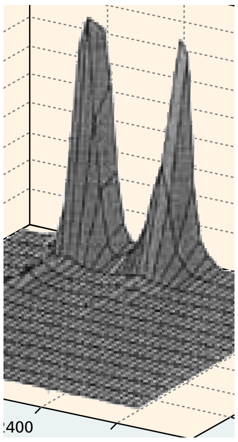
UNCOORDINATED REAL-TIME VIDEO TRANSMISSION IN WIRELESS MULTICODE CDMA SYSTEMS: AN SMPT-BASED APPROACH

FRANK FITZEK, ACTICOM GMBH
MARTIN REISSLEIN, ARIZONA STATE UNIVERSITY
ADAM WOLISZ, TECHNICAL UNIVERSITY BERLIN



The transport of video over wireless links is a challenging problem due to the stringent playout deadlines of the video frames and the unreliability of the wireless links.

ABSTRACT

We consider the real-time transmission of encoded video from distributed, uncoordinated wireless terminals to a central base station in a multi-code CDMA system. Our approach is to employ the recently proposed simultaneous MAC packet transmission (SMPT) approach at the data link layer (in conjunction with UDP at the transport layer). We consider the real-time transmission of both video encoded in an open loop (i.e., without rate control) and video encoded in a closed loop (i.e., with rate control). We conduct extensive simulations and study quantitatively the trade-off between video quality, transmission delay (and jitter), and number of supported video streams (capacity). We find that the simple-to-deploy SMPT approach achieves significantly higher video quality and smaller delays than the conventional sequential transmission approach, while ensuring high capacity. In typical scenarios, with SMPT the probability of in-time video frame delivery is more than twice as large as with sequential transmission (for given delay bounds). Our results provide guidelines for the design and dimensioning of cellular wireless systems as well as ad hoc wireless systems.

INTRODUCTION

Video traffic is expected to account for a large fraction of the traffic in future wireless networks. Generally, the transport of video over wireless links is a very challenging problem. This is due to the stringent playout deadlines of the video frames (as well as the variability of the frame sizes for open loop encodings) and the unreliability of the wireless links. For streaming services (e.g., the Web-based streaming of prerecorded video clips) that do not require real-time interactions, these challenges can be overcome by relaxing the timing constraints with receiver side buffering and taking advantage of multi-user diversity. Multi-user diversity is based on the observation that in a typical wireless system with multiple users, at any point in time, some users experience favorable transmission conditions on their wireless links, while others experience adverse transmission conditions. (This is due to typically location-dependent, time-varying, and bursty errors on wireless links.) The basic idea of multi-user diversity is to transmit at any point in time only over the links currently experiencing favorable transmission conditions. If a link is currently experiencing adverse transmission conditions (which would require high power levels or several retransmissions), no transmissions are scheduled for this link, and the receiver continues video playback from its buffer. (See, e.g., [1] for a detailed study of such a system.) Applications that require real-time interactions (e.g., videoconferencing, telemedicine, and games), however, allow only very limited relaxation of the timing constraints (i.e., very limited receiver side buffering). In addition, taking advantage of multi-user diversity typically requires central scheduling of packet transmissions. This central scheduling requires an established signaling and coordination structure, which is not available in the emerging ad hoc wireless networks. In summary, *real-time* video transmission by *uncoordinated* wireless terminals is especially challenging.

In this article we develop and evaluate simple yet quite effective techniques for real-time video

Supported in part by the National Science Foundation through grant Career ANI–0133252, the State of Arizona through the IT301 initiative, and the BMBF (German Science Ministry) through the TransiNet project.

transmission by distributed uncoordinated wireless terminals in a common interference environment (e.g., local cluster in an ad hoc network or cell in a cellular network). Our approach is to employ simultaneous medium access control (MAC) packet transmission (SMPT) techniques recently developed and evaluated for the transmission of data traffic (e.g., FTP traffic). The basic idea of SMPT is to transmit multiple packets (on multiple CDMA codes) in parallel to make up for packets lost due to errors on the wireless channel. While SMPT techniques have been studied extensively for the transmission of data traffic without any fixed deadlines [2, references therein], they have not yet been studied in the context of continuous media (e.g., video) with strict timing constraints (except for the initial study [3], which focused on the stabilization of TCP throughput for video; see a later section for details).

In this article we study the real-time transmission (with UDP as the transport protocol) of video encoded with rate control as well as video encoded without rate control. Video encoded with rate control has typically only moderate bit rate variations on typically small timescales. On the other hand, encoding video without rate control, which requires less complexity (and saves cost and energy at the wireless device), results in video traffic with large bit rate variations over many different timescales (including long ones) [4]. In addition, video encoded without rate control typically exhibits long-range dependence (or self-similarity). Consequently, we find that different variations of the SMPT technique are appropriate for the different types of encoded video. The so-called *slow-healing* SMPT mechanism, which resorts to transmitting multiple MAC packets in parallel only after suffering loss on the wireless link, is suited to relatively smooth video encoded with rate control. On the other hand, the so-called *fast-start* and *slow-start* SMPT mechanisms, which transmit from the outset on multiple parallel codes, are suited to bursty video encoded without rate control.

Throughout our study we pay close attention to implementation aspects. Our focus is on techniques that are simple to deploy in practical systems and give tangible performance improvements, rather than techniques that are optimized for performance at the expense of increased complexity. The presented SMPT mechanisms operate exclusively at the data link layer and do not require any higher-layer information. Also, the SMPT mechanisms do not require any coordination among the wireless terminals in the cluster of an ad hoc network or the cell of a cellular network. The SMPT mechanism running in a given wireless terminal schedules the MAC packet transmissions completely independent from the other wireless terminals in the cluster (or cell). The wireless terminals "feel" each other only through the interference level generated by their transmissions. By varying the number of used code-division multiple access (CDMA) codes and monitoring the success of its own transmissions, the SMPT mechanism in a given wireless terminal probes the capacity of the cluster (cell). In typical scenarios the SMPT mechanism more than doubles the probability of in-time video frame delivery.

Related Work

The area of video transmission in wireless environments has attracted a great deal of attention recently and a large body of literature on the topic has emerged. Several schemes have been proposed for improving the video quality by employing adaptive video coding schemes, see for instance [5, 6]. Our approach is orthogonal to these adaptive video encoding schemes in that we adapt the scheduling of the transmissions of the MAC packets carrying the video (instead of the encoding of the video). A hybrid scheme employing forward error correction (FEC) and automatic repeat request (ARQ) is proposed in [7]. The ARQ component in [7] does not transmit multiple packets simultaneously in response to lost packets. Thus, our SMPT approach is orthogonal to [7] in that it may be used as a refined ARQ component in the hybrid scheme of [7].

In [3] we studied the streaming of video using TCP as the transport protocol. This article differs from [3] in two important aspects. First, [3] focuses on video streaming with delays on the order of 1 s. In this article, on the other hand, we focus on real-time transmission with delays on the order of less than a few hundred milliseconds, which allow of real-time communication, as needed for interactive applications, such as videoconferences, telemedicine, or games. Second, [3] studied a "reliable" video service that temporarily pauses the video playout when the client's consumption exceeded the supply of video information (and does not drop any video frames). This article, on the other hand, considers a real-time (albeit lossy) video service that trades off tight delay bounds for some small loss (whereas [3] guarantees no loss at the expense of large delay and playback pauses).

In [1] we developed a prefetching scheme for the streaming of video in wireless environments. This prefetching scheme may be employed for downlink streaming as well as uplink streaming, and does also support real-time transmission when the tolerable delays are on the order of a few video frame periods. The fundamental difference between the scheme studied in this article and the prefetching scheme is that [1] requires centralized scheduling and knowledge of the video traffic (frame sizes, playout deadlines), whereas the scheme studied in this article is for the uncoordinated transmission by the link layers of distributed wireless terminals (and does not require knowledge of the traffic).

We note that video transmission in multicode CDMA systems is also studied in [8]. The scheme proposed in [8] is similar to ours in that multiple codes are used in parallel to accommodate the variable sized video frames (of a given video). The main difference between [8] and our scheme is that [8] requires a significant amount of coordination among the videos being transmitted (e.g., the video streams are aligned such that a typically large intracoded I-frame of one video stream does not coincide with the I-frame of another video stream). Our scheme, on the other hand, does not require any coordination among the ongoing video flows, and is thus well suited for wireless networks with little or no

By varying the number of used CDMA codes and monitoring the success of its own transmissions, the SMPT mechanism in a given wireless terminal probes the capacity of the cluster (cell). In typical scenarios the SMPT mechanism more than doubles the probability of in-time video frame delivery.

