# H.264/AVC RATE ADAPTATION FOR INTERNET STREAMING

*Thomas Schierl and Thomas Wiegand*

Fraunhofer Institute for Telecommunications – Heinrich Hertz Institute (HHI)
Image Processing Department
Einsteinufer 37, 10587 Berlin, Germany
{schierl,wiegand}@hhi.fraunhofer.de

## ABSTRACT

The delivery of video that efficiently adapts its data rate to changing network characteristics is one of the most challenging tasks in the design of today's real-time multimedia streaming systems. In this work, the primary focus is on network congestion and its effects on unicast real-time streaming using the H.264/AVC video coding standard. We propose a combination of temporal scalability and bit-stream switching for adapting the sending rate of video to the actually available data rate assuming a fair sharing of the available data rate. Furthermore, a TCP-friendly congestion control algorithm using retransmission has been integrated in order to improve reliability and fairness of multimedia streaming. Experimental results illustrate the effectiveness of the proposed approaches.

## 1 INTRODUCTION

Real-time streaming over the Internet has become an important way of delivering multimedia data to the user. Although presently there is an absolute dominance of proprietary Internet streaming systems, the new, highly-efficient H.264/AVC video coding standard [1] is gaining increasing acceptance and may help establishing open and more flexible standards for Internet streaming.

Transport of multimedia data in a real-time streaming with its limiting delay constraints is compounded with the problem of network congestion and other typical network conditions, such as transmission errors, delay, and jitter. Network congestion arises at overloaded network nodes. In such a case, e.g., the arrival rate at the overloaded node(s) is higher than the maximum service rate of the node(s). This results in dropping of packets at the overloaded node(s). When detecting a packet loss, the sending rate should be lowered by the sender in order to be fair to other affected data streams and to keep the networks functional. After some time, the sender can probe again whether a higher data rate maybe available, and in such a case increase the sending data rate and provide better quality signals. The above described behavior is managed by a so-called congestion control algorithm. Most of these algorithms also include a mechanism for retransmission of lost packets. Using feedback from the client, such as acknowledgments of a correct transmitted packet (ACK) or the loss of a packet (NACK), the sender may also adapt its sending rate accordingly.

The efficiency increase of all video codecs including H.264/AVC over still picture codecs is achieved through the exploitation of temporal statistical dependencies by motion compensation, i.e., predicting pictures by referencing other pictures. However, if a mismatch between the reference pictures at encoder and decoder occurs (e.g. by loosing a reference picture or reconstructing it at different qualities), the decoded video quality typically suffers significantly. Therefore, one of the main problems in a streaming scenario is the adaptation of the sending rate for the media streams. Our proposed methodology of rate adaptation of H.264/AVC encoded video is based on a combination of two basic functionalities:

1. ***Bit-Stream Switching (BSS)***: A set of pre-encoded video streams of the same content are stored with different pre-determined data rates. Each of these video streams contains periodically inserted so-called Instantaneous Decoder Refresh (IDR) pictures as entry points for switching between the various data rates.
2. ***Temporal Scalability (TS)***: The video stream contains temporally predicted pictures which do not serve as reference pictures for other pictures and can therefore be dropped without affecting the correct decoding of the other pictures in the stream. This way, a hierarchy of temporal dependencies can be created.

This paper is organized as follows. Section 2 describes the adaptive H.264/AVC-based streaming system including discussion of transport protocols and packetization. In Section 3, we present simulation results for an empirical analysis of video quality in presence of an emulated bottleneck by using different rate adaptation strategies.

## 2 ADAPTIVE H.264/AVC STREAMING

Fig. 1 shows an overview of the proposed adaptive streaming system. Key element is the rate adaptation at the server which adjusts the media rate depending on the feedback analysis in order to use the available and fairly shared data rate. The sender also includes a retransmission method which evaluates feedback messages generated at the client's side. Furthermore, the client comprises a receiver and data buffers for data exchange. The received media data are delivered to the media decoders and presented in a synchronized way.

