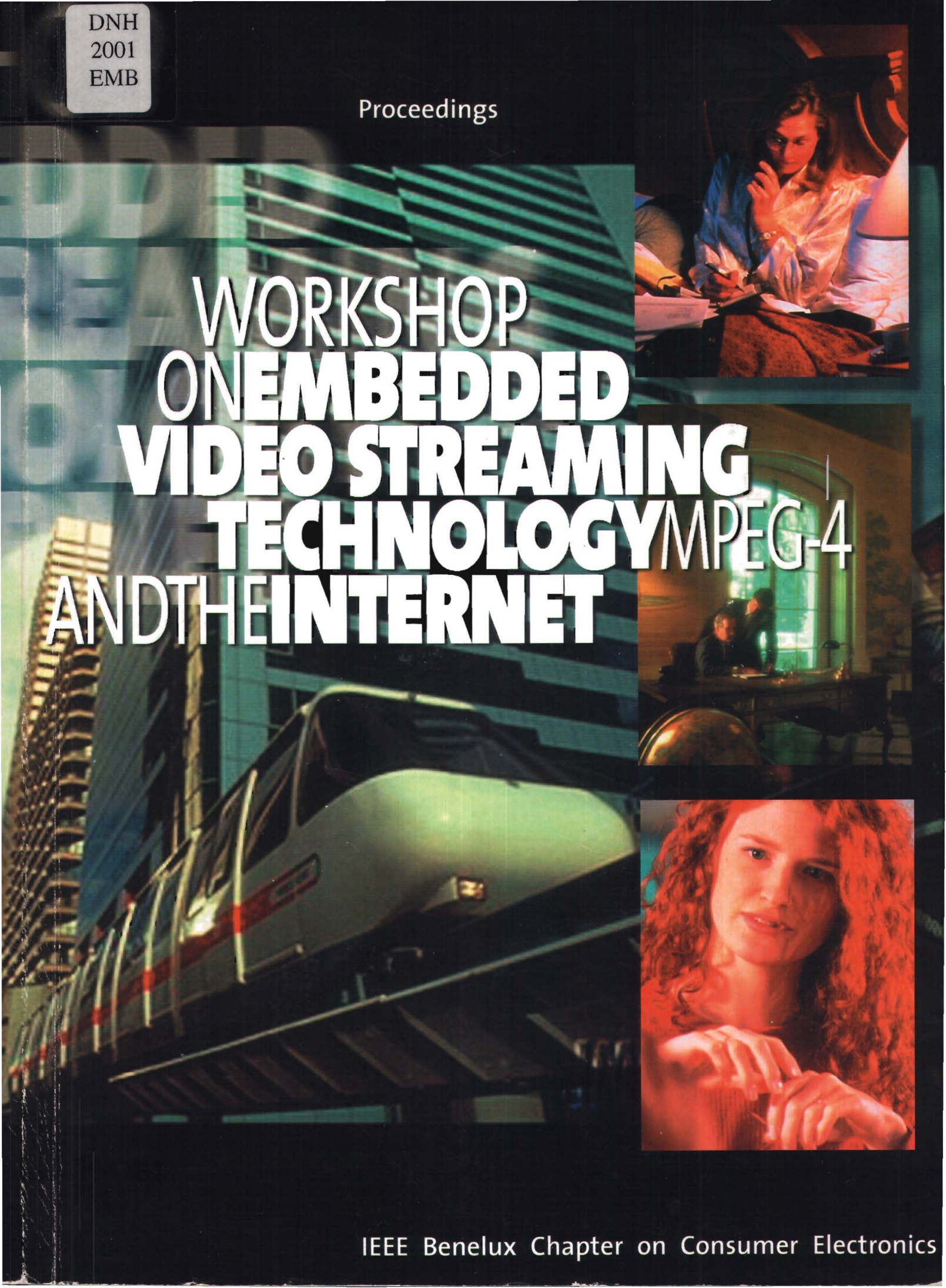
Embedded video streaming technology (MPEG-4) and the Internet : proceedings of a one-day workshop, Eindhoven, December 20, 2001

Proceedings



WORKSHOP ON EMBEDDED VIDEO STREAMING TECHNOLOGY MPEG-4 AND THE INTERNET

IEEE Benelux Chapter on Consumer Electronics

Peter H.N. de With (ed.)

Embedded Video Streaming Technology (MPEG-4) and the Internet

Proceedings of a one-day workshop

organized by

the IEEE Benelux Chapter on Consumer Electronics

in co-operation with the EESI

December 20, 2001

Sponsors

Embedded

Embedded video streaming technology (MPEG-4) and the Internet : proceedings of a one-day workshop, Eindhoven, December 20, 2001 / ed. by Peter H.N. de With. - Eindhoven : Technische Universiteit Eindhoven, 2001.
ISBN 90-386-0991-4
NUGI 832
Subject headings: image compression : proceedings / embedded systems : proceedings / digital television / video technology
CR Subject Classification (1998): I.4.2, C.3, C.2.5

CONTRIBUTORS

Jan van der Meer is with Philips Consumer Electronics, Eindhoven, The Netherlands, and has been involved in the creation of a series of digital video standards and products. Initially he was involved in the predevelopment of the first digital video processing systems. He joined the MPEG standard committee from the beginning and contributed since 1989 to the developments of the MPEG-1, MPEG-2 and MPEG-4 standards. In 1990 he was responsible for developing the CD-i standard and later involved in succeeding products based on MPEG-1, such as Video CD. He was also responsible for the development of Digital TV, based on MPEG-2. He is world-wide acknowledged for his contributions to MPEG. Currently he is with MP4Net, the Business Group within Philips that focuses on MPEG-4 based products, where he is also involved in standard bodies for specific application areas where the use of MPEG-4 is discussed, such as 3GPP, ISMA and DVB.

John W. Woods received the Ph.D. degree from the Massachusetts Institute of Technology in 1970. Since 1976 he has been with the ECSE Department at Rensselaer Polytechnic Institute, where he is currently Director of the NSF I/U Center for Next Generation Video. Dr. Woods was co-recipient of 1976 and 1987 Senior Paper Awards of the now IEEE Signal Processing (SP) Society. He served on the editorial board of Graphical Models and Image Processing and was chairman of the Seventh Workshop on Multidimensional Signal Processing in 1991. He served as Technical Program Co-chairman for the First IEEE International Conference on Image Processing (ICIP) held in Austin, Texas in November 1994. He received the Signal Processing Society Meritorious Service Award in 1990. He is currently on the editorial board of the IEEE Transactions on Video Technology. He received a Technical Achievement Award from the IEEE Signal Processing Society in 1994. He received the IEEE Third Millenium medal in 2000. Dr. Woods is a Fellow of the IEEE.

A.J. Han Vinck is a full professor in Digital Communications at the University of Essen, Essen, Germany, since 1990. He studied electrical engineering at the University of Eindhoven, The Netherlands, where he obtained his Ph.D. in 1980. His interest is in Information and Communication theory, Coding and Network aspects in digital communications. Professor Vinck is the initiator of the Japanese-Benelux workshops on Information theory and the International winter-meeting on Coding, Cryptography and Information theory. In 1997 he was the chairman for the IEEE ISIT in Ulm. He started research and still supports the organization of the series of conferences on Power Line Communications and its Applications (ISPLC). From 1998-2001 he was elected chairman of the Benelux Information and Communication Theory Society. Professor Vinck serves on the Board of Governors of the IEEE Information Theory Society since 1997 (until 2003). In 2001 he was elected 1st vice president of the IEEE Information Theory Society.

Mihaela van der Schaar graduated in electrical engineering from Eindhoven University of Technology, the Netherlands, in April 1996. In the same year she joined Philips Research Laboratories Eindhoven, where she became a Research Scientist in the TV Systems Department working on several projects investigating low-cost high quality video compression techniques and their implementation for TV, computer and camera systems. Between 1998 and April 2001, she was an expatriate at Philips Research Briarcliff, USA, in the Video Communications Department. Currently, she is a Senior Member Research staff at Philips Research USA, where she is involved in the research of video coding techniques for Internet and wireless video streaming and leads a team of researchers working on scalable video coding, networking, and streaming algorithms. Since 1999, she is an active participant to the MPEG-4 video standard, contributing to the "Fine Granularity Scalability" tool. Her research interests include video and graphics coding and video streaming over unreliable channels and she co-authored more than 30 conference and journal papers in this field and holds several patents. In July 2001, she chaired a special session on "Internet video streaming" at ISAS/SCI 2001.

Warner R.Th. ten Kate graduated cum laude in electrical engineering at the Delft University of Technology in 1982. His work was awarded by the Delft University Fund in 1983. During the final stage his research was directed at solar cells of amorphous silicon and silicon radiation detectors. He received the Ph.D. degree in 1987. Since 1988, Dr ten Kate has been working in the Acoustics group of Philips Research Laboratories. His main research interest was multichannel surround sound systems; first in HDTV-environments, later in the context of psychoacoustic data compression. This work has contributed to the MPEG-2 International Standard (ISO/IEC-13818). Another area of interest was to obtain optimal diffuse reflection, on which he worked a period in collaboration with the media company Polygram. In 1995 he moved his interests to multimedia applications, particularly concerning the use of internet technology in TV type of applications. In this area work has been performed on hypermedia for presentation systems and on technology to improve streaming over IP networks. As an active member in some W3C working groups he contributed a.o. to the specification of the SMIL recommendations. His main interests focus on media streaming. He has written a multitude of international papers and holds a number of patents in the field.

Egbert Jaspers was born in Nijmegen, The Netherlands, in 1969. He graduated in electrical engineering from the Venlo Polytechnic in 1992 and subsequently, he joined Philips Research Laboratories in Eindhoven. He continued his education at the Eindhoven University of Technology, and graduated(MSc)in electrical engineering in 1996. Afterwards, he joined Philips Research Labs Eindhoven, where he became a member of the TV Systems Department. There he worked on video compression for digital HDTV recording. Currently he is involved in the research of programmable architectures and their implementation for consumer systems and working towards a Ph.D. degree. In 2000 he received an IEEE Consumer Electronics Section Paper Award.

Peter H.N. de With graduated (MSc.) in electrical engineering from the University of Technology Eindhoven and received his Ph.D. degree from the University of Technology Delft, The Netherlands in 1992. He joined Philips Research Labs Eindhoven in 1984, where he became a member of the Magnetic Recording Systems Department. From 1985 to 1993 he was involved in several European projects on SDTV and HDTV recording and he contributed as a coding expert to the DV standardization. In 1994 he became a member of the TV Systems group, where he was leading the design of advanced programmable video architectures. In 1996, he became senior TV systems architect and in 1997, he was appointed as full professor at the University of Mannheim, Germany, at the faculty Computer Engineering. In 2000, he joined CMG Eindhoven as a principal consultant and he became professor at the University of Technology Eindhoven, at the embedded systems institute (EESI). He has written numerous papers on video coding, architectures and their realization and holds over 40 patents. In 1995 and 2000, he co-authored papers that received the IEEE CES Transactions Paper Award. In 1996, he obtained a company Invention Award. In 1997, Philips received the ITVA Award for its contributions to the DV standard. Mr. de With is a senior member of the IEEE, program committee member of the IEEE CES and board member of various working groups and currently chairman of the Benelux Information and Communication Theory Group.

