Streaming-Viability Analysis and Packet Scheduling for Video Over In-Vehicle Wireless Networks

Qiong Li, *Member, IEEE*, Yiannis Andreopoulos, *Member, IEEE*, and Mihaela van der Schaar, *Senior Member, IEEE*

***Abstract*—State-of-the-art vehicles are now being equipped with multiple video channels for video-data transmission from multiple surveillance cameras mounted on the automobile, navigation videos reporting the traffic conditions on the planned route, as well as entertainment-multimedia streaming for passengers watching on rear-seat monitors. Wireless LANs provide a low-cost and flexible infrastructure for these emerging in-vehicle multimedia services aimed at the driver's and passengers' safety, convenience, and entertainment. To enable the successful simultaneous deployment of such applications over in-vehicle wireless networks, we propose delay-sensitive streaming and packet-scheduling algorithms that enable simple, flexible, and efficient adaptation of the video bitstreams to the instantaneously changing video source and wireless-channel characteristics while complying with the *a priori* negotiated quality-of-service (QoS) parameters for that video service. Our focus is on real-time low-cost solutions for multimedia transmission over in-vehicle wireless networks that are derived based on existing protocols defined by QoS-enabled networks, such as the IEEE 802.11e standard. In addition, the aim of this paper is to couple the proposed solutions with a novel multitrack-hinting method that is proposed as an extension of conventional MP4 hint tracks in order to provide real-time adaptation of multimedia streams to multiple quality levels for different in-vehicle applications, depending on their importance and delay constraints. First, the scheduling constraints for these simultaneous wireless video-streaming sessions are analytically expressed as a function of the negotiated QoS parameters. This is imperative because a video stream received from an in-vehicle road-surveillance camera will have a different set of delay and quality constraints in comparison to that of traffic monitoring received from remote video cameras located on the planned route. Hence, transmission parameters, such as peak data rate, maximum burst size, minimum transmission delay, maximum error rate, etc., will differ for the various video streams. For this reason, new low-complexity packet-scheduling algorithms that can fulfill diverse QoS streaming conditions are proposed and analyzed. The proposed algorithms produce viable schedules (i.e., strictly QoS-compliant) that jointly consider the delay constraints and the in-vehicle video-receiver-buffer conditions. Hence, these scheduling schemes can completely avoid the underflow or overflow event of the receiving-device buffer while guaranteeing the agreement between the real-time video traffic and the predetermined traffic specification reached during QoS negotiation for various in-vehicle video channels. When combined with multitrack hinting, an integrated flexible system for adaptive multimedia streaming over QoS-enabled in-vehicle wireless networks can be constructed. We demonstrate the viability of the proposed scheduling mechanisms experimentally by using real video traces under multiple quality levels, as derived by the multitrack-hinting design. In addition, simulations under realistic conditions are also performed to validate the ability of the method to satisfy buffer-occupancy constraints.**

***Index Terms*—In-vehicle wireless networks, packet scheduling, quality of service (QoS), video streaming.**

I. Introduction

MULTIMEDIA-STREAMING applications over wireless networks have already been deployed in homes, campuses, and offices over the past several years. Recently, this trend is starting to extend to high-end vehicles, where multiple audiovisual applications are now deployed at a commercial level [1]. At the same time, wireless support in vehicles is becoming popular due to the cost decrease of wireless-LANs infrastructures, the ease-of-service, and the reduction of wiring requirements, all of which are very attractive for vehicle applications [2]–[6]. Multimedia services in vehicles provide a large range of informational services for the driver and passengers such as multiview road-surveillance videos from multiple cameras mounted on the vehicle, video shots highlighting the traffic conditions on the roads and highways on the planned route [1], entertainment-multimedia applications for in-seat passengers [4], [5], etc. Each of these video streams has a different importance (e.g., visual aids for safety and vehicle guidance versus in-vehicle entertainment); hence, different quality-of-service (QoS) guarantees are required for each video stream. Consequently, successful deployment of multiple real-time multimedia applications over such in-vehicle QoS-enabled wireless channels is expected to be very challenging.

Various solutions have been developed for multimedia transmission over QoS-enabled wireless networks at different layers of the protocol stack (see [7]–[11] for a review on this topic). For instance, the IEEE 802.11e standard [12] has adopted an admission-control mechanism based on which multimedia applications can reserve time for transmitting their bitstreams during each service interval. The reservation is performed statically, prior to the actual transmission, by declaring its

Manuscript received August 21, 2006; revised January 7, 2007 and March 26, 2007. This work was supported in part by the National Science Foundation under Grant CCF 0541867 and Grant CNS 0509522 and in part by the University of California under the UC MICRO Program. The review of this paper was coordinated by Dr. W. Zhuang.

Q. Li is with the Diabetes Care Division, Bayer HealthCare, Tarrytown, NY 10591 USA (e-mail: qiong.li.b@bayer.com).

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

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

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2007.901927

multimedia traffic specification (TSPEC). This allocation strategy guarantees that the resources are divided among the participating wireless transmitters based on their TSPEC parameters. Hence, each wireless application will need to adhere to this negotiated TSPEC, independent of its instantaneous channel conditions or bitstream (traffic) characteristics.

Similarly, international telecommunications standardization committees [14], [15], as well as existing overlay network infrastructures [30], [31], provide application-program interfaces for real-time applications to negotiate the needed QoS with the network using reservation protocols [16]. However, it should be noted that these QoS negotiations are mainly aimed at in-home entertainment or informational and promotional multimedia services within offices and are, thus, often performed only once, which is prior to the actual transmission. Hence, they do not consider the rapid-link-reliability and time-varying characteristics of in-vehicle wireless networks [5], [6] or the highly diverse and time-varying multimedia content often produced for driver and passenger services [1], [4].

To enable the successful deployment of multimedia services over in-vehicle wireless networks, the QoS negotiation cannot frequently be performed for such multimedia applications due to their delay-sensitive characteristics that require uninterrupted availability of resources. Hence, to ensure the continuity of these multiple in-vehicle video services, multimedia-streaming applications will need to continuously adapt to these time-varying network and source variations while adhering to the negotiated QoS parameters.

We focus on multimedia-streaming systems deploying QoS-enabled in-vehicle wireless networks. These systems need to possess the following basic features to support real-time streaming applications over such networks.

1) Models need to be used to describe multimedia traffic and drive the QoS negotiation for the different in-vehicle services.
2) For a given multimedia bitstream, the packet scheduling needs to be adapted to fulfill the prespecified TSPEC used in the QoS negotiation.
3) QoS renegotiation is allowed and can be initiated by the transmitter, receiver, or proxy.
4) A large number of concurrent sessions should simultaneously be supported, and the system should provide graceful quality degradation by prioritizing different parts of the multimedia bitstreams based on their distortion impact.

In this paper, we propose a streaming solution for in-vehicle wireless multimedia services that require QoS guarantees. We assume that multimedia services are transmitted using a flexible format that can generate on-the-fly prioritized versions of the source data, e.g., using scalable video coding [32], [33], [42]. To enable adaptive transmission of prioritized content, we propose the deployment of multitrack hinting [34] to generate multiple hint tracks, which allow real-time QoS adaptation. An independent TSPEC can be used for the QoS negotiation of each track to allow graceful degradation.

Second, scheduling algorithms are developed for transmitting the packets in the various hint tracks while jointly considering time-varying network conditions (e.g., due to vehicle mobility), receiver-buffer conditions, and content-traffic characteristics to determine viable packet departure times that adhere to the negotiated QoS parameters. Hinting tools can be considered in these scheduling algorithms to generate packet schedules that sustain constant playback without experiencing any buffer under- or overflow events.

Summarizing, we propose an integrated wireless in-vehicle multimedia-streaming solution that combines the following three components: 1) an algorithm to determine the TSPEC parameters to be used for QoS negotiation by the various multimedia services based on the knowledge of traffic and network characteristics, as well as the delay constraints required for the successful deployment of each application (road-surveillance videos, in-vehicle entertainment, etc.); 2) a buffer model, which explicitly considers the resource-constrained receiver prebuffer time of the in-vehicle devices and the maximum delay allowed for buffering prior to decoding; and 3) a flexible and efficient viable scheduling strategy for the various packets that considers instantaneous changes in network conditions and source characteristics and, importantly, the critical importance and delay associated with each service. A viable scheduling strategy is defined as one that can generate packet streams that adhere to the QoS negotiation while fulfilling delay constraints and completely avoiding receiver-buffer under- or overflow events. The major advantage of the proposed scheduler design is its ability to adapt the streaming based on the time-varying end-to-end delay constraints. Instead of assuming a constant delay, it assumes a bounded but time-varying network delay. This is very suitable for in-vehicle wireless networks, where end-to-end network delay may vary rapidly.

The remainder of this paper is organized as follows. Section II reviews related previous research. Section III introduces the multitrack-hinting format and discusses its advantages when applied to adaptive QoS streaming. Section IV develops the basic analysis methodology for packet scheduling. Section V presents a multitrack-hinting algorithm derived from the analysis framework. Section VI evaluates the proposed methods and algorithms through simulations. Our conclusions are presented in Section VII.

II. Related Work

Video bitstreams can be created and stored for transmission using a file format such as the standard MPEG-4 file format [18]. Streaming is facilitated by the hint tracks, which are sets of structured metadata derived based on the compressed bitstreams. A hint track contains information on packet-payload offsets, sizes, protocol-specific settings, and packet departure times and, therefore, can significantly reduce the complexity of packetization and scheduling at transmission time. Hence, using hinting, advanced packetization and scheduling algorithms can be deployed.

However, existing hinting mechanisms are not suitable for scalable-coded bitstreams as they do not allow for flexibility in the creation of substreams from the entire compressed information. This is a required feature for layered and scalable-coding methods, where compressed contents can virtually be

structured into layers with different delivery priorities determined by mutual dependency and relative importance to the final decoded quality. Notice that this is an important feature for in-vehicle wireless video streaming because different streams have different quality and delay requirements. Rate adaptation is achieved in this scalable format by dropping packets from low-priority/low-distortion layers in real-time. Hence, hinting methods need to be developed that efficiently exploit the flexibility associated with the scalable bitstream.

Packet-scheduling algorithms [25]–[28] were developed that are able to optimize the rate-distortion (R-D) performance given time-varying channel and source characteristics. In the context of these studies, packet scheduling is a model-driven optimization process in which packets are selected for (re)transmission in such a way that the distortion is minimized. A comprehensive analysis and formulation of R-D optimization (RaDiO) via packet scheduling is presented in [25]. The apparent complexity of this method, which limited its suitability for real-time streaming, motivated the studies in [26]–[28] to seek low-complexity solutions that may be applied to real-time streaming.

QoS adaptability can also be fulfilled through layered-streaming techniques [29]–[31]. In layered streaming, instead of conducting packet-by-packet optimization as with packet scheduling, video/audio layers generated from scalable-coding methods [17], [20], [21], [32], [33], [42] can be turned on/off in real-time to meet network-rate constraints. The effectiveness of layered streaming depends on the deployed system architectures [29], [34]. Alternative studies address in-network bitstream adaptation, such as transcoding, media-data filtering, intelligent dropping and marking, and QoS mapping [13], [35], [36].

As stated previously, a majority of the previous studies on packet scheduling and layered streaming focused on bandwidth adaptation, effectively dealing with packet losses. However, for in-vehicle video streaming, due to possible limited resource availability at the receiver and the desire to have short-initiation latencies for the streaming application, these algorithms need to be augmented with receiver-buffer control to avoid buffer over- or underflow, which can lead to dramatic quality degradation. The problem of buffer control has been addressed by previous studies [37]–[39]. However, the study in [37] assumes the context of real-time encoding and variable-bit-rate (VBR) channel model to perform buffer control through on-the-fly rate control. The study in [38] describes a generalized reference-decoder model, in which preencoded and stored content can be delivered over time-varying communication channels using multiple leaky-bucket models, each with a different control rate. The selection of the leaky-bucket model depends on the real-time streaming scenario, such as the maximum disk speed when applied for local playback. For each specified leaky-bucket model, there is a requirement on the maximum buffer size and the minimum start latency that the decoder has to follow in order to avoid any over- or underflow event. Since the study in [38] assumes a constant channel delay, its application is limited to networks that are able to enforce roughly constant end-to-end delay, such as ATM networks. Under the assumption of best effort IP networks, the study in [39] presents an Integrated Transport Decoder buffer model that performs priority retransmission for recovery of lost packets to sustain continuous decoding and presentation of scalable-video-coded content. We build on these previous studies and provide a new streaming solution that is able to adhere to prenegotiated QoS parameters for in-vehicle wireless video streaming, depending on the importance of the various video streams.

