server.

One important aspect not treated in this analysis concerns the expected distortion-reduction under losses of motion-vector information. In general, this results in nonlinear MSE variations across various reconstructed frames. As a result, we mark the motion-vector packets with the maximum expected distortion-reduction of their corresponding error frame in order to ensure that they are prioritized versus their corresponding texture bitstream in the error frames. In addition, if a certain motion-vector packet is not received, we zero the corresponding percentage of pixels linked with the appropriate reference frame in order to incorporate the effect of lost motion information in the estimation of (12)–(14).

Finally, by segregating all video packets $\mathrm{vp}_j^{\mathrm{g}}$ (with j indicating the video-packet number) into certain classes, if the sum of sizes of two or more packets in the same class is smaller than the maximum application-packet size, a packet aggregation (PA) process merges these packets thereby forming aggregated packets ap_a (with a indicating the aggregated packet number. The class segregation is formed based on the packet spatio–temporal vector g as well as based on the data dependencies among packets. More specifically, two packets are set in the same class if they belong to the same GOP, are of the same type (texture or motion-vector packet), spatial resolution, frame type (H or L), the same color channel (luminance or chrominance channels), and, either do not have any dependencies on other video packets, or depend on packets of the same temporal level. Packet aggregation within each class occurs in a breadth-first scan through the temporal decomposition of each GOP; this means that, in the vast majority of cases, packets of the same temporal level are grouped together. It is important to notice that the aggregation process is affected by the number of transmission paths (P); in particular, only packets that have frame-indices offset by $k \cdot P$, with $k \in \mathbb{Z}_+$, can be aggregated. This is mandated from

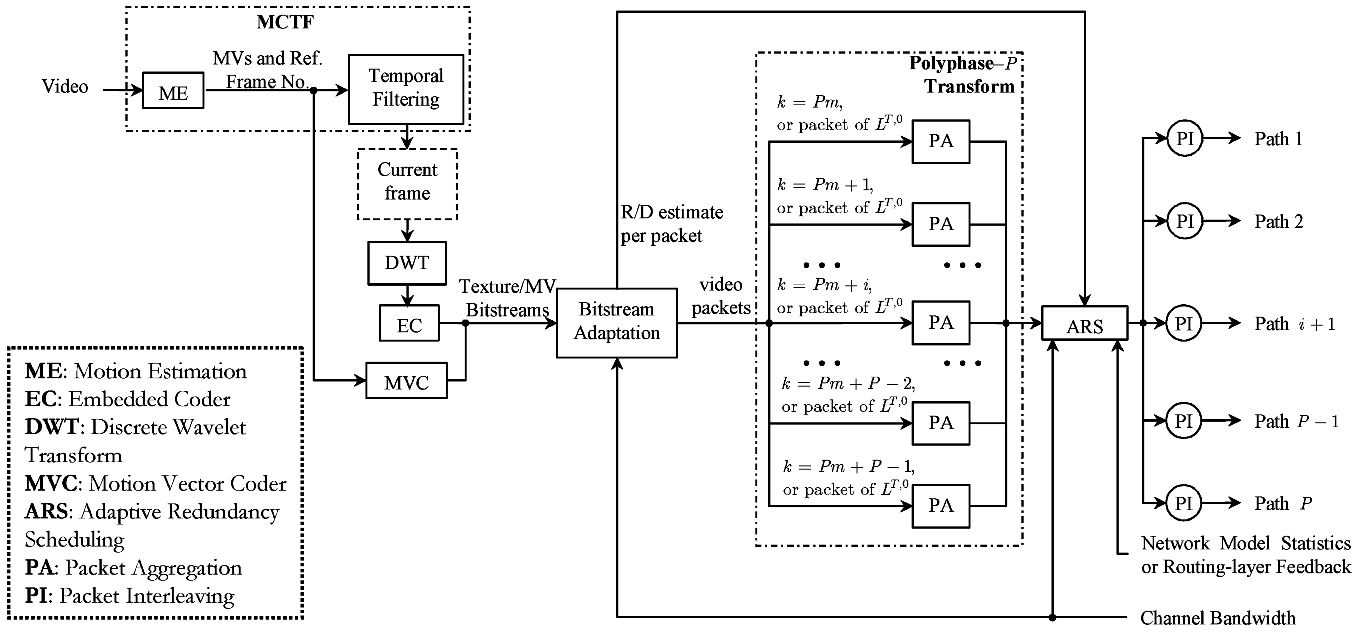


Fig. 6. Application-layer system overview.

the initialization of the packet-separation algorithm described in the following section.

The process of packet aggregation is necessitated by the fact that several video packets generated by the bitstream adaptation process can be very small in size in comparison to the maximum application-packet size. This is illustrated in Table III, where it is shown that the average video-packet size increases according to the temporal level. This is a problem similar to the different video-packet sizes for P- and B- frames in conventional MPEG coding. However, it becomes more significant in our case due to the multilevel temporal decomposition structure of MCTF. Each aggregated packet ap_a is associated with a corresponding rate increment and a list of distortion decrements, denoted by $\Delta r[\mathrm{ap}_a]$, $\mathbf{\Delta d}[\mathrm{ap}_a]$, respectively, defined as

$$\{\Delta r[\mathrm{ap}_a], \mathbf{\Delta d}[\mathrm{ap}_a]\} = \left\{ \sum_{\forall i:\mathrm{vp}_i^{\mathrm{g}} \in \mathrm{ap}_a} \Delta r\left[\mathrm{vp}_i^{\mathrm{g}}\right], \left[\Delta d\left[\mathrm{vp}_i^{\mathrm{g}}\right]\right]_{\forall i:\mathrm{vp}_i^{\mathrm{g}} \in \mathrm{ap}_a} \right\}. \quad (15)$$

Notice that the individual video-packet distortion-decrements in vector $\mathbf{\Delta d}[\mathrm{ap}_a]$ cannot simply be added as in the case of the rate increments, as their contribution to the model estimated distortion is calculated through the hierarchical application of (3), (14), and (12)–(13). In particular, for a group of packets that correspond to the vector of distortion decrements $\mathbf{\Delta d}$, we denote the average estimated distortion-reduction of a reconstructed GOP after T levels of MCTF reconstruction as $\Delta D = \mathcal{T}_T(\mathbf{\Delta d})$. For example, $\mathcal{T}_1(\mathbf{\Delta d})$ is given by (12) and (13), where the expected mean square error $\mathrm{E}\{e_M^2\}$ [with M given in (11)] defined by replacing the result of (3) in (14).

C. Packet-Level Optimized Redundancy of Video via Polyphase Transform and Motion Compensated Temporal Filtering

The proposed video streaming architecture is outlined in Fig. 6. The front end of the proposed system consists of an implementation of the MCTF-based coding process. As shown in Fig. 6, the created video packets are initially separated into P classes, depending on their frame index k within each temporal level [29]. Notice that the video packets of frame $L^{T,0}$ exist on all classes. In this way, an amount of redundancy is initially introduced in each path and independent decoding from any path becomes possible [29]. In MCTF-based coding, the frame $L^{T,0}$ is the equivalent of an intra-frame of conventional MPEG coding: due to the temporal dependencies of MCTF, the existence of certain video packets of $L^{T,0}$ on both paths is a requirement for the independent reconstruction of visually-meaningful results from each path. However, as it will be shown in the next section, the amount of redundancy existing on each path is adaptively established based on network feedback, the proposed path-failure modeling, and the derived distortion-reduction estimates.