SMPT does not require any coordination among the ongoing video flows, and is thus well suited for wireless networks with little or no coordination among the wireless terminals, such as ad hoc networks.

coordination among the wireless terminals, such as ad hoc networks.

We finally note that mechanisms for the transmission of scalable video (i.e., video encoded into multiple layers) over wireless links are discussed in a number of studies, such as [9, 10]. These mechanisms strive to schedule the individual layers to maximize overall video quality. Throughout this article we focus on video encoded into a single layer (i.e., nonscalable).

Real-Time Video Transmission with SMPT

In this section we discuss the transmission of video traffic using the SMPT approach. At the sending wireless terminal the video is encoded into video frames. Each video frame is immediately passed down the networking protocol stack through the UDP transport protocol and the IP network protocol (each of which appends its header to the video frame). For simplicity, we assume that each video frame is encapsulated into a single transport layer segment. (Thus, we use the terms *segment* and *video frame* interchangeably in our discussion of the lower-layer mechanisms.) At the data link layer, the segment (video frame plus UDP and IP protocol headers) is partitioned into fixed size link-layer packet data units (LPDUs). (Padding is used to fill the last LPDU for a given video frame.) With the standard *sequential* transmission approach, the LPDUs are transmitted in send-and-wait fashion using one single CDMA code. (Throughout, a slotted timing structure is assumed, where one CDMA code provides sufficient transmission capacity to transmit one LPDU in one slot of duration τ_{slot}.) An LPDU successfully received by the receiving terminal is immediately acknowledged, and the next LPDU is transmitted in the subsequent slot. (We assume that the acknowledgment for a successful LPDU is returned and processed before the next LPDU is sent in the next slot; this is feasible with typical hardware configurations of wireless communication systems.) If an LPDU is lost on the wireless link, the sender retransmits the lost LPDU until it is received successfully, and then moves on to the next LPDU [11].

Slow-Healing SMPT for Video Encoded with Rate Control

We now briefly review the SMPT approach (referring the interested reader to [11] for a more detailed discussion) and describe how to transmit video encoded with rate control in the SMPT approach. The basic idea of SMPT is to transmit multiple LPDUs in parallel using multiple CDMA codes (one for each LPDU) when an LPDU has been lost on the wireless link. Suppose an LPDU is not successfully received (and hence not acknowledged). In the most basic SMPT scheme, in the next slot the sending terminal transmits the lost packet *and* the subsequent LPDU (which would have been transmitted in that slot, had there not been a link error) on two CDMA codes. If these LPDUs are successfully received and acknowledged, the sender returns to sending one packet using one CDMA code. Otherwise (i.e., if the packets are not successful), the terminal sends three LPDUs (the two unsuccessful LPDUs plus the LPDU next in line) using three CDMA codes. This process continues until the LPDUs are successful or the terminal has "ramped up" to using a prespecified maximum number R of CDMA codes (where typically $R = 8$, due to the physical limitations of the radio front ends of practical wireless devices; for low-cost devices typically $R = 3$). We note that modern wireless systems, such as IS-95 (Rev. B) and Universal Mobile Telecommunications System (UMTS), allow for the deployment of SMPT, since these systems provide the capability to adapt transmission rates by varying the number of used codes. (We also remark that conceptually the transmission rate could be adapted by varying the spreading gain or a combination of varying the number of codes and the spreading gain. However, to fix ideas for our study, we consider a system that varies the number of codes and keeps the spreading gain fixed.)

To save precious wireless transmission resources (and energy) as well as reduce the created interference, the ramping mechanism of SMPT can be combined with a link probing mechanism. The basic idea of link probing is to probe the link after an LPDU was not acknowledged. In our SMPT context, the sending terminal retransmits the lost LPDU (as a link probe) using only one single CDMA code until this probing LPDU is acknowledged. The terminal then starts to build the SMPT ramp to clear the backlog that has accumulated during the probing. With the slow-healing variation of SMPT the terminal ramps up by using two CDMA codes in the slot right after the probing LPDU was received and acknowledged successfully, three codes in the subsequent slot, and so on, until all R codes are used. If at any point an LPDU is not acknowledged, the terminal returns to probing and rebuilds the ramp after a successful probing LPDU. This probing refinement is especially effective with the highly correlated bursty errors of a typical wireless link (usually modeled as a two-state Markov Chain, see a later section for details) and is employed throughout the remainder of this study.

In this study on video transmission using SMPT we employ the outlined slow-healing SMPT approach for video encoded with rate control. The video frames are partitioned into LPDUs and buffered in a data link buffer of size L_{Queue}. This buffer is used to smooth out short-timescale variation of rate-controlled video (and holds the LPDUs that are backed up during link probing). The motivation for using slow-healing SMPT for rate-controlled video is that rate-controlled video typically has only moderate variations on relatively short timescales around a prespecified target bit rate [4]. Slow-healing SMPT, which strives to stabilize the throughput of the wireless link, is therefore well suited for this type of video traffic. An important advantage of slow-healing SMPT is that it generally has very good interference properties [2]; it has a small variance of the code usage in a wireless cell, reducing the demands on power control, and has small intercell interference to neighboring cells.

Fast/Slow-Start SMPT for Video Encoded Without Rate Control

For video encoded without rate control, slow-healing SMPT is not well suited. This is because video encoded without rate control exhibits a highly variable bit rate over a wide range of timescales (including long ones). A moderate-sized buffer at the link layer that is ideally drained at a constant rate is therefore not very efficient in serving this type of video traffic. Instead, the link layer needs to adapt to the varying bit rate of the video traffic. To achieve this we employ the more aggressive *slow-start* and *fast-start* SMPT mechanisms. With fast-start SMPT the wireless terminal transmits R LPDUs in parallel using R CDMA codes as long as there are backlogged LPDUs in the queue and all LPDUs are acknowledged. When an LPDU is not acknowledged the terminal begins to probe the channel with one LPDU per slot. Once the probing LPDU is acknowledged, the terminal resumes transmitting with R codes. With slow-start SMPT, on the other hand, the terminal starts building a ramp whenever more than one LPDU is in the queue (even when the backlog is not due to previous link errors). The terminal increases the number of used codes by one for each slot in which all LPDUs are acknowledged up to the maximum of R codes. When an LPDU is not acknowledged, the terminal starts probing the channel with one LPDU. Once the probing LPDU is acknowledged, the terminal resumes building the ramp (provided there are backlogged LPDUs in the queue). By striving to work off any backlog in the link layer buffer, the start SMPT mechanisms adapt to the varying bit rates. We note, however, that these start mechanisms generally lead to a higher variability of the total number of used codes in a cell of a wireless cellular system, thus putting more burden on the cell's power control. Also, the interference seen in neighboring cells is larger than in the less aggressive slow-healing approach.

Performance Metrics

Continuous media applications (e.g., video) have stringent timing constraints. For the case of video, the receiver has to decode and display a new frame with every frame period (typically 40 ms or multiples thereof) [4]. If a video frame is not completely received by its playout deadline, the receiver loses (part or all of) the frame. The loss may result in jerky motions or artifacts in the video, which reduce the perceived video quality.

