SAVE: An Algorithm for Smoothed Adaptive Video over Explicit Rate Networks

N.G. Duffield K. K. Ramakrishnan Amy R. Reibman
AT&T Labs–Research

Abstract— **Supporting compressed video efficiently on networks is a challenge because of its burstiness. Although a large number of applications using compressed video allow adaptive rates, it is also important to preserve quality as much as possible. We propose a smoothing and rate adaptation algorithm for compressed video, called SAVE, that is used in conjunction with explicit rate based control in the network.**

SAVE smooths the demand from the source to the network, thus helping achieve good multiplexing gains. SAVE maintains the quality of the video and ensures that the delay at the source buffer does not exceed a bound. We show that SAVE is effective by demonstrating its performance across 28 different traces (entertainment and teleconferencing videos) that use different compression algorithms.

Keywords— **Compressed Video, Rate Control, Smoothing, Multiplexing.**

I. Introduction

The desired quality of the video to be delivered to the receiver varies widely, depending on the application, the potential cost to the user, and the network infrastructure that is available for transporting the video. We believe that there is a relatively large class of applications that can tolerate some variability in the perceived quality of the video, including video teleconferencing, interactive training, low-cost information distribution such as news, and even some entertainment video. We also believe that these compressed video sources are rate adaptive, in that it is possible to modify the source rate dynamically by adjusting the video encoding parameters so as to be responsive to the conditions in the network. One parameter particularly suitable to the task of adaptation is the quantization parameter.

Network transport of video has been an active area of research, with the methods proposed for transport of compressed video spanning the spectrum of services that may be offered by the network: Constant Bit Rate (CBR), Variable Bit Rate (VBR), Available Bit Rate (ABR) and Unspecified Bit Rate (UBR). Compressed video is inherently bursty with rate fluctuations happening over multiple time scales. Since a compressed video source is bursty, the use of constant bit rate (CBR) transport, requires the use of a local buffer for smoothing [9]. Buffer overflow or underflow is prevented by continually adjusting the quantizer step size resulting in *variable quality*. The advantage of CBR transport is that it makes admission control simple. In unrestricted (or open-loop) VBR transport, the inherently bursty traffic from the coder is transported over the real-time VBR service class, thereby potentially providing greater multiplexing gain [6], [10] than with CBR.

Among the promising approaches for adapting to the short-term fluctuations in the rate required for video are Renegotiated CBR (RCBR) [9] and feedback-based congestion control [12], [13], [14], [16]. The approach described in [14] attempts to achieve the goals of increasing the multiplexing gain by frequently negotiating bandwidth between the source and the network, with the desire to be responsive to the needs of the video source, while at the same time relying on the adaptation of source rates to match available bandwidth. The scheme uses the Explicit Rate based congestion control [1], [11] mechanisms described for ATM's ABR service for rate negotiation while maintaining low queueing delays in the network.

In [14], the response of the video source to insufficient bandwidth available from the network was to reduce the source rate by modifying the quantization value. The degradation in the quality of the video was limited by requesting a minimum cell rate (MCR) for the connection. The mechanisms did not make any further attempts to manage the quality degradation. The admission control was relatively simple, based on the sum of the MCR values for all connections being less than the link capacity. Although the scheme suggested in [14] may be applicable in many situations, we feel it is important to examine in greater depth the extent of the reduction in the source quality that is imposed by altering the quantization parameter in response to the feedback received from the network. In this paper, we examine algorithms in the source that would aid in significantly reducing this quality degradation.

If an individual source is less bursty, then it is easier to support that flow by allocating a lower rate to the flow. Compressed video is bursty over multiple time scales: at the individual frame level, possibly due to the compression algorithm and its periodic nature; at the scene level, due to changing activity and detail within a given scene;

AT&T Labs–Research, Rm. A175, 180 Park Avenue, Florham Park, New Jersey 07932; duffield@research.att.com
AT&T Labs–Research, Rm. A155, 180 Park Avenue, Florham Park, New Jersey 07932; kkrama@research.att.com
AT&T Labs–Research, Rm. 3-233, 100 Schultz Drive, Red Bank, NJ 07701; amy@research.att.com

and finally between scenes, due to the different scene contents. We attempt to achieve a multiplexing gain using the explicit rate mechanisms both at the scene-level and the between-scene time scales. We expect the short-term variability between adjacent frames to be absorbed in the source adaptation buffer. We look at algorithms that smooth the traffic from an individual video source to make the demand on the network more predictable over a short time. Smoother traffic allows the rate allocation by the network to be made more accurately and easily over this short time scale.

One of the important issues is the gain from having several sources multiplexed together. The multiplexing gain is achieved as a result of the overlapping of the "peaks" and the "valleys" of the different sources of video on a link at a given time. The aggregate flow tends to be less bursty than the individual flows. The larger the number of simultaneously active flows, the higher the potential for multiplexing gain.

There has been considerable work examining the effectiveness of smoothing of stored video; see e.g., [17], [18], [19], [20], [22]. These require advance knowledge of at least a part of the future sequence of frame sizes. This results in a corresponding playback delay of a few seconds. On the other hand, our approach to smoothing is to make it suitable for a wide range of video applications including interactive ones. Our goal is to meet tight delay and quality constraints.

In this paper, we propose an adaptive source smoothing algorithm which can be used to smooth compressed video on-line. It is desirable that the smoothing algorithm be relatively insensitive to the particular video sequence: i.e., it should work well enough for the same parameter settings, for a wide-range of video sequences with different content. We call the algorithm SAVE (Smoothed Adaptive Video over Explicit rate networks). We show that SAVE maintains the quality of the video within acceptable levels and ensures that the delay introduced in the source buffer is within bounds, under a variety of conditions. We also show that there are significant multiplexing gains to be achieved using SAVE.

While a significant set of the results presented here are based on a trace-driven frame level simulation, without the network, we have also used a detailed network simulation at the cell level which have validated our results with the frame level simulation [5]. Using the simpler frame level simulation enabled us to explore the parameter space and the efficacy of our scheme over a larger number of traces more rapidly.

The next section describes the framework under which we study our algorithm, and in Section III we describe the evaluation criteria. In Section IV we describe the details of SAVE and in Section V we present detailed single-source evaluations of SAVE for a variety of traces and network conditions. Section VII describes issues related to adapting SAVE to observed quality. We examine the benefits of SAVE when multiplexing several video sources in Section VI. After a discussion on tunability of achieved quality in Section VII, we conclude.

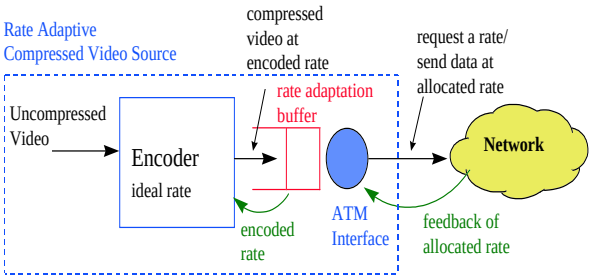

Fig. 1. FRAMEWORK: for Rate Adaptive Video in an Explicit Rate Environment

II. FRAMEWORK

Figure 1 shows the framework under which we study the effectiveness of adapting compressed video sources in a rate-controlled network. The uncompressed video from the source is typically fed to an encoder, which is capable of encoding a frame such that the maximum size of the compressed output frame does not exceed a given number of bits. Our work is applicable to coding schemes such as JPEG and MPEG. We do not consider coding schemes such as layered coding, where the different frequency components are transmitted separately, and the rate adaptation is accomplished by choosing not to carry the high frequency components.

The output of the encoder is fed to a rate-adaptation buffer at the source that is used to accommodate the variability in the demand from the encoder. It also accommodates the difference between the rate coming into the buffer and the output rate from the source into the network. The output rate of the source is controlled by the explicit-rate congestion control algorithm which specifies the rate at which the source may transmit data into the network.

