Technische Universität München
Lehrstuhl für Kommunikationsnetze
Fachgebiet Medientechnik

Proxy-based Video Transmission: Error Resiliency, Resource Allocation, and Dynamic Caching

Wei Tu, M.Sc. (TUM)

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:		Univ.-Prof. Dr.-Ing. Norbert Hanik
Prüfer der Dissertation:	1.	Univ.-Prof. Dr.-Ing. Eckehard Steinbach
	2.	Prof. Pascal Frossard (Ecole Polytechnique Fédérale de Lausanne, Schweiz)

Die Dissertation wurde am 29.09.2008 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 26.01.2009 angenommen.

To my beloved family and beautiful Munich

Abstract

Digital video transmission over today's communication networks still faces some challenges. Transmission errors, packet losses or time-varying delay significantly impact the quality of video applications. In this thesis, proxy-based transmission frameworks are considered to improve the quality of video transmission from several perspectives.

To improve the robustness of video transmission over error prone mobile networks, a proxy-based transmission strategy is proposed. The proposed system recovers from transmission errors on the downlink by retransmitting corrupted packets. Retransmission for conversational video is enabled by encoding the video with a fixed prediction distance which corresponds to the Round-Trip-Time on the downlink. For transmission errors on the uplink, a dynamic reference picture selection scheme is applied to predict the next frame from error-free video content, so that the error propagation caused by this error can be stopped. In both cases, the proxies are located at or close to the base station. This enables faster error recovery by sending feedback to the sender on the uplink and by receiving feedback from the receiver and resending corrupted packets on the downlink. The two parts, although individually designed, work well in concert. This proxy-based reference picture selection scheme leads to a significant performance improvement compared to traditional end-to-end error recovery approaches.

The second part of the thesis discusses the congestion control and resource assignment issue for video applications on congested network nodes. Incautious dropping of video packets leads to significant quality degradation. Proxies executing intelligent rate adaptation algorithms are able to give a better solution. Different algorithms are proposed for the scenarios when pure streaming video, pure conversational video, or both streaming and conversational videos are available. By employing the sent along side information, the proxy shapes the rate of incoming videos in a rate-distortion optimized way. Both conventional single layer videos and the newly appeared scalable videos are considered and corresponding solutions are investigated. Simulation results show that a huge improvement is achieved by the proposed methods running on the proxy nodes.

Proxy-based frameworks are also widely used in video on demand applications. The proxy located close to the clients is able to shorten the response latency from the system and de-

crease the traffic on the Internet between the proxy and the server. To address the weakness of the proposals in the literature, a dynamic segment caching algorithm is proposed. The design of the algorithm running on the proxy is based on the conclusion drawn from a sophisticated subjective test, where users prefer an immediate response from the system and a small deviation of the starting point is fully tolerable. The proposed approach considers the popularity difference between video fragments, therefore video content with high popularity is more likely to be cached. Furthermore, the proxy is also able to select a better serving mode to serve the user requests. Experimental results show that a much higher user satisfaction can be achieved by the proposed scheme compared with the conventional approaches.

The three parts of the thesis address the three important issues of video transmission over today's communication networks. The proposed proxy-based frameworks and the corresponding approaches significantly improve the quality of video applications.

Kurzfassung

Die Übertragung digitaler Videodaten über heutige Kommunikationsnetze ist trotz großer Fortschritte in den letzten Jahren noch immer eine große Herausforderung. Übertragungsfehler, Paketverluste und variable Verzögerungszeiten beeinflussen deutlich die Qualität von entsprechenden Anwendungen. Diese Arbeit beschäftigt sich mit Ansätzen, mit denen die Qualität der Videoübertragung bezüglich mehrerer Aspektive verbessert werden kann.

Um die Robustheit der Videoübertragung über fehleranfällige Mobilfunknetze zu verbessern, wird eine proxy-basierte Übertragungsstrategie vorgeschlagen. Das vorgeschlagene System erholt sich von Übertragungsfehlern auf dem Downlink durch Rückübertragung der beschädigten Pakete. Die Rückübertragung für dialogorientiertes Video wird durch eine feste Prädiktionslänge bei der Videocodierung ermöglicht, die von der Round-Trip-Time auf dem Downlink abhängt. Für die Übertragungsfehler im Uplink wird eine dynamische Referenzbild-Auswahl angewendet, um das nächste Bild fehlerfrei vorhersagen zu können, so dass die Fehlerfortpflanzung gestoppt werden kann. In beiden Fällen befinden sich die Proxies auf oder in der Nähe einer Basisstation. Dies ermöglicht eine schnellere Fehlererholung des Systems durch die Fehlermeldung an den Sender auf dem Uplink und durch den Empfang von Rückmeldungen vom Empfänger und die Rückübertragung der beschädigten Paketen auf dem Downlink. Die beiden Teile, obwohl individuell gestaltet, funktionieren gut zusammen. Diese proxy-basierte Referenzbild-Auswahl führt zu einer deutlichen Leistungssteigerung im Vergleich zu herkömmlichen Ansätzen zur Ende-zu-Ende-Fehlerbehebung.

Der zweite Teil der Dissertation diskutiert die Staukontrolle und Ressourcenzuordnung für Video-Anwendungen auf überlasteten Netzknoten. Unvorsichtiges Abwerfen von Video-Paketen führt zu einer signifikanten Qualitätsverschlechterung. Proxies, die intelligente Ratenanpassungsalgorithmen ausführen, können eine bessere Lösung bieten. Verschiedene Algorithmen werden für die unterschiedliche Szenarien Streaming-Video, dialogorientiertes Video, sowie kombiniertes Streaming- und dialogorientiertes Video vorgeschlagen. Durch die Verwendung der mitversandten Seiteninformation, reguliert der Proxy die Rate der ankommenden Videoströme mit einem raten-verzerrungsoptimierten Verfahren. Sowohl herkömmliche

Single-Layer-Videos als auch skalierbar codierte Videos werden untersucht und entsprechende Lösungen vorgestellt. Simulationsergebnisse zeigen, dass eine beachtliche Verbesserung erreicht wird, wenn die vorgeschlagenen Methoden auf dem Proxy-Knoten laufen.

Proxy-basierte Ansätze werden auch in Video-on-Demand-Anwendungen häuftig verwendet. Der Proxy in der Nähe der Clientgeräte kann die Antwortzeit verkürzen und den Datenverkehr im Internet zwischen dem Proxy und dem Server reduzieren. Um auf die Schwächen der in der Literatur vorgeschlägenen Ansätze einzugehen, wird ein dynamischer Segment-Caching-Algorithmus vorgeschlagen. Der vorgestellte Ansatz basiert auf einer Schlussfolgerung aus aufwändigen subjektiven Tests, in denen sich gezeigt hat, dass die Nutzer eine sofortige Reaktion des Systems vorziehen und eine kleine Abweichung vom Ausgangspunkt vollkommen toleriert wird. Das vorgeschlagene Verfahren berücksichtigt die unterschiedliche Popularität zwischen Video-Fragmenten, wodurch der Video-Inhalt mit hoher Popularität auch mit hoher Wahrscheinlichkeit zwischengespeichert wird. Außerdem kann der Proxy auch der Benutzeranfrage einen besseren Dienstmodus bieten. Experimentelle Ergebnisse zeigen, dass durch die vorgeschlagene Methode im Vergleich zu den konventionellen Ansätzen eine deutlich höhere Zufriedenheit der Nutzer erreicht werden kann.

Die drei Teile der Dissertation befassen sich mit drei wichtigen Themen der Videoübertragung über heutige Kommunikationsnetze. Die vorgeschlagenen proxy-basierten Ansätze und die entsprechenden Algorithmen verbessern die Qualität von Video-Anwendungen deutlich.

Acknowledgements

I am extremely grateful to my advisor Prof. Eckehard Steinbach for his excellent guidance throughout my Ph.D study. His creative idea, constant encouraging, friendly criticism and kind help benefit me quite a lot. I have learned from him not only the knowledge in the research topics, but more important, the way of doing scientific research.

I would like to thank Prof. Pascal Frossard for accepting to be the second auditor of my thesis and Prof. Norbert Hanik for heading the committee.

Also many thanks are given to my colleagues in the Institute of Communication Networks, especially the members from the Media Technology Group. A lot of valuable knowledge sharing, creative discussions and pleasure chatting make my life in the institute exciting and unforgettable. Special thanks to Yang Peng, Hu Chen and Fan Zhang for the "kick session" after lunch and the sc games on weekend, which make a lot fun during my Ph.D study.

I would also like to thank Günter Liebl, Dr. Hrvoje Jenkač and Dr. Jacob Chakareski. I really enjoy the joint work and valuable discussions with them. Especially, with our effort, the work with Günther Liebl obtained the "Best Paper Award" in Packet Video Workshop in 2007.

Last but not least, I would like to express my deep gratitude to my parents, my wife Jessie for their unselfish support, patience and love, and also many thanks to my son for the happiness he brings to me.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Description	Definition
ARD	Accelerated Retroactive Decoding	page 17
AVC	Advanced Video Coding	page 15
BS	Base Station	page 14
CBR	Constant Bit Rate	page 4
CGS	Coarse Grain Scalability	page 60
CIF	Common Intermediate Format (352x288 pixels)	page 85
CNN	Cable News Network	page 84
CPU	Central Processing Unit	page 4
DC	Distortion Chain	page 47
DECA	Dynamic sEgment-based Caching Algorithm	page 83
DW	Decision Window	page 48
DPB	Decoding Picture Buffer	page 18
DSL	Digital Subscriber Line	page 4
EGOP	End-GOP	page 104
FDDI	Fiber Distributed Data Interface	page 4
FDRPS	Fixed-Distance RPS	page 14
FEC	Forward Error Correction	page 15
FLRPS	Frame Level RPS	page 24
F-MDDE	Feedback-based MDDE	page 18
FMO	Flexible Macroblock Ordering	page 15
HTTP	Hypertext Transfer Protocol	page 5
IPv4	Internet Protocol version 4	page 5
IPv6	Internet Protocol version 6	page 5
LAN	Local Area Network	page 1
LRU	Least Recently Used	page 83
MB	Macroblock	page 15
MDC	Multiple Description Coding	page 15
MDDE	Multi-Decoder Distortion Estimation	page 17
MGS	Medium Grain Scalability	page 60

Abbreviation	Description	Definition
MPEG	Moving Picture Experts Group	page 1
MS	Mobile Station	page 14
MSE	Mean Square Error	page 8
NACK	Negative Acknowledgment	page 18
NAL	Network Abstraction Layer	page 60
NEWPRED	New Prediction	page 18
PAPA	Popularity-Aware Partial cAching	page 82
PDA	Personal Digital Assistant	page 5
PRD	Priority-based Random Dropping	page 43
PRED	Priority-based Random Early Dropping	page 43
PRPS	Proxy-based RPS	page 26
PSNR	Peak-Signal-to-Noise Ratio	page 8
QCIF	Quarter Common Intermediate Format (176x144 pixels)	page 27
QoS	Quality of Service	page 1
QP	Quantization Parameter	page 22
RD	Rate-Distortion	page 10
RESCU	Recovery from Error Spread using Continuous Updates	page 17
RIMU	Random INTRA Macroblock Update	page 16
RNG	Random Number Generator	page 97
RPS	Reference Picture Selection	page 17
RS	Reed Solomon	page 15
RTT	Round-Trip-Time	page 2
SLRPS	Slice Level RPS	page 24
SLRPSEC	SLRPS with Error Concealment	page 25
SNR	Signal-to-Noise Ratio	page 44
SVC	Scalable Video Coding	page 59
TCP	Transmission Control Protocol	page 5
TV	Television	page 84
VBR	Variable Bit Rate	page 4
VCR	Video Cassette Recorder	page 80
VDW	Virtual Decision Window	page 49
VoD	Video on Demand	page 11
VoIP	Voice over IP	page 3
VLC	VideoLAN Client	page 84
WAN	Wide Area Network	page 1
WGOP	Whole-GOP	page 104
WLAN	Wireless Local Area Networks	page 4

Chapter 1

Introduction

Multimedia information, including audio, video, image and text, is widely used in our daily communication. Among them, video and audio are continuous and should be processed and played according to their timeliness. Otherwise, the quality of the playout is degraded and some of the information included in the media is lost. Transmitting video data over wireline or wireless networks leads to additional challenges. Uncompressed video has a high data rate compared with other media types and it is typically not possible to transmit the raw digital video data directly. Videos compressed with state-of-the-art hybrid video codecs (e.g., MPEG-2, MPEG-4, H.264) are very sensitive to transmission errors or packet losses, which might lead to severe quality degradation. Therefore, well designed network infrastructures are desired to improve the Quality of Service (QoS) for video transmission.

1.1 End-to-end System Architecture

The classic definition of a transmission system includes a sender, a receiver and a transmission network in between, as shown in Fig. 1.1. The end-equipments in the figure are not necessarily computers, but include all communication terminals (e.g., telephones, set-top boxes, televisions, handheld devices). Similarly, the network cloud also includes all kinds of networks along the transmission path (e.g., Internet, Wide Area Network (WAN), Local Area Network (LAN), Cellular or Wireless Networks).

Figure 1.1: A traditional communication system

In order to speed up the transmission, people try to keep the network nodes with low complexity, which simply forward the packets passing through them. The famous paper "End-to-end arguments in system design" [SRC84] published in 1984 suggests to preclude the implementation of any kind of higher-level functions within a network. This means that the transmission network is almost transparent to the user data sent through it and all control mechanisms have to be realized by the end equipments, such as: rate adaption, error recovery, transformation, etc. However, as mentioned in [BCZ97] and [Moo02], such a purely end-system-controlled structure has the following shortcomings.

- Additional complexity has to be added on the end systems, which is unfavorable for the design of handheld devices.
- When the end-to-end Round-Trip-Time (RTT) is long, users might experience a large latency of the feedback from the other side in the session. When feedback information is used for error recovery (e.g., retransmission), the efficiency is significantly decreased.
- End users can only control the rate of their own session. However, the available transmission resource changes frequently in the best effort Internet if no additional resource reservation is supported. The end user is not able to know the information of all network nodes and links on the transmission path.

1.2 Proxy-based System Architecture

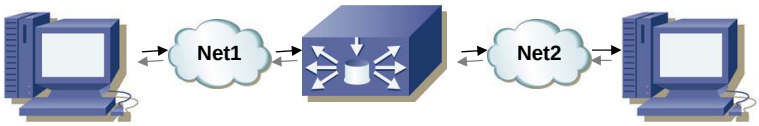


Figure 1.2: A proxy-based communication system

To overcome the above mentioned problems and to achieve more efficient transmission, proxies are widely used in todays network for a huge variety of services. As shown in Fig. 1.2, a proxy is an intermediary placed in the path between the sender and receiver, separating the network cloud into two subnetworks marked as Net1 and Net2. The proxies are normally built on top of the conventional network nodes (e.g., gateways, routers, switches, base stations, access point, service controller, etc.) to enable additional computation and storage capacity of the nodes.

Proxy-based approaches can be classified according to the type of application they are involved in, such as Web-browsing [CLZ99, ASTP03, PZ07, WSA08, DA99, SSV99, BV07,

SPvSA07], Voice over IP (VoIP) [GWA07, TZM08, RQ08], Video on Demand [WSAT02, WW07, FLSA+01, CSY+05, ILL07], and others. A more meaningful way of classification follows the main task or function executed by the proxies. In the following, some of the most important tasks and functions of these intermediaries in the proxy-based systems are introduced.

- **Resilience and Recovery** are particularly important for the transmission of time-critical traffic over those networks with large and/or variable end-to-end delays. Proxies can be setup on the transmission path to improve the error resilience of data transmission or when an error happens, to enable a fast recovery. For that, proxies should be located at the network nodes close to the "bottleneck links" [BBAC06], which have the highest bit error rate or packet loss rate. When the transmission is over heterogeneous networks including wired and wireless networks, the proxies are normally placed at the edge of the wireless networks, as they are typically more error prone and resource constrains than the wired networks.

  The contribution of the proxies for resilience and recovery is mainly due to the following two facts: (1) They are able to collect more accurate information of the network condition as they are close to the links with impairments. (2) The delay of the feedback information from the receiver or downstream node is much smaller than that in an end-to-end infrastructure. Based on the timely and accurate information of the channel, the most suitable error protection rate or error recovery scheme can be determined. For instance, joint source-channel coding [PM03, MGC+01], or encoder-decoder resynchronization [BBAC06, MW00, TS04, WFC02, BG03, YYW04, MSH03].

- **Caching Management** is probably one of the oldest tasks of proxies and it is even where the name "proxy" comes from. It is quite normal that people in the same department or same group have similar interest and a lot of information will be visited therefore many times. Without proxies, all requested information has to be downloaded from the remote content server for many times, which generates a lot of traffic. However, this additional traffic can be avoided if a proxy server sits at the closest common network node on the path to the server (e.g., the gateway of a LAN). Another benefit that can be brought by using a proxy server is that the response latency is much smaller compared with the system without proxy. This is due to the fact that the round-trip-time between the client and the proxy is in general much smaller than that between the client and the server. As some of the content has been cached, when requested, it can be forwarded directly from the proxy, therefore experiencing a much lower latency.

  As proxies can not have infinite storage capacity, only a portion of previously requested content can be cached. In principle, video content with high popularity (i.e., hitting

rate) should be cached to reduce the traffic between the server and the proxy. People also try to cache the parts with high rate in a Variable Bit Rate (VBR) traffic, so that the necessary transmission rate over a Constant Bit Rate (CBR) channel can be decreased. As the popularity changes over time, the proxy should be able to manage its cache, replacing the old content with the new one. Many cache management schemes have been proposed for example, in [CLZ99, ASTP03, PZ07, WSA08, DA99, SSV99, BV07, SPvSA07, WSAT02, WW07, FLSA+01, CSY+05, ILL07]. A detailed review of the state-of-the-art in cache management approaches is given in Chapter 4.

- **Content Adaptation** aims at transforming the payload for optimized transmission and presentation at the user device. The specific adaptation pattern used is mostly determined by the available shared transmission rate on the central networks (e.g., in [MFW01]) and the application requirements which may consider the following issues: the type of access networks (DSL, Modem, FDDI, WLAN, cellular) and the characteristics of the device, such as its computational resource (CPU, memory, power), output capabilities (screen size, resolution, color-depth) and supported protocols or standards (e.g., HTML, MPEG-4). Many research works have considered the heterogeneity of user devices, especially for mobile devices. For example, in [HBL+98, WOM+00, LL02, AGL+03, MCCdL06, CCL06], the content adaptation is for one or several types of multimedia objects. Besides that, Fox *et al.* have proposed a novel adaptation scheme in [FGCB98], which considers the network condition and device characteristics jointly for the rate adaptation for text, images and video.

  The content adaption can be carried out at the terminals, but the additional complexity will be very expensive if they are handheld devices having limited capabilities. Even when the terminal is powerful (most likely the server at the sender side), it still has the problem to have enough knowledge about the specifications of end user devices as well as the varying network status. Therefore, the adaption on the proxy close to the "Bottleneck link" or client is more efficient. With more available information, the adaption can be performed on-the-fly by the proxy to support variations on the network and device characteristics, while saving the power of the client.

- **Resource Management** tries to optimally assign the available shared resources (e.g., power, memory, bandwidth) to the users/flows. Normally, to satisfy the requirement of one application, more than one type of resources has to be considered. Therefore, it is important to consider the relationship among different types of resources that are needed. An approach based on so called "Resource Vector" is proposed in [LW03] to adapt the resources needed by the application. The proxy-based resource management also enables the applications to react promptly to the resource status in the network

nodes and therefore offers a better performance guarantee and more flexibility to the applications.

The proxy sometimes also acts as a scheduler to manage the resource assignment on the time axis. It can, for instance, assign more transmission slots first to the applications with high priority or to the users currently with good connection in wireless networks.

- **Protocol Translation** is needed when the traffic is transmitted over more than two networks, which run with different protocols. For instance, most communication protocols used for wired networks are usually not suitable for wireless networks because of the big difference in their characteristics. Proxies are widely used to translate protocols. For instance, in [ZWX06], a protocol transition between IPv4/IPv6 is implemented on the proxy, so that future networks equipped with IPv6 can be seamlessly connected with the current IPv4 based Internet. Another possible protocol transition example for the proxy could be that between the conventional TCP and the so called TCP Split Connection Protocol used in the wireless networks as described in [Ela02].

- **Security and Privacy** is one of the most important topics in the design of communication systems. Proxies can be used to decentralize the authentication process so that its efficiency and reliability can be greatly improved compared to centralized authentication. With a proxy sitting between a WAN and a LAN, most of the power demanding encryption calculations can be shifted from the client to the proxy. [Neu93] demonstrates the implementation of a proxy-based authorization system, where the authorization scheme is migrated from the conventional systems.

- **Session Management** tries to maintain, recover or migrate the current session when it has to be terminated for some reasons. For instance, a user is browsing Internet with his PDA on his way to home. When he arrives at home, he would like to continue the browsing session, however, on the desktop, which has large screen and strong power. Several approaches have been proposed to enable the session management in a proxy-based framework. For example, [SCK02] introduces a browser session preservation and migration infrastructure that allows a user to take a snapshot of an active web session state on a browser and retrieve the snapshot at a later time on a browser to continue the same active web session on any device. [CSV+05] builds up a protocol atop HTTP, which enables the session hand-off in Web applications. By exploiting a proxy-based architecture, this protocol is also able to work on existing applications and Web infrastructure.

The basic idea of a proxy and a proxy-based infrastructure for different types of applications has been introduced above. This dissertation mainly focuses on the usage of proxies

for video applications, such as Video Conferencing, Video on Demand, and Video Streaming. These applications challenge the current network infrastructure compared with other traditional applications. They need high bandwidth, low delay and QoS support from the underlying networks. The current Internet is characterized as a best effort network. Therefore, packet transmission over Internet is still subject to random packet delay and losses, which severely degrades the desired QoS. Cellular networks, although have evolved to 3G, still can not catch up the step of the user requests to high quality video-based services. Furthermore, video applications themselves are sensitive to errors as a result of compression and entropy coding.

Among the main tasks and functionalities of a proxy-based framework, the first four, namely *Resilience and Recovery*, *Caching Management*, *Content Adaptation* and *Resource Management* have the highest relation to video applications. Firstly, resilient transmission strategies or efficient recovery methods are needed to compensate the sensitivity of video applications to transmission errors. Secondly, the large volume of video leads to a lot of congestion in the network. By optimal content adaptation and resource management, the congestion can be decreased, avoided or even eliminated. Finally, when caching is enabled, not only the traffic on the WAN can be greatly decreased, the short response latency and small delay jitter enabled by the proxy are also extremely important to video applications. This dissertation is going to study these four aspects, and proposals which fully explore the benefit of proxy-based infrastructure will be given to improve the QoS of video transmission.

1.3 Digital Video Preliminaries

In this section, some background knowledge about digital video is introduced.

1.3.1 Video Compression and Transmission

Uncompressed video requires huge storage and transmission capacities (e.g., 168 GB for a 90 minutes movie with 720×576@25Hz). As there are many similarities inside one frame or between neighboring frames, significant redundancy exists in raw videos. The purpose of video compression is to decrease the size of video objects by removing this redundancy and by introducing acceptable deviations from the original. Fig. 1.3 shows the basic structure and components of a video coder. The “Motion Estimation/Compensation” module is used to extract the temporal redundancy between neighboring frames. “Transform” and “Quantization” are also used in video coding, which significantly improve the compression ratio with small or even unnoticeable quality loss compared with the original video.

According to the way of prediction, there are three types of frames that are most commonly used. When a video frame is INTRA encoded (i.e., an I-frame), only the spatial redundancy

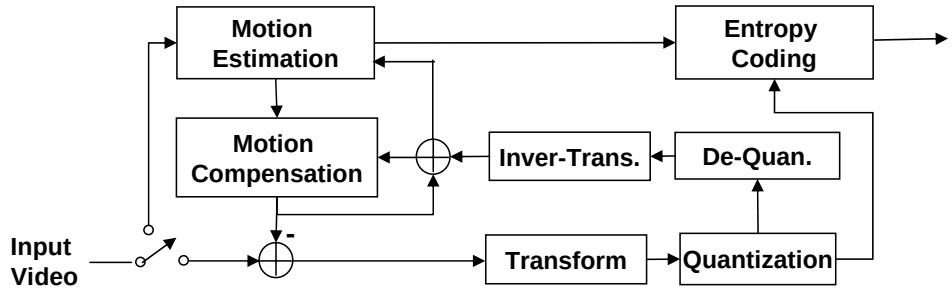
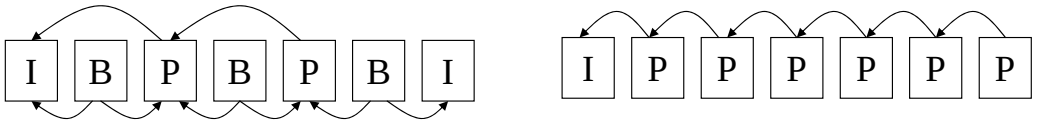


Figure 1.3: Common structure of a hybrid video coder

is considered. Each I-frame can be decoded independently as they do not perform any prediction from other frames. INTER coded frames (i.e., P-frame) significantly improve the coding efficiency by predicting from one previous frame near by, as more redundancy lies on the temporal domain between neighboring frames. Bi-directionally predicted frames (B-frame) further improve the compression ratio by using one previous and one future frame as reference frames. However, additional delay is introduced as it can be coded only after the future reference frame has been encoded. Fig. 1.4(a) shows how different types of frames do their prediction. Based on the prediction relation, it is obvious that the I-frames are most important, then the P-frames and the B-frames are of least importance.



(a) IBPBP structure for streaming videos (b) IPPP structure for conversational videos

Figure 1.4: Typical encoding structures

According to the requirements of different applications, videos can also be encoded in different structures as shown in Fig. 1.4. Compared to video downloading, in streaming video applications (e.g., Video on Demand), video is consumed while being delivered, allowing viewing in real time. After users select a movie or television program, it is retrieved as quickly as possible and played on the client device. One of the main benefits of streaming video is that the users do not need to spend a long time and large storage cost to download the whole video file to the local disk. For streaming video applications, the end-to-end delay constraint is normally several seconds. As shown in Fig. 1.4(a), B-frames are inserted between P/I-frames to achieve high compression efficiency. Furthermore, a coding structure called Group of Picture (GOP) is also employed by periodically inserting I-frames. One GOP includes one I-frame and all its successive frames until the next I-frame. As an I-frame is independently decodeable, this structure significantly improves the robustness by resynchronizing the encoder

and the decoder. The number of frames in one GOP and the number of B-frames between two I/P-frames can be adjusted according to the requirement on bitrate or resiliency. For conversational video applications (e.g., video telephony, video conferencing), the end-to-end delay constraint is normally restricted, and should be smaller than 200ms. The additional coding delay caused by the insertion of B-frames is not acceptable. Moreover, frequently inserting I-frames is also not feasible because it might lead to large fluctuation in bit rate, which is critical for conversational applications with extremely low delay. Therefore, the coding structure for conversational video is normally IPPPP... as shown in Fig. 1.4(b). Of course, there are also some further coding variations for the conversational video to improve its robustness, which will be introduced in more details in Section 2.2.

In addition to the conventional hybrid coding structure, scalable video coding [SMW07] emerges in these years as a promising video coding scheme. The scalable video has normally temporal scalability (i.e., number of frame per second), special scalability (i.e., resolution) and quality scalability (i.e., quantization level). Subbitstreams with lower frame rate or lower resolution or coarser quantization can be easily extracted from the original bitstream, which leads to a graceful quality degradation when the network can not fulfill the request.

Video frames are encoded, put into packets with some headers and then sent to the receiver through the network. The larger the payload size of a packet, the bigger the influence when bit errors or packet losses occur. On the other side, with small payload size, more cost has to be paid for the packet headers. In principle, it is quite acceptable that one video frame is sent in one packet. However, part of a frame (e.g., one row of macroblock) can be encoded independently and sent in one packet to improve the robustness at the price of low encoding efficiency.

1.3.2 Quality Evaluation

As the commonly used compression is lossy, the reconstructed video frame at the receiver side is not identical to the original one. When transmission errors or packet losses exist, the quality degradation can be dramatic. To judge the quality of a received video sequence, subjective evaluation is of course the most accurate one, which fully reflects the perceptual quality. However, subjective evaluation is every expensive. A good approximation to evaluate the quality of a received video frame is the Peak-Signal-to-Noise Ratio (PSNR), which is defined as

$$PSNR = 10 \cdot log_{10}(\frac{255^2}{MSE}), \tag{1.1}$$

where 255 is the largest possible value for a pixel, as any component of one pixel is presented with 8 bits. MSE is the Mean Squared Error of one video frame and can be calculated as

$$MSE = \frac{1}{X \cdot Y} \sum_{i=1}^{X} \sum_{j=1}^{Y} (f_{i,j} - \hat{f}_{i,j})^2, \tag{1.2}$$

where X and Y represents the number of horizontal and vertical pixels of the video frame, respectively. $f_{i,j}$ and $\hat{f}_{i,j}$ are the original and reconstructed value for pixel (i, j) in the frame. To evaluate the quality of a video, the mean PSNR value is calculated over all video frames, which is presented as

$$\overline{PSNR} = \frac{1}{L} \sum_{i=1}^{L} PSNR_i. \tag{1.3}$$

Normally, there are three components for a video stream consisting of L frames, one luminance (Y) and two chrominance components (C_r, C_b). As the human visual system is most sensitive to the luminance component, in this thesis, only the PSNR of the luminance component (Y-PSNR) is calculated for evaluation.

PSNR is able to present the quality of a video sequence, but can not give a comprehensive evaluation for some applications that are highly dependent on delay effects. There is no universal metric as PSNR that can be used to objectively evaluate the effect of delay to the service quality. In this case, the subjective user satisfaction should be a possible metric that can be used. For example, the influence of initial delay to user satisfaction might be only evaluated subjectively as done in Chapter 4.

1.4 Summary of Major Contributions

This dissertation studies the possible benefits that a proxy-based network infrastructure can bring to the transmission of digital video. In this work, some of the proposals adopt conventional techniques in the new framework, while other proposals improve previous ones for proxy-based networks. All of them lead to a significant improvement of the QoS for video applications. The major contributions are summarized as follows.

- **Error Resiliency**

A complete solution for conversational applications over mobile networks is proposed for mobile senders and mobile receivers [TS04, TS05, TS09]. Proxies are setup on the base stations where mobile users are allocated. The proposed system recover from errors on the downlink by retransmission of lost packets, which is enabled by using fixed distance reference picture selection when encoding and setting up proxy on base station to perform fast retransmission. The error recovery on the uplink is enabled by a dynamic reference picture selection scheme that only references error free parts of reference frames. The individual error recovery strategies for uplink and downlink can be also perfectly combined and give a complete solution for conversational applications over mobile networks. In order to quantify the benefits of the proposed approach, several state-of-the-art error resiliency schemes have been implemented and some of them are improved for comparison.

- **Congestion Control**

When congestion happens, rate adaptation and optimal resource assignment are able to release the congestion or at least achieve a better quality for constrained resources. A Rate-Distortion(RD)-optimized scheme is proposed to selectively drop video frames and assign the transmission resources to the flows from multiple users. Novel methods to extract rate-distortion side information from conventional single layer video and multi-layer scalable video are investigated. Frame dropping strategies that can be employed on any network node in wired networks for streaming videos [TKS04] and conversational videos [TCS08] are also developed. When both type of videos are available, a new RD-optimizer framework [TCS06, TCS08] is proposed including two scheduling schemes, namely mean-rate-based scheduling and buffer-fullness-based scheduling. The approach for multiple scalable videos [MTS07] is also proposed to adapt the streams to the available transmission resources with graceful quality degradation.

- **Dynamic Caching**

A subjective test environment [MTS08] is implemented with a well designed man-machine interface to compare the achievable user satisfaction for two serving modes: delayed playout starting from the requested point and immediate playout with a deviation of the starting point. Based on the subjective test results [STS07, MTS08] that users prefer the latter mode, videos are proposed to be divided into variable length segments with prefix. A novel iterative merging process is proposed to generate the dynamic segment structure according to the popularity of video content and the optimal serving mode [LTS08, TSMLed]. Furthermore, a general two level cache structure is also developed, which improves the efficiency of caching algorithms by considering both short term and long term popularity and is applicable to all proxy caching approaches.

1.5 Dissertation Organization

The rest of the dissertation is arranged as follows. In Chapter 2, a review of the state-of-the-art error resilient schemes for conversational video transmission in mobile networks is given. Then, a proxy-based framework is introduced, including the error recovery schemes for wireless downlink and wireless uplink, individually. The two approaches can be combined for end-to-end mobile communications. Simulation results are presented to show the significant improvements achieved by employing proxies on the base stations. Finally, complexity issues are discussed to show the applicability of the proposed scheme.

In Chapter 3, the contribution of a proxy to resolve the congestion problem in the Internet is investigated. Methods to extract side information for different video coding structures and applications are introduced. Based on the sent along RD side information, optimal rate

shaping and resource allocation approaches are proposed. Both the performance and the cost of the proxy-based approaches are compared and analyzed.

In Chapter 4, the proxy acts as a virtual content server to decrease the initial delay of a Video on Demand (VoD) system. State-of-the-art proxy caching schemes are first reviewed. Afterwards, the design and execution of the subjective test and the corresponding results are reported. Based on that, a novel proxy caching framework and the dynamic segment-prefix structure is proposed. The working principle and implementation of the system is then described in details. Finally, trace-based simulation results are shown and discussed.

This dissertation concludes with Chapter 5, where the main conclusions from all the different lines of research are summarized, and then, several promising directions and related problems for future research are pointed out.

Chapter 2

Error Resilient Conversational Video

In this chapter, the benefits and use case of a proxy for error resiliency and error recovery are studied. A proxy-based infrastructure is designed to improve the error robustness of conversational video transmission over wireless networks. It provides the solution to different scenarios when mobile users are involved in the communication. With the support of proxies, the error recovery becomes much more efficient compared to conventional end-to-end approaches.

2.1 Introduction

With the further evolution of mobile networks, packet-oriented video services are expected to be among the most popular ones and may be the key factor for success. Wireless video applications without real-time constraints (e.g., Multimedia Messaging Service) have been successfully introduced to the market. However, conversational video communication over packet-switched wireless networks, which are characterized by very low delay requirements, remains challenging.

Digital video compression significantly decreases the data rate for video content by exploring the spatial and temporal redundancy among video frames, which leads to high sensitivity of compressed video to transmission errors. If video packets are lost or corrupted during transmission, the video quality degrades even with error concealment. Moreover, the mismatch of the reference frame(s) at encoder and decoder leads to error propagation both in time and space. Hence, an error resilient transmission scheme is essential to achieve good quality in a wireless video communication system.

As wireless networks are typically much more error-prone and unstable than wired networks, the problem of real-time video transmission over wireless networks is addressed in this chapter. In this work, an adaptive frame dependency management strategy for video telephony applications is proposed where mobile users are participating in the conversation as

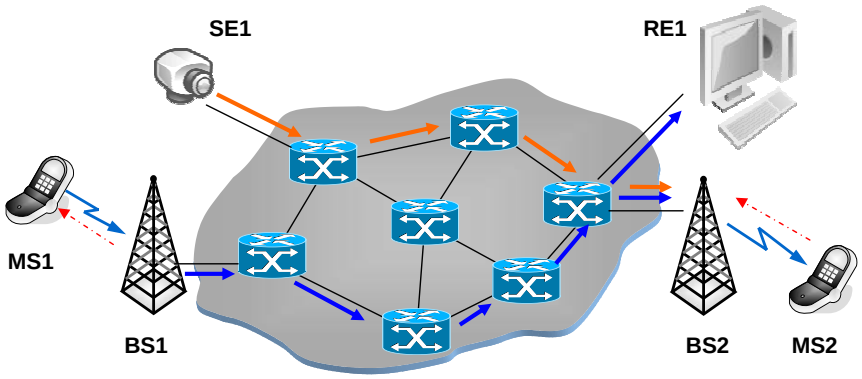


Figure 2.1: Mobile video telephony scenario

shown in Fig. 2.1. In the following, it is assumed that the base stations can send feedback about lost packets to the sender and can retransmit lost packets to the receiver. Another assumption is that the decoder has enough computational resources to decode retransmitted slices fast enough to use them to stop error propagation. If the transmission involves only a wireless downlink (SE1 to MS2 in Fig. 2.1), the downlink packet loss is handled by using Fixed-Distance Reference Picture Selection (FDRPS) in combination with a proxy-based retransmission of lost packets. The reference frame to be used for prediction is determined as a function of the Roud-Trip-Time (RTT) between the base station (BS2) and the receiver (MS2) on the downlink. If the transmission involves only a wireless uplink (MS1 to RE1 in Fig. 2.1), feedback about lost packets is sent from the base station (BS1) to the sender (MS1), so that the encoder can use this information to predict the next frame to be encoded from the video parts which are not negatively acknowledged. If both the sender and the receiver are located in mobile networks (MS1 and MS2) and hence both a wireless uplink and a wireless downlink are involved in the end-to-end communication, the two above mentioned approaches are combined. Feedback information between the mobile users and co-located base stations (BS1 and BS2) significantly decreases the feedback delay compared to the case where feedback goes directly from the receiver (MS2) to the sender (MS1), and thus greatly improves the efficiency of error recovery.

The remainder of this chapter is organized as follows. Section 2.2 introduces selected state-of-the-art error robustness schemes in detail as some of them will be later used for performance comparison. In Section 2.3, the proposed framework for error robust video telephony in mobile networks is stepwise described. Section 2.4 presents the simulation results that show the improvements achieved by the proposed scheme compared to the reference schemes described in Section 2.2. The complexity of all schemes is analyzed and compared in Section 2.5, followed with a short summary of this chapter in Section 2.6.

2.2 State-of-the-art

Numerous studies have been performed to improve the error resiliency for video transmission over lossy channels. Overview of error resilient video transmission strategies can be found, for instance, in [WZ98, WWWK00], where [WWWK00] shows the schemes which have considered the constraints of real-time video communication. Error resiliency tools defined for H.264/AVC are described in [KXMP06] (e.g., Redundant Slicing and Flexible Macroblock Ordering (FMO) [BRCG07]). Error concealment [WHV+02, BGMO03, PKL03, HCLL05] is the simplest way to decrease the effect of packet losses during transmission, where only the decoder is involved in. However, simple error concealment schemes can not provide satisfactory quality, while others have very high complexity. Therefore, more sophisticated error resiliency schemes are needed. In this section, the state-of-the-art approaches for the robust transmission of time-critical video applications are briefly reviewed. Some of them, used for comparison in Section 2.4, are introduced in more details.

2.2.1 Error Resilience without Feedback

In many cases, the transmission channel is unidirectional and no back channel is available, which means that the channel and the corresponding status of the receiver are unknown to the sender. To improve the error robustness of the compressed video, the sender has to estimate the instantaneous status of the receiver with the given statistic information of the transmission channel. Based on the estimation, some protections can be added during the encoding or packetization by the sender to shield the data from error or to improve the capability of recovering from error. INTRA macroblock (MB) update [WLSC98, LV00, Sto02] encodes selective macroblocks in INTRA mode to improve the error resiliency by cutting off the temporal dependency. RD-optimized mode selection, e.g., in [ZRR00, SFG00, YR03, SHW03, YR07] further improves the efficiency of mode selection. The multiple state encoding scheme in [Apo01] takes the advantage of Multiple Description Coding (MDC) and path diversity to improve the error robustness of video transmission over lossy networks. The effective packet loss probability is decreased by sending multiple descriptions over two or more different transmission paths with different characteristics. The error resiliency of video applications can be improved not only by optimal source coding but also by channel coding. Forward Error Correction (FEC) coding introduces redundant packets so that packet erasures can be detected and corrected by the receiver. Packet level Reed Solomon (RS) codes described in [TTM06, ZEP+06, DMM05] are widely used in packetized data transmission over the Internet. Assume that each video frame is transmitted in k packets, n packets are generated using RS code. As long as any k out of n packets are correctly received, this video frame can be reconstructed without error. Unequal error protection assigns more protection bits to packets with high importance. For instance, in [HSLG99], the protection ratio differs for

packets belonging to different layers. A comparison between MDC and FEC is investigated in [SLVM06] and the conclusion is drawn that both have their advantages and drawbacks and the choice must be driven by the given design constraints and priorities. In the following, two of the above mentioned techniques used for comparison are described in more details.