For the purpose of our study we express the timing constraints in the delay and jitter bounds standardized in RFC 1193, which we briefly review here for convenience. Suppose at time t_0 a video frame is generated and instantaneously passed through the transport and network layer down to the data link layer. Suppose the video frame (plus higher-layer protocol headers) is partitioned into N LPDUs. With sequential transmission the minimal delay of the video frame is clearly

$$D_{\min} = N \cdot \tau_{\text{slot}}.$$

This minimal delay is attained when there are no errors on the wireless link (i.e., when no LPDU needs to be retransmitted). However, typically there are some wireless link errors that result in some LPDU(s) being dropped and subsequently retransmitted. Suppose that the transport layer at the receiver accepts only segments that arrive with a delay no larger than

$$D_{\max} = \min\{\tau_{\text{delay}}, D_{\min} + \tau_{\text{jitter}}\} \quad (1)$$

where τ_{delay} denotes the deterministic delay bound, and τ_{jitter} denotes the deterministic delay-jitter bound defined in [12]. We define $D_{\max}$ as the *transmission window* within which a segment has to be transmitted (delivered) from the sender to the receiver in order to be considered successful. Suppose that the segment arrives at time t_2 at the transport layer at the receiver. The delay D of a successful segment satisfies

$$D = t_2 - t_0 \leq \tau_{\text{delay}}. \quad (2)$$

The jitter J of the segment is

$$J = t_2 - (t_0 + N \cdot \tau_{\text{slot}}) \leq \tau_{\text{jitter}}. \quad (3)$$

In our quantitative study we consider the following metrics:

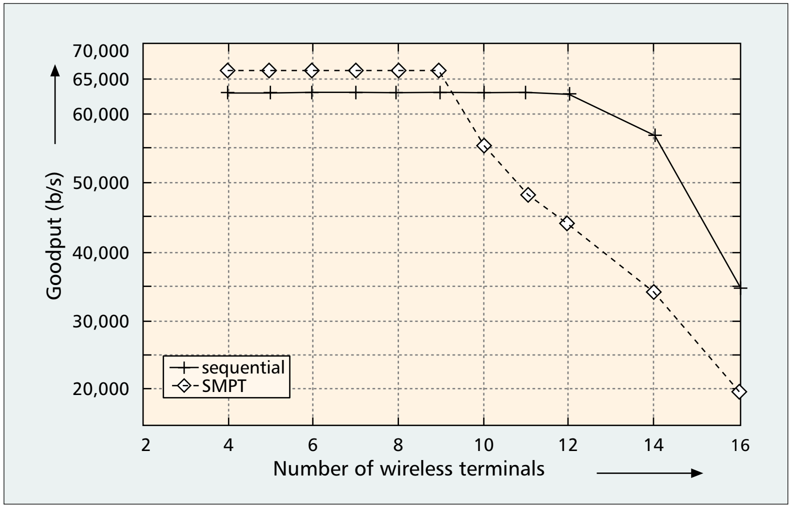
- The jitter J as defined in Eq. 3. In our simulations we record the jitter of all successful segments (video frames) and report the resulting average jitter.
- The *probability of successful video frame (segment) delivery* for a given delay constraint τ_{delay}.
- The *goodput* (in bits per second) is obtained by summing the sizes of the link layer payloads of all successfully transmitted LPDUs (in bits) and dividing this sum by the duration of a given video flow. (The link layer payload consists of the video frame plus UDP and IP headers plus padding, but does not include FEC and link layer header. Due to the UDP and IP protocol headers as well as the padding, the goodput may be larger than the average bit rate of the video streams.)

Additionally, we consider the *capacity*, which we define as the number of simultaneous video flows that can be supported in a given wireless cell while meeting specific delay constraints and goodput requirements.

Simulation Scenario

To evaluate the transmission of video using SMPT mechanisms we have developed comprehensive simulation programs based on `ptolemy` and `ns-2`. Because of space constraints we give here only a brief overview of the simulation programs and refer the interested reader to [11] for details. In our simulations we consider a scenario where multiple distributed wireless (and possibly mobile) terminals transmit video (exactly one stream per terminal) to a central base station. We chose this uplink transmission scenario within a wireless cell to fix ideas. Our SMPT mechanisms do not rely on the cellular structure; in fact with our SMPT mechanisms each wireless terminal schedules its LPDU transmissions without any knowledge of the other terminals' activities. (The wireless terminals only "feel" each other through the shared interference environment within the cell.) The simulation programs simulate the entire network protocol stack at each sending terminal and the corresponding protocol stacks at the

With slow-start SMPT, the terminal starts building a ramp whenever more than one LPDU is in the queue. The terminal increases the number of used codes by one for each slot in which all LPDUs are acknowledged up to the maximum of R codes.



■ Figure 1. *Goodput as a function of number of wireless terminals* (L_{Queue} = *100 LPDUs).*

base station (which serves as a receiver for the video streams).

For the simulation of video traffic we use frame size traces of 25 videos encoded with rate control (for the scenario with rate control) and frame size traces of 25 videos encoded without rate control (for the scenario without rate control). These traces are described in more detail at the beginnings of the next few sections. In each scenario, for each of the ongoing video streams we randomly pick one out of 25 (respective) frame size traces as well as a random phase into the selected traces.

Each video frame is encapsulated into a single UDP transport layer segment and passed down through the IP network layer to the link layer. At the link layer the video frame (plus UDP and IP headers) is partitioned into LPDUs of 128 bytes each. (80 bytes of link layer payload (video frame, UDP and IP protocol headers, padding) + 47 bytes of FEC + 1 byte of link layer header.) The LPDUs are placed into the data link buffer of default size L_{Queue} = 100 LPDUs = 12.8 kbytes.

At the physical layer, pseudo-noise spreading sequences are used. Each wireless link (consisting of up to R parallel code channels) is modeled using the two-state ("good" and "bad") Markov Chain (Gilbert-Elliot) model with typical settings for the transition probabilities between the two states [6, 11]. In the "bad" state all LPDUs sent over the link are dropped with probability one. In the "good" state an improved Gaussian approximation is used to evaluate the bit error probability on the link as a function of the total number of used pseudo-noise codes (i.e., interference level) in the cell. The LPDU drop probability is then calculated from the bit error probability assuming BCH(1023, 640, 41) forward error correction for each LPDU. The slot length is fixed at τ_{slot} = 10 ms. The bit rate for one CDMA channel (in the wireless terminal to base station direction) is 64 kb/s.

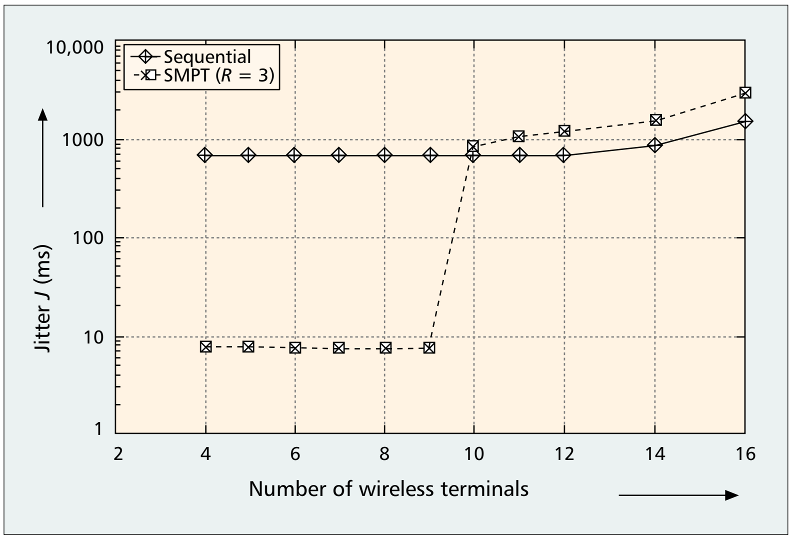
All simulations are run until the 99 percent confidence interval of the metrics of interest is less than 1 percent of the corresponding sample mean.

Simulation Results for Video Encoded With Rate Control

For the simulation of the transmission of video encoded with rate control we use the 25 frame size traces of video encoded in H.263 with a target bit rate of 64 kb/s available from [4]. These H.263 encodings are characteristic of video encoded with rate control. The video traffic has only small to moderate variations around the target bit rate. The spreading gain is set to 16 throughout this section.