The source uses the *Available-Bit-Rate* (ABR) service defined by the ATM Forum [1], using the explicit rate option. The source is allowed to request a rate from the network, and the network responds with an allocated rate based on the contention for resources in the network, with the assurance that the rate allocated will not go below a minimum rate negotiated at connection set-up time. The minimum rate that the source requests may be selected so that the quality for the video does not go below a minimum acceptable level.

It is possible to keep the queueing delays seen by connections using the ABR service fairly small, since the

explicit-rate schemes can ensure that the aggregate rate of all sources sharing a link remains below the link bandwidth. To ensure that the delays experienced by video flows are reasonably small, it may be necessary to separate those flows that are admission controlled and require low delay (e.g., video) from others that may not be admission controlled (e.g., bursty data). Concomitant with it may be a service discipline at the switches to serve the separate classes in proportion to the rates allocated to each of the classes (e.g., as in [4]).

In [14], the source buffer was limited implicitly through the use of set points to modify the quantization value. There was no strict bound placed on the amount of delay contributed by the source. The source buffer may be significant enough to be the dominant component in the total end-to-end frame delay. Understanding the delay contributed by the source and attempting to limit it, with high probability, is desirable. Thus, we avoid the frame arriving late at the receiver and effectively being lost. In this work we seek to keep the delay in the source buffer bounded, by ensuring that the encoded frame does not exceed a maximum size. When the network provides an allocation that is lower than the requested rate, two actions are possible:

- The source buffer builds up, thus increasing the delay. The rate adaptation algorithm recognizes this extra delay and implements corresponding modifications in the smoothing algorithm.
- When the rate adaptation algorithm is unable to maintain the delay within the target, it can then modify the quantization parameter to reduce the input into the buffer.

The distributed explicit rate allocation algorithm should converge to the eventual *max-min fair* allocations [2] for the individual flows within a reasonably short time. Simplistically, the time it takes to converge is approximately twice the round-trip time multiplied by the number of bottleneck rates [3]. This convergence time is based on having a stable demand from the individual sources during the period of convergence of the distributed algorithm. Our smoothing algorithm attempts to keep the short-term variability in the demand relatively small, thus helping the network's explicit rate algorithm to converge. Although it is unlikely that the demand will be constant over the several round-trip times that may be required for achieving strict max-min fairness, we believe that smoothing the demand will be helpful in achieving better "average" fairness compared to when the source rates change on a much shorter time-scale. For a detailed description of the end-system policies and switch policies that assure max-min fairness while maintaining small queues, we refer the reader to [1], [3], [11], [15].

We shall describe the operation of SAVE by means of the following data rates illustrated in Figure 1:

- Ideal Rate: This is the rate that is required by the encoder to code the frame at ideal quality.
- Encoded Rate: This is the rate given to the encoder based on the algorithm for smoothing and rate-adaptation. We assume that the encoder will precisely meet the rate it is given as the target for a frame, as long as the encoded rate is less than the ideal rate for the frame.
- Requested Rate: This is the rate that the source requests from the network based on the smoothing algorithm, the state of the source buffer, and the ideal rate for the frame.
- Allocated Rate: This is the rate returned from the network, after a feedback delay, in response to the source's requested rate.

We use the term *cropping* to describe the reduction from the ideal rate to the encoded rate. (Cropping entails a lowering in encoding detail rather than a truncation of the size of the image). In contrast, the term *rate-reduction* will be used when the allocated rate is less than the requested rate.

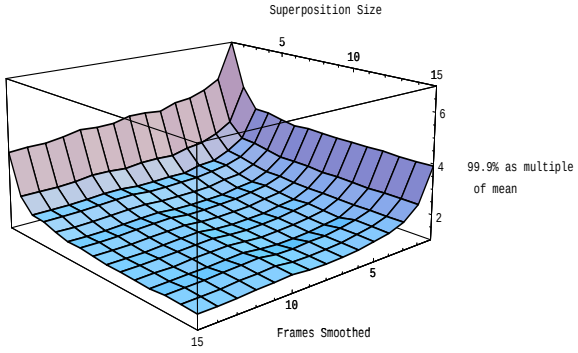

Fig. 2. EFFECT OF SPATIAL AND TEMPORAL SMOOTHING: Aggregated and smoothed MPEG-1 video. 99.9% of aggregate ideal rate, as a multiple of mean rate for fixed smoothings of 1 to 15 frames, aggregation of 1 to 15 traces.

A. Smoothing and Multiplexing Gain

Smoothing the source demand decreases the bandwidth which must be allocated to an individual flow, or allocated per flow in an aggregate. Further, the explicit rate allocation algorithms benefit from source demands that are smooth, as described above. To demonstrate the gain arising from smoothing within flows and smoothing across flows, we examine the benefits of multiplexing several MPEG-1 traces that are smoothed using a simple fixed moving window averaging, for different values of the window. We find that the multiplexing across sources is enhanced by smoothing within individual sources, as seen in Figure 2. We display the bandwidth requirement to satisfy 99.9% of the demand, for successive aggregations of up to 15 MPEG-1 traces, when smoothed over increasing smoothing windows of up to 15 frames. Greater smoothing within a source makes the predictive task of admission

control easier since smoother individual flows are less likely to have fluctuating demands in the future. However, there is less multiplexing gain that can be obtained across sources. Although the mean requested rate from the SAVE algorithm we propose in this paper is higher than the mean ideal rate as a consequence of working to a delay target, there is a dramatic reduction in the peak rate requirement for the requested rate compared to the peak ideal rate. The benefits of aggregating multiple sources is also quite dependent on the periodic structure (such as a GOP structure) of the compressed video. If the I-frames are aligned, then the multiplexing of several unsmoothed videos may not show a substantial reduction in the aggregate demand, because of the alignment of the peak requirements. Temporal smoothing reduces this effect.

III. Evaluation Criteria

In this section we describe our criteria to evaluate the performance of the overall system: delay, quality and networking performance. While the criteria for video quality are somewhat heuristic, they are based on known characteristics of how humans subjectively rate the quality of video [8].

Source Delay. It is important that the source buffer not introduce delay so large that it eats into the delay budget of the network; excessive delays make the network less attractive for real-time services. We assume that there is a sufficiently large playout buffer at the receiver to overcome delay jitter; the primary concern for our work is the aggregate delay introduced in the source buffer and the network. We assume that the source buffer can be large enough to accommodate some amount of variability in the frame sizes and also, to a limited extent, differences between the encoded rate and the network's allocated rate. We assume that an overall delay budget around 200 milliseconds to 300 milliseconds is acceptable, and that, of this, a delay target of about 100 milliseconds for the source buffer is reasonable, for interactive applications. To evaluate our ability to meet these delay considerations, we will look at the probability that frames exceed the source buffer delay budget. We desire to keep this probability as small as possible.

Quality and Adaptation. It is desirable to keep the quality of the video transmitted by the source as close as possible to the ideal quality. However, our premise is that sources are adaptive enough that when a lower value than the ideal rate is infrequently imposed on the encoded rate, the video quality at the receiver does not suffer a significant perceptual impairment. When cropping does occur, we attempt to do so as gracefully as possible. This means that not only should the amount of cropping be small, but should also not occur frequently. Moreover, when cropping does occur, the number of consecutive frames suffering cropping should be relatively small.

In evaluating the operation of SAVE, we look at the pattern of cropping over the entire sequence. We strive to keep cropping below 20%, only exceeding this level for at most 0.1% of all the frames. At the rates in question, this amount of cropping will degrade video quality of a single frame, as measured by the Peak Signal to Noise Ratio (PSNR), by about 1-2 dB. We believe that greater than this amount of degradation is generally perceivable by moderately experienced viewers.

We also look at the pattern of cropping as a function of time. We look at the number of consecutive frames with a rate reduced by more than a given threshold. For example, we look at the distribution of the number of consecutive frames which suffer a cropping of 20% or more.

Channel Capacity. Finally, we look at the number of sources that may be multiplexed within a link of a given capacity when our delay constraints and rate reduction criteria are met. This is eventually the criterion that will guide us to choose one algorithm over another, because the network will be able to support a larger number of sources without any significant quality degradation.