Program

09.00-09.30 hrs	Registration and coffee
09.30 hrs.	Welcome address, Prof.dr.ir. Peter H.N. de With, (EESI, Eindhoven Univ. of Technology, CMG)
09.40-10.30 hrs.	Ir. Jan van der Meer (Philips Consumer Electronics, MP4Net) "Developments and Overview of MPEG-4 Streaming"
Break	
11.00-11.50 hrs.	Prof.dr. John W. Woods and I.V. Bajic (Rensselaer Polytechnic Institute, USA) "Domain-based Multiple Description Coding of Images and Video"
11.50-12.40 hrs.	Prof.dr.ir. Han A.J. Vinck (University of Essen, Germany) "Transmission and coding problems for (e.g. IP) networks"
Lunch	
13.50-14.40 hrs.	Ir. Mihaela van der Schaar (Philips Research Briarcliff, USA) "Fine Granular Scalability in MPEG-4 for wireless streaming"
14.40-15.30	Dr.ir. Warner ten Kate (Philips Research Labs, Eindhoven) "Multicast Streaming: Sharing Continuous Media on the Internet"
15.30 hrs.	Closing

Contents

Preface

The IEEE Chapter on Consumer Electronics in the Benelux has been founded recently and supports events that are related to the application area of Consumer Electronics. This application area is growing continuously, because computing and storage devices are shrinking by means of advances in technology development. An example of technology improvement is the increased density of transistors in a chip, which enables advanced functionality inside e.g. portable systems. Another example of technology improvement is the increased density of media for storage applications. This phenomenon allows Gigabyte hard disks in personal computers, high-quality movies on an optical disk or very small cassettes in camcorders. Both examples provide important fundaments for the attractive consumer electronics products currently existing in the market.

In the past decade, digital coding of video signals has resulted in a number of significant standards and most of these standards are used in practice. This is because after the tremendous success of digital audio on Compact Disc and all its derivatives (CD-ROM, etc.), there was a need of digital video and the consumers were willing to step into digital video technology, where it was proposed in a cost-efficient way. The most important standard for moving digital video signals is beyond doubt the MPEG standard, of which the MPEG-2 video standard is most widely applied.

In parallel with this, computers were connected and grouped inside networks and the Internet emerged. Portable computers were embraced and the connection to video equipment such as camcorders became apparent. This is just an example how computers and conventional consumer electronics are merging. The communication paradigm from computer networking is gradually becoming part of the consumer electronics area, leading to communicating consumer video over the Internet. This reasoning has brought us to the theme of the workshop: Streaming video technology (MPEG-4) and the Internet.

MPEG-4 is a very recent standard proposal from 2000 and positions itself in the middle of the developments mentioned above. It is a standard intended for a broad range of video applications, from low-rate video conferencing at 64 kbit/s up to broadcast TV in the Mbit/s range. One of the focus points of the workshop is that the Internet channel suffers from packet loss, so that digital video is interrupted possibly leading to severe errors in the recovered bit stream. MPEG-4 offers various techniques, e.g. scalability, to cope with such errors without decoding problems while maintaining running video.

The robustness and scalability theme is addressed in the lecture of Dr. Mihaela van der Schaar of Philips Research Briarcliff when the fine granular scalability concept of MPEG-4 is discussed. She has participated actively in the MPEG-4 committee and was personally involved in special extensions of the standard. We are pleased that the consumer electronics division of Philips also contributes by the lecture of Jan van der Meer, who is internationally acknowledged for his contributions to MPEG for more than 10 years. He will present an overview of the MPEG-4 standard, and relate it to other international developments.

We are particularly honoured that Prof.dr. John Woods will address specific extensions and technologies for robust packetized transmission of digital video. Prof. Woods, affiliated with the Rensselaer Polytechnic Institute in the USA, is a leading expert in video coding and well known in the field via numerous publications, awards and a number of books. His book about Subband Coding from the early nineties is widely accepted as a standard textbook on image coding. We are very happy to have him participating in the workshop program.

The current Internet and related techniques for solving network problems (packet loss and congestion) using advanced channel coding, are discussed by Prof. Han Vinck of the University of Essen, Germany. Prof. Vinck has been working on networks and channel codes since the eighties. He is the organizer of numerous events on coding topics, a founder of the power line communication research in Germany, a leading person in the coding community and he is the upcoming president of the IEEE Information Theory Society. We are pleased that he contributes to the workshop. Dr. Warner ten Kate of Philips Research Eindhoven is addressing Internet streaming as the central theme. He has become well known for his work on MPEG audio coding, contributions to the MPEG standard and for his early Internet streaming experiments. His contribution fits perfectly in the workshop program.

The workshop was organised by the IEEE Benelux Chapter on Consumer Electronics. The Chapter is part of the international IEEE CE Society. In the past year, the Chapter has contributed to other events and is now widening its activities. A liaison with the Audio Engineering Society is under consideration. The Chapter gratefully acknowledges the Eindhoven Embedded Systems Institute (EESI) of the University of Technology Eindhoven for its contributions to this workshop (e.g. these proceedings) and co-hosting the day. The theme of the workshop and various issues in the MPEG-4 standard such as FGS framework are perfect examples of embedded systems and therefore the co-operation was smoothly established. The Chapter also acknowledges the co-sponsoring of CMG Eindhoven.

These proceedings contain a mixture of slide copies and recent papers addressing the themes of the individual lectures (as well as a paper from E.G.T. Jaspers about a prospective contribution to the preliminary program). This mixed approach was chosen to give maximum flexibility to the authors with minimum effort, thereby allowing the input of the latest material. Material from the workshop can also be found on the EESI web site (www.eesi.tue.nl).

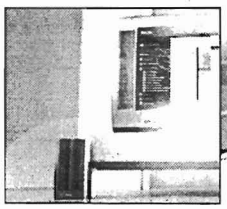
Peter H.N. de With

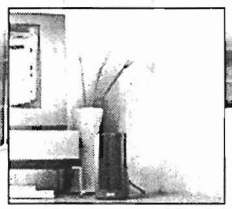
Board member IEEE Benelux Chapter on Consumer Electronics,
Eindhoven Embedded Systems Institute (EESI),
University of Technology Eindhoven, The Netherlands.

MPEG-4 Streaming
Developments and overview

Jan van der Meer
Philips Digital Networks
Building WDB-106
jan.vandermeer@philips.com

Presentation to IEEE Workshop on Embedded Video Streaming
Eindhoven, December 20, 2001



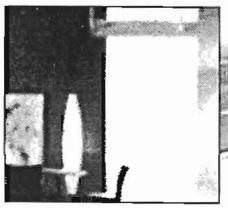


Let's make things better

PHILIPS

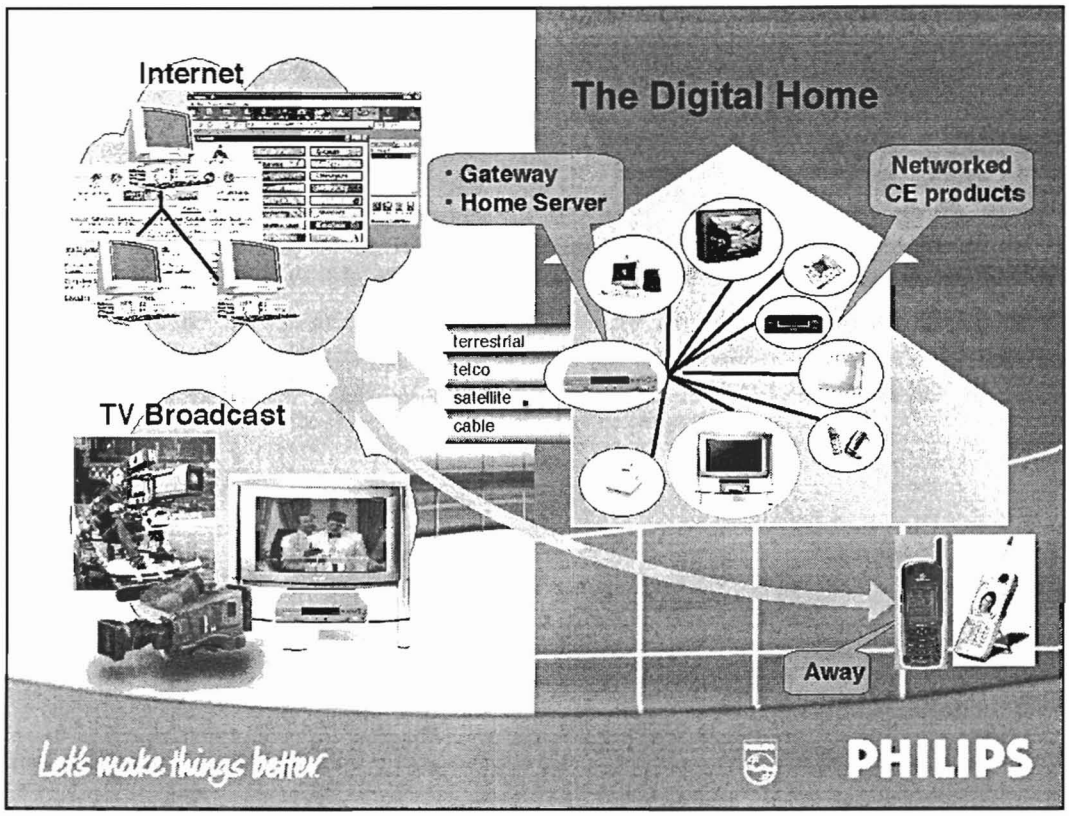
Outline

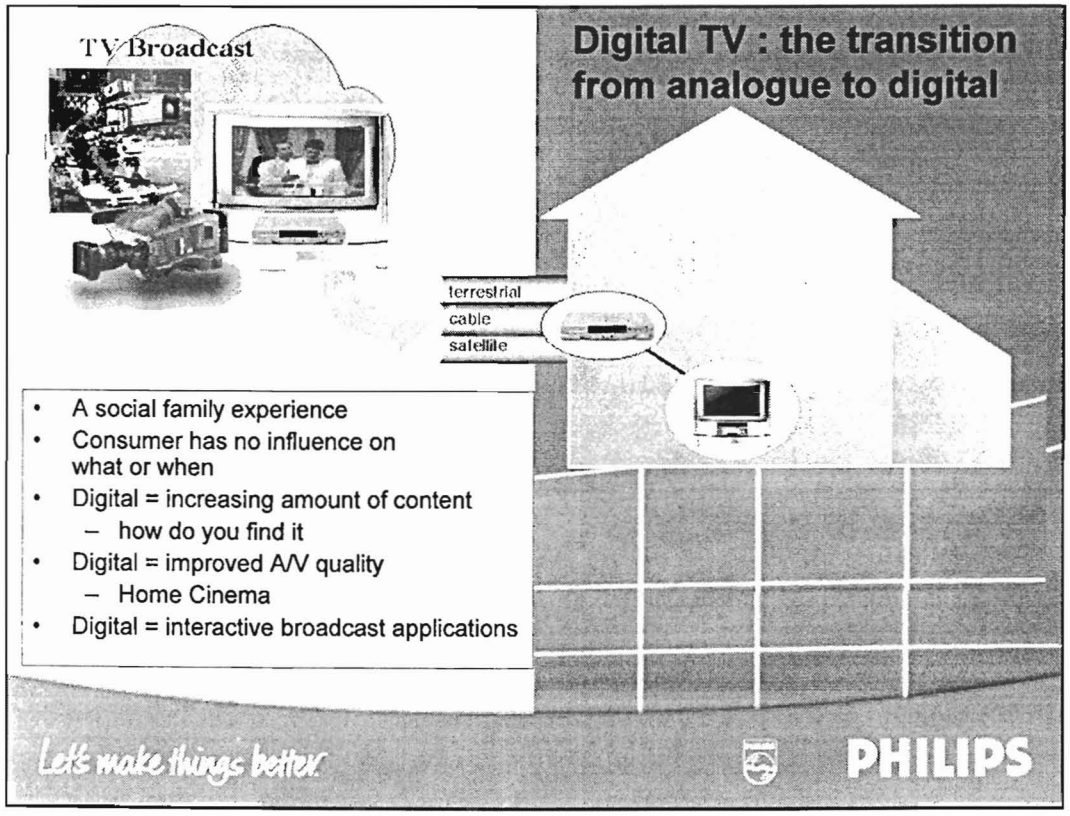
- The Streaming Environment
  - The Digital Home
  - Internet
  - Broadcast
  - Mobile
- MPEG-4 Streaming
  - Technology
  - Applications
- Standardization
- Conclusions