2.2.1.1 Random INTRA Macroblock Update

As mentioned in the introduction, the sensitivity of a video stream is mainly due to the high temporal dependency. If the dependency between frames is reduced or the reference frames at encoder and decoder are resynchronized, the mismatch between the two can be decreased or removed. The temporal error propagation can be stopped if INTRA-coded pictures or MBs are inserted periodically. An INTRA frame normally has a much larger size than INTER frames and thus leads to a big fluctuation in bitrate, which is not suitable for real-time transmission. When the statistics of the transmission channel are approximately known to the encoder, a better approach is to encode a certain percentage of MBs in every frame in INTRA mode to stop the error propagation. Random placement of the INTRA MBs has been shown to be efficient. Moreover, the Random INTRA MB Update (RIMU) [Sto02] approach has been integrated into almost all reference software implementations of standard codecs and is widely used, especially when there is no feedback information available. However, without accurate information about the channel statistics, the efficiency of RIMU is limited, which is particularly true if the packet loss rate changes rapidly over a wide range.

2.2.1.2 RD-optimized Mode Decision

As mentioned above, INTRA coded MBs are independently decodeable by cutting off the temporal dependency from previous frames. In RD-optimized mode decision, the selection of INTER/INTRA mode is determined using a Lagrangian cost function:

$$J = arg \min_{o \in O}(D(o) + \lambda R(o)), \tag{2.1}$$

where $R(o)$ is the number of bits generated for the current MB when encoded with mode o and $D(o)$ is the corresponding distortion (e.g., MSE) associated with this mode. O is the set of available coding modes. Conventionally, $D(o)$ considers only the encoding distortion caused by quantization, which achieves an optimal RD performance only when the transmission is error free.

In the presence of packet loss, the total distortion is composed of both the encoding distortion and transmission distortion [SFG00]:

$$D_{et}(o) = D_e(o) + D_t(o), \tag{2.2}$$

where $D_e(o)$ and $D_t(o)$ represent the encoding and transmission distortion, respectively. The encoding distortion $D_e(o)$ is the distortion introduced by quantization while the transmission

distortion $D_t(o)$ refers to the distortion that is observed in the presence of transmission errors after error concealment. Generally speaking, the "+" operation in (2.2) represents a joint consideration of the two types of distortion rather than a mathematical summation operation. However, as discussed in [SFG00], a good approximation of the total distortion $D_{et}(o)$ is obtained by summing up the encoding and the transmission distortion. This is because the two distortions show only little correlation over a wide range of source rates and loss probabilities. By replacing $D(o)$ in (2.1) by $D_{et}(o)$ from (2.2), the cost function that is suitable to make an error robust mode decision for the transmission of compressed video over lossy networks is obtained.

A Multi-Decoder Distortion Estimation (MDDE) extension of H.264/AVC is proposed in [SHW03], where a powerful yet computationally demanding method is introduced to estimate the expected reconstruction distortion. K copies of random channels with the same statistics as assumed for the real transmission channel are employed at the encoder to simulate the expected distortion at the decoder. If the K channels are identically and independently distributed, then as $K \to \infty$, it follows by the strong law of large numbers that

$$\frac{1}{K}\sum_{k=1}^{K} D_{et}^{k}(o) = E(D_{et}(o)). \tag{2.3}$$

If K is not large enough, the estimation in (2.3) will be inaccurate and affect the distortion estimation for later frames. However, increasing K adds considerable computational complexity at the encoder.

2.2.2 Feedback based Error Recovery

When the receiver and the sender communicate bi-directionally, precise channel and receiver status can be reported to the sender. Therefore, more efficient error protection or recovery schemes can be employed based on that. An overview of feedback based error control methods is given in [GF99]. Reference Picture Selection (RPS) schemes [FNI96, ITU96, WFS00, LFG02] use feedback about lost or correctly received packets to restrict the prediction from those image areas that have been successfully decoded. Alternative ways to exploit feedback information from the receiver have also been proposed. Recovery from Error Spread using Continuous Updates (RESCU) [RJ00] changes the frame dependencies in a video sequence such that a retransmission of lost information can be used for error recovery with the help of Accelerated Retroactive Decoding (ARD) [Gha96]. Error tracking [SFG97, TS04], for instance, uses feedback about lost packets at the sender to reconstruct the error propagation. Corrupted areas are encoded in INTRA mode which leads to error recovery without introducing additional delay. The achievable performance of error tracking is mainly a function of the RTT between the sender and the receiver. Feedback information can also be used to further

improve the effectiveness of RD-optimized mode decision as show in [WFS00, LFG02, LC04]. In the following, some comparison schemes are described in more details.

2.2.2.1 Feedback-based Multi-Decoder Distortion Estimation (F-MDDE)

As introduced in Section 2.2.1.2, the MDDE approach proposed in [SHW03] assumes that the encoder does not receive feedback about successfully received or lost packets from the decoder. In some particular application scenarios, a feedback channel is available and the feedback information can help the encoder to update the distortion estimation status. Here the MDDE scheme is extended to work with feedback information.

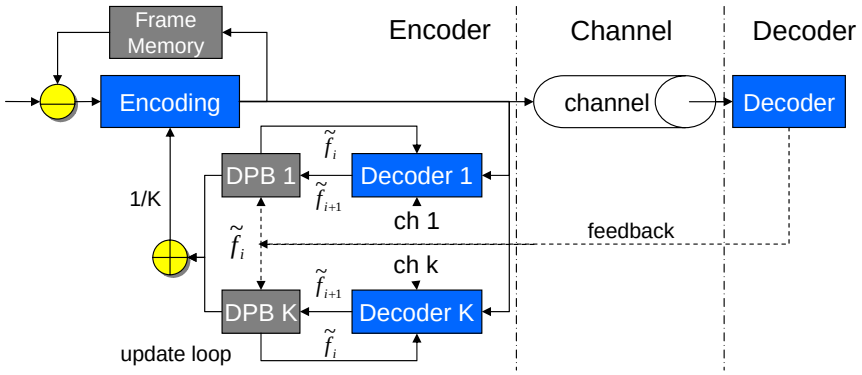


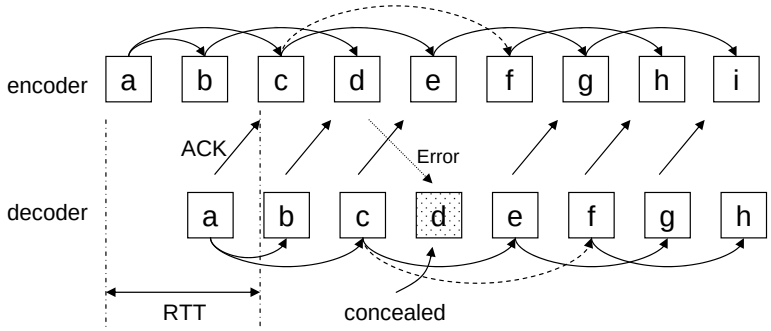
Figure 2.2: Multi-decoder distortion estimation with feedback

Fig. 2.2 shows the structure of the Feedback-based MDDE (F-MDDE) as used for comparison in this work. When a NACK for the lost packet (slice s of frame i) is received by the encoder, it first performs error concealment for slice s in frame i and replaces frame i in all K Decoding Picture Buffers (DPBs) with the concealed frame $\tilde{f}_i$. Let this updated i^{th} frame pass again through K random channels, we get the updated version of the estimated frame $\tilde{f}_{i+1}$ and store it back in the DPBs. This update procedure continues until the most recent frame in the DPBs has also been updated. The number of update loops is determined by the RTT between the sender and the receiver. If the RTT is equal to N frame intervals, such kind of update should be run for $N-1$ times.

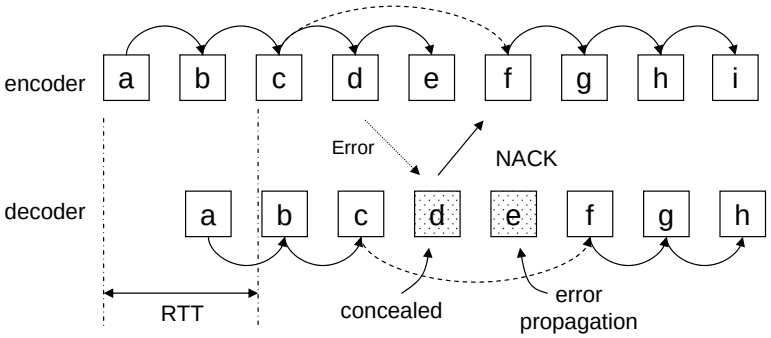
2.2.2.2 NEWPRED

NEWPRED [FNI96] [ITU96] uses the feedback about lost packets or correctly received packets to prevent the prediction from those image areas that have been corrupted.

In ACK-based NEWPRED (*A-NEWPRED*), only those frames that have been positively acknowledged are used as a reference frame. Fig. 2.3(a) illustrates A-NEWPRED for a RTT of two frame intervals. If frame d in Fig. 2.3(a) is corrupted and not acknowledged, frame f will then use the acknowledged frame earlier than d as the reference frame, which is frame c



(a) ACK based NEWPRED (*A-NEWPRED*)



(b) NACK based NEWPRED (*N-NEWPRED*)

Figure 2.3: NEWPRED for a RTT of 2 frame intervals

in this example. The following frames g and h use e and f as reference and can be correctly decoded. The coding efficiency of *A-NEWPRED* is determined by the RTT between the encoder and decoder. *A-NEWPRED* achieves the highest coding efficiency for instantaneous feedback, which leads to a prediction from the most recent frame in the error-free case. A large RTT leads to a larger prediction distance and thus lower coding efficiency.

NACK-based NEWPRED (*N-NEWPRED*) uses the most recent frame for motion-compensated prediction in the absence of negative acknowledgments as shown in Fig. 2.3(b). In case NACKs are received, the prediction distance is increased. Assume again that frame d is corrupted during the transmission. It is concealed at the decoder and a NACK for frame d is sent back to the encoder as shown in Fig. 2.3(b). The next frame f to be encoded switches its reference to the last successfully decoded frame c, because d is corrupted and the error also propagates to frame e. Frame g uses frame f as the reference frame. If frame f is successfully decoded as assumed in Fig. 2.3(b), the error propagation caused by the loss of frame d is terminated. In case frame f is also negatively acknowledged, the error would propagate to frame g. Frame h would then again use c as the reference. The coding efficiency of *N-NEWPRED* is higher than that of *A-NEWPRED* when no error occurs, because in this case always the most recent frame is used as the reference. The coding efficiency decreases with increasing packet loss rate. Furthermore, when the RTT is large, many frames suffer from error propagation, which

degrades the video quality significantly.

2.2.2.3 RESCU

The main idea of RESCU [RJ00] is to change the frame dependencies in a video sequence such that a retransmission of lost information can be used for error recovery despite the low delay requirements of real-time video communication. In RESCU, every n^{th} frame (frame a, d, g in Fig. 2.4) is a so called periodic frame that references a previous periodic frame which is n frame intervals away. Frames in between two consecutive periodic frames predict only from their immediately preceding periodic frame. If a non-periodic frame is lost, only this frame itself is affected. As shown in Fig. 2.4, if the periodic frame d is lost, the displayed frames d to f are affected by error concealment and error propagation. A retransmission is triggered by the NACK sent to the sender. If the retransmission arrives before the time instant when the next periodic frame g is to be decoded, frame d will be immediately decoded, which produces an error free reference for frame g. Please note that, in order to make RESCU work, the receiver has to be able to decode retransmitted information faster than real-time.

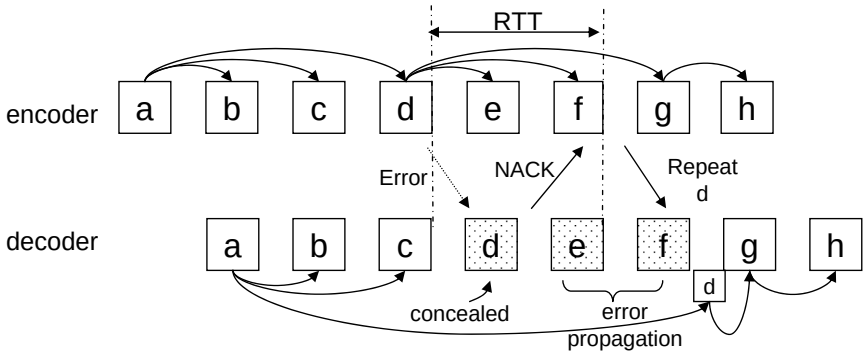


Figure 2.4: RESCU for a RTT of 2 frames interval

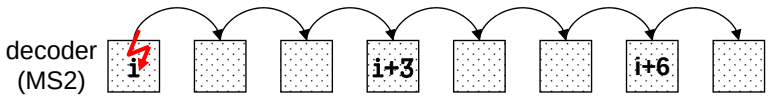
2.3 Proxy-based Reference Picture Selection

In this section, proxy-based error resilient transmission of conversational video is proposed for the cases when either the receiver is a mobile user, or the sender is a mobile user, or both are. It is assumed that during encoding, when no transmission error is reported, the encoder always uses only one previous frame as a reference. Feedback packets are assumed to be strongly protected and be error free.

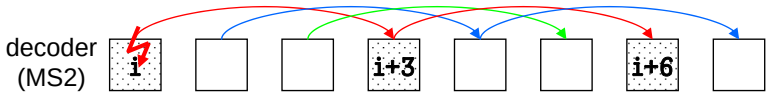
2.3.1 Downlink Error Recovery

The scenario where the receiver is in a mobile network and the sender is located in a wired network is first considered. In this case, the main target is to improve the error robustness

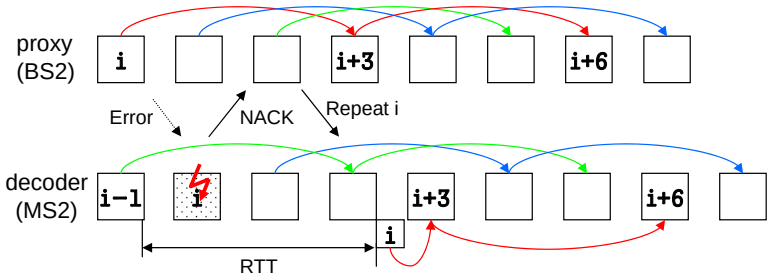
on the wireless downlink.



(a) Error propagation for FDRPS with distance 1.



(b) Error propagation for FDRPS with distance 3.



(c) FDRPS with distance 3 using proxy based retransmission and accelerated decoding (FDRPS+ARQ)

Figure 2.5: Error propagation for FDRPS when frame i is corrupted

Retransmission of lost information triggered by feedback from the receiver is considered to be one of the most suitable error resiliency approaches for traditional data communication applications. The big advantage of feedback-based retransmission is its inherent adaptiveness to varying loss rates. Retransmissions are only triggered if the information is actually lost. The overhead encountered is therefore a direct function of loss rate and the sender does not need to receive or estimate information about the expected channel condition. For bi-directional conversational services like video telephony, however, the benefit of packet retransmission is limited because of the stringent one-way latency requirement which is typically in the range of 150∼250ms.

For conversational video, if the current frame is predicted from the most recent frame as shown in Fig. 2.5(a), the retransmission of a lost packet will typically arrive too late at the receiver to be used for the decoding process because of the strict delay constraints for conversational video. Moreover, with the typical IPPP... structure, if frame i is corrupted by packet loss, the error will propagate to the following frames, shown as the dotted boxes in Fig. 2.5(a). This prediction structure is referred as Fixed-Distance Reference Picture Selection (FDRPS) with distance 1 in the following. The error recovery can be facilitated by adjusting the prediction distance N in number of frames between the reference frame and the encoding frame to match the RTT on the downlink as shown in Fig. 2.5(b). It is assumed that the RTT in Fig. 2.5(b) corresponds to 3 frame intervals and hence frame $i+3$ uses frame i as

its reference. If the same loss as before happens, only one of the N prediction groups is affected. In addition, the increased distance to the reference frame gives additional time for the retransmission of lost packets. Successfully retransmitted packets can then be used to stop error propagation as will be explained in the following. In Fig. 2.5(c), when frame i is corrupted during the transmission, it is played out after applying error concealment. The next $N-1$ frames belong to the other prediction groups, which are assumed to have error free reference frames and can be displayed without impairments. As long as N is larger than the downlink RTT in number of frames, the first retransmission of the lost frame i can be accomplished before decoding and displaying frame $i+N$. If the retransmission succeeds, frame i is decoded using ARD and the error free reconstructed frame i is put into the DPB, which now provides an error free reference picture for frame $i+N$ belonging to the same prediction group. If the first retransmission fails, frame $i+N$ is shown also with error concealment at its display time and the second retransmission is triggered. In this case, when frame i is received error free, both frame i and $i+N$ have to be re-decoded to make the reference picture of frame $i+2N$ error free. At a packet loss rate of 10%, two retransmissions already reduce the residual packet loss rate to 0.1%, which is quite small and leads to limited impairments. In such an extreme situation of multiple successive losses of a packet and its retransmissions, a feedback message can be sent directly to the encoder to ask for a resynchronization, which will stop the error propagation. However, the end-to-end RTT determines how fast the error recovery can happen.

As mentioned in Section 1.3.1 conversational video is usually encoded with slice structure in order to improve the error resiliency. One video frame normally consists of several slices, which can be independently decoded. Therefore, one slice is also encapsulated into one packet in this chapter. When one or several packets from frame i are corrupted, instead of retransmitting the whole frame, only those affected packets have to be resent.

As can be seen from Fig. 2.5(c), the RTT on the downlink is a key factor influencing the efficiency of the proposed error recovery scheme. Because the encoder can be left out from this procedure, any network node able to perform the retransmission can be used. In the considered scenario, the base station BS2 in Fig. 2.1 is the closest point to the mobile receiver. Running the retransmission proxy on this base station or at least as close as possible to it leads to the minimum round trip delay. Fig. 2.6 shows the required bitrate as a function of the prediction distance N for two standard video test sequences, *Foreman* and *Salesman*. All points on the curves in Fig. 2.6 are obtained with the same quantization parameter (QP) of 28 during the encoding, which means that the PSNR values on the same curve are very similar but not exactly identical. Although the coding efficiency in the error-free case is decreased because of the increased prediction distance, it will be shown in Section 2.4 that even for low loss rates this effect is compensated by the greatly improved error-resilience.

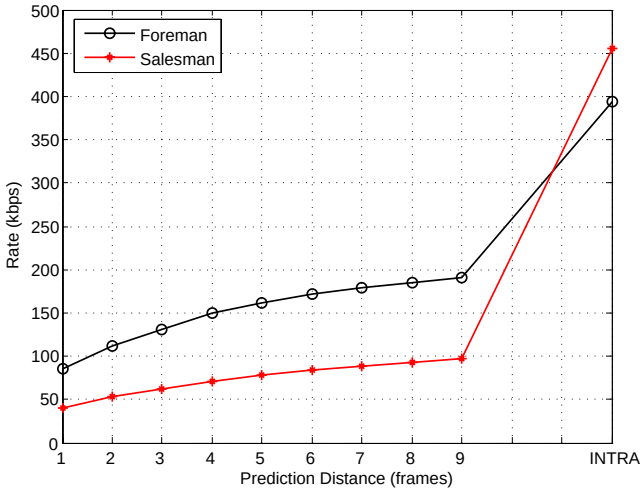


Figure 2.6: Coding efficiency as a function of the prediction distance for two test sequences using H.264/AVC with QCIF @ 15Hz

The proposed FDRPS+ARQ approach for downlink error recovery has some similarity with RESCU [RJ00] as described in Section 2.2.2.3. However, in comparison to RESCU, the proposed approach considers proxy-based retransmission of lost information on the downlink and therefore avoids low coding efficiency because of the large end-to-end delay encountered when using RESCU. Also, different to RESCU, it does not matter which frame is affected by packet loss in the proposed scheme. In RESCU, if the periodic frame is corrupted, all frames which depend on the periodic frame and will be displayed before the retransmission succeeds are affected. For the FDRPS+ARQ approach, failed packets on the downlink will be retransmitted for at most two times. If the retransmission arrives in time, the error free frame is reconstructed using accelerated decoding and error propagation is stopped. In case both retransmissions get lost, only a subsequence of the video will be affected by error propagation. This allows stopping the display of this subsequence and asking for a resynchronization frame. When comparing these two schemes, RESCU favors a channel with low packet loss rate and the FDRPS+ARQ approach performs better at higher packet loss rates, which will also be proved by the simulation results reported in Section 2.4.6.

2.3.2 Uplink Error Recovery

The error recovery strategy described in the last section is designed for packet losses on the downlink. If the sender is in a mobile network (MS1 in Fig. 2.1) and the receiver (RE1) is connected to a wired network, error recovery from packet losses on the uplink is more important.

As the sender here is also the encoder, similar approaches such as NACK-based NEW-

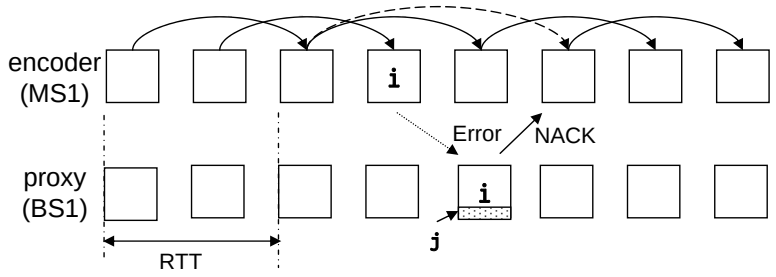
PRED [FNI96] (described in detail as N-NEWPRED in Section 2.2.2.2) can be used for the error robustness on the uplink. As long as no NACK is received by the sender, N-NEWPRED uses the most recent frame as the reference frame. This frame, however, might be corrupted during the transmission or affected by error propagation as explained in Section 2.2.2.2. In order to perform the prediction always from an error free frame, a small change is made here compared to the original N-NEWPRED scheme. Instead of using the immediately preceding frame as the reference frame in the absence of NACKs, prediction is made from the frame with a distance to the current frame that corresponds to the RTT on the uplink. Therefore, RTT is also a key factor for the uplink error recovery scheme. It is shown in Section 2.4.6 that this modification leads to improved performance compared with the original NEWPRED. Again, the proxy is proposed to be setup on the base station (BS1 in Fig. 2.1). This proxy is responsible for checking the integrity of video packets sent over the uplink and returning corresponding NACKs to the sender. If a video packet is not successfully received by BS1, it returns a NACK to MS1. When the NACK is received by the encoder, the following three possible actions can be taken by the encoder to stop the error propagation.

2.3.2.1 Frame Level RPS (FLRPS)

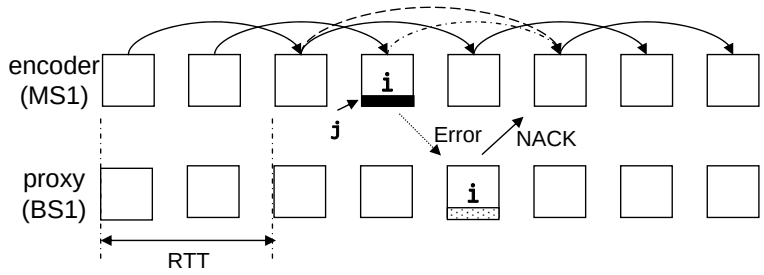
In case a packet is lost on the uplink, BS1 returns a NACK to MS1 and the sender reacts to the NACK by changing the reference frame for the next frame to be encoded. The frame with lost packet(s) is not used as a reference for any following frames. As illustrated with the dashed arrow in Fig. 2.7(a), frame $i + 2$ is predicted from the most recent error-free frame earlier than frame i, which is frame $i - 1$ in the example. If frame $i - 1$ is also corrupted, the prediction turns to frame $i - 2$, and so on and so forth. As a result, this packet loss on the uplink will only affect one single frame at the receiver, which will be later displayed using error concealment.

2.3.2.2 Slice Level RPS without Error Concealment (SLRPS)

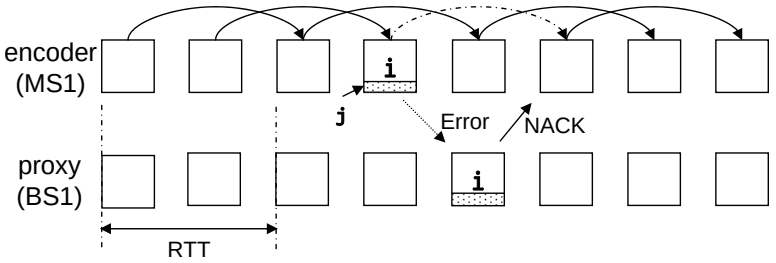
FLRPS stops error propagation by always using an error free frame as the reference frame. However, a single packet loss in a frame leads to an increased prediction distance, which degrades the coding efficiency. In SLRPS the dynamic RPS is applied on the slice level. As shown in Fig. 2.7(b), when one packet (slice j) in frame i is lost on the uplink, the concealed frame i is still used as a potential reference frame, while excluding the missing area. Additionally, frame $i - 1$ also serves as a reference frame. Because of the strong correlation between temporally consecutive frames, most parts of frame $i+2$ are still predicted from frame i, and only a small part of the frame takes a corresponding part in frame $i - 1$ as reference, illustrated with the dash-dotted arrow and dashed arrow in Fig. 2.7(b), respectively. In case



(a) Frame level RPS (FLRPS)



(b) Slice level RPS without error concealment (SLRPS)



(c) Slice level RPS with concealment (SLRPSEC)

Figure 2.7: Adaptive RPS triggered by feedback from the base station to the sender

more than half of the packets in frame i and frame $i-1$ are lost, the prediction will also include frame $i-2$.

2.3.2.3 Slice Level RPS with Error Concealment (SLRPSEC)

A small alternation in SLRPS is to perform error concealment for the corrupted frame as soon as the encoder learns about the loss. This is possible due to the proposed modification of N-NEWPRED, which adjusts the regular prediction distance to the RTT of the uplink. The concealment is performed before encoding the next frame which predicts from the corrupted frame. If this strategy is applied, again just one reference frame is needed. The error concealment scheme should be the same as that used at the decoder. Fig. 2.7(c) represents the error recovery performed when packet j of frame i is lost. The error caused by packet loss can be concealed temporally, spatially or using a combination of both. Frame $i+2$ is then predicted from the concealed frame i, shown with the dash-dotted arrow in Fig. 2.7(c).

The advantages of this approach are the lower motion estimation complexity and the reduced memory requirement compared to SLRPS. However, additional complexity is introduced by the error concealment.

2.3.3 Combination

In the previous two sections, the proposals for error recovery from packet losses on the uplink and downlink have been presented, respectively. In this section, the case when both the sender and the receiver are in mobile networks is considered. The two error recovery schemes can be employed individually on the uplink and downlink. From an end-to-end transmission point of view, they cooperate and complement each other well. In the following, this combined framework is referred to as the Proxy-based Reference Picture Selection (PRPS) scheme, which provides an efficient and error robust solution for the end-to-end conversational video application for mobile users.

Let us assume that in Fig. 2.1, MS1 is the sender and MS2 is the receiver. The video packets are sent uplink to base station BS1 and from there to BS2. BS2 forwards the video stream downlink to the receiver MS2. It is also assumed that during the connection establishment process, the RTT on the downlink is signaled to the encoder. If the RTT on the uplink is greater than or equal to the RTT on the downlink, no adjustment is needed for the adaptive RPS scheme used on the uplink. Otherwise, the prediction distance is adjusted to the RTT that is observed on the downlink. In other words, the larger RTT determines the default prediction distance being used by the encoder.

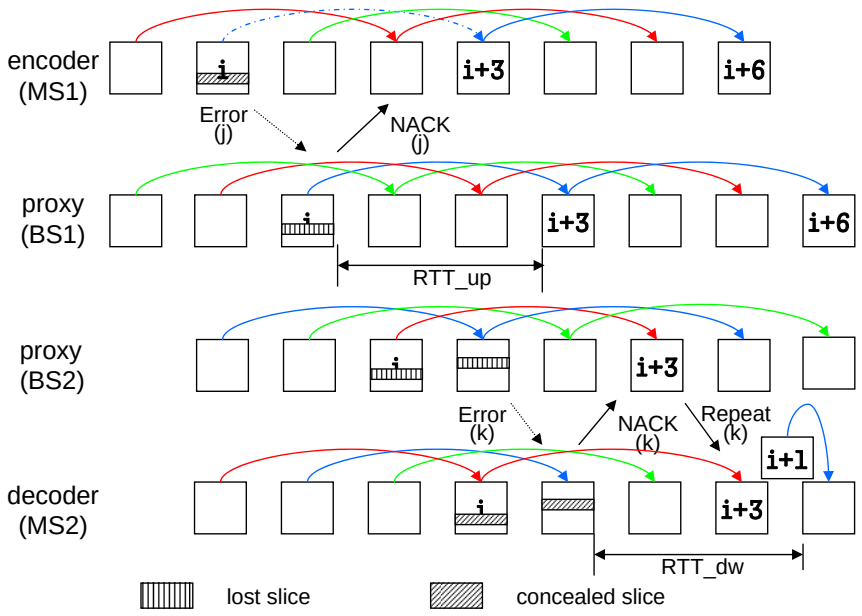


Figure 2.8: Error robust mobile video telephony using the proposed PRPS framework.

Fig. 2.8 shows an example that illustrates how both parts work together when SLRPSEC is used for the uplink error recovery. The RTT on the downlink is assumed to be larger than the RTT on the uplink and corresponds to 3 frame intervals. Hence, the default prediction distance is selected to be 3. This is illustrated in the top row of Fig. 2.8. When packet j of frame i gets lost on the uplink, the encoder performs error concealment for frame i and uses it as the reference for frame i+3. This lost slice will be concealed and displayed at the decoder (see bottom row of Fig. 2.8). The lost packets on the uplink from MS1 to BS1 do not arrive at BS2, which saves some transmission rate on the downlink and can be used for retransmission of lost packets. When slice k in frame i+1 gets lost on the downlink, the NACK from MS2 is received by BS2 and slice k is retransmitted before sending frame i+4. Frame i+1 is then re-decoded without error and is used as the reference for frame i+4. As frame i+3 is encoded with the concealed frame i as the reference frame, at the decoder using the identical reference frame leads to a resynchronization between the encoder and decoder. Please note, here only the SLRPSEC is used as an example on the uplink, however, the other two schemes FLRPS and SLRPS can also be used in the proposed framework.

2.4 Experimental Results

The purpose of this section is to evaluate the performance of the proposed framework under different network conditions. First, comprehensive results for some comparison schemes introduced in Section 2.2 are investigated. Then they are compared with the proposed proxy-based RPS (PRPS) framework.

H.264/AVC test software version JM 11.0 [HHIb] is used as the video codec. The first 300 frames of the test sequences *Foreman* and *Salesman* at QCIF resolution are encoded at 15fps with an I-P-P-P... structure. A slice is defined to correspond to one row of MBs and put into one packet for transmission. The default error concealment techniques defined in JM 11.0 [WHV+02] are used at the receiver for display. In case a whole frame is lost, it is concealed by copying the previous reconstructed frame. The maximum RTT on the uplink and downlink is assumed to be 200ms, which corresponds to a prediction distance of 3 frames (RTT=3 frames) between the current frame and the reference frame at a frame rate of 15fps. The end-to-end RTT including the wireless and wireline networks is assumed to be 400ms, which corresponds to 6 frame intervals. A random packet loss channel model and a burst packet loss channel using the two-state Gilbert-Elliott model are employed in the simulation. Without specification, the average burst length for the burst loss model is set to be 5 packets and n% packet loss means n% packet loss on the uplink and n% packet loss on the downlink. For each simulation, 100 different channel realizations are tested and. As mentioned in Section 1.3.2, the averaged PSNR value of the luminance component as reconstructed and displayed at the receiver is reported.

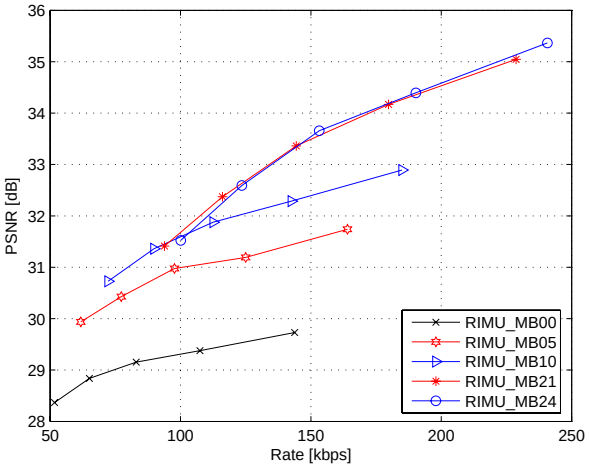


Figure 2.9: RD performance of RIMU for the *Foreman* sequence and 1% random packet loss in both uplink and downlink

2.4.1 RIMU

As described in Section 2.2.1.1, RIMU is widely used to improve the error resilience of video transmission when no feedback channel is available. The performance strongly depends on the accurate estimation of the packet loss rate on the transmission channel. An improper INTRA MB rate leads to very low performance. Fig. 2.9 shows the RD performance of RIMU over a 1% random loss channel for different numbers of INTRA updated MBs per frame. *RIMU_MB00* shows a special case when the encoding mode of each MB is determined by the RD cost function in the codec, which considers only quantization error. Without any protection, very poor performance is obtained even at such a low packet loss rate. *RIMU_MBn* shows the performance when n randomly selected MBs in each frame are encoded in INTRA mode. As illustrated in the figure, the optimal update rate changes when the target bit rate varies. As the optimal n depends not only on the packet loss rate, but also on the video content and the target bit rate, it is hard to determine. Therefore, the highest achievable quality of RIMU at each loss rate is used in Section 2.4.6, which forms an upper bound on the performance of RIMU. This upper bound is obtained by running through all possible update rates and picking the one which leads to the best performance.

2.4.2 F-MDDE

In this section, the performance of the original MDDE (implemented in JM 11.0) introduced in Section 2.2.1.2 is compared to the F-MDDE scheme introduced in Section 2.2.2.1. According to [SHW03], when the number of decoders at the encoder K is equal to 30, the estimation of the distortion is already quite accurate while the computational complexity is still reasonable.

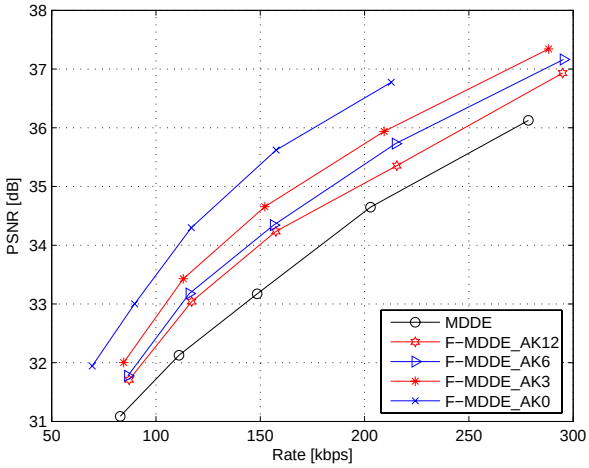
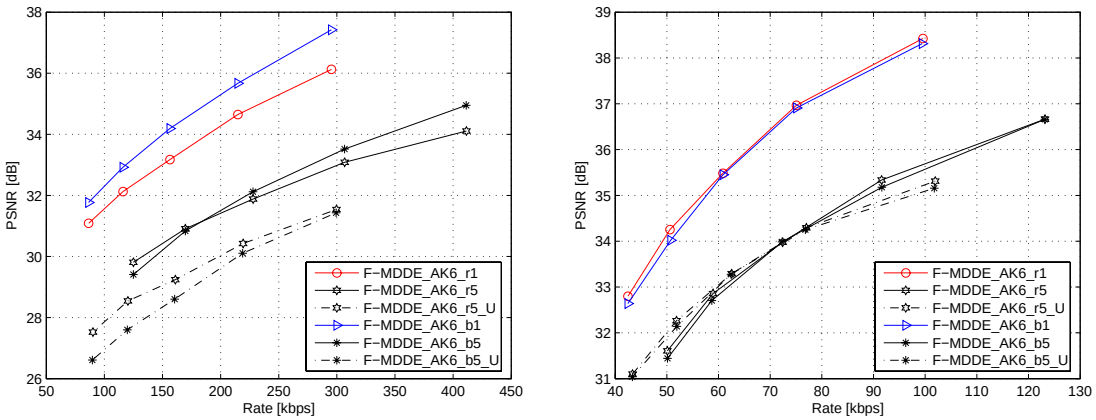


Figure 2.10: RD performance of MDDE and F-MDDE with K=30 for the *Foreman* sequence and 1% random packet loss in both uplink and downlink



(a) Foreman (b) Salesman

Figure 2.11: RD performance of F-MDDE for a RTT of 6 frames for the *Foreman* and *Salesman* sequences

Therefore, K=30 is used for all following simulations. In the F-MDDE implementation, the same RD mode decision as in MDDE is employed. The distortion is re-estimated using updated decoding pictures in the DPB after evaluating the feedback information.

Fig. 2.10 shows the results for MDDE and F-MDDE based RD-optimized mode decision for a channel with 1% random packet loss. *F-MDDE_AKx* represents the F-MDDE with x frames delay of the acknowledgment. The F-MDDE approach outperforms the MDDE by 0.5 dB to 2 dB. The smaller the RTT, the more accurate the distortion estimation and thus the better the performance. When instantaneous feedback is available, the encoder knows exactly the reference frame at the decoder side. This is typically not possible in practice, however, the

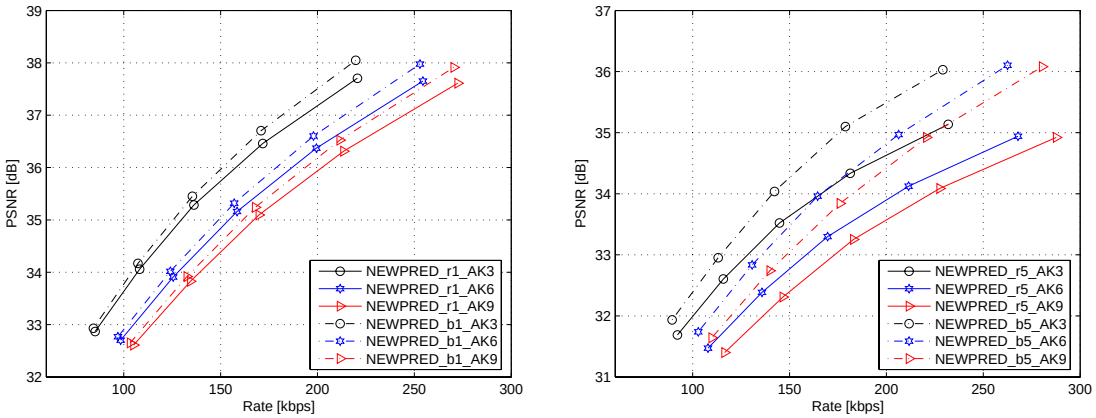
F-MDDE_AK0 curve in Fig. 2.10 gives an upper bound on the performance of F-MDDE. For the same QP used during encoding, a significantly smaller rate at a slightly reduced PSNR is observed when compared to *F-MDDE_AK3*. This is due to the fact that only some of the most severely affected MBs in the current frame are encoded in INTRA mode when there are some impairments in the decoded picture, which significantly increases the coding efficiency. However, the RD-optimized mode decision also selects the INTER mode for those MBs with small distortion, which leads to some error propagation.