Impact of the Number of Wireless Terminals

In Fig. 1 we plot the goodput as a function of the number of wireless terminals (which is equivalent to the number of ongoing video streams). We give the goodput for the sequential transmission mode and the slow-healing SMPT approach (with a maximum number of R = 3 parallel CDMA codes for a given terminal). The corresponding jitter results are presented in Fig. 2. In this experiment we do not impose any delay bound τ_{delay} or jitter bound τ_{jitter}. Thus, at the receiver, there are no video frames discarded due to violated playback deadlines. Loss occurs only when the LPDUs carrying parts of a video frame find the link layer buffer full. We observe from the figures that with SMPT up to nine video streams can be supported with a goodput of 66 kb/s and an average jitter J smaller than 10 ms. The sequential transmission mode achieves a goodput of 63 kb/s for up to 12 video streams. However, the average jitter of the sequential transmission mode is roughly 1 s throughout. This is unacceptable for real-time video transmission, especially for interactive applications, such as videoconferences. We define the *operational phase* (capacity) as the range of video streams over which the QoS provided is stable. For the scenario shown in Figs. 1 and 2, the operational



■ Figure 2. *Average jitter* J *as a function of number of wireless terminals* (L_{Queue} = *100 LPDUs).*

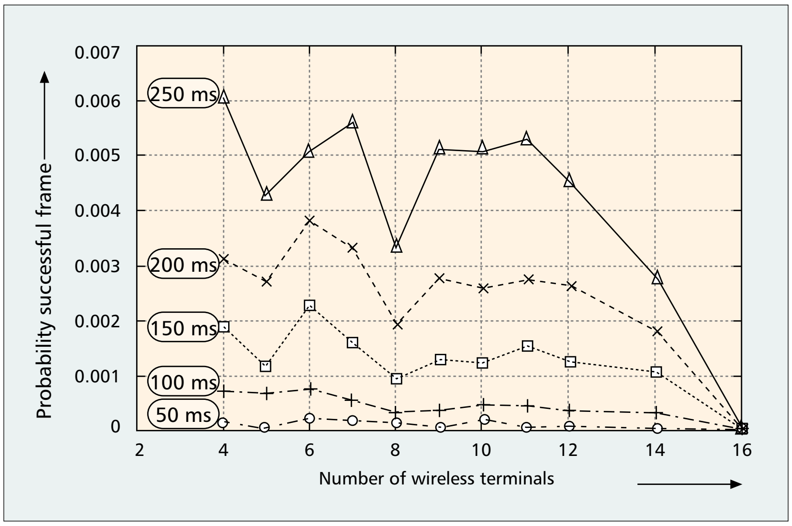
phase for the sequential transmission mode is 12 video streams and for SMPT it is nine video streams. We observe that the jitter increases steeply when the capacity of the SMPT system is exceeded, that is, when the number of video streams increases from 9 to 10 (an 11 percent increase in the cell load). To explain this, note that SMPT stabilizes the throughput by using more codes when LPDUs get backlogged in the link layer buffers of the wireless terminals. Once the traffic load is increased beyond the capacity, the interference level increases significantly, leading to more dropped LPDUs and thus more backlog. In response, SMPT uses more codes, trying to clear the backlog. However, by using more codes the interference level is further increased. This in turn leads to more dropped LPDUs and more backlog, which SMPT is not able to clear when the system is loaded beyond its capacity. As we observe from Fig. 2, when the SMPT system is loaded beyond its capacity it gives about the same jitter performance as sequential transmission.

We thus observe that there exists a trade-off between the range of the operational phase (capacity) and QoS provided in terms of goodput and jitter. For a smaller operational phase the QoS provided by SMPT is much better than the QoS provided by sequential transmission. (We note that the average jitter studied here is only a first coarse assessment of the performance. Even with an average jitter below a certain threshold τ_{jitter}, video frames may miss their deadline if the variability of the jitter is large. We therefore study the performance with respect to a fixed delay constraint in the next section and the jitter distribution in detail in a later section as well as [11].

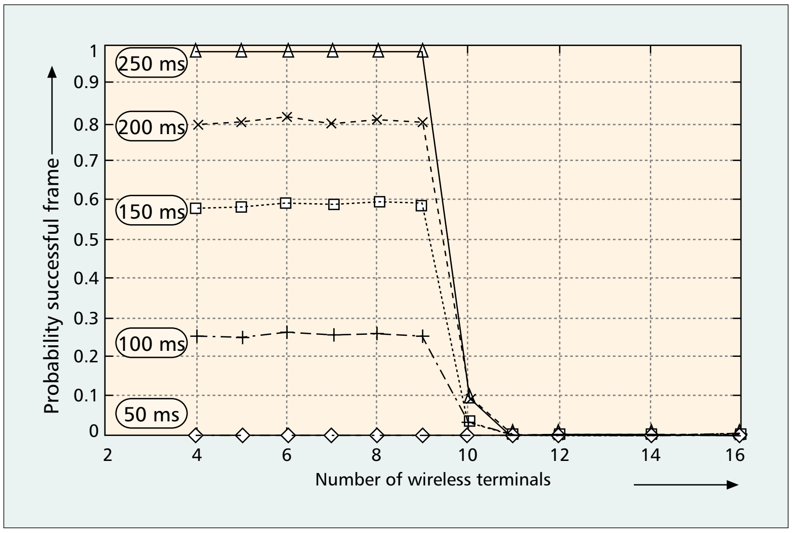
Impact of Delay Constraint τ_{delay}

In Figs. 3 and 4 we plot the probability of successfully delivering a video frame as a function of the number of ongoing video streams for the sequential transmission mode and slow-healing SMPT. We give the probability of successful video frame delivery for the delay bounds τ_{delay} = 50, 100, 150, 200, and 250 ms. We note that the link layer at the sending wireless terminal is not aware of these delay bounds. The link layer simply tries to transmit the LPDUs in its buffer; it is not aware of the fact that the LPDUs carry video frames with playout deadlines. Thus, our approach preserves the isolation of the layers of the networking protocol stack and allows for the deployment of our mechanisms in low-cost wireless terminals. At the receiver's transport layer only the video frames meeting the delay bound are passed up to the application. We observe from Fig. 3 that only a very small fraction (less than 1 percent) of the video frames is transmitted successfully with the sequential transmission mode for the chosen delay bounds. As expected, larger delay constraints result in a larger probability of successful video frame delivery. However, even for a delay bound of τ_{delay} = 250 ms, less than 1 percent of video frames are transmitted successfully. (We note that this very poor performance is due to the relatively large default size of the link layer buffer of L_{Queue} = 100 LPDUs.) The link layer is not aware of the video frame deadlines and transmits all the LPDUs in the buffer, even though they may carry video frames that have already missed their deadline. The larger the buffer the more delay the LPDUs may experience in the buffer. We study the impact of the buffer size in detail in a later section.

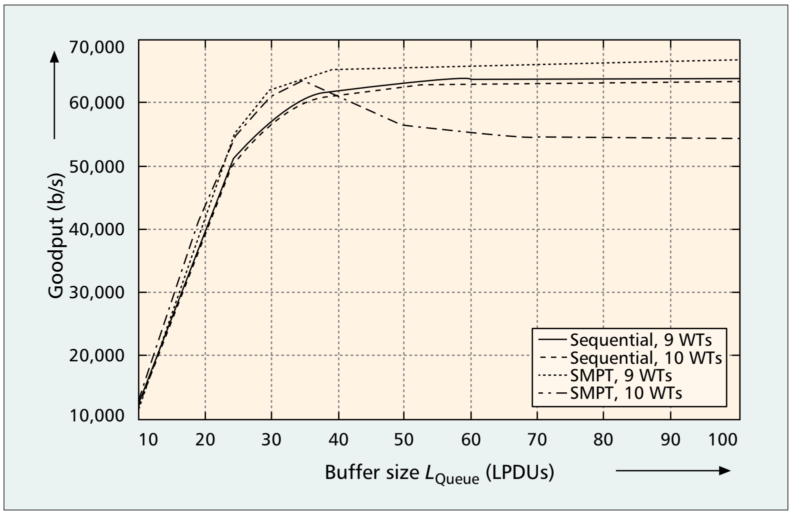
We observe from Fig. 4 that with the slow-healing SMPT mechanism the probability of sending a video frame successfully is very high. For nine and less wireless terminals (each sending one video stream) the success probability for a delay constraint of τ_{delay} = 250 ms is 98 percent. Smaller delay constraints τ_{delay} decrease the success probability, but even for τ_{delay} = 150 ms, the success probability is still around 60 percent. We conclude that for its operational phase up to nine ongoing video streams, the slow-healing SMPT mechanism achieves high probabilities of successful video



■ Figure 3. *Probability of successfully delivering a video frame for delay constraints τ_{delay} = 50, 100, 150, 200, and 250 ms for the sequential transmission mode (L_{Queue} = 100 LPDUs, fixed).*



■ Figure 4. *Probability of successfully delivering a video frame for delay constraints τ_{delay} = 50, 100, 150, 200, and 250 ms for the slow-healing SMPT mechanism (L_{Queue} = 100 LPDUs, fixed).*