IV. The SAVE Algorithm

The SAVE algorithm adapts to the demands of the video source, but imposes controls just when the uncontrolled demand of the sources would lead to excessive delay. The algorithm comprises two parts. The *rate request algorithm* specifies how the source requests bandwidth from the system. The *frame quantization algorithm* specifies how the frame sizes are controlled in order to avoid excessive delay. We now motivate these two parts by examining the characteristics of the source video.

A. Heuristics for the Requested Rate

We estimate the rate required of the network as the maximum of two components: one reflecting the short term average rate requirements, and the other based on the typical large frame at the scene time-scale.

Our work is based on compressed video that uses compression algorithms such as MPEG. In this case there may be a number of relatively large frames that have been intra-frame coded and occur periodically. These may be interspersed with inter-frame coded frames that are typically smaller; these carry motion information and temporal changes from the previous frame. With MPEG, the I-frames are typically large frames. A Group of Pictures (GOP) typically consists of one I frame followed by a series of other (not–I) frames, although this need not necessarily be the case.

It is desirable to have a network rate which is smoothed over the GOP period in order to avoid systematic fluctuations within it, otherwise poor performance can result. Without smoothing, the allocation of rate could systematically lag demand, and the mismatch of a large alloca-

tion with a small demand leads to wastage of network resources. So one determinant of our required rate will be the smoothed rate $r_{\rm sm} = f_{\rm sm}/\tau$ where $f_{\rm sm}$ is the ideal frame size, smoothed over some window of $w_{\rm sm}$ frames, and τ is the inter-frame time. For encodings without a periodic structure we could in principle take $w_{\rm sm}$ as small as 1.

While $r_{\rm sm}$ gives the average requirement over a small number $w_{\rm sm}$ of frames, the larger frames in the smoothed set will suffer increased delay. These large frames are potentially the I-frames in MPEG-2, the initial frames of a new scene, or those frames generated when there is considerable activity in the scene. We aim to allocate a rate in order that the frame delay does not exceed some target $\tau_{\rm max}$. So we keep track of the maximum frame size $f_{\rm max}$ at the time-scale of scenes of a few seconds, i.e., over some window $w_{\rm max}$ which may be substantially larger than the smoothing window $w_{\rm sm}$. There are evidently correlations in the frame sequences, especially over intervals of a few seconds (e.g., at the scene time-scale). By keeping track of the peak sizes over this time-scale and ensuring that our rate is adequate to meet their requirements, we can be responsive to the requirements of the scene. Also, we can be responsive to the needs of any periodic, relatively larger I-frames in an I-B-P frame structure for MPEG compression.

However, just as correlations decay over larger time-scales, the use of the finite window $w_{\rm max}$ means that the changes in activity at the scene level are reflected in $f_{\rm max}$. Nonetheless, we do find it useful to keep a historical estimate of $f_{\rm max}$ through autoregression. The rate we associate with our estimate $f_{\rm max}$ of the maximum frame size is $r_{\rm max} = f_{\rm max}/\tau_{\rm max}$, i.e., the rate required to drain the frame of size $f_{\rm max}$ from an empty buffer within the delay target $\tau_{\rm max}$. The requested rate in SAVE is at least the maximum of ($r_{\rm sm}$, $r_{\rm max}$). In practice we find that the ratio $f_{\rm max}/f_{\rm sm}$ is usually larger than τ_{max}/τ. Because the large frames are comparatively isolated, the source buffer will usually be relatively empty immediately before the large frame enters it if the requested rate is allocated.

The requested rate has a safety factor built into it. We ask for a *small* over-allocation of the rate, over and above the rate that was computed above. This allows us to drain the buffer, especially when it has been built up. This is in keeping with the intuition that the mean rate requested is likely to be slightly higher than the mean of the source's ideal rate, just so that we can ride out peaks in the source's ideal rate. We also assume that there is an initial rate allocated to the source, to drain the first few frames, until the feedback arrives from the network in response to the request generated when the first frame is being encoded and transmitted.

B. Heuristics for Frame Quantization

In parallel with the rate above, we also keep track of available size $f_{\rm avail}$ for a frame to be drained within the delay target $\tau_{\rm max}$, given the current rate allocation and buffer occupancy. This size is supplied to the encoder. If the ideal frame size exceeds the available size $f_{\rm avail}$, the quantization level in the encoder is adjusted in order that the encoded frame size is reduced to meet $f_{\rm avail}$ if possible, but never by more than some fixed factor. Clearly it is desirable that the frequency and amount of cropping are not excessive. One of the consequences of tracking the maximum frame size on a scene time-scale is that it is typically only the first frame of a new scene which is vulnerable to cropping. We now specify the two parts of the SAVE algorithm precisely.

C. Specification of the Rate Request Algorithm

First we collect together the notation for the ideal frame sizes and parameters of the *Rate Request Algorithm*. $f(n)$ is the ideal size of $n^{\rm th}$ frame; $w_{\rm sm}$ is the size of the smoothing window in number of frames; $w_{\rm max}$ is the window for the local maximum $r_{\rm max}$ expressed as a number of frames; $\tau_{\rm max}$ is the delay target; τ is the inter-frame time; α is an autoregressive factor in $[0, 1]$ for the local maximum frame size; $\beta \geq 1$ is the factor for systematic over-demand of rate request. These are used to construct three rates as follows:

$r_{\rm sm}$: the smoothed ideal rate of the $n^{\rm th}$ frame

$$r_{\rm sm}(n) = (\tau w_{\rm sm})^{-1} \sum_{i=0}^{w_{\rm sm}-1} f(n-i), \qquad (1)$$

with the convention $f(n) = 0$ for $n \leq 0$.

$r_{\rm max}$: The local maximum rate for the ideal frame

$$r_{\rm max}(n) = \tau_{\rm max}^{-1} \max_{i=0}^{w_{\rm max}-1} f(n-i). \qquad (2)$$

$r_{\rm ar}$: An autoregressive estimate of the historical local maximum rate. This is defined by mixing in the history of $r_{\rm max}(n)$ only for those frames at which it changes. Let $n_i, i \in \mathbf{N}$ denote the embedded sequence of frames at which $r_{\rm max}(n)$ changes, i.e., $r_{\rm max}(n_i) \neq r_{\rm max}(n_i - 1)$ for the n_i but for no other n. Set $R_{\rm max}(i) = r_{\rm max}(n_i)$ and define the autoregressive sequence $R_{\rm ar}$ by:
$R_{\rm ar}(0) = 0$ and

$$R_{\rm ar}(i) = \alpha R_{\rm ar}(i-1) + (1-\alpha) R_{\rm max}(i), \qquad (3)$$

for $i \in \mathbf{N}$. Then our historical estimate of the local maximum frame rate is

$$r_{\rm ar}(n) = R_{\rm ar}(i(n)), \qquad (4)$$

where $i(n) = \max\{i : n_i \leq n\}$ is the index of the last frame at which the local maximum changed.

The rate requested of the network at the time of the n^{th} frame is then

$$r_{\text{req}}(n) = \beta \max\{r_{\text{sm}}(n), r_{\max}(n), r_{\text{ar}}(n)\}. \quad (5)$$

D. Specification of the Frame Quantization Algorithm

The Frame Quantization Algorithm has the following data and parameters. We work at the time-granularity of a frame duration. We consider the n^{th} frame to be encoded during the $(n-1)^{\text{st}}$ frame time, and placed in the buffer at the start of the n^{th} frame time. We introduce some additional notation as follows: $f(n)$ is the ideal size of n^{th} frame; $b(n)$ is the buffer level after adding the n^{th} encoded frame; $r_{\text{all}}(n)$ is the rate allocated during the n^{th} frame time; γ is the minimum proportion of ideal frame size to be encoded. From these, we derive two sequences of frame times:

$f_{\text{avail}} : f_{\text{avail}}(n)$ is the estimated available size for a frame conforming to the delay bound $\tau_{\max}$. This is contingent upon the source's smoothing buffer occupancy $b(n)$ after the n^{th} encoded frame has been added to the buffer; the most recently allocated rate from the network is used to estimate future rates:

$$f_{\text{avail}}(n) = \tau_{\max} r_{\text{all}}(n-1) - \max\{0, b(n) - \tau r_{\text{all}}(n-1)\}. \quad (6)$$

$f_{\text{enc}} : f_{\text{enc}}(n)$ is the actual size of the n^{th} frame after encoding, based on maximum frame size calculated at the start of its encoding period:

$$f_{\text{enc}}(n) = \min\{f(n), \max\{f_{\text{avail}}(n-1), \gamma f(n)\}\}. \quad (7)$$

Note then that the encoded frame size is based on the rate allocated two frames previously.