Fig. 2.11 shows the performance of F-MDDE for different channel models for a fixed RTT of 6 frames. At 1% packet loss rate, the same performance for the random loss (*F-MDDE_AK6_r1*) and the burst loss channel (*F-MDDE_AK6_b1*) can be observed for the *Salesman* sequence. For the *Foreman* sequence at 1% loss rate, the performance for the burst loss channel is significantly better than for the random loss channel because of the stronger error propagation for random losses.

An important assumption made by MDDE is the correct knowledge of the average packet loss rate on the transmission path. Sometimes this information might not be available or the estimation of it might be wrong. The two dash-dotted curves in Fig. 2.11(a) show the reconstruction quality of the *Foreman* sequence when the assumed packet loss rate is set to be 2% while the real packet loss rate is 5% on the uplink and downlink, respectively. The underestimated packet loss rate leads to fewer INTRA MBs with less bitrate but also much lower PSNR. For the low motion *Salesman* sequence in Fig. 2.11(b), the underestimation is not so critical, because the saved rate can fully compensate the degradation of the video quality when the rate is lower than 75 kbps.

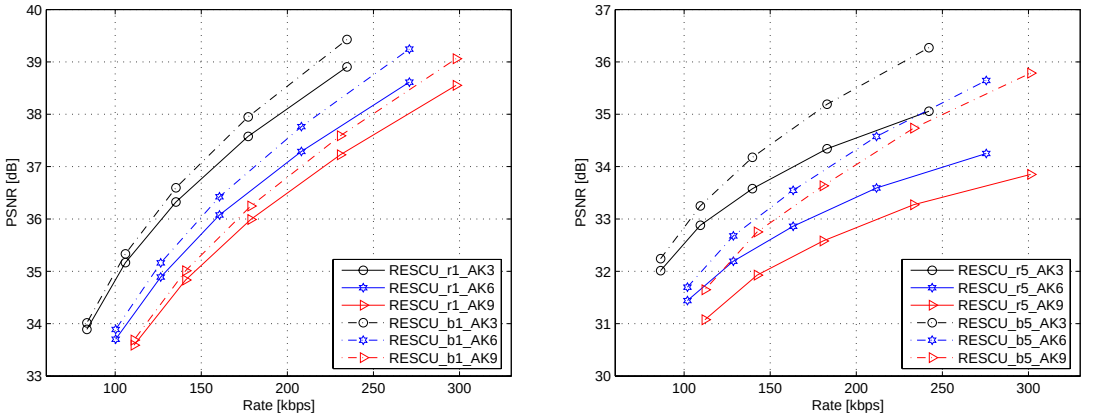
2.4.3 NEWPRED

Here, only the ACK based NEWPRED (A-NEWPRED) is examined, because according to [FNI96], A-NEWPRED and N-NEWPRED have similar performance. Fig. 2.12(b) shows the RD performance curves for different RTTs of A-NEWPRED when it is employed as the error resilience approach for 1% and 5% packet loss channel, respectively. At the same packet loss rate, A-NEWPRED performs better for burst losses (e.g., *NEWPRED_b1_AK6*) than random losses (e.g., *NEWPRED_r1_AK6*) because the implementation here is frame based, where even a single packet loss leads to the switching of the reference frame and a larger prediction distance. Burst packet loss with consecutive packet losses but same number of total lost packets results in a smaller prediction distance on average and thus a better performance. With 5% packet loss rate, the gap is even bigger between the random loss and burst loss. When the same transmission channel is used, the end-to-end RTT dominates the performance of A-NEWPRED. With an end-to-end RTT of 3 frames, *NEWPRED_r1_AK3* achieves about 1.5 dB improvements when compared with *NEWPRED_r1_AK9*, which has a



(a) 1% uplink 1% downlink packet loss (b) 5% uplink 5% downlink packet loss

Figure 2.12: RD performance of NEWPRED for the *Foreman* sequence for different RTTs



(a) 1% uplink 1% downlink packet loss (b) 5% uplink 5% downlink packet loss

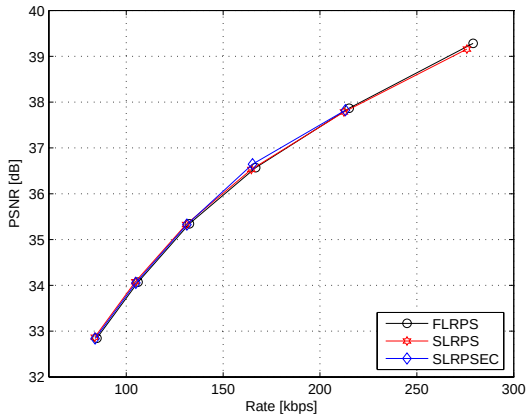
Figure 2.13: RD performance of RESCU for the *Foreman* sequence for different RTTs

much larger RTT.

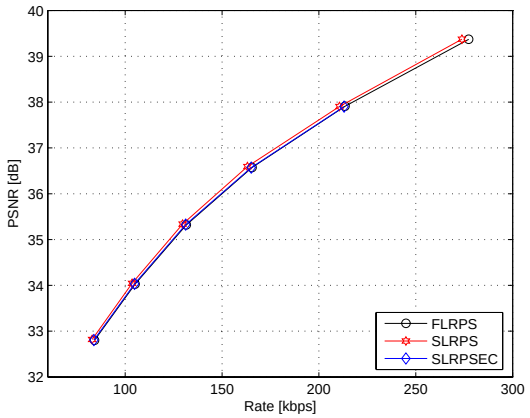
2.4.4 RESCU

As described in Section 2.2.2.3, RESCU works end-to-end between the sender and receiver. When packets belonging to the periodic frame are lost, retransmissions of those lost packets are triggered. However, frames till the next periodic frame will be affected by the error propagation from this periodic frame.

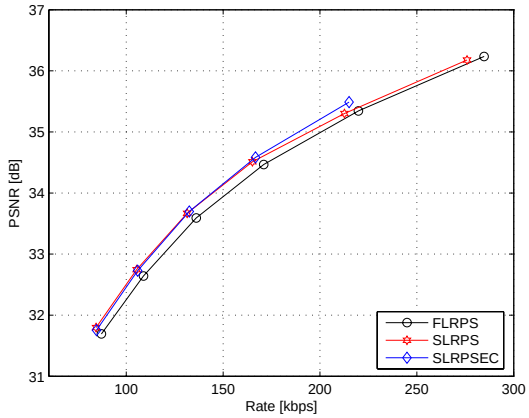
Fig. 2.13 shows the performance of the RESCU approach for the two different channel types. *RESCU_r1_AK3* and *RESCU_b1_AK3* in Fig. 2.13(a) illustrate the performance of RESCU for 1% random and burst packet loss channels, respectively, for a RTT of 3 video



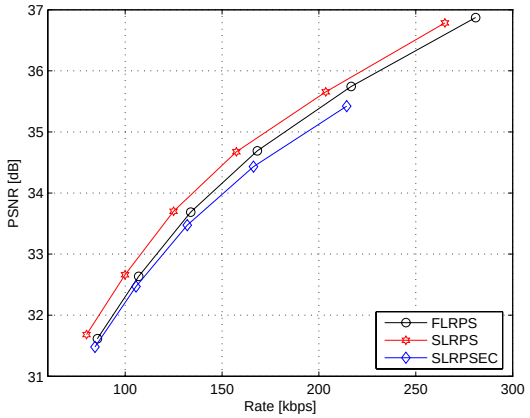
(a) 1% uplink 1% downlink random packet loss



(b) 1% uplink 1% downlink burst packet loss



(c) 5% uplink 5% downlink random packet loss



(d) 5% uplink 5% downlink burst packet loss

Figure 2.14: Performance of the adaptive RPS schemes used for uplink error recovery, *Foreman*, RTT of 3 frames

frames. When the RTT becomes larger, the distance between the two periodic frames also has to be increased. As all non-periodic frames are predicted from the immediately preceding periodic frame, the coding efficiency is therefore degraded. Meanwhile, the error propagation also becomes more severe when a periodic frame is corrupted. Therefore, the larger the RTT, the lower the reconstruction quality at the receiver. At the same packet loss rate, RESCU over burst channel performs better than that over random channel, because fewer frames will be corrupted by the packet loss when the loss is bursty and the reconstruction errors in non-periodic frames do not propagate to later frames. Fig. 2.13(b) shows the performance of RESCU for 5% packet loss. The gap between the curves for different RTTs is larger than in Fig. 2.13(a), which shows that the performance of RESCU degrades significantly at high packet loss rates.

2.4.5 Adaptive RPS in Uplink

In Section 2.3.2, three adaptive RPS approaches for uplink error recovery are introduced. In this section, the performance of these three approaches is compared when combined with the same error recovery scheme on the downlink (FDRPS+ARQ). To keep the error concealment simple at the encoder, the slice at the same spatial position in the previous frame is copied to conceal the lost slice in the current frame. At the decoder, the frame to display uses the standard error concealment scheme in JM 11.0 [WHV+02], while the frame put into the decoder reference picture buffer is concealed using the same concealment approach as that used at the encoder.

Fig. 2.14 illustrates the performance of FLRPS, SLRPS and SLRPSEC at different channel conditions. At low loss rate (1%), it happens very rarely that two consecutive frames are corrupted, which means the prediction distance typically needs to be increased by only one frame. As shown in Fig. 2.6, a one step prediction distance increase leads to a 10-40% rate increase. With only 1% loss in the uplink, at most 1% of the total frames need to reselect their reference pictures, which results in at most 0.4% rate increment. Therefore, the three approaches have almost the same performance in this case. If the wireless channel has 5% random loss, the 2% rate increment can already be seen in the corresponding sub-figure. If the loss is bursty at 5%, performance gaps can be clearly observed among the three schemes. SLRPSEC has the lowest performance in this case because the standard error concealment is less efficient for burst losses.

2.4.6 Proxy-based RPS

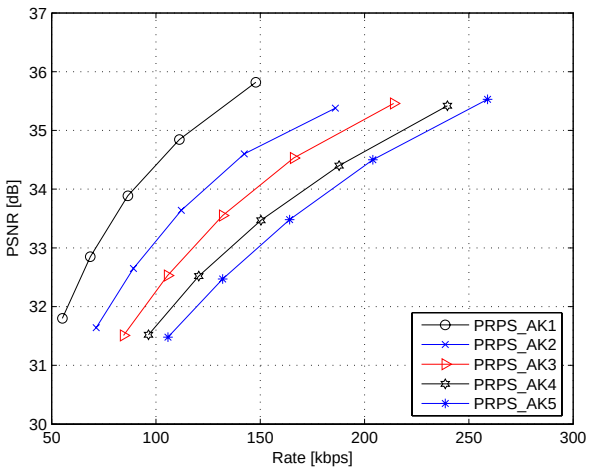
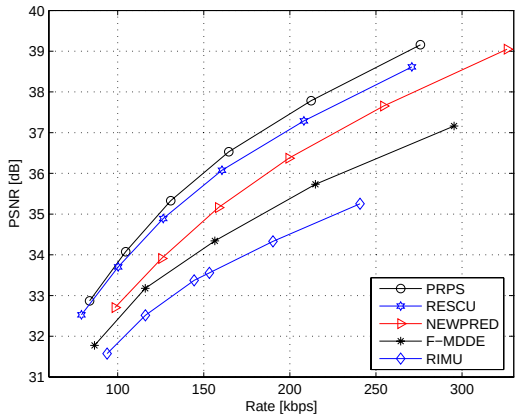
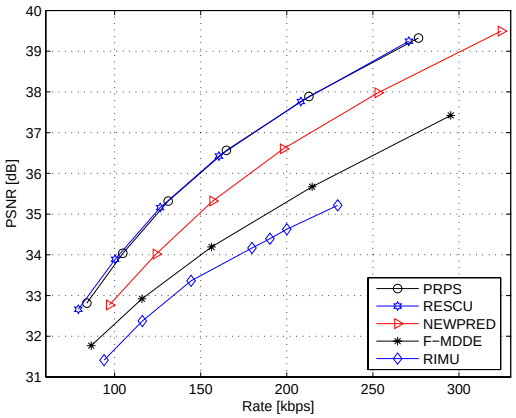


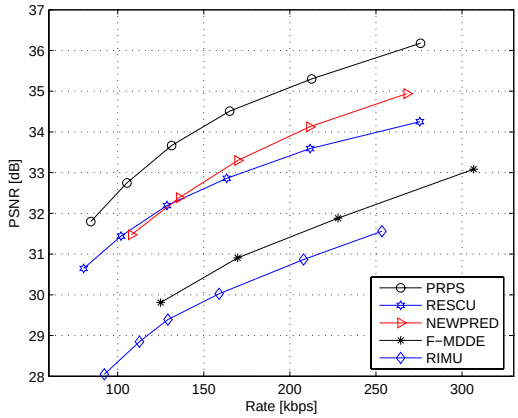
Figure 2.15: RD performance of PRPS as a function of the RTT on the uplink and downlink for a 5% packet loss channel. The mean burst length is 5 packets. The test sequence is *Foreman*.



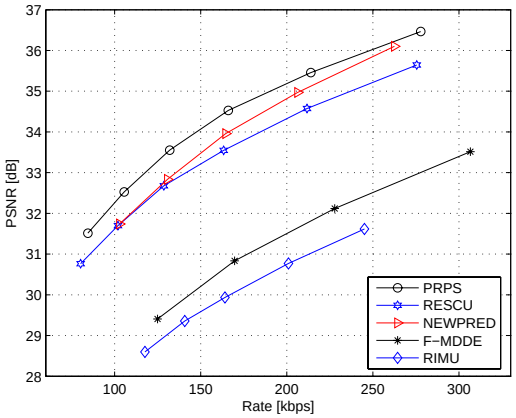
(a) 1% uplink 1% downlink random packet loss



(b) 1% uplink 1% downlink burst packet loss



(c) 5% uplink 5% downlink random packet loss



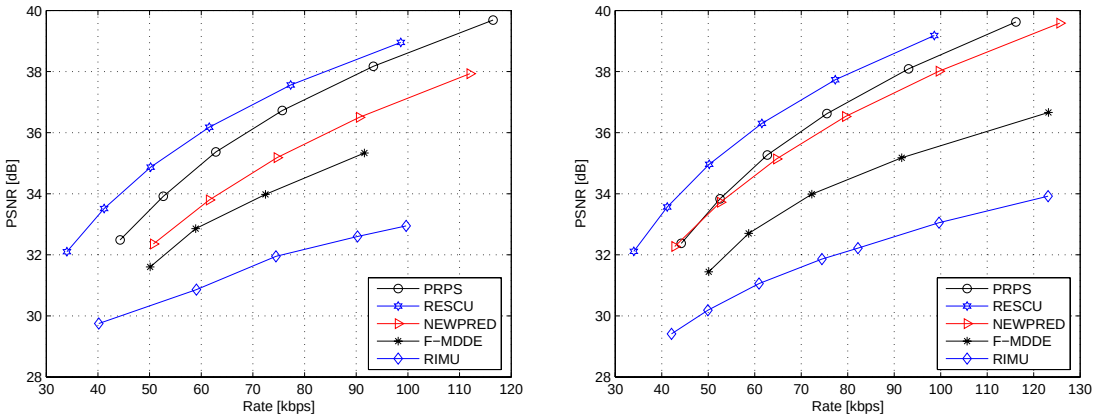
(d) 5% uplink 5% downlink burst packet loss

Figure 2.16: Performance of PRPS and the comparison schemes for the *Foreman* sequence

In this section, the performance of the proposed PRPS scheme as described in Section 2.3.3 is investigate for various RTTs on the uplink and the downlink. Then, it is compared with the state-of-the-art error robustness approaches examined above. SLRPS is used as the error recovery scheme on the uplink in the following in the PRPS framework.

Fig. 2.15 shows the RD performance of PRPS as a function of the RTT on the wireless links. *PRPS_AK1* represents the case when the maximum RTT on the uplink and downlink is 1 frame interval, which means that the feedback information can be obtained almost instantaneously. As expected, this ideal condition achieves the highest RD performance and the larger the RTT, the lower the performance. Please note that the performance degradation is not a linear function of the RTT. For increasing RTT, the additional performance degradation decreases.

As mentioned in Section 2.4.1, to get the curves for RIMU in Fig. 2.16, the number of INTRA MBs is varied from 0 to 80 per frame and the simulation run with the best performance



(a) 5% uplink 5% downlink random packet loss (b) 5% uplink 5% downlink burst packet loss

Figure 2.17: Performance of PRPS and the comparison schemes for the *Salesman* sequence

is picked. The RIMU scheme shown here for comparison therefore works better than it would perform in practice where the current loss rate is normally unknown to the sender and picking the optimum INTRA refresh rate would be impossible. For the feedback based approaches RESCU, NEWPRED and F-MDDE in Fig. 2.16, the end-to-end RTT is set to be 6 frames. Due to the faster feedback provided by the proxies in the proposed approach, the RTT is set to 3 frames on both uplink and downlink and no additional delay in the wired network is assumed.

As shown in Fig. 2.16, the proposed approach (PRPS) outperforms the other schemes for all channel conditions. RESCU performs second best at low loss rate. Especially, when the loss is bursty, fewer periodic frames are corrupted and it achieves the same RD curve as PRPS. For higher loss rate (5%), the error propagation caused by the loss of periodic frames degrades the performance of RESCU significantly and NEWPRED performs better. The other two approaches, F-MDDE and RIMU have much lower performance. F-MDDE outperforms RIMU as a result of the RD-optimized mode decision with feedback. However, the distortion still needs to be estimated through multiple decoders, which leads to a performance gap of up to 2.5dB compared to the three approaches which have an exact decoder side information.

Compared to the *Foreman* sequence, the *Salesman* sequence has a much lower motion activity. At low loss rate, all approaches have a very close performance. Therefore, in Fig. 2.17 only the performance for 5% random and burst packet loss is plotted. RESCU performs better than PRPS because *Salesman* is not so sensitive to packet loss. At the same reconstruction quality, PRPS has a higher bitrate because of the coding structure it uses. However, when the RTT or the packet loss rate increases, the performance of RESCU declines much faster than that of PRPS.

Fig. 2.18 shows the mean reconstruction quality in PSNR for the fives schemes as a function

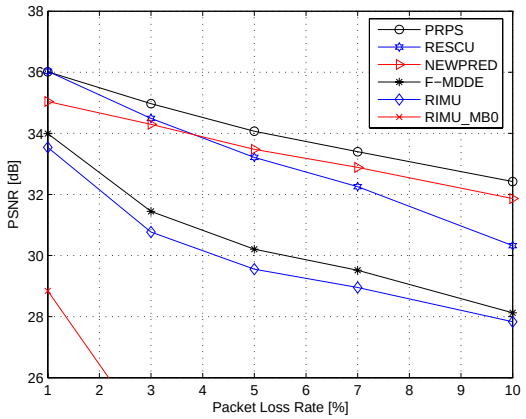


Figure 2.18: Mean reconstruction quality as a function of packet loss rate for a mean packet burst loss length of 5 for the *Foreman* sequence

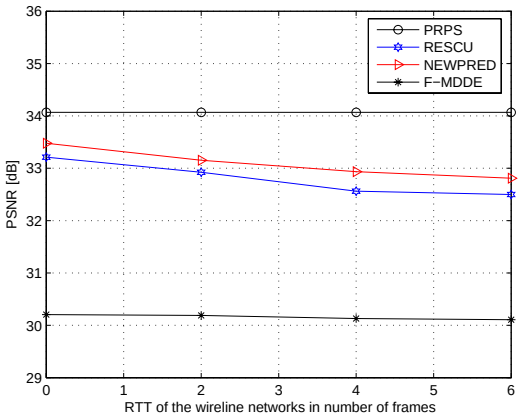


Figure 2.19: Mean reconstruction quality as a function of RTT for 5% packet loss rate for a mean burst length of 5 for the *Foreman* sequence

of loss rate for the *Foreman* sequence. As the wireless channel is normally bursty, in this experiment, the Gilbert-Elliott model is used to generate the up- and downlink test channels with 1%, 3%, 5%, 7%, 10% packet loss rate and keep the average burst length to be 5 packets. The available transmission rate is restricted to be 150 kbps including all overheads, such as the retransmission of lost packets. The RIMU curve again connects the best points with optimal update rate at all loss rates. The proposed PRPS scheme outperforms all other schemes at all loss rates and significant improvements can be observed in Fig. 2.18. Compared with RIMU_MB0*, the lower coding efficiency of all error resilient schemes is compensated by the much higher performance even for 1% packet loss, which supports the statement in Section 2.3.1. Moreover, the trade-off between NEWPRED and RESCU can be easily made, where RESCU is more efficient for low packet loss rates and NEWPRED performs better for high packet loss rates.

Fig. 2.19 illustrates the impact of the RTT on the mean reconstruction quality for the *Foreman* sequence for all feedback based approaches. The transmission channel is again assumed to have a capacity of 150 kbps at 5% packet loss rate with a mean burst length of 5 packets. The RTT on the uplink and downlink is assumed to be 3 frames. The x-axis represents the RTT of the wired network between the two base stations. When it equals to 0, it means there is no additional delay in the wired networks, which leads to an upper bound for the end-to-end error resiliency schemes. In this case, the end-to-end RTT is equal to 6 frames (3 frames on the uplink, 0 frames on the wired link and 3 frames on the downlink). The proposed PRPS scheme has a constant performance when the RTT between the two base stations increases. F-MDDE also shows a stable but much lower performance because of its

*The baseline approach in H.264/AVC codec without any protection

inaccurate distortion estimation. The performance of NEWPRED and RESCU declines when the RTT increases and the larger the RTT in the wired network, the bigger the gap between them and the PRPS approach.

Table 2.1: RD performance when only one mobile user is in wireless network

Approaches	PRPS	RESCU	NEWPRED	F-MDDE	RIMU
uplink (dB)	35.478	34.969	33.748	31.999	31.256
dwlink (dB)	34.772	34.927	33.734	31.821	31.240

As described in Section 2.3, two different error recovery schemes for uplink and downlink have been proposed. So far, only the performance when the two schemes work in concert has been shown. Table 2.1 gives the performance when either a wireless uplink or a wireless downlink is involved in the end-to-end transmission. All results in Table 2.1 are obtained for 5% burst packet loss either on the uplink or the downlink for the *Foreman* sequence at 150 kbps including all overheads. The RTT for the other schemes is still assumed to be 6 frames and 3 frames for PRPS. *PRPS_uplink* performs much better than the other approaches with the advantage of small RTT. Similarly, *PRPS_dwlink* outperforms the other approaches and has very close performance to RESCU. RESCU performs well here because there is only 5% packet loss in total on the transmission path, while in the previous simulations, 5% loss on the uplink and 5% loss on the downlink result in up to 10% packet loss for the end-to-end transmission. When the loss rate is low, the error propagation in RESCU is limited and is compensated by the relatively higher coding efficiency.

2.5 Complexity Analysis

As shown in Section 2.4, all error resilient video transmission strategies improve the quality of the conversational video applications for mobile users. However, because of the limited computational resources and memory of mobile terminals, not all above mentioned schemes can be applied in practice. Therefore, in this section, the computational complexity and memory requirements of these approaches are evaluated.

RIMU is one of the error resiliency features which have been included in the H.264/AVC codec [HHIb]. Given the number of MBs that should be encoded in INTRA mode, a random number is generated for each MB to determine its mode. By this early mode decision, the complexity of the mode decision for the whole frame is reduced and almost no additional complexity or storage requirement is introduced.

Most of the RD-based mode decision approaches lead to high computational complexity. All possible rate and distortion combinations are examined to achieve an optimal RD performance. In MDDE, the encoder has to perform K times decoding when encoding one single

frame, which is a brute force way to get the estimated distortion. Even if the K is set to be 30, this scheme still imposes a very heavy burden on the mobile terminals. If the feedback information with a RTT of N frames should also be included, N steps updating of DPB as well as additional decoding of $30 \cdot N$ frames are required by F-MDDE. At the same time, a N frames memory buffer is needed at the encoder for both approaches.

NEWPRED and RESCU have almost the same complexity. As in this work, they always predict from one single reference frame and hence the computational complexity is almost the same as for conventional encoding. With a RTT of N frames, NEWPRED needs to store the previous N frames in the buffer, no matter if it is A-NEWPRED or N-NEWPRED. In contrast, RESCU has a moderate storage requirement. As all non periodic frames only predict from the most recent periodic frame and are not used as reference frames at the encoder, only one periodic frame has to be stored.

In the PRPS approach, the decoder needs to report the status of the received packets, which is needed in all feedback based approaches. In addition, accelerated decoding has to be supported by the decoder, which is also required by RESCU. As the decoding has much lower complexity than encoding, it will not add too much burden to the decoder. At the encoder, with FLRPS, no additional complexity is needed. With SLRPS, more than one reference frame will be involved for motion estimation if the default reference frame is not error free. As the motion search range is limited within 16 pixels and one row of MBs is put into one slice, only part of the frame is affected if the reference frame is corrupted. As shown in Fig. 2.20, if the 5^{th} slice in frame i is corrupted, in frame i+3, only slice 4, 5 and 6 will be possibly affected. These three slices predict from two frames and all other slices in frame i+3 just use frame i as the reference frame, which limits the additional complexity on motion estimation. The SLRPSEC always uses a single reference frame with the additional cost of performing error concealment.

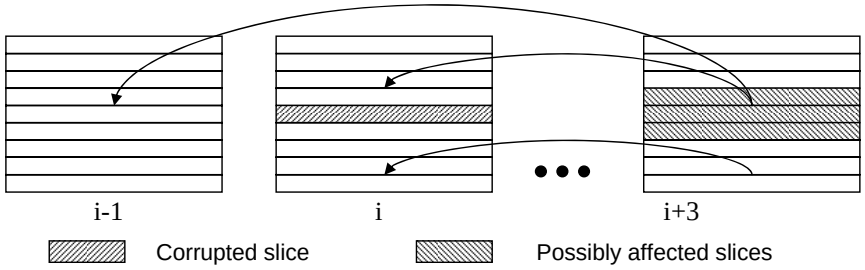


Figure 2.20: Search range of the slice level RPS without error concealment

The storage cost of the proposed approach is limited by the involvement of proxies. With an end-to-end RTT of N frames, the encoder only needs to store at most $\frac{N}{2} + L$ frames with SLRPS, even in the worst case when there is no transmission delay between the two base stations. Here L is the additional reference frame determined by the burst length of

the channel, which should be much smaller than N. SLRPSEC and FLRPS use always one reference frame and hence need to store only $\frac{N}{2}$ frames in the PRPS scheme.

2.6 Chapter Summary

In this chapter, a low complexity proxy-based framework for error resilient transmission of conversational video in wireless environments is presented. Fixed distance reference picture selection is combined with retransmission of lost packets to deal with losses on the downlink. The prediction distance is adjusted according to the RTT on the downlink which provides the opportunity to retransmit lost packets and to use successfully retransmitted packets to stop error propagation with accelerated decoding. This strategy is combined with adaptive RPS on the uplink triggered by feedback from the base station to the sender. The PRPS proposed here is fully standard-compatible when using H.264/AVC and is of very low complexity as it requires only little processing at the base stations. The two major assumptions are: (1) the base stations can send feedback about lost packets to the sender and can retransmit lost packets to the receiver. (2) the decoder has enough computational resources to decode retransmitted slices fast enough to use them to stop error propagation. Both of them can be fulfilled in real systems. As a final point, the PRPS approach can be combined with other error resiliency approaches for which even better performance is expected.

Chapter 3

RD-Optimized Rate Shaping

In this chapter, the benefits and use cases of a proxy for rate adaptation and resource allocation in the presence of network congestion are studied. Different solutions for streaming video and conversational video applications are proposed to improve the overall video quality among all participants.

3.1 Introduction

Recently, video based multimedia applications become more and more popular. A typical application is video streaming, where videos are pre-encoded and stored on the server. A client retrieves the video content from the server and starts the playout with a short delay (e.g., 1-5 seconds). Another major application is conversational video (for instance, video conferencing), which requires a very low delay that is smaller than 200 ms. In both cases, the compressed video streams are sent through the network and played out by the receiver.

Although many video based services have been successfully launched and widely used, it is still challenging to transmit high quality videos. Firstly, the limited available transmission resource can not fulfill the huge amount of traffic generated by the video applications. Secondly, video packets are typically transmitted using best-effort service in today's Internet. When the network is congested, the packet forwarding service at network nodes will be significantly degraded.

Herein, the scenario illustrated in Fig. 3.1 is considered, where multiple video streams pass through a network node (e.g., multimedia gateway, relay server, router ...) with limited forwarding resources. The video packets can be temporarily cached in the node's buffer, but if the overload persists, the buffer will overflow and some packets will be lost. This kind of random packet loss significantly degrades the video quality. Furthermore, transmission resource might be wasted for sending the packets in the buffer, which have missed their

playout deadline. Therefore, a more intelligent way of frame dropping is needed according to network status. For this, proxies with more computational power are setup on top of the network nodes in the Internet. In this chapter, several rate-distortion optimized rate shaping strategies are proposed and compared, which can be employed by the proxies running on such overloaded network nodes to achieve the highest overall quality of the streams for the given forwarding resource R_{out}.

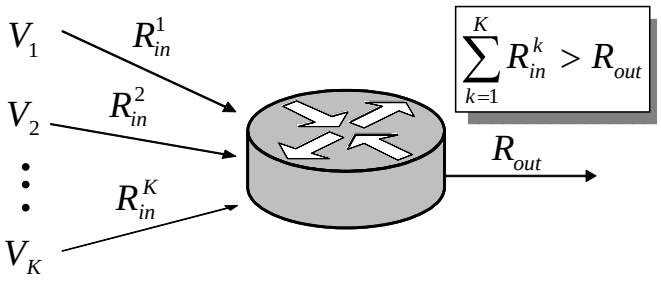


Figure 3.1: A network node with K incoming video streams sharing the same outgoing link. The aggregate input rate is larger than the available output rate.

In Fig. 3.1, the incoming video streams can be streaming and/or conversational videos encoded with standard codecs (e.g., MPEG-2, H.263, H.264/AVC) in a single layer structure. Nowadays, most of the video traffic on the Internet is encoded in this way. Therefore, it is worth to study and find appropriate rate adaption solutions for them. However, owing to the dynamic heterogeneous network conditions and different user requirements, the conventional coding structure can not fulfill some requirements of the new media applications as it has limited freedom of scalability. The scalable extension of H.264 [SMW06] emerges as a new technology enabling scalable video coding and provides more flexible rate adaptation. Herein, solutions are also given when all the incoming streams are scalable encoded videos. Please note, in this chapter, without specification, *Streaming Video* and *Conversational Video* represent the two types of applications encoded with single layer. Since additional delay is introduced by scalable video coding, *Scalable Video* is used only for streaming video applications in this dissertation.

The rest of this chapter is organized as follows. Section 3.2 reviews the state-of-the-art addressing congestion for video traffic. In Section 3.3 to Section 3.5, frame dropping strategies and scheduling strategies are proposed for streaming and conversational videos. Rate shaping methods for scalable video are then described in Section 3.6. Section 3.7 analyzes the computational complexity as well as the additional overhead to validate the feasibility of the proposed schemes. Experimental results are given in Section 3.8 to show the significant improvements that can be achieved by the proposed rate shaping strategies. Section 3.9 summarizes this chapter.

3.2 State-of-the-art

3.2.1 Transcoding

In a heterogeneous network environment, transcoding can be performed to adapt the source rate of a pre-encoded bitstream to the available transmission rate. Overviews of transcoding techniques for video traffic are given in [VCS03] and [XLS05]. DCT domain transcoding [EA95, ZL96, SOR03, CFCL00] is one of the most popular schemes that are commonly used. DC coefficients are kept because they provide the most significant information and AC coefficients are selectively dropped to fit the transmission rate. A rate-distortion model is proposed in [YVXL03] to determine which AC coefficients should be dropped to fit the available rate with a smallest distortion increment. A pixel domain transcoding algorithm is proposed in [YXS00], which outputs better performance with additional complexity. [CC03] considers a joint source-channel transcoding, where information of previous packets is assumed to be at hand so that the source rate and channel coding rate for the pre-encoded bitstreams can be optimally adjusted. Transcoding schemes for low delay streaming video in wireless networks are proposed in [LG03, VCC02]. Although a lot of simplifications have been made in the above works, transcoding is still computationally expensive. It can be employed at the edge of the network (e.g., content server as proposed in most of the above works). However, it is not suitable to be used by the network node that has to rapidly forward packets of many different users.

3.2.2 Frame Dropping

Another way to prune the video source can be accomplished by frame level dropping. A comparison of several frame dropping strategies are reported in [KHK04]. In [FLKP98, LC99, COH03], static priority labels for I-, P- and B-frames are used to perform Priority-based Random Dropping (PRD) for streaming video. In particular, video frames are dropped according to their priority labels. Random selection is performed among frames with the same priority label. Priority-based Random Early Dropping (PRED) [MFW01] improves PRD by early dropping of lower priority frames at certain predefined buffer fullness levels [RFC98]. Nonetheless, static priority labels can not accurately describe the importance of video frames. For example, the first P-frame in a GOP is in most cases much more important than the last P-frame, although they belong to the same priority class of frames. Some other methods are also used to dynamically calculate frame priority for streaming video. For example, decodability, block types and motion energy are used to assign priorities to video packets in [KHK04], [KKH04] and [EPS03, LZQ03], respectively. Individual deadlines of video packets are used to selectively drop low priority video packets in [ZNAT99] and [LLLW04]. The priority based frame dropping strategy has also been used together with other techniques. For instance, in

[KWCK05], the static priority based frame dropping is performed jointly with transcoding. Furthermore, the frame dropping is combined with rate switching in [SW04].

Rate-distortion optimization is originally used in video compression and joint source channel coding. For example, coding mode selection in video compression [SW98, SW02], or bit allocation between source coding and channel coding [ZBPK03]. A RD-based error robust video delivery scheme is also proposed in [ZRR01, MYRM04]. Later, it becomes widely used to minimize the distortion with constrained resource by optimal selection of content to transmit. Chou *et al.* proposed a rate-distortion chain [CM01] to determine an optimal transmission pattern. Low complexity RD-optimized packet dropping approaches are described in [Bou02, Bou03, CAT+04, CAW+04], where error concealment is performed when calculating the influence of each video packet to the whole video quality when they are discarded. A novel error propagation model is proposed in [LAG03] to extract the RD information at very low complexity.

3.2.3 Scalable Video

As shown above, many works have been done for the flexible transmission of streaming video in the Internet. However, further exploration of scalable video is still limited. A two stage frame dropping based on scalable video is proposed in [ZA01] to improve the performance from single layer coded video stream. Pahalawatta *et al.* consider the resource allocation for multiple scalable videos in wireless environment with an application-aware scheduling strategy in [PPBK07]. A fairly different work on dynamic sharing of radio resources in a wireless system by combining scalable video coding with appropriate radio link buffer management for multi-user streaming services is presented in [LSWS06]. However, in these three works, only the SNR scalability is involved in addition to the temporal scalability. The original resolution is kept at the expense of a decrease of the frame rate although in many cases, the users would rather have a higher frame rate than a higher resolution.

3.2.4 Multiuser Optimization

State-of-the-art techniques for adaptive streaming reviewed above reshape the pre-encoded video sequences in case of scarce transmission resources. However, most of them consider only a single user environment. In current Internet, the most common scenario is the sharing of transmission resource by multiple users/flows. To consider those users with similar characteristics separately leads to an inefficient solution. Therefore, a solution for joint optimization is highly desired. [HPZ+98] is maybe the first study of video streaming in a multi-user scenario. However, in that work, it is assumed that instantaneous reports of network congestion are available at the sender, which is not realistic in the present Internet. Furthermore, it also considers a real-time encoding scheme, which can be dynamically adjusted according to

the feedback information. Obviously, it is not applicable to the pre-encoded videos. [CF05] is probably most closely related to the approaches proposed in this chapter, as it also considers RD-optimized frame dropping from multiple video streams. However, the inaccurate side information used in this scheme limits its performance. Moreover, only streaming video is considered in [CF05]. The case when conversational videos or both types of videos are involved is not considered. This issue is, however, addressed by the proposals given in this chapter.

3.3 Rate Shaping for Streaming Video

In this section, the priority-based frame dropping schemes are first introduced. Then, the extended utility-based frame dropping strategies and the corresponding Hint Track (HT) side information for streaming video are presented, which are used for comparison in Section 3.8 together with the priority-based frame dropping. Afterwards, the definition and the procedure of constructing the accurate side information (Distortion Matrix and Rate Vector) with low complexity are presented. Based on this side information, a RD-optimized frame dropping strategy depending on the current buffer fullness of the network node is proposed.

3.3.1 Priority-based Random Early Dropping (PRED)

Priority-based Random Dropping (PRD) makes dropping decisions based on fixed priority labels assigned to the video frames. With conventional coding schemes, frames can be prioritized according to their frame type: I-, P- or B-frame. When frames from multiple video streams arrive at a network node simultaneously, it is the I-frames among them that have the highest priority to be placed into the node's buffer. It may happen that some of these I-frames cannot be placed into the buffer and therefore they are dropped even before the buffer is totally full. In this case, P-frames are tried next to be placed into the buffer, followed then by the remaining (if any) B-frames. This strategy leads to the most efficient usage of the node's buffer. However, the loss of I-frames and P-frames has a dramatic influence on the reconstruction video quality.

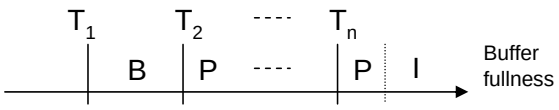


Figure 3.2: Example settings of dropping thresholds for PRED

PRED sets thresholds for dropping according to the number of priorities available for the video streams. Here, three priority levels $\{I,P,B\}$ are available so that only two dropping

thresholds are needed. But if different priority levels are assigned to all P-frames according to their positions in the GOP, $n = \frac{L_G}{L_B+1}$ thresholds can be set, where L_G is the length of the GOP and L_B denotes the number of B-frames between two I/P-frames. As shown in Fig. 3.2, when the buffer fullness reaches T_1, the least important B-frames are all dropped. The last P-frame in the GOP (least important) is dropped when T_2 is reached. I-frames have the highest priority level and should not be early dropped, so all P-frames are dropped when the buffer fullness reaches the highest threshold (T_n). Early dropping of less important frames reduces the likelihood of having to drop more important frames at a later time.

3.3.2 Utility-based Frame Dropping

Rate-Distortion Hint Tracks, proposed in [CAW+04, CAW+05], are measured by feeding a specific loss pattern to the decoder and summing up the resulting increase in MSE over all affected frames of the video sequence. In this section, the usage of Hint Tracks as well as the utility-based frame dropping strategy [CAW+04] are extended to streaming videos with GOP structure.

3.3.2.1 Side information - Hint Tracks

For streaming video with an IBBPBBP... GOP structure, two frames that belong to two different GOPs can be considered independently to calculate their contribution to the overall distortion affecting the compressed video sequence, in case these frames are dropped. Given that the n^{th} frame in display order in a GOP is a P- or I-frame and is lost, the dependent B- and P-frames are not dropped. Instead, the decoder performs error concealment for the dropped frame and keeps on decoding. The error propagation is terminated by the I-frame at the beginning of the next GOP, as shown in Fig. 3.3. For B-frames, as they are not used as reference frames, there is no error propagation to other frames.