III. Proposed Multitrack Hinting and In-Vehicle Network QoS Specification

In this paper, our emphasis is on the development of packet-scheduling analysis and algorithms under certain QoS guarantees stemming from each video's importance, as well as from the in-vehicle network infrastructure. To this end, the usage of hint tracks introduced in the MPEG-4 systems part provides a syntactic means for storing scheduling information of media packets that significantly simplifies the operation of an in-vehicle wireless streaming server. In this section, we begin by proposing an extension to the concept of MPEG-4 hinting tracks termed "multitrack hinting" (Section III-A). This extension enables a more flexible format that is suitable for streaming solutions adaptable to the provided in-vehicle network QoS guarantees, as discussed in Section III-B.

A. MPEG-4 Hint Track and Proposed Multitrack Hinting

The MPEG-4 standardization body has developed a standard media-file format (.mp4) [18] that contains timed media information for multimedia presentation, either locally or remotely (such as streaming). This format is deliberately designed with high flexibility and extensibility in order to facilitate interchange, management, editing, and presentation of the media. The standard file format has an inherent hierarchical structure. The basic building blocks used in the construction of mp4 files are called boxes. A box is a specially designed data structure that contains a certain type of media data. Each box has a type name, reflecting the type of data it contains. In addition, a box can contain other boxes to recursively form a hierarchical structure. The general structure of mp4 file format for streaming is shown in Fig. 1(a). Normally, an mp4 file starts with a root box called moov. The moov box further contains other boxes such as boxes for storing elementary bitstreams, boxes for storing synchronization information (or called movie tracks), and boxes for storing hints used by the streaming server to generate packets out of the elementary bitstreams (these boxes are called hint tracks). On the highest level of abstraction, an mp4 file can be viewed as a structure containing elementary bitstreams generated by encoders, movie tracks to guide the video player for local playback, and hint tracks for streaming the media over packet-based networks. The arrows in Fig. 1 indicate that the movie tracks are related to elementary streams and the hint tracks to the movie tracks. The movie tracks contain information (timing and data pointers) that a player will use to extract the corresponding media data for presentation at the designated time. Hint tracks contain information (such as timing and data for packet headers).

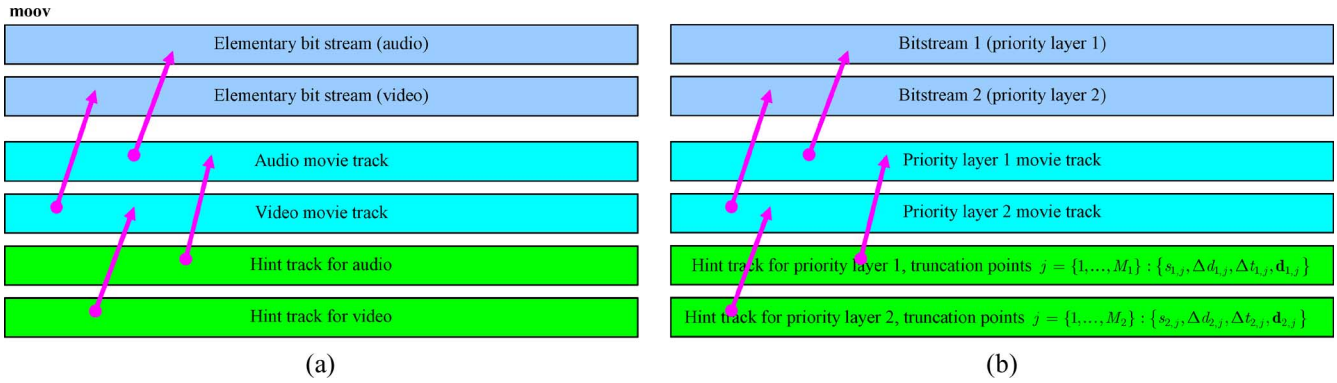


Fig. 1. (a) MPEG-4 file format for an example audio/video-media stream. (b) Proposed multitrack hinting for a video stream, where two layers are used (for illustration purposes). Each layer $l = \{1, 2\}$ consists of a number of truncation points M_l, with each point containing hints for the stream size $(s_{l,j})$, distortion reduction $(\Delta d_{l,j})$, relative playback time $(\Delta t_{l,j})$, and vector of dependencies $(\mathbf{d}_{l,j})$.

We extend the standard media-file format, as shown in Fig. 1(b), to create the multitrack-hinting concept, which was first introduced in our previous study in [34]. Two later studies [40], [41] have also proposed to make use of MPEG-4 hint tracks for adaptive QoS streaming. In [40], the use of R-D hint track is recommended to store precomputed characteristics of compressed media such that the complexity of RaDiO at runtime can significantly be reduced. The study in [41] proposed an R-D-complexity model to characterize compressed bitstreams. Under such a model, streaming adaptation can also consider the decoding complexity, besides R-D tradeoffs.

For each elementary stream (e.g., video), we partition the stream in priority layers $l = \{1, \ldots, L\}$ [with $L = 2$ in the example of Fig. 1(b)] based on the expected distortion reduction at the decoder and the spatio-temporal compression structure [34]. This is following conventional layered-coding principles [17], [21] and can be applied to any scalable or layered video coder or to simulcast transmission of multiple spatio-temporal versions of the same content [21]. Each priority layer l is further partitioned into a number of truncation points $j = \{1, \ldots, M_l\}$, with the independent components forming application-layer (video) packets. The maximum number of truncation points per layer M_l depends on both the coding dependencies and the maximum permissible data payload for application-layer packets. Each individual truncated part (video packet) j of a priority layer l is hinted by its size $s_{l,j}$, the expected distortion reduction $\Delta d_{l,j}$ incurred by using this packet at the decoder, the relative playback time $\Delta t_{l,j}$ (in reference to the previous packet's playback deadline), and the vector of dependencies $\mathbf{d}_{l,j}$, which indicates on which other packets (if any) this bitstream part depends. Dependencies may be imposed in application-layer packets when the maximum permissible packet size is not large enough to accommodate a truncated part of a priority layer.

The use of multitrack hinting provides the possibility for different coding methods and diverse elementary bitstream-syntax structures to be supported by the same server in a common fashion, independent of server design and implementation. This provides the possibility of using the same server infrastructure to offer streaming service in different environments (i.e., in-vehicle wired networks or wireless networks). In order to decide how to schedule video packets under network-provided guarantees of service, one needs to assume a QoS mechanism. This is elaborated in the following section.

B. QoS Adaptation—Transport-Specification (TSPEC) Model

A certain TSPEC with a set of predetermined parameters can be passed by applications to the network layer in order to make a QoS request. In this way, each in-vehicle network node may perform resource allocation for a particular video-streaming session based on the submitted TSPEC model. Typically, relevant TSPEC parameters include the following: Peak data rate $R_{\max}$, mean data rate R_a, maximum burst size σ, worst case delay $D_{\max}$, average packet size K_a, maximum packet size $K_{\max}$, and maximum packet-error rate $E_{\max}$. We denote TSPEC as $\Gamma(R_{\max}, R_a, \sigma, K_a, K_{\max}, D_{\max}, E_{\max})$.

The TSPEC metrics can be grouped into two subsets: the traffic characteristics $(R_{\max}, R_a, \sigma, K_a, K_{\max})$ and the required network guarantees $(D_{\max}, E_{\max})$. Traffic characteristics relate to the particular in-vehicle wireless video stream; high-priority streams such as views from in-vehicle cameras relating to vehicle guidance or driver assistance are typically given higher bandwidth and larger burst size bound to ensure high quality. Lower priority streams such as entertainment or views received from roadside cameras concerning traffic information typically reserve smaller bandwidth [1]. Network guarantees are also tuned to the video-stream priority and real-time requirements. In-vehicle streaming of entertainment videos typically tolerates large latency (quantified by the worst case delay $D_{\max}$) as compared to real-time surveillance streams.

A simple way to view the interaction of the multitrack-hinting specification and a certain TSPEC for an in-vehicle streaming session is as follows. Since all hint-track layers predetermine packet-payload sizes and their relative playback time, assuming a certain scheduling mechanism for a number of layers l, $1 \leq l \leq L$, the traffic characteristics can be predefined, and they can easily be expressed by a TSPEC request to the network. Conversely, given a certain negotiated TSPEC, one may determine a scheduling mechanism for a number of hint layers l, $1 \leq l \leq L$ in order to simultaneously comply with the specification and maximize the received video quality. In the remainder of this paper, we are mostly concerned with the latter aspect. In particular, our focus is on analytically expressing the

TABLE I
SYMBOLS AND THEIR DEFINITIONS

Symbol	Section	Definition
l and L	III.A	priority layer of video stream, and total number of priority layers
j and M_l	III.A	jth truncation point of priority layer l, and total number of truncation points (packets) of layer l
$s_{l,j}$, $\Delta d_{l,j}$, and $\Delta t_{l,j}$	III.A	size, distortion reduction, and relative playback deadline of packet j of priority layer l
$\mathbf{d}_{l,j}$	III.A	vector of dependencies of packet j of layer l
$R_{\max}$, R_a, σ, $K_{\max}$, K_a	III.B	peak data rate, mean data rate, max. burst size, max. packet size, and average packet size
$D_{\max}$, and $E_{\max}$	III.B	worst-case delay, and max. packet error rate
$\Gamma(\ldots)$	III.B	Transport Specification (TSPEC)
$\mathcal{P}_{l,j}$	IV.A	tuple characterizing each video block (packet) j of priority layer l
m and s_m, M_{playback}	IV.A	index and size of stream segment with the same playback deadline, total playback deadlines
$\Delta t_m^{\text{playback}}$	IV.A	relative playback time of stream segment m
$\mathcal{S}_m$	IV.A	tuple of stream segment m scheduled to be transmitted
$\mathcal{R}_m$	IV.A	tuple of stream segment m including layer prioritization, packet ordering and protection
N_m	IV.A	distortion layers within the packets of a stream segment m with the same playback deadline
$\Delta t_m^{\text{transmit}}$, $\Delta t_m^{\text{receive}}$	IV.B	relative transmission time, relative receival time of stream segment m
$\mathcal{R}_m^{\text{schedule}}$, and $\mathcal{R}_m^{\text{receive}}$	IV.B	tuple characterizing segment m : scheduled stream for transmission, and received stream
$W_{M_{\text{playback}}}$	IV.B	time window for averaging the incoming traffic
T_B, and T_D	IV.B	pre-buffer time, and max. buffer time
$\mathcal{B}^{\text{receive}}$	IV.B	receiver buffer tuple
$\Lambda_{\mathcal{B}^{\text{receive}},\Gamma}(\mathcal{R}_m)$	IV.B	scheduler under TSPEC Γ and receiver buffer $\mathcal{B}^{\text{receive}}$
$\{m_{\text{start}}^{\text{viable}},\ldots,m_{\text{end}}^{\text{viable}}\}$	IV.C	viable range of packets for transmission of stream segment (scheduled packet) m
m_D	IV.C	last packet (upper bound) before decoding stream segment m ($m \le m_{\text{start}}^{\text{viable}} \le m_{\text{end}}^{\text{viable}} \le m_D$)

conditions for the existence of viable schedules complying with a given TSPEC and deriving the best possible schedule out of the viable set. Before we proceed in the analysis of TSPEC parameters and their interaction with the specified bitstream, for clarity, we summarize the key notations used in this paper in Table I. The first column denotes the symbol. The second column contains the section where the symbol is first defined. The third column holds the definition for the symbol within the context of the derivation.

IV. PACKET SCHEDULING—CONCEPTS AND VIABILITY

Given a certain multitrack-hint specification, packet scheduling is concerned with the following: 1) the establishment of which packets out of which layers should be transmitted and the protection mechanism corresponding to the expected error rate and 2) the establishment of each packet's departure time. Concerning the first point, the size of each individual packet is bounded by the maximum transport unit of an end-to-end path and the semantics of the elementary bitstream. Protection typically consists of error-correction mechanisms [19], [22], [23], [27], [28] involving channel coding or simple retransmissions.[1] In both cases, the result consists of additional video packets linked to the layer-truncation points of Fig. 1(b) and having similar hint descriptions in terms of size, expected distortion reduction, playback deadline, and dependencies. Concerning the second point, each packet's departure time is set such that, apart from complying with the overall traffic characteristics of the TSPEC, under- or overflow of the in-vehicle receiver buffer is avoided. This requires the knowledge of receiver-side buffer conditions including prebuffer time and maximum buffer size.