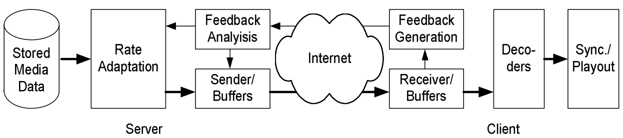


**Fig. 1**: Overview of the adaptive streaming system

### 2.1 Temporal Scalability in H.264/AVC

H.264/AVC supports a concept known as temporal scalability (TS), which is based on a layered representation of coded pictures. A base layer corresponding to a low picture rate can be enhanced by additionally temporal enhancement layers. TS is achieved by the requirement that encoded pictures of a higher layer are not referenced by pictures of lower layers. Exploiting temporal statistical dependencies is the task of motion-compensated predictive coding in order to provide high coding efficiency.

An example for the (temporal) layering of pictures is shown in Fig. 2. The video is divided into groups of pictures (GOPs) with each GOP starting with an IDR picture, which cuts off all inter-picture dependencies from any picture decoded prior to the given IDR picture. Therefore, each GOP is independently decodable. In Fig. 2, some of the possible references, i.e., temporal picture dependencies are illustrated by connecting arrows. Note that H.264/AVC allows also the use of multiple reference pictures [1].

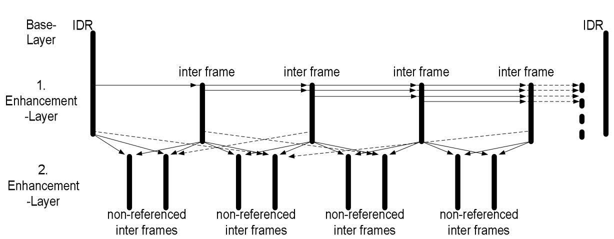


**Fig. 2:** Example of H.264/AVC temporal scalable layers

Additionally, it should be emphasized that H.264/AVC allows the use of non-referenced P-Slices for temporal scalability in the Baseline profile as well. In our three-layered example, the second enhancement layer consists of non-referenced pictures only, the first enhancement layer contains unidirectional predictive pictures, and the base layer consists of IDR pictures. Assuming, e.g., a picture rate of 25 fps, a distance of IDR pictures of 25 pictures, and two non-referenced pictures between each pair of referenced pictures, dropping of the second enhancement layer would result in a third of the original picture rate. By additionally dropping the first enhancement layer, only one picture per second would be transmitted.

The data rate reduction that is obtained by dropping enhancement layers usually depends on the encoding parameters and the characteristics of the video. For our test conditions, we typically measured reductions of data rate between 30 and 40 % by dropping the second enhancement layer depending on the chosen target bit-rate and the chosen encoding parameters. Since dropping of enhancement layers results also in a picture-rate reduction which can be an annoying artifact, we found that TS should preferably only be a method for short-term data rate adjustment.

The loss behavior for different picture types has been analyzed with respect to the objective video quality at the receiver for layered bit-streams according to the structure of Fig. 2. A typical result is shown in Fig. 3 depicting the loss rate of picture types against PSNR. The curved marked with the diamond-shaped symbol shows the case when a particular percentage of non-reference pictures is lost. The curve marked with the square symbol shows the case when a particular percentage of reference pictures except for the IDR pictures is lost, while the curve marked with the triangles shows the case when a particular percentage of all pictures except for IDR pictures is lost. As expected, this analysis shows a severely negative effect for the video quality at the receiver side when reference pictures are lost. Thus, avoiding the loss of reference pictures should be one of the main principles for a video rate adaptation algorithm using TS.

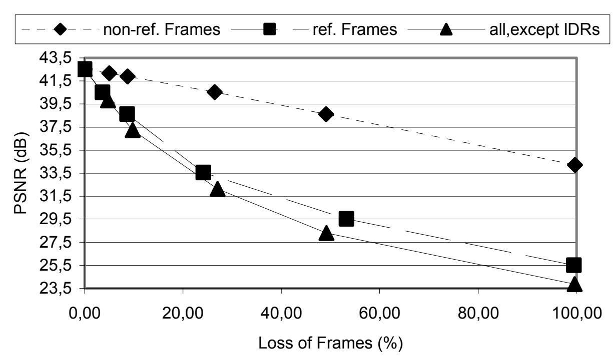


**Fig. 3**: PSNR vs. picture loss (352x208 pels, 640 kbit/s)

### 2.2 H.264/AVC Bit-stream Switching

Another technique for data rate adaptation of stored media is given by bit-stream switching (BSS). In [2], it was proposed to use such a strategy within a proprietary streaming system.