Let's make things better

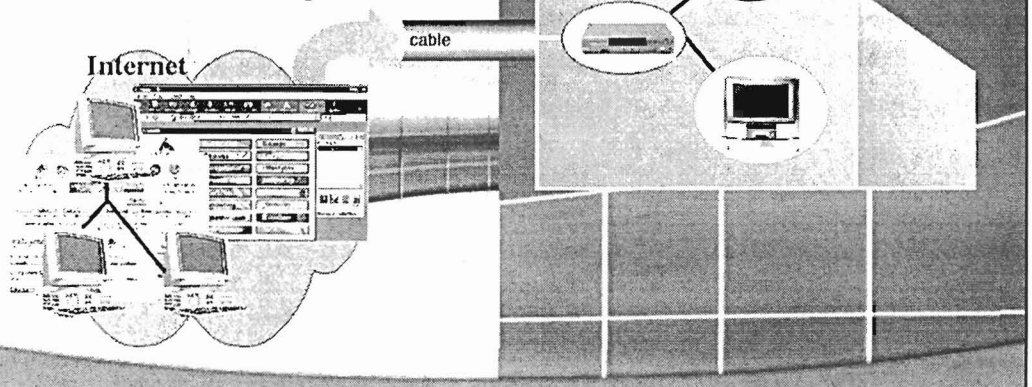
PHILIPS


The Digital Home
Internet
• Gateway
• Home Server
Networked CE products
terrestrial
telco
satellite
cable
TV Broadcast
Away
Let's make things better
PHILIPS

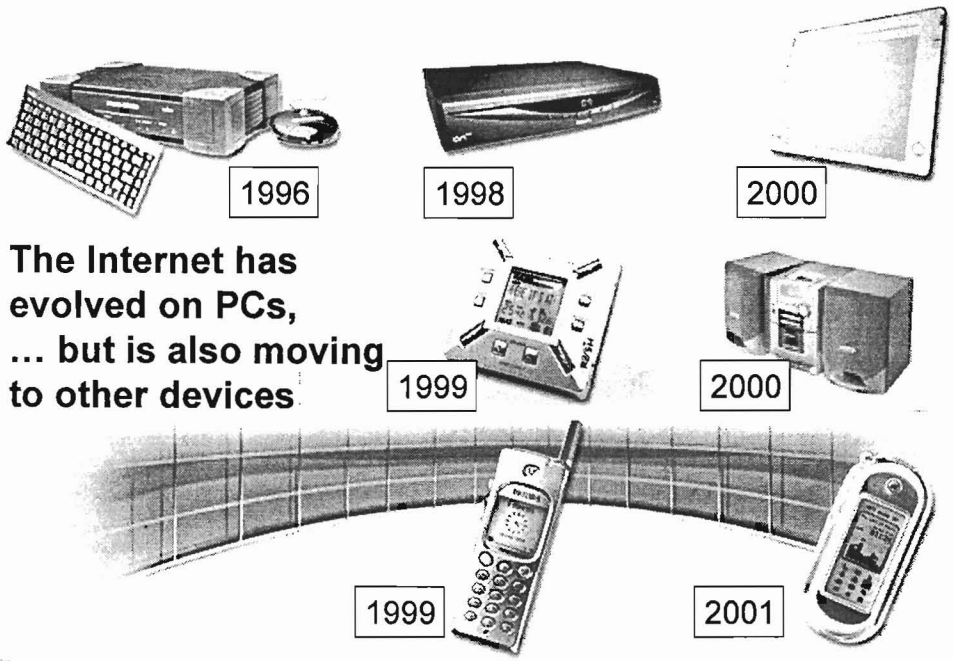

Digital TV : the transition from analogue to digital
TV Broadcast
terrestrial
cable
satellite
• A social family experience
• Consumer has no influence on what or when
• Digital = increasing amount of content
– how do you find it
• Digital = improved A/V quality
– Home Cinema
• Digital = interactive broadcast applications
Let's make things better
PHILIPS

The Internet

- an unlimited amount of information
- consumer determines what and when
- an individual experience
- search engines / portals help you
- gathering information, ordering books, emails
- Speech, audio, video
  - download, streaming



Let's make things better PHILIPS



The Internet has evolved on PCs, ... but is also moving to other devices

Let's make things better PHILIPS

Content constraints for non-PC devices

- Different User Interface
  - smaller screen format; sometimes interlaced
  - different screen distances
  - sometimes no key-board/mouse
- No single player download mechanism
- Less memory / hard-disk capacity

➡ Evolution of device specific content domains
  - PC / Digital Television / Mobile

Let's make things better

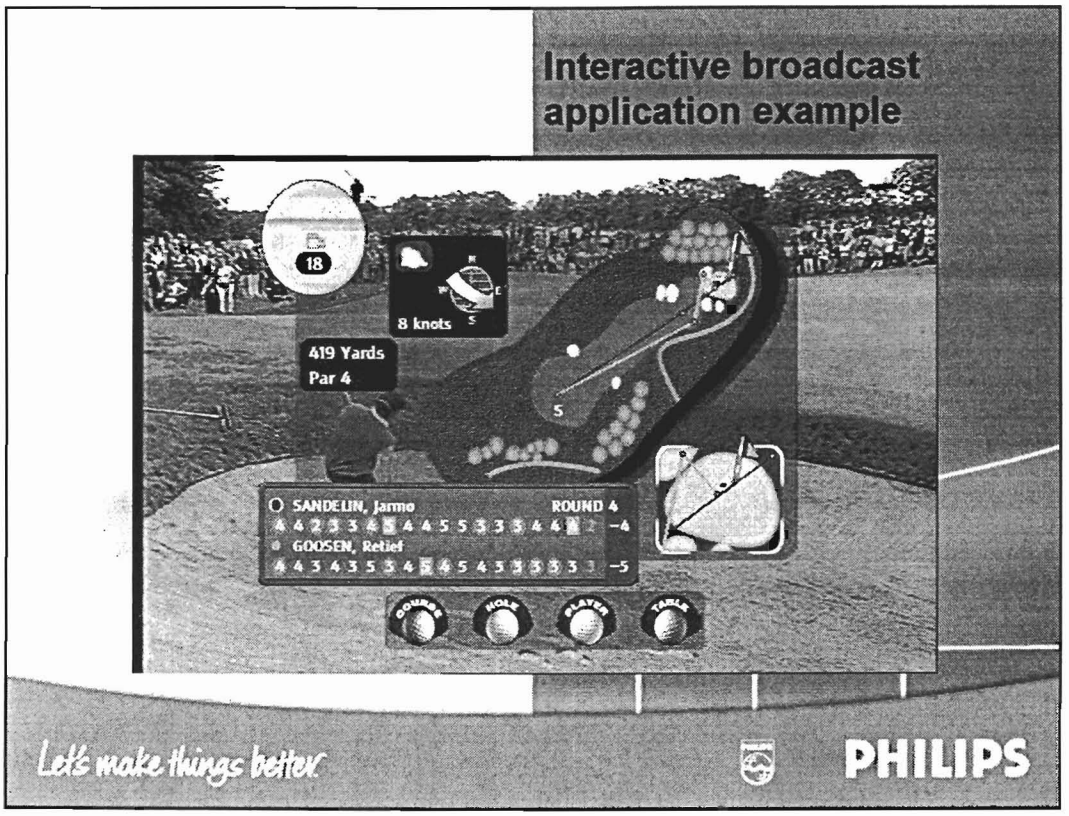
PHILIPS

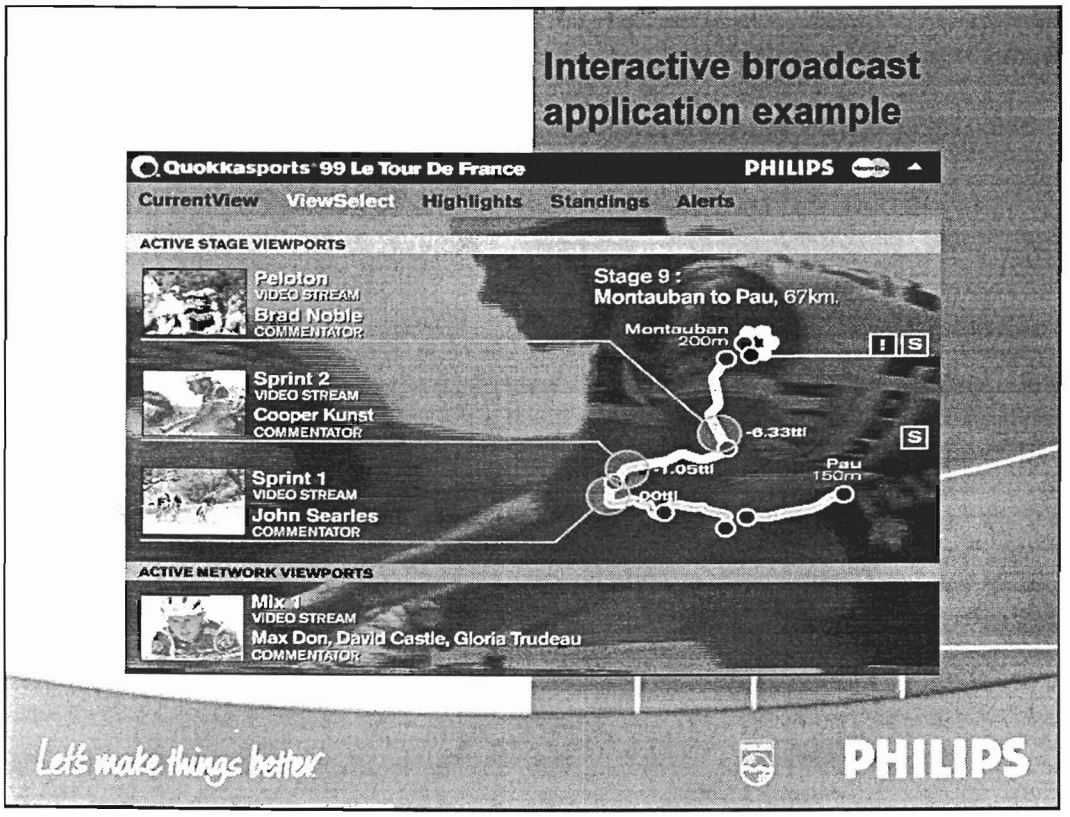
Digital TV : Interactive Broadcast Applications