Although the polyphase separation creates information to be transmitted on each of the P paths, the essential adaptation process to the paths' characteristics is performed in the adaptive redundancy scheduling (ARS) module (Fig. 6), which is elaborated in more detail in the following subsection. Through ARS, the packets to be transmitted via each path are established. For each of the P paths, the packet interleaving (PI) module adjusts the transmission order of the packets according to their processing order at the decoder. This process is generally transmitting packets in a bottom-up manner through the temporal pyramid, i.e. the packets corresponding to the coarsest temporal levels are transmitted first.

Although the proposed multipath video-transmission architecture is partially motivated by previous work on polyphase-transform MDC [39] and multiple-state encoding with path diversity [23], Fig. 6 shows that two significant conceptual differences exist between the proposed framework and the related work.

Firstly, the proposed method generates redundancy during the bitstream adaptation stage. As a result, there is no loop between the video encoding and the redundancy-generation. Instead, coarse and fine-quantized descriptions of each video GOP are transmitted together by exploiting the layered structure of spatio–temporal embedded coding. In practice, this means that the proposed technique can work equivalently with precompressed content. In terms of applications, apart from the obvious complexity gain from such a process, this separates content generation and content protection for network transmission. This can be very important in today's distributed multimedia environment where content may be compressed and transmitted through various media, some of which may not be requiring advanced protection from channel failures.

A second advantage of the proposed scheme is that it operates directly at the bitstream extraction and packetization stage and, hence, it can respond and adapt faster to network failures and varying bandwidth. In addition, this allows the explicit effects of the packetization process to be taken into account during the optimized adaptation process of ARS. These topics are elaborated in more detail in the following section.

D. *Adaptive Redundancy Scheduling (ARS)*

In this section we present an algorithm for optimized packet scheduling in two different scenarios. The first scenario (Section III-D1) involves the existence of network feedback in terms of failure notifications from the proxy server. The second scenario (Section III-D2) assumes no feedback and optimizes the redundancy level by transmission of a subset of packets on all paths based on each path's probability of failure and the failure duration period. In both cases, we utilize the precomputed rate-distortion hint information for the packet data and no actual parsing or decoding of compressed information takes place. As a result, all the proposed algorithms have low complexity.

In this paper, we denote the P paths via the vector $\mathbf{u}$ (of length P). In general, the commonly-used dyadic temporal decomposition of MCTF fits more naturally to a streaming system with $P = 2$; hence we shall focus on this case for our simulation results. However, since the proposed algorithm is generic, it is presented for an arbitrary number of paths. Moreover, it is important to notice that for $P = 3$, a three-band temporal decomposition can be used for MCTF [31] without apparent sacrifice in coding efficiency. For higher values of P, we note that one can easily extend the proposed schemes for $P = 2^k$, $k \in \mathbb{Z}_+$.

After the video-packet aggregation of the current GOP (Fig. 6), a total of $C_{\mathbf{u}}$ packets, denoted by $\mathrm{ap}_{c_{\mathbf{u}}}^{\mathbf{u}}$ $(1 \le c_{\mathbf{u}} \le C_{\mathbf{u}})$, and their associated with their rate-distortion information $\{\Delta r[\mathrm{ap}_{c_{\mathbf{u}}}^{\mathbf{u}}], \mathbf{\Delta d}[\mathrm{ap}_{c_{\mathbf{u}}}^{\mathbf{u}}]\}$ are generated for each path so that the path bandwidth $R_{\mathbf{u}(i)}$ is matched.[2]

1) Adaptive Packet Scheduling With Network Feedback: The process of adaptive packet scheduling in P paths is performed with a GOP granularity. A buffering scheme at the decoder side ensures that, under constant bitrate transmission, the decoder does not run out of received data. Under this scenario, the worst-case overall transmission and decoding latency was estimated to be around four GOPs,[3] i.e., approximately 2.5 s if we assume a GOP of 16 frames at a frame-rate of 30 frames/s. We assume that a failure-initiation and failure-termination notification is sent from the proxy server upon the detection of link failures or routing instability. The maximum delay for such notification events is set to 200 ms and packet transmission occurs in intervals of 100 ms.

For each GOP, we initially assume that all paths are operational. Hence, initially, any existing redundancy among the packets to be transmitted via all paths is removed. For example, if originally all the packets of frame $L^{T,0}$ are included in all paths (as indicated in Fig. 6), then they are removed from all but one path. Consequently, all the packets of each path are sorted according to their relative distortion-reductions. This is performed by applying the distortion modeling of Section III-B for each packet starting from the last temporal level. If a packet has dependencies, we assume these packets to be present at the decoder side so as to establish the relative distortion-reduction of each individual application-layer packet under the proposed distortion modeling framework. In this way, P separate packet lists are created with the packets included each list having decreasing importance in terms of distortion reduction. For the duration where no path failure is detected, for each path, the packets are prioritized for transmission based on their position in the sorted lists. Once a failure notification is received for one or more paths, the transmission continues through the remaining paths by redistributing the packets of the failed path(s) in all or some of the working paths, starting at the point where the failure was detected.

The redistribution of packets among the existing paths is performed according to the distortion-reduction estimates. Assume that d paths have failed during the transmission of the current GOP, with $1 \le d < P$; for each d, there are $P!/d!(P-d)!$ combinations of failed paths. For each combination m, $1 \le m \le P!/d!(P-d)!$, we denote the vector of successful-paths by $\mathbf{s}_m$ (of length $P-d$) and the vector of unsuccessful (failed) paths by $\mathbf{f}_m$ (of length d). The packet distribution algorithm starts by establishing the paths $\mathbf{f}_m(i)$ and $\mathbf{s}_m(j)$ with the highest and lowest total estimated distortion-reduction, respectively; the total estimated distortion-reduction of a path is established by summing all the individual packet distortion reductions with the distortion modeling framework of Section III-B. At this point, all packets $\mathrm{ap}_{c_{\mathbf{f}_m(i)}}^{\mathbf{f}_m(i)}$ with

$$\forall c_{\mathbf{f}_m(i)} : \mathcal{T}_T\left(\mathbf{\Delta d}\left[\mathrm{ap}_{c_{\mathbf{f}_m(i)}}^{\mathbf{f}_m(i)}\right]\right) \ge \max_{\forall c_{\mathbf{s}_m(j)}} \left\{\mathcal{T}_T\left(\mathbf{\Delta d}\left[\mathrm{ap}_{c_{\mathbf{s}_m(j)}}^{\mathbf{s}_m(j)}\right]\right)\right\} \quad (16)$$

are moved in path $\mathbf{s}_m(j)$. The process continues with the next path from $\mathbf{f}_m$, until all the failed paths have been covered, or the delay constraints for the transmission window are exceeded. Notice that the decision to move packets is invariant to the packet size as, after packet aggregation, the vast majority of packets have the same size, which is determined by the application-layer packet size, as shown in Table III.

[2]In the case of constant-bitrate (CBR) transmission, it is common to assume that, for every path $\mathbf{u}(i)$, $1 \le i \le P$, we have $R_{\mathbf{u}(i)} = R/P$ for the duration of the transmission of each GOP.

[3]We did not take into account the decoder execution time in these measurements as, under real-time decoding, this overhead is minimal versus the overall delay.