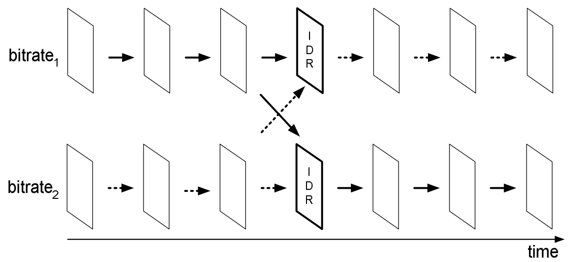


**Fig. 4:** H.264/AVC bit-stream switching

As illustrated in Fig. 4, IDR pictures in H.264/AVC cannot only be used for random access but also for switching between streams coded at different target bit-rates but using the same GOP structure [1]. The target bit-rate of a stream can be adjusted by changing the quantization parameters (QPs) used for encoding, since the QP determines the reconstruction accuracy of transform coefficients.

By making bit-streams of different data rates for the same video content available at the server, it is possible to obtain a coarse grained SNR-scalability, which allows to adapt the data rate by switching between the different bit-streams.

Fig. 5 shows video quality in terms of PSNR against different bit-rates for a given video. Switching between bit-streams results in a lowered video quality, but produces no picture-rate reduction like TS. Since switching is only possible at IDR pictures, the temporal rate of this picture type determines the maximum possible switching frequency.

### 2.3 Media Transport, Congestion Control and Feedback

Using the Real Time Transport Protocol (RTP) [3] over UDP allows the application to react to delays, losses, and congestions, since UDP is an unreliable and connectionless transport protocol. Therefore, RTP conveys information like packet sequence number, media composition timestamp, payload type and fragmentation information. Further rules for packetizing media data are specified in the RTP payload formats, for H.264/AVC see [4].

TCP is the one and only standard for congestion control protocols used on the Internet. Therefore, congestion controlled data flows should meet the requirement of TCP-friendliness [5]. That means, in case of congestion, all streams using the same bottleneck should share the available data rate in a fair manner such that every TCP-friendly congestion controlled stream gets the same portion of the bottleneck bit-rate.

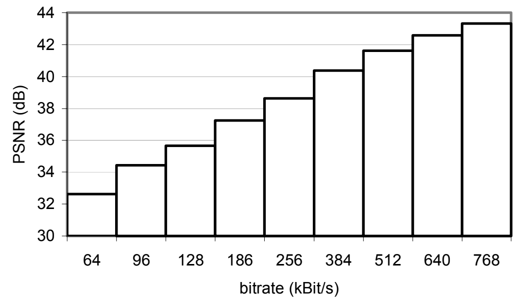


**Fig. 5:** Bit-rate against PSNR (352x208 pels)

We implemented a binomial congestion control algorithm, as proposed in [6]. This algorithm follows an Inverse Increase Additive Decrease (IIAD) behavior. The consequence of this behavior is less aggressive data rate reduction compared to TCP's Additive Increase and Multiplicative Decrease (AIMD) behavior when detecting congestion. On the other hand, the intensity of probing for new available data rate is also lower. Packets lost by congestion are retransmitted within the same RTP stream.

Such a congestion control is based on a window $W$ which indicates at a time $t$ how many packets are allowed to be sent. If a packet loss is indicated the window is decreased (D) and if a correctly received packet is indicated, the window is increased (I). The following equation gives the recalculated size of the window.

$$D: W_{t+\delta t} = W_t - \beta; \quad 0 < \beta < 1$$
$$I: W_{t+R} = W_t + \alpha / W_t; \quad \alpha > 0$$

The values $\alpha$ and $\beta$ are constants that control the intensity of the window adjustments. We used values of $\alpha = 1$ and $\beta = 0.5$ for our investigations. $\delta t$ is the time offset at which the next packet loss is detected. $R$ is the round trip time (RTT) after which a packet can be acknowledged.