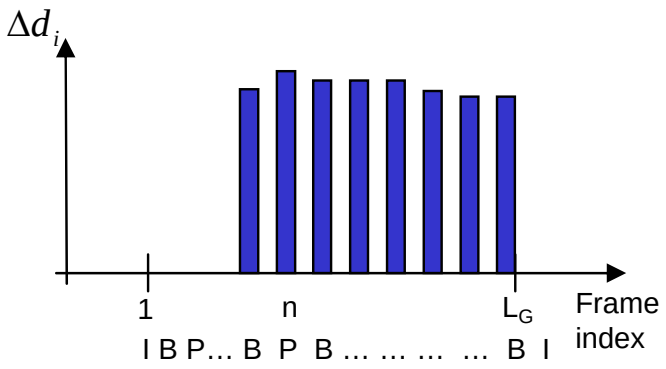


Figure 3.3: Error propagation in an IBPBP... structure

Now, the overall distortion due to dropping a P-frame or an I-frame can be estimated as

$$D^0(n) = \sum_{i=n-L_B}^{L_G} \triangle d(i), \tag{3.1}$$

where i starts from $n - L_B$ because the L_B frames prior to the n^{th} frame in the GOP are B-frames, which use the lost frame n as a reference frame. For example, L_B=1 in Fig. 3.3. D^0 represents the overall distortion calculated under the assumption that each frame loss is independent. This quantity is a good estimate of the actual overall distortion only when the loss rate is very low and hence the packet losses are not bursty. For a more accurate distortion estimation, further $D^1 \ldots D^{L_G-1}$ have to be employed, which calculate the additional distortion caused by dropping the current frame given which of the previous frames in the same GOP have been already dropped. It should be noted that each of these quantities corresponds to a Distortion Chain (DC) of different order i, i.e., DC^i for $i = 0, 1, ..., L_G$. In particular, DC^i is a distortion modeling framework that can be used to predict the overall distortion affecting a reconstructed video sequence in the case of frame dropping or loss introduced in [CAT+04]. Its accuracy in distortion prediction increases with the order of the DC. However, at the same time the computational complexity and memory requirements increase with i. At high loss rates or when successive frame losses occur, RD Hint Tracks based on D^0 and D^1 lead to inaccurate distortion estimation and hence suboptimal dropping decisions. The performance of the utility-based approach for frame dropping that employs Hint Tracks also depends on the number of frames that are considered jointly when the optimization is performed. Here, a window of size W can be used so that at every moment W frames from each stream are taken into account for the optimization.

3.3.2.2 Frame Dropping Approaches

In the Hint Track framework, for DC^0, the distortion (ΔD) and rate (ΔR) information associated with a video frame comprise respectively the additional distortion affecting the reconstructed video sequence and the corresponding data rate reduction*, when a single video frame from the compressed video stream is dropped. The distortion-per-bit utility for a frame is then calculated as the ratio $\Delta D/\Delta R$. In the approach described here, the current incoming (n^{th}) frames from all K simultaneously arriving videos are sorted in decreasing order, according to their distortion per-bit utility. Then, from the head of the sorted list, frames are placed into the node's buffer until no additional frame can be placed into it. However, in this case optimization is done only among video frames that correspond to a single time instance (one frame slot), which limits the efficiency of the optimization. For pre-encoded streaming video, it is possible to have side information for future frames as well, so that a

*In this case, the rate saving is equal to the size of the dropped frame in bytes or bits.

decision window could be employed that includes both current and future frames. This will enable a more efficient frame selection and therefore optimization.

Decision Window (DW) Streaming video applications have moderate delay constraints. Therefore, a Decision Window (DW) can be set before the router buffer as shown in Fig. 3.4. It can hold $W * K$ frames, where W is the length of the decision window. Dropping decisions are made only when there are $W * K$ frames in the DW. All these frames are sorted based on their distortion-per-bit utility and are then placed into the buffer in that order, as explained earlier. Once frames have been placed into the outlink buffer, their order cannot be changed, nor they could be removed from the buffer afterwards.

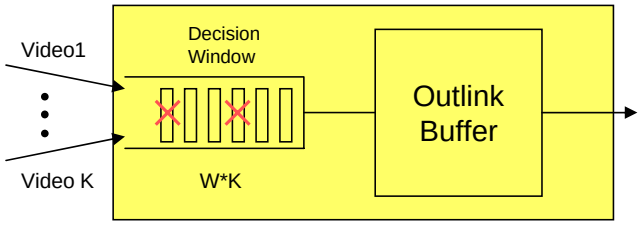


Figure 3.4: Frame dropping decision with a decision window

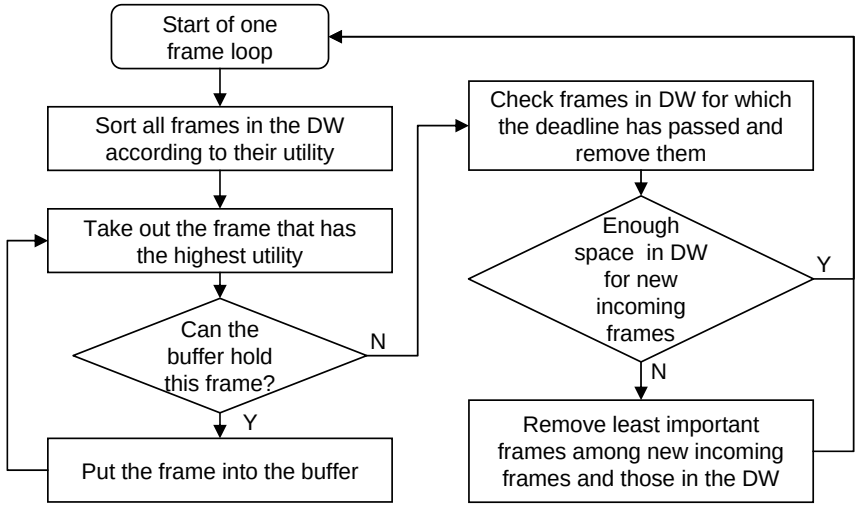


Figure 3.5: DW: Algorithm flow chart

The optimization algorithm for managing the DW is run on every new arrival of a video frame or a group of them, if they arrive simultaneously. The flow chart of the algorithm is shown in Fig. 3.5. At every frame slot, there is no limit on the number of frames that can be placed into the outlink buffer from DW. Furthermore, when an "important" frame is selected to be put into the buffer, but can not be put there because of buffer overflow, the system stops the current loop and waits for the next loop, so that big frames with high utilities always have

a chance to be sent out. It is likely that some frames in the DW may never be selected to be moved into the outlink buffer due to their low utility. Therefore, the algorithm removes all expired frames from the DW whenever such a condition is detected: the delivery deadline of a frame has passed. Finally, if there is not enough space in the DW for the new incoming frames, an importance check is run on the frames from the DW and the new incoming frames, together. Then, a set of least important frames are selected to be discarded such that the remaining frames can fit into the DW.

Using a DW enables a frame dropping selection from a larger pool of candidate frames. However, an additional processing delay is introduced depending on the window size.

Virtual Decision Window (VDW) For low latency video streaming applications, the additional delay introduced by a DW is usually unacceptable. Therefore, as an alternative, a virtual decision window can be employed in such scenarios. As in the DW case, a VDW comprises $W * K$ current and future incoming frames, however dropping decisions are only made on the current K frames from the VDW. To this end, it is still assumed that RD side information for future frames can be made available at the network node prior to their actual arrival.

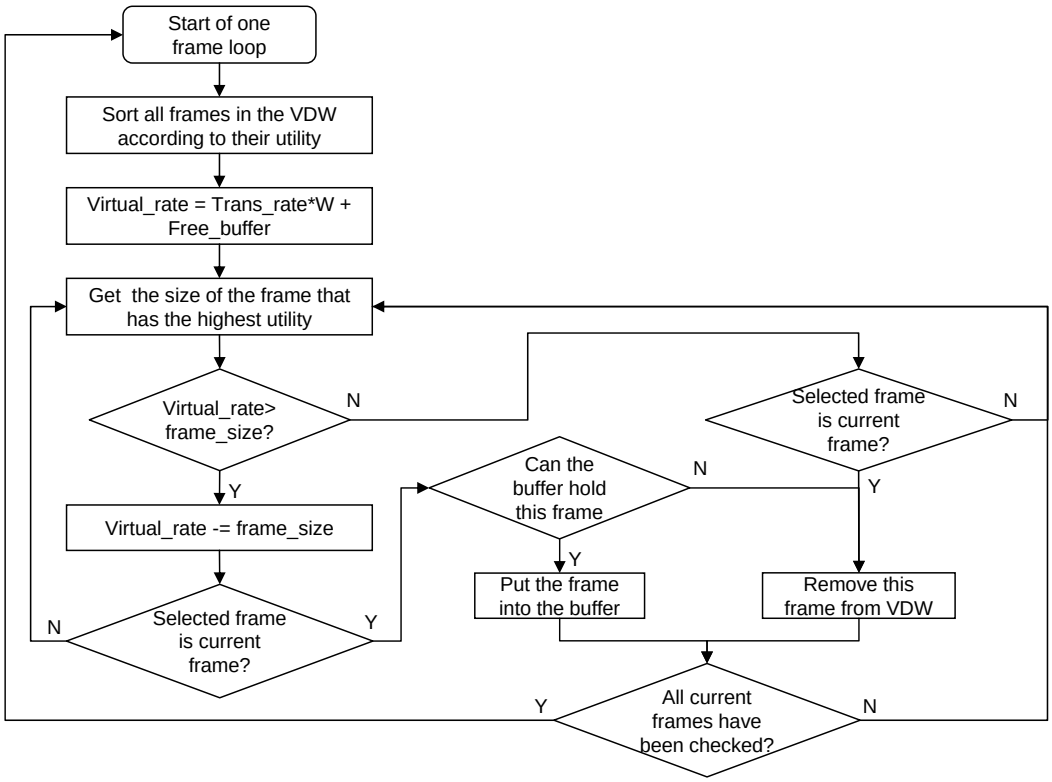


Figure 3.6: VDW: Algorithm flow chart

The operation of a VDW is performed as follows. First, the cumulative number of bits/bytes that can be spent on the frames in the VDW is computed as the product of

the forwarding data rate times the length of VDW in time units (frame slots) plus the remaining (unoccupied) space in the node's outlink buffer. This quantity is denoted as the *Virtual_rate*. Then, frames in the VDW are sorted according to their utilities, in decreasing order. Next, from the head of the sorted list frames are selected until all the current frames in the VDW have been checked. In particular, when a frame from the sorted list is selected, the *Virtual_rate* is first checked to see if it is larger than the size of this frame in bytes. If this is true, the size of this frame is subtracted from the *Virtual_rate* and whether this frame is a current frame is checked next. If that is also true, and there is enough space in the node's buffer, this frame is placed there and the algorithm proceeds to the next frame in the sorted utility list. Otherwise, if there is not enough room to place a current frame into the buffer at this point, the selected current frame is simply dropped, and the next frame in the sorted list is processed.

On the other hand, if upon selecting a frame from the list, the *Virtual_rate* is detected to be smaller than its size, checking if this frame is a current one is the next step. If that holds, this frame has to be dropped, as there is not enough rate to support its transmission on the outgoing link. The complete flow chart of the VDW management algorithm is shown in Fig. 3.6.

3.3.3 Cost Function-based Video Frame Dropping

In Section 3.3.1, the idea of PRED is reviewed and the benefit obtained by "early" dropping is discussed. In Section 3.3.2, the utility-based frame dropping with approximated RD side information (Hint Tracks) are presented. In this section a cost function based approach with Distortion Matrix [TKS04] is proposed, which takes advantage of more accurate RD side information to enable more flexible frame dropping decisions, while still using the buffer fullness info for early dropping.

3.3.3.1 Distortion Matrix and Rate Vector

The Distortion Matrix can be used to calculate the distortion caused by dropping frames in a GOP structured video stream. When calculating the reconstruction distortion, it is assumed that a simple "copy and freeze previous frame" error concealment scheme is employed by the decoder. In particular, a missing frame and all of its descendants[†] are replaced, at reconstruction, by the decoder with the temporally nearest previous frame that has been decoded. Note that this is done regardless of the presence status, at the decoder, of the descendant frames. Hence the name of the concealment scheme.

[†]These are the frames in the encoding chain that depend on the missing frame in order to be decoded, i.e., decompressed.

The proposed approach to frame dropping follows this logic. When the proxy on a network node drops an arriving video frame, it will subsequently drop all its dependent frames that prospectively will arrive at the node afterwards. Therefore, a video frame drop pattern comprises in this case an incoming frame that is dropped at present together with its descendant frames that will prospectively[‡] be dropped afterwards. The increase in reconstruction distortion affecting a video stream caused by a frame drop pattern is the sum of the individual increments in reconstruction distortion for the concealed video frames. That is because the frames that have been decoded do not contribute to the increase in reconstruction distortion. The Distortion Matrix for a GOP with $IB_1B_2P_1B_3B_4P_2B_5B_6$ structure is given below,

$$\begin{array}{r} R: \\ I: \\ P_1: \\ P_2: \\ B_1: \\ B_3: \\ B_5: \end{array} \begin{bmatrix} d_I^R & d_{B_1}^R & d_{B_2}^R & d_{P_1}^R & d_{B_3}^R & d_{B_4}^R & d_{P_2}^R & d_{B_5}^R & d_{B_6}^R \\ / & d_{B_1}^I & d_{B_2}^I & d_{P_1}^I & d_{B_3}^I & d_{B_4}^I & d_{P_2}^I & d_{B_5}^I & d_{B_6}^I \\ / & / & / & / & d_{B_3}^{P_1} & d_{B_4}^{P_1} & d_{P_2}^{P_1} & d_{B_5}^{P_1} & d_{B_6}^{P_1} \\ / & / & / & / & / & / & / & d_{B_5}^{P2} & d_{B_6}^{P2} \\ / & / & d_{B_2}^{B_1} & / & / & / & / & / & / \\ / & / & / & / & / & d_{B_4}^{B_3} & / & / & / \\ / & / & / & / & / & / & / & / & d_{B_6}^{B_5} \end{bmatrix} \tag{3.2}$$

where $d_{f_{loss}}^{f_{rep}}$ represents the increase in MSE distortion that is observed when replacing frame f_{loss} by f_{rep} as part of the concealment strategy. The column left to the matrix shows the replacement frame f_{rep} for every row of the matrix. For instance, $d_{B_1}^I$ represents the additional reconstruction distortion if the first B-frame of the GOP is lost and is therefore replaced by the I-frame of that GOP. R is a frame from the previous GOP that is used as a replacement for all the frames in the current GOP if the I-frame of the current GOP is lost. As a worst case assumption, the I-frame of the previous GOP is used as the replacement frame in this case.

The entries of the Rate Vector correspond to the sizes of the video frames expressed in bytes. Then, the proxy sums up the size of an incoming frame and the sizes of its descendant frames are summed up to determine the prospective rate saving achieved by dropping these frames.

3.3.3.2 Frame Dropping Approach

The cost function based frame dropping approach relies strongly on the current buffer fullness level. If the buffer is empty or is lightly loaded, no frames should be dropped. However, when the buffer fills up, frames that have the least impact on the perceived quality at the receiver should be dropped first. The decision which frames to drop is jointly made for all video streams. Given the RD side information introduced in Section 3.3.3.1, the proxy can

[‡]In case they do arrive at the node.

perform RD-optimized frame dropping. For this, the proxy checks the current buffer fullness and minimizes the Lagrangian cost function

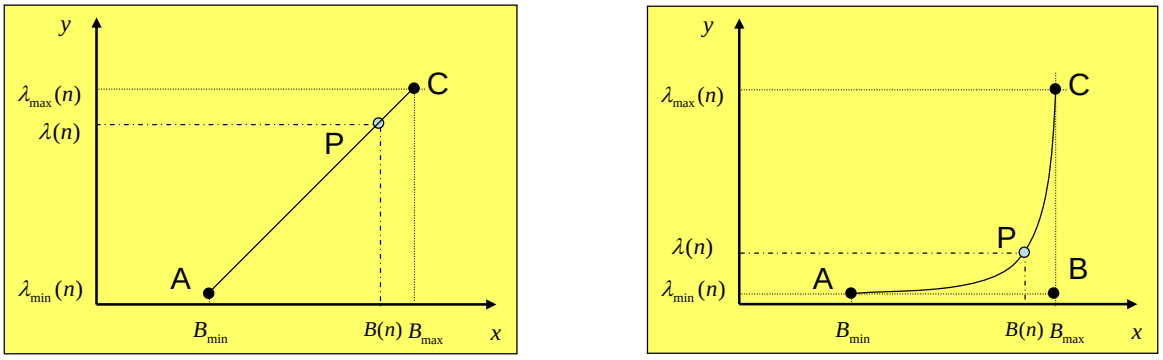
$$J_p(n) = \sum_{k=1}^{K} \Delta D_p^k(n) - \lambda(n) \sum_{k=1}^{K} \Delta R_p^k(n) \tag{3.3}$$

to determine the optimal drop pattern. In (3.3), n is the current (discrete) time instant (slot), $\Delta D_p^k(n)$ is the additional distortion introduced in video k for a given drop pattern p, and $\Delta R_p^k(n)$ is the corresponding rate saving in bytes.

When the Distortion Matrix and Rate Vector described in Section 3.3.3.1 are used, a dropping decision should comply with the following rules. If the current frame that arrives at the network node is an I-frame, it can either be dropped or be sent to the outgoing link buffer. If it is dropped, this means that all the following P- and B-frames in the same GOP cannot be decoded and have to be dropped also. This dropping strategy leads to a significant increase in distortion for this GOP but at the same time reduces the sending rate to zero for this GOP. If the I-frame is not dropped at this moment, the subsequent P-frames and B-frames can still be selected for dropping. This will lead to reduced distortion but also the rate saving will be smaller. Decision can also be made to drop only the B-frames if the P-frames should be kept. Again, the additional distortion will be reduced but also the rate saving will be even smaller.

Therefore, if the current incoming frame is an I-frame, there are in total 4 dropping choices $\{\mathcal{I}, \mathcal{P}, \mathcal{B}, \mathcal{N}\}$, where $\mathcal{I}$ denotes dropping the whole GOP, $\mathcal{P}$ stands for dropping the subsequent P- and B-frames in the GOP, while $\mathcal{B}$ signifies dropping all B-frames in the current GOP only and $\mathcal{N}$ stands for "drop nothing". If the current frame is a P-frame, the choices are reduced to $\{\mathcal{P}, \mathcal{B}, \mathcal{N}\}$. If the current frame is a B-frame, the choices are also $\{\mathcal{P}, \mathcal{B}, \mathcal{N}\}$, where $\mathcal{B}$ denotes the case of dropping all the remaining B-frames in the GOP, including the current one, and $\mathcal{P}$ stands for dropping the subsequent (relative to the current B-frame) P- and B-frames.

Now, if the number of possible drop patterns at time n for video k is denoted as $A^k(n)$, then for K videos the dropping set $\mathscr{P}(n)$ is obtained including $\prod_{i=1}^{K} A^i(n)$ different drop patterns. One of the drop patterns will minimize (3.3). This pattern represents the optimal dropping strategy at time n. In order to perform this minimization, a reasonable value has to be determined for the Lagrangian multiplier $\lambda(n)$ in (3.3). In this work, $\lambda(n)$ is determined as a function of the buffer fullness $B(n)$. If the buffer is empty, certainly no video frame is to be dropped. This has to be reflected by an appropriate choice of $\lambda(n)$. On the other hand, if the buffer is full, $\lambda(n)$ should be selected such that all incoming frames are dropped as queueing them in the outlink buffer would fail anyway. In order to determine appropriate values for $\lambda(n)$ for any buffer level, a minimum buffer fullness B_{min} is defined, below which no dropping should happen. Meanwhile, a maximum buffer fullness B_{max} is also defined, above which all incoming frames are dropped. The two buffer fullness levels B_{min}, B_{max} and the corresponding dropping strategies lead to two extreme values for the Lagrange multiplier



(a) Linear interpolation

(b) Quadratic interpolation

Figure 3.7: Interpolation of $\lambda(n)$ between $\lambda_{min}(n)$ and $\lambda_{max}(n)$ as a function of the current buffer

$\lambda_{min}(n)$ and $\lambda_{max}(n)$. The values for $\lambda(n)$ between B_{min} and B_{max} can be interpolated and two different interpolation schemes for $\lambda(n)$ are considered.

Fig. 3.7(a) illustrates a linear interpolation of $\lambda(n)$ between $\lambda_{min}(n)$ and $\lambda_{max}(n)$ as a function of the current buffer fullness $B(n)$. Hence, it can be calculated as

$$\lambda(n) = \frac{B_{max} - B(n)}{B_{max} - B_{min}} \cdot \lambda_{min}(n) + \frac{B(n) - B_{min}}{B_{max} - B_{min}} \cdot \lambda_{max}(n). \tag{3.4}$$

Linear interpolation is the simplest way to interpolate $\lambda(n)$. An interpolation function that leads to more aggressive dropping if the buffer fullness approaches B_{max} can be realized by quadratic interpolation of $\lambda(n)$, as shown in Fig. 3.7(b). With three control points A, B and C, a quadratic Bézier curve can be defined for $\lambda(n)$ with

$$\begin{cases} A &=& (A_x, A_y) &=& (B_{min}, \lambda_{min}(n)) \\ B &=& (B_x, B_y) &=& (B_{max}, \lambda_{min}(n)) \\ C &=& (C_x, C_y) &=& (B_{max}, \lambda_{max}(n)) \end{cases} \tag{3.5}$$

$$\begin{cases} P_x &=& (1-t)^2 \cdot A_x + 2t \cdot (1-t) \cdot B_x + t^2 \cdot C_x \\ P_y &=& (1-t)^2 \cdot A_y + 2t \cdot (1-t) \cdot B_y + t^2 \cdot C_y \end{cases} \tag{3.6}$$

The interpolated point $P = (P_x, P_y)$ moves on this curve from A to C by varying the parameter t from 0 to 1. For a given $B(n)$, t can be determined and then $\lambda(n) = P_y$ from (3.6).

In order to determine $\lambda_{min}(n)$, (3.3) is evaluated for every drop pattern and select $\lambda_{min}(n)$ such that the minimum of (3.3) is obtained for the drop pattern where nothing is dropped in all K video streams. This means that

$$\begin{aligned} J_{p_n}(n) &= \textstyle\sum_{k=1}^{K} \triangle D^k_{p_{\mathcal{N}}}(n) - \lambda_{min}(n) \sum_{k=1}^{K} \triangle R^k_{p_{\mathcal{N}}}(n) \\ &\leq \textstyle\sum_{k=1}^{K} \triangle D^k_{p}(n) - \lambda_{min}(n) \sum_{k=1}^{K} \triangle R^k_{p}(n), \\ &\text{for } p \in \mathscr{P}(n) \text{ and } p \neq p_{\mathcal{N}}, \end{aligned} \tag{3.7}$$

where $p_{\mathcal{N}}$ represents the pattern when no frame dropping occurs in any of the video streams. As $J_{p_{\mathcal{N}}}(n)$ equals zero, this leads to

$$\begin{aligned} &\lambda_{min}(n) \leq \frac{\sum_{k=1}^{K} \triangle D_p^k(n)}{\sum_{k=1}^{K} \triangle R_p^k(n)} \\ &\text{for } p \in \mathscr{P}(n) \text{ and } p \neq p_{\mathcal{N}} \end{aligned} \tag{3.8}$$

and $\lambda_{min}(n)$ is picked to be as big as possible while still satisfying all the inequalities in (3.8). The value for $\lambda_{max}(n)$ is derived in a similar fashion. For this, the minimization of (3.3) should now lead to the decision of dropping as many frames as possible (drop pattern $p_{\mathcal{I}}$), which leads to

$$\begin{aligned} J_{p_{\mathcal{I}}}(n) &= \sum_{k=1}^{K} \triangle D_{p_{\mathcal{I}}}^k(n) - \lambda_{max}(n) \sum_{k=1}^{K} \triangle R_{p_{\mathcal{I}}}^k(n) \\ &\leq \sum_{k=1}^{K} \triangle D_p^k(n) - \lambda_{max}(n) \sum_{k=1}^{K} \triangle R_p^k(n), \\ &\text{for } p \in \mathscr{P}(n) \text{ and } p \neq p_{\mathcal{I}}. \end{aligned} \tag{3.9}$$

This results in

$$\begin{aligned} &\lambda_{max}(n) \geq \frac{\sum_{k=1}^{K} \triangle D_{p_{\mathcal{I}}}^k(n) - \triangle D_p^k(n)}{\sum_{k=1}^{K} \triangle R_{p_{\mathcal{I}}}^k(n) - \triangle R_p^k(n)} \\ &\text{for } p \in \mathscr{P}(n) \text{ and } p \neq p_{\mathcal{I}} \end{aligned} \tag{3.10}$$

and $\lambda_{max}(n)$ is picked to be as small as possible while still satisfying all inequalities in (3.10).

3.4 Rate Shaping for Conversational Video

As mentioned in Section 1.3.1, conversational video is typically encoded in an IPPPPP... form and no B-frame is used. Therefore, no "early" or even priority based dropping as mentioned in Section 3.3.1 can be employed for conversational video. Video frames of multiple users are put into the buffer in a round robin way and dropped if the buffer can not hold them. As conversational video does not have a GOP structure, the Distortion Matrix also cannot be used here to perform dropping decisions. Hence, RD Hint Tracks obtainable in a similar way as in Section 3.3.2 are used as the side information and again a utility based frame dropping is performed to selectively drop the least important frames. However, the Decision Window is always equal to one here, because of the strict delay constraint of conversational video. Please note, unlike the conversational video used in Chapter 2, one video frame is transmitted in one packet in this chapter. The reason is that typically a larger packet size is used in wired networks than in wireless networks.

3.4.1 Side Information for Conversational Video

RD Hint Tracks are measured by feeding a specific loss pattern to the decoder and summing up the resulting increase in MSE over all affected frames of the video sequence. Without periodic I-frames in conversational video, there is no re-synchronization between the encoder

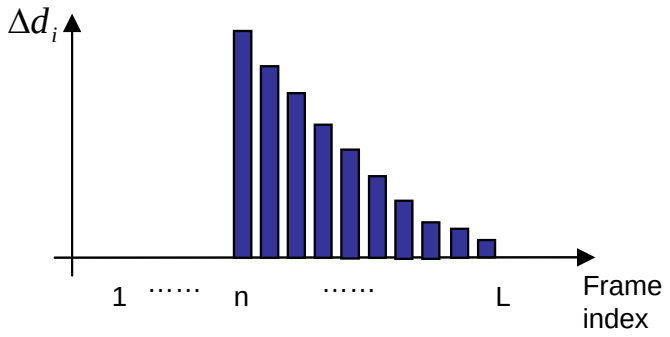


Figure 3.8: Error propagation for a single frame loss

and decoder. Therefore, in order to increase the error resilience of the video stream to packet losses during transmission, slices (or rows) of macroblocks in video frames can be intra updated periodically, usually in a round-robin fashion. This is the so called partial intra update. Fig. 3.8 illustrates the error propagation when frame n is lost under the assumption that there is no remaining error propagating from earlier frames. The total distortion in this case is the sum of the distortions of all the following frames until the end of the video stream. However, with partial intra update, the error propagation by the loss of frame n is assumed to be totally stopped after an equivalent intra update period of M frames[§]. Therefore, only the individual distortions up to frame $M+n$ need to be considered. Please note, here the Hint Tracks are calculated under the assumption that the losses of each frame are independent, which is the so called zeroth-order distortion chain model DC^0 in [CAT+04]. This side information gives accurate distortion estimation when there is only one frame loss in the M consecutive frames. Of course higher order Hint Tracks can be constructed by feeding some loss patterns with more losses. However, they have very high costs in terms of computational complexity as well as a huge storage requirement.

Since the future frame information for conversational video is unknown, it is impossible to pre-measure the DC^0 value associated with a given loss pattern. Therefore, the model proposed in [LAG03] is used to predict/estimate the distortion values $\triangle d(n+i)$ associated with future frames $n+i$ in the case of loss/drop of frame n. In particular,

$$\triangle d(n+i) = \begin{cases} \triangle d(n) \cdot \gamma^i \cdot (1 - i/M), & \text{for } 0 \leq i \leq M \\ 0, & \text{otherwise,} \end{cases} \tag{3.11}$$

where M is the equivalent intra update period, as explained above, and i indicates the distance between the future (concealed) frame and the lost frame n. In (3.11), $\Delta d(n)$ is the MSE information sent along with the video stream, representing the distortion of the current frame n in the case when this is the only lost frame and it is concealed by copying the previous

[§]This is the number of frames needed to intra refresh all the macroblock locations in a video frame using this approach.

frame. The attenuation factor γ^i ($\gamma < 1$) accounts for the effect of spatial filtering and the term 1-i/M accounts for the intra update. Finally, the overall additional distortion $\triangle D(n)$ affecting the video sequence due to the loss of frame n, including error propagation into future frames, is then calculated as

$$\triangle D(n) = \sum_{i=0}^{M} \triangle d(n+i). \tag{3.12}$$

As mentioned in Section 3.3.2, only the size of the individual frames in bits/bytes is sent together with the distortion information.

3.4.2 Frame Dropping Strategy

Unlike the streaming video, the importance of future frames in conversational video is unavailable. Therefore, it does not make sense to make dropping decisions for conversational videos until the buffer is unable to hold the new incoming frames. In particular, all new incoming frames are placed at the tail of the buffer queue if there is enough space left. Otherwise, only the importance of these frames are compared and a dropping decision is made.

A similar utility-based frame dropping strategy as in Section 3.3.2.2 is used here for conversational videos. Please note, because of the tight delay constraint, optimization is done only among newly incoming video frames that correspond to a single time instant (one frame slot), i.e., the size of the decision window is equal to one.

3.5 Rate Shaping for Streaming and Conversational Videos

In Section 3.3 and Section 3.4, the side information and dropping strategies for streaming and conversational videos have been discussed, respectively. In this section, the scenario when both types of video pass through the network node simultaneously and share one outgoing link is considered.

3.5.1 Proposed Framework

As shown in Fig. 3.9, the proposed RD-optimizer performs two independent dropping decisions for K incoming streaming video and N conversational video, as proposed in [TCS06]. The surviving (not being dropped) frames are stored in two independent classification buffers. The buffer for conversational video is relatively small in order to limit the forwarding delay experienced by these frames as this type of video application requires low latency. On the other hand, the classification buffer for streaming video is larger due to the more relaxed requirement on the delivery delay in this case. A scheduler is located behind the two buffers, which dynamically assigns the shared resource (forwarding data rate) to the two buffers by fetching video packets from them and putting them into the shared outgoing link buffer.

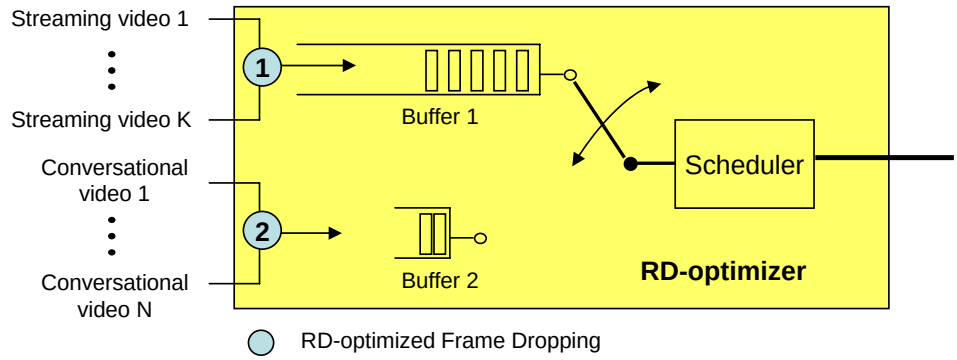


Figure 3.9: Structure of the RD-optimizer for frame dropping of streaming and conversational video

For streaming video, the cost-function based dropping strategy introduced in Section 3.3.3.2 is employed. Conversational video then adopts the dropping strategy introduced in Section 3.4.2. For streaming video, information about future frames is taken into account. When the dropping decision is made for conversational videos, the importance of the current frame can be only compared with previous frames. As the selected frame is first put into the classification buffer, which is assumed to be accessed by the RD-optimizer running on the proxy, frame replacement for this buffer is enabled. When new frames arrive at the node and the classification buffer is full, frames in the buffer with lower utility than the new incoming frame will be marked as dropping candidates. If the buffer space released by dropping these frames is enough to put in a new frame, they are physically dropped from the temporal buffer. On the other hand, if the released space is not enough to hold the new frame, it means the new frame is either too big or is not important enough for the reconstruction quality of the corresponding stream. Then this new frame is dropped and the marked frames in the buffer are recovered.

3.5.2 Scheduling Strategies

Two separate classification buffers are employed to limit the additional delay experienced by the conversational video streams, as explained earlier. Compressed video has a variable bit-rate, and hence fixed resource assignment in terms of forwarding data rate sometimes wastes resources and leads to unnecessary frame dropping. With a dynamic resource assignment in place, the multiplexing of the multiple streams decreases the variation of the bit-rate and provides for more efficient resource utilization. Here, two schemes for dynamic assignment of the data rate on the outgoing link are proposed.

3.5.2.1 Short-term mean rate based scheduling

Compressed video streams are typically VBR, so when the outgoing link provides a transmission rate equal to the mean data rate of the incoming video stream, most likely some packets will be dropped if there is only a very small buffer at the node. But if the assignment is able to adaptively follow the variability of the stream's bit-rate, the node's forwarding resources can be more efficiently used. Without the knowledge of the sizes of future frames for conversational video, the future bit rate can be only estimated, given the knowledge of the incoming data rate history. Here, a straightforward way can be used which is to take F past frames from each stream as an estimation window. The current resource assignment is then calculated as follows:

$$r^i_{SV} = \sum_{k=1}^{K} \sum_{j=i-F-1}^{i-1} R^k_j \tag{3.13}$$

$$r^i_{CV} = \sum_{n=1}^{N} \sum_{j=i-F-1}^{i-1} R^n_j \tag{3.14}$$

$$S^i_{SV} = R_{out} \cdot \frac{r^i_{SV}}{r^i_{CV} + r^i_{SV}} \tag{3.15}$$

$$S^i_{CV} = R_{out} - S^i_{SV}. \tag{3.16}$$

In the equations above, r^i_{SV} and r^i_{CV} are the sum of bytes from the previous F frames of K streaming videos and N conversational videos, respectively. S^i_{SV} and S^i_{CV} represent the assigned transmission rate to the two buffers. R_{out} is the total transmission rate on the outgoing link and it is assumed to be constant during the whole transmission. With the same formulae (3.15) and (3.16), dynamic resource assignment for variable data rate on the outgoing rate can also be accommodated by considering R_{out} to be a function of time.

3.5.2.2 Buffer fullness based scheduling

Buffer fullness based scheduling is an efficient way for the scheduler to avoid buffer overflow. When a buffer is heavily loaded, it means its incoming rate of traffic is bigger than the assigned service rate and therefore new incoming frames are likely to be dropped. In this case, a large portion of the outlink rate should be assigned to this buffer. On the other hand, when one of the two buffers is lightly loaded, it can still hold some new incoming frames. Hence, more transmission slots should be assigned to the other buffer. Furthermore, when the two buffers have roughly the same fullness, it is not efficient to assign the same amount of resource to each of them, as their corresponding incoming rates may differ significantly. This is because the two buffers serve two different types of applications: streaming video and conversational video that usually have different data rates. Hence, a weight is assigned to each buffer according to

their incoming rates, and the forwarding resource is distributed among them based on these weights.

The mean rates calculated with (3.13) and (3.14) represent the most recent (short term) rates feeding the two buffers. Since they vary rapidly over time, employing them to determine the buffer weights may actually be inappropriate in this case. In particular, they may overly influence the resource allocation among the two buffers, thereby rendering their instantaneous fullness less important. Therefore, in order to avoid this effect, (3.17) and (3.18) are employed instead, which supply more stable cumulative mean rates.

The transmission rate assigned to the streaming videos at frame i can then be calculated with (3.19), and the remaining transmission capacity is assigned to the conversational videos.

$$r_{SV}^{i} = \frac{1}{K \cdot (i-1)} \cdot \sum_{k=1}^{K} \sum_{j=1}^{i-1} R_j^k \tag{3.17}$$

$$r_{CV}^{i} = \frac{1}{N \cdot (i-1)} \cdot \sum_{n=1}^{N} \sum_{j=1}^{i-1} R_j^n \tag{3.18}$$

$$S_{SV}^{i} = R_{out} \cdot \frac{r_{SV}^{i} \cdot B_{SV}^{i}}{r_{SV}^{i} \cdot B_{SV}^{i} + r_{CV}^{i} \cdot B_{CV}^{i}} \tag{3.19}$$

Here r_{SV}^{i} and r_{CV}^{i} are respectively the mean incoming rates of the streaming videos and the conversational videos from the beginning until frame $i-1$. B_{SV}^{i} and B_{CV}^{i} denote respectively the fullness in percentage of the two buffers at the time instance when the i^{th} frames of every stream arrive at the node.

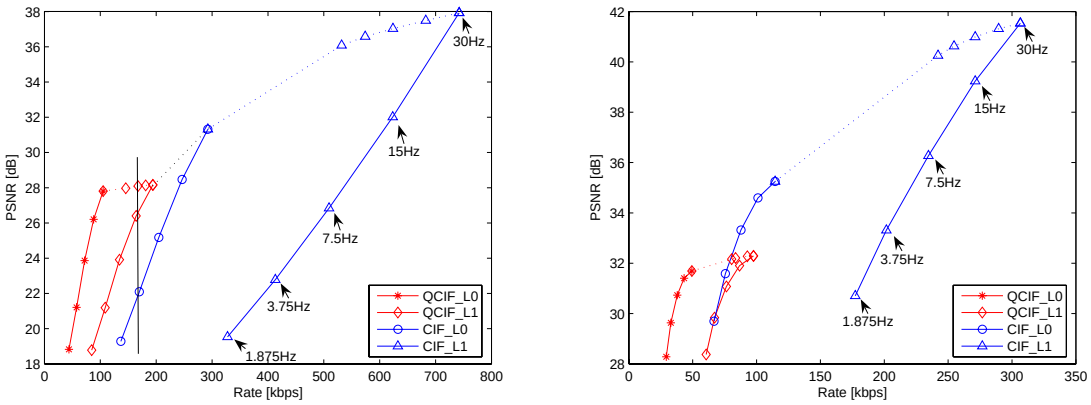
3.6 Rate Shaping for Scalable Video

So far, all rate shaping approaches introduced and proposed above work for the conventional single layer video. Frame dropping in this case only decreases the frame rate and the performance degradation is only in terms of temporal resolution. However, with the development of Scalable Video Coding (SVC), the new coding scheme supplies much more flexible rate shaping choices for pre-encoded video streams. Sub-bitstreams can be extracted at multiple bitrates to fulfill the available transmission resource with a graceful quality degradation. Nevertheless, in current studies of SVC, most of them only consider the rate adaptation for one scalable video stream. Few proposals have been made for the case as shown in Fig. 3.1, where multiple scalable videos share the transmission resource. As different videos have different characteristics, their optimal scaling paths might also be different. In the following, a method is proposed to prepare the RD side information for scalable videos, on which the corresponding RD-optimized rate adaptation approach relies. The side information consists of an optimal combination of scalability modes, i.e., spatial, temporal and SNR scalability for each of the

video streams. The RD approach decides an optimal scaling pattern for every sequence and drops packets that are least important to the reconstruction quality.

3.6.1 Side Information for Scalable Video

The scalable extension described in [SMW06] enhances the H.264/AVC coding scheme to support spatial, temporal and quality/SNR scalability for higher compression efficiency and flexibility. In H.264/SVC, the sender encodes a video into a base layer that corresponds to the minimally acceptable quality and one or more enhancement layers that improve the video quality if received together with the base layer. The coded video data is organized into NAL (Network Abstraction Layer) units, each of which corresponds to a packet containing an integer number of bytes. SVC can make a flexible combination of certain layers by discarding the corresponding NAL units in order to adapt to different bitrates, frame rates or spatial resolutions of the video content.