In Section IV-A, we discuss the layer prioritization and ordering mechanisms, combined with layer protection. Section IV-B presents the various aspects of the utilized streaming model, while Sections IV-C and D present the proposed viability constraints.

A. Layer Prioritization, Packet Ordering, and Protection Mechanisms

Based on the multitrack-hint specification of a layered or scalable bitstream, each video block (packet) j, $1 \le j \le M_l$, of layer l $(1 \le l \le L)$ is characterized by the tuple $\mathcal{P}_{l,j} = \{s_{l,j}, \Delta d_{l,j}, \Delta t_{l,j}, \mathbf{d}_{l,j}\}$. For streaming of offline compressed video content (e.g., a movie in an in-vehicle entertainment system), all $\mathcal{P}_{l,j}$ can be generated *a priori* in order to assist the scheduling and packet transmission. In particular, during the streaming session, layer-prioritization and packet-ordering mechanisms sort the tuples $\mathcal{P}_{l,j}$ corresponding to each stream segment hierarchically in three classes based on the following: 1) playback deadlines $\Delta t_{l,j}$; 2) block dependencies within all the blocks having the same playback-deadline index[2] m, with $1 \le m \le M_{\text{playback}}$ and M_{playback}, which are the total playback deadlines of the stream segment of interest; and 3) distortion reduction within the tuples of each m from step 2). The final result contains a series of packets characterized by the two-tuples $\mathcal{S}_m = \{s_m, \Delta t_m^{\text{playback}}\}$, where s_m is the size of packet $m(\forall m : s_m \le K_{\max}$ for TSPEC compliance), and $\Delta t_m^{\text{playback}}$ is its relative playback deadline (in relation to the previous packet), $m = \{1,\ldots,M_{\text{playback}}\}$. Notice that

[1]More advanced concepts combining such approaches with multipath transmission can be envisaged; however, they tie the scheduling to a particular application framework, e.g., multihop wireless networks or the Internet, and as such, they deviate from the scope of this paper.

[2]Even though each packet is assigned each own playback deadline $\Delta t_{l,j}$, in reality, packets within a certain time interval are grouped into M_{playback} classes, where all packets within a class have the same playback deadline.

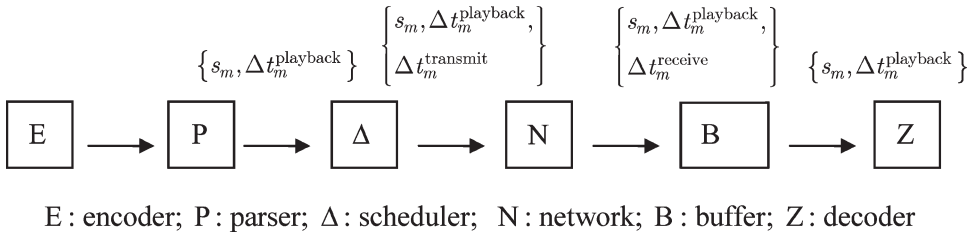


Fig. 2. Transferring of application-layer packets via the in-vehicle streaming system. The encoder E can be any in-vehicle surveillance camera or simply prestored entertainment video or streamed video from an outside network (e.g., from a roadside camera). The decoder Z is the in-vehicle receiver, e.g., a monitor for driver assistance or a rear-seat entertainment system.

each playback deadline may contain more than one video packet, which are sorted for transmission based on their dependencies and their relative distortion reduction. In particular, steps 2) and 3) from the above discussion could be interchanged depending on whether the final schedule should be distortion- or dependence-prioritized. We remark here that several scalable coders tend to minimize or completely alleviate dependencies across different video packets [42], thereby making distortion prioritization the dominant criterion. In the remainder of this paper, we shall jointly indicate the video packets within one delay deadline index as "video packet" or simply "packet," since once they are scheduled for transmission they follow a fixed transmission order.

Protection mechanisms can be applied in the deadline-dependency-distortion scheduling in a variety of ways. For example, for each deadline class (video packet) m, a layered forward-error-correction (FEC) scheme may be applied [24], where additional FEC packets are incorporated such that errors can be corrected given the TSPEC parameter $E_{\max}$. For example, if we assume (without loss of generality) that $E_{\max}$ expresses the maximum packet-error probability and that each video packet m contains N_m distortion-reduction layers, a set of N_m FEC codes could be used where each code n, $1 \leq n \leq N_m$ adds redundancy proportional to the relative importance of each distortion-reduction layer. Similarly, if packets can be retransmitted based on automatic-repeat-request schemes, N_m different retransmission limits could be set for each distortion-reduction class. In total, if each packet is additionally protected by appropriate FEC codes or retransmissions based on $E_{\max}$, the protection mechanism will result in a series of video packets $\mathcal{R}_m$, $m = \{1, \ldots, M_{\text{playback}}\}$.

Overall, the process of layer prioritization, packet ordering, and protection can be described as $\mathcal{P}_{l.j} \rightarrow \mathcal{S}_m \rightarrow \mathcal{R}_m$, with the final result $\mathcal{R}_m = \{s_m, \Delta t_m^{\text{playback}}\}$ consisting of an ordered set of video packets m, $m = \{1, \ldots, M_{\text{playback}}\}$, and each video packet m consisting of multiple-quality/protection layers (N_m).

B. Data-Flow Model for Streaming

The data flow in a streaming process is shown using Fig. 2. In the figure, the encoder (E) produces a bitstream, and the syntax-aware parser (P) performs the layer prioritization, packet ordering, and protection described in the previous section. The scheduler will then augment the prioritized video packets with the scheduled relative transmission time $\Delta t_m^{\text{transmit}}$ (in relation to the previous video packet's transmission time), thereby forming the scheduling tuples $\mathcal{R}_m^{\text{schedule}} = \{s_m, \Delta t_m^{\text{playback}}, \Delta t_m^{\text{transmit}}\}$. After going through the network (N), it is once again transformed into another new sequence of $\mathcal{R}_m^{\text{receive}} = \{s_m, \Delta t_m^{\text{playback}}, \Delta t_m^{\text{receive}}\}$, with $\Delta t_m^{\text{receive}}$, which is the relative arrival time at the decoder buffer (B). Finally, the received sequence is consumed by the decoder (Z), and the decoder extracts the data out of the buffer following exactly the same trace defined by the sequence $\mathcal{S}_m$. A viable scheduler should ensure that each video packet arrives at the buffer in time so that a complete sequence $\mathcal{S}_m$ can be recovered by the decoder. This means that the arrival process will not cause buffer over- or underflow when data departure from the buffer follows the schedule $\mathcal{R}_m^{\text{schedule}}$.

Given a certain TSPEC, we assume that the network uses a token-bucket model to enforce and serve each data stream arriving at a QoS-capable in-vehicle network node [12]. In this model, two buckets are used in concatenation, with the first one for policing peak rate $R_{\max}$ and the second for policing mean rate R_a and maximum burst size σ. Let $R_{W_{M_{\text{playback}}}}$ denote the incoming data rate averaged in a time window $W_{M_{\text{playback}}}$ that corresponds to the interval of transmission of M_{playback} packets, i.e., the average rate corresponding to tuples $\mathcal{R}_m^{\text{schedule}}$ $m = \{1, \ldots, M_{\text{playback}}\}$. Then

$$R_{W_{M_{\text{playback}}}} = \frac{1}{W_{M_{\text{playback}}}} \sum_{m=1}^{M_{\text{playback}}} s_m. \tag{1}$$

If the transmission time of the initial packet is t_1^{transmit}, then all the packets to be transmitted will be sent within $[t_1^{\text{transmit}}, t_1^{\text{transmit}} + W_{M_{\text{playback}}}]$. An ideally acceptable data stream to the token-bucket model would have to be conditioned by

$$\forall m, w_m, \text{ with } 0 \leq w_m \leq W_{M_{\text{playback}}} : R_{w_m} \leq R_{\max} \text{ and } 0 \leq (R_{w_m} - R_a) w_m \leq \sigma. \tag{2}$$

i.e., for any transmission interval w_m within $[t_1^{\text{transmit}}, t_1^{\text{transmit}} + W_{M_{\text{playback}}}]$, the corresponding traffic rate R_{w_m} [defined equivalently to (1)] should be smaller than the peak rate, and the token bucket should never experience over- or underflow.[3]

[3]In (2), as well as in subsequent derivations relating to the leaky bucket, we always assume that the state of the leaky bucket at the beginning of the transmission interval of interest is taken into account, i.e., the mean token rate is R_a, adjusted accordingly.

The receiver-side buffer is modeled by the tuple $\mathcal{B}_{\text{receive}} = \{T_B, T_D\}$, where T_B is the prebuffer time before starting decoding each packet (stream segment), and T_D is the maximum permissible buffer time (set based on the buffer size or other receiver constraints). In our analysis, the buffer sizes are measured in terms of playback time, and T_D is constant based on the physical buffer size, but T_B is varying based on the received packets. Notice that if the buffer constraints are given in bits, for any incoming traffic averaged in a time window $W_{M_{\text{playback}}}$ and based on the calculation of $R_{W_{M_{\text{playback}}}}$ from (1), the buffer sizes can be converted to bits by $R_{W_{M_{\text{playback}}}} \cdot T_B$ and $R_{W_{M_{\text{playback}}}} \cdot T_D$.

To deliver a given bitstream represented by $\mathcal{R}_m$, we would have to perform the following steps.

1) Assume the settings of the receiver-side buffer model $\mathcal{B}_{\text{receive}}$, according to the buffer-delay requirements of this particular application and the receiver-side resource availability.
2) Construct an appropriate TSPEC model $\Gamma(R_{\max}, R_a, \sigma, K_a, K_{\max}, D_{\max}, E_{\max})$ by considering the traffic characteristics of the given bitstream, as well as the network-latency and error-handling capability of the codec/streaming system (quantified by FEC capabilities or retransmission possibilities).
3) Determine the departure time of each packet using a scheduling function (referred to as a scheduler) $\forall m : \Lambda_{\mathcal{B}_{\text{receive}},\Gamma}(\mathcal{R}_m) \rightarrow \mathcal{R}_m^{\text{schedule}}$.

We refer to such a design process as finding a streaming solution to a given bitstream. It is now obvious that a complete streaming solution for a given bitstream consists of three components: $\{\mathcal{B}_{\text{receive}}, \Gamma, \Lambda_{\mathcal{B}_{\text{receive}},\Gamma}\}$, namely, the assumed buffer model, the TSPEC model, and the designed scheduler, respectively.

C. Viability of Streaming Solution

Apparently, we can construct many different streaming solutions (i.e., different $\{\mathcal{B}_{\text{receive}}, \Gamma, \Lambda_{\mathcal{B}_{\text{receive}},\Gamma}\}$) for a given bitstream expressed by $\mathcal{R}_m$. The three components of any streaming solution can be specified empirically and independently without following any general principles. However, empirical solutions tend to provide inefficient network-resource utilization and poor video quality as perceived by the end users. Hence, in this paper, we are concerned with the conditional optimization of $\Lambda_{\mathcal{B}_{\text{receive}},\Gamma}$, given $\mathcal{B}_{\text{receive}}$ and Γ. In particular, among many possible solutions for a given bitstream, some can make full use of the network resources reserved from QoS negotiation Γ (i.e., avoiding overprovisioning of network resources but also avoiding TSPEC violation) and satisfy receiver-side buffering conditions $\mathcal{B}_{\text{receive}}$ (i.e., prevent buffer over- and underflow events). We refer to such solutions as viable solutions and their corresponding schedulers as being viable. In the following, we formulate these constraints mathematically and present a method to check the viability of a solution $\Lambda_{\mathcal{B}_{\text{receive}},\Gamma}$.

Assume that the time window w_m used by the token-bucket model to calculate the arrival rate is sufficiently large such as $w_m \cdot R_{\max} \gg K_{\max}$. Following the previous discussion, we can easily derive two TSPEC constraints that need to be fulfilled for a viable streaming solution.