Please note that we could use every other transmission rate / congestion control algorithm in the proposed system without any problem. Such a rate adaptation can also be based on different, simpler feedback mechanisms like the use of the RTCP Receiver Report in addition with a RTCP buffer report like proposed by 3GPP [7].

For the investigations presented in this paper we have used an RTCP ACK feedback similar to what is proposed in [8] including a sequence number and a bit-mask for selectively acknowledged sequence numbers following the actual sequence number. That results in a transmission of a grouped block of ACKs. Furthermore the playout delay of the client buffer is signaled by the client via RTSP to the server and that value can be updated as needed. An update is especially needed in case of extremely bad network condition, when a rebuffering at the client is required.

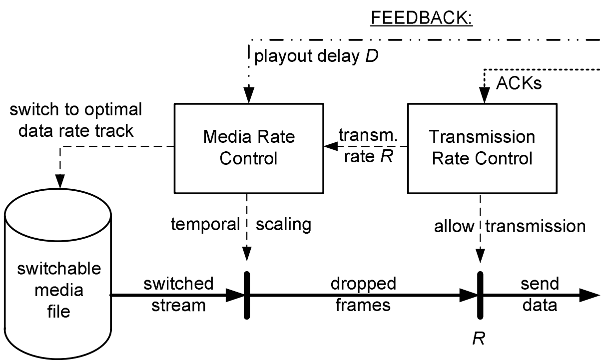


**Fig. 6:** Transmission and media rate adaptation

### 2.4 Transmission rate and media rate adaptation algorithm

The system's media rate adaptation depends on the transmission rate control and the client buffer status, see Fig. 6. The transmission rate $R$ is a result of the IIAD congestion control algorithm. The transmission of each media packet, which is scheduled by the server for sending, is controlled by the transmission rate control. The delayed transmission of a packet has an effect on the client buffer playout delay $\Delta$. If the reported and estimated playout delay goes below a scaling-threshold $S_i$, the media rate adaptation algorithm is activated, see Fig. 7. The algorithm contains three scaling thresholds at the server. If the delay $\Delta$ is below of these thresholds, bit stream switching or temporal scalability is deployed. See following table for a typical set-up.

| Scaling Threshold $S_i$ | Deployed Adaptation Technique |
|---|---|
| $S_1$ , e.g. $\Delta_s$ - 500 ms | Switch to optimal data-rate track / Drop 1st layer (non-ref. pictures) |
| $S_2$ , e.g. $\Delta_s$ - 1000 ms | Switch to optimal data-rate track / Drop 1st – 2nd layer (+ref. pictures) |
| $S_3$ , e.g. $\Delta_s$ - 1500 ms | Switch to optimal data-rate track / Drop 1st – 3rd layer (+IDRs) |

For the scaling threshold calculation we are assuming a pre buffer $\Delta_s$ at playout startup of 3 sec and a RTT below 1 sec.

The important parts of this algorithm are described as follows in pseudo code:

*IF( $\Delta \leq S_i$ ) THEN activate rate adaptation*

In this case, the algorithm first tries to switch to a lower data rate track x for that the maximum data rate is $TR_x$.

*IF( find track* x *with $TR_x < R$ )*
*THEN switch to track* x
*ELSE IF( $D \leq S_i$ )*
*THEN drop layers up to layer* i

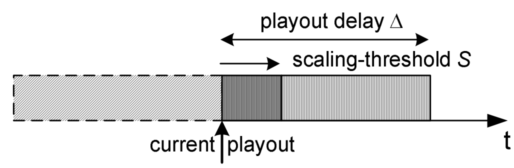


**Fig. 7:** Scaling threshold ($S_i$)

If no lower data rate track is available or bit stream switching cannot be applied, because the next possible switching picture is too far, the system starts dropping pictures. If we are already in picture dropping mode, the next lower layer will be dropped.

When network conditions improve, which means that the playout delay $\Delta$ is again above of scaling threshold $S_i$, the algorithm stops dropping a certain layer and tries to switch to the next higher available bit-rate.