(a) *Foreman* (b) *Mother&Daughter*

Figure 3.10: Operational RD points for scalable coding of video sequences with H.264/SVC

The scalability levels of H.264/SVC used in this work are illustrated in Fig. 3.10(a) and Fig. 3.10(b), which present the RD performance for the *Foreman* and *Mother&Daughter* sequences, respectively. Each bitstream provides two spatial resolutions (QCIF, CIF) and five different temporal resolutions with frame rates of 1.875, 3.75, 7.5, 15, and 30 Hz. As an effective GOP size of 16 pictures is used for both spatial layers, the lowest supported temporal resolution corresponds to the collection of all key pictures (I-frame of each GOP). Additionally, one quality enhancement layer can be transmitted to provide quality scalability for each spatial resolution layer, which corresponds to a quality improvement that can be obtained by decreasing the QP with a value of 6 in the H.264 JSVM codec [HHIa]. However, this multi-layer concept, referred to as Coarse Grain Scalability (CGS), only provides a limited number of RD points. By using the Medium Grain Scalability (MGS) mode, additional operational RD

points can be easily generated. One quality enhancement layer is now divided into several MGS layers, which are ordered by their importance. Fig. 3.10 gives some achievable rate examples obtained by transmitting some of the MGS layers for the *Foreman* and the *Mother&Daughter* sequences at full frame rate. Although not shown in Fig. 3.10, RD points for the MGS layers at low frame rates can be also obtained easily. The PSNR values of the decoded substreams are measured after reconstructing them to the same temporal and spatial resolution as the original sequence (CIF@30Hz).

As can be seen from Fig. 3.10, different scalability candidates can be selected for a particular rate. For example, in Fig. 3.10(a), different RD points can be obtained at a rate of about 170 kbps (shown with the straight line) with significant differences in PSNR: (1) QCIF resolution at 30 fps encoded with two MGS layers. (2) QCIF resolution at 15 fps with four MGS layers. (3) CIF at 3.75 fps encoded with no MGS layer. The RD side information only consists of the maximum PSNR points at each rate. For different video sequences, the order or the number of scalability points on the envelope might also be different. For instance, the *Mother&Daughter* sequence has very low motion and therefore, the spatial resolution is more important than the temporal resolution. As shown in Fig. 3.10(b), CIF resolution at 15 fps without MGS layer has similar rate as QCIF resolution at 30 fps with all MGS layers, however, it has a much higher PSNR.

The RD side information can be used by the proxy to dynamically decide in a RD-optimized way which packets (NAL units) of which layers of the respective video streams should be dropped in case of network overload. Given all RD points of every sequence, the node selects the optimal scalability modes for all incoming scalable video sequences to meet the available transmission resource.

3.6.2 Rating Shaping Algorithm

When the total data rate of all the incoming video streams exceeds the outgoing transmission rate, video packets have to be dropped. The RD side information sent along with the scalable video bitstreams is used to selectively forward meaningful data which makes the reconstructed video quality as high as possible.

The optimal solution can be found in a brute force way. The algorithm goes through all the possible combinations of operation points among all users and selects the one which has a total rate no larger than the outlink rate while leading to the smallest distortion. Alg. 1 shows the full-search rate shaping process. In this example, the set of all active users $\mathscr{K}$ includes K users. P_k defines the total number of RD points of user k and p_k is the index of the next available RD point for user k which specifies the achievable quality and needed rate. R_{in} is the total rate assigned to all incoming video streams and should be kept no larger than the given outlink rate R_{out}. This full search method can always find the most efficient

Algorithm 1: Full-search Rate Shaping Algorithm

```
begin
    forall k ∈ 𝒦 do
        p_k = 1;
    D_min = ∞;
    forall p_1 = 1 : P_1 do
        forall p_2 = 1 : P_2 do
            ⋮
            forall p_K = 1 : P_K do
                R_in = Rate_1^{p_1} + Rate_2^{p_2} + ··· + Rate_K^{p_K};
                if R_in ≤ R_out then
                    D_temp = D_1^{p_1} + D_2^{p_2} + ··· + D_K^{p_K};
                    if D_temp < D_min then
                        D_min = D_temp;
                        P = (p_1, p_2, ..., p_K);
end
```

Algorithm 2: Utility-based Rate Shaping Algorithm

```
begin
    forall k ∈ 𝒦 do
        p_k = 1;
    R_in = 0;
    𝒦' = 𝒦;
    while R_in < R_out do
        forall k ∈ 𝒦' do
            Calculate U_k^{p_k} = ΔD_k^{p_k} / ΔRate_k^{p_k};
        Choose user k with maximum U_k^{p_k},  (k ∈ 𝒦');
        if R_in + ΔRate_k^{p_k} ≤ R_out then
            R_in = R_in + ΔRate_k^{p_k};
            p_k = p_k + 1;
        else
            repeat
                Choose user k with the next maximum U_k^{p_k};
                if R_in + ΔRate_k^{p_k} ≤ R_out then
                    R_in = R_in + ΔRate_k^{p_k};
                    p_k = p_k + 1;
                    Break;
            until all candidates have been tried ;
            Break;
        if p_k == P_k then
            𝒦' = 𝒦' − k;
end
```

scaling pattern, however, it leads to very high computational complexity as the number of users increases.

A suboptimal algorithm but with much lower computational complexity is the utility-based rate shaping approach. As shown in Alg. 2, the assignment of outlink capacity for a particular user depends on its utility U, defined in $\Delta D/\Delta Rate$. ΔD is the decrease in distortion from the current RD point to the next RD point and $\Delta Rate$ is the corresponding

rate increment in kbps. The ratio of the two represents the quality improvement for a rate unit. The user with the maximum U is picked as candidate and the system checks whether the required rate can be fulfilled. If yes, it assigns the resource with the respective scaling pattern to this user and moves p of this user to the next RD point on the side information curve. If not, it checks the user with the second highest U and calculates if the required rate can be allocated, then the third and so on. This process is repeated until all the users are checked and all the available resources have been optimally assigned to the video streams. If one user has reached his last RD point, no further quality improvement is possible and U equals to zero. In this case, this user is excluded from the following optimization steps to reduce the complexity of the optimization process. Please note that the key pictures from the base layer of any video stream in most cases have a much higher utility than all other parts of the video. Although the algorithm starts with $R_{in} = 0$, it will not happen that these key pictures of any user are dropped unless the outlink capacity is too small to hold all of them.

3.7 Computational Complexity

In this section, the computational complexity and the storage requirements of the RD-optimized frame dropping strategies proposed in this chapter are discussed.

3.7.1 Memory Cost

PRD/PRED are based on static priority labels assigned to every frame, which are included in the bitstream, so there is no additional storage cost for PRD/PRED.

As shown in [TKS04], the Distortion Matrix has $\frac{1}{2} \cdot L_G \cdot \left(3 + \frac{L_G}{L_B+1}\right)$ entries for a GOP consisting of L_G frames with L_B B-frames between two P- or I-frames. However, less entries need to be stored in reality as in the cost function in (3.3), only the overall (cumulative) additional distortion caused by selecting a dropping choice for a current frame is considered. In particular, as explained in Section 3.3.3.2, there are at most four possible dropping decisions that can be made for each frame. Therefore, no more than four distortion values need to be associated to one video frame. Furthermore, given that the additional distortion is zero when nothing is dropped, there are only two remaining choices for which distortion values need to be stored in the case of P- and B-frames, and three such values in the case of I-frames. Hence, the Distortion Matrix can be compacted into $2 \cdot L_G + 1$ entries for each GOP.

When the Hint Track framework based on the DC^0 is employed, frame drop patterns are constructed by considering every video frame independently. Therefore, only L entries for a video stream with L frames need to be stored and sent as side information in the case of Hint Track DC^0. However, when higher order distortion chain models are used in the Hint Track framework, the memory requirements are more demanding. In particular, the number of

distortion values that need to be stored increases polynomially with the order of the distortion chain. For example, $L \cdot (L-1)/2$ entries need to be stored in the case of Hint Track DC^1.

The rate information that needs to be stored is the same in both approaches and comprises the sizes of the video frames, as explained in Section 3.3.3.1 and Section 3.4.1. Hence, there are L rate entries for L frames. Furthermore, for a given drop pattern, the associated rate reduction represents the sum of the sizes of the dropped frames in the case of the Hint Track framework, while for the Distortion Matrix approach, this quantity includes in addition the sizes of all dependent frames.

For SVC streams, the size of the side information is determined by the length of rate incremental step. As shown in Fig. 3.10, if the total number of scalable patterns is P_k, the side information for each possible switching point of a SVC stream consists of P_k pairs of entries.

3.7.2 Computational Complexity

PRD drops the new incoming frame randomly when the buffer can not hold them and PRED starts dropping frames with different priorities when the buffer fullness exceeds the predefined thresholds. Dropping decisions are made frame by frame by checking the buffer level and the priority labels of the frames. Therefore, the computation just lies on the sorting of frames with different priority labels, which in the worst case is $O(L \cdot K \cdot logK)$ for K videos each has L frames.

The cost function based frame dropping strategy for streaming video offers up to four possible dropping choices for every frame, which leads to an upper bound of 4^K drop patterns for K incoming streaming videos. As the distortion and rate saving for every drop pattern need to be calculated to select the optimal one, the computational complexity is very high in this case. However, in the cost function in (3.3), only one $\lambda(n)$ is used at every frame slot. For this reason, minimizing $J(n)$ is the same as minimizing $J_p^k(n)$ separately for each stream. Hence, the cost function can be rewritten as

$$\begin{array}{ll} arg\ \min J(n) & = \sum_{k=1}^{K} arg\ \min J_p^k(n) \\ \text{for } p \in \mathscr{P}(n) & \text{for } p \in \mathscr{P}^k(n) \end{array} \tag{3.20}$$

so that the maximum number of possible drop patterns is reduced to $4 \cdot K$. Including the computation of $\lambda(n)$, the total calculation complexity is $O(8 \cdot K \cdot L)$ for K videos each of L frames length. Please note that this is the worst case that in practice is actually unattainable. That is because frame dropping decisions are only made when the buffer fullness reaches a predefined threshold. Furthermore, dropping decisions affecting future frames reduce the number of prospective drop patterns when the optimization is performed again, at the next frame slot.

With the utility-based approach for conversational video, the individual frames are considered independently for the DC^0 model. Therefore, there are only two possible dropping choices for every frame, to drop or not to drop and the resulting overall computational complexity is $O(N \cdot log(N) \cdot L)$, where $N \cdot log(N)$ is the cost for sorting the importance at every frame slot. In particular, with the classification buffer in the hybrid scenario, assume that W frames are in the temporal buffer that need to be sorted according to their distortion-per-bit utility, the resulting computational complexity is $O((W+N) \cdot log(W+N) \cdot L)$ in this case.

To make the dropping decision for K scalable video streams, the complexity of the Full-search approach is $\prod_{k=1}^{K} P_k$. In the Utility-based approach, the main computation is simply to sort all $P_k \cdot K$ dropping patterns, which equals to $O(\sum_{k=1}^{K} P_k \cdot log(\sum_{k=1}^{K} P_k))$.

3.8 Experimental Results

In this section, the performance of several frame dropping strategies for streaming and conversational videos are to be examined. First, the improvement achieved by the proposed RD-optimized frame dropping strategies introduced in Section 3.3 and Section 3.4 is investigated. Then, the performance of the frame dropping optimizer from Section 3.5 that considers both streaming and conversational videos is evaluated. Finally, the performance of the rate shaping algorithm for the scalable videos in Section 3.6 is also discussed.

3.8.1 Simulation Setup

The single layer videos streams employed in the experiments are encoded with the H.264 MPEG-4/AVC codec [HHIc] with a frame rate of 25Hz. Long test sequences are generated by concatenating several short test sequences. For streaming video, each short sequence is appended at the tail of the resulting long sequence in integer multiples of the associated GOP length. This means that a number of frames at the end of a short sequence may be left out, if its length is not an integer multiple of the GOP size. In Table 3.1, the entries in the first row under the names of the short sequences represent their corresponding lengths in number of frames. For example the sequence *Carphone* is 380 frames long. Furthermore, the entries in each of the following rows, when moving towards the bottom of Table 3.1, represent the relative order of concatenation of the short sequences, for each of the resulting long sequences. For example, the long sequence SV_1_20 represents a concatenation of the short sequences: *Claire*, *Miss America*, *Foreman*, *Claire*, *Carphone*, *Mother&Daughter*, in this order. The test sequences are named SV_X_Y for streaming video, where Y stands for the length of the GOP and X is the index of the video. The number of B frames between two P or I frames is set to be 1 in our experiments. For conversational videos, the name is CV_X. The encoding structure for conversational videos is IPPP... with an INTRA update interval of $M = 18$.

Table 3.1: Construction information of all test sequences

Test sequence	Car-phone	Claire	Fore-man	Grand-ma	Miss America	Mother Daughter	Sales-man	Suzie
# of frames	380	270	400	300	150	320	220	150
SV_1_20	5	1,4	3		2	6		
SV_2_22		5,6		4	3	1	2	
SV_3_24		4	2	6	3	1,5		
SV_4_26		3		4	1	2,6	5	
CV_1	1	3	4			5		2
CV_2	3		1	2,6	5			4
CV_3		2		3,6		1,5	4	
CV_4	6	3			2	4	1,5	

Table 3.2: Encoding characteristics of the test sequences

Name	SV_1	SV_2	SV_3	SV_4	Sum/Avg
mean Rate (kbps)	92.44	67.99	69.55	50.60	280.58
mean Y-PSNR (dB)	38.63	38.32	38.10	38.24	38.33
Name	CV_1	CV_2	CV_3	CV_4	
mean Rate (kbps)	122.07	119.06	67.81	116.42	425.36
mean Y-PSNR (dB)	37.57	37.26	37.36	37.69	37.47

Table 3.2 summarizes the encoding (rate and quality) characteristics of the eight test sequences employed in our experiments. Furthermore, the entries in the last column in Table 3.2 represent respectively the sum of the mean rates and the average PSNR values for each of the two categories: streaming video and conversational video. As shown in Section 3.7.1, one GOP streaming video with L_G frames needs $2 \cdot L_G + 1$ and L_G entries for the distortion and the rate information, respectively. With the assumption that each entry needs two bytes, each frame in SV_1 needs on average 6.1 bytes, which results in 0.152 kbps overhead traffic. Compared to the bitrate of the video stream at 92.44 kbps, this less than 0.2% overhead can be ignored. For the conversational video, the number of distortion entries is even smaller and compared to the bitrate of the video stream the overhead for the side information is insignificant.

In order to avoid the prospective loss of the very first I-frame for every test sequence, these frames are assumed to have been forwarded by the network node and that all dropping decisions are made after the arrival of the second frame of each stream. For this reason, the initial buffer load is set to be 7.5 KB out of 16 KB (total buffer size) in the case of streaming video when the frame dropping process starts. For conversational video, because of the strict delay constraint, the buffer size is set to be 5 KB. Again, the influence of the first I-frames is ignored and the initial buffer load is set to be 0 byte.

In order to evaluate the performance of the proposed RD-optimized rate shaping approach for scalable video, it is compared with a static priority-based rate shaping strategy. Different test sequences are used, as high resolution test sequences are needed to examine the performance of spatial scalability of SVC. Therefore, four video sequences in CIF resolution are employed. All video sequences are encoded using the JSVM software [HHIa] at 30 fps and

have a GOP size of 16 frames. The characteristics of all encoded bitstreams are summarized in Table 3.3. The results are calculated by averaging the reconstruction quality of all the four video sequences.

Table 3.3: Characteristics of test video sequences encoded with JSVM

Sequence	Bitrate Range (kbps)	PSNR Range (dB)
Carphone	41.56-818.72	22.81-38.46
Container	44.30-457.05	23.27-39.04
Foreman	43.84-742.29	18.82-37.93
Mother&Daughter	29.34-306.57	28.29-41.53

3.8.2 Steaming Videos

In this section, the performance of different frame dropping strategies that have been introduced in Section 3.3 are presented individually with different parameter settings, followed by a comparison among all these schemes.

3.8.2.1 Threshold settings for PRED

The implementation of PRED is straightforward and the only important point here is to select the proper thresholds for the random dropping of B-frames (T_1) and P-frames (T_2). The four streaming videos introduced in the previous section are employed as test videos here. Note that no conversational videos are employed in the experiments, as no static priority labels can be established for them ahead of time. The operation of PRED on such content reduces to random dropping without priorities. To get the best performance, all possible values for T_1 are tested from 30% to 100% of the buffer fullness and T_2 is always bigger than or equal to T_1.

In Fig. 3.11, the average reconstruction quality of the four streaming videos is shown as a function of T_1 and T_2 at different outlink transmission rates, which are represented with different surfaces in the figure. In principle, the higher the transmission rate, the higher the reconstruction video quality. However, it can be seen that at low rates, the performance surface is not flat and a big performance drop can be observed when large values for T_1 and T_2 are selected. The performance at higher rates is more stable, as the observed reduction in video quality due to an improper selection of thresholds does not exceed 1$\sim$1.5 dB here. The upper and lower performance bounds of PRED are shown in Table 3.4, which are the highest and lowest points on each surface in Fig. 3.11. The normalized rate is the percentage of available transmission rate versus the mean rate of all users. It can be seen from the table that a large performance gap exists between the two bounds for the case when the transmission rate on the forwarding link is much smaller than the mean aggregate source rate of the videos.

However, it is not easy in practice always to select the optimal thresholds such that the upper performance bound is achieved.

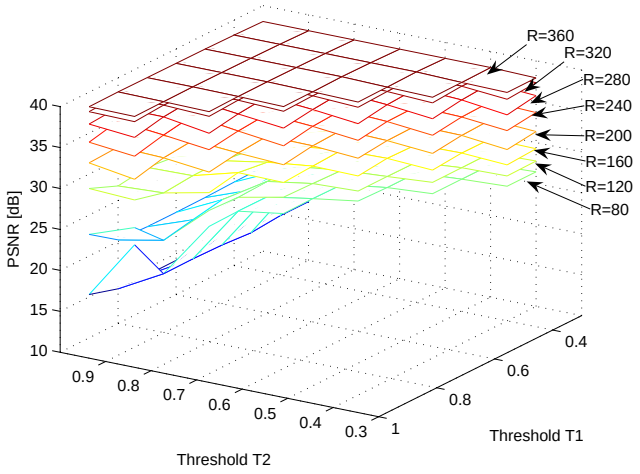


Figure 3.11: Performance of PRED thresholds, R represents the outlink rate in kbps

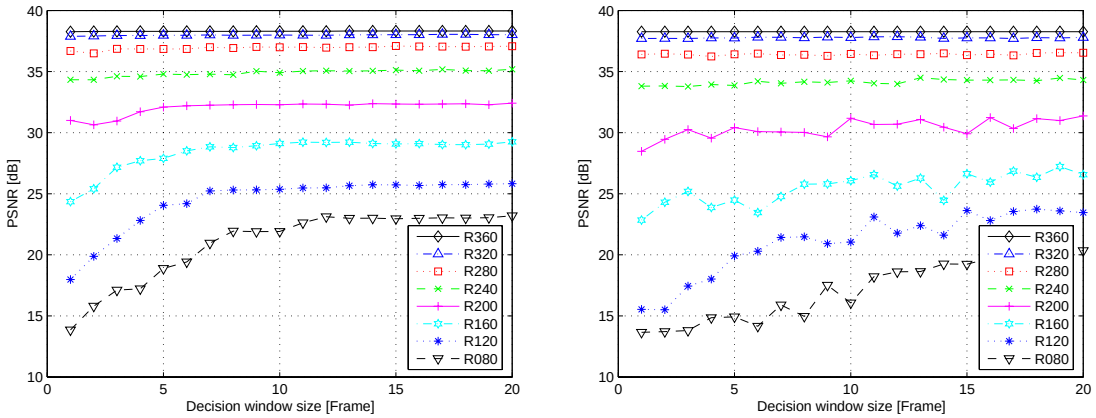
Table 3.4: Performance bounds of PRED

Rate (kbps)	80	120	160	200	240	280	320	360
Normalized Rate	0.285	0.428	0.570	0.713	0.855	0.998	1.140	1.283
PSNR_{max}(dB)	26.43	27.75	29.37	31.91	35.18	36.83	37.82	38.23
PSNR_{min}(dB)	14.03	16.67	21.68	28.17	32.77	34.73	36.50	37.47

As described in Section 3.3.1, different dropping thresholds can be applied for different P-frames depending on their position in a GOP. For example, if the start point for dropping frames is at 50% of the buffer size, and then each successive dropping threshold to be associated with a further increment of 5%, a performance upper bound for the case when only two thresholds are used is achieved. This means that more accurate frame dropping decisions can be made, when finer priority steps in terms of frame dropping are employed.

3.8.2.2 DW/VDW for Utility-based Dropping for Steaming Video

Here, the performance of the utility-based dropping strategy is compared with DW and VDW and how the size of the decision window influences the reconstruction quality of the streaming videos is also investigated. Fig. 3.12(a) shows the reconstruction quality when a DW is used for utility-based frame dropping as a function of the decision window size. The different graphs (curves) in Fig. 3.12(a) correspond to different outgoing link rates. It can be seen clearly that increasing the size of the decision window improves video quality only for low outlink transmission rate values. In particular, by increasing the window size an up to 5 dB improvement is observed when the transmission rate is lower than 160 kbps. This rate value represents 57% of the aggregate data rate of the incoming videos. On the other hand, once



(a) Decision Window (b) Virtual Decision Window

Figure 3.12: Utility-based frame dropping for streaming videos

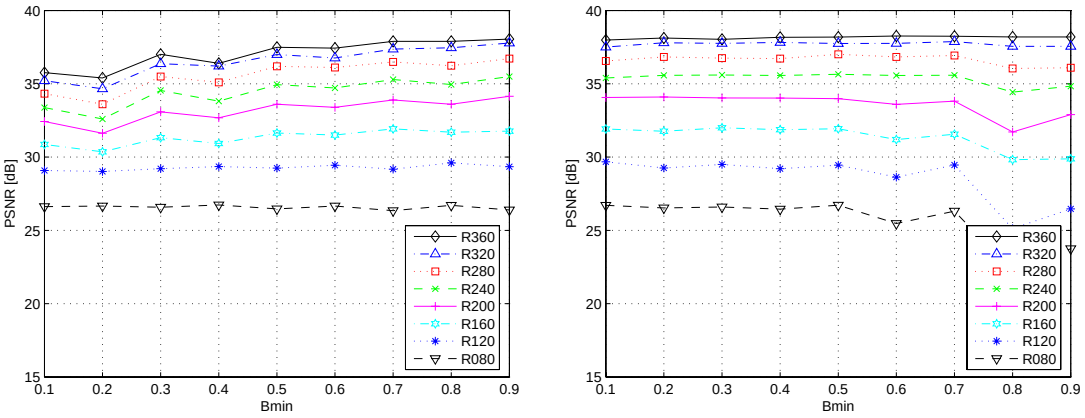
the transmission rate becomes large enough, the resulting frame dropping performance is no longer affected by the varying decision window size. Specifically, when the forwarding rate becomes equal to or greater than the overall data rate of the videos, fewer individual frames are lost, which in turn leads to a very small variation in quality for different window sizes.

Fig. 3.12(b) shows the performance curves for different outgoing link rates as a function of the VDW size. Also in this case, the dominant factor affecting performance is the forwarding rate and the decision window again only helps at lower rates. As seen from Fig. 3.12(b), the performance curves are not monotonically increasing as a function of the window size. They exhibit some fluctuations, which are due to the requirement that in this case frame dropping decisions are only made for the current frames in the VDW. This additional requirement for the VDW case was introduced in order to limit the additional delay introduced when frame dropping decisions are performed over a window of frames, as explained in Section 3.3.2.2. If the performance of VDW in Fig. 3.12(b) is compared with that of DW from Fig. 3.12(a), a performance loss of several dBs is observed at almost all transmission rates that are smaller than the aggregate data rate of the videos.

3.8.2.3 Cost function-based RD-optimized frame dropping

In the following, the influence of λ on the performance of cost function based frame dropping is examined for the two interpolation methods introduced in Section 3.3.3.2. Fig. 3.13 shows the results for the cost function-based frame dropping strategy when using quadratic and linear interpolation for the multiplier λ, respectively. In all simulations, B_{max} is fixed to be 100% of the buffer size.

Quadratic interpolation exhibits a degraded quality when very high values for B_{min} are selected at very low outlink rates, as shown in Fig. 3.13(b). This is because quadratic interpo-



(a) Linear interpolation (b) Quadratic interpolation

Figure 3.13: Performance of cost function based frame dropping

lation leads to aggressive frame dropping decisions when the buffer fullness approaches B_{max} and is far away from B_{min}. Setting B_{min} to be bigger than 0.8 results in late dropping of less important frames and which in turn causes unnecessary loss of some frames with high importance. The curves are smooth and flat when B_{min} is smaller, as the dropping decision is very moderate when the buffer is lightly loaded. When linear interpolation is used, small values for B_{min} at high outlink rates lead to unnecessary dropping of some frames with low importance. To summarize, selecting B_{min} larger than 0.5 is fine for linear interpolation and for quadratic interpolation, B_{min} should be selected smaller than 0.6. By selecting B_{min} between 0.5 and 0.6 good results are obtained for both schemes.

3.8.2.4 Performance Comparison Among All Frame Dropping Schemes for Streaming Video

In this section, the performance of the frame dropping schemes for streaming video is compared as a function of the forwarding data rate on the outgoing link. In Fig. 3.14, *PRD* denotes the pure priority based frame dropping, which already performs better than the real network node in the Internet, where no proxy is employed. *PRED* here fixes the thresholds T1 and T2 to be 70% and 90%, respectively, of the buffer fullness, while the *PRED_UB* curve in Fig. 3.14 corresponds to the upper bound from Table 3.4. *U_DW* represents the utility-based frame dropping using DW and *U_VDW* illustrates the performance of utility-based frame dropping using VDW. Both of them use a decision window size of 20 frames, which leads to best performance. The proposed RD-optimized cost-function based dropping strategy that uses Distortion Matrix as side information is shown as the *CF_DM* curve in the figure, where B_{min} is selected to be 0.6.

PRD performs the worst at all the rates, as can be seen from Fig. 3.14. *PRED* also

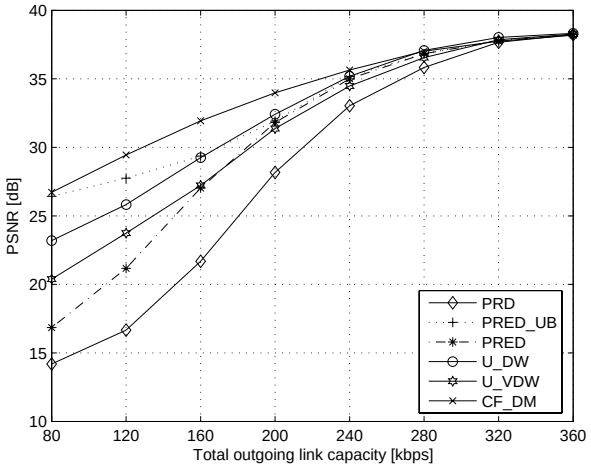


Figure 3.14: Performance comparison of frame dropping schemes for streaming video

shows a poor performance at low link rates, while *PRED_UB* performs better than *U_DW* and *U_VDW* at low link rates, which is due to two factors. The first is the proper selection of dropping thresholds and the second is the inaccurate distortion estimation of the Hint Tracks DC^0 model at low transmission rates. *CF_DM* outperforms all other schemes as a result of its accurate distortion estimation and dynamic adjustment of the dropping aggressiveness according to the buffer fullness level.

3.8.3 Conversational Videos

In this section, the utility-based frame dropping for conversational video is compared with the pure random dropping in a round robin fashion. When a video packet arrives, if the outgoing link buffer can still hold it, the packet is put into the buffer, otherwise, this packet is simply dropped. For the utility-based approach, when N new incoming frames arrive at the node and the buffer can not hold all of them, they are sorted according to their utility and put into the buffer one after another until the buffer is full.

Table 3.5 shows the averaged PSNR values for the four test conversational videos at different outgoing link rates. The mean score of the four videos (boldfaced numbers) presents the overall reconstruction quality. The utility-based frame dropping outperforms the random frame dropping in the range of middle to high rates, because at very low rates, consecutively dropping of a large number of frames lead to an inaccurate estimation of distortion. However, the performance of individual users shows that it is more fair by using the utility-based approach (maximum difference from 0.9∼5.5 dB) compared to the pure random dropping approach (maximum difference from 3.8∼10 dB). Therefore, in addition to the overall quality, the proposed approach also shows a good characteristic with respect to the fairness among users.

Table 3.5: Comparison of utility-based dropping and random dropping

Outl. rate (kbps)	120	180	240	300	360	420	480	540
Norm. rate	0.282	0.423	0.564	0.705	0.846	0.987	1.128	1.270
Utility-based Frame Dropping								
CV1(dB)	15.01	19.60	22.53	27.23	29.96	32.20	34.01	36.07
CV2(dB)	15.60	21.07	23.30	28.46	31.11	33.03	35.12	36.45
CV3(dB)	20.50	21.30	23.34	27.23	30.39	32.76	34.25	36.06
CV4(dB)	18.52	21.37	25.12	27.83	31.40	34.02	35.82	36.93
Mean(dB)	**17.41**	**20.83**	**23.57**	**27.69**	**30.71**	**33.00**	**34.80**	**36.38**
Random Frame Dropping								
CV1(dB)	13.68	19.96	21.09	25.25	26.57	29.75	31.71	35.17
CV2(dB)	16.40	16.64	21.15	23.52	28.93	28.60	34.43	33.34
CV3(dB)	16.44	26.26	26.25	29.17	31.69	34.99	34.11	37.02
CV4(dB)	23.67	24.49	28.75	29.97	32.61	33.84	36.38	37.21
Mean(dB)	**17.55**	**21.84**	**24.31**	**26.98**	**29.95**	**31.79**	**34.16**	**35.69**

3.8.4 Streaming and Conversational Videos

In this experiment, the proposed joint optimizer for streaming and conversational video is compared with a reference scheme that uses PRED and round robin (PRED/RR) for these two types of video applications, respectively. In particular, streaming video provides three types of static priority labels, as explained earlier. Therefore, in the reference scheme PRED can be employed on the streaming videos by early dropping of B- or P-frames. On the other hand, for conversational video, all the frames, except the very first one, are P-frames and hence there is no static priority difference among them. Therefore, when multiple frames arrive simultaneously at the network node, a simple round robin scheme (over the conversational videos to which these frames belong) is employed to determine how many of them can be placed into the corresponding buffer for conversational video.

The proposed optimizer uses the Distortion Matrix for streaming video and Hint Tracks for conversational video. In the case of conversational video, (3.11) and (3.12) are employed to estimate the overall distortion associated with the dropping of a single frame, as explained in Section 3.4.1. In (3.11) and (3.12), the equivalent intra update period M is set to be 18 frames and the attenuation factor γ is set to be 0.997. Finally, for comparison purposes the hypothetical case is also considered when the distortion incurred by dropping frames can also be pre-calculated for conversational video.

Fig. 3.15 shows the performance improvement achieved by the proposed RD-optimized strategy for dropping frames from both streaming and conversational videos. Several instances of the proposed optimizer are considered in Fig. 3.15. In particular, *RD_FIX* denotes the proposed optimizer with fixed resource assignment of 40% to the streaming videos and 60% to the conversational videos. Note that this assignment corresponds to the overall average data rates for these two types of videos. Furthermore, *RD_BUF* and *RD_RAT* in the figure represent the buffer fullness based and the short term mean rate based scheduling strategies

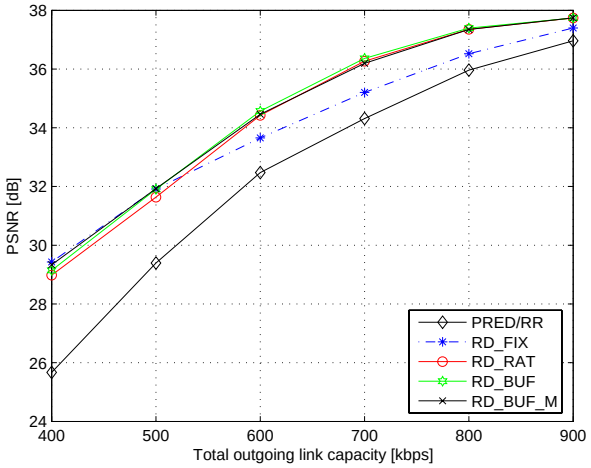


Figure 3.15: Performance comparison of the proposed RD-optimizer and PRED/RR for streaming and conversational videos

introduced in Section 3.5.2, respectively. In the case of *RD_RAT*, F in (3.13) and (3.14) is set to be 10 frame slots in this experiment.

First, it can be seen that when the outgoing link rate is larger than the mean incoming rate (when the normalized rate is larger than 1), the performances of the RD-optimizer and PRED/RR are similar. However, there is still a performance improvement of 1 dB at 900 kbps. This is because even at this rate, frame dropping from the conversational videos needs to occur in PRED/RR, whenever the incoming data rate of the video streams peaks, as the small buffer for conversational videos can not hold too many frames at once. The RD optimizer deals more successfully with this situation, since the optimized frame dropping has more opportunities to drop the least important frames even if they have been in the classification buffer waiting to be scheduled. At the same time, the dynamic resource assignment saves away some spare transmission slots from the streaming videos, that can be appropriately reallocated to the conversational videos afterwards, as explained in Section 3.5.2. When the outgoing rate is smaller than the total traffic rate, an improvement of around 3 dB is observed, as shown in Fig. 3.15.

Table 3.6: Assignment of forwarding date rate

Total_Rate(kbps)	200	300	400	500	600	700	800	900
Normalized Rate	0.283	0.425	0.567	0.708	0.850	0.992	1.133	1.275
As._Rate_SV(kbps)	72.86	104.82	140.40	180.10	227.64	258.44	272.88	279.18
(%)	(.36)	(.35)	(.35)	(.36)	(.38)	(.37)	(.34)	(.31)
As._Rate_CV(kbps)	127.14	195.18	259.60	319.90	372.36	441.56	527.12	620.82
(%)	(.64)	(.65)	(.65)	(.64)	(.62)	(.63)	(.66)	(.69)

Furthermore, at low rates, the performances of *RD_FIX*, *RD_BUF* and *RD_RAT* are almost the same. However, at high rates the schemes with dynamic resource assignment

perform much better, because reassigning some of the resources from the streaming video buffer to the conversational video buffer will not influence the quality of streaming video significantly, as these resources are typically saved when the low data rate sections of the incoming streams occur at the node. But with fixed resource allocation, these unused resources from the streaming video are wasted, which leads to degraded performance at high outgoing link rates compared to the case of dynamic resource assignment. The last two rows in Table 3.6 give the assigned transmission resources to the streaming videos and conversational videos when the buffer fullness based scheduling strategy is used, respectively. More transmission resources are assigned to the conversational videos compared to their mean bitrates. This is the consequence of the small size of the classification buffer due to the tight delay constraint of the conversational videos.

Finally, pre-computed Hint Tracks for conversational video are not available in practice, but here they are calculated anyhow in order to examine if the approximation from (3.11) and (3.12) leads to accurate results. Experiments show that pre-computed Hint Tracks (RD_BUF_M) for the conversational videos and the approximation (RD_BUF) obtained using (3.11) and (3.12) lead to almost identical performance results, as can be seen from Fig. 3.15. The estimation bias from the model in (3.11) and (3.12) does not affect the results, because the relative values of the distortion-per-bit utility among the individual frames are preserved in either case.

3.8.5 Scalable Video

So far, the performance of the proposed rate shaping algorithms for conventional single layer video has been examined. In this section, the performance of RD-optimized rate shaping for scalable video is to be shown.

The proposed RD-optimized rate shaping approaches are compared with a scheme that uses static priorities for SVC videos, whose working principle follows the fixed priority of NAL units. The base layer is the most important layer as all higher layers depend on it for decoding. The respective MGS layers have the next level of importance. The spatial enhancement layer comes next in the priority order followed by their MGS layers. For static rate adaptation, MGS layers of the enhancement layer (lowest priority) are dropped one after the other according to the importance among them. The spatial enhancement layer is dropped next in the same spirit. Eventually, the MGS layers of the base layer are dropped next followed by temporally dropping the frames in the base layer if congestion persists. For multiple SVC videos, only when all units with the lowest priority label are dropped, the packet dropping of next priority level can be started.

Fig. 3.16 illustrates the achievable improvements in averaged reconstruction quality by the proposed RD-optimized approaches. *SVC-PR* stands for the priority-based approach.

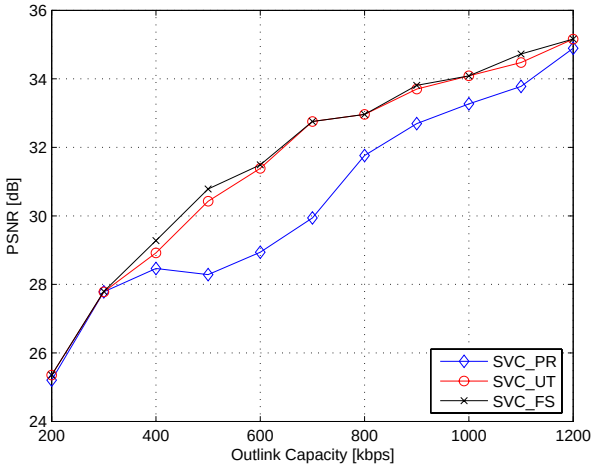


Figure 3.16: Performance evaluation of SVC

SVC-UT is the proposed utility-based approach and *SVC-FS* shows the highest achievable performance through the full-search method in a brute force way. *SVC-UT* achieves a performance improvement of up to 2.5 dB compared with *SVC-PR* using the static priority labels. In comparison to *SVC-FS*, *SVC-UT* achieves a very close performance but much lower complexity as discussed in Section 3.7.2. This shows that the utility-based approach is able to find in most cases the optimal scaling paths for all involved videos.

For a given outgoing rate, the RD-optimized approach searches for the set of rate-distortion points which maximizes the sum of the PSNR values for all users. However, it might result in unfairness among different video sequences because of their different characteristics. To overcome this problem, the overall PSNR is maximized based on the precondition that a minimum reasonable quality should be achieved for all the video streams. In order to reach the required quality, appropriate operational RD points should be determined and corresponding resources are assigned to all users. If the system can not provide the needed resources, the quality has to be reduced. On the other hand, if there is still some transmission rate left, it will be assigned to users in a similar way as described in Alg. 2 (See Section 3.6.2).

Fig. 3.17 shows the result of the approach considering a basic quality of each user. *L0T0F0* means for every user, at least the key frame of the base layer should be guaranteed. *L0T4F0* and *L0T4F1* stand for the cases where the whole base layer at full frame rate with and without all corresponding MGS layers should be transmitted, respectively. *L1T4F0* guarantees that the video is in CIF resolution and at full frame rate. *L0T0F0* is identical to the *SVC-UT* curve in Fig. 3.16 and can be considered as the performance upper bound of the utility-based approach. This is because the key frames of the base layer will always be included first for their high utility as mentioned in Section 3.6.2. This rule is also true for *L0T4F0* because of the high utility for sending the whole base layer. However, the curve starts later since the quality

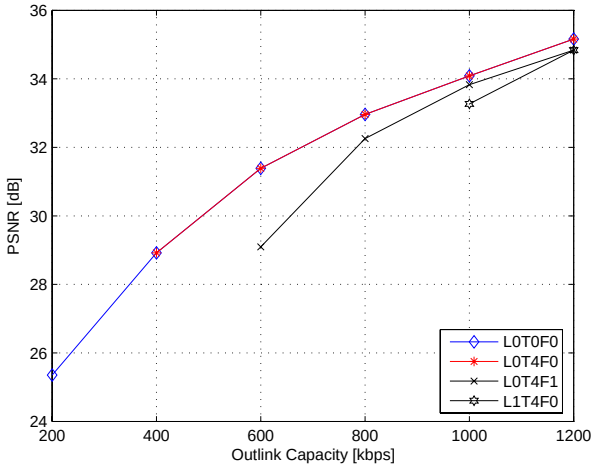


Figure 3.17: Fair RD-optimization achievable with SVC

requirement can not be fulfilled if the transmission rate is below that minimum requirement. *L0T4F1* and *L1T4F0* have some slight performance loss compared with the utility-based approach with full freedom. However, in return, some basic QoS can be achieved. The higher the requirement for the base quality, the higher the requirement in fairness and as expected the bigger the gap from the upper bound. Please note that different minimum quality constraints for even finer layers (e.g., MGS layers) or a lowest PSNR value can easily be added to this framework.