- Application delivered with A/V broadcast data
  - "teletext" improved with graphics, video, audio, links
- Optional return channel
- (Complementary) delivery over IP
- Application formats
  - Proprietary : Canal plus / OpenTV / Liberate
  - Evolving open standard: MHP (Europe, USA)

Let's make things better

PHILIPS


Interactive broadcast application example
18
8 knots
419 Yards
Par 4
SANDELIN, Jarmo
ROUND 4
GOOSEN, Retief
COURSE
HOLE
PLAYER
TABLE
Let's make things better
PHILIPS


Interactive broadcast application example
Quokkasports 99 Le Tour De France
PHILIPS
CurrentView
ViewSelect
Highlights
Standings
Alerts
ACTIVE STAGE VIEWPORTS
Peloton
VIDEO STREAM
Brad Noble
COMMENTATOR
Sprint 2
VIDEO STREAM
Cooper Kunst
COMMENTATOR
Sprint 1
VIDEO STREAM
John Searles
COMMENTATOR
Stage 9 :
Montauban to Pau, 67km.
Montauban
200m
-6.33ttl
-1.05ttl
Pau
150m
ACTIVE NETWORK VIEWPORTS
Mix 1
VIDEO STREAM
Max Don, David Castle, Gloria Trudeau
COMMENTATOR
Let's make things better
PHILIPS

Mobile

- Speech
- Messaging (SMS)
- Informational services (WAP)
- Multimedia messaging (MMS)
- Packet switched streaming (PSS)
  - multimedia slide shows
    - speech, graphics, stills, audio, video, …
    - download and streaming

Let's make things better

PHILIPS

Broadcast and Mobile

- A variety of application formats
  - proprietary formats
  - evolving open standards
    - 3GPP: MMS, PSS
    - DVB: MHP
- Delivery over IP important focus
  - 3GPP
  - DVB services over IP

Let's make things better

PHILIPS

MPEG-4 Streaming

MPEG-4 Standard
- no replacement of MPEG-1/2 (at least initially ...)
- instead : New features
  - very efficient at low bitrates
  - object oriented
  - sophisticated multimedia applications
- organized in many parts
- abstract standard : no transport definition
  - transport over MPEG-2 (AMD 7 to MPEG-2 Systems)
  - transport over IP (joint effort between IETF and MPEG)

Let's make things better. PHILIPS

MPEG-4 Features (1)

MPEG-4 visual
- natural textures, still images, video
  - textures and still images wavelets, video DCT based
- synthetic objects
  - face and body animation at very low bitrates
- arbitrary shapes
- transparency plane, in addition to YUV planes
- high coding efficiency at low bitrates
  - obvious choice for streaming over Internet
- scalability
  - spatial, temporal, FGS

Let's make things better. PHILIPS

MPEG-4 Features (2)

MPEG-4 Audio
- a range of codecs for speech and music
- high level of scalability
- seamless adaptation to varying bandwidth

MPEG-4 Graphics
- VRML based
- binary format for transport
- streaming of animations at a few kbit/sec

Let's make things better PHILIPS

MPEG-4 Features (3)

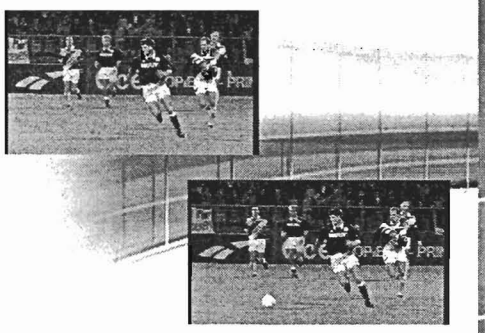
MPEG-4 Scenes
- MPEG-4 is about Objects !
- Scenes define how to compose objects
- Dynamic composition
  - composition may vary in time and space
- Suitable for Personalized Services
  - user specific composition
- Efficient transport format

Let's make things better PHILIPS

Example Application (1)

MPEG-4 object oriented technology

- Audience attractor for Pay TV
  - background
    - video : soccer stadium
    - audio : ambient sound in stadium
  - foreground
    - video : the football
    - audio : commentary voice



Let's make things better

PHILIPS

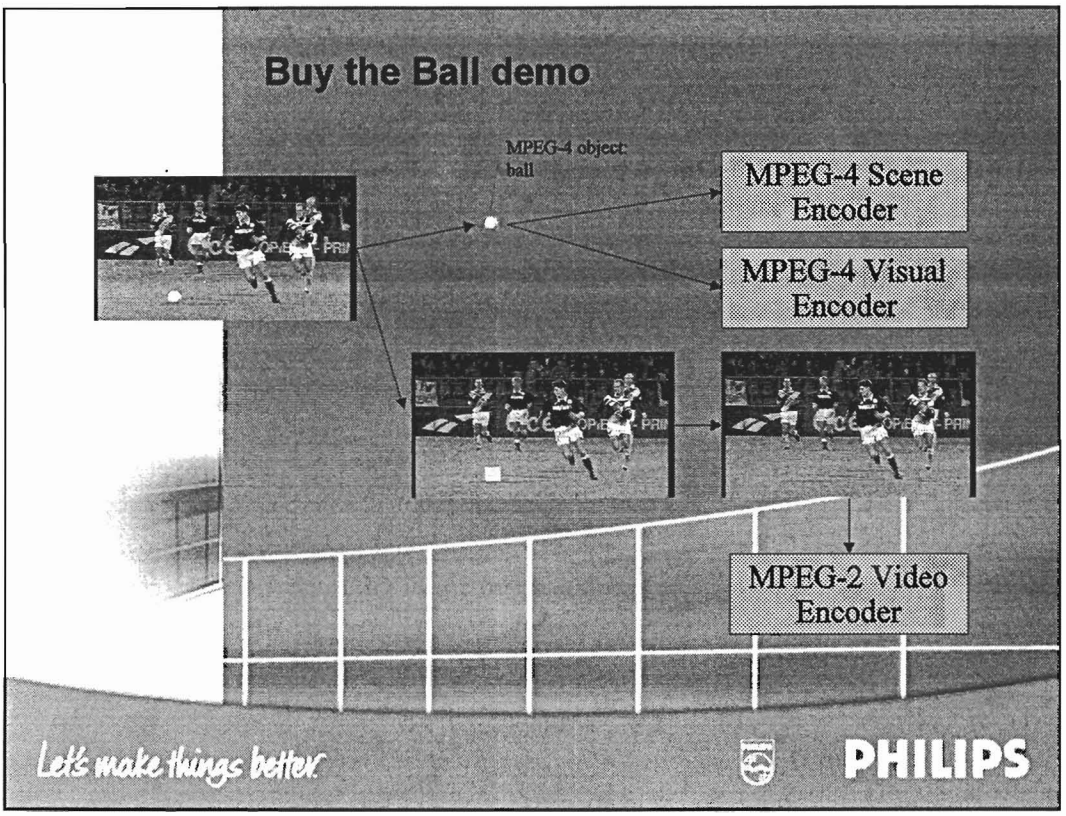
Buy the Ball

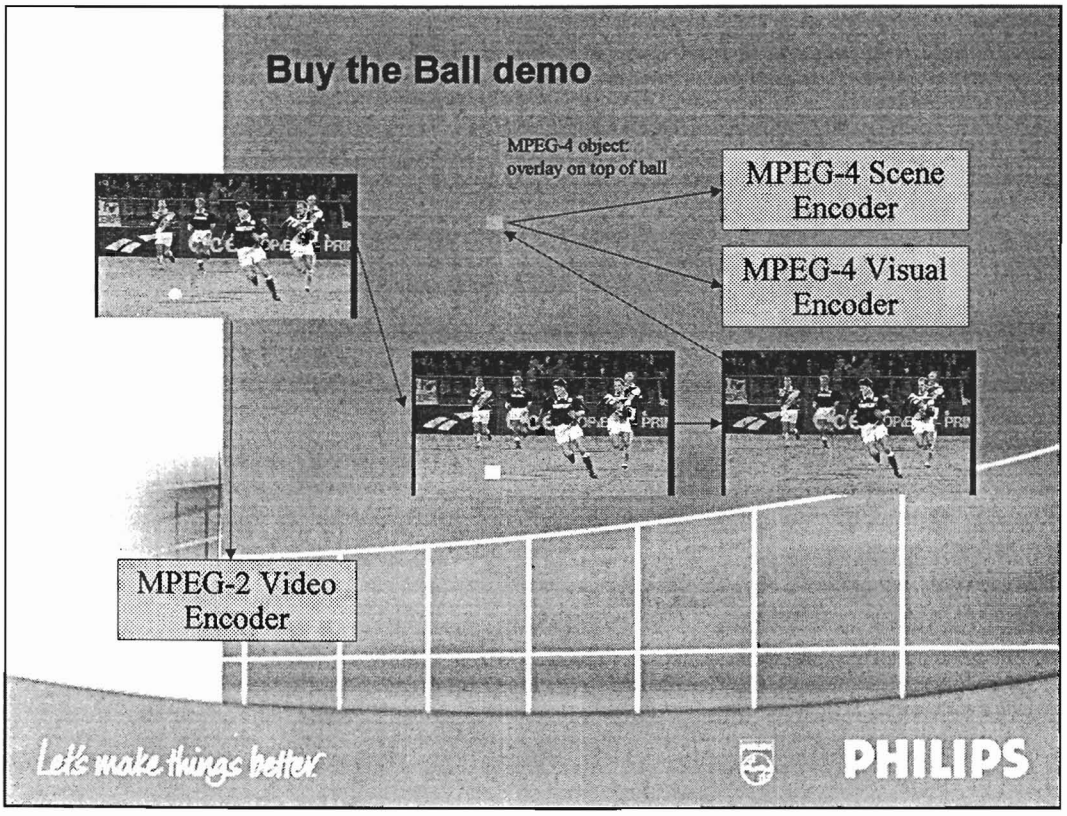
3 Euro only for entire match

Buy the Ball demo

Let's make things better

PHILIPS

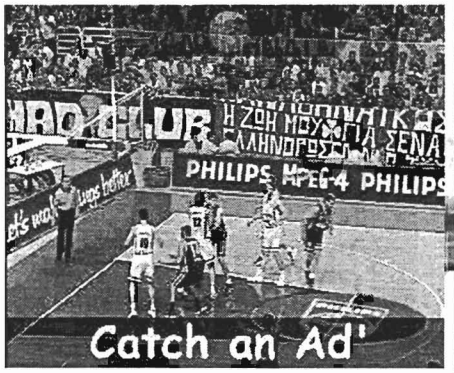

Buy the Ball demo
MPEG-4 object:
ball
MPEG-4 Scene Encoder
MPEG-4 Visual Encoder
MPEG-2 Video Encoder
Let's make things better
PHILIPS


Buy the Ball demo
MPEG-4 object:
overlay on top of ball
MPEG-4 Scene Encoder
MPEG-4 Visual Encoder
MPEG-2 Video Encoder
Let's make things better
PHILIPS

Example application (2)

MPEG-4 object oriented technology

- Target advertising

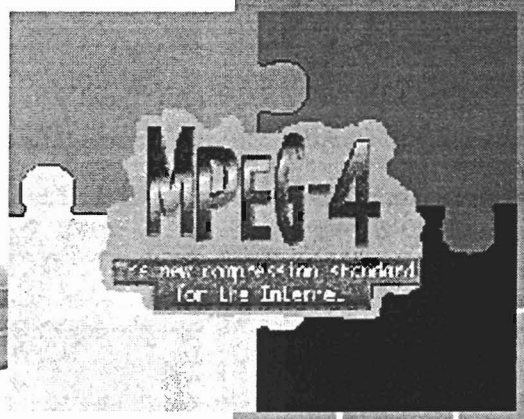


Let's make things better

PHILIPS

Example application (3)

Arbitrary shapes



Let's make things better

PHILIPS

Example application (4)

Streamed text end graphics

Let's make things better

PHILIPS

Potential MPEG-4 applications

- Streaming over IP
- Enhanced broadcast
- Storage in Set Top Boxes
- Personalized services
- Scalable services
- Combining all of the above
  - single application format
  - (very) rich multimedia applications
- But .... Already many existing formats
  - HTML, SMIL, MHP, etc.
  - Business Model ?