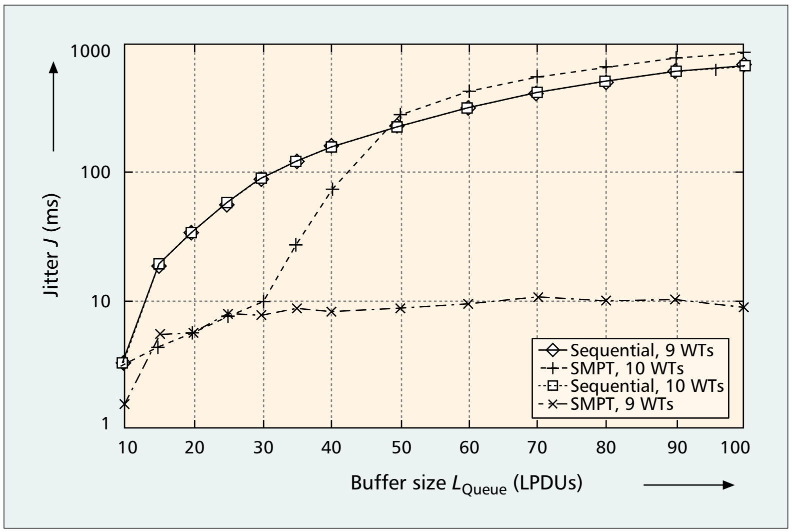
■ Figure 5. *Goodput as a function of the link buffer size* L_{Queue}.

frame transmission. With 10 wireless terminals the SMPT system is loaded beyond its capacity, and we observe a sharp dropoff in the success probability.

The Impact of Link Layer Buffer Size L_{Queue}

For the simulation results reported so far, the buffer at the link layer was set to L_{Queue} = 100 LPDUs = 12.8 kbytes. We now vary the size of the link layer buffer L_{Queue}. It is well known that smaller buffers reduce the jitter. However, smaller buffers also result in larger loss (and hence a smaller probability of successful video frame transmission), and consequently in lower goodput. We now investigate this trade-off quantitatively. Figures 5 and 6 give the goodput and jitter as functions of the link layer buffer size L_{Queue}. We investigate the situation where 9 and 10 wireless terminals transmit rate-controlled video simultaneously using either the sequential transmission mode or slow-healing SMPT. We observe that for the sequential transmission mode the cell load (either 9 or 10 WTs) has no significant impact, and as expected, both the jitter and the goodput decrease as the buffer size is decreased from the default value of 100 LPDUs to smaller values. For a buffer size of L_{Queue} = 30 LPDUs, for instance, the average jitter is J = 100 ms and the goodput is 56 kb/s. The implication of this experiment is that the performance of the sequential transmission mode is very sensitive to the link layer buffer size. For a relatively small buffer of 30 LPDUs the sequential transmission mode reaches its maximum goodput. Further increases in the buffer do not affect the goodput, but increase the jitter dramatically (and thus lead to a small probability of successful video frame delivery, as seen in the previous section). For buffer sizes between 5 and 30 LPDUs there is no significant difference between the performance of SMPT for 9 or 10 wireless terminals. Note that the operational phase for SMPT includes nine wireless terminals with a buffer size of 100 LPDUs. For nine terminals the average jitter J is always below 10 ms and the goodput reaches 66 kb/s for large buffers. For 10 wireless terminals supported with SMPT the jitter is below 10 ms for small buffers (30 LPDUs or less); for larger buffers the jitter increases dramatically. The good news from this experiment is that within its operational range of up to nine ongoing video streams, SMPT is relatively insensitive to buffer size (as long as the buffer has a certain minimum size, 40 LPDUs in our setting); this simplifies the configuration of the mechanism in practice.

As noted above, real-time video transmission requires that the video frames be delivered within a tight delay bound. To achieve a tight delay bound the buffers should be small. In Figs. 7 and 8 we plot the probability masses of the jitter J (in ms) as a function of the video frame size (in bytes). The link layer buffer is set to L_{Queue} = 40 LPDUs. There are nine wireless terminals sending one video stream each in the cell. Only the jitter values of successful video frames (i.e., video frames that had none of their LPDUs dropped due to a full link layer buffer) are considered. All figures have the bimodal frame size distribution, typical for video encoded with H.263 with rate control (see [4] for details), in common. We observe that SMPT gives significantly smaller jitter J than the sequential transmission mode. For SMPT most of the probability mass is located at jitter values less than 75 ms. On the other hand, with the sequential transmission mode, video frames are very likely to experience jitter values in the range between 200 and 400 ms.



■ Figure 6. *Average jitter* J *as a function of the link layer buffer size* L_{Queue}.

Simulation Results for Video Encoded without Rate Control

In this section we study the uncoordinated real-time transmission of video encoded without rate control (i.e., in an open loop). For these simulations we used the frame size traces of 25 videos encoded in MPEG-4 with fixed quantization parameters of 10 for I-frames, 14 for P- frames, and 18 for B-frames [4]. This open loop encoding avoids the complexity introduced by rate control. (Also, it results in a constant video quality at the encoder output, whereas the video quality at the encoder output is slightly variable when rate control is employed.) However, the traffic produced by open loop encoding is highly variable. Not only are the individual video streams highly variable in the sizes of their video frames, but also the different video streams differ significantly in their frame size statistics, for example, the video streams vary in their mean bit rate (see [4] for the exact statistical properties of the 25 video traces used). These variabilities pose a particular challenge for network transport. We demonstrate that the simple-to-deploy slow- and fast-healing SMPT mechanisms are able to transport this highly variable traffic efficiently in real time over the wireless links.

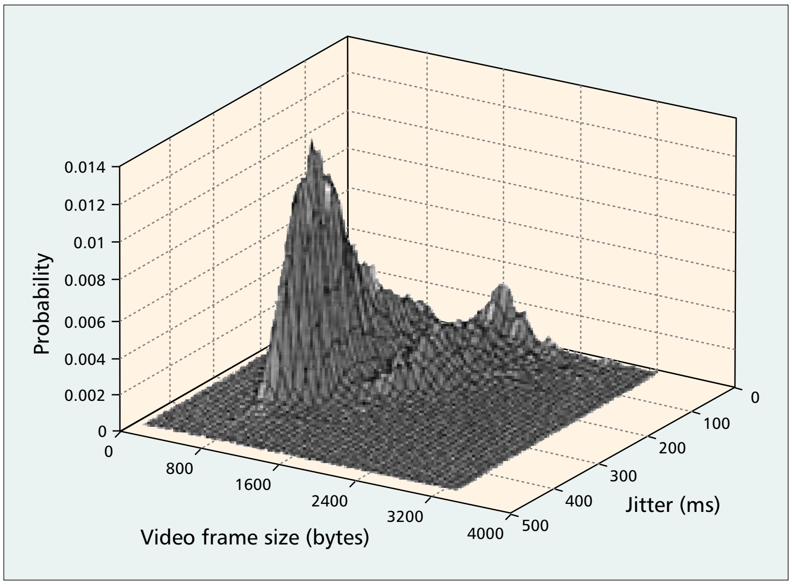
Throughout this section the spreading gain is set to a default value of 32 (whereas it was 16 throughout the preceding section). The reason for this larger spreading gain setting is that CDMA systems generally achieve better statistical multiplexing for larger spreading gains (especially for video traffic, see [11] for a detailed study). In the previous section the goal of the slow-healing SMPT transmission mechanism was to stabilize the throughput of the wireless link to the target bit rate of the rate-controlled video encodings; which have only small variations and hence require only a small amount of statistical multiplexing. For the highly variable open loop encoded video considered in this section, a significant amount of statistical multiplexing is required for efficient transport.