Constraint 1 (Peak Rate Bound): For any m_1, $m_2 (1 \leq m_1 \leq m_2 \leq M_{\text{playback}})$, where $\sum_{k=m_1}^{m_2} \Delta t_k^{\text{transmit}} = w_m$, the arrival process generated by a scheduler for a given bitstream should satisfy

$$\sum_{k=m_1}^{m_2} s_k \leq w_m \cdot R_{\max} \tag{3}$$

which also implies that $\sum_{k=m_1}^{m_2} s_k \leq (\sum_{k=m_1}^{m_2} \Delta t_k^{\text{transmit}}) R_{\max}$. □

Constraint 2 (Average Rate Bound): For any m_1, $m_2 (1 \leq m_1 \leq m_2 \leq M_{\text{playback}})$, the arrival process generated by a scheduler for a given bitstream should meet the condition that

$$0 \leq \sum_{k=m_1}^{m_2} s_k - R_a \sum_{k=m_1}^{m_2} \Delta t_k^{\text{transmit}} \leq \sigma. \tag{4}$$

The last constraint prevents any buffer over- or underflow at the receiver side. □

Lemma 1 (Delay Bound—Based on the Definition of Worst Case Network Delay): If the network delay is bounded by $D_{\max}$, then, for any $m (1 \leq m \leq M_{\text{playback}})$, we have the following delay bound:

$$\sum_{k=1}^{m} \Delta t_k^{\text{receive}} - \sum_{k=1}^{m} \Delta t_k^{\text{transmit}} | \ldots | \leq D_{\max}. \tag{5}$$

For streaming applications, this is the most important property of QoS-enabled networks. Based on this property, we are able to find deterministic schedulers that can form viable streaming solutions. In addition, the property is still applicable if $D_{\max}$ is viewed as the delay-jitter bound instead of delay bound. □

Lemma 2 (Viable Range of Arrivals): Assume a receiver-side buffer modeled by the tuple $\mathcal{B}_{\text{receive}} = \{T_B, T_D\}$ and the last packet $m_D (m \leq m_D \leq M_{\text{playback}})$ that can arrive before the decoder starts decoding packet m. In this context, m_D serves as an upper bound for the possible packets that can be accommodated by the receiver buffer prior to decoding packet m. In a practical streaming scenario, within the decoding interval of packets $m-1$ and m, i.e., $\Delta t_m^{\text{playback}}$, packets $m_{\text{start}}^{\text{viable}}, m_{\text{start}}^{\text{viable}} + 1, \ldots, m_{\text{end}}^{\text{viable}}$ are received (see Fig. 3) with $m \leq m_{\text{start}}^{\text{viable}} \leq m_{\text{end}}^{\text{viable}} \leq m_D$. Hence, we may assume that $\sum_{k=m_{\text{start}}^{\text{viable}}}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{receive}} \approx \Delta t_m^{\text{playback}}$. We can associate $m_{\text{start}}^{\text{viable}}$ with T_B by

$$T_B = \sum_{k=m}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{playback}} \tag{6}$$

i.e., the prebuffer time corresponds to the packets existing in the buffer (packets $\{m, m+1, \ldots, m_{\text{start}}^{\text{viable}} - 1\}$) prior to the

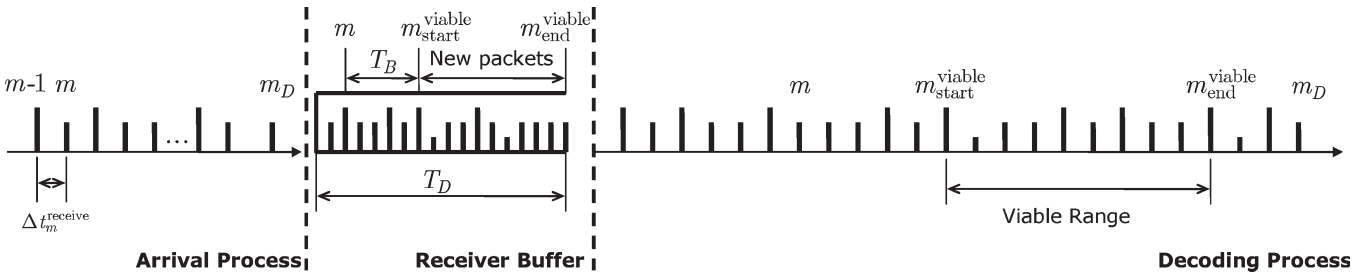


Fig. 3. Arrival process, receiver buffer, and decoding process (with detailed illustration indicating the viable range) at the time instant when packet m is about to be decoded.

playback interval for packet $m(\Delta t_m^{\text{playback}})$. In the worst-case scenario where $m_{\text{start}}^{\text{viable}} = m$, we have $T_B \equiv 0$.

In order to avoid buffer under- and overflow, we have the following viable range of arrivals:

$$0 < \sum_{k=m_{\text{start}}^{\text{viable}}}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{playback}} + T_B \le T_D \tag{7}$$

i.e., the cumulative playback time of the packets arriving in the receiver buffer during the decoding of packet m plus the current prebuffer time is bounded by the worst-case receiver-buffer delay. This is shown pictorially in the middle part ("Receiver Buffer") of Fig. 3. □

According to the practical QoS conditions for an in-vehicle wireless network, (7) can be satisfied by transmitting all or some of the packets within the group of $\{m, \ldots, m_D\}$ packets, i.e., the packets $\{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$. We refer to such a packet range as the viable range of packet m.

The viable range simply specifies the range of packets that can safely arrive at the buffer without causing any over- or underflow events when packet m is to be decoded. Consequently, a viable scheduler should ensure that when packet m is to be decoded, some or all of the packets from its viable range should have arrived at the buffer. In general, the viable schedule can be expressed by packets $\{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$ and their scheduled (relative) transmission times. In addition, the viable range of m can be adjusted by changing T_D and T_B, as seen from (6) and (7). Finally, the departure time $\Delta t_k^{\text{transmit}}$ of each packet k in the viable range $\{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$ from the server application to the network interface timed after the prebuffering period is to be in the time range $[\sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{transmit}}, \sum_{k=1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}}]$, which we refer to as the departure-time span of the viable range.

Lemma 1 shows that, under the worst-case situation, when the series of packets arrives at the receiver buffer, the time period of $\sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{transmit}}$ (the departure time period of packets just before the viable range of packets) may be stretched by network jitter to a maximum of $\sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{transmit}} + D_{\max}$, while $\sum_{k=m_{\text{start}}^{\text{viable}}}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}}$ (the departure time period of packets in the viable range—which is also determined by the scheduler) may maximally be decreased to $\sum_{k=m_{\text{start}}^{\text{viable}}}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} - D_{\max}$. Therefore, the arrival time-span of the viable range, which is defined as

$$\left[\sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{receive}}, \quad \sum_{k=1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{receive}}\right]$$

is worse-case bounded by $[\sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{transmit}} + D_{\max}, \sum_{k=1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} - D_{\max}]$.

This property implies that, after time $\sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{transmit}} + D_{\max}$, it is ensured that at least $m-1$ packets have passed through the receiver buffer because they have already been decoded. Similarly, before time instant $\sum_{k=1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} - D_{\max}$, there are at most $m_{\text{end}}^{\text{viable}} - m$ packets at the receiver buffer. In other words, if packet m is to be decoded (or consumed) in the time period of

$$\left[\sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{transmit}} + D_{\max}, \quad \sum_{k=1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} - D_{\max}\right]$$

there would be no buffer over- or underflow event. These derivations are grouped together in the third constraint for a viable solution given as follows.

Constraint 3 (Receiver Buffer): Assume $\mathcal{R}_m^{\text{schedule}}$ is generated by the scheduler of a streaming solution for a given packet m, there exists a corresponding viable range of packets $\{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$, and packet m is associated with its viable range of packets by

$$\begin{aligned} D_{\max} &\le \sum_{k=1}^{m} \Delta t_k^{\text{playback}} - \sum_{k=1}^{m_{\text{start}}^{\text{viable}}-1} \Delta t_k^{\text{transmit}} \\ &\le \sum_{k=m_{\text{start}}^{\text{viable}}}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} - D_{\max} \end{aligned} \tag{8}$$

in order to guarantee that there will not be a buffer over- or underflow event during the streaming session. □

As a special case, if $m_{\text{end}}^{\text{viable}} = M_{\text{playback}}$, this means that when packet m is decoded, even if all M_{playback} packets arrive, this does not cause buffer overflow, and therefore, the right bound of (8) is no more needed. In addition, this constraint reveals how $\Delta t_k^{\text{transmit}}$, $k = \{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$, generated by a viable scheduler, should be related to $\Delta t_m^{\text{playback}}$ in order

to avoid buffer over- and underflow when under the assumption of a network-delay boundary.

D. Method for Viability Checking

It would be useful to have a method that can check the existence of viable scheduler(s) $(\Lambda_{\mathcal{B},\Gamma})$ for a given bitstream $\mathcal{R}_m$ consisting of packets $k = \{1, \ldots, m\}$ for any m, $m = \{1, \ldots, M_{\text{playback}}\}$, under the assumed TSPEC model (Γ) and receiver-buffer model $(\mathcal{B})$. By combining the three previous constraints, we develop such a method as follows.

Lemma 2 shows that, for each packet m, a corresponding viable range $\{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$ exists under certain conditions. Based on constraint 2 (Average Rate), we define the remaining burst size σ' for the transmission time between packet one and packet $m_{\text{start}}^{\text{viable}}$ as

$$\sigma' = \sigma - \sum_{k=1}^{m_{\text{start}}^{\text{viable}}} s_k - \sum_{k=1}^{m_{\text{start}}^{\text{viable}}} \Delta t_k^{\text{transmit}} \cdot R_a. \tag{9}$$

Considering constraint 1 (Peak Rate), we have

$$\sum_{k=1}^{m_{\text{start}}^{\text{viable}}} \Delta t_k^{\text{transmit}} \geq \max\left\{ \frac{1}{R_a}\left(\sum_{k=1}^{m_{\text{start}}^{\text{viable}}} s_k - \sigma' \right), \frac{1}{R_{\max}} \sum_{k=1}^{m_{\text{start}}^{\text{viable}}} s_k \right\}. \tag{10}$$

Meanwhile, to avoid the underflow of the token bucket, we have

$$\sum_{k=1}^{m_{\text{end}}^{\text{viable}}} s_k \geq \sum_{k=1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} \cdot R_a \tag{11}$$

or

$$\sum_{k=1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} \leq \frac{1}{R_a} \sum_{k=1}^{m_{\text{end}}^{\text{viable}}} s_k. \tag{12}$$

By combining (8), (10), and (12) together, we reach the final viability constraint

$$\max\left\{ \frac{1}{R_a}\left(\sum_{k=1}^{m_{\text{start}}^{\text{viable}}} s_k - \sigma' \right), \frac{1}{R_{\max}} \sum_{k=1}^{m_{\text{start}}^{\text{viable}}} s_k \right\} + D_{\max} \leq \sum_{k=1}^{m} \Delta t_k^{\text{playback}} \leq \frac{1}{R_a} \sum_{k=1}^{m_{\text{end}}^{\text{viable}}} s_k - D_{\max}. \tag{13}$$

If there exists a group of packets $\{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$ with $m \leq m_{\text{start}}^{\text{viable}} \leq m_{\text{end}}^{\text{viable}} \leq m_D$ such that (13) holds for all the packets in the group, then there exists a viable scheduler for the given bitstream under the assumed TSPEC model and receiver-buffer model. Notice that since (13) is derived from combining the three constraints of a viable solution together, it contains all the TSPEC characteristics $R_{\max}$, R_a, σ (except for the packet sizes K_a, $K_{\max}$ that are taken into account during the video-packet formation, as explained in Section IV-A). As a result, we have demonstrated by construction that (13) is a necessary and sufficient condition for a packet m to have a viable solution. In addition, as it will be shown in the following section, based on the proposed multitrack-hinting concept, it is always possible to modify the assumed TSPEC model and buffer conditions to make a bitstream viably schedulable according to (13).

V. Algorithms

In the previous section, we discussed the constraints of a viable steaming solution and developed a method for checking the availability of viable schedulers for a given bitstream when under an assumed TSPEC model and buffer model. However, the remaining question would be how to obtain such a scheduler once the given bitstream passes the viability checking.

In this section, we first present an iterative algorithm (Sections V-A and B) that can generally be applied to bitstreams that pass the viability checking to obtain viable schedulers. We then extend this method to the case of multitrack hinting (Section V-C).

A. Scheduling Algorithm

There could exist many viable schedulers for a given bitstream under given TSPEC and buffer constraints. We present in the following an algorithm that can find such a scheduler.