Let's make things better

PHILIPS

Evolving standardization efforts on MPEG-4

Streaming and download only

- 3GPP
  - MPEG-4 video
  - MPEG-4 AAC
  - MPEG-4 BIFS for subtitling (?)
  - No scalability support
    - limited number of terminals per transmitter
    - point to point connection to terminal
    - multicast service "transcoded" at 3GPP gateway
    - additional costs in terminal not justified

Let's make things better PHILIPS

Evolving standardization efforts on MPEG-4

- ISMA (Cisco, Sun, Apple, IBM, Kasenna, Philips, ...)
  - MPEG-4 video
  - MPEG-4 CELP and AAC
  - Complete End-to-End protocol stack
- Broadcast
  - DVB
    - QCIF: 3GPP
    - CIF, 601, HDTV: ISMA
  - wide adoption of DVB solution expected

Let's make things better PHILIPS

Standardization progress in MPEG

- H.26L
  - joint video team ITU / MPEG
    - Part 10 of MPEG-4
  - very efficient video compression (50 % efficiency gain)
  - not compatible with current MPEG-4 video
  - ongoing complexity analysis
  - potential replacement of existing MPEG-2/4 video
    - in some application area's
- Likely to create high momentum
  - streaming applications

Let's make things better

PHILIPS

Conclusions

- MPEG-4 is very powerful standard
  - streaming
  - multimedia applications
    - many existing formats
- MPEG-4 will probably be mainly about streaming
  - interoperable streaming services
  - use possible from any application format
    - e.g. URL
  - "gateway" from IT industry to CE devices

Let's make things better

PHILIPS

Thank You

References

- MPEG : http://www.cselt.it/mpeg/
- M4IF : http://www.m4if.org/
- MP4Net : www.mpeg-4player.com/
- ISMA : http://www.isma.tv/
- 3GPP : http://www.3gpp.org/
- DVB : http://www.dvb.org/

Let's make things better.

PHILIPS

Domain-based multiple description coding of images and video

John W. Woods and Ivan V. Bajic

Center for Next Generation Video

Rensselaer Polytechnic Institute, Troy, NY

woods@ecse.rpi.edu, ivanb@cipr.rpi.edu

Overview

- Motivation for multiple description (MD) coding and previous work
- Domain-based multiple descriptions
- Applications: dispersive packetization of images and video
- Concatenated multiple descriptions
- Conclusions

1

Motivation for MD coding

- Multiple descriptions are a form of joint source-channel coding, well suited for real-time data transmission over lossy packet networks
- MDs can be made "equally important" - matched to the nature of Internet traffic, which is not prioritized
- They offer low sensitivity to assumptions about network state
- Graceful degradation; any subset of descriptions can produce a useful approximation to the source message - fidelity improves as the number of received descriptions increases

2

Previous work on MD coding

- MD **scalar quantization** (Vaishampayan *IEEE Trans. IT*, 1993; Servetto et al. *IEEE Trans. IP*, 2000)

  Overlapping quantizer cells encoded independently

- MD **correlating transforms** (Wang, Orchard, and Reibman *IEEE MMSP workshop*, 1997)

  Introduce controlled correlation after decorrelating transform

- MD **tight frame expansion** (Goyal et al. *ICIP*, 1998)

  "Increase the number of basis vectors"

- MD **forward error correction** (Puri and Ramchandran *Asilomar* 1999; Mohr, Riskin, and Lander *ICIP*, 1999)

  Rate-distortion optimized unequal error protection (UEP) for embedded bitstreams

3

Domain-based multiple descriptions

Advantages

- No FEC, hence bandwidth efficient
- No assumptions about channel state
- Easy error concealment
- Graceful degradation

Disadvantages

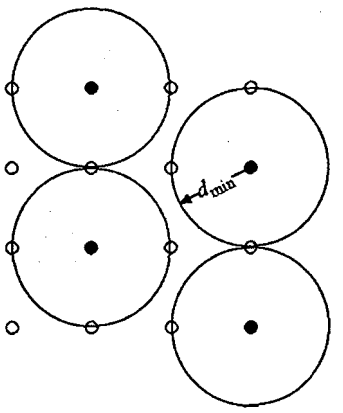
- Requires modification of the codec
- Error concealment by interpolation, so only approximate recovery of lost information is possible

4

Domain-based multiple descriptions

- Each description is a subsampled version of the signal
- Sampling by domain (lattice) partitioning

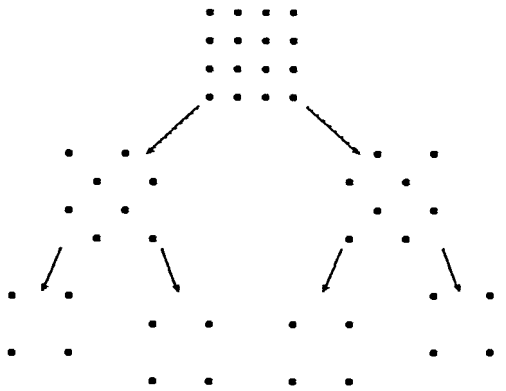
- To ensure having as many neighbors for error concealment as possible, maximize the minimal distance between samples within each partition
- Equivalent to sphere packing on a lattice



5

Design of domain-based MDs

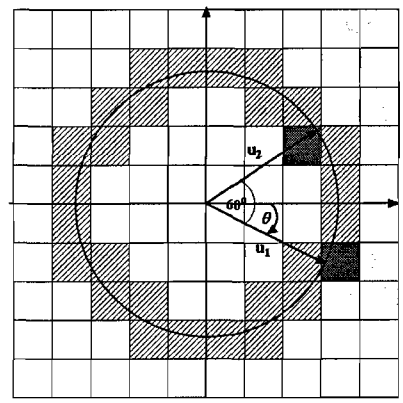
- Possible solution: lattice partitioning by squaring construction (for TCM code design)
- But it works only for 2^n partitions



6

Design of domain-based MDs (cont.)

- Better solution: look for sublattices of Z^2 which are approximately hexagonal
- Works for any number of partitions, at least as good as squaring construction



7

Domain-based multiple descriptions for images

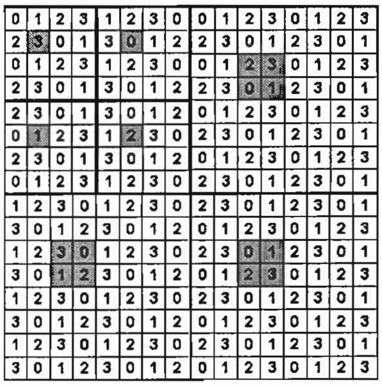
- Each block corresponds to a pixel
- Number in the block is the number of the packet where that pixel should be stored.
- Dispersion – like numbers are far from each other in the matrix
- Example: dividing an 8×8 image into 8 partitions

0	1	4	5	0	1	4	5
3	2	7	6	3	2	7	6
4	5	0	1	4	5	0	1
7	6	3	2	7	6	3	2
0	1	4	5	0	1	4	5
3	2	7	6	3	2	7	6
4	5	0	1	4	5	0	1
7	6	3	2	7	6	3	2

8

Domain-based multiple descriptions for subbands

- Create partition for one subband, then extend by modulo shifting
- Example: partitioning an 8×8 image into 4 partitions
- Observe: coefficient "trees" are dispersed across packets



0	1	2	3	1	2	3	0	0	1	2	3	0	1	2	3
2	3	0	1	3	0	1	2	2	3	0	1	2	3	0	1
0	1	2	3	1	2	3	0	0	1	2	3	0	1	2	3
2	3	0	1	3	0	1	2	2	3	0	1	2	3	0	1
2	3	0	1	3	0	1	2	0	1	2	3	0	1	2	3
0	1	2	3	1	2	3	0	2	3	0	1	2	3	0	1
2	3	0	1	3	0	1	2	0	1	2	3	0	1	2	3
0	1	2	3	1	2	3	0	2	3	0	1	2	3	0	1
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3

Illustration of error concealment

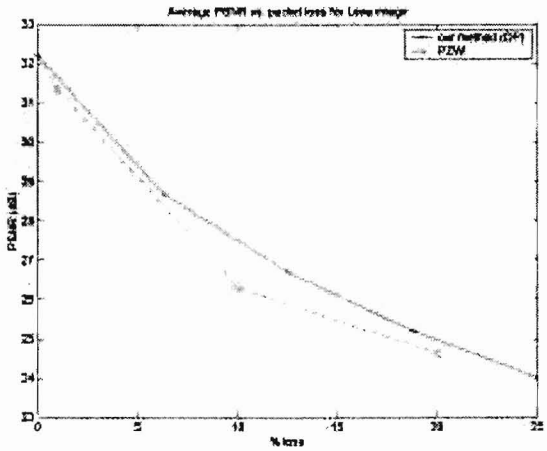
- 512×512 Lena, encoded at 0.21 bpp, packetized into 16 packets
- 12.5 % packet loss, bilinear error concealment

Lena at 12.5% packet loss
no concealment, PSNR = 16.5dB

Lena at 12.5% packet loss
bilinear concealment, PSNR = 26.9dB

Comparison with PZW on Lena

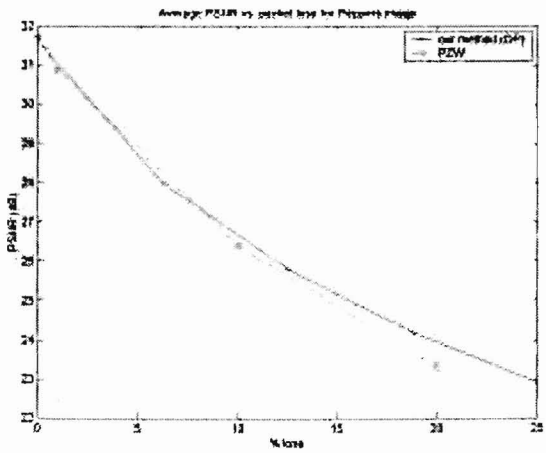
- 512×512 Lena image, encoded at 0.21 bpp