TABLE III
THE IMPORTANCE OF PACKET AGGREGATION. IN THE FIRST FOUR COLUMNS, THE AVERAGE (OVER THE ENTIRE SEQUENCE) VIDEO-PACKET SIZE (IN BYTES) IS SHOWN FOR THE FRAMES OF A FOUR-LEVEL TEMPORAL DECOMPOSITION. THE LAST COLUMN DEMONSTRATES THE PACKET SIZE AFTER THE AGGREGATION PROCESS. THE MAXIMUM APPLICATION-LAYER PACKET SIZE WAS SET TO 1000 BYTES. THE SEQUENCE "MAD CYCLIST" WAS USED FOR THIS EXAMPLE (CIF RESOLUTION, 900 FRAMES, EXTRACTED AT 500 kbps WITH THE UTILIZED MCTF CODEC [30])

Temporal level	1	2	3	4	**Aggregated**
Average size (bytes)	135	268	632	982	971

In each case, once the path bandwidth $R_{\mathbf{s}_m(i)}$ is met in a working path $\mathbf{s}_m(j)$, the transmission of the packets of $\mathbf{s}_m(j)$ stops and the remaining packets are discarded. In our experiments, the minimum path-failure duration always exceeded the transmission time of one GOP. As a result, once a path fails during the transmission of the current GOP, no attempt is made to check whether the path has been re-established within the GOP transmission intervals. This is only performed once the packets of the next GOP are processed. Without being overly complex, this algorithm guarantees that all the working paths $\mathbf{s}_m$ can eventually converge to equalized distortions from the addition of packets from the failed paths $\mathbf{f}_m$.

2) Model-Based Optimized Packet Redundancy: This case is based on the proposed path-failure network modeling framework of Section II. If we define the average length of a streaming session to be W s, for any path $\mathbf{u}(i)$, $1 \leq i \leq P$, the modeling framework can provide an estimate for

- the expected number of link failures $F_{\mathbf{u}(i)}(z_{\mathbf{u}(i)})$, with $z_{\mathbf{u}(i)}$ denoting the link number of $\mathbf{u}(i)$ (that has a total of $Z_{\mathbf{u}(i)}$ links) and $1 \leq z_{\mathbf{u}(i)} \leq Z_{\mathbf{u}(i)}$,
- the expected average service-disruption time due to a link-failure, denoted by $t^{\text{idle}}_{\mathbf{u}(i)}$.

Assuming that, for each path, no two links fail at the same time,[4] the estimated disruption time within W ms is given by $t^{\text{idle}}_{\mathbf{u}(i)} \cdot \sum_{z_{\mathbf{u}(i)}=1}^{Z_{\mathbf{u}(i)}} F_{\mathbf{u}(i)}(z_{\mathbf{u}(i)})$. In the "blind" case with no network feedback, we can assume a uniform distribution of link failures within the timeframe of the streaming session. Hence the probability of a path failure for any path $\mathbf{u}(i)$ is

$$q_{\mathbf{u}(i)} = \frac{t^{\text{idle}}_{\mathbf{u}(i)}}{W} \sum_{z_{\mathbf{u}(i)}=1}^{Z_{\mathbf{u}(i)}} F_{\mathbf{u}(i)}\left(z_{\mathbf{u}(i)}\right). \tag{17}$$

Alternatively, if we assume that a failure-termination packet is received for path $\mathbf{u}(i)$ at the server, the probability of a new failure on $\mathbf{u}(i)$ for the k-th GOP after the failure termination is given based on the Weibull distribution with the parameters defined as in Section II. We note that, for any two paths $\mathbf{u}(i)$, $\mathbf{u}(j)$, the probability $q_{\mathbf{u}(i)}$ is independent of $q_{\mathbf{u}(j)}$ only if the paths do not have shared links.[5]

As mentioned before, concerning the actual packets transmitted via each path, independent decoding from *any one* path should be possible. As a result, the most significant packets in terms of distortion-reduction are reproduced on all paths. A simple example is demonstrated in Fig. 6 where the video-packets of frame $L^{T,0}$ are included on all paths. To generalize this concept, we define as *redundancy group* $\mathbf{w}_r$ the vector containing a total of C_r packets transmitted over each path with these packets originally belonging to paths $\mathbf{r}_{\text{path}}$ and their in-path position indicated by the vector $\mathbf{r}_{\text{in-path}}$ (both vectors of length C_r). The optimized estimation of the parameters $\{C_r, \mathbf{r}_{\text{path}}, \mathbf{r}_{\text{in-path}}\}$ that define $\mathbf{w}_r$ is the topic of this subsection. Under the assumption of an existing redundancy group, its distortion-reduction vector is

$$\mathbf{\Delta d}_{\mathbf{w}_r} = \left[\mathbf{\Delta d}\left[\text{ap}^{\mathbf{r}_{\text{path}}(1)}_{\mathbf{r}_{\text{in-path}}(1)}\right], \ldots, \mathbf{\Delta d}\left[\text{ap}^{\mathbf{r}_{\text{path}}(C_r)}_{\mathbf{r}_{\text{in-path}}(C_r)}\right]\right]. \tag{18}$$

Moreover, for the remaining packets received via any path $\mathbf{u}(i)$, the distortion-reduction vector is

$$\mathbf{\Delta d}_{\mathbf{u}(i)} = \left[\mathbf{\Delta d}\left[\text{ap}^{\mathbf{u}(i)}_{c'_{\mathbf{u}(i)}}\right]_{\forall c'_{\mathbf{u}(i)} \in [1, C_{\mathbf{u}(i)}] : \text{ap}^{\mathbf{u}(i)}_{c'_{\mathbf{u}(i)}} \notin \mathbf{w}_r}\right]. \tag{19}$$

As explained before, the total number of packets on each path, $C_{\mathbf{u}(i)}$, is always determined so that the path bandwidth $R_{\mathbf{u}(i)}$ is not exceeded.

During the transmission of each GOP the following situations are possible at the receiver:

- the number of failed paths is zero; this happens with probability $\varphi_0 = \prod_{i=1}^{P}(1 - q_{\mathbf{u}(i)})$;
- d out of P paths are not operational $(1 \leq d < P)$; the probability that a certain combination m $(1 \leq m \leq P!/d!(P-d)!)$ of d failed paths occurs is expressed by $\varphi_{d,m} = \prod_{i=1}^{d} q_{\mathbf{f}_m(i)} \prod_{j=1}^{P-d}(1 - q_{\mathbf{s}_m(j)})$ with $\mathbf{s}_m$, $\mathbf{f}_m$ defined in Section III-D1;
- no path is operational; this happens with probability $\varphi_P = \prod_{i=1}^{P} q_{\mathbf{u}(i)}$.

By definition, for the current GOP, the average distortion-reduction for the case of φ_P is zero, since no packets were received out of any path. Consequently, this case is not included in our optimization problem. As a result, the expected distortion-reduction for the current GOP under all practical situations can be written as

$$\begin{aligned} \Delta D^{\text{GOP}} =& \varphi_0 \mathcal{T}_T\left(\left[\mathbf{\Delta d}_{\mathbf{u}(1)}, \ldots, \mathbf{\Delta d}_{\mathbf{u}(P)}, \mathbf{\Delta d}_{\mathbf{w}_r}\right]\right) \\ &+ \sum_{d=1}^{P-1} \sum_{m=1}^{\frac{P!}{d!(P-d)!}} \varphi_{d,m} \mathcal{T}_T \\ &\times \left(\left[\mathbf{\Delta d}_{\mathbf{s}_m(1)}, \ldots, \mathbf{\Delta d}_{\mathbf{s}_m(P-d)}, \mathbf{\Delta d}_{\mathbf{w}_r}\right]\right). \end{aligned} \tag{20}$$

In the last equation, the explicit packet distortion-reductions may be replaced from (19). As a result, with the help of the proposed distortion modeling framework, (20) provides the expected distortion-reduction occurring from any selected packet transmission scenario for each path. If (17) is used for the path-failure probabilities $q_{\mathbf{u}(i)}$ then we use no dynamic channel

[4]This corresponds to the worst-case scenario.