Assume that $\Delta t_1^{\text{transmit}}, \ldots, \Delta t_{m-1}^{\text{transmit}}$ are already determined and the proper value for $\Delta t_m^{\text{transmit}}$ needs to be found. The algorithm can be summarized as follows.

1) Calculate $\Delta t_m^{\min\_\text{transmit}}$ and $m_{\text{end}}^{\text{viable}}$ (with $m \leq m_{\text{end}}^{\text{viable}} \leq m_D$) such that the time range $t_{\text{viable}} = [\sum_{k=1}^{m-1} \Delta t_k^{\text{transmit}} + \Delta t_m^{\min\_\text{transmit}}, \sum_{k=1}^{m-1} \Delta t_k^{\text{transmit}} + \Delta t_m^{\min\_\text{transmit}} + \sum_{k=m+1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}}]$ satisfies constraints 1, 2, and 3.
2) Select a proper value for $\Delta t_m^{\text{transmit}}$ such that $\sum_{k=1}^{m-1} \Delta t_k^{\text{transmit}} + \Delta t_m^{\text{transmit}} \in t_{\text{viable}}$.

Notice that, in this case, we are not concerned with the particular starting packet of the viable range $(m_{\text{start}}^{\text{viable}})$ as we are dealing with the scheduling of packet m, which is always the lower bound for $m_{\text{start}}^{\text{viable}}$. The specific steps to derive $\Delta t_m^{\text{transmit}}$ are given in the following.

Solution Step 1: From inequalities (10) and (12) and considering that the viable range $\{m_{\text{start}}^{\text{viable}}, \ldots, m_{\text{end}}^{\text{viable}}\}$ in the worst case consists of only packet m (i.e., we replace both $m_{\text{start}}^{\text{viable}}$ and $m_{\text{end}}^{\text{viable}}$ by m), we have

$$\max\left\{ \frac{1}{R_a}\left(\sum_{k=1}^{m} s_k - \sigma' \right), \frac{1}{R_{\max}} \sum_{k=1}^{m} s_k \right\} - \sum_{k=1}^{m-1} \Delta t_k^{\text{transmit}} \leq \Delta t_m^{\text{transmit}} \leq \frac{1}{R_a} \sum_{k=1}^{m} s_k - \sum_{k=1}^{m-1} \Delta t_k^{\text{transmit}}. \tag{14}$$

The last equation represents the first range of $\Delta t_m^{\text{transmit}}$. The first step of the solution assures that constraints 1 and 2 are met.

Solution Step 2: For the mth packet, we find the minimum transmission time $\Delta t_m^{\min\_\text{transmit}}$ as a function of the (maximum possible) viable range of packets $\{m, \ldots, m_{\text{end}}^{\text{viable}}\}$ based on the right inequality of (8). This is performed by the replacement of $m_{\text{start}}^{\text{viable}}$ by m, which is the lower bound for $m_{\text{start}}^{\text{viable}}$, and by setting equality instead of inequality on the right side of (8) in order to get the minimum transmission time, as required by the definition of $\Delta t_m^{\min\_\text{transmit}}$. This gives

$$\Delta t_m^{\min\_\text{transmit}} = \sum_{k=1}^{m} \Delta t_k^{\text{playback}} - \sum_{k=1}^{m-1} \Delta t_k^{\text{transmit}} - \sum_{k=m+1}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{transmit}} + D_{\max} \quad (15)$$

and we can also calculate the maximum viable range of packets $m_{\text{end}}^{\text{viable}}$ based on (7) and under the assumption of $m_{\text{start}}^{\text{viable}} = m$ (which means that $T_B \equiv 0$) by imposing the equality

$$\sum_{k=m}^{m_{\text{end}}^{\text{viable}}} \Delta t_k^{\text{playback}} = T_D. \quad (16)$$

Now, by applying the viability constraint of (14) for $m_{\text{end}}^{\text{viable}} - m - 1$ times, i.e., replacing m by $m+1, \ldots, m_{\text{end}}^{\text{viable}}$, and by selecting the minimum of the two bounds[4] given by (14), we finally obtain the value of $\Delta t_{m_{\text{end}}^{\text{viable}}}^{\text{transmit}}$, which leads to a second range for $\Delta t_m^{\text{transmit}}$

$$\Delta t_m^{\min\_\text{transmit}} \le \Delta t_m^{\text{transmit}} \le \Delta t_{m_{\text{end}}^{\text{viable}}}^{\text{transmit}}. \quad (17)$$

Solution Step 3: Assuming that the ranges of the two previous steps overlap with each other, i.e., that a viable schedule is possible, the final range for $\Delta t_m^{\text{transmit}}$ would be within the common range of the two intervals of (14) and (17).

In the final stage, we need to decide which value for $\Delta t_m^{\text{transmit}}$ should be chosen from the final range. Similar to what is performed when deriving the upper bound for the range of (17), the strategy would be to pick $\Delta t_m^{\text{transmit}}$ that will allow for the maximum of the final viable range for the next packet, i.e., the maximum interval for $\Delta t_{m+1}^{\text{transmit}}$. Notice that if the bitstream can pass the viability checking, then the final viable range will exist for any m, which means that there will exist a common range of the two intervals of (14) and (17) for any m. We conclude this section by examining some properties of a viable scheduler.

B. Scheduler Properties

Property 1 (Overprovision Friendly): Assume that Δ is a viable scheduler constructed under assumptions of TSPEC model Γ' and buffer condition B', and Γ'' and B'' are overprovisioned TSPEC model and buffer condition such that $R'_{\max} \le R''_{\max}$, $R'_a \le R''_a$, $\sigma' \le \sigma''$, $D'_{\max} > D''_{\max}$, $T'_B \le T''_B$, and $T'_D \le T''_D$, then (B'', Γ'', Δ) can also form a streaming solution that guarantees the prevention of buffer over- and underflow events.

Property 2 (Pause Friendly): When a pause event happens in a streaming process, the scheduler can be resumed at the pause point, as long as proper receiver-side buffering is performed before playback is restarted. More precisely, the buffering amount can particularly be calculated by assuming that the network delays experienced by all data packets up to the pause point are constant, i.e., zero.

Assuming that the pause point is at packet $m_{\text{pause}} > m_D$, then the maximum prebuffer amount at resume can easily be calculated as

$$T_B^{\max} = \sum_{k=m}^{m_{\text{pause}}-1} \Delta t_k^{\text{playback}} - \sum_{k=m_D}^{m_{\text{pause}}-1} \Delta t_k^{\text{transmit}}. \quad (18)$$

Proof: When the streaming is resumed after a pause, if a valid (i.e., viable) buffering state can be recovered at the receiver side, then the original scheduler can be reused.

We need to find a valid buffering state that the mth packet may observe. The viable scheduler is constructed to work with any delays that fall into the range of $[0, D_{\max}]$. A constant delay equal to zero is also in this range, and it is, in fact, the worse case with respect to buffer fullness, since it guarantees uninterrupted instantaneous delivery of all packets. Therefore, assuming that the scenario where all packets from m_D up to the pause point experience constant delay (0) is possible, the scheduler should be designed to be able to be accommodating. We can calculate T_B from (18) for this worst-case buffering state.

C. Multitrack Hinting

In this section, we discuss three possible multitrack-hinting methods that may be derived from the previous scheduling algorithm.

Method 1 (Scheduling for Multicasting via Independent QoS-Layer Negotiations): Assume that a scalable coded video consists of dividable coding blocks and that each is identified by a triplet $(l, \Delta t_{l,j}^{\text{playback}}, s_{l,j})$, where $l \in [1, L]$ is the layer index. The following steps lead to a multitrack-hinting solution.

1) Set $l = 1$.
2) For the subbitstream $(l, \Delta t_{l,j}^{\text{playback}}, s_{l,j})$, where $j \in [1, M_B]$, which corresponds to layer l, construct the appropriate TSPEC model Γ_l, and specify a receiver-side buffer condition B_l.
3) Apply the viable scheduler design presented in the previous section to subbitstream $(l, \Delta t_{l,j}^{\text{playback}}, s_{l,j})$ independently and obtain a scheduler Δ_l.
4) Construct the hint track H_l for layer l using the time information derived from scheduler Δ_l, increase $l = l + 1$.
5) If $l \le L$, go to step 2) or else terminate.

The set $\{\mathrm{H}_l : l \le L\}$ forms a multitrack-hinting solution for this particular video.

[4]Selecting the minimum of the two bounds for each of the $m+1, \ldots, m_{\text{viable}}$ ensures there will be sufficient range for selection of a value for all of them.

Remarks:

1) The obtained hint tracks of this set are independent from each other and, therefore, are proper for multicasting of scalable video—the QoS of each layer can independently be negotiated with the network.
2) Even though each layer is streamed in an optimal way individually, the overall usage of network resource in this fashion is suboptimal—it does not take advantage of the multiplexing gain by merging layers together. For this reason, we propose method 2 (see as follows).

Method 2 (Scheduling for Unicasting via Aggregated QoS-Layer Negotiations): Assume the same scalable video as in Method 1. The following steps lead to another multitrack-hinting solution.

1) Set $l = 1$.
2) Construct bitstream A_l by merging coding blocks that belong to layer 1 to l, increase $l = l + 1$.
3) If $l \leq L$, go to step 2) or else continue to step 4).
4) Set again $l = 1$.
5) For bitstream A_l, construct a proper TSPEC model Γ_l, and specify a receiver-side buffer condition B_l.
6) Apply the viable scheduler design presented in the previous section to subbitstream A_l independently and obtain a scheduler Δ_l.
7) Construct the hint track H_l for layers accumulated from 1 to l using the time information derived from scheduler Δ_l, increase $l = l + 1$.
8) If $l \leq L$, go to step 5) or else terminate.

The set $\{\mathrm{H}_l : l \leq L\}$ forms another multitrack-hinting solution to this particular video.

Remarks:

1) In this case, each hint track corresponds to a merged bitstream that is constructed from a subset of all layers. Apparently, these hint tracks contain overlapping video layers.
2) Each hint track H_l, combined with corresponding assumption of Γ_l and B_l, provides a viable streaming solution for the merged bitstream covering layers 1 to l.
3) The set of hint tracks is optimized for being used independently in a scalable-video-unicast scenario. Any receiver may switch among these hint tracks to adapt the transmission rate.

Both methods 1 and 2 are applied to the same video and generate two sets of multitrack hints. The server can selectively apply any one of the two according to encountered in-vehicle wireless-network conditions and video-streaming QoS requirements.

Following a similar methodology, other multitrack-hinting methods can be proposed that satisfy particular application needs, e.g., selective prefetching of content or adaptive media playout [43].

VI. Experiments

Up to this point, we have described an integrated streaming framework for QoS-enabled in-vehicle wireless networks—multitrack hints, an extension of MPEG-4 file format, for storing the hinting information (or simply, hints) that can be used to simplify the packetization procedure in the process of scalable video streaming when adaptive QoS is demanded. Within such a framework, we also developed the theoretical procedures to validate the hints (or the schedulers) in the sense that the output traffic from the server for a particular streaming application guided by the hints (or the schedulers) would not break any prenegotiated QoS agreements while sustaining an uninterrupted playback experience (termed as viability). More importantly, we developed a scheduling algorithm for arbitrary hints (or schedulers) that would automatically be viable under the assumed network and buffer conditions. For such a framework to work under adaptive QoS, the core component would be the viable scheduling algorithm. The rest of the framework is just a flexible syntactic specification that provides a data structure that can wrap around the timing information (or the hints) generated by the scheduling algorithm.

In this section, we conduct simulation experiments using a QoS-enabled wireless-network simulator to verify the theoretical conclusions that have been developed in the analysis of this paper. We are interested to demonstrate the following results.

1) The developed scheduling algorithm can generate schedulers (or hints) that are viable as expected.
2) Such algorithms can be applied to seek a proper streaming solution for a scalable bitstream under particular network or buffer constraints.

Both properties are very important for in-vehicle wireless video streaming as they ensure uninterrupted playback, satisfying predetermined network and video QoS guarantees.

A. QoS-Enabled Wireless Network

We assume that the underlying in-vehicle network provides QoS to video-streaming applications such that the admission control will be performed by the network and that some QoS-negotiation mechanism is available to the involved server and receiver to submit QoS requests to the network such that end-to-end connections can be established. During the negotiation, we assume that the TSPEC model is used by the application and the in-vehicle network to exchange information on the agreements of accepted traffic characteristics and QoS requirements. Once the negotiation is completed and the required resources are allocated, the network will use the token-bucket model (as discussed previously) at the network interface to police the arrival traffic.