11

Comparison with PZW on Peppers

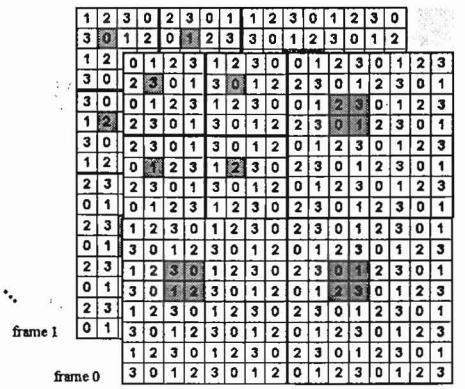
- 512×512 Peppers image, encoded at 0.21 bpp



12

Domain-based multiple descriptions for intraframe video

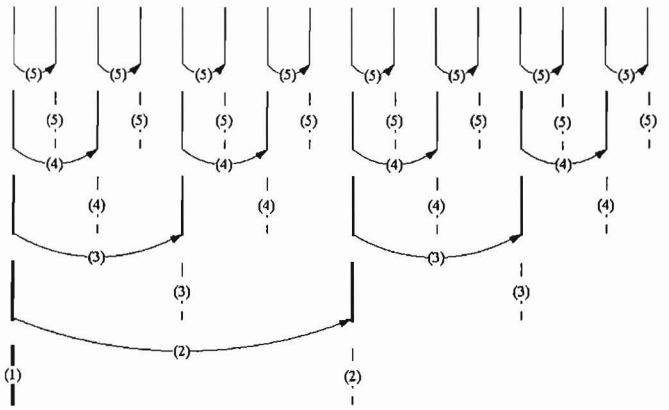
- Create partition for one frame, then extend by modulo shifting
- Example: partitioning two 8×8 frames into 4 partitions



13

Domain-based multiple descriptions for motion-compensated video

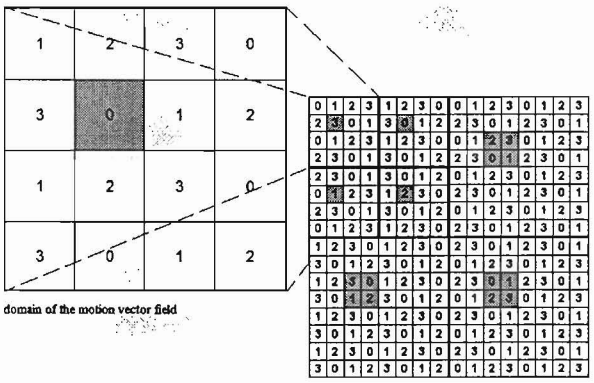
- GOP structure in the motion compensated 3-D subband video coder has 5 data layers



14

Domain-based multiple descriptions for motion-compensated video (cont.)

- In each layer, extend the partitions from high frequency frames to motion vector fields by modulo shifting



15

Experiment

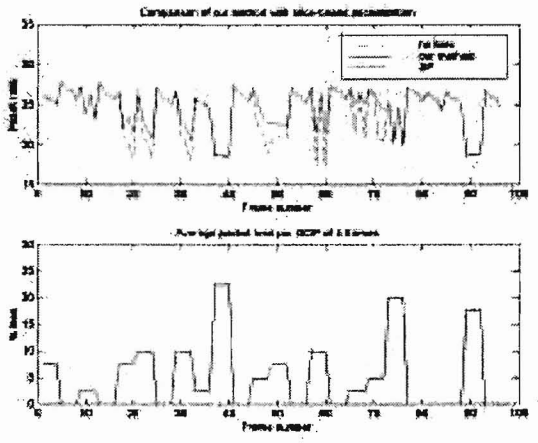
- Football sequence (SIF resolution, 30fps)
- Encoded at 973 kbps, GOP of 4 frames, average PSNR of 26.3 dB
- Simulated transmission of 96 video frames in two scenarios:
  - Lower loss (5.4%)
  - Higher loss (10.4%)
- Compared against slice-based packetization (H.323)
- In both cases, missing data (subband samples or motion vectors) estimated as the median of however many of the 8 nearest neighbors are available

Loss	PSNR for SP	PSNR for DP	Gain (DP over SP)
5.4%	24.0 dB	24.6 dB	+0.6 dB
10.4%	21.8 dB	23.2 dB	+1.4 dB

16

Frame-by-frame PSNR results

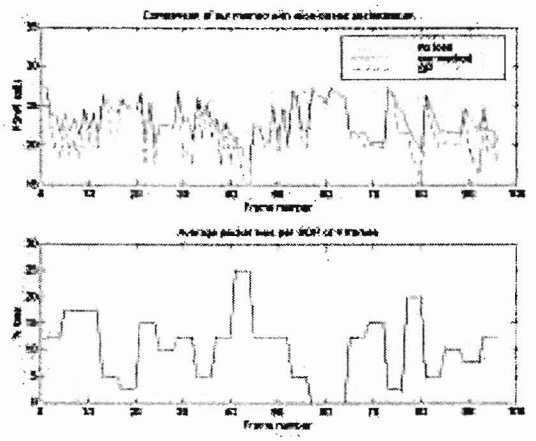
- 5.4% loss



17

Frame-by-frame PSNR results

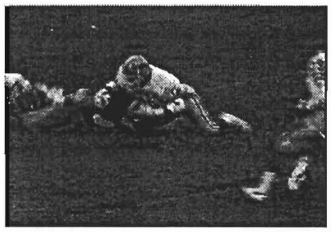
- 10.4% loss



18

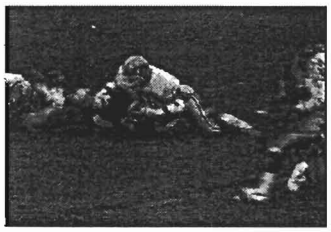
Experimental results, playout @ 30 fps

SP, 5.4 % loss

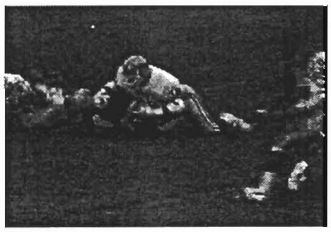


DP, 5.4 % loss

SP, 10.4 % loss



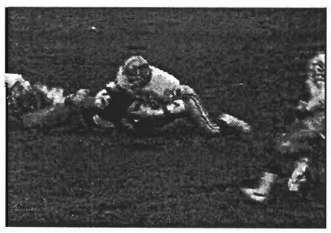
DP, 10.4 % loss



19

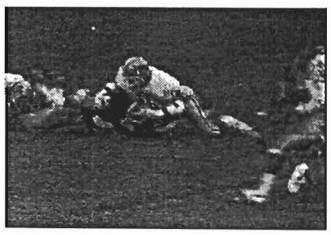
Experimental results, playout @ 10 fps

SP, 5.4 % loss

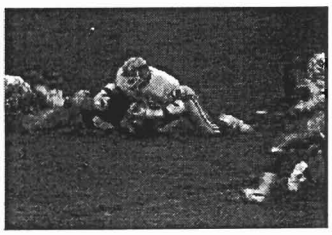


DP, 5.4 % loss

SP, 10.4 % loss



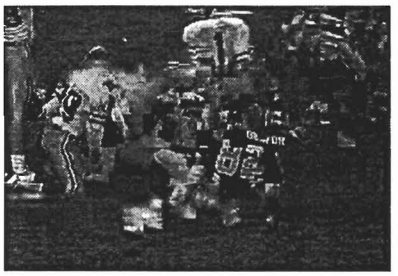
DP, 10.4 % loss



20

Dispersive packetization benefits error concealment

- Frame 80, 20 % loss



Slice-based packetization

PSNR = 15.8 dB

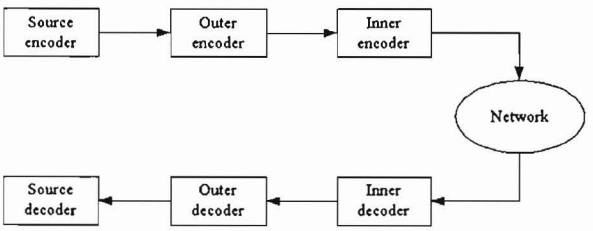
Dispersive packetization

PSNR = 19.5 dB

21

Next step: combination with FEC

- Use domain-based MD and FEC-based MD in concatenated fashion, similar to concatenated coding

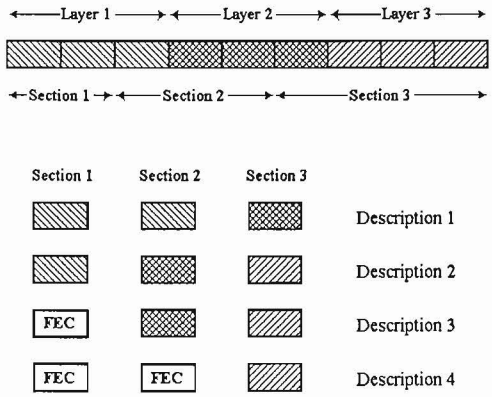


- Domain-based MD acts as the outer code, FEC acts as the inner code
- Error concealment attempts to correct errors which cannot be corrected by FEC

22

Creating concatenated multiple descriptions

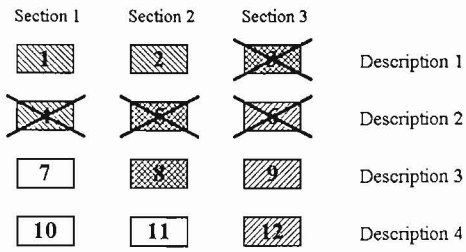
- Dispersively packetized frame-rate scalable video bit stream divided into sections
- Each section assigned appropriate amount of FEC (RS codes)



23

Example of error resilience of concatenated multiple descriptions

- Packets 9 and 12 are decodable (VLC does not cross packer boundaries)
- Packets 4 and 5 recovered by FEC
- Packet 3 recovered by error concealment in layer 2
- Packet 6 recovered by error concealment in layer 3



24

Design of concatenated multiple descriptions

- Let D_i be reduction in distortion carried by packet i
- Want to create N descriptions
- Divide packet stream into M sections
- Section j has $N_j - N_{j-1}$ data packets and $K_j = N - (N_j - N_{j-1})$ FEC packets
- Let p be the packet loss probability
- For $j = 1, 2, \ldots, M$, we have

$$q_j = \sum_{k=0}^{K_j} \binom{N}{k} p^k (1-p)^{N-k}$$

Probability that section j can be recovered by FEC

$$D^{(j)} = \sum_{i=N_{j-1}+1}^{N_j} D_i$$

Reduction in distortion in section j

25

Design of concatenated multiple descriptions (cont.)

- Assume that in any given section, each packet carries approximately the same reduction in distortion
- Expected reduction in distortion at the receiver is

$$E[D] = \sum_{j=1}^{M} D^{(j)} \left(q_j + (1-p)(1-q_j) \right)$$

- The term in summation associated with q_j accounts for sections which are recovered by FEC
- The term associated with $(1-p)(1-q_j)$ accounts for the data packets (information symbols of RS codewords) which are received in the sections which cannot be recovered by FEC
- Error concealment not explicitly included in this model

26

Design of concatenated multiple descriptions (cont.)