[5]The case with shared links can be solved in the same manner by establishing the probability of failure per link. However, since the case of multipath transmission with a large number of shared links among the paths will lead to intervals with complete loss of video information (hence no QoS can be guaranteed), it is less interesting for video streaming.

feedback, except for a training session for the establishment of average link failures $F_{\mathbf{u}(i)}(z_{\mathbf{u}(i)})$ and average disruption time $t_{\mathbf{u}(i)}^{\mathrm{idle}}$. Alternatively, if the Weibull distribution is used for the calculation of $q_{\mathbf{u}(i)}$, then we assume that the server receives packets indicating failure-termination events.

We can now define and solve the problem of optimal allocation of packet redundancy as follows.

Problem Statement

For the transmission of the current GOP over P paths, given that, for each path $\mathbf{u}(i)$ $(1 \le d \le P)$ the following data are provided:

A) the probabilities φ_0 and $\varphi_{d,m}$, with $1 \le m \le P!/d!(P-d)!$ and $1 \le d \le P-1$;

B) an initial set of application-layer packets $\mathrm{ap}_{c_{\mathbf{u}}(i)}^{\mathbf{u}(i)}$ allocated at each path $\mathbf{u}(i)$ and their associated rate-distortion reduction estimates $\{\Delta r[\mathrm{ap}_{c_{\mathbf{u}}(i)}^{\mathbf{u}(i)}], \mathbf{\Delta d}[\mathrm{ap}_{c_{\mathbf{u}}(i)}^{\mathbf{u}(i)}]\}$;

C) the bandwidth $R_{\mathbf{u}(i)}$ of each path $\mathbf{u}(i)$ during the GOP transmission interval

determine the optimal redundancy group $\mathbf{w}_r^*$ with parameters $\{C_r^*, \mathbf{r}_{\mathbf{path}}^*, \mathbf{r}_{\mathrm{in-path}}^*\}$, i.e., the subset of packets that should be transmitted via all paths so that the expected distortion-reduction ΔD^{GOP} is maximized.

Having the analytical expression of ΔD^{GOP} from (20) and the predetermined probabilities of path failures, ΔD^{GOP} becomes a function of $\{C_r^*, \mathbf{r}_{\mathrm{path}}^*, \mathbf{r}_{\mathrm{in-path}}^*\}$ and $\mathbf{\Delta d}_{\mathbf{u}(i)}$.

In order to establish the solution, initially all the GOP packets $\mathrm{ap}_{c_{\mathbf{u}}}^{u}$ are sorted together, according to their relative distortion-reductions $\mathcal{T}_T(\mathbf{\Delta d}[\mathrm{ap}_{c_{\mathbf{u}}}^{\mathbf{u}}])$, calculated individually as for Section III-D1. Consequently, the redundancy group consisting of C_r packets (and their corresponding vectors $\mathbf{r}_{\mathrm{path}}$, $\mathbf{r}_{\mathrm{in-path}}$) with the maximum distortion reduction can be found by selecting the first C_r packets of the sorted list of packets. At this point, the optimal solution can be found by adjusting the value of C_r until convergence to the maximum value for ΔD^{GOP} is achieved. The bisection method can be used for this process.

Specifically, we start with an initial value C_r^1 that corresponds to the number of video-packets corresponding to the $L^{T,0}$ frame and a step $\Delta C_{\mathrm{step}}^1 = C_r^1/2$. Then a series of iterations are performed:

$$C_r^i = C_r^{i-1} + k(i)\Delta C_{\mathrm{step}}^i \tag{21}$$

with

$$k(i) = \begin{cases} 1, & \text{if } \Delta D^{\mathrm{GOP}}(C_r^i) > \Delta D^{\mathrm{GOP}}(C_r^{i-1}); \\ -1, & \text{if } \Delta D^{\mathrm{GOP}}(C_r^i) < \Delta D^{\mathrm{GOP}}(C_r^{i-1}); \\ 0, & \text{otherwise} \end{cases}$$

and

$$\Delta C_{\mathrm{step}}^i = \begin{cases} (\Delta C_{\mathrm{step}}^{i-1}/2), & \text{if } k(i) \ne k(i-1); \\ \Delta C_{\mathrm{step}}^{i-1}, & \text{otherwise} \end{cases}$$

until convergence is achieved, i.e. $k(i) = 0$. For each iteration the vectors $\mathbf{r}_{\mathrm{path}}$, $\mathbf{r}_{\mathrm{in-path}}$, $\mathbf{\Delta d}_{\mathbf{u}}$ and $\mathbf{\Delta d}_{\mathbf{w}_r}$ are calculated based on the sorted list of GOP packets and the bandwidth constraints R_{u}. Since we are only modifying the value of C_{r} and then calculate the expected GOP distortion based on (20), under the context of the proposed solution the obtained maximum value for ΔD^{GOP} is guaranteed to be the global maximum if $\Delta D^{\mathrm{GOP}}(C_r)$ approximates a polynomial function of maximum order two. By applying curve fitting techniques to the experimentally-derived points of $\Delta D^{\mathrm{GOP}}(C_r)$ incurred by all the GOPs of the tested video material, this assumption was verified empirically.

IV. Performance Evaluation

A. Accuracy of the Distortion Estimation

In order to evaluate the accuracy of the proposed distortion-estimation algorithm, a number of typical test sequences were encoded with the utilized MCTF-based video coder [30]. The codec uses advanced multihypothesis prediction and a normalization process for the update step [30], thereby producing different pixel weighting factors for the temporal filtering of each GOP. In our experiments, we used the 5/3 temporal filter with a maximum of two hypotheses and variable-block size prediction with block sizes ranging from 64×64 down to 4×4 pixels. The temporal decomposition was applied for a total of four levels, and the search range for the motion estimation was set to ± 16 pixels for all temporal levels, with the full-search (exhaustive) algorithm. This configuration encapsulates one of the most generic instantiations of MCTF-based coding. Concerning the spatial transform and entropy coding, the 9/7 filter-pair was used for each L and H frames; embedded quantization and entropy coding of the wavelet representation of each frame was subsequently applied as described in our previous work [30]. During the encoding, rate-distortion estimates are collected for each frame at the end of each fractional-bitplane pass [27]. The utilized codec allows for bitstream extraction at a number of bitrates; for each case, the packetization process separates the bitstream of each frame into video packets and associates them with their corresponding rate increment and distortion decrement based on the accumulated rate-distortion information, as explained in Section III-B.