*IF( $\Delta > S_i$ AND dropping layer j active, with $j \geq i$ )*
*THEN reduce dropping, drop up to layer* i-1

*IF( $\Delta > S_{min}$* AND *track x exists, with $TR_x > TR_{curr}$ )*
*THEN switch up to track* x

The data rate and playout delay are tested in fixed time intervals. There is one testing interval for scaling up and one for scaling down. In order to guarantee that there are no loops in switching and scaling, after scaling down the testing interval for scaling up $I_{su}$ is increased by $\Delta w_{su}$ and the scaling down interval is decreased to a fixed value $I_{sd}$, which represents the time interval for the minimal bitrate measuring window. For each down scaling the scaling up interval $I_{su}$ is increased again by $\Delta w_{su}$. If the algorithm detects a $\Delta$ above the minimum scaling threshold in a scaling up interval, $I_{su}$ and $I_{sd}$ are set to initial default values. In case of having a good client buffer the system switches up to a higher data rate track, which has a bitrate fitting into the available network bandwidth. By that the system achieves optimal quality with the available link bandwidth.

### 2.5 Implementation

All the techniques described above have been implemented into the proposed real-time streaming system. We are using a pre-buffering of 1 second maximum at the client in order to achieve retransmission, sending-rate control with BSS, and TS. A maximum amount of 32 packets are acknowledged by an RTCP ACK packet. An algorithm keeps the feedback of the session below 5 % of the session data rate as proposed in [8].

Furthermore, the server optimally exploits the available data rate that means streaming at data rates higher than the media stream rate is possible. A maximum client pre-buffer size limits that behavior. The video streams at different bit-rates are delivered to the server by the AVC file format [9].

## 3. EXPERIMENTAL RESULTS

We have used NISTNET [10] for emulating network behavior including bit-rate limitation for transmission sessions. Two PCs have been connected via a Linux-router using NISTNET. We compared the TS approach with a combinational approach of TS and BSS. Two streams each using one of these techniques are transmitted through a bottleneck emulated by NISTNET. Additionally, two TCP streams are transmitted via the NISTNET router. The round trip time RTT was set to 60 msec. Packet loss only occurs due to queue overflows in the emulated router. The maximum router queue size corresponds to about 40 packets, which is equal to 384 kbit with an average packet size of 1200 bytes for the 640 kbit/s stream.

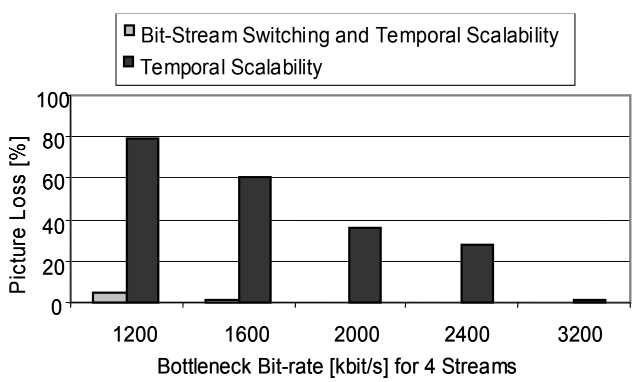


**Fig. 8:** Picture loss against bottleneck bit-rate, two multimedia streams with max.700kbit/s and two TCP streams

In order to emulate a realistic multimedia transmission, we also transmitted an AAC audio stream at 66 kbit/s in conjunction with the video stream. The temporally scaled video stream has an average bit-rate of 640 kbit/s and the data rates for BSS are 96, 128, 186, 256, 384, 512, and 640 kbit/s. Thus, the resulting maximum data rate per multimedia session is about 706 kbit/s on average.

Fig. 8 shows the picture loss percentage against the overall available bottleneck data rate for the two rate adaptation techniques by using two multimedia streams with max. 700 kbit/s and two TCP data-streams, that use the available data rate. We found that the available data rate is fairly shared between the multimedia and the TCP streams. That means that nearly a quarter of the bottleneck bit-rate is available per stream. The combination of BSS and TS can significantly reduce picture loss in bit-rate constrained scenarios. In this context, the number of dropped pictures has not a linear dependency on the achieved rate reduction, since the pictures are not of equal size.