Impact of the Number of Wireless Terminals

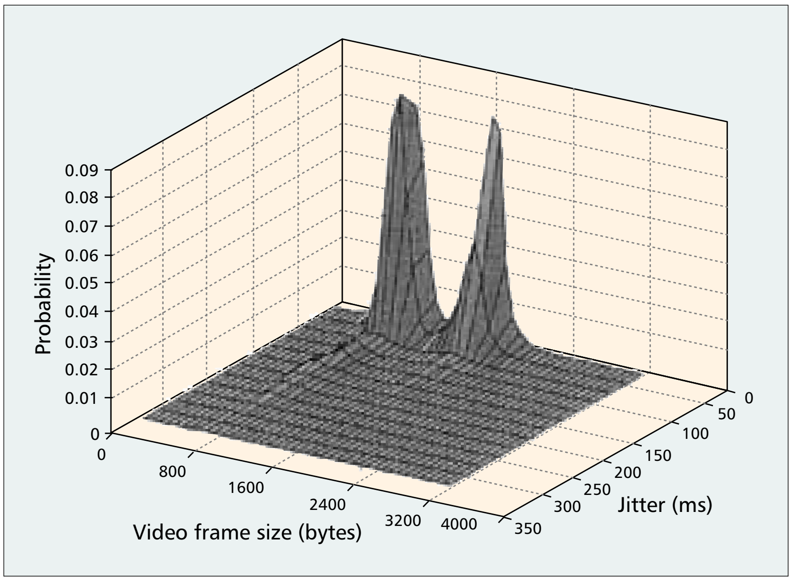
We first investigate the impact of the number of wireless terminals in the cell on the goodput and the average jitter J. We consider three transmission approaches:

- Sequential with a bit rate of 64 kb/s (i.e., a spreading gain of 32) and a bit rate of 128 kb/s (i.e., a spreading gain of 16)
- Slow-start SMPT (with up to $R = 8$ parallel code channels)
- Fast-start SMPT (with up to $R = 8$ parallel code channels).

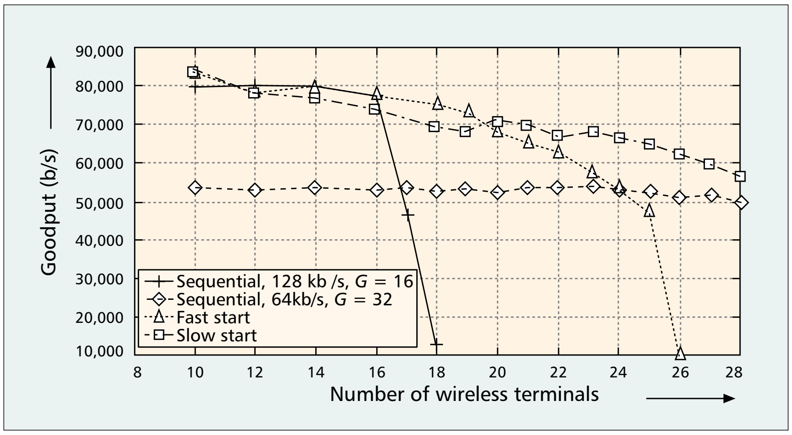
In Fig. 9 we plot the goodput as a function of the number of wireless terminals for the different transmission approaches. We observe that the sequential transmission mode with 64 kb/s has a stable goodput of 53 kb/s over the entire considered range of the number of wireless terminals (each transmitting one video stream). The sequential transmission mode with 128 kb/s, which effectively transmits always on two CDMA code channels and therefore produces significantly larger interference, achieves a higher goodput of approximately 80 kb/s up to 16 ongoing video streams. Up to 20 ongoing video streams the fast-start SMPT approach achieves the highest goodput. For more video streams the goodput drops slowly to the level of the sequential transmission mode with 64 kb/s. The slow-start SMPT approach gives a slightly smaller goodput than the fast-start SMPT approach for less than 20 video streams, but for more video streams it achieves the largest goodput. For a larger number of streams the goodput of slow-start SMPT degrades gracefully.



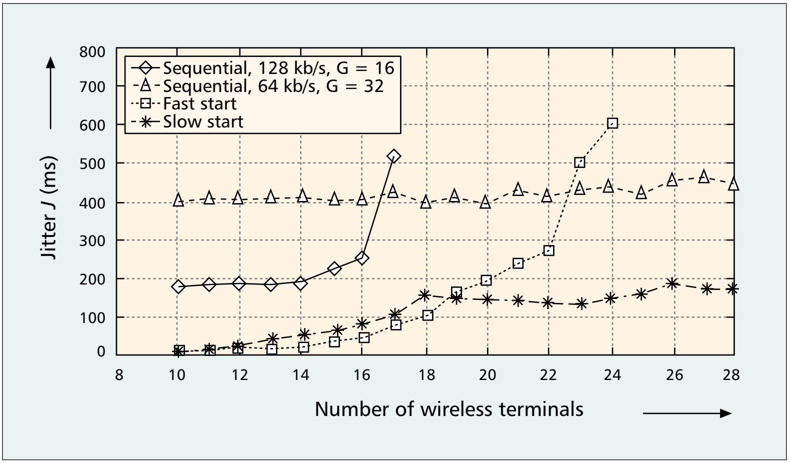
■ Figure 7. *Probability masses for video frame jitter* J *as a function of the video frame size. (Sequential transmission,* L_{Queue} *= 40 LPDUs, 9 video streams).*



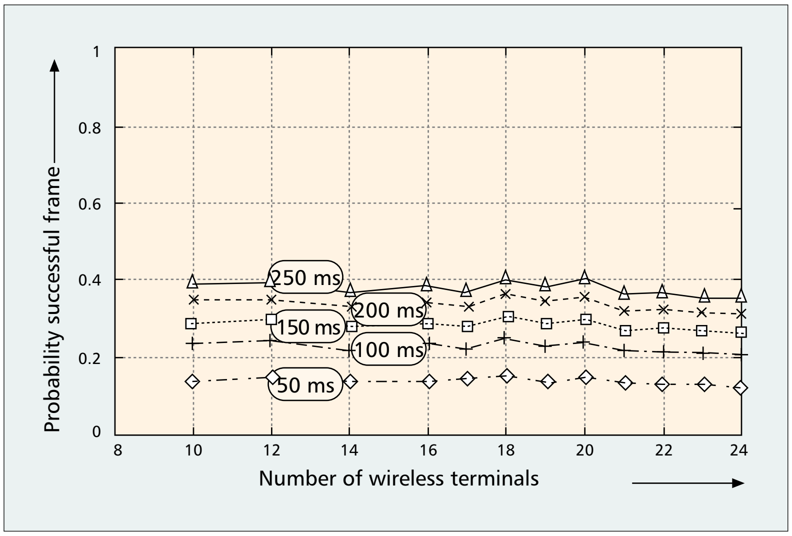
■ Figure 8. *Probability masses for video frame jitter* J *as a function of the video frame size. (slow-healing SMPT,* L_{Queue} *= 40 LPDUs, 9 video streams).*



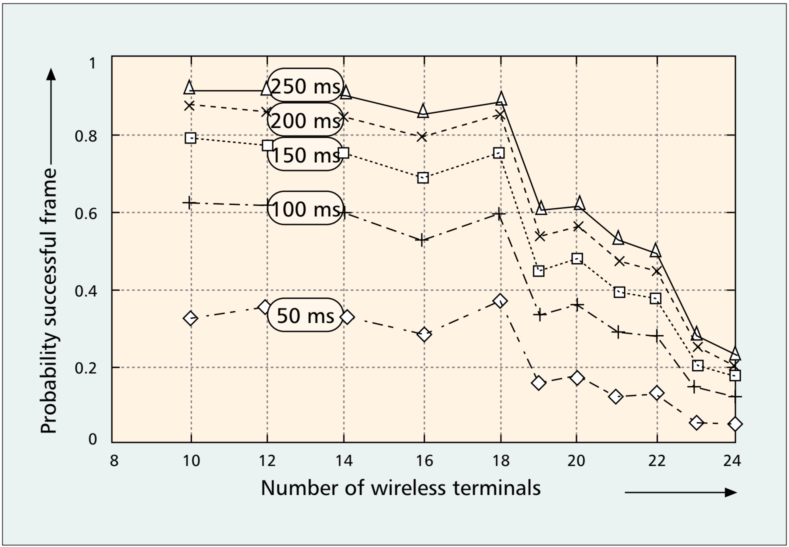
■ Figure 9. *Goodput as a function of the number of wireless terminals (*L_{Queue} *= 100 LPDUs).*



■ Figure 10. *Average jitter* J *as a function of the number of wireless terminals (*L_{Queue} *= 100 LPDUs).*



■ Figure 11. *Probability of successfully delivering a video frame with delay constraints of* τ_{delay} *= 50, 100, 150, 200, and 250 ms for the sequential transmission mode (*L_{Queue} *= 100 LPDUs).*