Fig. 7 demonstrates four typical examples of the match between the model-based distortion estimation and the actual PSNR results generated by decoding each sequence. As mentioned before, the model estimation is based on the weighting of the distortion estimates for the distortion-reduction incurred by processing each packet. In practice, we found that the model-estimated distortion tends to overestimate the actual decoding PSNR. For this reason, the model-estimated results of Fig. 7 were (uniformly) linearly scaled to the maximum PSNR measurement of each sequence. It is important to notice that the scaling performed for each sequence does not affect the proposed packet scheduling algorithms of the previous section, as this corresponds to a linear scaling of the distortion-reduction estimates used in (16) and (20), and hence does not affect the relative distortion-comparison operations performed during the iterations of (21).

As shown in the figure, after this scaling, the model-based distortion estimation predicts the transient behavior of the decoded-frames distortion relatively well. This is quantified by the

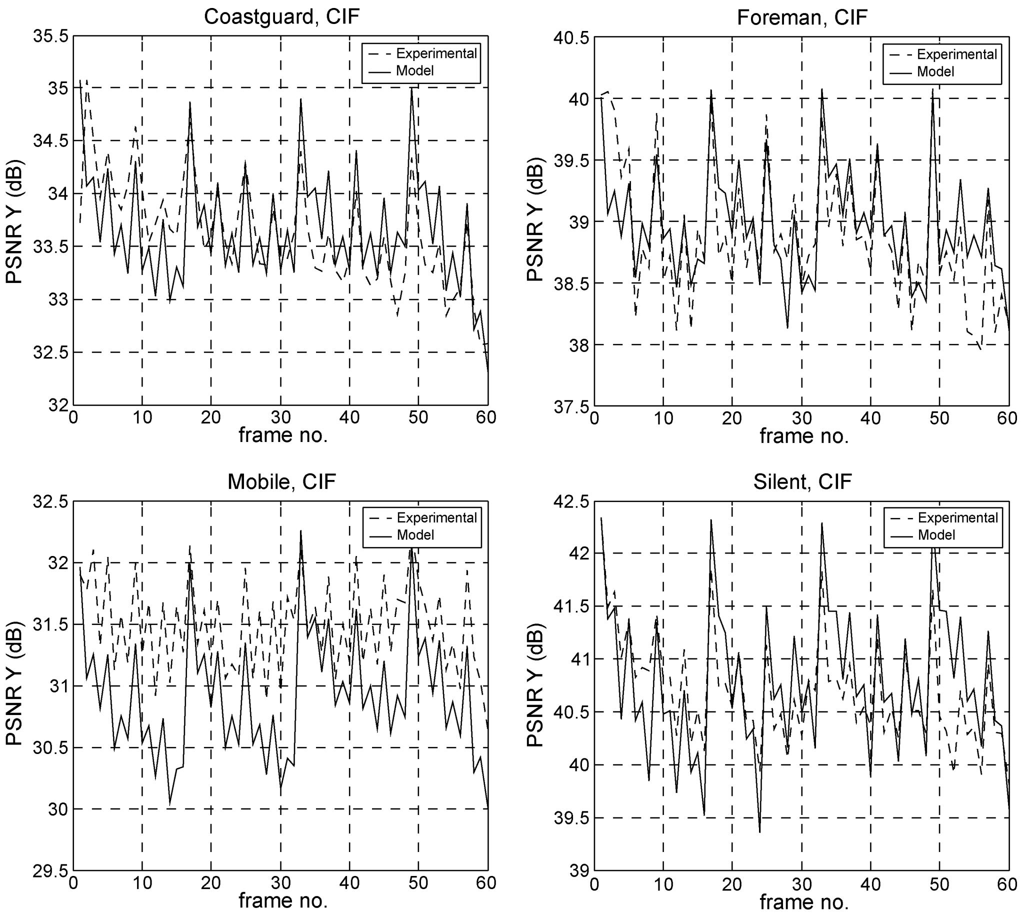


Fig. 7. Examples of PSNR estimation with the proposed model. The first 60 frames of four typical CIF sequences are demonstrated; all sequences were extracted at 1024 kbps and the "Experimental" PSNR was measured from each decoded sequence. All the "Model" estimated points for each sequence were uniformly scaled to the maximum PSNR measurement.

TABLE IV
MEAN ABSOLUTE ERROR BETWEEN THE UNIFORMLY-SCALED MODEL-BASED DISTORTION ESTIMATE AND THE REAL DISTORTION, AS MEASURED FOR TWO REPRESENTATIVE BITRATES IN FOUR CIF TEST SEQUENCES

Sequence	Bitrate (kbps)	Mean Absolute Error (MAE) Between Experimental and Model-generated PSNR (dB)	MAE Variance
Coastguard	256	0.358	0.068
	1024	0.363	0.078
Foreman	256	0.270	0.042
	1024	0.259	0.044
Mobile	256	0.345	0.064
	1024	0.571	0.091
Silent	256	0.336	0.070
	1024	0.347	0.070
Average	**256**	**0.327**	**0.061**
	1024	**0.385**	**0.071**

first and second sample moments of the absolute error between the experimental and theoretical results, presented in Table IV.

B. Simulating Realistic Network Dynamics

We built a java-based simulator to emulate intra-domain routing dynamics in the presence of link failures and to implement the algorithm used to compute SD time (Table II). The inputs to the simulator are complete network topology specifications, BGP prefix distribution, and traffic load along different links in the network. We categorize the network nodes as large, medium and small, depending on the traffic amount they generate.

We consider 20% of the nodes as large nodes, 30% as medium nodes and the rest as small nodes. The traffic matrix for all the networks is generated using this model. This network model is based on PoP (Point of Presence) level network topologies of various tier-1 ISPs. We distribute BGP prefixes proportional to the traffic between nodes, i.e., large traffic flow from a source node to destination node implies that the source node reaches a large number of prefixes in the Internet through the destination

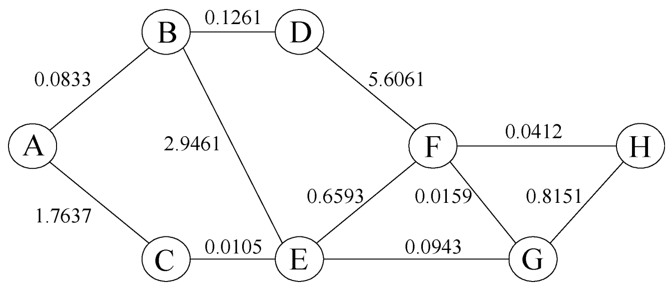


Fig. 8. Considered network topology.

TABLE V
THE THREE DIFFERENT PATHS THAT WERE USED IN OUR SIMULATIONS AND THE AVERAGE SERVICE DISRUPTION TIME DUE TO A LINK FAILURE ON EACH PATH

Path #	Links in Path	t^{idle}(msec)
1	A→C→E→G→H	3491.9
2	A→B→E→F→G→H	3196.9
3	A→B→D→F→H	3111.3

node. Based on these inputs, the simulator runs Dijkstra's SPF algorithm to find the shortest path from every node to all other nodes in the network. It then simulates single link failures and executes Dijkstra's SPF algorithm again, to find new paths in the network. The SD time for various paths are calculated based on the algorithm of Table II.

We use a scenario where the packet size is 1000 bytes and each path has a fixed (constant) bandwidth of 256 Kbps. Hence, in total we can achieve 512 Kbps (CBR) by using both paths. The duration of the streaming session was set to $W = 3\cdot10^5$ ms. The network topology that was considered is depicted in Fig. 8, which is a subgraph of an ISP topology. The numbers indicated on each link show the average number of failures of the link during the duration of the streaming session. The time between failures of links in the network was considered to follow a Weibull distribution.