E. Rate Allocation in a Frame Level Simulation

We simulate the allocated rate as a delayed version of the requested rate multiplied by a noise process in order to simulate partial satisfaction of rate requests by the network. The delay represents the time taken for the network to respond to rate requests. Until the first such response, we assume that a fixed rate is allocated by the network. We regard this as being allocated by the network at connection setup time, possibly as a function of parameters supplied by the source.

The noise process in the allocated rate arises as follows. We assume an admission control scheme which guarantees that over-subscription of the link capacity by the aggregate requested demand is sufficiently rare and short-lived. We assume that the rate allocation mechanisms in the network act by allocating a rate to each source in proportion to its requested rate, so that the total allocated rate over all sources is no greater than the link capacity [14]. We can investigate the frequency, duration and magnitude of such events through simulations of the aggregate requested rate, and in particular its crossing of the link capacity. Figure 8 is typical; rare excursions above a level may be bursty. For this reason, for simulations, we model the dependence of the allocated rate on the requested rate by using a two-level model with the following parameters. ρ is the proportion of rate request allocated during congestion; $p(n)$ is a Markov chain on the state space $\{\rho, 1\}$; T_1 is the mean lifetime of state $\{1\}$; T_ρ is the mean lifetime of state $\{\rho\}$; δ is the network feedback delay measured in frame times; r_0 is the initial rate supplied by network.

With these definitions, the allocated rate is then simulated by

$$r_{\text{all}}(n) = \begin{cases} p(n) r_{\text{req}}(n-\delta) & n > \delta \\ r_0 & n \le \delta \end{cases} \quad (8)$$

The buffer evolution is

$$b(n) = f_{\text{enc}}(n) + \max\{0, b(n-1) - r_{\text{all}}(n-1)\tau\}. \quad (9)$$

V. Analysis Results

In this section we describe results of analysis and a frame-level simulation of the SAVE smoothing algorithm with a variety of different video traces and encodings.

A. The Experimental Traces

The frame-size traces we have used fall into 5 sets. Table I describes the individual traces in more detail.

A. An MPEG-2 encoding of a 40680 frame portion of "The Blues Brothers", with M=1. There is no periodic structure. The frame rate was 24 frames per second.

B. A 174138 frame MPEG-1 trace of "Starwars" movie [7]. The GOP is 12 frames with an IBBPBBPBBPBB pattern. Frame rate was 24 frames per second.

C. H.261 encodings of 5 video-teleconferences, of either 7500 or 9000 frames. Each has an initial I-frame, followed by P frames only. Frame rate was 30 fps.

D. H.261 encodings of 2 video teleconference traces. No periodic structure. Frame rate was 25 fps.

E. 19 MPEG-1 traces, each with 40,000 frames, compiled by Rose. They originate from cable transmissions of films and television; see [21] for further details. The GOP is 12 frames with an IBBPBBPBBPBB pattern. For our experiments we assumed a uniform rate of 24 fps.

B. Source Rate Behavior

First, we look at the dynamic behavior of the smoothed rates produced by SAVE to show its benefits, in terms of both reducing the magnitude (because the buffer can drain out large frames over a period greater than a frame time but within the delay bound) and the variability (because the smoothing algorithm tries to find a piecewise average rate for the request) of the *requested rate* in comparison

set	name	type	length frames	rate fps	mean bits/fr.	peak bits/fr.
A	Blues Brothers	ent	40680	24	54823	339304
B	StarWars	ent	174138	24	15598	185267
C	C1	vtc	9000	30	10495	99250
	C2	vtc	9000	30	28330	112549
	C3	vtc	9000	30	10736	78464
	C4	vtc	9000	30	11704	78004
	C5	vtc	7500	30	7214	88141
D	D1	vtc	45000	25	66202	322048
	D2	vtc	38137	25	168720	539616
E	mrbean	ent	40000	24	17647	229072
	asterix	ent	40000	24	22349	147376
	atp	ent	40000	24	21890	190856
	bond	ent	40000	24	24308	244592
	dino	ent	40000	24	13078	119632
	fuss	ent	40000	24	27129	187176
	lambs	ent	40000	24	7312	134224
	movie2	ent	40000	24	14288	172672
	mtv	ent	40000	24	24604	229200
	mtv2	ent	40000	24	19780	251408
	news2	ent	40000	24	15358	189888
	race	ent	40000	24	30749	202416
	sbowl	ent	40000	24	23506	140840
	simpsons	ent	40000	24	18576	240376
	soccer	ent	40000	24	25110	190296
	star2	ent	40000	24	9313	124816
	talk2	ent	40000	24	17915	132752
	talk	ent	40000	24	14537	106768
	term	ent	40000	24	10905	79560

TABLE I

TRACE PROPERTIES: vtc=video-teleconference, ent=entertainment: movie or television. Trace Sets B and E are MPEG1; Trace Set A is MPEG 2; Trace Sets C and D are H.261

to the *ideal rate* at the video source. We also demonstrate, using the differences between the ideal rate and the *encoded rate*, that reductions in the quality of the video—due to frame cropping (especially over 20%) in order to meet the delay bound—can be made extremely rare.

In all of our analysis in this section, we use a $w_{\max}$ of 1000 frames, and a systematic over-demand of the rate request, β = 1.05 (i.e., a 5% over allocation). The minimum proportion of the ideal frame size to be encoded (maximum cropping level, γ) was 0.5. The initial rate request is chosen to be approximately the long-term mean ideal rate, and the target for the delay at the source buffer was chosen to be 90 milliseconds. For traces A and B, the smoothing window was 12 frames (i.e., $w_{\rm sm} = 12$). For the video teleconferencing traces, $w_{\rm sm} = 1$.

We initially examine the behavior of our algorithm using trace A. Figure 3 shows the three rates for the trace A over a short interval of 2000 frames to show the detailed behavior. The ideal rate ranges from roughly 50000 bits to 350000 bits per frame. The encoded rate overlaps with, and is only infrequently less than, the ideal rate. There are 5 occurrences where the encoded rate is less than the ideal rate.

The requested rate is relatively flat, with a small number of steps for the 2000 frame sequence. The requested rate does not change substantially at the shorter-lived peaks of the ideal rate. This is because these peaks are accommodated in the buffer without exceeding the delay target. But a burst of large frames, such as the burst occurring around frame 35000, increases the local smoothed rate $r_{\rm sm}$, and hence the requested rate, in order to accommodate the burst. The first large frame in this burst which would violate the delay bound causes an increase in the maximum rate $r_{\max}$ which will persist for $w_{\max} = 1000$ frames. So although the ideal rate drops down after the burst, we keep the requested rate high in anticipation of further large frames from the current scene (i.e., based on the expectation of high short-term correlation). If the inactivity period were to last longer than $w_{\max}$, then we would see the requested rate drop down. In this example, we see yet another burst around frame number 35500 with several large frames. This causes a further increase in the requested rate to meet the requirements.

A guideline for quality we used was that no more than 0.1% of the frames should suffer more than 20% cropping. In our experiments only a proportion of 0.001057 of the total frames suffered more than 20% cropping. In fact, only a proportion of 0.003220 suffered any cropping at all, over the 40680 frames.