3.9 Chapter Summary

In this chapter, the scenario that multiple video streams passing through a congested network node is considered. Novel methods are proposed to construct the RD side information of packetized video content during encoding, which is sent along the video streams. Based on this side information and low layer information (e.g., mean traffic rate, mean transmission capacity, the buffer fullness level), RD-optimized frame dropping strategies for streaming and conversational video applications are presented. When both types of video application are involved, a so called RD-optimizer is proposed. Different types of videos are shaped with corresponding dropping strategies and resource assignment schemes are also proposed to assign the shared transmission rate to different types of videos. Instead of pure temporal scalability of conventional video, scalable video provides temporal, spatial and also quality scalability. Therefore, rate shaping proposals for scalable video are also presented. The storage cost and the computational complexity of all comparison and proposed schemes are compared and discussed. In the simulation part, the performance of the reference schemes is extensively investigated for more comprehensive comparison. Simulation results show a

significant improvement in video quality is achieved over previous approaches, by a judicious selection of side information and optimized rate shaping strategy.

Chapter 4

Popularity-Aware Partial Caching

In this chapter, the benefits and use cases of proxies for caching management in Video on Demand (VoD) systems are studied. As shown in the literature, proxies are widely used for web services and also for videos applications. Because of the limited caching size on the proxy and the huge volume of video objects, only a small portion of them can be cached. Efficient caching content selection and management is essential to the quality of VoD services. Taking the initial delay as the main criteria for quality evaluation, a dynamic segment-prefix structure and the corresponding caching management algorithm are proposed, which trade off between the initial delay and the deviation of starting point to achieve the highest user satisfaction.

4.1 Introduction

VoD systems allow users to select and watch video over a network as part of an interactive entertainment system. Most of the VoD systems "stream" content, where video is consumed while being delivered. The main benefits of streaming video is that the users do not need to spend a long time and large storage cost to download the whole video file to the local disk. However, it also has much higher requirements to the underlying networks, for instance, the start up latency, the delay jitter, etc.

Proxies are widely used in web browsing and can also be employed to improve the performance of video streaming. By deploying a proxy server close to the client (e.g., on the gateway), video playback with low latency and reduced network traffic can be achieved. This chapter considers the classic scenario for VoD as shown in Fig. 4.1, where a streaming proxy is located on the gateway, through which all clients in the LAN connect to the Internet. It can be assumed naturally that the connection between the clients and the proxy is characterized by high transmission rate and low latency. If the requested video content has already been accessed by one user and is cached on the proxy, the initial delay for the second and later

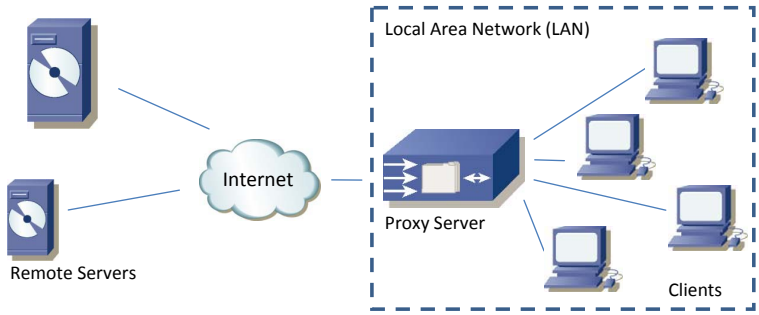


Figure 4.1: Server-Proxy-Client network structure for VoD applications

users is significantly decreased compared to the case when content has to be loaded from the remote server. Since the proxy has only finite storage capacity, a dynamic cache management scheme has to be employed to decide which videos or which parts of a video to cache. An optimal caching algorithm should result in

- **Low initial delay** The system should react to the user with a short delay. Most frequently accessed content should be cached on the proxy and therefore can be served with very low delay.
- **Low traffic** The proxy caches the most popular content so that a large amount of user requests can be served locally. This avoids downloading the same content for many times from the remote server, which saves the traffic between the proxy and the server.
- **VCR functionality** The users are able to interact with the system to selectively view the most interesting content to them instead of pure passive watching. Accessing granularity finer than a whole video should be provided.
- **Small client buffer** The delay jitter should be compensated by the proxy instead of the clients. The proxy decides when and which frame to be sent to a user, so that a very small buffer is needed by the clients (e.g., several frames). This is particularly important if the clients are handheld devices, where additional memory is very expensive.

To address the above mentioned issues, a number of studies have been performed to explore the benefits of a proxy server for video streaming applications. Caching algorithms are proposed to minimize the data traffic between the server and the clients. For instance, *Video Staging* approaches [ZWDS00, MD02, CCHO05] cache the video portions on the proxy that exceed the given transmission rate so that a lower bandwidth is required to serve the video over a CBR channel. In [JBI03], both the characteristics of the video objects and the quality of the connections between the servers and the proxy are taken into account to determine the optimal video portions for caching. Miao and Ortega propose in their work [MO99, MO02] caching strategies that consider an efficient control and usage of the client

buffers. Buffer underflow and overflow are avoided with small initial delay. Both issues, the data traffic and the client buffer control, have been discussed and possible solutions are given by Oh and Song in [OS06, OS07], where frames are selectively cached so that the normalized client buffer size is minimized. Furthermore, approaches for transcoding proxy caching are presented in [LTS05, QLS+05, KL07], where the proxy caches different versions of the same video content to deal with heterogeneous user requirements.

Several partial caching approaches have been developed to decrease the initial delay experienced by a VoD user. Video content can be temporally divided into small units and some of these units are cached on the proxy to enable a fast playback. *Prefix Caching* [SRT99, PLC01] caches only the frames at the beginning of popular video clips to minimize the average initial delay. Segment-based approaches have been proposed to enable the cache update with a finer granularity. *Exponential Segmentation* [WYW01] divides the video object such that succeeding segments double in size and the replacement always starts from the largest segment and an extended version of it namely *Skyscraper Segmentation* is introduced in [WYW04]. The *Lazy Segmentation* approach in [CWZ+05] determines the segment length of a new incoming video as late as possible according to the user access record. The video is then divided into same length segments according to the mean viewing length and only the first several segments are cached when replacement is performed to free cache space. Segmentation of video objects can be in the time domain but also in quality direction. For instance, in [RK01] and [LCX04], caching algorithms for multi-layer (scalable) video are proposed. A real implementation of the segment-based proxy caching infrastructure is reported by Chen *et al* in [CSWZ07] to show the feasibility of this approach. *Chunk level cache replacement* methods further improve the flexibility of cache management. [HNG+99] combines neighboring units to form chunks, each consisting of some prefix segments and some suffix segments. Suffix segments are never cached and the dropping of prefix segments starts from the tail of each prefix. The length of the chunks is predetermined according to the popularity of video objects. A dynamic chunk size scheme is proposed in [BS03], where the chunk length in one video is fixed and is always an integer multiple of current cached units. A larger chunk size is assigned to the videos with higher popularity and the replacement always starts from the end of the chunk having the smallest size. All above approaches assume that the playout always starts from the beginning of the video towards the end sequentially and random access to video content is not considered. This issue is addressed in a recent work by Wang and Yu [WY07], where a fragmental caching structure is proposed to enable interactive VCR functionality.

Popularity is one of the most important factors that should be considered during the design of an efficient caching strategy. Long-term movie popularity model in VoD systems has been derived in [GBW97], which shows how the popularity of video objects changes with time. In [ASP98], real access log files show that for VoD services the popularity between video objects

follows a Zipf Distribution [Zip49], which says that most of the user requests are focused on a limited number of video objects. Different ways of calculating popularity are introduced and compared in [YL03]. *Static* counts the hitting frequency of a daily or weekly log file. *Accumulated* takes the history popularity into account and assigns different weights according to their time distance. *Dynamic* uses a sliding window to count the hitting rate in a short period of time. Popularity difference exists not only between video objects, but also inside one video. For instance, [YCDW06] reveals the internal popularity distribution of a video object by studying the log file of a commercial VoD system. Users start watching from the beginning of a movie sequentially and stop somewhere if the movie is uninteresting to them. In [ZSL05], the log files of a VoD system in the university also shows the diversity of popularity inside one video file. However, it is reported that most of the users access the video randomly rather than using sequential playout. High access rates are always observed at the most attractive parts of the movie. Because of the random access, two popularity distributions should be considered, namely playing frequency and seeking frequency. As shown in [ZSL05], the two distributions are similar and the peaks are consistent.

Popularity-aware caching strategies, as described for instance in [JBI03, PLC01, YL03], use the popularity of served videos to decrease the initial delay and to improve the caching efficiency. Videos with high popularity are more likely to be cached and those with low popularity are not cached or only a small part is cached. Segment-based partial caching with explicit consideration of video popularity can significantly improve the performance of VoD systems as shown in [WYW01, WYW04, CWZ+05, RK01, LCX04, CSWZ07, HNG+99, BS03]. However, most of the proxy caching algorithms assume that video files are always played from the beginning and continuously towards the end. Therefore, they put emphasis on the beginning of video files, while the rest receives less attention. Generally, this assumption does not always hold in practice. As shown in [ZSL05], for VoD services most of the video content is randomly accessed instead of being played sequentially. Although [WY07] enables random access, only video level popularity is considered, which prevents a further improvement of caching efficiency.

In [STS07], a Popularity-Aware Partial cAching (PAPA) algorithm has been proposed. This approach is based on the results of an online subjective test, which has shown that VoD users prefer immediate feedback from the system even if the video does not start playing at exactly the desired starting point. Therefore, in PAPA, video files are divided into fixed-length segments (similar to “chunks” in [HNG+99] and “fragments” in [WY07]), consisting of a prefix and a suffix. The prefix length in number of GOPs is determined by the network condition (RTT, rate, delay, jitter, etc.) between the proxy and the server where the requested video is stored and is chosen such that it is big enough to ensure continuous playback at the client. As every GOP is independently decodeable, the GOP size determines the granularity for random

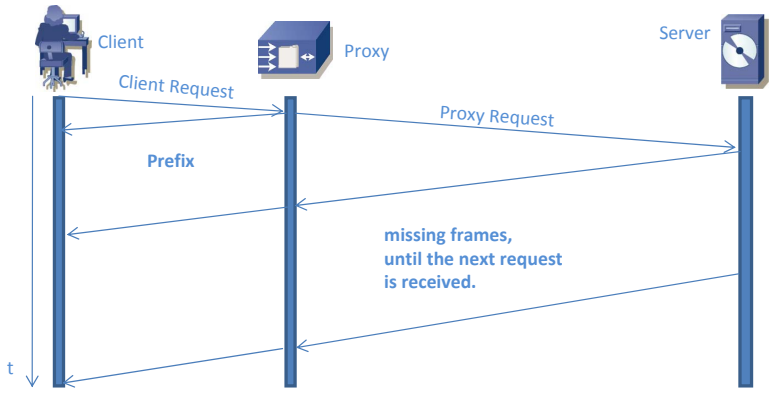


Figure 4.2: Fast playback with prefix caching

access. However, the playout in PAPA always starts from the beginning of a segment that a user request falls in. This leads to a small deviation of the starting point (called "early start" in this work), but as long as the prefix of the segment is completely available in the cache, the playback can start immediately. As shown in Fig. 4.2, when a request arrives at the proxy, the prefix of the segment where the requested GOP belongs to is forwarded to the user immediately if it is cached on the proxy. Meanwhile, a request is sent to the content server for missing video content. Downloaded frames are forwarded to the user when the whole prefix has already been sent. Furthermore, instead of evaluating the popularity of the whole video file, the popularity of every segment is evaluated, which leads to a more accurate popularity analysis and more flexible cache updates. When cache space has to be freed, PAPA drops all suffix frames before dropping any prefix frame to avoid waiting time for the client. Although the initial waiting time is minimized as long as all prefix frames are available, each segment has the same expected early start time without considering the large difference in their popularity. This limits the performance of PAPA.

All above "prefix" related approaches decrease the initial delay by starting the playout always from a cached prefix. However, the effect of this constraint to the users has not been considered. In this dissertation, an improved version of subjective test environment is developed and more representative results are obtained, which further support the conclusion from the preliminary tests in [STS07] that users prefer a small starting point deviation compared to initial delay. Moreover, A mathematical expression for the relation between the initial delay and the early start time with respect to user satisfaction is conducted. The proxy cache is divided into two parts, namely Level one (L1) cache and Level two (L2) cache. All newly incoming video fragments are cached in the L1 cache following the LRU (Least Recently Used) rule and periodically the L2 cache is managed with the proposed Dynamic sEgment-based Caching Algorithm (DECA). DECA inherits the concepts of "segment-prefix structure", "internal popularity of video" and "early start" from PAPA and further improves them. In

DECA, instead of using a fixed segment/chunk size as in [WY07] and [STS07], the size of each segment is variable and changes as a function of the dynamically updated popularity. The individual segment size is determined by the estimated weighted user satisfaction, which is calculated from the popularity, the initial delay and the early start time of all GOPs belonging to this segment. Furthermore, DECA considers both the contribution of initial delay and early start, and makes a trade-off between them to achieve the highest possible average user satisfaction.

The remainder of this chapter is organized as follows. In Section 4.2, the subjective test platform as well as the obtained results are introduced. A general two-level proxy caching framework is proposed in Section 4.3. Based on the subjective test results, the proposed DECA approach is described in detail in Section 4.4, which can be employed in the proposed framework. The evaluation metric is given in Section 4.5 and corresponding simulation results are shown in Section 4.6. Finally, Section 4.7 summarizes this chapter.

4.2 Subjective Tests

Based on the results from the initial subjective tests reported in [STS07], the conclusion can be drawn that users prefer early start to initial delay. PAPA makes use of this observation and allows for early start to minimize the waiting time under the assumption that the segment length is fixed and small. However, using short segment lengths also means that more content needs to be cached, which decreases the caching efficiency. In this section, new subjective tests are carried out, which allow us to determine the expected user satisfaction as a function of initial delay and early start time. Equipped with this functional relationship, the proxy is able to serve a user request with the mode which leads to higher satisfaction. Furthermore, in the initial tests, video sequences have been shown to the subjects explicitly with the expected starting point and the real starting point, which is somewhat unfair to the early start mode. In the new test, the interactivity with the system is improved by adding the functionality of random access using a slide bar.

4.2.1 Test Setup

To develop a media player with full functionality and friendly user interface, the open source Video LAN/VLC .Net bindings [Vid] is used as a core player and C# is used to develop the interface. The media player, shown in Fig. 4.3, is called "TUMPlayer", which has the functionalities of *Play*, *Pause*, *Stop*, *Fast Forward/Rewind*, and controllers for *Volume*, *Thumbnail View* and *Full Screen*.

Three videos with different characteristics have been included in the tests: *News*, *Sport* and *Movie*. The *News* is freshly captured from CNN TV news. The football match in

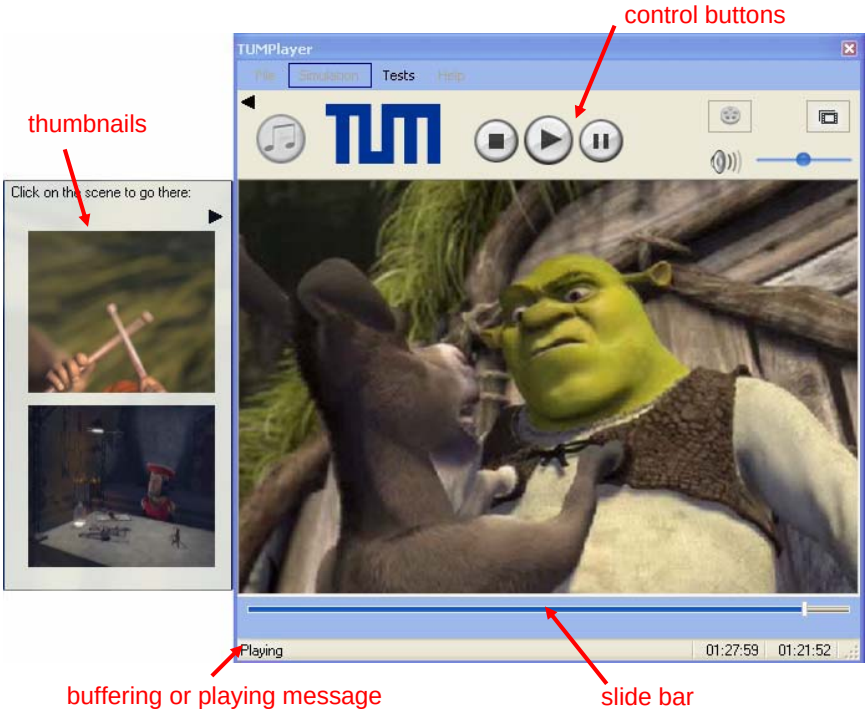


Figure 4.3: User interface of the TUMplayer

the *Sport* category is the final of "Copa Libertadores 2007". Finally, the film "Shrek-I" is used to represent the *Movie* category. All three video clips are encoded using MPEG-1 with CIF@25fps. More properties of the test videos are shown in Table 4.1.

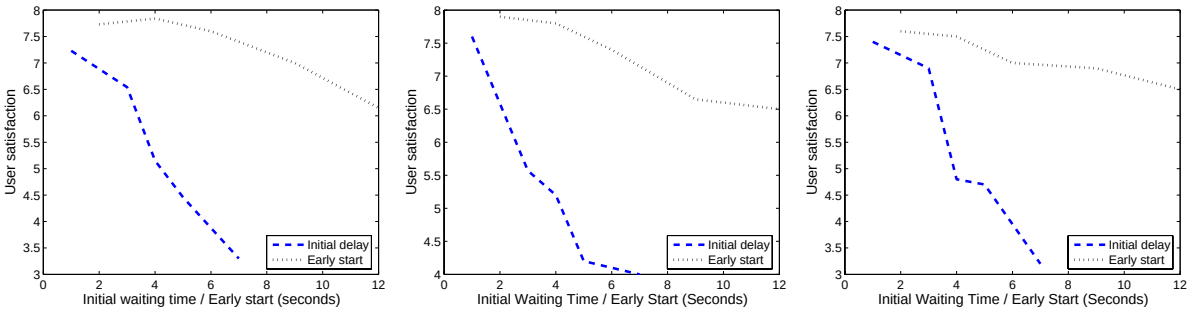
Table 4.1: Properties of the videos for subjective tests

Video Name	Total number of frames	Duration
News	39,504	00:26:20
Sport	72,441	00:48:17
Movie	131,976	01:27:59

The subjective test is conducted as follows. When a volunteer starts the test, two frames selected from two popular scenes are popped up on the left side of the player as shown in Fig. 4.3. The test person is told to access the popular scene by clicking on the pop-up thumbnail frames. After clicking, according to our specification, the system serves randomly with one of the following two modes with a given parameter:

1. **Mode I** – The playout of the video starts after an initial delay of 1, 3, 4, 5 or 7 seconds exactly at the selected scene. In addition, a "Buffering..." text message is displayed in a panel at the bottom of the player.

2. **Mode II** – The playout starts immediately without any initial delay but with a shift of 2, 4, 6, 9 or 12 seconds prior to the selected frame.



(a) News (b) Sport (c) Movie

Figure 4.4: Results of the subjective VoD performance evaluation test

The test person is asked to wait until the scene they clicked on appears in order to make sure that the early start becomes noticeable. After watching the given scenes, the test person is also asked to access the video clip randomly using the slide bar at the bottom of the player to get an additional impression about the interactivity of the system. Finally, a score is given by the test person for one video sequence under one operation mode, which represents his/her satisfaction with the system.

4.2.2 Results

Fig. 4.4 shows the obtained results of our subjective tests. The X-axis represents the initial waiting time or the early start time in seconds. The Y-axis shows the user satisfaction on a scale from 1 to 10, where 1 indicates the worst user experience and 10 is the best. All points on the curves are derived by averaging the scores from 28 test persons. The dashed curve represents the client satisfaction as a function of the initial delay and shows that the satisfaction of the user declines rapidly with increasing waiting time. The dotted curve illustrates the user satisfaction as a function of the early start time and shows a much slower degradation of user satisfaction. Hence, the conclusion can be drawn that users are more comfortable with the early start than the initial delay. In other words, when clients are searching for some particular content, they pay more attention to the initial delay and a small shift of the starting point is more tolerable. Please note that the duration of the movie and the slide bar navigation play an important role in the tests. Following the design of real VoD systems [ZSL05], long videos are employed. As the size of the slide bar in the player is fixed, the shift of the slider for the same playout duration varies according to the length of the video. A very small movement is observed in this case, which makes the early start unnoticeable during slide bar interaction. On the contrary, clips with very short duration will show a large movement of the slider on the track bar, which is then annoying as early start becomes observable when searching with the slide bar. In this case, coarser access granularity

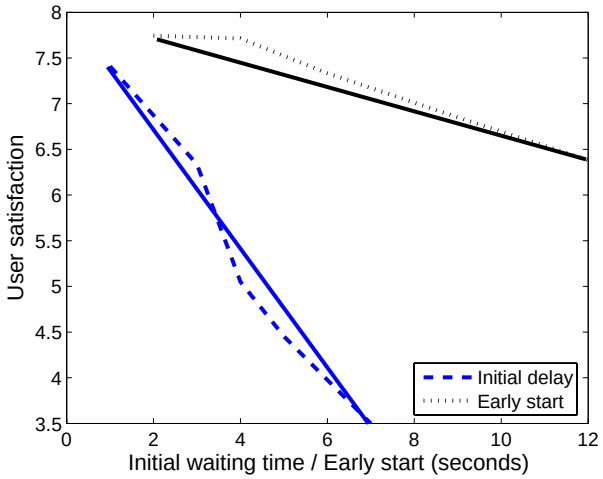


Figure 4.5: Averaged user score and approximated user satisfaction model

on the segment level can be offered by disabling the requests to later GOPs in the segment.

Another observation from Fig. 4.4 is that different types of videos lead to similar results. This shows that a universal user satisfaction model can be derived which is independent of the video type. By averaging the results for the *News*, *Sport* and *Movie* videos, the new user satisfaction curves can be obtained as shown in Fig. 4.5. The curves are further approximated in the following with two linear functions:

$$G_{WT} = max(0, -0.653 \cdot t_{WT} + 8) \tag{4.1}$$

$$G_{ES} = max(0, -0.137 \cdot t_{ES} + 8), \tag{4.2}$$

where t_{WT} and t_{ES} represent the initial delay and the early start time in seconds, respectively. G_{WT} and G_{ES} denote the user satisfaction score for the two different serving modes, respectively. For zero waiting time or zero early start time, a score of 8 is obtained in (4.1) or (4.2). This reflects that during the subjective test, the test persons were too conservative to give the full points. Therefore, in the following, score 8 is assumed to be the highest score.

4.3 Proxy Caching Structure and Working Principle

In many conventional VoD cache management schemes, the whole cache is managed by a single algorithm and hence might lead to unnecessary or improper replacement. To improve the efficiency of cache management, a two-level cache structure should be employed. As shown in Fig. 4.6, the entire cache is divided into two parts, namely L1 cache and L2 cache, which can properly manage the caching of videos with short-term popularity and long-term popularity.

When a user request arrives at the proxy, the proxy checks the L1 cache and L2 cache as a whole. If there is a cache hit, the proxy starts forwarding the requested content immediately

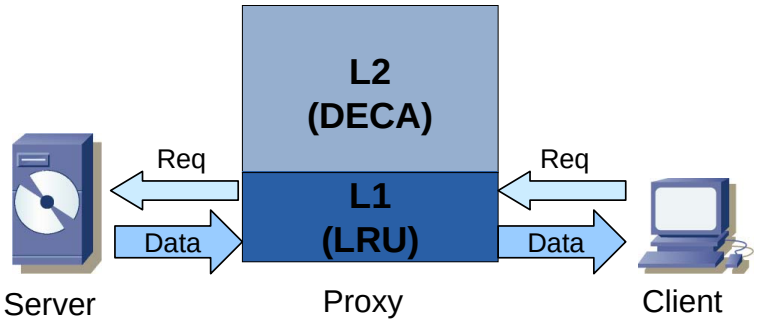


Figure 4.6: Two-level cache structure

to the client. If the result is a cache miss, the proxy has two options. It can either wait for the loading of the missing part from the server or start from the closest cached prefix. For the first option, the missing GOPs are requested from the remote server and delivered to the client after some delay. For the second option, the closest prefix that is completely available before the intended starting point is sent to the client right away. In both cases, missing frames in the requested segment and successive segments are retrieved from the remote server to achieve a continuous playback. All received video content passes through and is cached in the L1 cache before being delivered to the client. Once the L1 cache is full, the LRU caching update algorithm is used to release some space for newly downloaded content. The L2 cache is regularly managed using the new cache replacement algorithm to be described in the next sections. Unpopular fragments will be cleared and those with high popularity will be fetched from the L1 cache or from the content server. The update period of the L2 cache is normally much longer than that of the L1 cache, and could for instance be daily or weekly. When the distributions of popularity at two successive update time points are very similar, the update of L2 cache can be even skipped.

With the two level caching structure, the proxy cache can be managed in a more efficient way. For instance, when a video fragment is requested for the first time, it will be fully loaded from the remote server and temporally cached in the L1 cache. If it is popular, the frequent user requests will prevent it from being replaced, although the accumulated hitting rate of this fragment is still low. Later, when the L2 cache is updated, the hitting rate of this fragment might have been high enough and it is then cached in the L2 cache. On the contrary, if the requested video fragment is of low popularity and is just occasionally visited, it will be replaced soon after being temporally cached in the L1 cache and it has no chance to enter the L2 cache. This ensures that the cached content in the L2 cache will not be replaced by those fragments which are rarely requested.

4.4 Cache Update with Dynamic Segment Structure

In this section, the variable segment structure and some important definitions for the Dynamic sEgment-based Caching Algorithm (DECA) are first introduced. Then the algorithm will be presented in detail.

4.4.1 Segment-Prefix Structure

As shown in [WY07], more cache resources should be assigned to the video fragments with high popularity to make the caching more efficient. Therefore, instead of using a fixed segment structure, a more flexible segment-prefix structure is proposed, so that video fragments with higher popularity can have smaller segment size to enable a small or even no initial delay and early start. On the other hand, larger segments are mostly used for those video parts with low popularity. This is different to the proposal in [HNG+99], where the smallest replacement unit is a chunk. In this work, the replacement happens at the GOP level rather than at the chunk level.

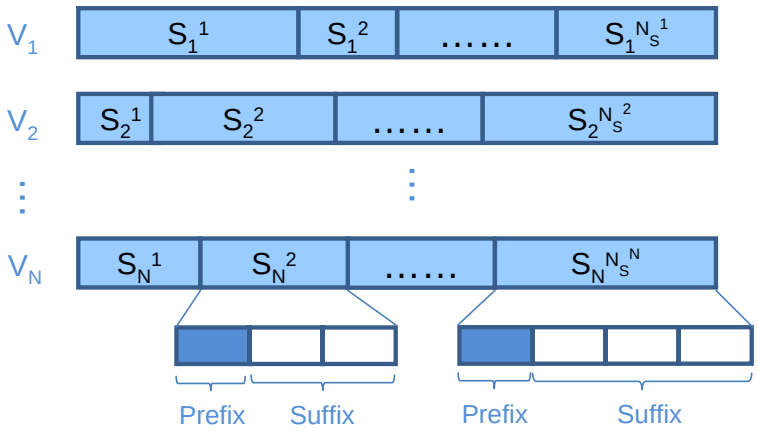


Figure 4.7: Variable size segment structure

Fig. 4.7 shows the segment structure with variable segment size. As illustrated, there are N videos altogether, namely V_1 to V_N, each of them has N_S^v $(v = 1 \ldots N)$ segments. Generally speaking, different videos may have different segment size, and the segment size within the same video may also vary significantly. Each segment consists of two parts: a prefix and a suffix. The length of the prefix or suffix is expressed as the number of GOPs it contains. Take the N^{th} video in Fig. 4.7 as an example. The second segment of V_N consists of 1 prefix GOP and 2 suffix GOPs, while the last segment comprises 1 prefix GOP and 3 suffix GOPs. The length of the prefix (L_P) is determined by the network condition between the server and the proxy. For a certain video file, it is assumed that it will be retrieved from only one content server and hence, the same network condition applies to all segments of a video. As a result,

they have identical prefix length. For simplicity but without loss of generality, assumption is also made that all videos have the same prefix length in this work. The minimum segment size is equal to the prefix length, which is also the finest achievable random seek granularity for an immediate playout unless L_P successive GOPs from the requested point are cached on the proxy.

The variable size segment structure empowers the replacement algorithm to keep up with the changing popularity distribution and also provides a way to adjust the granularity of random access. Video fragments with high popularity have smaller segment size, therefore finer granularity and vice versa. Furthermore, this variable segment structure also ensures that all prefix GOPs are cached by adjusting the length of individual segments. Therefore, if the early start mode is selected by the system, no additional waiting time will be experienced by the user. Otherwise, if the initial delay mode is more favorable, the playback will start exactly at the requested point.

4.4.2 Serving Mode Selection

As mentioned in the last section, different modes (i.e., initial delay or early start) can be selected to serve the user requests. This section introduces how the waiting time and early start time can be calculated for the user request to a particular GOP. Based on that, the optimal serving mode is determined.

The expected waiting time for a particular request is the time needed to download L_P successive GOPs from the requested starting point if any of them are not cached on the proxy. Only when they are all available, the playout can be started. These L_P GOPs might go across the segment boundary and include some prefix GOPs of the next segment, which should always be cached according to the design of DECA. Therefore, only the loading time for missing suffix GOPs has to be considered. The waiting time for a request to the i^{th} GOP of segment j in video v can be calculated as

$$t_{WT}^{(v,j,i)} = \sum_{n=max(L_P+1,i)}^{min(L_S^{(v,j)},i+L_P-1)} \frac{R_G^{(v,j,n)}}{r^v}, \tag{4.3}$$

where $R_G^{(v,j,n)}$ is the size of the n^{th} GOP in the j^{th} segment of video v. $L_S^{(v,j)}$ is the length of the segment j in number of GOPs and L_P is the length of the prefix in number of GOPs. r^v is the transmission rate between the proxy and the server where video v is stored.

The early start time is determined by the distance between the requested GOP and the first GOP in the segment. It can be calculated as

$$t_{ES}^{(v,j,i)} = \frac{L_G}{F_r} \cdot (i-1), \tag{4.4}$$

where L_G is the length of one GOP in number of frames and F_r is the frame rate of the video sequence. The ratio L_G/F_r converts the early start time in number of GOPs to the early start time in seconds.

In DECA, the playout does not necessarily start from the beginning of a segment. For large segments, if the desired starting point is too far away from the beginning of the segment, the waiting mode might be selected as the serving mode. This is because in this case, early start will lead to a low user satisfaction, which might be even lower than the user satisfaction obtained by downloading the requested part from the remote server. Assume that the early start time would be 12 seconds and alternatively the waiting time would be 1 second, users would prefer waiting for 1 second rather than encountering a starting point deviation of 12 seconds. On the other side, when the requested GOP is not far away from the beginning of the segment it belongs to, the early start mode is typically preferred. Once the early start time and the waiting time for the requested GOP is obtained from (4.3) and (4.4), the scores for the two modes can then be calculated by evaluating the score functions (4.1) and (4.2). The two scores are compared and the mode leading to higher user satisfaction is finally selected, which is presented as

$$G^{(v,j,i)} = max(G_{ES}^{(v,j,i)}, G_{WT}^{(v,j,i)}). \tag{4.5}$$

4.4.3 GOP Level Popularity

As most VoD users access the video content randomly [ZSL05], different parts of a video may have very different popularity [YCDW06]. For example, fragments with goals in a football match will usually be visited more frequently than other fragments if random access is enabled. Let's suppose that two videos V_1 and V_2 are cached on the proxy. The overall popularity of V_1 is higher than the popularity of V_2. However, the most popular fragment of V_2 has a popularity exceeding most parts of V_1. If a cache replacement algorithm works with video level popularity, video frames in V_2 will always be removed before those in V_1 because V_2 has a lower overall popularity, even the most popular part of it might not survive in this process. Obviously, a more efficient algorithm should delete those parts in V_1 with lower popularity rather than the most popular part in V_2. As GOPs are independently decodeable, they define the finest access granularity. Therefore, instead of evaluating the popularity for the entire video as in [JBI03, PLC01, YL03], the popularity used in DECA is measured for every GOP during a predefined time interval.

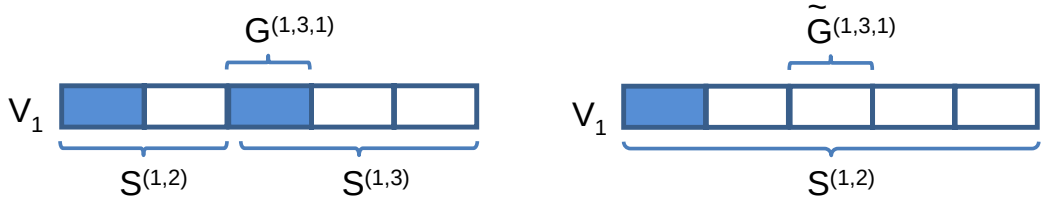
4.4.4 Cost Calculation

In this section, an important metric called "cost" is introduced, which is used during the merger process to build up the variable segment structure shown in Fig. 4.7. The "cost" mentioned above represents the price to be paid for one byte of free cache space if two neigh-

boring segments are merged together, of course the cheaper the better. The pairwise cost is a function of three different parameters. Popularity is the first parameter. More popular video fragments are intended to be kept, therefore the higher the popularity, the bigger the cost should be. The second aspect to consider is the user satisfaction. When two segments are merged, user satisfaction degrades because of the deletion of the second segment and therefore larger early start or more waiting time is experienced. This degradation is expected to be as small as possible, therefore the bigger the decrease of the satisfaction score is, the larger the cost should be. The last issue is the released cache space. The larger the free space created after a merger, the smaller the cost should be.By the merger of two neighboring segments, the waiting time of the last L_P GOPs in the first segment increases because the prefix of the second segment is now missing. Meanwhile, both the waiting time and the early start time for the GOPs in the second segment might also differ. Therefore, the user satisfaction degradation of GOPs in both segments should be considered. Based on the above arguments, we define the cost for segment pair j of video v as:

$$C^{(v,j)} = \frac{\sum_{s=j}^{j+1} \sum_{i=1}^{L_S^{(v,s)}} (p^{(v,s,i)} \cdot \Delta G^{(v,s,i)})}{\Delta B^{(v,j)}}, \tag{4.6}$$

where $p^{(v,s,i)}$ is the popularity of GOP i in segment s. $\Delta G^{(v,s,i)}$ denotes the decrease of the user satisfaction score for the *i-th* GOP in segment s if we merge the pair together. The summation of this weighted user satisfaction degradation gives the total price of this merger. $\Delta B^{(v,j)}$ is the corresponding increase in free cache space. A large cost value means the pair may have a high popularity or the user satisfaction declines severely or only limited free cache space is generated after the merger. We tend to keep such a pair on the proxy.



(a) Before merging (b) After merging

Figure 4.8: Example of cost evaluation for segment merging

The increase in free space ΔR is easy to obtain. It is simply the size of the prefix of segment $S^{(1,3)}$ of video V_1 in the example shown in Fig. 4.8(a). To get the decrease in user satisfaction ΔG of GOP i in $S^{(1,3)}$, the user satisfaction score of GOP i before and after the merger, $G^{(1,3,i)}$ and $\tilde{G}^{(1,3,i)}$ respectively, should be calculated. Finally, the decrease of user satisfaction for GOP i by this merger can be calculated by:

$$\Delta G^{(1,3,i)} = G^{(1,3,i)} - \tilde{G}^{(1,3,i)}. \tag{4.7}$$

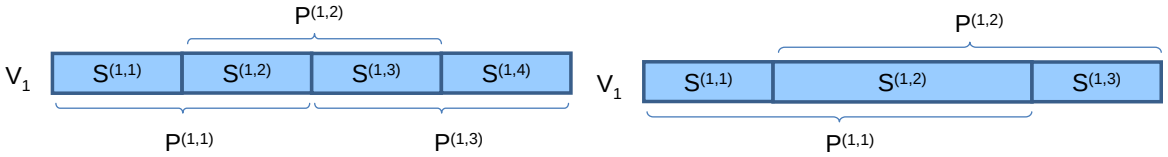
In the following, an example is given to show how the user satisfaction degradation of the first GOP in $S^{(1,3)}$ can be calculated. Before the merger, it is the first GOP of the segment and belongs to the prefix. Therefore, it can be played without delay and starting point deviation, which achieves a user satisfaction score of 8. After the merger, it becomes the third GOP in segment $S^{(1,2)}$ in Fig. 4.8(b). When it is requested, the proxy can either download the missing part from the remote server and then deliver it to the user, or send the prefix to the user instead and meanwhile download the missing GOPs. The former option results in 1 second initial delay on average while the latter option leads to an early start of 2 seconds but no initial delay. In this example, it is assumed that the frame rate in Hz equals to the GOP length in number of frames and the transmission rate between the server and the proxy equals to the mean rate of the video stream. According to (4.1) and (4.2), the user satisfaction score for 1 second waiting time is 7.347 and the score for 2 seconds early start is 7.726. Hence, the early start mode will be chosen to serve the request to this GOP, therefore the score of it is 7.726. The score degradation $\Delta G^{(1,3,1)}$ can then be calculated with (4.7) and it equals to 0.274. The score degradation $\Delta G^{(1,3,2)}$ and $\Delta G^{(1,3,3)}$ for the other two GOPs in the new segment $S^{(1,2)}$ in Fig. 4.8(b) can be obtained in the same way.

4.4.5 Replacement Algorithm

In this section, the procedure to determine the variable size of each segment by recursively merging of neighboring segments is presented. After the algorithm is employed to create the variable segment structure, the L2 cache is updated accordingly by removing all suffix GOPs and prefetching all prefix GOPs of the new segment structure. The updated cached content achieves an improved user satisfaction score according to the long term popularity. Therefore, this procedure is called as "cache update" in the following.

The cache update algorithm is invoked during the regular maintenance of the L2 cache on the proxy server that takes place once per day for instance. With the current segment structure and the popularity distribution available to it, the replacement algorithm works as follows. It first browses through all videos, checks the neighboring segments in a video pairwise, calculates the merging cost in (4.6) for each pair and saves it to an array. After all the segment pairs have been checked, the algorithm sorts all cost values in ascending order and starts merging from the first pair, i.e., the one with the smallest cost. Merging a segment pair means to join the two segments together and make one bigger segment out of them. All GOPs of the second segment in this pair become suffix GOPs and are completely removed to release cache space.