Fig. 9 shows the resulting PSNR measured as dB against the bottleneck bit-rate in kbit/s. In case of picture loss we took the last available picture for the PSNR calculation. The best quality results in our experiments have been achieved by using the combination of BSS and TS.

Furthermore, the RTCP feedback data rate fraction has been kept below 5 % of the multimedia session bit-rate.

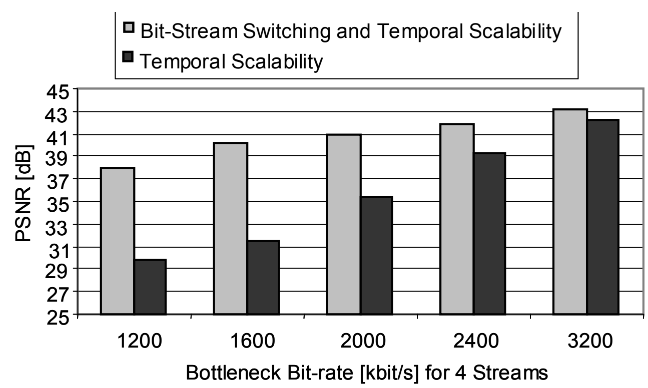


**Fig. 9:** PSNR against bottleneck bit-rate, two multimedia streams with max.700kbit/s and two TCP streams

## 4. CONCLUSIONS

We have presented an adaptive streaming system that has been implemented based on open standards. We mainly focused on the problem of data rate adaptation for H.264/AVC encoded video, and have proposed the combination of TS and BSS together with a universally deployable media rate adaptation algorithm. In particular, by incorporating a prioritized retransmission for TS and a congestion control, we achieved a kind of reliable streaming within a unicast real-time streaming environment. The experimental results verify that the combination of TS and BSS should be the preferred method.

Future work will be devoted to the use of the scalable extensions of H.264/AVC [11] for data-rate adaptation in the present unicast but also in multicast, broadcast and peer-to-peer transmission.

## 5. REFERENCES


[1] ITU-T Recommendation H.264 & ISO/IEC 14496-10 AVC, *Advanced Video Coding for Generic Audiovisual Services*, 2003

[2] G. Conklin, G. Greenbaum, K. Lillevold, A. Lippman, Y. Reznik, *Video coding for streaming media delivery on the Internet*, IEEE Trans. on Circuits and Systems for Video Technology 2001; 11(3):269-281.

[3] H. Schulzrinne, S. Casner, Frederick, Jacobson, *RTP: A Transport Protocol for Real Time Applications, RFC 3550*, Internet Engineering Task Force, July 2003.

[4] S. Wenger, M. Hannuksela, T. Stockhammer, M. Westerland, D. Singer, *RTP payload Format for JVT Video*, Internet Engineering Task Force, June 2004.

[5] J. Mahdavi, S. Floyd, *TCP-Friendly Unicast Rate-Based Flow Control,* Technical Note - end2end-interest mailing list, January 1997.

[6] D. Bansal, H. Balakrishnan, *Binomial Congestion Control Algorithms*, IEEE INFOCOM, April 2001.

[7] 3GPP TSG-SA4 PSM SWG, *TS 26.234 Rel.6, Transparent end-to-end Packet switched Streaming Services (PSS): Protocols and codecs*, May 2004

[8] J. Ott, S. Wenger, N. Sato, C. Burmeister, J. Rey, *Extended RTP Profile for RTCP-based Feedback (RTP/AVPF)*, Internet Engineering Task Force, January 2004.

[9] ISO/IEC JTC1/SC29/WG11 CODING OF MOVING PICTURES AND AUDIO, *Text of ISO/IEC 14496-15/FDIS – Part 15: AVC File Format*, August 2003.

[10] National Institute of Standards and Technology, *NISTNET*, http://snad.ncsl.nist.gov/itg/nistnet/

[11] H. Schwarz, D. Marpe, T. Wiegand, *SNR-scalable Extension of H.264/AVC,* JVT-035, December 2003