Tables II and III present summary statistics for traces A and B respectively, with the smoothing. For trace A, the mean ideal rate is about 54823 bits (per frame), the mean encoded rate 54755 bits. The mean requested rate is 100983 bits, a factor of 1.8 over the mean ideal rate. Inspection of the quantiles of the frame sizes show the benefit of SAVE. The 99.0^{th} percentile for the requested rate is slightly higher than for the ideal rate. However, this is the largest value for the rate request, while the quantiles of the ideal rate increase, reaching a maximum (the $100^{\rm th}$ percentile) which is over twice the maximum requested rate. Notice also that the encoded rate is only slightly less than the ideal rate, implying that the quality degradation is small. Table II also shows that the delay is well-behaved, with the maximum delay introduced by the source buffer within the target of 90 milliseconds that we chose for this run.

Because the requested rate is so much smaller than the peaks of the ideal rate from the source, higher multiplexing gains are likely to be achieved. For example, the requested rate has a peak near 150000 bits per frame in comparison to the ideal rate's peak near 300000 bits per frame. We investigate the multiplexing properties of the requested rate in more detail in Section VI.

We also examined the performance of SAVE with the StarWars movie, trace B. The quality is even better, with only 8.610^{-5} of the frames suffering more than 20% cropping, and only 0.00092 of the frames suffering any cropping at all.

%-ile	Ideal	Encoded	Requested	Src. Delay
50.0	48888.00	48888.00	98675.00	21.0
90.0	86496.0	86393.60	134534.00	42.0
95.0	100616.80	100216.80	143564.05	49.0
99.0	140059.76	138501.48	164653.00	72.0
99.5	171801.20	170317.44	164653.00	79.0
99.9	236743.51	231572.49	164653.00	88.0
99.95	247337.04	244964.27	164653.00	89.66
99.99	290190.42	286291.59	164653.00	90.0
100.0	339304.00	339304.00	164653.00	90.0
mean	54823.41	54755.78	100983.68	24.45

TABLE II

STATISTICS OF DATA RATES AND DELAY: Trace A; Ideal, Encoded and Requested Rates (bits/frame) and Source Delay (milliseconds).

%-ile	Ideal	Encoded	Requested	Src. Delay
50.0	8069.0	8069.0	43638.0	9.0
90.0	40369.5	40364.5	55661.0	42.0
95.0	58637.0	58619.0	62192.0	57.0
99.0	84380.7	84243.4	84906.0	80.0
99.5	92758.3	92388.5	89898.0	84.0
99.9	111790.1	111457.2	89898.0	87.0
99.95	127631.1	127060.5	89898.0	87.0
99.99	161954.7	159132.9	89898.0	88.0
100.0	185267.0	185267.0	89898.0	93.0
mean	15598.2	15591.8	44809.7	16.5

TABLE III

STATISTICS OF DATA RATES AND DELAY: Trace B; Ideal, Encoded and Requested Rates (bits/frame) and Source Delay (milliseconds).

C. Behavior of Delay in Source Buffer

Figure 4 shows the behavior of the delay introduced per frame at the source buffer in milliseconds for the 2000 frame sequence. We have a target of 90 milliseconds for this delay, beyond which we crop the frame. The delay behavior mimics the pattern of behavior for the rates. However, the larger delays occur not just at the peaks for the rates, but also at the transitions from a low activity to a higher activity sequence. For example, around frame 34250, the rate change is relatively small. But, the delay observed for the frame is large. This is because the requested rate was tracking a lower rate, which causes a buffer buildup for this new, only slightly larger frame. An increase in the requested rate tracks this new level of activity, which then brings the buffer occupancy down. The primary observation to make is that the buffer occupancy, reflected by the delay, is kept low in periods of lower activity, so that when a new large frame shows up, or when there is a scene change, there is adequate free buffer to accommodate the large frames.

D. Degradation in Quality due to Frame Cropping

A concern with this form of adaptation of compressed video is the degree to which we compromise on quality. A quantitative value we have chosen for acceptability is that no more than 0.1% of the frames should suffer more than 20% cropping. The extent of degradation in quality for trace A, with varying feedback delays from the network, are shown in Table IV. The target for the proportion of frames not suffering more than 20% cropping is clearly met. In fact, only 0.3% to 0.4% of the frames suffer any cropping at all.

As seen in the table, the degradation in quality shows a slight sensitivity to the feedback delay. Even with a feedback delay of 4 frame times (about 166 milliseconds) the maximum source buffer delay was 105 ms (not shown). Also shown in Table IV is a more detailed measure of quality: the number of consecutive frames that suffer more than a threshold (20%) of cropping. (We call such frames "failures"). We expect that lower the number of consecutive failures, the better the quality. In fact, to be conservative, when the number of consecutive successes (i.e., frames with less than 20% cropping) is less than a GOP (12 frames from Trace A), we amalgamate these into the previous string of failures. We also identify the string of frames that are successes (i.e., no frame in a string of consecutive frames suffer cropping greater than the threshold of 20%). In the table we see that the mean value for the number of consecutive failures ranges from 3.25 to 4, for varying feedback delays. In contrast, the mean value for the number of successful frames is around 2000. When the feedback delay is 4 frames, the longest string of failures was 8 frames, for the entire trace of 40680 frames. Furthermore, the total number of consecutive-string failures was only 26 and total number of frames suffering more than 20% cropping was 67.

E. Modeling Rate Reduction due to Contention

When the aggregated requested rate exceeds capacity at a link, the network will reduce the rate allocated to each source in proportion to its request. The proportion of rate reduction is such that the total allocated rate is equal to the link capacity We model the effects of such contention on a single source by a two-state (ON-OFF) Markov process,

Feedback	Cropping			Mean	Mean
Delay	$> 0\%$	$> 20\%$	$= 50\%$	# Success	# Fail
1 frame	0.0032	0.0011	7.3e-5	2256.7	3.28
2 frames	0.0036	0.0012	9.8e-5	2030.7	3.25
3 frames	0.0037	0.0014	1.2e-4	2030.1	3.85
4 frames	0.0042	0.0016	1.5e-4	1933.1	4.00

TABLE IV

CROPPING STATISTICS IN FRAME-LEVEL SIMULATION. Trace A, for network feedback delay of 1 to 4 frames (= 42 to 166 ms). Maximum cropping is $\gamma = 50\%$. Also shown are mean number of successive frames having less than 20% cropping, and mean number of successive frames failing the 20% cropping criterion.

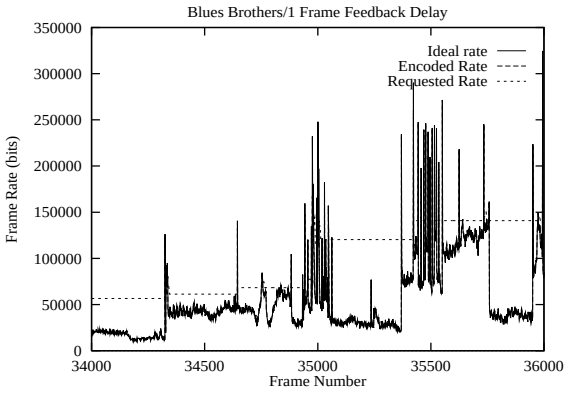

Fig. 3. DATA RATES: Ideal, Encoded and Requested Rate for Trace A, feedback delay=1 frame time.

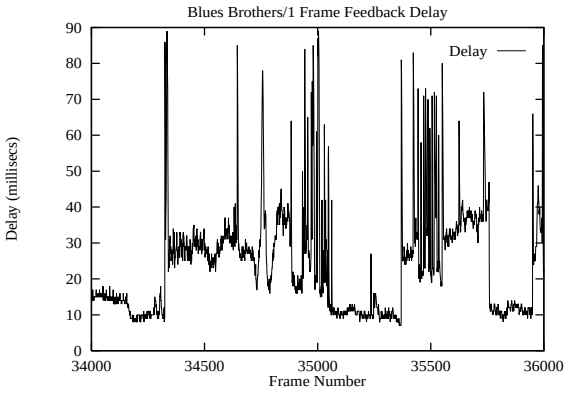

Fig. 4. SOURCE DELAY: Trace A, feedback delay=1 frame time.

as described in Section IV. During the ON-period, the allocated rate is the requested rate. During the OFF period the allocated rate is a proportion ρ of the requested rate.