The simulated paths are shown in Table V. The table also presents the average expected service disruption time due to a link failure as calculated by the algorithm of Table II. Notice that paths 1 and 3 do not have any common links, while paths 2 and 3 share link $\mathrm{A} \rightarrow \mathrm{B}$.

C. Performance of Streaming With Optimized Redundancy

In our experiments, we used two standard MPEG test sequences provided for streaming simulation purposes in the recent MPEG Call for Evidence Test on scalable video coding [28], namely "Mad Cyclist" (CIF resolution) and "Big Ships" (SIF resolution). The test sequences have smaller duration than the length of the streaming session. As a result, they were repeated in order to cover the duration of W ms. Moreover, each streaming simulation was performed a number of times, in order to incorporate different network behavior in terms of path failures. The two different schemes proposed in Sections III-D1 and III-D2 were simulated and the results were compared in terms of mean PSNR values. In addition, for comparison purposes, we used a scheme that inserts fixed redundancy in the bitstreams of each path by reproducing a certain percentage of the most significant (in terms of distortion-reduction) texture and motion-vector packets. The redundancy percentage was set to 15%, since we experimentally verified that this redundancy ensures that at least one packet exists on both paths for the L-frame of each GOP. In this way, we avoid the effect of loosing the L-frame information completely, which would lead to very bad picture quality. In all cases, in the case of lost motion-vector information for error frame k of temporal level t, the motion vector data of error frame $k/2$ of temporal level $t+1$ was used. If the motion-vector data of that frame was lost as well, the motion vectors are set to zero. Although more advanced concealment techniques may be envisaged, the utilized approach performed well for our comparison purposes.

An overview of our obtained results can be seen in Table VI. The adaptive method utilizing failure notifications from the proxy server (APS) performs best. The proposed model-based optimized packet redundancy (OPR), which uses (17) with the derived SD times and link-failure probabilities (Table V and Fig. 8, respectively), underperforms APS by approximately 0.6 dB on average (in terms of mean PSNR). However, it also outperforms the ad-hoc method that uses a fixed-redundancy percentage (FRP) by approximately 1.0 dB. In order to illustrate the impact of distortion-modeling in the proposed adaptive streaming schemes, "FRP no DM" present the obtained results when using the FRP method without packet distortion estimates. In this case, we are prioritizing the packets in a heuristic manner according to their dependencies, the frame type they belong to, and the frame's temporal decomposition level using an ad-hoc weighting scheme for the packet significance. The results demonstrate that, even if smart heuristics are used for packet prioritization, the proposed distortion-modeling framework benefits the streaming scheme by approximately 0.7–0.8 dB on average under the same transmission scenario.

We note that, in contrast to the APS scheme which uses explicit network feedback, both OPR and FRP schemes do not require any network feedback and rely solely on network and decoding-distortion modeling. Moreover, the results of Table VI demonstrate that, concerning the average PSNR after multiple runs, each method performs almost identically over the two different pairs of paths. This is explained by the fact that the percentage of time when a simultaneous failure occurred in links of both paths tends to be very small in comparison to the streaming-session interval.

Indicative average PSNR results across intervals of 30 s can be seen in Fig. 9. The results demonstrate that the proposed OPR systematically outperforms the ad-hoc scheme that sets a fixed amount of redundancy. Moreover, for the vast majority of cases, distortion-modeling benefits the proposed framework versus heuristic techniques.

A low packet redundancy benefits the proposed approaches since less sacrifice in coding efficiency is made for the time intervals that are failure free. As a result, on average, the OPR approach converges to the use of low redundancy. The average redundancy percentage used in the OPR approach was found to be 4% and 3.5% for "Mad Cyclist" and "Big Ships," respectively. However, the quality degradation during a path failure can become more severe in this case in comparison to the APS scheme and the FRP scheme. This is demonstrated in Table VII, where within the duration of a path failure, the quality provided by the

TABLE VI
AVERAGE PSNR FOR LUMINANCE AND CHROMINANCE CHANNELS AND MEAN PSNR FOR TWO TEST SEQUENCES AND TWO COMBINATIONS OF PATHS. THE MEAN PSNR IS DEFINED AS $\mathrm{mean\_PSNR} = (1/6)(4\mathrm{PSNR\_Y} + \mathrm{PSNR\_U} + \mathrm{PSNR\_V})$; APS REPRESENTS THE ADAPTIVE PACKET SCHEDULING TECHNIQUES BASED ON ROUTING-LAYER NOTIFICATIONS (SECTION III-D1); OPR REPRESENTS THE OPTIMIZED PACKET REDUNDANCY BASED ON SECTION III-D2; FRP REPRESENTS THE FIXED REDUNDANCY PERCENTAGE SCHEME, WHICH IS DEVISED AS A BASELINE FOR COMPARISON PURPOSES; "FRP, NO DM" REPRESENTS THE FRP SCHEME WITHOUT DISTORTION MODELING

Sequence	Pair of Paths used	Method	PSNR (Y)	PSNR (U)	PSNR (V)	Mean PSNR
Mad Cyclist	1, 3	APS	29.50	37.01	39.67	**32.45**
		OPR	28.35	35.99	39.05	**31.41**
		FRP	27.13	35.12	38.44	**30.35**
		FRP, no DM	26.21	34.81	37.61	**29.54**
	2, 3	APS	29.49	37.02	39.68	**32.44**
		OPR	28.35	36.03	39.07	**31.41**
		FRP	27.12	35.12	38.44	**30.34**
		FRP, no DM	26.20	34.81	37.58	**29.53**
Big Ships	1, 3	APS	35.66	42.08	43.43	**38.03**
		OPR	35.36	41.95	43.31	**37.78**
		FRP	34.13	41.33	42.87	**36.79**
		FRP, no DM	33.25	40.91	42.52	**36.07**
	2, 3	APS	35.62	42.04	43.39	**37.98**
		OPR	35.38	41.94	43.31	**37.80**
		FRP	34.16	41.35	42.87	**36.81**
		FRP, no DM	33.21	40.90	42.53	**36.05**

TABLE VII
AVERAGE PSNR VALUES DURING FAILURE FREE PERIODS, AND DURING THE PERIOD WITH THE MAXIMUM-LENGTH PATH FAILURE

Sequence	Interval (30 sec)	APS Mean PSNR (dB)	OPR Mean PSNR (dB)	FRP Mean PSNR (dB)	FRP, no DM Mean PSNR (dB)
Mad Cyclist	Failure-free	32.78	31.99	30.26	29.99
	Failures occurred	32.06	31.46	30.03	29.38
Big Ships	Failure-free	38.41	38.09	36.84	35.64
	Failures occurred	37.54	36.43	36.65	35.20

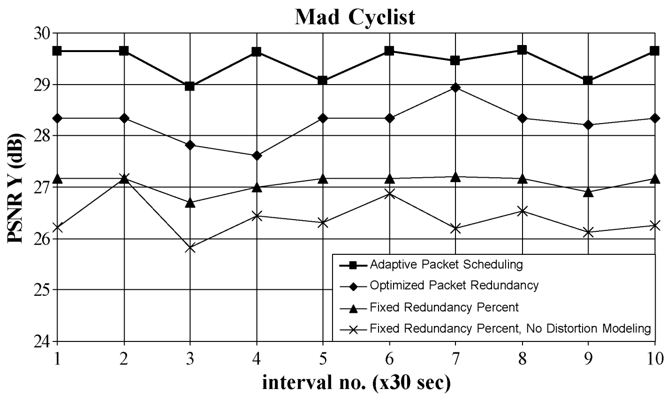


Fig. 9. Average PSNR results in intervals of 900 frames (30 s).