Every time the DECA algorithm is called, it runs iteratively and the pair with the lowest cost is merged in each loop. As the two segments in this pair are now replaced by a new segment, the cost information of the pairs with neighboring segments has to be updated. For



(a) Before merging (b) After merging

Figure 4.9: Example of pair information update for segment merging

example, as shown in Fig. 4.9(a), when pair $P^{(1,2)}$ is selected and merged, a new segment (i.e., $S^{(1,2)}$ in Fig. 4.9(b)) is created. The pairs $P^{(1,2)}$ and $P^{(1,3)}$ before the merger consist of segments $S^{(1,2)}$ and $S^{(1,3)}$, which no longer exist afterwards. Therefore, their cost has to be recalculated. Please note, pair $P^{(1,3)}$ in Fig. 4.9(a) becomes the second pair $P^{(1,2)}$ in Fig. 4.9(b). After updating the information of related pairs in the cost array, the next loop starts. This procedure continues until all remaining content can be stored in the cache.

When DECA is run for the first time, an initial segment structure is built up, where all the GOPs are intact and every successive L_P of them are grouped together to form one initial segment. The merging operation starts from such an initial segment structure and stops when a target cache size is reached. As the merging operation goes on, there will be fewer segments in the cache. The final segment structure resulting from the merging operation comprises segments of different lengths. They all have the same prefix length whereas their suffix length differs. After running DECA using the initial segment structure, all suffix parts currently cached will be deleted. If the new prefix GOPs are not available in the L2 cache, they are fetched from the L1 cache if they are cached there. Otherwise, they have to be reloaded from the remote server.

Generally speaking, the cache replacement algorithm can use the output of the last round and perform further combinations or updates based on that. If the popularity distribution between two successive updates stays similar, none or only partial update of the L2 cache is needed. In this case, most of the segment pairs are kept unchanged and only those fragments which have large changes in their popularity need to be handled. For those having a decreased popularity, segment pairs will be merged to leave some space. On the other side, if some video parts have an increased popularity, the suffix GOPs of those segments should have already been stored in the L1 cache because they are more frequently visited between the two update points. Furthermore, there could be some new videos in the L1 cache, which also have a relatively high popularity and therefore should also be segmented and partially fetched to the L2 cache. In both cases, as most of the data can be fetched from the L1 cache, little additional traffic between the proxy and server will be generated. In the experiments, for the sake of simplicity, the replacement algorithm always takes an initial segment structure as its input. However, it is found from the experiments that these two approaches lead to similar results.

4.5 Performance Evaluation Metric

In this section, the evaluation metric used to examine the performance of DECA is introduced. The mean early start time and the mean waiting time of the system are calculated. Based on that, the final score is obtained according to (4.1) and (4.2), which reflects the user satisfaction for the VoD system.

Section 4.4.2 has shown how the optimal mode can be determined for a particular user request. As for every request, only one mode is selected, either the waiting time or the early start time is zero. Hence, a threshold T^v can be defined, which represents the switching point between the two modes for video v. When the requested GOP has index inside a segment larger than T^v, the waiting mode leads to higher user satisfaction and the early start time is zero. Otherwise, the early start mode is preferred and the waiting time becomes zero. It can be obtained from (4.1) and (4.2) as

$$T^v = 4.77 \cdot \frac{L_P^v \cdot R_G^v \cdot F_r}{L_G \cdot r^v}, \tag{4.8}$$

where R_G^v is the average GOP size of video v, which is used here as an approximation. (4.3) and (4.4) are then rewritten as

$$t_{WT}^{(v,j,i)} = \begin{cases} 0 & \text{if } i \leq T^v \\ \sum_{n=max(L_P+1,i)}^{min(L_S^{(v,j)},i+L_P-1)} \frac{R_G^{(v,j,n)}}{r^v} & \text{if } i > T^v \end{cases} \tag{4.9}$$

and

$$t_{ES}^{(v,j,i)} = \begin{cases} \frac{L_G}{F_r} \cdot (i-1) & \text{if } i \leq T^v \\ 0 & \text{if } i > T^v. \end{cases} \tag{4.10}$$

Based on (4.1) and (4.9), the mean waiting time of the whole system can be determined by

$$E_{WT} = \sum_{v=1}^{N_V} \sum_{j=1}^{N_S^v} \sum_{i=1}^{L_S^{(v,j)}} t_{WT}^{(v,j,i)} \cdot p^{(v,j,i)}, \tag{4.11}$$

where $p^{(v,j,i)}$ represents the normalized access frequency of GOP i in the j^{th} segment of video v. Similarly, based on (4.2) and (4.10) the averaged early start time of the whole system is

$$E_{ES} = \sum_{v=1}^{N_V} \sum_{j=1}^{N_S^v} \sum_{i=1}^{L_S^{(v,j)}} t_{ES}^{(v,j,i)} \cdot p^{(v,j,i)}. \tag{4.12}$$

The mean waiting time and the mean early start time are important performance parameters of the system. However, with any one of the two, it is hard to give a comprehensive evaluation of the system. Instead, both aspects have to be taken into account simultaneously. Therefore, the user satisfaction score in (4.5) is adopted and the overall performance of the VoD system is calculated as

$$G = \sum_{v=1}^{N_V} \sum_{j=1}^{N_S^v} \sum_{i=1}^{L_S^{(v,j)}} G^{(v,j,i)} \cdot p^{(v,j,i)}. \tag{4.13}$$

4.6 Experimental Results

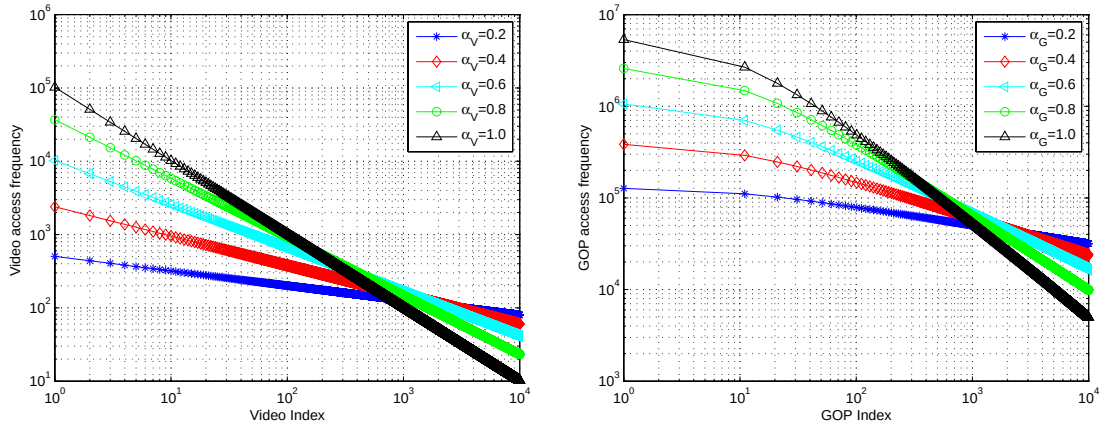
In this section, the simulation setup is first briefly introduced. Using the evaluation metric defined in Section 4.5, the performance of the proposed DECA approach is evaluated for different parameter setups. Then, comparison schemes are introduced and some of their properties are discussed. Afterwards, the performance of DECA and the reference schemes are examined and compared.

4.6.1 Simulation Setup

In this work, online video trace files [Ari] are used instead of real video sequences. Necessary information about all the videos hosted on the remote server is included in the trace files. Four sequences are included in the simulation, however, a larger number of test sequences can be easily added and they achieve similar results. The videos are encoded into CIF@30fps, 16 frames per GOP, 3 B-frames between two I/P-frames, using the H.264/AVC codec with QP equal to 28. Table 4.2 shows the major properties of these videos.

Table 4.2: Properties of the videos used in the simulation

Video Name	Length(min)	Size(MB)	Bitrate(kbps)
Silence of the Lambs	30	159	144
Star Wars IV	30	161	156
NBC 12 News	30	612	439
Tokyo Olympics	74	902	306



(a) Zipf distribution (b) K-Transformed Zipf distribution

Figure 4.10: Example of video and GOP level access frequency, which is used as popularity distributions in our experiments.

The second issue is how to obtain the request events to represent the user behavior. Known from the literature, the user request frequency for video objects in a VoD system can

be modeled with a Zipf distribution [Zip49], and the user request frequency for object i is modeled as

$$f_i = \frac{1}{i^{\alpha} \cdot \sum_{j=1}^{m} j^{-\alpha}}, \tag{4.14}$$

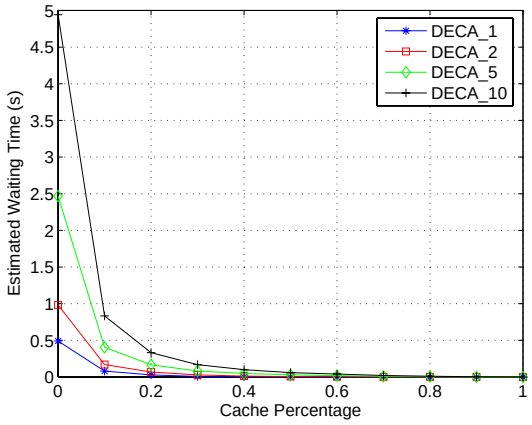
where m is the number of distinct objects. α is the parameter to adjust the skewness of the Zipf distribution, which is larger than 0 and less than or equal to 1.0. The larger the α, the bigger the difference of popularity between objects. According to [YCDW06], the internal popularity of a video (i.e., the access frequency of all GOPs) follows a K-Transformed Zipf distribution. Therefore, a Zipf RNG (random number generator) and corresponding transformations with K_x=10 and K_y=400 are employed to generate the requests to the GOPs in all videos. The α for the video level Zipf distribution is marked as α_V and as α_G for the GOP level K-Transformed Zipf distribution. The access frequency with different α values for both distributions is shown in Fig. 4.10, taking 10000 units as an example. In the experiments, without specification, both α are set to be 0.8 according to [WY07]. Please note that real user request log files can also be adopted into the proposed algorithms and should achieve similar results.

4.6.2 Performance of DECA

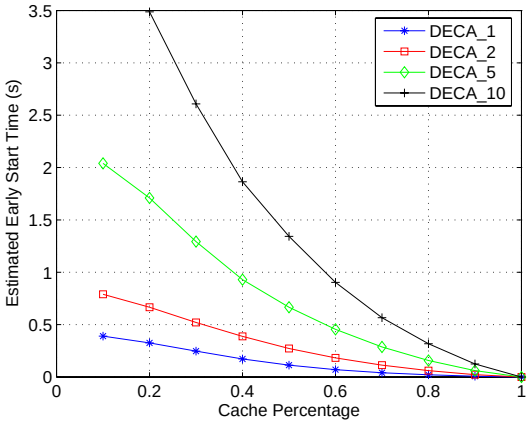
In this section, the performance of the DECA approach for different scenarios is investigated. Fig. 4.11 illustrates different performance metrics as a function of the percentage of total video content that can be cached on the proxy. *DECA_X* in the figure represents the DECA algorithm with a prefix length of X GOPs.

Fig. 4.11(a) shows the mean waiting time for different prefix lengths, which corresponds to different network conditions between the proxy and the server(s) as mentioned in Section 4.4.1. When the cache percentage is 0%, i.e., no video content is cached, any requested video content needs to be loaded from the remote server. This leads to a larger initial delay for *DECA_10* as it has more data as prefix that needs to be loaded from the remote server compared to DECA with smaller prefix size. Therefore, it has a longer waiting time when nothing is cached on the proxy. Thanks to the dynamic segment structure of DECA, all prefix GOPs can be cached, which leads to a significant decrease of waiting time when the cache percentage increases. When the caching rate is very small, e.g., smaller than 10%, although the prefixes are all available, the segments are too long so that "wait for loading" is more preferable than a large deviation of starting point. Therefore, some initial waiting time is observed for the requests to some middle to unpopular parts. However, zero initial delay can be achieved as soon as about 60% of the video content is cached.

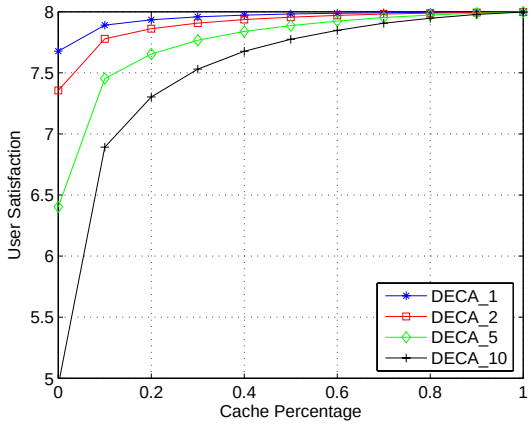
Fig. 4.11(b) illustrates the expected early start time as a function of the caching percentage when DECA is employed to manage the cache. The early start time decreases when the cache percentage increases, as caching more content on the proxy results in fewer segment mergers and thus smaller segment size on average. Please note, a cache percentage of 0% is a special



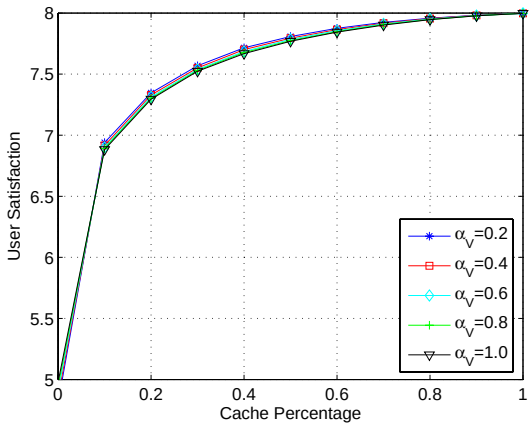
(a) Waiting Time (α_V=0.8, α_G=0.8)



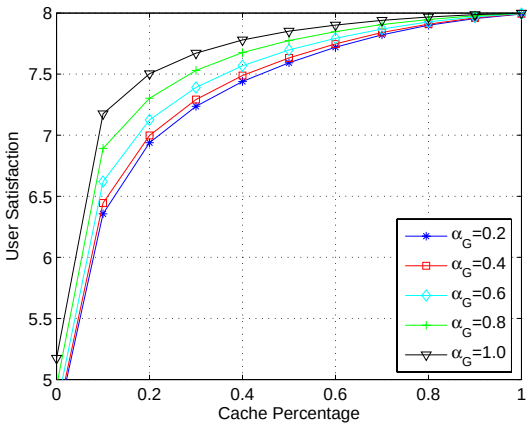
(b) Early Start (α_V=0.8, α_G=0.8)



(c) Final Score (α_V=0.8, α_G=0.8)



(d) Final Score (L_P=10, α_G=0.8)



(e) Final Score (L_P=10, α_V=0.8)

Figure 4.11: Performance of DECA as a function of cache percentage

case, where nothing is cached and the early start mode is disabled. DECA with smaller prefix size needs less cache size for each segment and therefore has more segments at the same

caching percentage. This leads to an overall smaller segment size throughout the video and thus smaller expected early start time.

The final score of DECA for different prefix lengths is shown in Fig. 4.11(c). The final score of DECA is obtained using (4.13), where the optimal serving mode has been determined by comparing the achievable scores of waiting time and early start time. *DECA_1* has the smallest waiting time and the shortest early start time, therefore, achieves the highest final score at all caching percentages. When more content is cached (i.e., larger than 50%), always the same mode will be selected, which leads to a very close performance between all the curves in Fig. 4.11(c). When the caching percentage is zero, i.e., all requested content needs to be downloaded from the server, *DECA_1* still performs the best because of its smallest prefix size and therefore least amount of data to cache before the playback starts.

Fig. 4.11(d) and Fig. 4.11(e) illustrate the user satisfaction score for the (K-Transformed) Zipf distribution with different α values when the prefix length L_P is equal to 10 GOPs. The user satisfaction shows only a small change when α_V varies, because the limited number of videos involved in the simulation and thus the change of popularity among videos is not dramatic. When α_V is fixed, a lower user satisfaction score is clearly observed when α_G decreases. This is because the difference of popularity between video GOPs is not so dramatic for small α_G values. Therefore, caching the most popular video parts contributes less to the overall performance than when $\alpha_G = 1.0$. Nevertheless, DECA shows a consistent performance independent of the selection of α for the assumed request distribution. In the following simulations, when the influence of α to the final score of all schemes is investigated, α_V is always set to be 0.8 and only α_G is considered.

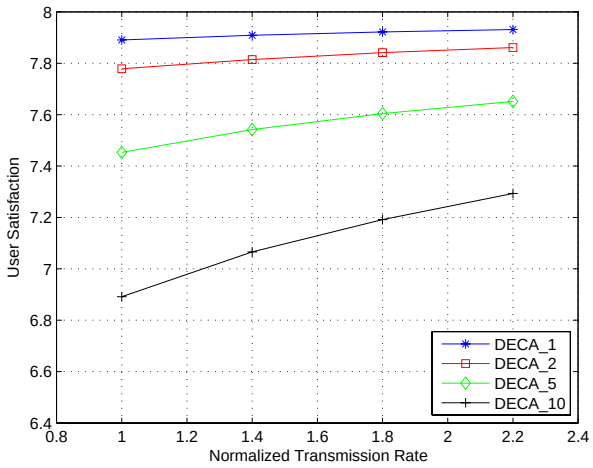


Figure 4.12: User satisfaction as a function of the available transmission rate at a cache percentage of 10%. (α_V=0.8, α_G=0.8)

In the above simulations, the transmission rate between the proxy and the content server

is assumed to be equal to the mean rate of the video stream. However, if the available transmission rate is larger than the mean rate, the waiting time to load the same amount of data decreases. As the early start time has no relation with the transmission rate, the "initial waiting" mode becomes more attractive and will be selected more frequently. With a faster connection to the server, the overall performance of the system improves.

Fig. 4.12 illustrates the achievable user satisfaction score of *DECA_X* as a function of the available transmission rate between the server and the proxy. The X-axis shows the normalized transmission rate, which is the ratio of the available transmission rate over the mean rate of the bitstream. Naturally, the larger the transmission rate is, the higher the achievable user satisfaction. As can be observed from Fig. 4.12, the increase of transmission rate is most helpful to the segment structures with a larger prefix size, where *DECA_10* shows the biggest improvement. This is because a larger absolute time saving for loading the missing GOPs is achieved for *DECA_10*, which results in a remarkable improvement of user satisfaction according to (4.1). Fig. 4.12 shows the results when α_G equals 0.8. When smaller α_G values are used, similar results are observed, however, the improvement by increasing the rate is even larger. This is because in DECA, segments with high popularity are of very short length (i.e., the segment only consists of a prefix) and requests to them can be served with neither initial delay nor early start. Therefore, when α_G becomes smaller, more requests benefit from the higher transmission rate resulting in smaller waiting time, which leads to an improvement of the overall user satisfaction.

According to Section 4.3, the whole cache is divided into two levels. DECA is employed to manage the L2 cache, while the L1 cache uses the LRU algorithm for the updating. The performance of DECA has been shown in the above when it controls the whole cache for different simulation setups. The influence of the L1 cache is not shown here because no real access log files are available and the Zipf distribution can represents a statistic hitting rate without the information of request order. However, the performance improvement when including L1 cache is obvious when the popularity changes significantly. For example, if there is a new video published by the server after the last run of DECA, by assigning 20% of the cache as L1 cache, this popular video might be fully cached on the proxy. It leads to some degradation from the DECA perspective because of the decreased caching size. However, a significant gain can be obtained by employing the L1 cache, which leads to high user satisfaction for this popular video. On the contrary, if the popularity change is very small and no new video comes in during the two updating points, letting DECA work for the whole cache leads to better results.

4.6.3 Performance of PAPA

In this section, the performance of the PAPA approach, on which DECA builds, is investigated. As mentioned in the introduction, PAPA is a cache management scheme that uses a fixed segment-prefix structure and it is used as one of the comparison schemes in the next section.

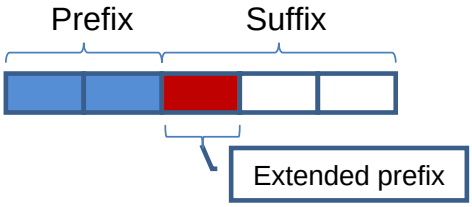
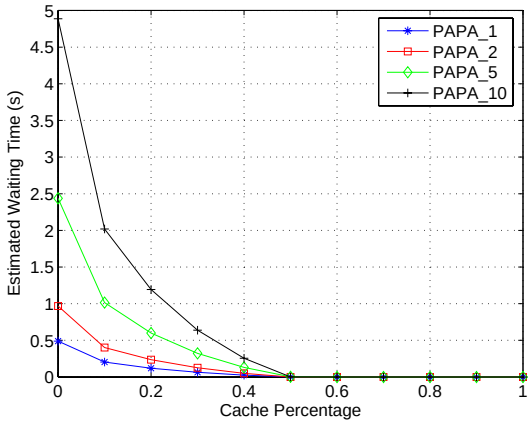


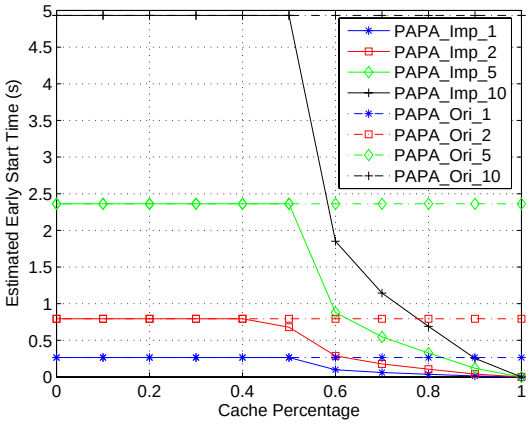
Figure 4.13: One complete suffix GOP is cached

Here, an improved version of PAPA compared to the original description in [STS07] is implemented. Instead of always starting the playout from the beginning of one segment, a closer starting point can be considered if some suffix frames are also cached to form an extended prefix as shown in Fig. 4.13. In this case, the playout can be started from a GOP which has a distance of L_P GOPs to the first unavailable suffix. By serving the client from a closer point to the desired starting point, the expected early start time is decreased. In the extreme case, if all video content is cached on the proxy, no early start will be experienced. For example, to respond to a user request to the fourth GOP in Fig. 4.13, instead of starting with the first GOP in this segment, the playout can start with the second GOP, which together with the third GOP act as the prefix for this request. This leads to again no waiting time, but smaller shift from the desired starting point.

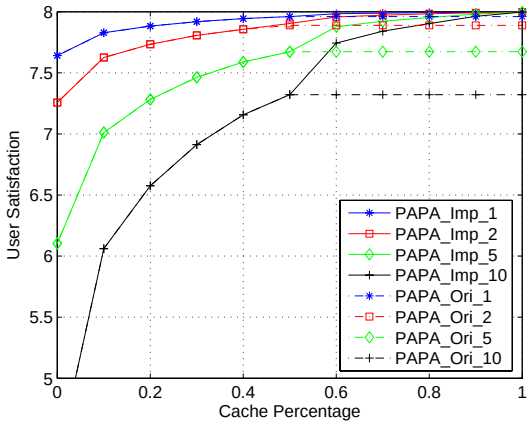
Fig. 4.14 shows the performance of PAPA as a function of the percentage of cached content. The dash-dotted curves and the solid curves in the figure represent the original PAPA approach (*PAPA_Ori*) proposed in [STS07] and the improved version of PAPA (*PAPA_Imp*) introduced above, respectively. Both schemes achieve the same estimated waiting time and therefore they are not distinguished in Fig. 4.14(a). As shown in Fig. 4.14(b), a significant improvement is achieved by extending PAPA with a more flexible starting point. The early start time declines for the improved PAPA approach when the cache percentage exceeds 50% because the suffix length is equal to the prefix length in this experiment. Therefore, if more than 50% of the video content can be cached, all prefix GOPs and some of the suffix GOPs are cached, which enables a closer starting point. The early start time of *PAPA_Imp_10* declines the fastest as it has the longest segment size and the smallest number of segments. For the same amount of additional caching capacity, *PAPA_Imp_10* is able to cache more suffix GOPs with high popularity. Fig. 4.14(c) shows the resulting user satisfaction scores. *PAPA_Imp* outperforms *PAPA_Ori* when more than 50% of the video content can be cached, which is the result of the smaller early start time of *PAPA_Imp* by the flexible selection of starting point.



(a) Waiting Time



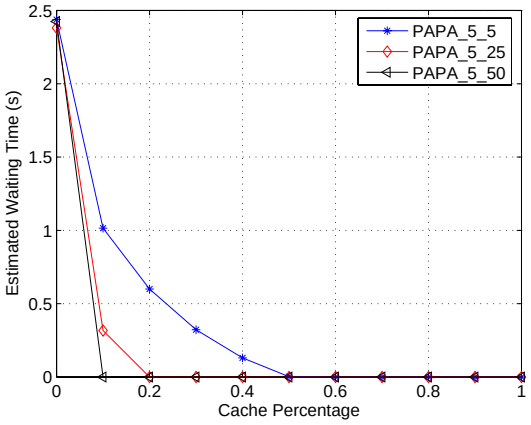
(b) Early Start



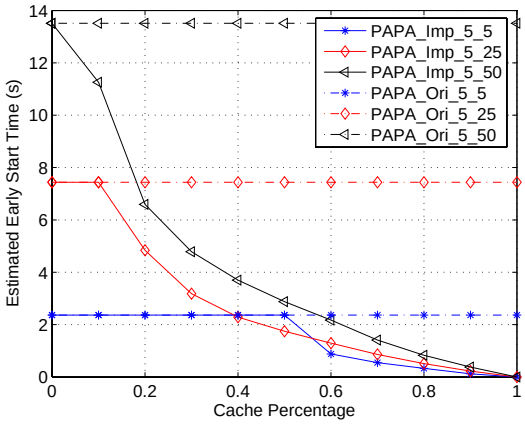
(c) Final Score

Figure 4.14: Performance of PAPA as a function of the percentage of cached content for different prefix lengths. The suffix has the same length as the prefix. (α_V=0.8, α=0.8)

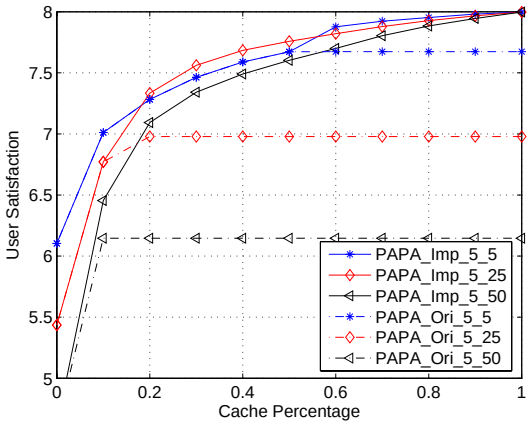
Fig. 4.15 shows a performance of PAPA for different suffix lengths. *PAPA_X_Y* denotes the segment structure with X prefix GOPs and Y suffix GOPs. All the curves in the figure have the same prefix length of 5 GOPs. As shown in Fig. 4.15(a), the larger the suffix length is, the earlier the waiting time declines to zero. This is because the large segment size leads to fewer segments and therefore the total number of prefix GOPs is smaller. However, it is hard to say that the large segment structure always leads to better performance. In Fig. 4.15(b), the early start time is depicted. For the original PAPA approach, the early start time is fixed when the segment structure is determined, while the improved PAPA approach significantly decreases the early start time when some suffix GOPs are cached. Also because of fewer segments, the *PAPA_Imp_5_50* starts the decrease of early start for the lowest cache percentage, as only less than 10% of the video content belongs to the prefix. As suffixes are patched one after the other according to their popularity, *PAPA_Imp_5_50* still has the higher overall early start time than



(a) Waiting Time



(b) Early Start



(c) Final Score

Figure 4.15: Performance of PAPA as a function of the percentage of cached content for different suffix lengths. (L_P=5, α_V=0.8, α=0.8)

PAPA with other suffix lengths. Fig. 4.15(c) illustrates the final score for different segment structures. Small segment size leads to better performance at low cache percentage because of the significantly smaller early start time. A reasonably larger segment size performs better when the cache percentage increases to the medium range due to the fast decline of waiting time and early start time. At high caching percentage, there is no waiting time anyway and *PAPA_Imp_5_5* again performs the best.

The performance of PAPA for the request distributions with smaller α_G values is omitted as they show similar tendency as the results for DECA in Section 4.6.2.

4.6.4 Performance Comparison

In this section, the performance of DECA is compared with selected reference schemes. As the user satisfaction score is the most representative metric to evaluate the performance of

the system, the achievable final user satisfaction score is used as the criteria to make the comparison among different schemes.

As shown in the last section, the improved PAPA cache management scheme always performs the same as or better than the original PAPA approach in [STS07]. Therefore, the improved PAPA scheme is used for comparison. Furthermore, as different segment structures (i.e., different suffix lengths) achieve the highest final score for different percentages of cached content, always the highest score at each percentage is used in the comparison, which forms an upper bound of PAPA.

DECA will also be compared with cache management schemes which have no segment-prefix structure. In this case, each GOP is a possible starting point. Two approaches, on the frame level and on the GOP level will be compared. The End-GOP (EGOP) approach works on the frame level and drops frames evenly from each GOP from the end towards the beginning until the remaining part can be cached. Only when the last frames of all GOPs have been deleted, the second last frame in the GOP with the lowest popularity will be dropped. The Whole-GOP (WGOP) approach always deletes the whole GOP with the lowest popularity when not enough cache is available until all remaining GOPs can be held in the cache. These two strategies lead to the finest granularity but some initial delay unless the L_P successive GOPs from the requested point are available on the proxy. Early playout does not apply to these schemes. Similar as the segment-based schemes [WYW01, WYW04, CWZ+05] introduced in Section 4.1, the WGOP scheme also caches the video portions with the highest popularity, but with a much finer granularity. It supports both viewing modes, playing sequentially from the beginning or random access to video content. Obviously, WGOP should outperform the traditional segment-based approaches in [WYW01, WYW04, CWZ+05]. Therefore, WGOP is used as one of the comparison schemes in this work.

Fig. 4.16 shows the final user satisfaction score for the three comparison schemes. *PAPA_X* represents the upper bound of the improved PAPA. *WGOP_X* and *EGOP_X* denote the two no segment structure approaches, respectively. When the network condition between server and proxy is good (i.e., X=1), the four approaches achieve very similar performance as the initial delay for loading the unavailable requested data is very small, and most likely, "waiting" is the better mode. When the network condition is unfavorable (i.e., X=10), the performance gap between the different schemes becomes obvious. *DECA_10* performs the best at all cache percentages. The performance gain compared with *WGOP_10* is not so significant because for α_G=0.8 in the K-Transformed Zipf distribution, it gives strong emphasis to some GOPs and *WGOP_10* purely follows the popularity. Therefore, it pays high attention to popular GOPs by fully victimizing the GOPs with medium to low popularity. This is why at low cache percentage, it even performs better than PAPA. However, *PAPA_10* performs better than *WGOP_10* at middle to high cache percentage by enabling early start instead of pure

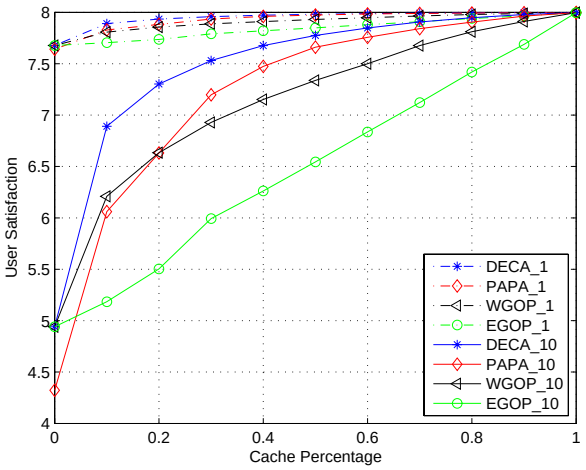


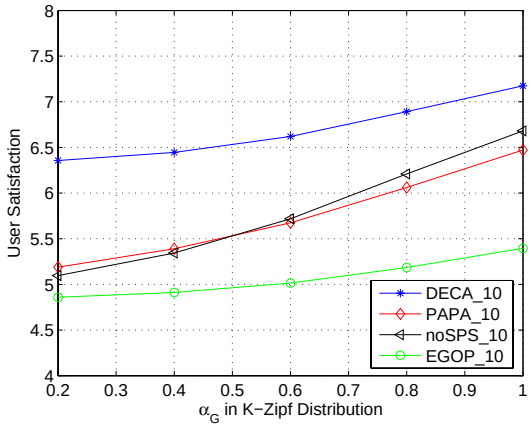
Figure 4.16: User satisfaction for DECA and comparison schemes as a function of cache percentage. (α_V=0.8, α_G=0.8)

waiting in WGOP. The *EGOP_10* approach considers the extreme fairness between GOPs, which wastes too much resources for the video content with low popularity. Without the help of early start, *EGOP_10* leads to the lowest overall performance.

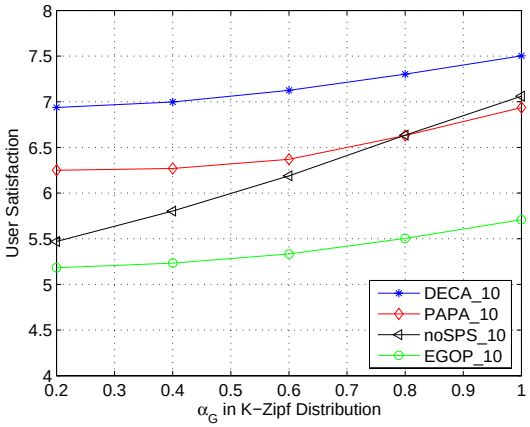
Fig. 4.17(a), (b), (c) show the final score for DECA and the comparison schemes as a function of α at a cache percentage of 10%, 20% and 30%, respectively. DECA performs again the best for all cache percentages and all α_G values and achieves an even larger performance gain compared with Fig. 4.16 when α_G is small. The performance of WGOP decreases dramatically with the decrease of α_G because the difference of popularity between fragments is much smaller and a purely popularity oriented approach becomes less efficient. The performance of PAPA improves significantly when the cache percentage increases, as the early start mode makes up for the score loss caused by the even distribution of prefixes.

4.7 Chapter Summary

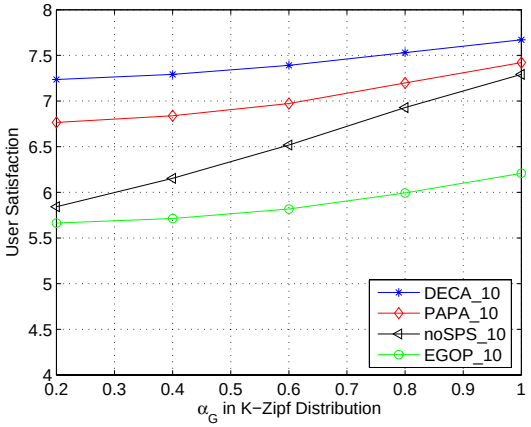
In this chapter, sophisticated subjective tests for VoD services are introduced together with the obtained results. It can be found that VoD users prefer a small shift of the playout starting point rather than experiencing a noticeable initial delay. Mathematical models are derived from the results, which show the influence of waiting time and early start time on the user satisfaction. A two level proxy-caching structure is proposed as a general framework that can be used for VoD services. A partial caching algorithm DECA is proposed, which introduces a variable size segment structure and flexible serving modes. This enables DECA to approach the real popularity distribution and dynamically adjust the segment size according to the current situations. The four issues mentioned in Section 4.1 are all addressed by the



(a) Cache Percentage = 10%



(b) Cache Percentage = 20%



(c) Cache Percentage = 30%

Figure 4.17: User satisfaction of DECA and comparison schemes as a function of α_G in the Zipf distribution.(α_V=0.8, L_P=10)

proposed approach. By caching popular video content and the two level caching structure, the overall initial delay and traffic between the proxy and the server are decreased. The "early start" operation mode further shortens the initial waiting time. The segment-prefix offers the possibility of random access to video content. Popular parts have much finer granularity than the unpopular parts, which is also reasonable. As the proxy determines the serving mode, it also determines when and which frame to be forwarded to the clients, which avoids the buffering at the client and minimizes the terminal buffer size. Simulation results also show that significant performance improvements are achieved by the proposed scheme compared with related approaches.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

In this thesis, proxy-assisted video transmission schemes have been studied. Compared with the traditional end-to-end infrastructure, a lot of benefits can be obtained with the support of proxies. For instance, the response time or the feedback time is much shorter, the information about the current channel status is more accurate, etc. Three main benefits that the proxy might contribute to the transmission of different video applications are addressed.

In Chapter 2, the proposed PRPS approach provides an efficient solution to improve the error resiliency for conversational video transmission over mobile networks. Retransmission of lost packets is one of the most efficient ways for error recovery. It is enabled on the downlink by employing a fixed-distance prediction when encoding. Dynamic reference picture selection improves the error robustness of video transmission on the uplink by always using "clean" video part for prediction. Three ways of performing reference picture selection are proposed and compared. The efficiency of both error recovery schemes used on uplink and downlink is significantly improved with the help of proxies located on the base stations close to the sender and receiver. The two schemes can be employed individually when only a wireless uplink or wireless downlink is involved in the transmission. However, they also work well in concert when both the sender and the receiver are mobile users. The PRPS achieves significantly improved performance compared with state-of-the-art approaches. Finally, the computational complexity of PRPS and all reference approaches is analyzed. PRPS is characterized with very low complexity and has a high applicability.

The congestion control and resource allocation issues are addressed in Chapter 3. Different video applications, streaming video and conversational video, as well as different video coding schemes, single layer video and scalable video are considered. Several apparatus are proposed to extract the rate-distortion side information (e.g., Distortion Matrix). With the sent along

side information, the proxy on the congestion node is able to do joint optimization among all participants in a RD-optimized way. For that, several frame dropping approaches are also proposed (e.g., utility-based approach, cost-function based approach). In case that only one type of video application is involved, optimally selecting important frames automatically assigns the shared resources to individual users. Otherwise, scheduling strategies are also proposed to efficiently assign the transmission resources to different types of video applications. The proposed apparatus and algorithms lead to a more efficient usage of transmission resources and therefore a much higher video quality. The storage and computation costs among different approaches are analyzed and compared, which gives the metric for trading off between performance and complexity.

In Chapter 4, a sophisticated subjective test environment is implemented for VoD services. The test results averaged from 28 test persons show that users prefer a small shift of the playout starting point rather than experiencing a noticeable initial delay. From the subjective test scores, mathematical models are derived, which show the influence of waiting time and early start time on the user satisfaction. Based on that, a two level proxy-caching framework and the prefix-segment based partial caching algorithm DECA are proposed. The former can be used as a general framework for any proxy caching system, which is able to consider both short term and long term popularity. DECA introduces a variable size segment structure and flexible serving modes. The real popularity distribution can always be considered by the dynamic adjustment of segment length according to the current situations. The proposed scheme takes the advantages of previous proposals (e.g., prefix, segmentation) while recovers the weakness from them (e.g., sequential playout, video level popularity). Simulation results show significant performance improvements compared to related approaches.

5.2 Future Work

Although the proposed frameworks and methods outperform the state-of-the-art approaches, there is still room for improvement.

As the PRPS is standard compatible, it can also be employed together with other error resiliency schemes (e.g., FEC), which is expected to achieve even better performance. In the proposed scheme, the error recovery of each user is independently considered. Further optimization can be made by the dynamic assignment of resources for retransmission on the downlink. Video packets having large contribution to the reconstruction quality should have higher priority and therefore more chances to be recovered if they are corrupted during the transmission.

The rate shaping approach for scalable videos is proposed in Chapter 3. When evaluating the quality, the reconstructed video is always compared with the original video with the highest temporal and spatial resolution. However, because of the heterogeneity of user equipments,

they might have different requirements to the video quality. For instance, for normal mobile users, QCIF resolution and maybe a frame rate of 15 Hz are also enough considering the power consumption. Therefore, optimization can be further improved and becomes more comprehensive when the heterogeneity of user requirements is also considered.

Moreover, the rate shaping approaches proposed in Chapter 3 consider the scenario only for wired networks, where the status of each user is relative stable. In wireless networks, according to the changing channel condition of individual users, the cost for sending the same amount of data differs. A first approach for streaming video over wireless networks has been proposed in [LTS07]. Further extension to also conversational videos will make this topic more comprehensive.