The TSPEC parameters that are engaged in the experiments include the set of parameters: $\{R_{\max}, R_a, \sigma, D_{\max}, K_{\max}\}$. At the receiver, the buffer condition is specified by the pair of parameters (T_B, T_D).

B. Scalable Video Coding

The scalable video bitstream used in the experiments is produced by a state-of-the-art real-time motion-compensation

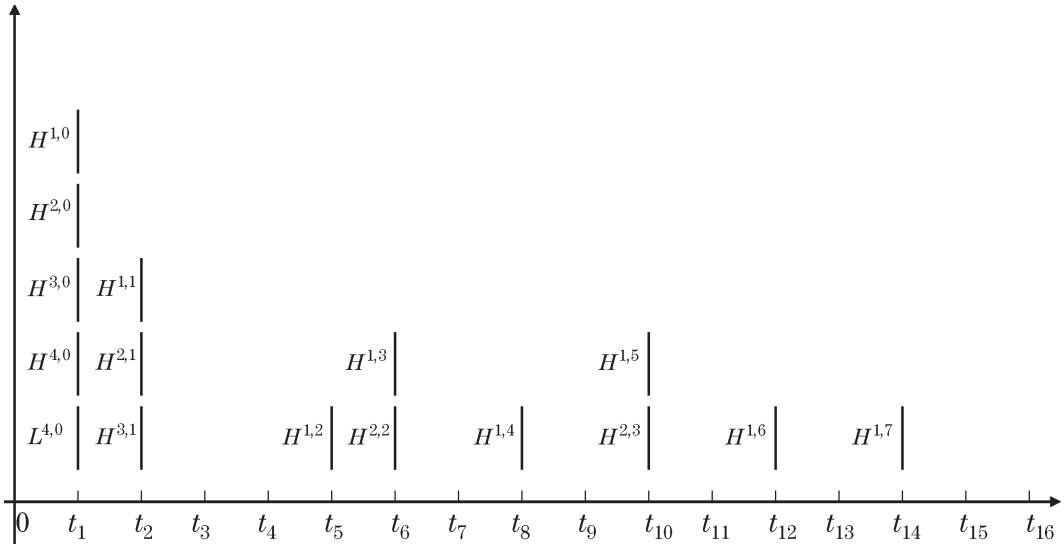


Fig. 4. Frame-decoding deadlines of a GOP consisting of 16 frames relative to the start [44].

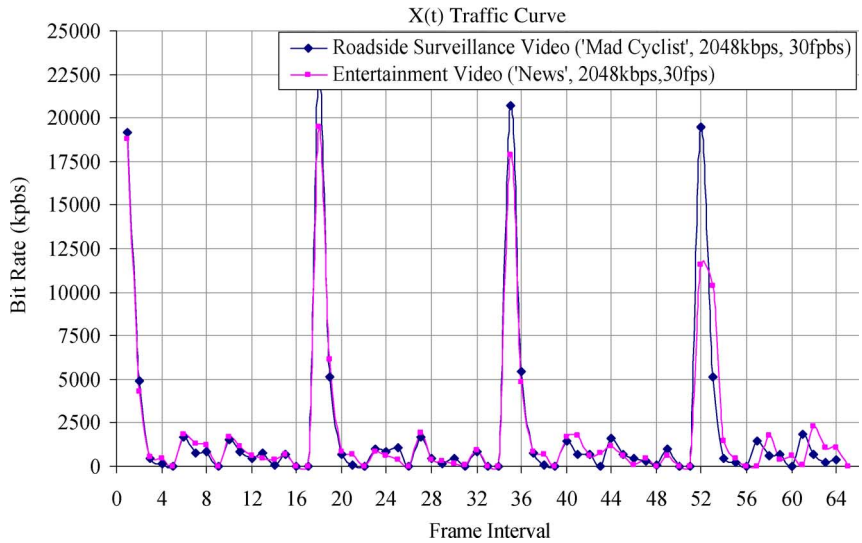


Fig. 5. Frame sizes of a selected video trace under different values for $D_{\max}$.

temporal-filtering (MCTF)-based codec called SIV, which was introduced in [42]. The compressed bitstream has a group-of-pictures (GOP) size of 16 frames. Within each GOP, four temporal-decomposition levels are performed followed by the spatial discrete wavelet transform and individual compression of each frame using embedded quantization and context-based entropy coding. The obtained frames in the temporal decomposition of each GOP are indexed as shown in Fig. 4 [44], where we also illustrate the decoding deadline of these frames relative to the start of this GOP.

The compressed bitstream generated in this way is normally referred to as VBR video. With such a bitstream, scalability can be achieved in both the spatial and temporal dimensions. For example, along the spatial dimension, since the wavelet transform is used for the compression of each individual frame [42], spatial scalability can easily be achieved by viewing the data unit from a particular spatial decomposition level as one video layer (or, even further, each individual bit-plane within a decomposition level can be viewed as an individual layer, which would be referred to as SNR scalability). Similarly, along the temporal dimension, frames from the same temporal-decomposition level can be viewed as one layer. In this case, the compressed bitstream can easily be divided into four temporal layers in a straightforward manner.

The traffic characteristics, i.e., frame size of the chosen bitstream, are shown in Fig. 5 for the first 64 frames of a typical roadside-monitoring MPEG video sequence (four GOPs—each consisting of the temporal decomposition frames of Fig. 4). Under maximum packet size $K_{\max} = 1000$ B, the frames of the bitstream can be split into data units (packets) and modeled as a sequence of pairs $(\Delta t_m^{\text{playback}}, s_m)$, as discussed in Section IV, where $m = \{1, \ldots, M_{\text{playback}}\}$ and where $M_{\text{playback}} = 8$ for each GOP, as shown from the frame-decoding deadlines of Fig. 4. To transmit such a sequence (or trace) without the use of the QoS-enabled scheduling, the average data size (in kilobits per second) per playback (decoding) deadline of each GOP is demonstrated by the dotted line in Fig. 6. It is shown that the traffic characteristics of the chosen bitstream without the use of

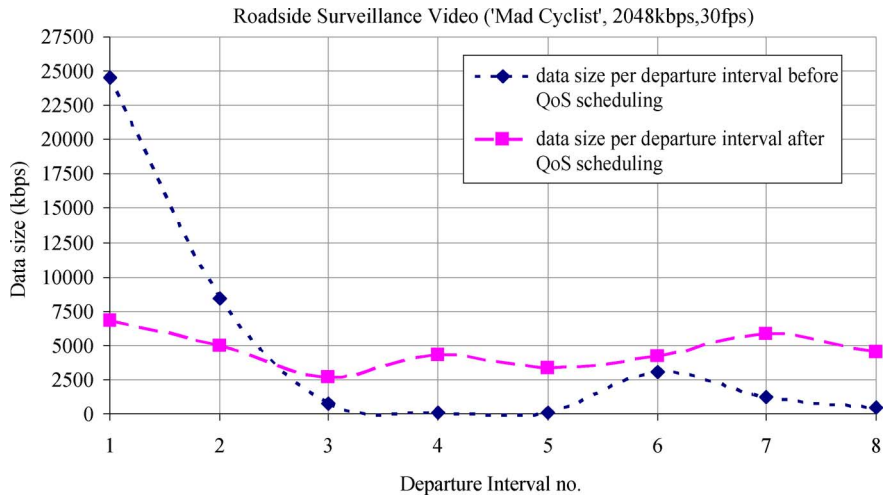


Fig. 6. Data size per departure interval for each GOP of a roadside-surveillance video trace before and after applying QoS scheduling.

TABLE II
PARAMETERS SELECTED FOR TSPEC (Γ) AND RECEIVER BUFFER (B)

$R_{\max} = 1.5R_a$	R_a (kbps)	σ (bytes)	$D_{\max}$(s)	$K_{\max}$(bytes)	T_D (s)	T_B(s)
527.30	351.53	20000	0.02	1000	2.0	0.2

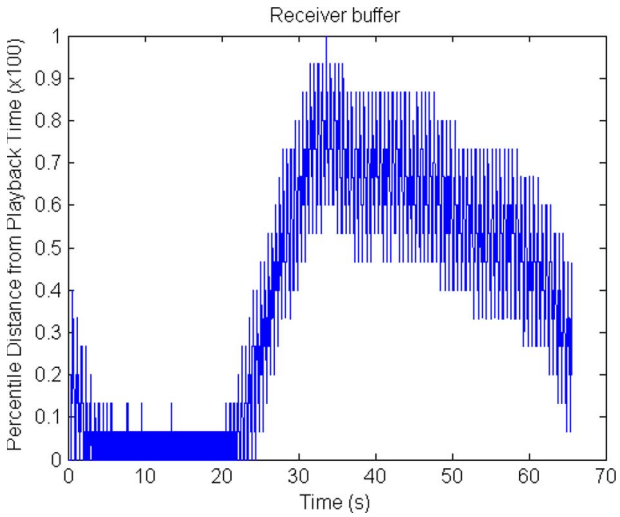


Fig. 7. Buffered video (measured in playback time) versus time after decoding start.

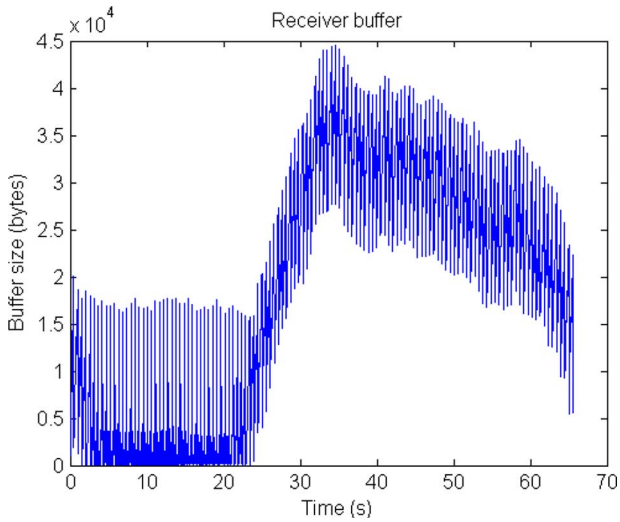


Fig. 8. Buffered video (measured in bytes) versus time after decoding starts.

QoS-enabled scheduling tend to be very bursty: an issue that was also studied in our prior work [44].

C. Results

In the following, we apply the analysis procedures and the scheduling algorithm developed in previous sections to this bitstream to show the effectiveness of the proposed methods.

1) Viable Scheduling: In this paper, we want to verify that, with a given TSPEC model (Γ) and receiver-side buffer conditions (B), if the bitstream can pass the viability-checking procedures as presented in Section IV-D, then the algorithm developed in Section V-A should generate a viable scheduler (Δ) for this bitstream. Consequently, the triplet $(\Gamma, \mathrm{B}, \Delta)$ determines a streaming solution for this bitstream. This was experimentally confirmed with all the test cases we conducted for which different Γ and B were constructed: Whenever the bitstream passed the viability checking, a viable scheduler was always obtained with the proposed scheduling algorithm. The following is such an example. Table II shows the parameters of the chosen Γ and B.

Under such a Γ and B, we confirmed that the bitstream $(\Delta t_m^{\text{playback}}, s_m)$ generated for the traffic pattern of Fig. 5 can clearly pass the viability checking, and the corresponding scheduled version of the bitstream, $(\Delta t_m^{\text{transmit}}, s_m)$ is obtained via the scheduling algorithm. For such a scheduled sequence, the average data size per playback deadline of each GOP is demonstrated by the dashed line in Fig. 6. It is clearly shown from the figure that, after carrying out the QoS-enabled packet scheduling, the scheduled sequence demonstrates much smoother departure characteristics, as compared with that of the original sequence.

TABLE III
MINIMUM BURST SIZE σ UNDER GIVEN VALUES FOR BUFFER TIME T_B

T_B (s)	0.1	0.2	0.3	0.4	0.5	0.6
σ (bytes)	22000	20000	16000	15000	14000	1000

We then apply this departure trace $(\Delta t_m^{\text{transmit}}, s_m)$ to a network simulator [45] that can set up an end-to-end network path and guarantee that the delay variation of the path is a random variable that has a limited variation range, i.e., the transmission delay is upper bounded by $D_{\max}$. Fig. 7 shows the buffered video versus time at the receiver, measured in percentile distance from the playback deadline. In this graph, 100% means maximum distance from the playback deadline, which corresponds to maximum operational buffer fullness. This is shown in Fig. 8, with the same buffered video versus time, but measured in bytes in the decoding buffer.