OPR method drops more severely than in the other methods in the "Big Ships" sequence. As a result, it can be said that the proposed model-based OPR approach presents a feedback-free mechanism that provides high quality for the failure-free periods, which is comparable to an adaptive method requiring network feedback; moreover, without specific knowledge of the occurrence of a failure, good visual quality can be provided within the failure duration.

V. CONCLUSIONS

In this paper a new joint source-network framework for video transmission over the Internet is presented. In terms of network modifications, we propose to deploy routing proxies to detect link or node failures directly. Upon failure detection, the proxy sends explicit notification to the multimedia server to report the starting time and location of a failure event or the restoration of a path. We also propose and develop a novel technique to estimate service disruption time after a failure event by taking into account transient routing dynamics during the convergence period of IGP. To take advantage of this accurate network information and adapt at a minimum complexity cost for the server, we deploy a novel adaptive scheme, which can generate redundancy at the packet level by exploiting the open-loop motion-compensated temporal filtering structure of state-of-the art video coders with the aid of distortion modeling for the receiver video quality. The adaptation can be performed based on real-time network information or based on a realistic model that can estimate the network behavior by tuning the redundancy factor for an improved video quality depending on the network condition. Our results indicate gains in performance as opposed to the fixed redundancy schemes that do not consider explicitly the network information or the expected distortion at the receiver.

REFERENCES

[1] Z. Miao and A. Ortega, "Optimal scheduling for streaming of scalable media," in *Proc. Asilomar Conf. Signals, Syst. and Comp.*, Nov. 2000, vol. 2, pp. 1357–1362.
[2] P. Frossard and O. Verscheure, "Joint source/fec rate selection for quality-optimal MPEG-2 video delivery," *IEEE Trans. Image Processing*, vol. 10, pp. 1815–1825, Dec. 2001.

[3] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Trans. on Multimedia*, vol. 8, no. 2, pp. 390–404, Apr. 2006.

[4] C. Boutremans, G. Iannaccone, and C. Diot, "Impact of link failures on VoIP performance," in *Proc. ACM Int. Workshop Netw., Oper. Syst. Sup. for Digit. Audio and Video, NOSSDAV-02*, May 2002.

[5] G. Iannaccone, C.-N. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures in an IP backbone," in *Proc. ACM SIGCOMM Internet Meas. Workshop*, Nov. 2002.

[6] C. Labovitz, A. Ahuja, A. Abose, and F. Jahanian, "An experimental study of BGP convergence," in *Proc. ACM SIGCOMM Internet Meas. Workshop*, 2000.

[7] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proc. ACM Internet Meas. Workshop*, 2000.

[8] A. Markopoulou, G. Iannaccone, S. Bhattacharrya, C. N. Chuah, and C. Diot, "Characterization of failures in an IP Backbone network," in *Proc. IEEE Infocom*, 2004.

[9] T. Nguyen, P. Mehra, and A. Zakhor, "Path diversity with forward error correction (pdf) system for packet switched networks," in *Proc. IEEE Infocom*, April 2003.

[10] "Python routing toolkit," [Online]. Available: http://ipmon.sprint.com/pyrt/

[11] "GNU Zebra, free routing software," [Online]. Available: http://www.zebra.org/

[12] D. Oran, "OSI IS-IS intra-domain routing protocol," RFC 1142, Feb. 1990.

[13] J. Moy, "OSPF Version 2," RFC 2328, April 1998.

[14] M. R. Civanlar, "Internet video," in *Advances in Multimedia: Systems, Standards, and Networks*. : Marcel Dekker, 2000.

[15] R. Rejaie, M. Handley, and D. Estrin, "Quality adaptation for congestion controlled video playback over the Internet," in *Proc. ACM SIGCOMM Internet Meas. Workshop*, 1999.

[16] J. Chakareski and B. Girod, "Computing rate-distortion optimized policies for streaming media with rich acknowledgements," in *Proc. IEEE Data Compr. Conf., DCC-04*, Mar. 2004, pp. 202–211.

[17] D. Rubenstein, J. Kurose, and D. Towsley, "Detecting shared congestion of flows via end-to-end measurement," *IEEE/ACM Trans. Networking*, vol. 10, no. 3, pp. 381–395, Jun. 2002.

[18] J. Chakareski, J. Apostolopoulos, W. Tan, S. Wee, and B. Girod, "Distortion chains for predicting the video distortion for general packet loss patterns," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, May 2004, vol. 5, pp. 1001–1004.

[19] K. Stulmuller, N. Farber, M. Link, and B. Girod, "Analysis of video transmission over lossy channels," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 1012–1032, Jun. 2000.

[20] U. Hengartner, S. B. Moon, R. Mortier, and C. Diot, "Detection and analysis of routing loops in packet traces," in *Proc. ACM SIGCOMM Internet Measurement Workshop*, Nov. 2002.

[21] G. Iannaccone, C.-N. Chuah, S. Bhattacharyya, and C. Diot, "Feasibility of IP restoration in a tier-1 backbone," *IEEE Network*, Mar. 2004.

[22] V. Goyal, "Multiple description coding: compression meets the network," *IEEE Signal Process. Mag.*, no. 9, pp. 74–93, Sept. 2001.

[23] J. G. Apostolopoulos, "Reliable video communication over lossy packet networks using multiple state encoding and path diversity," in *Visual Commun. and Image Proc., VCIP'01*, Jan. 2001.

[24] Y. Wang and S. Lin, "Error resilient video coding using multiple description motion compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 438–452, June 2002.

[25] R. Puri, K.-W. Lee, K. Ramchandran, and V. Bharghavan, "Forward error correction (FEC) codes based multiple description coding for Internet video streaming and multicast," *Signal Process.: Image Commun.*, vol. 16, no. 8, pp. 745–762, May 2001.

[26] A. Secker and D. Taubman, "Motion-compensated highly scalable video compression using an adaptive 3D wavelet transform based on lifting," in *Proc. IEEE Int. Conf. Image Proc., ICIP 2001*, Oct. 2001, vol. 2, pp. 1029–1032.

[27] D. Taubman and M. Marcellin*, JPEG2000: Image Compression Fundamentals, Standards and Practice*. Norwell, MA: Kluwer, 2002.

[28] *Call for Evidence on Scalable Video Coding Advances*, IEC JTC1/SC29/WG11, w5559, MPEG, Pattaya, Thailand, March 2003.

[29] M. van der Schaar and D. Turaga, "Multiple description scalable coding using wavelet-based motion compensated temporal filtering," in *Proc. IEEE Intern. Conf. Image Proc. ICIP-03*, Sept. 2003, vol. 2, pp. 489–892.

[30] Y. Andreopoulos, A. Munteanu, J. Barbarien, M. van der Schaar, J. Cornelis, and P. Schelkens, "In-band motion compensated temporal filtering," *Signal Process.: Image Commun., Special Issue on Subband/Wavelet Interframe Video Coding*, vol. 19, no. 7, pp. 653–673, Aug. 2004.

[31] C. Tillier, B. Pesquet-Popescu, and M. van der Schaar, "Highly scalable video coding by bidirectional predict-update 3-band schemes," in *Proc. IEEE Int. Conf. Accoust., Speech, and Signal Process.*, May 2004, vol. 3, pp. 125–128.