■ Figure 12. *Probability of successfully delivering a video frame with delay constraints of* τ_{delay} *= 50, 100, 150, 200, and 250 ms for the fast-start SMPT mechanism (*L_{Queue} *= 100 LPDUs).*

In Fig. 10 we plot the average jitter J as a function of the number of wireless terminals. Sequential transmission with 64 kb/s gives an average jitter J of 400 ms. This jitter is too large for real-time video transmission, considering that for videoconferencing the jitter constraint is typically τ_{jitter} = 150 ms. By doubling the bit rate a smaller jitter (which is still above the threshold for videoconferencing) is achieved for up to 16 ongoing video streams. Only the SMPT mechanisms achieve jitter values that are acceptable for real-time communication. For up to 18 wireless terminals in the cell the fast-start SMPT mechanism gives a slightly smaller jitter than the slow-start SMPT mechanism. With a larger number of ongoing video streams in the cell, the fast-start SMPT mechanism becomes unstable. This is because the fast-start SMPT mechanism always uses all R CDMA code channels without taking the interference level in the cell (governed by the other terminals' activities) into consideration. The slow-start SMPT mechanism, on the other hand, gives an average jitter below 150 ms for up to 24 ongoing video streams. The slow-start SMPT mechanism gives stable performance as the cell load increases since it probes out the capacity in the cell by slowly increasing the number of used CDMA codes.

Impact of Delay Bound τ_{DELAY}

In Figs. 11–13 we plot the probability of successfully delivering a video frame as a function of the number of wireless terminals for the sequential transmission with 64 kb/s, the fast-start SMPT mechanism, and the slow-start SMPT mechanism.

We plot the probability of successful video frame delivery (i.e., the probability that a video frame is delivered to the receiver with a delay value smaller than the delay bound τ_{delay}) for τ_{delay} = 50, 100, 150, 200, and 250 ms. We note that the delay bound is not known at the sender's link layer. The sender simply tries to send the video frames as fast as possible. We observe from Fig. 11 that for the sequential transmission the probability of sending a video frame successfully is constant over the entire considered range of the number of wireless terminals in the cell. The success probability depends only on the delay constraint. For a delay constraint of τ_{delay} = 150 ms, the probability of in-time delivery of a video frame is around 30 percent. We observe from Fig. 12 that for the fast-start SMPT approach the probability of sending a video frame successfully is over 70 percent if the number of ongoing video streams is less than 19, and the delay constraint τ_{delay} is 150 ms or larger. We observe from Fig. 13 that for slow-start SMPT (in contrast to sequential transmission) the probability of successful transmission of a video frame decreases with each additional video stream. However, the success probability with slow-start SMPT is always larger than with sequential transmission.

Impact of Link Layer Buffer Size L_{QUEUE}

In Fig. 14 we plot the average jitter J as a function of the link layer buffer size L_{Queue} for 18 wireless terminals and different transmission schemes. Figure 15 gives the corresponding video frame loss probabilities. (Recall that a video frame is lost only when one of the LPDUs carrying the frame finds the link layer buffer full.) The buffer length is given in LPDUs, where each LPDU is 128 bytes (and carries 80 bytes of link layer payload). We observe that the SMPT mechanisms achieve significantly smaller average jitter values and smaller loss probabilities than the sequential transmission mode. Among the SMPT mechanisms, fast-start SMPT performs slightly better than slow-start SMPT. While for small buffers the jitter and loss probabilities do not differ significantly among the transmission approaches, for larger buffers the differences are very pronounced. Note that larger buffers result in high-cost end systems. For the sequential transmission scheme a buffer size of $L_{\text{Queue}} = 20$ LPDUs (= 2.56 kbytes) is a reasonable choice. For this buffer size J is small and the loss probability is about as small as it can be with sequential transmission. A larger buffer does not provide significant improvement for sequential transmission. For the SMPT mechanisms, on the other hand, a buffer size between 50 and 70 LPDUs (6.4–9 kbytes) gives both small J and small loss probability.

Overall, we conclude from the simulation results presented in this section that the fast- and slow-start SMPT mechanisms make efficient transmission of highly variable open loop encoded video over wireless links possible. The fast-start SMPT mechanism performs very well for a limited number of simultaneous video streams, but becomes unstable when the number of streams grows beyond a certain threshold (of 18 streams with our parameter setting). The slow-start SMPT mechanism performs almost as well as the fast-start SMPT mechanism when the number of video streams is small. In contrast to fast-start SMPT, slow-start SMPT does not become unstable; it achieves significantly higher goodput values as well as smaller jitter values and smaller loss probabilities than sequential transmission even for a large number of video streams. Thus, slow-start SMPT appears to be a good choice of link layer transmission mechanism for video encoded without rate control.

Conclusion

We have studied the uncoordinated real-time transmission of video by distributed wireless clients in a wireless cell. We have demonstrated that simple-to-deploy simultaneous MAC packet transmission (SMPT) mechanisms enable the efficient transmission of video with tight real-time constraints on the order of 150 ms, thus enabling real-time applications such as video-conferencing, games, and telemedicine. For video encoded with rate control we found that the slow-healing SMPT mechanism achieves high goodput and small video frame loss probability while supporting a large number of simultaneous video streams in the cell (i.e., giving high cell capacity). For the more bursty video traffic resulting from encoding without rate control we found that the slow- and fast-start SMPT mechanisms provide efficient transmission

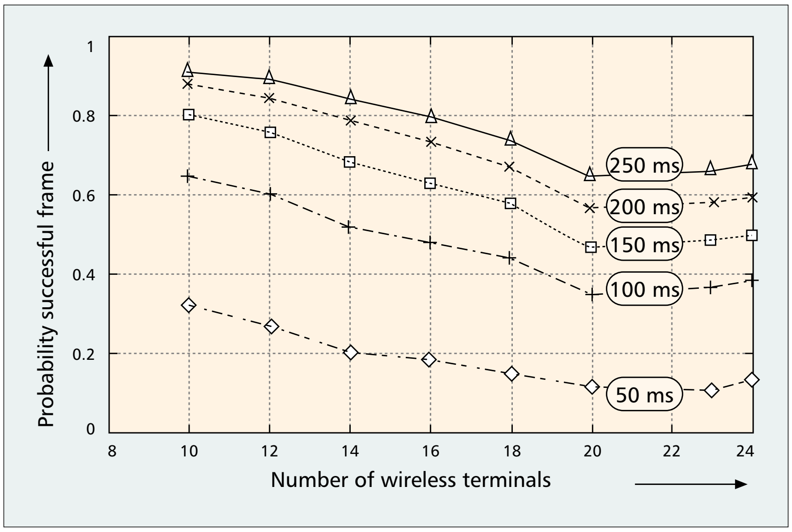


Figure 13. *Probability of successfully delivering a video frame with delay constraints of τ_{delay} = 50, 100, 150, 200, and 250 ms for the slow-start SMPT mechanism (L_{Queue} = 100 LPDUs).*

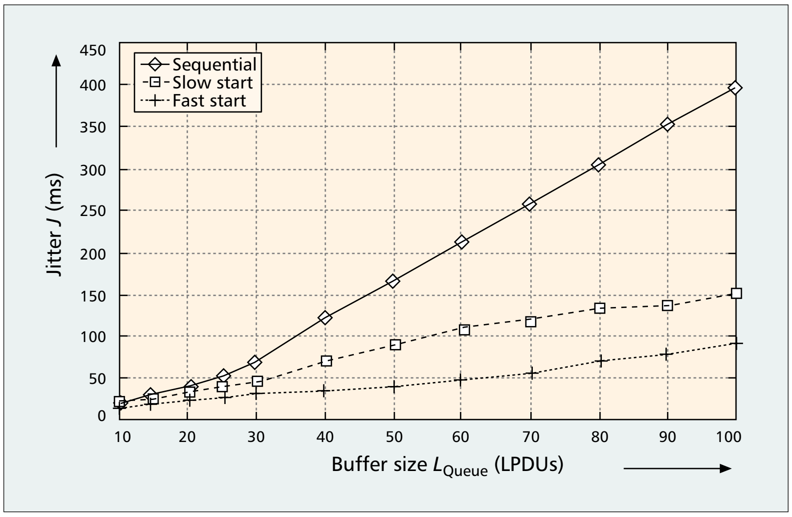


Figure 14. *Average jitter* J *as a function of the link layer buffer size* L_{Queue} *for 18 wireless terminals and different transmission schemes.*