We observe that the buffered video never experiences any under- or overflow events at the receiver buffer, even though, occasionally, the buffered video was close to being empty. Therefore, the previously developed viability-checking method and scheduling algorithm are verified by the experiments.

D. Streaming-Solution Design

For a given bitstream, many different streaming solutions represented by triplets $(\Gamma, \mathrm{B}, \Delta)$ may possibly be designed. The developed viability-checking method and the scheduling algorithm provides a powerful mechanism for conducting such designs.

As an example to illustrate the procedures for streaming-solution design, we vary only parameters T_B and σ while keeping the rest of the parameters in Table II as before. In the paper, we first set a value for one of the two varying parameters and then seek a proper value for the other one so that the resulted solution is viable. In particular, for each given T_B, the viability-checking method is used to find a minimum σ so that the resulting triplet $(\Gamma, \mathrm{B}, \Delta)$ will be a viable streaming solution for the given bitstream. Table III shows the values of σ derived under different values of T_B.

Since the table shows the minimum burst size σ for each given T_B, when we construct the T_B versus σ plot shown in Fig. 9, the curve actually delineates the viable domain from the nonviable one, under the assumption that the other parameters remain constant.

To validate the importance of the proposed streaming-solution design, Table IV includes the average TXOP and number of admitted videos for HCCA-based [5] in-vehicle streaming using the proposed scheduling. A comparison is carried out with two recent works [44], [45]. The results have been generated with the following settings for the simulator of [45]: one access point (server) and one in-vehicle wireless station (receiver), 100-ms beacon interval, 50-ms superframe interval (SI), and 40-ms contention-free period for HCCA scheduling within each SI. Table IV shows that the proposed method

[5]HCCA: HCF Controlled Channel Access. HCF stands for Hybrid Coordination Function. It represents a new medium-access-control method proposed for IEEE 802.11e [45].

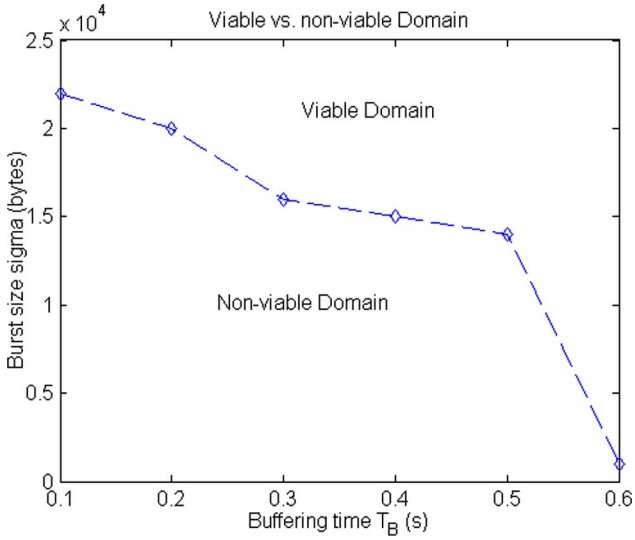


Fig. 9. Domains of viable versus nonviable combinations of T_B and σ.

TABLE IV
AVERAGE TXOP AND NUMBER OF ADMITTED VIDEO STREAMS (FLOWS) IN 802.11E HCCA FOR THE PROPOSED SCHEDULING (BASED ON NS-2 SIMULATIONS BASED ON THE SIMULATOR OF [45]). FOR COMPARISON PURPOSES, THE RESULTS OF THE SINGLE-FLOW SCHEDULING OF [44] AND THE QUEUING-BASED SCHEDULING OF [45] ARE PRESENTED

Proposed method		Single-flow of [44]		Queuing-based [45]	
TXOP (ms)	Admitted video streams	TXOP (ms)	Admitted video streams	TXOP (ms)	Admitted video streams
3.89	4	5.04	3	5.11	3
2.28	6	3.28	5	3.95	4

increases the number of admitted video streams to HCCA because it can precisely delineate the viable domain for a scheduler and guarantee the generation of such a scheduler for the input video stream.

Similar experiments can also be performed for other possible pairs such as $(T_B, R_{\max})$, (T_B, R_a), or even $(R_{\max}, \sigma)$, etc. Furthermore, these design processes can even be applied independently to layers of the bitstream. In this way, multiple hint tracks, as discussed in previous sections, are easily obtained (see next section).

These results highlight the benefits associated with the potential deployment of existing wireless-LAN technology for connectivity within state-of-the-art vehicles. Using such an infrastructure will lead to a reduction of wires within the vehicle, enhanced support for driver assistance through surveillance, and adaptability to various applications and streaming conditions, while, at the same time, ensuring proper video delivery for high-quality in-vehicle entertainment or in-vehicle critical-safety-surveillance applications.

The proposed hinting method (or algorithms) can also be applied directly to the bitstream layers generated by multitrack

TABLE V
TSPEC PARAMETERS AND BUFFER MODELS OBTAINED FROM MULTITRACK-HINTING EXPERIMENTS WITH FOUR LAYERS, CORRESPONDING TO THE FOUR TEMPORAL-DECOMPOSITION LEVELS OF THE MCTF CODEC

Layer(s)	$R_{\max}$	R_a (kbps)	σ (bytes)	$D_{\max}$ (s)	$K_{\max}$ (bytes)	T_B (s)	T_D (s)
Layer 1	6.0 R_a	19.04	22000	0.02	1000	0.6	4.0
Layer 2	6.0 R_a	66.96	22000	0.02	1000	0.6	6.0
Layer 3	6.0 R_a	64.96	22000	0.02	1000	0.6	4.0
Layer 4	7.0 R_a	200.56	22000	0.02	1000	0.6	3.0
Layer 4&3	3.0 R_a	265.52	22000	0.02	1000	0.3	2.0
Layer 4&3&2	2.0 R_a	332.48	22000	0.02	1000	0.3	2.0
Layer 4&3&2&1	1.5 R_a	351.53	22000	0.02	1000	0.1	2.0

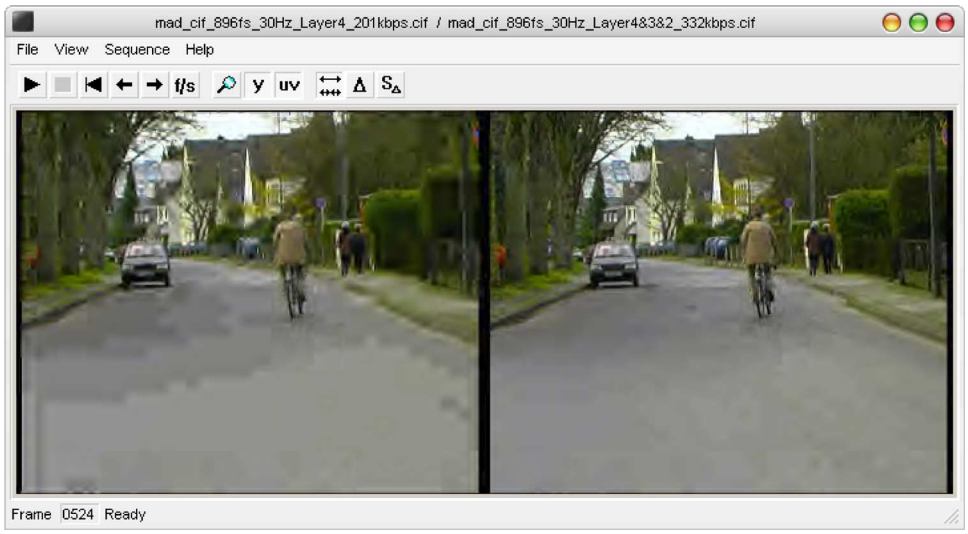


Fig. 10. Typical visual quality for a roadside-surveillance video under different combination of layers corresponding to the results of Table V. (Left) Layer 4 only. (Right) Layers 4, 3, and 2.

TABLE VI
CATEGORIES OF IN-VEHICLE VIDEO STREAMS AND THEIR CORRESPONDING QOS REQUIREMENTS

Video Category	Number of quality layers required (following Table 5)	Worst-case Delay ($D_{\max}$) and max. buffer time
Real-time road surveillance from cameras mounted on the vehicle	Layer 4&3&2&1	30 ms, 50 ms
Entertainment video streaming	Layer 4&3&2	2~3 sec, 5 sec
Road surveillance from remote roadside cameras on the preplanned route	Layer 4, or Layer4&3	5~8 sec, 10 sec

hints. In this paper, we follow multitrack-hinting methods 1 and 2 (as discussed in Section III-B) to generate departure traces [i.e., generate a trace of $(\Delta t_m^{\text{transmit}}, s_m)$] for each layer, or a combination of layers, using the scheduling algorithm. In the design of the streaming solution for each layer, we set upper bounds for R_a, σ, $D_{\max}$, and $K_{\max}$ and then tried to find the minimum viable combination of ($R_{\max}$, T_D, T_B). The corresponding TSPEC and the obtained buffer-model parameters are summarized in Table V. They were derived by applying the scheduling algorithm to either each individual layer (following method 1 of Section V-C) or to combinations of layers (following method 2 of Section V-C). A typical visual-quality example for a roadside-surveillance-camera video corresponding to the results of Table V is shown in Fig. 10. As shown in the figure, increasing the number of layers improves the quality. Notice that, even though, for both cases of the figure, the visual quality is also impaired from the interference and noise present at the physical layer (which in the indicated case led to approximately 8% packet loss), the use of a scalable coder and the QoS-reservation mechanism that enables contention-free access to the medium ensures robustness to transmission errors. Based on our experiments, we present in Table VI a summary of different layer requirements, as well as delay constraints for various categories of in-vehicle video streams.

When applying the departure traces to the network simulator [45], we verified that no buffer under- or overflow event occurred for any of the traces, which demonstrates that all the obtained streaming solutions for the individual layers or combination of layers are viable.

It is not surprising to observe that the peak-rate ($R_{\max}$) requirement of a streaming solution is reduced when the associated bitstream is a combination of layers: More layers merged together lead to smaller peak-rate requirements for the TSPEC. The same trend is also observed with regard to T_D and T_B requirements.

VII. Conclusion

We proposed several streaming and packet-scheduling algorithms for simple, flexible, and efficient transmission of video bitstreams over QoS-enabled in-vehicle IP networks. For a given layered video bitstream, these algorithms can quantitatively predict the viability of a particular streaming solution for a selected TSPEC model, a receiver-buffer condition, and a packet-scheduling strategy. When combined with the multitrack-hinting format that was also proposed in this paper, these methods form a flexible and efficient framework that can effectively perform adaptive video streaming over QoS-enabled wireless networks. This makes the proposed algorithms particularly suitable for upcoming in-vehicle wireless media networks where different streams have different quality and delay requirements (e.g., surveillance versus entertainment—Table VI), and the streaming conditions are diverse due to interference and varying QoS requirements. Our simulation results demonstrate the practical viability of the derived scheduling solutions in terms of compliance to the given TSPEC. In addition, network simulations show that the derived viable solutions avoid buffer over- or underflow at the receiver side and enable a higher number of admitted streams in comparison to other approaches from the literature. These features, combined with the use of multiple tracks via the proposed hinting mechanism, enable the construction of arbitrary streaming solutions, simultaneously satisfying transmission, receiver-buffer, and video-quality constraints for the diverse conditions of in-vehicle wireless multimedia networks. Future research will investigate the deployment of a proposed solution in a real-world vehicle environment and try to quantify the performance of the proposed solution in the presence of additional interference. Moreover, various competing physical-layer standards will be investigated in conjunction to our proposed higher layer solution.

References

[1] *Fujitsu Develops SmartCODEC (TM) Streamlined Image Compression Technology for Automotive Applications*. [Online]. Available: http://www.fujitsu.com/global/news/pr/archives/month/2006/20060921-01.html

[2] G. Leen and D. Heffernan, "Expanding automotive electronic systems," *Computer*, vol. 35, no. 1, pp. 88–93, Jan. 2002.

[3] G. Leen and D. Heffernan, "Vehicles without wires," *IET Comput. Control Eng. J.*, vol. 12, no. 5, pp. 205–211, Oct. 2001.