- Problem: given the number of descriptions N, maximal allowable rate R_{max}, and the packet loss probability p, find M and $N_1, N_2, \ldots, N_M$, so as to

maximize $E[D] = \sum_{j=1}^{M} D^{(j)}\left(q_j + (1-p)(1-q_j)\right)$

subject to $N_0 = 1$, and for $j = 1, 2, \ldots, M$

$$K_j = N - (N_j - N_{j-1})$$

$$q_j = \sum_{k=0}^{K_j} \binom{N}{k} p^k (1-p)^{N-k}$$

$$D^{(j)} = \sum_{i=N_{j-1}+1}^{N_j} D_i$$

N_j integer

total rate $\leq R_{max}$

27

Design of concatenated multiple descriptions (cont.)

- Simplification: assume the number of FEC packets decreases by 1 in each subsequent section (similar to Puri & Ramchandran's scheme)
- Simplified problem involves only two integer variables: M and N_1
- Can be solved by exhaustive search in $O(N^2)$ steps

28

Experiment

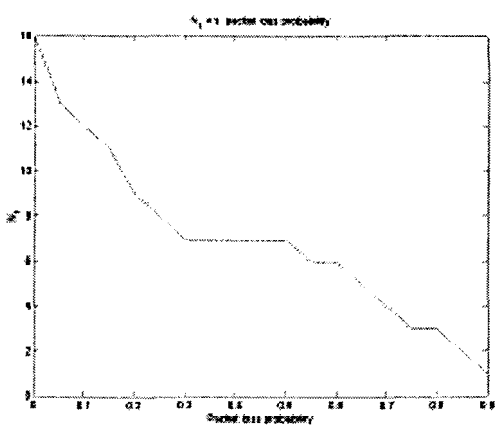
- Football sequence (SIF resolution, 30fps), with a GOP of 4 frames, maximal allowable rate R_{max} = 1.2 Mbps
- N = 16 descriptions for each GOP
- Operational distortion-rate curve was obtained by encoding the video at 1.2 Mbps, and then decoding layer by layer, with the assumption that all the packets in the same layer carry the same reduction in distortion (due to dispersive packetization)
- We solved the problem for various packet loss probabilities in the range $0 \leq p \leq 0.9$
- Optimization procedure gave M = 3 sections in all cases

  $\Rightarrow$ 3×16 = 48 packets per section, about 420 bytes each
- N_1 decreases as p increases

29

Experimental results

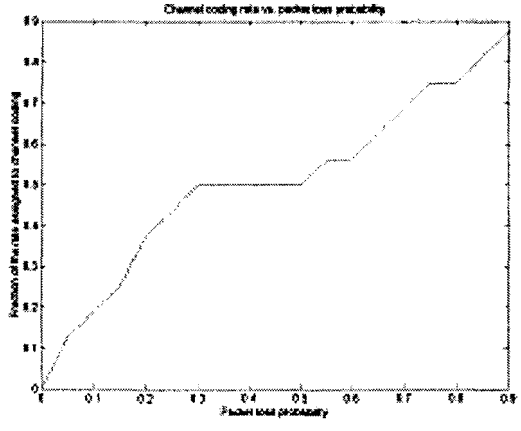
- Plot: optimal value of N_1 as a function of packet loss probability p
- Note: lower N_1 means more FEC



30

Experimental results (cont.)

- Plot: percentage of the rate assigned to channel coding as a function of packet loss probability



31

Experimental results (cont.)

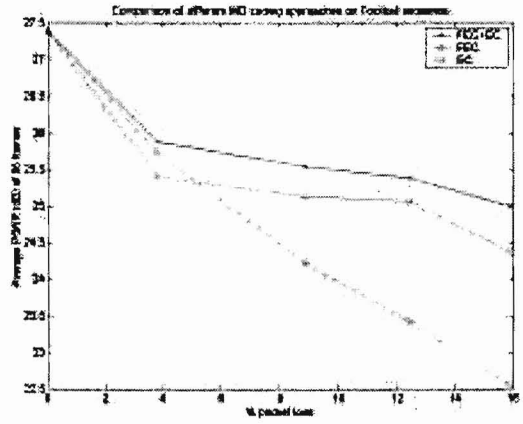
- Transmission was simulated for four different packet loss realiztions
- Table: Average PSNR results for 96 frames
- The system benefits from both FEC and error concealment

Loss	PSNR for EC	PSNR for FEC	PSNR for FEC+EC
3.8%	25.7 dB	25.4 dB	25.9 dB
8.9%	24.2 dB	25.1 dB	25.5 dB
12.1%	23.4 dB	25.1 dB	25.4 dB
15.9%	22.6 dB	24.4 dB	25.0 dB

32

Experimental results (cont.)

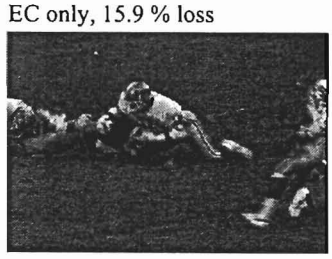
- Plot: average PSNR vs. packet loss
- Concatenated system better than either of the two components



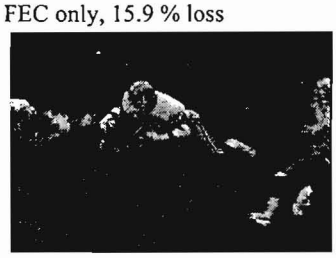
33

Experimental results, playout @ 30 fps

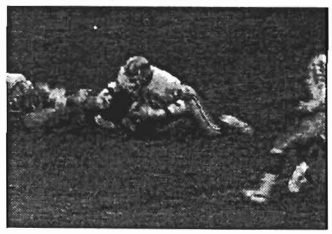
EC only, 15.9 % loss



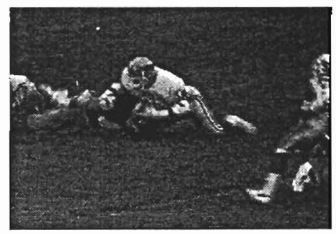
FEC only, 15.9 % loss



FEC + EC, 15.9 % loss



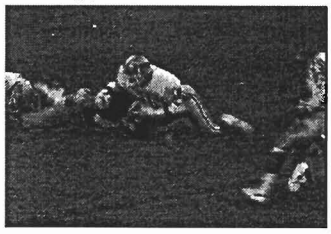
No loss



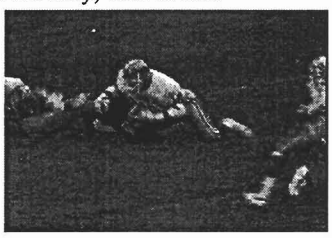
34

Experimental results, playout @ 10 fps

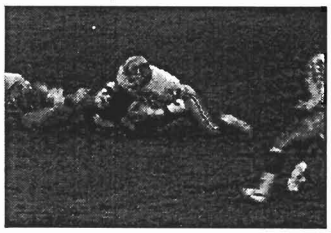
EC only, 15.9 % loss



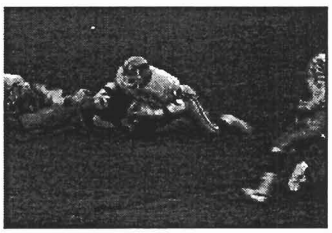
FEC only, 15.9 % loss



FEC + EC, 15.9 % loss



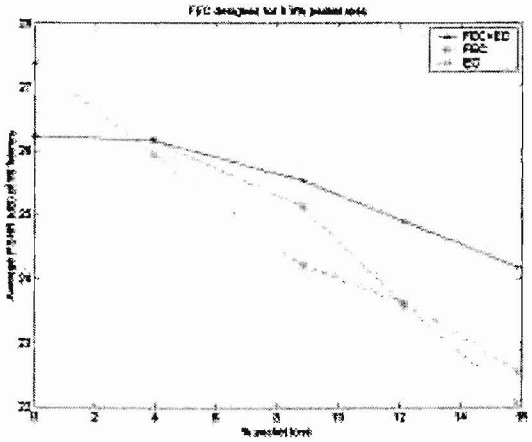
No loss



35

What if packet loss probability estimated incorrectly?

- Experiment: FEC designed for 8.9% packet loss, but actual packet loss varies ⇒ error concealment helps



36

Conclusions

- Domain-based multiple descriptions offer bandwidth-efficient way for image/video transmission at low loss rates (~5%)
- Can provide 0.5 – 1 dB advantage over other packetization methods using equivalent error concealment algorithms
- Can be integrated with other forms of MD codes to handle higher packet loss
- Dispersive packetization and error concealment help reduce the effects of imperfect channel estimation used for FEC design

37

Ideas for future work

- More advanced error concealment algorithms
- Extension to embedded coders (e.g. EZBC) to provide higher degree of scalability
- Integration with transport protocols

38

Domain-based Multiple Description Coding of Images and Video

Ivan V. Bajić and John W. Woods

Center for Image Processing Research and Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY 12180-3590.

This work was supported in part by the ARO grant DAAD 19-00-1-0559.

Abstract

In this paper we present a method of creating domain-based multiple descriptions of images and video. Descriptions are created by partitioning the domain of the signal into sets whose points are maximally separated from each other. This property enables simple error concealment methods to produce good estimates of lost signal samples. We present the approach in the context of Internet transmission of subband/wavelet-coded images and scalable motion compensated 3-D subband/wavelet-coded video, but applications are not limited to these scenarios.

Keywords

Multiple descriptions, robust video coding, error concealment, set partritioning.

I. Introduction

Internet video streaming is becoming an increasingly important way of world-wide distribution of information. Delivery of high quality video over a wide area network with large number of users poses great challenges for the video communication system design. To ensure network stability, video servers must share the network bandwidth fairly with other users. Hence, video needs to be encoded within the rate limit set by the network occupancy. On the other hand, in the face of variable network conditions and occurrence of packet losses and random delays, the compressed video bitstream needs to be sufficiently robust and flexible. This creates the need for new approaches to video coding, which combine high compression efficiency and robustness.

Multiple descriptions have recently attracted a lot of attention as a way of encoding and communicating visual information over lossy packet networks. In such a scenario, a multiple description (MD) coder separates the source message into several bit-streams (descriptions) which are then separately transmitted over the network. Each description is individually decodable so that loss of some of the descriptions will not jeopardize decoding of those descriptions that are correctly received. Fidelity of the received message improves as the number of received descriptions increases. Another desirable property is that descriptions can be made "equally important" or "balanced." This is important in the context of Internet transmission, where none of the packets receive preferential treatment.

Among the popular MD coding schemes is multiple description scalar quantization (MDSQ) [1], whose application to subband/wavelet image coding was reported in [2].

A typical MDSQ system is based on a set of so-called side quantizers which are used to produce different descriptions. At the receiver, the signal is de-quantized through the quantizer whose bins are intersections of quantization bins of all received descriptions. Redundancy is introduced due to the fact that side quantizers overlap, but their outputs are encoded independently. Other MD schemes introduce redundancy in different ways: through correlating transforms [3], frame expansions [4], or FEC [5].

In this paper we consider domain-based multiple descriptions which are created by partitioning the domain of the signal. The method is targeted at signals defined on discrete domains (specifically integer lattices), such as digital images or video frames. Each description is a subsampled version of the signal. The method is bandwidth-efficient since no FEC is used and no extra redundancy is introduced. Descriptions are designed to enable easy recovery of lost samples using very simple error concealment techniques (e.g. bilinear or median interpolation), which are suitable for low-complexity receivers.