[32] P. A. Chou, H. J. Wang, and V. N. Padmanabhan, "Layered multiple description coding," in *Proc. Packet Video Workshop*, Nantes, France, Apr. 2003.

[33] P. Chen and J. W. Woods, "Bidirectional MC-EZBC with lifting implementation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 10, pp. 1183–1194, Oct. 2004.

[34] L. Luo, F. Wu, S. Li, Z. Xiong, and Z. Zhuang, "Advanced motion threading for 3D wavelet video coding," *Signal Process: Image Commun., Special Issue on &ldquo;Subband/Wavelet Interframe Video Coding*, vol. 19, no. 7, pp. 601–616, Aug. 2004.

[35] T. Rusert, K. Hanke, and J.-R. Ohm, "Transition filtering and optimized quantization in interframe wavelet video coding," in *Proc. SPIE Vis. Commun. and Image Process., VCIP '03*, 2003, pp. 682–693.

[36] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.

[37] H. Schwarz, T. Hinz, H. Kirchhoffer, D. Marpe, and T. Wiegand, Technical Description of the HHI Proposal for SVC CE1 , ISO/IEC JTC1/SC29/WG11 (MPEG), m11244, Oct. 2004.

[38] A. C. Begen, Y. Altunbasak, O. Ergun, and M. H. Ammar, "Multipath selection for multiple description video streaming over overlay networks," *Signal Process.: Image Commun.*, vol. 20, no. 1, pp. 39–60, Jan. 2005.

[39] W. Jiang and A. Ortega, "Multiple description coding via polyphase transform and selective quantization," in *Proc. Visual Commun. Image Proc. (VCIP-99)*, San Jose, CA, Jan. 1999.

[40] R. Keralapura, C.-N. Chuah, G. Iannaccone, and S. Bhattacharyya, "Service availability: a new approach to characterize IP backbone topologies," in *Proc. IEEE Intern. Workshop on Quality of Service, IWQoS'04*, Jun. 2004.

[41] A. Munteanu, Y. Andreopoulos, M. van der Schaar, P. Schelkens, and J. Cornelis, "Control of the distortion variation in video coding systems based on motion compensated temporal filtering," in *Proc. IEEE Internat Conf. on Image Process., ICIP'03*, vol. 2, pp. 61–64.

[42] D. Anderson, H. Balakrishna, M. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. ACM Symp. Operating Syst. Principl. (SOSP)*, Oct. 2001.

[43] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Homan, J. Snell, A. Vahdat, G. Voelker, and J. Zahorjan, "Detour: a case for informed internet routing and transport," *IEEE Micro*, Jan. 1999.

[44] M. Castro, P. Druschel, A. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: high-bandwidth multicast in cooperative environments," in *Proc. ACM Symp. Operating Syst. Principl. (SOSP)*, Oct. 2003.

[45] J. Jannotti, D. K. Giford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, Jr., "Overcast: reliable multicasting with an overlay network," in *Proc. ACM Symp. Operating Syst. Design and Implementation (OSDI)*, Oct. 2000.

[46] J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman, and B. Weihl, "Globally distributed content delivery," *IEEE Internet Comput.*, vol. 6, no. 5, pp. 50–58, Sep. 2002.

Yiannis Andreopoulos (M'00) was born in 1977 in Aeghion, Greece. He received the electrical engineering diploma and an M.Sc. degree in signal and image-processing systems from the University of Patras, Patras, Greece, and the Ph.D. degree from the Vrije Universiteit Brussel, Brussels, Belgium.

Since October 2006, he is a Lecturer at the Queen Mary University of London, U.K. From May 2005 to September 2006, he was a Post-Doctoral Researcher at the University of California, Davis, and the University of California, Los Angeles. His research interests are in the fields of transforms, multimedia architectures and complexity modeling, video coding, and multimedia transmission through unreliable media, e.g., wireless networks and the Internet.

Dr. Andreopoulos contributed to the ISO/IEC JTC1/SC29/WG11 (MPEG) committee (Scalable Video Coding group) during 2002-2003.

Ram Keralapura received the B.E. degree in electrical engineering from Bangalore University, India, in 1998 and the M.S. degree in computer science from the University of Alabama, Huntsville, in 2000. He is currently pursuing the Ph.D. degree in the Electrical and Computer Engineering Department at the University of California, Davis.

From 2001 to 2002, he was a Software Engineer on optical switching technology at Sycamore Networks. His current research interests are in designing, building, analyzing, and managing distributed networked systems. He is also interested in Internet routing, traffic engineering, network security, cross-layer protocol design, and overlay/P2P networks.

Mihaela van der Schaar (SM'04) received the Ph.D. degree from Eindhoven University of Technology, The Netherlands, in 2001.

Prior to joining the Electrical Engineering Department faculty at the University of California, Los Angeles (UCLA) on July 1, 2005, between 1996 and June 2003, she was a Senior Researcher at Philips Research in the Netherlands and the USA, where she led a team of researchers working on multimedia coding, processing, networking, and streaming algorithms and architectures. From July 1, 2003 until July 1, 2005, she was an Assistant Professor in the Electrical and Computer Engineering Department at the University of California, Davis. She has published extensively on multimedia compression, processing, communications, networking and architectures and holds 27 granted U.S. patents and several more pending. Since 1999, she was an active participant to the ISO Motion Picture Expert Group (MPEG) standard to which she made more than 50 contributions and for which she received three ISO recognition awards. She was also chairing for three years the *ad-hoc* group on MPEG-21 Scalable Video Coding, and also co-chairing the MPEG *ad-hoc* group on Multimedia Test-bed.

Dr. van der Schaar was elected as a Member of the Technical Committee on Multimedia Signal Processing of the IEEE Signal Processing Society. She was an Associate Editor of IEEE TRANSACTIONS ON MULTIMEDIA and SPIE *Electronic Imaging Journal* from 2002- to 2005. Currently, she is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (T-CSVT) and an Associate Editor of the IEEE SIGNAL PROCESSING LETTERS. She received the NSF CAREER Award in 2004, IBM Faculty Award in 2005, and the Best Paper Award for her paper published in 2005 in the IEEE T-CSVT.

Chen-Nee Chuah (SM'06) received the B.S. degree in electrical engineering from Rutgers University, Piscataway, NJ, in 1995, and the M.S. and Ph.D. degrees in electrical engineering and computer sciences from the University of California, Berkeley, in 1997 and 2001, respectively.

She is currently an Associate Professor in the Electrical and Computer Engineering Department at the University of California, Davis (UCD). Before joining UCD, she held a visiting researcher position at Sprint Advanced Technology Laboratories. Her research interests are in the area of computer networking and distributed systems, Internet measurements, peer-to-peer systems, wireless/mobile networking, network security and performance modeling.

Dr. Chuah received the National Science Foundation CAREER Award in 2003 and the UC Davis College of Engineering Outstanding Junior Faculty Award in 2004. She has served on the technical program committee of several ACM and IEEE conferences and workshops (including INFOCOM, MOBICOM, SECON, IWQoS, and ICC). She received the ACM Recognition Award for co-organizing the First Workshop on Vehicular Ad Hoc Network (VANET), which was held in conjunction with ACM MobiCom 2004. She is currently serving as the TPC vice-chair for IEEE GLOBECOM 2006.