[4] K. Uehara, Y. Watanabe, H. Sunahara, O. Nakamura, and J. Murai, "InternetCAR—Internet connected automobiles," in *Proc. Internet Summit, INET*, in 1998. [Online]. Available: http://www.isoc.org/inet98/proceedings/1f/1f_2.htm

[5] T. Ernst, K. Uehara, and K. Mitsuya, "Network mobility from the Internet-CAR perspective," in *Proc. 17th Int. Conf. AINA*, Mar. 2003, pp. 19–25.

[6] J. Yin, T. Elbatt, G. Yeung, B. Ryu, S. Habermas, H. Krishnan, and T. Talty, "Performance evaluation of safety applications over DSRC vehicular ad hoc networks," in *Proc. 1st ACM Int. Workshop Veh. Ad-Hoc Netw.*, 2004, pp. 1–9.

[7] A. Campbell and G. Coulson, "QoS adaptive transports: Delivering scalable media to the desktop," *IEEE Netw.*, vol. 11, no. 2, pp. 18–27, Mar./Apr. 1997.

[8] B. Li, D. Xu, K. Nahrstedt, and J. W.-S. Liu, "End-to-end QoS support for adaptive applications over the Internet," in *Proc. SPIE—Symp. Voice, Video Data Communications*, Boston, MA, Nov. 1998, pp. 166–176.

[9] S. McCanne, M. Vetterli, and V. Jacobson, "Low complexity video coding for receiver-driven layered multicast," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 6, pp. 983–1001, Aug. 1997.

[10] N. Yeadon, "Quality of service filters for multimedia communications," Ph.D. dissertation, Lancaster Univ., Lancaster, U.K., May 1996.

[11] *Compressed Video Over Networks*, M. T. Sun and A. R. Reibman, Eds. New York: Marcel Dekker, 2001.

[12] *IEEE 802.11e/D5.0, Draft Supplement to Part 11: Wireless Medium Access Control (MAC) and physical layer (PHY) specifications: Medium Access Control (MAC) Enhancements for Quality of Service (QoS)*, Jun. 2003.

[13] C. Krasic and J. Walpole, "QoS scalability for streamed media delivery," Oregon Graduate Inst. Sci. Technol., Beaverton, OR, Tech. Rep. OGI CSE-99-011, Sep. 1999.

[14] Telcordia Technologies, *Next Generation Network (NGN) Services*. white paper. [Online]. Available: http://www.mobilein.com/NGN_Svcs_WP.pdf

[15] European Telecommunications Standards Institute (ETSI), *Digital Cellular Telecommunications System (Phase 2+); Universal Mobile Telecommunications System (UMTS); Service Requirements for the Internet Protocol (IP) Multimedia Core Network Subsystem (IMS); Stage 1*. (3GPP TS 22.228 version 5.7.0 Rel. 5).

[16] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, *Resource ReSerVation Protocol (RSVP)—Version 1 Functional Specification*, 1997. IETF RFC 2205.

[17] H. M. Radha, M. van der Schaar, and Y. Chen, "The MPEG-4 fine-grained scalable video coding method for multimediastreaming over IP," *IEEE Trans. Multimedia*, vol. 3, no. 1, pp. 53–68, Mar. 2001.

[18] D. Singer, W. Belknap, and G. Franceschini, *ISO Media File Format Specification: MP4 Technology Under Consideration for ISO/IEC 14496-1:2002 Amd 3*, Jul. 2001. Committee Draft, ISO/IEC JTC1/SC29/WG11, MPEG01/N4270-1.

[19] V. K. Goyal, "Multiple description coding: Compression meets the network," *IEEE Signal Process. Mag.*, vol. 18, no. 5, pp. 74–93, Sep. 2001.

[20] J. W. Woods and G. Lilienfield, "A resolution and frame-rate scalable subband/wavelet video coder," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 11, no. 9, pp. 1035–1044, Sep. 2001.

[21] D. Taubman and A. Zakhor, "Multi-rate 3-D subband coding of video," *IEEE Trans. Image Process*, vol. 3, no. 9, pp. 572–588, Sep. 1994.

[22] Y. Wang and Q. F. Zhu, "Error control and concealment for video communication: A review," *Proc. IEEE*, vol. 86, no. 5, pp. 974–997, May 1998.

[23] M. Podolsky, S. McCanne, and M. Vetterli, "Soft ARQ for layered streaming video," Comput. Sci. Dept., Univ. California, Berkeley, CA, Tech. Rep. UCB/CSD-98-1024, Nov. 1998.

[24] A. Majumda, D. G. Sachs, I. V. Kozintsev, K. Ramchandran, and M. M. Yeung, "Multicast and unicast real-time video streaming over wireless LANs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 6, pp. 524–534, Jun. 2002.

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

[26] Z. Miao and A. Ortega, "Expected run-time distortion based scheduling for delivery of scalable media," in *Proc. Int. Packet Video Workshop*, Pittsburgh, PA, Apr. 2002.

[27] G. Cheung and W. Tan, "Packet scheduling of streaming video with flexible reference frame using dynamic programming and integer rounding," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2006, pp. 1969–1972.

[28] J. Chakareski and P. Frossard, "Distributed packet scheduling of multiple video streams over shared communication resources," in *Proc. Multimedia Signal Process.*, 2005, pp. 1–4.

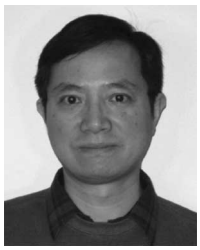
[29] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *Proc. ACM SIGCOMM*, 1996, pp. 117–130.

[30] R. Rejaie, M. Handley, and D. Estrin, "Layered quality adaptation for Internet video streaming," *IEEE J. Sel. Areas Commun.—Special Issue Internet QoS*, vol. 18, no. 12, pp. 2530–2543, Dec. 2000.

[31] Y. Cui and K. Nahrstedt, "Layered peer-to-peer streaming," in *Proc. ACM NOSSDAV*, Jun. 2003, pp. 162–171.

[32] J. R. Ohm, M. van der Schaar, and J. W. Woods, "Interframe wavelet coding—Motion picture representation for universal scalability," *Signal Process. Image Commun.—Special Issue Digital Cinema*, vol. 19, no. 9, pp. 877–908, Oct. 2004.

[33] H. Schwarz, T. Hinz, H. Kirchhoffer, D. Marpe, and T. Wiegand, *Technical Description of the HHI Proposal for SVC CE1*, Oct. 2004. ISO/IEC JTC1/SC29/WG11 (MPEG), m11244.
[34] Q. Li and M. van der Schaar, *A Flexible Streaming Architecture for Efficient Scalable Coded Video Transmission Over IP Networks*, Oct. 2002. ISO/IEC JTC 1/SC 29/WG 11 (MPEG), m8944.
[35] M. Hicks, A. Nagarajan, and R. van Renesse, "User-specified adaptive scheduling in a streaming media network," in *Proc. IEEE Open Architectures Netw. Program*, Apr. 2003, pp. 87–96.
[36] J. Shin, "Dynamic QoS mapping control for streaming video in relative service differentiation networks," *Eur. Trans. Telecommun.*, vol. 12, no. 13, pp. 217–229, 2001.
[37] C.-Y. Hsu, "Rate control for video transmission over variable rate channels," Ph.D. dissertation, Dept. Elect. Comput. Eng., Univ. Southern California, Los Angeles, CA, 1998.
[38] J. Ribas-Corbera, P. Chou, and S. L. Regunathan, "A generalized hypothetical reference decoder for H.264/AVC," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 674–687, Jul. 2003.
[39] H. Radha, Y. Chen, K. Parthasarathy, and R. Cohen, "Scalable Internet video using MPEG-4," *Signal Process. Image Commun.*, vol. 15, no. 1, pp. 95–126, Sep. 1999.
[40] J. Chakareski, J. G. Apostolopoulos, S. Wee, W. Tan, and B. Girod, "Rate-distortion hint tracks for adaptive video streaming," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 15, no. 10, pp. 1257–1269, Oct. 2005.
[41] M. van der Schaar and Y. Andreopoulos, "Rate-distortion-complexity modeling for network and receiver aware," *IEEE Trans. Multimedia*, vol. 7, no. 3, pp. 471–479, Jun. 2005.
[42] D. Taubman, D. Maestroni, R. Mathew, and S. Tubaro, *SVC Core Experiment 1, Description of UNSW Contribution*, Oct. 2004. ISO/IEC JTC1/SC29/WG11 (MPEG), m11441.
[43] M. Kalman, E. Steinbach, and B. Girod, "Adaptive media playout for low-delay video streaming over error-prone channels," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 6, pp. 841–851, Jun. 2004.
[44] M. van der Schaar, Y. Andreopoulos, and Z. Hu, "Optimized scalable video streaming over IEEE802.11a/e HCCA wireless networks under delay constraints," *IEEE Trans. Mobile Comput.*, vol. 5, no. 6, pp. 755–768, Jun. 2006.
[45] P. Ansel, Q. Ni, and T. Turletti, "An efficient scheduling scheme for IEEE 802.11E," in *Proc. IEEE Workshop Model. Optimization Mobile, Ad-Hoc Wireless Netw. (WiOpt)*, Cambridge, U.K., Mar. 2004. NS-2 802.11e HCCA software simulator. [Online]. Available: http://www-sop.inria.fr/planete/qni/fhcf/



Qiong Li (S'98–M'00) received the B.E. and M.E. degrees from Hefei University of Technology, Hefei, China, in 1986 and 1989, respectively, the Ph.D. degree in electric machines from Tsinghua University, Beijing, China, in 1995, and the Ph.D. degree in electrical and computer engineering from the University of Delaware, Newark, in 2000.

From 2000 to 2003, he was a Senior Member of the Research Staff with Philips Research USA, Briarcliff Manor, NY, and from 2003 to 2006, he was a Principal Network Architect with BrainMedia, NY. He is currently the Team Lead of Applications Software with the Diabetes Care Division, Bayer HealthCare, Tarrytown, NY. His research interests include computer networks, wireless communications, multimedia communications, and distributed medical-information-management system with emphasis on active queue management, cross-layer optimization, protocol design, streaming-server architecture, and human–machine interaction.

Yiannis Andreopoulos (M'00) received the Diploma in electrical engineering and the M.Sc. degree in signal processing systems from the University of Patras, Patras, Greece, in 1999 and 2000, respectively, and the Ph.D. degree in applied sciences with a thesis on scalable video coding and complexity modeling for multimedia systems from the University of Brussels, Brussels, Belgium, in 2005.

During his thesis work, he participated and was supported by the European Union Information Society Technologies-project Metadata for Advanced Scalable Video Coding Tools: a Future and Emerging Technologies project. During his postdoctoral work with the University of California at Los Angeles, he performed research on cross-layer optimization of wireless media systems, video streaming, and theoretical aspects of rate-distortion-complexity modeling for multimedia systems. Since October 2006, he has been a Lecturer with the Department of Electronic Engineering, Queen Mary University of London, London, U.K.

Dr. Andreopoulos made several decisive contributions to the ISO/IEC JTC1/SC29/WG11 (Motion Picture Experts Group) committee from 2002 to 2003 in the early exploration on scalable video coding, which has now moved into the standardization phase. In 2007, he was the recipient of the "Most-Cited Paper" award from the Elsevier EURASIP Journal *Signal Processing: Image Communication*, based on the number of citations his 2004 article "In-band motion compensated temporal filtering" received within a three-year period.

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

She is currently an Associate Professor with the Department of Electrical Engineering, University of California at Los Angeles. She has been an active participant in the ISO Motion Picture Expert Group Standard since 1999, to which she has made more than 50 contributions and for which she received three ISO recognition awards. She is the holder of 28 granted U.S. patents with several more pending.

Dr. van der Schaar was also elected as a member of the Technical Committees on Multimedia Signal Processing and on Image and Multiple Dimensional Signal Processing of the IEEE Signal Processing Society. She was an Associate Editor of the IEEE TRANSACTIONS ON MULTIMEDIA and the SPIE *Electronic Imaging Journal*. She is currently an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEM FOR VIDEO TECHNOLOGY and of the IEEE SIGNAL PROCESSING LETTERS. She was the recipient of the NSF CAREER Award in 2004, the IBM Faculty Award in 2005, the Okawa Foundation Award in 2006, the Best IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY Paper Award in 2005 and 2007, and the Most Cited Paper Award from the EURASIP Journal *Signal Processing: Image Communication* between 2004 and 2006. She is also the Coeditor (with P. Chou) of the book *Multimedia Over IP and Wireless Networks: Compression, Networking, and Systems.*