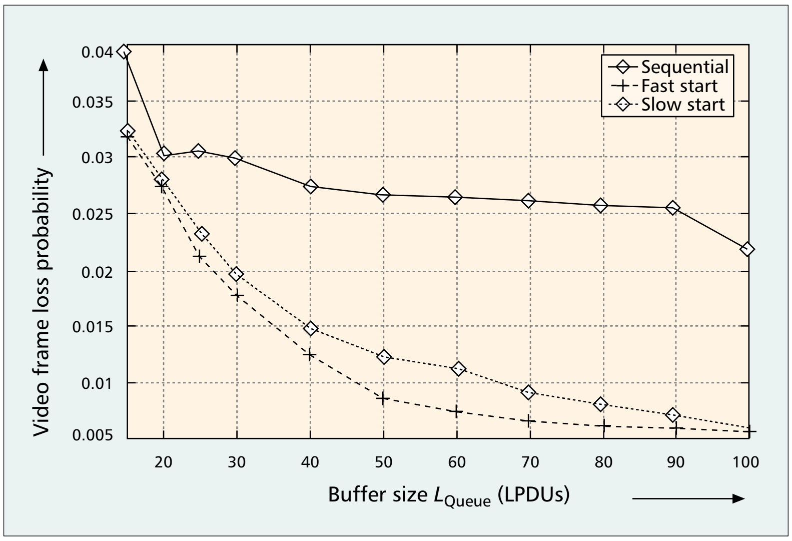


Figure 15. *Video frame loss probability as a function of the link layer buffer size* L_{Queue} *for 18 wireless terminals and different transmission schemes.*

The described SMPT mechanisms work exclusively at the link layer of the sending wireless terminal and do not require any information from higher protocol layers, thus preserving the isolation of the layers of the networking protocol stack and reducing cost and complexity.

scheduling to achieve high cell capacity and good video quality. The slow-start SMPT mechanism is particularly resilient and degrades gracefully as the load on the cell increases. We studied the impact of the link layer buffer, the only hardware component required by the SMPT mechanisms. We gave guidelines for the dimensioning of this buffer.

While we considered the transmissions from distributed wireless terminals to a central base station in a cellular wireless network in our simulations, we emphasize that the presented SMPT mechanisms can be deployed readily in ad hoc wireless networks. The SMPT mechanisms do not require any coordination of the transmissions among the distributed clients, and can thus be used for point-to-point transmissions in a cluster of a wireless ad hoc network.

We also note that throughout our focus has been on mechanisms that are low in complexity and cost and easy to deploy yet give tangible performance gains, rather than finding more complex mechanisms that further enhance performance. The described SMPT mechanisms work exclusively at the link layer of the sending wireless terminal and do not require any information from higher protocol layers, thus preserving the isolation of the layers of the networking protocol stack and reducing cost and complexity. Given the simplicity of the described mechanisms there are several avenues for future research and refinement. For instance, a refined mechanism could take advantage of the deadlines of the video frames for the scheduling at the link layer. This refined scheduling algorithm would drop a frame that will miss its playback deadline at the receiver and instead start to transmit the next video frame. (Recall that the simple mechanisms studied in this article do not assume any knowledge of the frames' deadlines and simply transmit all the LPDUs in the link layer buffer.) Note that the refinement would add complexity since its needs to obtain the frame deadlines (e.g., by parsing the RTP header). The refinement would improve the performance and the cluster (or cell) capacity by not transmitting video frames that would miss their playout deadline and thus reducing the interference level.

References

[1] F. Fitzek and M. Reisslein, "A Prefetching Protocol for Continuous Media Streaming in Wireless Environments," *IEEE JSAC*, vol. 19, no. 10, Oct. 2001, pp. 2015–28.

[2] F. Fitzek, R. Morich, and A. Wolisz, "Comparison of Multi-code Link-layer Transmission Strategies in 3Gwireless CDMA," *IEEE Commun. Mag.*, vol. 38, no. 10, Oct. 2000, pp. 58–64.

[3] F. H. P. Fitzek *et al.*, "Streaming Video Applications over TCP in CDMA Based Networks," *Proc. 3Gwireless 2002*, San Francisco, CA, May 2002, pp. 755–60.

[4] F. Fitzek and M. Reisslein, "MPEG-4 and H.263 Video Traces for Network Performance Evaluation," *IEEE Net.*, vol. 15, no. 6, Nov./Dec. 2001, pp. 40–54; video traces available at http://www.eas.asu.edu/trace

[5] H. Gharavi and S. M. Alamouti, "Multipriority Video Transmission for Third-Generation Wireless Communication Systems," *Proc. IEEE*, vol. 87, no. 10, Oct.1999, pp. 1751–63.

[6] C. Hsu, A. Ortega, and M. Khansari, Rate Control for Robust Video Transmission over Burst-error Wireless Channels, *IEEE JSAC*, vol. 17, no. 5, May 1999, pp. 756–73.

[7] H. Liu and M. El Zarki, "Performance of H.263 Video Transmission over Wireless Networks Using Hybrid ARQ," *IEEE JSAC*, vol. 15, no. 9, Dec.1997, pp. 1775–86.

[8] P.-R. Chang and C.-F. Lin, "Wireless ATM-based Multicode CDMA Transport Architecture for MPEG-2 Video Transmission," *Proc. IEEE*, vol. 87, no.10, Oct. 1999. pp. 1807–24.

[9] M. Naghshineh and M.W. LeMair, "End-to-end QoS Provisioning in Multimedia Wireless/Mobile Networks Using an Adaptive Framework," *IEEE Commun. Mag.*, vol. 35, no. 11, Nov. 1997, pp. 72–81.

[10] D. Wu, Y. T. Hou, and Y.-Q. Zhang, "Scalable Video Coding and Transport Over Broad-band Wireless Networks," *Proc. IEEE*, vol. 89, no. 1, Jan. 2001, pp. 6–20.

[11] F. H. P. Fitzek, "QoS Support for Multimedia Services in Wireless CDMA Systems," Ph.D. thesis, Telecom. Net. Group, Dept. Elec. Eng., Tech. Univ. Berlin, June 2002.

[12] D. Ferrari, "Client Requirements for Real-Time Communication Services," RFC 1193, Nov. 1990.

Biographies

FRANK FITZEK (fitzek@acticom.de) received his Dipl.-Ing. degree in electrical engineering from the University of Technology – Rheinisch-Westfälisch Technische Hochschule (RWTH), Aachen, Germany, in 1997, and his Ph.D. in electrical engineering from the Technical University Berlin, Germany, in 2002. He co-founded the start-up company acticom GmbH in Berlin in 2001. At acticom GmbH he is currently involved in the development of advanced wireless IP services for 3G. His current interests are in the areas of QoS support for voice and video over wireless networks, robust header compression, and mobile ad hoc networks.

MARTIN REISSLEIN (reisslein@asu.edu) is an assistant professor in the Department of Electrical Engineering at Arizona State University, Tempe. He received his Ph.D. in systems engineering from the University of Pennsylvania in 1998. From July 1998 through October 2000 he was a scientist with the German National Research Center for Information Technology (GMD FOKUS), Berlin, and a lecturer at the Technical University Berlin. He received the NSF Career award in 2002. He maintains an extensive library of video traces for network performance evaluation at http://www.eas.asu.edu/trace. He is associate editor-in-chief of *IEEE Communications Surveys and Tutorials* and has served on the technical program committees of IEEE INFOCOM and IEEE GLOBECOM.

ADAM WOLISZ [SM] (wolisz@ee.tu-berlin.de) obtained his Dipl.-Ing. in control engineering, Dr.-Ing., and Habilitation (both in computer engineering), respectively in 1972, 1976, and 1983, all at Silesian Technical University in Gliwice, Poland. From 1990 to 1993 he was with the Research Institute for Open Communication Systems of the German National Research Center for Computer Science (GMD-Fokus) in Berlin, heading activities on quantitative aspects of high-speed networks and multimedia systems. Since 1993 he is a chaired professor of electrical engineering and computer science (secondary assignment) at the University of Technology Berlin, where he is leading the Telecommunication Networks Group (TKN).