The dynamic segment structure proposed for proxy caching shows a promising performance improvement for VoD applications. Meanwhile, it can be further extended in the following directions.

- the optimal ratio between the L1 and L2 cache can be found based on the statistics derived from the log file of a real running system. A large L1 cache is preferable when more short term events are involved in the system and vice versa.

- The merge process has been introduced to release some caching space. A splitting process can also be developed when enough resources are available and more content is intended to be cached. Although more seldom used, it is a good complement to the merging process.

- VoD users can access to the video using slide bar, but thumbnail is another choice. To present the most exciting parts of the video, thumbnails can be extracted at the beginning of those parts, which are expected to be very frequently visited by clicking the thumbnails. Caching the segments after the thumbnails can further improve the user satisfaction. It is interesting to see the performance of a combination of thumbnail and dynamic segmentation used between neighboring thumbnails.

- Early start is used in this chapter to give an immediate feedback to user requests when the requested content is not cached on the proxy. However, other schemes can also be applied to conceal the initial delay. For example, showing some advertisements, some highlights of the day, the video content that most frequently requested, or some content that relates to the requested one, etc., would also be an option. Subjective tests should be investigated to compare the above options so that an optimal serving order can be determined. According to the content cached on the proxy, one of the above delay concealment schemes can be employed to improve the user satisfaction.

- A real VoD system equipped with all proposed functionalities can be built up to validate and further extend the proposed schemes.

Bibliography

[AGL+03] S. Ardon, P. Gunningberg, B. Landfeldt, Y. Ismailov, M. Portmann, and A. Seneviratne. March: a distributed content adaptation architecture. *International Journal of Communication Systems*, 16(1):97–115, January 2003. [cited at p. 4]

[Apo01] J. G. Apostolopoulos. Reliable video communication over lossy packet networks using multiple state encoding and path diversity. In *Proc. SPIE Visual Communications and Image Processing (VCIP'01)*, San Jose, CA, January 2001. [cited at p. 15]

[Ari] Arizona State University. Video traces for network performance evaluation. http://trace.eas.asu.edu/. [cited at p. 96]

[ASP98] S. Acharya, B. Smith, and P. Parnes. Characterizing user access to videos on the world wide web. In *Proc. ACM/SPIE Multimedia Computing and Networking (MMCN'98)*, San Jose, CA, January 1998. [cited at p. 81]

[ASTP03] K. Amiri, P. Sanghyun, R. Tewari, and S. Padmanabhan. DBProxy: a dynamic data cache for web applications. In *Proc. International Conference on Data Engineering (ICDE'03)*, Bangalore, India, March 2003. [cited at p. 2, 4]

[BBAC06] A. C. Begen, M. A. Begen, Y. Altunbasak, and M. R. Civanlar. Proxy selection for interactive video. In *Proc. IEEE International Conference on Communications (ICC'06)*, Istanbul, Turkey, June 2006. [cited at p. 3]

[BCZ97] S. Bhattacharjee, K. L. Calvert, and E. W. Zegura. Active networking and the end-to-end argument. In *Proc. IEEE International Conference on Network Protocols (ICNP'97)*, Atlanta, GA, October 97. [cited at p. 2]

[BG03] I. Bouazizi and M. Gunes. Selective proxy caching for robust video transmission over lossy networks. In *Proc. International Conference on Information Technology: Research and Education (ITRE'03)*, Newark, NJ, August 2003. [cited at p. 3]

[BGMO03] S. Belfiore, M. Grangetto, E. Magli, and G. Olmo. An error concealment algorithm for streaming video. In *Proc. IEEE International Conference on Image Processing (ICIP'03)*, Barcelona, Spain, September 2003. [cited at p. 15]

[Bou02] I. Bouazizi. Size-distortion optimization for application-specific packet dropping: The case of video traffic. In *Proc. IEEE International Symposium on Circuits and Systems (ISCAS'02)*, Scottsdale, Arizona, May 2002. [cited at p. 44]

[Bou03] I. Bouazizi. Size-distortion optimized proxy caching for robust transmission of MPEG-4 video. In *Proc. International Workshop on Multimedia Interactive Protocols and Systems (MIPS'03)*, Napoli, Italy, November 2003. [cited at p. 44]

[BRCG07] P. Baccichet, S. Rane, A. Chimienti, and B. Girod. Robust low-delay video transmission using H.264/AVC redundant slices and flexible macroblock ordering. In *Proc. IEEE International Conference on Image Processing (ICIP'07)*, San Antonio, TX, September 2007. [cited at p. 15]

[BS03] E. Balafoutis and I. Stavrakakis. Proxy caching and video segmentation based on request frequencies and access costs. In *Proc. IEEE International Conference on Telecommunications (ICT'03)*, Tahiti, France, February 2003. [cited at p. 81, 82]

[BV07] R. Buyya and S. Venugopal. Smart proxies for accessing replicated web services. *IEEE Distributed Systems Online*, 8(12):1–1, December 2007. [cited at p. 2, 4]

[CAT+04] J. Chakareski, J. Apostolopoulos, W. Tan, S. Wee, and B. Girod. Distortion chains for predicting the video distortion for general packet loss patterns. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'04)*, Montreal, Canada, May 2004. [cited at p. 44, 47, 55]

[CAW+04] J. Chakareski, J. Apostolopoulos, S. Wee, W. Tan, and B. Girod. R-D hint tracks for low-complexity R-D optimized video streaming. In *Proc. IEEE International Conference on Multimedia&Expo (ICME'04)*, Taipei, Taiwan, June 2004. [cited at p. 44, 46]

[CAW+05] J. Chakareski, J. Apostolopoulos, S. Wee, W. Tan, and B. Girod. Rate-distortion hint tracks for adaptive video streaming. *IEEE Transactions on Circuits and Systems for Video Technology*, 15(10):1257–1269, October 2005. [cited at p. 46]

[CC03] T. P. Chen and T. Chen. Rate shaping for video with frame dependency. In *Proc. IEEE International Conference on Multimedia&Expo (ICME'03)*, Baltimore, MD, July 2003. [cited at p. 43]

[CCHO05] S. H. Chang, R. I. Chang, J. M. Ho, and Y. J. Oyang. An optimal cache algorithm for streaming VBR video over a heterogeneous network. *Journal of Computer Communications*, 28(16):1852–1861, February 2005. [cited at p. 80]

[CCL06] C. Canali, V. Cardellini, and R. Lancellotti. Content adaptation architectures based on squid proxy server. *Journal of World Wide Web*, 9(1):63–92, March 2006. [cited at p. 4]

[CF05] J. Chakareski and P. Frossard. Rate-distortion optimized bandwidth adaptation for distributed media delivery. In *Proc. IEEE International Conference on Multimedia&Expo (ICME'05)*, Amsterdam, Netherlands, July 2005. [cited at p. 45]

[CFCL00] N. Celandroni, E. Ferro, F. Potortand A. Chimienti, and M. Lucenteforte. Dynamic rate shaping on MPEG-2 video streams for bandwidth saving on a faded satellite channel. *European Transactions on Telecommunications*, 2(4):363–372, 2000. [cited at p. 43]

[CLZ99] G. Cabri, L. Leonardi, and F. Zambonelli. Supporting cooperative WWW browsing: a proxy-based approach. In *Proc. Euromicro Workshop on Parallel and Distributed Processing (PDP'99)*, Funchal, Portugal, February 1999. [cited at p. 2, 4]

[CM01] P. A. Chou and Z. Miao. Rate-distortion optimized streaming of packetized media. Technical Report MSR-TR-2001-35, Microsoft Research, CA, February 2001. [cited at p. 44]

[COH03] H. Cha, J. Oh, and R. Ha. Dynamic frame dropping for bandwidth control in mpeg streaming system. *Multimedia Tools and Applications*, 19(2):155–178, February 2003. [cited at p. 43]

[CSV+05] G. Canfora, G. Di Santo, G. Venturi, E. Zimeo, and M. V. Zito. Proxy-based handoff of web sessions for user mobility. In *Proc. International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05)*, San Diego, CA, July 2005. [cited at p. 5]

[CSWZ07] S. Chen, B. Shen, S. Wee, and X. Zhang. SProxy: A caching infrastructure to support internet streaming. *IEEE Transactions on Multimedia*, 9(5):1062–1072, August 2007. [cited at p. 81, 82]

[CSY+05] S. Chen, B. Shen, Y. Yan, S. Basu, and X. Zhang. Fast proxy delivery of multiple streamng sessions in shared running buffers. *IEEE Transactions on Multimedia*, 7(6):1157–1169, December 2005. [cited at p. 3, 4]

[CWZ+05] S. Chen, H. Wang, X. Zhang, B. Shen, and S. Wee. Segment-based proxy caching for internet streaming media delivery. *IEEE Multimedia*, 12(3):59–67, July 2005. [cited at p. 81, 82, 104]

[DA99] J. Dilley and M. Arlitt. Improving proxy cache performance: analysis of three replacement policies. *IEEE Internet Computing*, 3(6):44–50, Nov/Dec 1999. [cited at p. 2, 4]

[DMM05] B. Du, A. Maeder, and M. Moody. A new approach on error resilient video coding for live video communication. *International Journal of Innovative Computing, Information and Control*, 1(4):701–713, December 2005. [cited at p. 15]

[EA95] A. Eleftheriadis and D. Anastassiou. Constrained and general dynamic rate shaping of compressed digital video. In *Proc. IEEE International Conference on Image Processing (ICIP'95)*, Washington, DC, USA, October 1995. [cited at p. 43]

[Ela02] H. Elaarg. Improving TCP performance over mobile networks. *ACM Computing Survey*, 34(3):357–374, September 2002. [cited at p. 5]

[EPS03] N. Eagle, T. Pun, and P. Srinivasan. Intelligent frame dropping for rate-distortion optimized video encoding at low bit rates. Technical Report project report EE398b, Stanford University, CA, 2003. [cited at p. 43]

[FGCB98] A. Fox, S. D. Gribble, Y. Chawathe, and E. A. Brewer. Adapting to network and client variation using infrastructural proxies: lessons and perspectives. *IEEE Personal Communications Magazine*, 5(4):10–19, August 1998. [cited at p. 4]

[FLKP98] W. Feng, M. Liu, B. Krishnaswami, and A. Prabhudev. A priority-based technique for the best-effort delivery of stored video. *SPIE/IS Multimedia Computing and Networking 1999*, 3654:286–300, December 1998. [cited at p. 43]

[FLSA+01] H. Fahmi, M. Latif, S. Sedigh-Ali, A. Ghafoor, P. Liu, and L. H. Hsu. Proxy servers for scalable interactive video support. *IEEE Computer*, 34(9):54–60, September 2001. [cited at p. 3, 4]

[FNI96] S. Fukunaga, T. Nakai, and H. Inoue. Error resilient video coding by dynamic replacing of reference pictures. In *Proc. IEEE GLOBECOM'96*, London, UK, November 1996. [cited at p. 17, 18, 24, 30]

[GBW97] C. Griwodz, M. Bör, and L. C. Wolf. Long-term movie popularity models in video-on-demand systems of the life of an on-demand movie. In *Proc. ACM International Conference on Multimedia*, Seattle, Washington, November 1997. [cited at p. 81]

[GF99] B. Girod and N. Färber. Feedback-based error control for mobile video transmission. *Proceedings of the IEEE*, 87(10):1707–1723, October 1999. [cited at p. 17]

[Gha96] M. Ghanbari. Post processing of late cells for packet video. *IEEE Transactions on Circuits and Systems for Video Technology*, 6(6):669–678, December 1996. [cited at p. 17]

[GWA07] V. K. Gurbani, D. Willis, and F. Audet. Cryptographically transparent session initiation protocol (SIP) proxies. In *Proc. IEEE International Conference on Communications (ICC'07)*, Glasgow, Scotland, June 2007. [cited at p. 3]

[HBL+98] R. Han, P. Bhagwat, R. LaMaire, T. Mummert, V. Perret, and J. Rubas. Dynamic adaptation in an image transcoding proxy for mobile web browsing. *IEEE Personal Communications Magazine*, 5(6):8–17, December 1998. [cited at p. 4]

[HCLL05] C. T. Hsu, M. J. Chen, W. W. Liao, and S. Y. Lo. High-performance spatial and temporal error-concealment algorithms for block-based video coding techniques. *ETRI Journal*, 27(1):53–63, March 2005. [cited at p. 15]

[HHIa] HHI. The JSVM reference software. http://ip.hhi.de/imagecom_G1/savce/index.htm. [cited at p. 60, 66]

[HHIb] HHI. JVT H.264/MPEG-4 AVC reference software. [cited at p. 27, 37]

[HHIc] HHI. The JVT H.264/MPEG-4 AVC reference software. http://iphome.hhi.de/suehring/tml/. [cited at p. 65]

[HNG+99] M. Hofmann, T. S. Eugene Ng, K. Guo, S. Paul, and H. Zhang. Caching techniques for streaming multimedia over the internet. Technical Report BL011345-990409-04TM, Bell Labs, Holmdel, NJ, April 1999. [cited at p. 81, 82, 89]

[HPZ+98] Y. T. Hou, S. S. Panwar, Z. L. Zhang, H. Tzeng, and Y. Q. Zhang. On network bandwidth sharing for transporting rate-adaptive packet video using feedback. In *Proc. IEEE Global Telecommunications Conference (GLOBECOM'98)*, Sydney, Australia, November 1998. [cited at p. 44]

[HSLG99] U. Horn, K. Stuhlmüller, M. Link, and B. Girod. Robust internet video transmission based on scalable coding and unequal error protection. *Image Communication, Special Issue on Real-time Video over the Internet*, 15(1):77–94, September 1999. [cited at p. 15]

[ILL07] A. T. S. Ip, J. Liu, and J. C. S. Lui. COPACC: An architecture of cooperative proxy-client caching system for on-demand media streaming. *IEEE Transactions on Parallel and Distributed Systems*, 18(1):70–83, January 2007. [cited at p. 3, 4]

[ITU96] *ITU-T/SG15/LBC-96-033 An error resilience method based on back channel signalling and FEC*. San Jose: Telenor R&D, January 1996. [cited at p. 17, 18]

[JBI03] S. Jin, A. Bestavros, and A. Iyengar. Network-aware partial caching for internet streaming media. *Multimedia Systems*, 9(4):386–396, October 2003. [cited at p. 80, 82, 91]

[KHK04] J. Kritzner, U. Horn, and M. Kampmann. Priority generation for video streaming using stream decodability. In *Proc. International Packet Video Workshop (PV'04)*, Irvine, CA, December 2004. [cited at p. 43]

[KKH04] J. Kritzner, M. Kampmann, and U. Horn. Comparison of frame dropping strategies for adaptive video streaming. In *Proc. International Picture Coding Symposium (PCS'04)*, San Francisco, CA, December 2004. [cited at p. 43]

[KL07] C. F. Kao and C. N. Lee. Aggregate profit-based caching replacement algorithms for streaming media transcoding proxy systems. *IEEE Transactions on Multimedia*, 9(2):221–230, February 2007. [cited at p. 81]

[KWCK05] J. Kim, Y. Wang, S. F. Chang, and H. M. Kim. An optimal framework of video adaptation and its application to rate adaptation transcoding. *ETRI Journal*, 27(4):341–354, August 2005. [cited at p. 44]

[KXMP06] S. Kumar, L. Xu, M. K. Mandal, and S. Panchanathan. Error resiliency schemes in H.264/AVC standard. *Elsevier Journal of Visual Communication & Image Representation*, 17(2):425–450, April 2006. [cited at p. 15]

[LAG03] Y. Liang, J. Apostolopoulos, and B. Girod. Analysis of packet loss for compressed video: Does burst loss matter? In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'03)*, Hongkong, China, April 2003. [cited at p. 44, 55]

[LC99] Y. Lu and K. J. Christensen. Using selective discard to improve real-time video quality on an ethernet local area network. *International Journal of Network Management*, 9(2):106–117, Mar/Apr 1999. [cited at p. 43]

[LC04] A. Leontaris and P. C. Cosman. Video compression for lossy packet networks with mode switching and a dual frame buffer. *IEEE Transactions on Image Processing*, 13(7):885–897, July 2004. [cited at p. 18]

[LCX04] J. Liu, X. Chu, and J. Xu. Proxy cache management for fine-grained scalable video streaming. In *Proc. IEEE International Conference on Computer and Communications (INFOCOM'04)*, Hongkong, China, March 2004. [cited at p. 81, 82]

[LFG02] Y. Liang, M. Flierl, and B. Girod. Low-latency video transmission over lossy packet networks using rate-distortion optimized reference picture selection. In *Proc. IEEE International Conference on Image Processing (ICIP'02)*, Rochester, NY, September 2002. [cited at p. 17, 18]

[LG03] Z. Lei and N. D. Georganas. A rate adaptation transcoding scheme for real-time video transmission over wireless channels. *Journal of Signal Processing: Image Communication*, 18(8):641–658, September 2003. [cited at p. 43]

[LL02] W. Y. Lum and F. C. M. Lau. A context-aware decision engine for content adaptation. 1(3):41–49, Jul-Sep 2002. [cited at p. 4]

[LLLW04] L. S. Lam, Y. B. Lee, S. C. Liew, and W. Wang. A transparent rate adaptation algorithm for streaming video over the internet. In *Proc. International Conference on Advanced Information Networking and Applications (AINA'04)*, Fukuoka, Japan, March 2004. [cited at p. 43]

[LSWS06] G. Liebl, T. Schierl, T. Wiegand, and T. Stockhammer. Advanced wireless multiuser video streaming using the scalable video coding extensions of H.264/MPEG4-AVC. In *Proc. IEEE International Conference on Multimedia&Expo (ICME'06)*, Toronto, Canada, July 2006. [cited at p. 44]

[LTS05] K. Li, K. Tajima, and H. Shen. Cache replacement for transcoding proxy caching. In *Proc. IEEE International Conference on Web Intelligence (WI'05)*, Compiegne, France, September 2005. [cited at p. 81]

[LTS07] G. Liebl, W. Tu, and E. Steinbach. Proxy-based transmission strategies for wireless video streaming. In *Proc. International Packet Video Workshop (PV'07)*, Lausanne, Switzerland, November 2007. [cited at p. 109]

[LTS08] X. Li, W. Tu, and E. Steinbach. Dynamic segment based proxy caching for video on demand. In *Proc. IEEE International Conference on Multimedia and Expo (ICME'08)*, Hannover, Germany, June 2008. [cited at p. 10]

[LV00] J. Y. Liao and J. Villasenor. Adaptive intra block update for robust transmission of H.263. *IEEE Transactions on Circuits and Systems for Video Technology*, 10(1):30–35, February 2000. [cited at p. 15]

[LW03] Y. Li and L. Wolf. Adaptive resource management in active network nodes. In *Proc. IEEE International Symposium on Computers and Communication (ISCC'03)*, Antalya, Turkey, Jun-Jul 2003. [cited at p. 4]

[LZQ03] T. Liu, H. Zhang, and F. Qi. Perceptual frame dropping in adaptive video streaming. In *Proc. IEEE International Symposium on Computers and Communication (ISCC'03)*, Antalya, Turkey, June 2003. [cited at p. 43]

[MCCdL06] I. Mohomed, J. C. Cai, S. Chavoshi, and E. de Lara. Context-aware interactive content adaptation. In *Proc. International Conference On Mobile Systems, Applications And Services (MobiSYS'06)*, Uppsala, Sweden, June 2006. [cited at p. 4]

[MD02] W. H. Ma and D. H. C. Du. Reducing bandwidth requirement for delivering video over wide area networks with proxy server. *IEEE Transactions on Multimedia*, 4(4):539–550, December 2002. [cited at p. 80]

[MFW01] R. Mahajan, S. Floyd, and D. Wetherall. Controlling high-bandwidth flows at the congested router. In *Proc. IEEE International Conference on Network Protocols (ICNP'01)*, Riverside, CA, November 2001. [cited at p. 4, 43]

[MGC+01] L. Munoz, M. Garcia, J. Choque, R. Aguero, and P. Mahonen. Optimizing internet flows over IEEE 802.11b wireless local area networks: a performance-enhancing proxy based on forward error correction. *IEEE Computer*, 39(12):60–67, December 2001. [cited at p. 3]

[MO99] Z. Miao and A. Ortega. Proxy caching for efficient video services over the internet. In *Proc. International Packet Video Workshop (PV'99)*, New York, NY, April 1999. [cited at p. 80]

[MO02] Z. Miao and A. Ortega. Scalable proxy caching of video under storage constrains. *IEEE Journal on Selected Areas in Communications*, 20(7):1315–1327, September 2002. [cited at p. 80]

[Moo02] T. Moors. A critical review of ‘end-to-end arguments in system design’. In *Proc. IEEE International Conference on Communications (ICC'02)*, New York, NY, Apr-May 2002. [cited at p. 2]

[MSH03] M. Meyer, J. Sachs, and M. Holzke. Performance evaluation of a TCP proxy in WCDMA networks. *IEEE Transactions on Wireless Communications*, 10(5):70–79, October 2003. [cited at p. 3]

[MTS07] R. Mahalingam, W. Tu, and E. Steinbach. RD-optimized rate shaping for multiple scalable video streams. In *Proc. IEEE International Conference on Multimedia and Expo (ICME'07)*, Beijing, China, July 2007. [cited at p. 10]

[MTS08] M. Muhammad, W. Tu, and E. Steinbach. Segment-based proxy caching for video on demand services. In *Proc. IEEE International Conference on Multimedia and Expo (ICME'08)*, Hannover, Germany, June 2008. [cited at p. 10]

[MW00] J. Meggers and M. Wallbaum. Application level error recovery using active network nodes. In *Proc. IEEE Symposium on Computers and Communications (ISCC'00)*, Antibes-Juan les Pins, France, July 2000. [cited at p. 3]

[MYRM04] E. Masala, H. Yang, K. Rose, and J. Carlos De Martin. Rate-distortion optimized slicing, packetization and coding for error resilient video transmission. In *Proc. IEEE Data Compression Conference (DCC'04)*, Snowbird, Utah, March 2004. [cited at p. 44]

[Neu93] B. C. Neuman. Proxy-based authorization and accounting for distributed systems. In *Proc. International Conference on Distributed Computing Systems (ICDCS'93)*, Pittsburgh, PA, May 1993. [cited at p. 5]

[OS06] H. R. Oh and H. Song. Scalable proxy caching algorithm minimizing client's buffer size and channel bandwidth. *Journal of Visual Communication and Image Representation*, 17(1):57–71, February 2006. [cited at p. 81]

[OS07] H. R. Oh and H. Song. Metafile-based scalable caching and dynamic replacing algorithms for multiple videos over quality-of-service networks. *IEEE Transactions on Multimedia*, 9(7):1535–1542, November 2007. [cited at p. 81]

[PKL03] Y. O. Park, C. S. Kim, and S. U. Lee. Multi-hypothesis error concealment algorithm for H.26L video. In *Proc. IEEE International Conference on Image Processing (ICIP'03)*, Barcelona, Spain, September 2003. [cited at p. 15]

[PLC01] S. H. Park, E. J. Lim, and K. D. Chung. Popularity-based partial caching for vod systems using a proxy server. In *Proc. IEEE International Parallel and Distributed Processing Symposium (IPDPS'01)*, San Francisco, CA, April 2001. [cited at p. 81, 82, 91]

[PM03] Y. Pei and J. W. Modestino. Interactive video coding and transmission over wired-to-wireless ip networks using an edge proxy. In *Proc. IEEE International Conference on Mulitmedia&Expo (ICME'03)*, Baltimore, MD, July 2003. [cited at p. 3]

[PPBK07] P. Pahalawatta, T. Pappas, R. Berry, and A. Katsaggelos. Content-aware resource allocation for scalable video transmission to multiple users over a wireless network. In *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'07)*, Honolulu, Hawaii, April 2007. [cited at p. 44]

[PZ07] S. Pradhan and A. Zaslavsky. A smart proxy for a next generation web services transaction. In *Proc. IEEE/ACIS International Conference on Computer and Information Science (ICIS'07)*, Montréal, Canada, December 2007. [cited at p. 2, 4]

[QLS+05] W. Qu, K. Li, H. Shen, Y. Jin, and T. Nanya. The cache replacement problem for multimedia object caching. In *Proc. IEEE International Conference on Semantics, Knowledge, and Grid (SKG'05)*, Beijing, China, November 2005. [cited at p. 81]

[RFC98] *RFC 2309 Recommendations on Queue Management and Congestion Avoidance in the Internet*, 1998. [cited at p. 43]

[RJ00] I. Rhee and S. Joshi. Error recovery for interactive video transmission over the internet. *IEEE Journal on Selected Areas in Communications*, 18(6):1033–1049, June 2000. [cited at p. 17, 20, 23]

[RK01] R. Rejaie and J. Kangasharju. Mocha: A quality adaptive multimedia proxy cache for internet streaming. In *Proc. International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV'01)*, Port Jefferson, NY, June 2001. [cited at p. 81, 82]

[RQ08] B. Rong and Y. Qian. An enhanced sip proxy server for wireless voip in wireless mesh networks. *IEEE Communications Magazine*, 46(1):108–113, January 2008. [cited at p. 3]

[SCK02] H. Song, H. Chu, and S. Kurakake. Browser session preservation and migration. In *Proc. international conference on World Wide Web (WWW'02)*, Hawaii, USA, May 2002. [cited at p. 5]

[SFG97] E. Steinbach, N. Färber, and B. Girod. Standard compatible extension of H.263 for robust video transmission in mobile environments. *IEEE Transactions on Circuits and Systems for Video Technology*, 7(6):872–881, December 1997. [cited at p. 17]

[SFG00] K. Stuhlmüller, N. Färber, and B. Girod. Adaptive optimal intra-update for lossy video transmission. In *Proc. SPIE Visual Communications and Image Processing (VCIP'00)*, Perth, Australia, June 2000. [cited at p. 15, 16, 17]

[SHW03] T. Stockhammer, M. Hannuksela, and T. Wiegand. H.264/AVC in wireless environments. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):657–673, July 2003. [cited at p. 15, 17, 18, 28]

[SLVM06] C. Soldani, G. Leduc, F. Verdicchio, and A. Munteanu. Multiple description coding versus transport layer fec for resilient video transmission. In *Proc. International Conference on Digital Telecommunications (ICDT'06)*, Côte d'Azur, France, August 2006. [cited at p. 16]

[SMW06] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable H.264/MPEG4-AVC extension. In *Proc. IEEE International Conference on Image Processing (ICIP'06)*, Atlanta, GA, October 2006. [cited at p. 42, 60]

[SMW07] H. Schwarz, D. Marpe, and T. Wiegand. Overview of the scalable video coding extension of the H.264/AVC standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(9):1103–1120, September 2007. [cited at p. 8]

[SOR03] U. Sezer, S. H. Oguz, and P. Ramanathan. Efficient rate adaptation of precompressed video to network constraints via controlled noise injection. In *Proc. IEEE International Symposium on Computers and Communication (ISCC'03)*, Antalya, Turkey, June 2003. [cited at p. 43]

[SPvSA07] S. Sivasubramanian, G. Pierre, M. van Steen, and G. Alonso. Analysis of caching and replication strategies for web applications. *IEEE Internet Computing*, 11(1):60–66, Jan-Feb 2007. [cited at p. 3, 4]

[SRC84] J. Saltzer, D. Reed, and D. Clark. End-to-end arguments in system design. *ACM Transactions on Computer Systems*, 2(4):277–288, November 1984. [cited at p. 2]

[SRT99] S. Sen, J. Rexford, and D. Towsley. Proxy prefix caching for multimedia streams. In *Proc. IEEE International Conference on Computer and Communications (INFOCOM'99)*, New York, NY, April 1999. [cited at p. 81]

[SSV99] J. Shim, P. Scheuermann, and R. Vingralek. Proxy cache algorithms: Design, implementation, and performance. *IEEE Transactions on Knowledge and Data Engineering*, 11(4):549–562, Jul/Aug 1999. [cited at p. 2, 4]

[Sto02] T. Stockhammer. Error robust macroblock mode and reference frame selection. In *Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG JVT-B102*, Geneva, Switzerland, January 2002. [cited at p. 15, 16]

[STS07] L. Shen, W. Tu, and E. Steinbach. A flexible starting point based partial caching algorithm for video on demand. In *Proc. IEEE International Conference on Multimedia and Expo (ICME'07)*, Beijing, China, July 2007. [cited at p. 10, 82, 83, 84, 101, 104]

[SW98] G. J. Sullivan and T. Wiegand. Rate-distortion optimization for video compression. *IEEE Signal Processing Magazine*, 15(6):74–90, November 1998. [cited at p. 44]

[SW02] H. Schwarz and T. Wiegand. An improved MPEG-4 coder using lagrangian coder control. In *Proc. International ITG Conference on Source and Channel Coding (SCC'02)*, Berlin, Germany, January 2002. [cited at p. 44]

[SW04] T. Schierl and T. Wiegand. H.264/AVC rate adaptation for internet streaming. In *Proc. International Packet Video Workshop (PV'04)*, Irvine, CA, December 2004. [cited at p. 44]

[TCS06] W. Tu, J. Chakareski, and E. Steinbach. Rate-distortion optimized frame dropping and scheduling for multi-user conversational and streaming video. In *Proc. International Packet Video Workshop (PV'06)*, Hangzhou, China, April 2006. [cited at p. 10, 56]

[TCS08] W. Tu, J. Chakareski, and E. Steinbach. Rate-distortion optimized frame dropping for multiuser streaming and conversational videos. *Advances in Multimedia, Article ID 628970*, January 2008. [cited at p. 10]

[TKS04] W. Tu, W. Kellerer, and E. Steinbach. Rate-distortion optimized video frame dropping on active network nodes. In *Proc. International Packet Video Workshop (PV'04)*, Irvine, CA, December 2004. [cited at p. 10, 50, 63]

[TS04] W. Tu and E. Steinbach. Proxy-based error tracking for H.264 based real-time video transmission in mobile environments. In *Proc. IEEE International Conference on Multimedia&Expo (ICME'04)*, Taipei, Taiwan, June 2004. [cited at p. 3, 9, 17]

[TS05] W. Tu and E. Steinbach. Proxy-based reference picture selection for real-time video transmission over mobile networks. In *Proc. IEEE International Conference on Multimedia&Expo (ICME'05)*, Amsterdam, Netherlands, July 2005. [cited at p. 9]

[TS09] W. Tu and E. Steinbach. Proxy-based reference picture selection for error resilient conversational video in mobile networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 19(2):151–164, February 2009. [cited at p. 9]

[TSMLed] W. Tu, E. Steinbach, M. Muhammad, and X. Li. Proxy caching for video on demand using flexible starting point selection. *IEEE Transactions on Multimedia*, accepted. [cited at p. 10]

[TTM06] J. Takahashi, H. Tode, and K. Murakami. A hybrid fec method using packet-level convolution and reed-solomon codes. *IEICE Transactions on Communications*, E89-B(8):2143–2151, March 2006. [cited at p. 15]

[TZM08] T. Tang and R. Peng Z. Mi. Adaptive service provisioning through context-aware sip proxy. In *Proc. International Conference on Networking and Services (ICNS'08)*, Gosier, Guadeloupe, March 2008. [cited at p. 3]

[VCC02] A. Vetro, J. Cai, and C. W. Chen. Rate-reduction transcoding design for wireless video streaming. *Journal of Wireless Communications and Mobile Computing*, 2(6):549–552, September 2002. [cited at p. 43]

[VCS03] A. Vetro, C. Christopoulos, and H. Sun. Video transcoding architectures and techniques: An overview. *IEEE Signal Processing Magazine*, 20(2):18–29, March 2003. [cited at p. 43]

[Vid] VideoLAN. VLC media player. http://www.videolan.org/vlc/. [cited at p. 84]

[WFC02] T. C. Wang, H. C. Fang, and L. G. Chen. Low-delay and error-robust wireless video transmission for video communications. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(12):1049–1058, December 2002. [cited at p. 3]

[WFS00] T. Wiegand, N. Färber, and K. Stuhlmüller. Error-resilient video transmission using long-term memory motion-compensated prediction. *IEEE Journal on Selected Areas in Communications*, 18(6):1050–1062, June 2000. [cited at p. 17, 18]

[WHV+02] Y. K. Wang, M. Hannuksela, V. Varsa, A. Hourunranta, and M. Gabbouj. The error concealment feature in the H.26L test model. In *Proc. IEEE International Conference on Image Processing (ICIP'02)*, Rochester, NY, September 2002. [cited at p. 15, 27, 33]

[WLSC98] M. H. Willebeek-LeMair, Z. Y. Shae, and Y. C. Chang. Robust H.263 video coding for transmission over Internet. In *Proc. INFOCOMM'98*, San Francisco, CA, 1998. [cited at p. 15]

[WOM+00] A. Warabino, S. Ota, D. Morikawa, M. Ohashi, H. Nakamura, H. Iwashita, and F. Watanabe. Video transcoding proxy for 3G wireless mobile Internet access. *IEEE Communications Magazine*, 38(10):66–71, October 2000. [cited at p. 4]

[WSA08] W. V. Wathsala, B. Siddhisena, and A. S. Athukorale. Next generation proxy servers. In *Proc. IEEE International Conference on Advanced Communication Technology (ICACT'08)*, Phoenix Park, Korea, February 2008. [cited at p. 2, 4]

[WSAT02] B. Wang, S. Sen, M. Adler, and D. Towsley. Optimal proxy cache allocation for efficient streaming media distribution. In *Proc. IEEE INFOCOM'02*, New York, NY, June 2002. [cited at p. 3, 4]

[WW07] D. Wang and F. WAN. A proxy-based architecture for multimedia transmission. In *Proc. WSEAS International Conference on Automation and Information (ICAI'07)*, Vancouver, Canada, June 2007. [cited at p. 3, 4]

[WWWK00] Y. Wang, S. Wenger, J. T. Wen, and A. K. Katsaggelos. Review of error resilient coding techniques for real-time video communications. *IEEE Signal Processing Magazine*, 17(4):61–82, July 2000. [cited at p. 15]

[WY07] J. Z. Wang and P. S. Yu. Fragmental proxy caching for streaming multimedia objects. *IEEE Transactions on Multimedia*, 9(1):147–156, January 2007. [cited at p. 81, 82, 84, 89, 97]

[WYW01] K. L. Wu, P. S. Yu, and J. L. Wolf. Segment-based proxy caching of multimedia streams. In *Proc. International Conference on World Wide Web (WWW'01)*, Hongkong, China, May 2001. [cited at p. 81, 82, 104]

[WYW04] K. L. Wu, P. S. Yu, and J. L. Wolf. Segmentation of multimedia streams for proxy caching. *IEEE Transactions on Multimedia*, 6(5):770–780, October 2004. [cited at p. 81, 82, 104]

[WZ98] Y. Wang and Q. Zhu. Error control and concealment for video communications: A review. *Proceedings of the IEEE*, 86(5):974–997, May 1998. [cited at p. 15]

[XLS05] J. Xin, C. W. Lin, and M. T. Sun. Digital video transcoding. *Proceedings of the IEEE*, 93(1):84–97, November 2005. [cited at p. 43]

[YCDW06] J. Yu, C. T. Chou, X. Du, and T. Wang. Internal popularity of streaming video and its implication on caching. In *Proc. IEEE International Conference on Advanced Information Networking and Application (AINA'06)*, Vienna, Austria, April 2006. [cited at p. 82, 91, 97]

[YL03] H. Yan and D. K. Lowenthal. Popularity-aware cache replacement in streaming environments. In *Proc. International Conference on Parallel and Distributed Computing Systems (PDCS'03)*, Reno, Nevada, August 2003. [cited at p. 82, 91]

[YR03] H. Yang and K. Rose. Recursive end-to-end distortion estimation with model-based cross-correlation approximation. In *Proc. IEEE International Conference on Image Processing (ICIP'03)*, Barcelona, Spain, September 2003. [cited at p. 15]

[YR07] H. Yang and K. Rose. Advances in recursive per-pixel end-to-end distortion estimation for robust video coding in H.264/AVC. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(7):845–856, July 2007. [cited at p. 15]

[YVXL03] P. Yin, A. Vetro, M. Xia, and B. Liu. Rate-distortion models for video transcoding. In *Proc. SPIE Conference on Image and Video Communications and Processing*, volume 5022, pages 479–488, Berlin, Germany, May 2003. [cited at p. 43]

[YXS00] J. Youn, J. Xin, and M. T. Sun. Fast video transcoding architectures for networked multimedia applications. In *Proc. IEEE International Symposium on Circuits and Systems (ISCAS'00)*, Geneva, Switzerland, May 2000. [cited at p. 43]

[YYW04] H. Yu, S. Yu, and C. Wang. An effective architecture for interactive wireless video communication. In *Proc. IEEE International Conference on Vehicular Technology (VTC'04)*, Los Angeles, CA, May 2004. [cited at p. 3]

[ZA01] B. Zheng and M. Atiquzzaman. Two stage frame dropping for scalable video transmission over data networks. In *Proc. IEEE Workshop on High Performance Switching and Routing (HPSR'01)*, Dallas, TX, May 2001. [cited at p. 44]

[ZBPK03] F. Zhai, R. Berry, T. N. Pappas, and A. K. Katsaggelos. A rate-distortion optimized error control scheme for scalable video streaming over the internet. In *Proc. IEEE International Conference on Multimedia&Expo (ICME'03)*, Baltimore, MD, July 2003. [cited at p. 44]

[ZEP+06] F. Zhai, Y. Eisenberg, T. N. Pappas, R. Berry, and A. K. Katsaggelos. Rate-distortion optimized hybrid error control for real-time packetized video transmission. *IEEE Transactions on Image Processing*, 15(1):40–53, January 2006. [cited at p. 15]

[Zip49] G. K. Zipf. *Human Behaviour and the Principles of Least Effort.* Addison-Wesley, Cambridge, MA, 1949. [cited at p. 82, 97]

[ZL96] W. Zeng and B. Liu. Rate shaping by block dropping for transmission of MPEG-precoded video over channels of dynamic bandwidth. In *Proc. ACM International Conference on Multimedia (MULTIMEDIA'96)*, Boston, MA, November 1996. [cited at p. 43]

[ZNAT99] Z. Zhang, S. Nelakuditi, R. Aggarwal, and R. P. Tsang. Efficient selective frame discard algorithms for stored video delivery across resource constrained networks. In *Proc. IEEE Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'99)*, New York, NY, March 1999. [cited at p. 43]

[ZRR00] R. Zhang, S. L. Regunathan, and K. Rose. Video coding with optimal inter/intra-mode switching for packet loss resilience. *IEEE Journal on Selected Areas in Communications*, 18(6):966–976, June 2000. [cited at p. 15]

[ZRR01] R. Zhang, S. L. Regunathan, and K. Rose. End-to-end distortion estimation for RD-based robust delivery of pre-compressed video. In *Proc. Asilomar Conference on Signals, Systems and Computers (ASILOMAR'01)*, Monterey, CA, November 2001. [cited at p. 44]

[ZSL05] C. Zheng, G. Shen, and S. Li. Distributed prefetching scheme for random seek support in peertopeer streaming applications. In *Proc. ACM Workshop on Advances in Peer-to-Peer Multimedia Streaming*, Hilton, Singapore, November 2005. [cited at p. 82, 86, 91]

[ZWDS00] Z. Zhang, Y. Wang, D. H. C. Du, and D. Su. Video staging: A proxy-server-based approach to end-to-end video delivery over wide-area networks. *IEEE/ACM Transactions on Networking*, 8(4):429–442, August 2000. [cited at p. 80]

[ZWX06] Z. Zhuang, J. Wu, and Q. Xia. Study & implementation of proxy-based IPv4/IPv6 transition mechanism. In *Proc. Asia-Pacific Conference on Communications (APCC'06)*, Busan, Korea, Aug-Sep 2006. [cited at p. 5]