In experiments we used OFF periods of mean duration $T_\rho = 50$ and ON periods with a mean duration of $T_1 = 300$. These appear fairly conservative when compared with the durations of periods of contention observed in the multiplexing experiments that we report in Section VI. Figure 5 shows the degradation as a function of ρ. The upper curve shows that the proportion of frames suffering any cropping at all increases nearly linearly as ρ is reduced, going from 0.4% when ρ is 0.95 to 9% when ρ is 0.5. But, the amount of degradation due to cropping by greater than 20% is less sensitive to ρ as long as ρ is greater than 0.9. The source buffer is able to absorb the difference in the requested rate and the allocated rate. However, when the network severely reduces the rate allocated (i.e., ρ is quite a bit less than 1), then the source buffer is no longer able to ride out these congestion events, and the video suffers the inevitable quality degradation.

Similarly, we found that even the higher percentiles (99.5^{th}) of the delay remain below 90 milliseconds when ρ varies from 1 down to 0.65. The delay in the source smoothing buffer, with the ON-OFF model for congestion episodes (shown in Figure 6), is not quite as sensitive (in our estimation) to reductions in ρ (in the range of 0.95 to 0.65) because the frames are being cropped in order to meet the delay target. However, the minimum allowed cropping γ (described in Section IV-D) was set to 0.5. As ρ approaches this value, progressively more frames that do not meet the delay target are placed in the buffer. We do not include detailed graphs, due to lack of space. The higher percentiles of delay increase somewhat more as ρ drops from 0.95, the 99.95^{th} percentile being 100.66 milliseconds at ρ =0.64, which is just around our target.

In conclusion, it is relatively important that we perform a sufficiently conservative admission control policy that results in ρ being above about 0.85. More substantial rate reductions result in penalizing the quality of the video. However, these observations are all based on a single trace. Subsequent sections will address the benefits of multiplexing several sources, and the impact of congestion causing the allocated rate to be smaller than the requested rate.

Trace	Mean rate			Prop Cropped $\geq$ 20%
	Ideal	Encoded	Requested	
A	54823	54755	100983	0.001057
B	15598	15591	44809	8.614e-05
C1	10495	10483	15615	0.000889
C2	28330	28297	33223	0.001667
C3	11704	11689	16561	0.000778
C4	10736	10723	15159	0.000222
C5	7214	7202	15303	0.001067
D1	66202	66196	95891	0.000067
D2	168720	168693	195624	0.000210

TABLE V

COMPARISON OF RATES AND CROPPING: Across trace sets A,B,C and D, mean ideal, encoded and requested rates; proportion of frames suffering $\geq$ 20% cropping

F. Comparison with Video Teleconferencing Traces

We now look at the effectiveness of the SAVE smoothing and rate-adaptation algorithm for the video-teleconferencing (VTC) traces. The traces we studied are from Group C as well as two other, higher-rate teleconference traces from group D. The smoothing parameters were: $w_{\max} = 1000$ and $w_{\rm sm} = 1$, and β = 1.05. We also include the results for trace A, whose parameters were identical, except for $w_{\rm sm} = 12$. All the results are for a network feedback delay of 1 frame time. The first 5 VTC traces are characterized by an initial, large I frame, followed by much smaller P frames. Although the delay for the first frame tends to be somewhat dependent

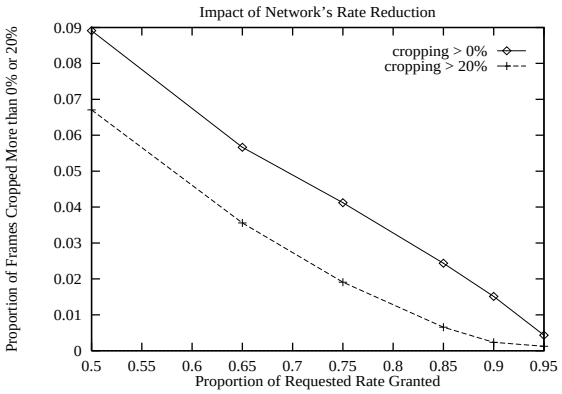

Fig. 5. SENSITIVITY OF CROPPING TO RATE REDUCTION: Proportion of Frames suffering cropping when network allocates a proportion ρ of requested rate. Trace A, feedback delay=1 frame time.

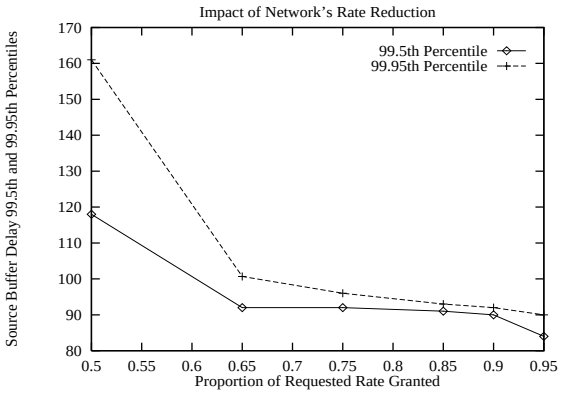

Fig. 6. SENSITIVITY OF DELAY TO RATE REDUCTION: Source Delay when network returns a proportion ρ of requested rate: Trace A, feedback delay=1 frame time.

on the initial rate, we have continued to choose the initial rate to be approximately equal to the long-term mean of the ideal rate. This tends to skew our results somewhat, with the measured quality degradation being higher than it would be if we were to ignore the first frame. Table V shows the summary statistics for 5 traces.

Because the first 5 VTC traces tend not to have a substantial peak to mean ratio, the benefit is primarily obtained from the source smoothing and rate-adaptation mechanism. The quality (measured by the proportion of frames suffering more than a 20% cropping) is very good across all the traces, and in particular for the two higher-rate VTC traces, D1 and D2. The mean requested rate is higher than the mean ideal rate by a factor ranging from 1.15 (for the longest trace with more variability) to 2.1 (for the shortest trace with the least variability). Some of this may be caused by the large requested rate for the first large I-frame, which may skew the result. In all of these cases, the peak requested rate is higher than the peak ideal rate by a factor of 1.05, which is precisely the factor β.

VI. MULTIPLEXING EXPERIMENTS AND ANALYSIS

In this section we report experiments on the multiplexing properties of the aggregate demand. We want to establish the bandwidth allocation which will be required per source, and see how this depends on the number of sources aggregated.

Experiments supplementary to those reported in Section V-E investigate the sensitivity of quality under SAVE to bursty rate reduction as modeled by the ON-OFF process described in Section IV-E. We find that an allocated rate down to about $\rho = 0.9$ times the requested rate can be tolerated for a wide variety of traces, even if this happens for say $T_\rho = 50$ frames out of every $350 = T_1 + T_\rho$.

If only a proportion of the rate requested by an individual source is allocated by the network, then quality is degraded. We examine what is the link capacity K needed to support a set of multiplexed streams on that link with adequate capacity. Consider, for a given set of frames n, the proportion of the rate allocated by the network to the aggregated requested rate, $\rho_{\rm agg}(n) = K/R(n)$ Here, $R(n)$ is the aggregate requested rate across all the sources.

Based on the observations in Section V-E, we deem that a link capacity K is adequate if:

- the average value of $\rho_{\rm agg}(n)$ is greater than ρ (≈ 0.9) for those frames for which $\rho_{\rm agg}(n) < 1$; and
- the proportion of frames that suffer degradation because $\rho_{\rm agg}(n) \leq R(n)$ is not more than $T_\rho/(T_1 + T_\rho)$, and the mean number of consecutive such frames is no more than T_ρ.

The smallest value of K which satisfied these conditions then becomes a target for admission control mechanisms.

We examined the aggregation properties of the 19 traces of set E, and 10 4000-frame segments of the single trace A. We get an initial view by showing quantiles of the aggregate requested rate (as multiple of its mean) as a function of aggregation size: in Figure 7 for set E. The range between minimum and maximum rate of the aggregate was about one half of the mean rate. For comparison, Trace A has better aggregation properties, the min–to–max range being only about one tenth of the mean. This is partly a reflection of the parameters used in the SAVE algorithm in each case: to obtain roughly equal quality per trace, $w_{\max}$ was 1000 for set A, 12 for set E. Thus we would expect the single flow requested rate to be smoother for set A, and hence that the aggregation properties are better.

We investigate the detailed performance of a large aggregate. We used the aggregate of all traces (in each set independently), and set the capacity K to a number of different high quantiles of the aggregate rate. We recorded

the statistics of excursions above this level: the mean length of runs of consecutive frames above and below K, the maximum frame above, and the maximum and mean of the proportionate rate reduction $1 - \rho_{\rm agg}$.

For trace set E, the time-series of the aggregate rate is shown in Figure 8. Note that the vertical scale starts at 500,000 (i.e., about twice the vertical range shown). The analysis of rate reduction is summarized in Table VII. Since the traces from this set have a 12-frame GOP, we did the analysis for two cases: with the I-frames aligned, and with random alignment. Whereas we would expect the performance on the randomly aligned case to be better due to smoothing across sources, we would not expect it to be much better, since the SAVE algorithm will typically request a rate determined by the largest frame size in the GOP. This is what we find: Table VII shows that for random alignment a link capacity of between 1.1 and 1.2 times the mean requested rate should be sufficient to fulfill the rate reduction criteria described above for this aggregate of 19 traces. (We investigated a number of random alignments of the I-frames; the conclusions from these are the same).

For set A, the bandwidth requirement was typically a smaller multiple of the mean aggregate demand than for trace set E , even though the aggregate was of about half as many (only 10) sources. The analysis of rate reduction is summarized in Table VI. This shows that, for this aggregation of 10 trace segments, one needs to allocate capacity less than 10% over mean aggregate demand in order to achieve sufficiently small rate reduction (as seen by the max. reduction value in Table VI).

We also investigated the effect of aggregation size on bandwidth requirements. We observed that the 10% rate reduction from the network (allocated rate being 90% of the requested rate) is acceptable even for aggregation sizes as small as 5, with a capacity allocation of $K = 1.2$ times mean aggregate request. Observing this, and observing the median of the ratio between the mean request rate and the mean ideal rate, we arrive at a rule of thumb: in order to guarantee sufficient quality per flow, the channel data capacity should be a little over twice the sum, over all the flows, of the mean ideal rate. Of course, this is preliminary, possibly applicable only to the set of traces we have examined. We excluded several plots supporting this conjecture due to space considerations.

VII. Variable Quality and Responsiveness

Assuming that the rate requested by a flow is satisfied according to the requirements described in Section V-E, the main determinant of per flow quality is $w_{\max}$, the window over which the local maximum frame size is determined. This suggests the following two ways to choose different values of $w_{\max}$ at the source adapter (that drains the source buffer).

%ile	Agg. frame (bits)	× of mean	mean # fr. below	mean # fr. above	max. # fr. above	max. red. (%)	mean red. (%)
75	1038378	1.03	130	42.1	310	3.3	1.0
90	1051190	1.05	144	16.2	57	2.1	0.6
99	1063088	1.05	565	6	24	1.0	0.4
100	1073571	1.07	All	0	0	0	0

TABLE VI

DYNAMICS OF LEVEL CROSSINGS: Trace A; crossings of aggregate requested rate above quantiles.For each quantile: mean runs of successive frames below quantile, mean and maximum runs above quantile. Maximum and mean of (1-rate/quantile) during runs above quantile.

%ile		Agg. frame (kbits)	× of mean	mean # fr. below	mean # fr. above	max. # fr. above	max. red. (%)	mean red. (%)
75	A	614.	1.04	141	65.2	856	21.5	4.1
	R	*617.*	*1.04*	*165*	*63.2*	*443*	*19.4*	*4.0*
90	A	640.	1.08	262	41.5	383	18.0	3.5
	R	*643.*	*1.09*	*296*	*37.5*	*170*	*16.0*	*3.2*
99	A	695.	1.18	1357	22.5	34	11.1	2.6
	R	*691.*	*1.17*	*1235*	*15.3*	*31*	*9.7*	*2.3*
99.9	A	738.	1.25	5704	10.3	16	5.5	2.0
	R	*728.*	*1.23*	*7990*	*9.6*	*14*	*4.8*	*2.2*
100	A	781.	1.32	All	0	0	0	0
	R	*765.*	*1.29*	*All*	*0*	*0*	*0*	*0*

TABLE VII

DYNAMICS OF LEVEL CROSSINGS: Trace set E; crossings of aggregate requested rate above various quantiles in Figure 8. Results are shown for two different aggregates of the 19 traces: with I-frames A=aligned and R=random alignment (in italics) For key see Table VI.

Variable Static Target Quality. Not all applications, or instances of them, will have the same quality requirements. For example, one expects that users of entertainment video will have stricter quality requirements than video teleconferences. Thus by tuning $w_{\max}$ appropriately, the desired quality can be obtained. See for example Table VIII for quality resulting from the application of different values of $w_{\max}$ to the trace of set A. As one might expect, higher quality requires higher requested bandwidth.

Dynamic Achievement of Target Quality. If a quality target is explicitly given, and quality can be measured at the adapter, then $w_{\max}$ can be increased until the target quality is reached. For example, with the quality target used in this paper, we could keep a historical estimate of the frequency with which $f_{\rm enc}/f(n)$ falls below 80%; if the estimate were to rise above 0.1% of the frames being cropped, an increase in $w_{\max}$ would be triggered. The consequent increase in requested bandwidth has implications for the network: it might violate the assumptions under which the flow was admitted. However, in any case such an adjustment should be in response to long

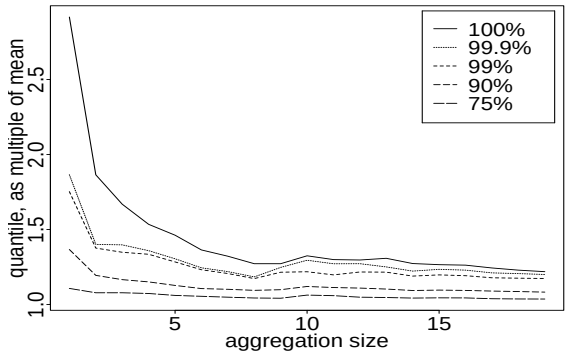

Fig. 7. QUANTILES OF THE AGGREGATED REQUESTED RATE: trace set E; quantiles expressed as a multiple of mean, for aggregations of 1 to 19 sources.

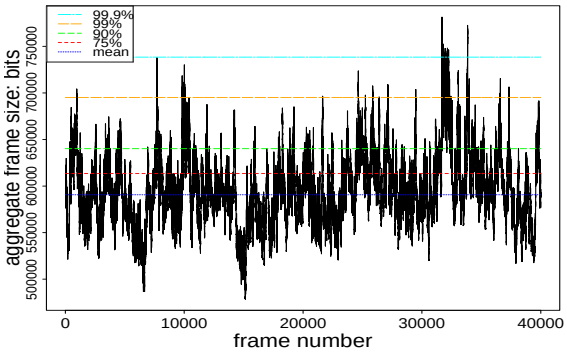

Fig. 8. AGGREGATED REQUESTED RATE: nineteen 40000-frame traces in set E. The vertical range is about 1/2 of the lowest value.

$w_{\max}$	request mean	cropping level			delay	
		>0%	>20%	=50%	mean	max.
1000	92236	0.0021	0.0007	7e-5	27	98
100	72856	0.0105	0.0048	0.0008	33	106
10	58610	0.0256	0.0161	0.0044	42	98
1	57607	0.0263	0.0166	0.0046	43	111

TABLE VIII

VARIATION IN QUALITY WITH MAXIMUM WINDOW $w_{\max}$ Trace A. Mean ideal rate is 54823. $w_{\rm sm} = 6$, $\tau_{\max} = 100$ms. 1 frame network feedback delay. Maximum cropping is $\gamma = 50\%$.

term failures in quality, based on average quality over a far larger timescale than $w_{\max}$. If such time-scales are long compared with the those at which the flows enter and leave the network, then upward readjustment of the long term mean requested rate should not compromise the admission control mechanism.

VIII. CONCLUSIONS

Efficiently transporting compressed video over a network has been a challenge. The desire has been to achieve high multiplexing gain, while satisfying an individual flow's quality requirement. We have proposed a smoothing and rate adaptation algorithm, called SAVE. SAVE is a follow on to the approach initially described in [14]. It uses the explicit rate based control mechanisms to transport compressed video. The rate control mechanism attempts to match the network's short-term allocation to the variations in the requirements of the video source at the time-scale of a scene. When necessary, we adapt the encoder by modifying the quantization parameter, so that we can carry the compressed video over the network with negligible loss and have minimal impact on the perceived quality. The combination of the explicit rate algorithm and the use of a smoothing buffer at the source helps us achieve a higher multiplexing gain. Source smoothing also helps us maintain a smooth requested rate of the network, thus aiding in the convergence of the explicit rate allocation algorithm.

SAVE has the following components:

- A source smoothing algorithm that finds a rate that is the maximum of:
  - a smoothed rate over a small moving window that is intended to capture the short-term average behavior, smoothing over the periodic structure introduced by the compression algorithm. The smoothed rate would drain the average frame over an inter-frame time.
  - an estimate of the maximum frame size over a sufficiently long term, so as to capture the characteristics of the peak rate required for a *scene*. This maximum frame has to be drained within the delay bound for the source buffer.
- A rate adaptation mechanism at the source that determines a target frame size so that a bound on the delay contributed at the source smoothing buffer is not exceeded.
- Finally, a rate is requested from the network, as a demand in the explicit rate congestion control mechanism used by the network to allocate a rate to this flow.

We show that SAVE is a very effective on-line smoothing algorithm. It is quite simple to implement and we feel it is intuitively appealing. In contrast to previous work, we find that using SAVE, we are able to meet much more stringent quality targets, and with the parameter $w_{\max}$ we can meet variable quality targets if needed. We see that SAVE is effective for a wide variety of compressed video sources: entertainment and teleconference sources with different compression algorithms: MPEG-1, MPEG-2 and H.261. The source adaptation mechanism proposed is quite robust, which is very desirable.

We looked at the effect of multiplexing several traces with SAVE. As a result of our smoothing algorithm, the ratio between the aggregate peak and the aggregate mean of the requested rate is relatively small. This is true, es-

pecially in comparison to the corresponding ratio for the ideal rate. We also examine the effect of varying the amount of capacity available, ranging from the mean of the aggregate requested rate to its peak. We show that when the capacity is at least 90% of the aggregate requested rate's *peak*, the quality and delay targets for an individual flow are still met. When the rate allocated by the network is adequate to achieve a desired stringent quality target with a given $w_{\max}$, any further increases in the network capacity does not improve the quality substantially.

SAVE is relatively insensitive to the setting of most of its parameters. In fact, for most of the traces we have used the same set of parameters. The only parameter that was changed was the smoothing window, $w_{\rm sm}$, to adapt to the periodic GOP structure. In some cases, especially for traces in set E, it was sufficient to use a $w_{\max}$ of 12 to meet our quality target. Using a larger $w_{\max}$ would only improve the quality.

We demonstrated the effectiveness of SAVE using a frame-level simulation. We have also confirmed our findings with a full-fledged cell-level network simulation.

REFERENCES

[1] "ATM Forum Traffic Management Specification Version 4.0," Draft Specification ATM Forum/95-0013R11, ATM Forum, March 1996.

[2] Bertsekas, D., Gallagher, R., Data Networks, ch. 6, Prentice Hall, Englewood Cliffs, N.J., 1992.

[3] Charny, A., Ramakrishnan, K. K., Lauck, A., "Time Scale Analysis and Scalability Issues for Explicit Rate Allocation in ATM Networks", *IEEE/ACM Transactions on Networking*, Vol. 4, No. 4, August 1996.

[4] Demers, A., Keshav, S., Shenker, S., "Analysis and Simulation of a Fair Queueing Algorithm", Proceedings of the ACM SIGCOMM'89 Conference, Sept. 1989.

[5] Duffield, N.G., Ramakrishnan, K. K., Reibman, A. R., "Issues of Quality and Multiplexing when Smoothing Rate Adaptive Video", Proceedings of the Proceedings of 8th International Workshop on Network and Operating System Support For Digital Audio and Video, July 1998.

[6] Elwalid A., Heyman D., Lakshman T. V., Mitra D, Weiss A., "Fundamental Bounds and Approximations for ATM Multiplexers with Applications to Video Teleconferencing", *IEEE Journal on Selected Areas in Communications: Special Issue on Fundamental Advances in Networking*, pp. 1004-1016, August 1995.

[7] Garrett, M.W., Willinger, W., Analysis, modeling and generation of self-similar vbr traffic. In *Proceedings ACM Sigcomm 94*,London, UK, August 1994, pp.269-280.

[8] Girod, B., "Psychovisual aspects of image communications", *Signal Processing*, vol. 28, pp. 239-251, 1992.

[9] Grossglauser, M., Keshav, S., Tse, D., "RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic", Proceedings of the ACM SIGCOMM'95 Conference, Sept. 1995.

[10] Heyman D., Tabatabai A., Lakshman T. V., "Statistical Analysis and Simulation Study of VBR Video Teleconference Traffic in ATM Networks", *IEEE Transactions on Circuits and Systems for Video Technology*, pp. 49-59, March 1992.

[11] Kalampoukas, L., Varma A., Ramakrishnan, K.K. "An Efficient Rate Allocation Algorithm for ATM Networks Providing Max-Min Fairness", Proceedings of 6th IFIP International Conf. on High Performance Networking, HPN '95, Spain, Sept. 11-15, 1995.

[12] Kanakia, H., Mishra, P. P., Reibman, A., "An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport", Proceedings of the ACM SIGCOMM'93 Conference, Sept. 1993.

[13] Kanakia, H., Mishra, P.P., Reibman, A., Packet Video Transport in ATM Networks with Single-Bit Feedback. In *Proceedings of the Sixth International Workshop on Packet Video*, Portland, Oregon, September 1994.

[14] Lakshman, T.V., Mishra, P.P., Ramakrishnan, K.K., "Transporting Compressed Video over ATM Networks with Explicit Rate Feedback Control", *Proceedings of the IEEE Infocom 1997 Conference*, Kobe, Japan, April 1997.

[15] Ma, Q., Ramakrishnan, K.K., "Queue Management for Explicit Rate Based Congestion Control", *Proceedings of the ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, June 1997.

[16] Mishra, P.P., Fair Bandwidth Sharing for Video traffic sources using Distributed Feedback Control. In *Proceedings of IEEE GLOBECOM*, Singapore City, Singapore, Nov. 1995.

[17] Ott, T.J., Lakshman T. V., Tabatabai A., "A Scheme for Smoothing Delay Sensitive Traffic Offered to ATM Networks", Proceedings of IEEE Infocom 1992, pp. 776-785, May 1992.

[18] Reibman, A.R., and Berger, A.W., "On VBR video teleconferencing over ATM networks," in *IEEE Global Telecommunications Conference (GLOBECOM)*, December 1992.

[19] Reibman, A.R., and Berger, A.W., "Traffic descriptors for VBR video teleconferencing over ATM networks", *IEEE/ACM Trans. on Networking*, vol. 3, no. 3, pp. 329-339, June 1995.

[20] Rexford, J., Sen, S., Dey, J., Feng, W., Kurose, J., Stankovic, J., Towsley, D., "Online smoothing of live, variable bit-rate video", Proceedings NOSSDAV, 1997.

[21] Rose, O., "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems". University of Wuerzburg. Institute of Computer Science Research Report Series. Report No. 101. February 1995.

[22] Salehi, J.D., Zhang, Z., Kurose, J.J., Towsley, D., "Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing", *Proceedings of the ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems*, pp. 222-231, May 1996.