The paper is organized as follows. In Section 2 we introduce domain-based multiple descriptions and analyze some of their properties. The choice of optimization criteria used in the design is explained and solution to the optimization problem is discussed. In Section 3 we explain how the concept of domain-based multiple descriptions can be applied to robust image and video and transmission. Sections 4 and 5 contain results and conclusions, respectively.

II. Domain-based multiple descriptions

A. Preliminaries

Consider a signal $f : \mathcal{D} \rightarrow \mathcal{R}$ that we wish to transmit, with domain $\mathcal{D}$ and range $\mathcal{D}$. In the case of digital images, the domain is typically a subset of the $\mathbb{Z}^2$ lattice. For example, for an $N_1 \times N_2$ image, $\mathcal{D} = \{0, 1, ..., N_2 - 1\} \times \{0, 1, ..., N_1 - 1\}$. Suppose we wish to create P descriptions of the signal. This would amount to constructing P non-empty sets $S_0, S_1, ..., S_{P-1}$, such that

$$\bigcup_{i=0}^{P-1} S_i = \mathcal{D}.$$

Collection $C = \{S_0, S_1, ..., S_{P-1}\}$ is a *cover* of the domain $\mathcal{D}$. Constructing C which maximizes some suitably defined objective function is known as a *set covering* problem in

integer programming [6]. In the special case when S_i are disjoint, that is

$$S_i \cap S_j = \emptyset, i \neq j,$$

C defines a *partition* of the domain $\mathcal{D}$. Consequently, finding optimal C in this case is a *set partitioning* problem. Both set covering and set partitioning problems are NP-hard in general. Sets S_i are identified with descriptions of the signal, in the sense that signal samples at points in S_i form the i-th description. Formally, i-th description is (S_i, F_i), with

$$F_i = \{f(\mathbf{x}) : \mathbf{x} \in S_i\}, i = 0, 1, ..., F - 1.$$

In case of packet-based transmission, each description is a packet, or equivalently, each packet is considered to be a description.

If C is a cover but not a partition, then at least one signal sample appears in more than one description. This represents simple repetition coding in the signal domain and provides a method of adding extra protection to "important" signal samples. If some descriptions are corrupted or lost, copies of "important" samples may be available in other descriptions that are correctly received. The level of "importance" of any given sample can be used as an indication of the number of descriptions it should be included in. This is a way of achieving unequal error protection on a sample-by-sample basis.

On the other hand, if C is a partition of $\mathcal{D}$, then there is no added redundancy in representing the signal by its P descriptions. This does not mean that samples from lost or corrupted descriptions cannot be recovered. On the contrary, there is usually enough redundancy in the signal itself to estimate the data which is missing, provided partitions are carefully designed. The estimation process is commonly referred to as the (passive) error concealment. This case is particularly attractive as no extra redundancy is added. Here we focus exclusively on this case which, in a way, represents a worst-case scenario. Of course, by extending partitions to covers, one can achieve higher level of error resilience, at the expense of increased bit rate.

Consider the case when no error concealment is used. Suppose that a zero-mean signal has been quantized at the encoder to a distortion of $D_{\min}$ and suppose that we have created P descriptions of the signal, each encoded into a separate packet, so that i-th description

(packet) carries with it a reduction in distortion of D_i. Let D_R be the distortion at the receiver. Before receiving any of the packets, decoder approximates the source signal by a zero-signal and has a maximal distortion $D_R = D_{\max}$ equal to the signal variance. As the packets arrive at the decoder, the approximation of the source signal improves. If all packets arrive correctly, distortion at the decoder will be the same as the distortion at the output of the encoder, $D_R = D_{\min}$, and we have $\sum_{i=0}^{P-1} D_i = D_{\max} - D_{\min}$. Let X_i be the indicator variable for the reception of i-th packet, so that $X_i = 1$ if i-th packet is received, and $X_i = 0$ otherwise. Then $p_i = E[X_i]$ is the probability of reception of the i-th packet, and $1 - p_i$ is its loss probability. If no packets receive preferential treatment, as is the case in the Internet today, then the probability of reception is the same for all of them, so $p_i = p$ for all $i = 0, 1, ..., P - 1$. The expected distortion at the receiver is

$$E[D_R] = E\left[D_{\max} - \sum_{i=0}^{P-1} X_i D_i\right] = D_{\max} - p\sum_{i=0}^{P-1} D_i = pD_{\min} + (1-p)D_{\max}.$$

Hence, without error concealment, expected distortion at the receiver does not depend on the specific values of D_i. Since each description (packet) has the same probability of being lost, each signal sample has the same probability of being lost, regardless of the packet in which it is stored. When error concealment is used, the distortion will depend on the ability of decoder to estimate the missing samples from those that are available. Therefore, in this context, optimal descriptions are those which allow the best error concealment. Design of such descriptions is discussed next.

B. Optimal domain partitioning

In order to formulate the partitioning problem as an optimization problem, one must choose the objective function. Ideally, for a given number of partitions P (and given all other necessary parameters), we wish to come up with a partition $C = \{S_0, S_1, ..., S_{P-1}\}$ which will minimize the expected distortion under all possible transmission scenarios. This seems like an extremely difficult task. In a typical transform coding application, expected distortion at the receiver will depend, among other things, on the interplay between the type of signal, the type of transform applied to the signal, the type of entropy coder, and the error concealment algorithm applied at the decoder. Apart from presenting a modelling problem, any particular choice of the above mentioned components may have

its own (different) optimum. For this reason we will choose a much simpler objective function which is likely to give good results in all cases.

First, note that successful estimation of missing signal samples relies on the correlation (autocorrelation) of the signal. The correlation is typically a decreasing function of distance, so we expect that whatever the choice of the estimation method, we should get the best results if we estimate the missing sample from its immediate neighborhood. Second, the quality of the estimate should not decrease as we increase the size neighborhood. This is easy to see by considering two nested neighborhoods of a missing sample. Let $\mathbf{x}$ be the location of the missing sample, with $\mathbf{x} \in D_1 \subset D_2$, and let $\widehat{f_{D_i}}(\mathbf{x})$ denote the estimate of $f(\mathbf{x})$ from the samples in $D_i, i = 1, 2$. Then $\widehat{f_{D_2}}(\mathbf{x})$ can be at least as good an estimate of $f(\mathbf{x})$ as $\widehat{f_{D_1}}(\mathbf{x})$, since we can always choose not to use any samples in $D_2 \backslash D_1$, and thus obtain $\widehat{f_{D_2}}(\mathbf{x}) = \widehat{f_{D_1}}(\mathbf{x})$. Hence, we can expect that the distortion at the receiver will be a non-increasing function of the size of the available neighborhood of a missing signal sample, for typical estimation algorithms. Experimental evidence supports this observation [7].

Based on the analysis above, we choose the objective function to be the overall minimal intra-partition distance, to be defined below. Such a choice is aimed at enabling successful error concealment for any error concealment algorithm. Minimal intra-partition distance of i-th partition is defined to be

$$d_{\min}^{(i)} = \min_{\substack{\mathbf{x},\mathbf{y} \in S_i \\ \mathbf{x} \neq \mathbf{y}}} d(\mathbf{x},\mathbf{y}), \tag{1}$$

where $d(\mathbf{x},\mathbf{y})$ is the distance between $\mathbf{x}$ and $\mathbf{y}$, here chosen to be the Euclidean distance. The overall minimum intra-partition distance (subsequently referred to simply as the minimal distance) is the minimum of all $d_{\min}^{(i)}$, that is

$$d_{\min} = \min_{i=0,1,\ldots,P-1} d_{\min}^{(i)}. \tag{2}$$

Obviously, $d_{\min}$ is a function of C, i.e. $d_{\min} = d_{\min}(C)$.

Given the number of partitions P, the optimal (in terms of minimal distance) partition C^* of the signal domain is

$$C^* = \arg\max_{|C|=P} d_{\min}(C) \tag{3}$$

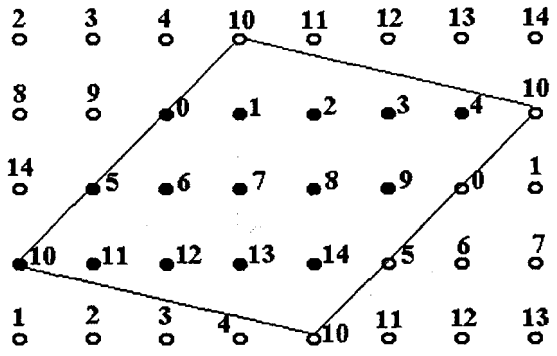


Fig. 1. Example of two-dimensional lattice partitioning into 15 partitions.

where $|C|$ is the number of elements in C. The solution to this problem in two dimensions, based on lattice partitioning, was proposed in [8]. In a later section we explain how it can be extended to higher dimensions. The solution is based on the observation that this problem is equivalent to sphere packing in $\mathbb{Z}^2$. Partitions produced by the algorithm in [8] are intersections of translation-equivalent sublattices of $\mathbb{Z}^2$ with the signal domain $\mathcal{D}$. In this case minimal intra-partition distance is uniform across partitions, i.e. $d_{\min}^{(i)} = d_{\min}^{(j)}, \forall i, j$. Also, partitions are uniformly distributed across the domain, so each description will contain approximately the same number of signal samples, ensuring that descriptions are equally important (balanced).

As an example, Figure 1 shows how $\mathbb{Z}^2$ lattice can be partitioned into $P = 15$ partitions, with $d_{\min} = \sqrt{17} \approx 4.12$. The pattern of black dots (enclosed in the parallelogram, labelled 0 through 14) is repeated throughout the lattice, and then domain $\mathcal{D}$ is cut out of the lattice. This operation of cutting out $\mathcal{D}$ from $\mathbb{Z}^2$ does not increase $d_{\min}$. The label assigned to the point shows the index of the partition to which the point belongs, e.g. if the point is labelled '7', it belongs to S_7. This labelling defines the *partitioning function* $p : \mathbb{Z}^2 \rightarrow \{0, 1, ..., P-1\}$ which we use in a later section to describe the partitioning of image and video data. In the remainder of this section we analyze descriptions produced in this way and typical scenarios in their transmission over lossy packet networks.

C. Analysis of domain-based multiple descriptions

If $d_{\min}$ is the minimal distance of $C = \{S_0, S_1, ..., S_{P-1}\}$, then each point $\mathbf{x}$ in the domain $\mathcal{D}$ is surrounded by the ball

$$B_{d_{\min}}(\mathbf{x}) = \{\mathbf{y} : d(\mathbf{x}, \mathbf{y}) < d_{\min}\},$$

of radius $d_{\min}$, centered at $\mathbf{x}$, which does not contain any points from the partition containing $\mathbf{x}$. Now suppose that one description, say (S_m, F_m), is lost/corrupted during transmission. Every $\mathbf{x} \in S_i$ is surrounded by a certain number of available neighbors which can be used to estimate $f(\mathbf{x})$. Table I shows how many neighbors are available around any missing sample, as a function of $d_{\min}$. The first column of the table contains the first few values of $d_{\min}$. The second column shows the minimal number of partitions P needed to achieve the corresponding $d_{\min}$. The third column shows the number of available neighbors (assuming single lost description) around any missing sample in the interior of the domain, and the last column shows the order of the interpolation method which can be used to estimate the missing sample. The "order" of the method refers to the maximal order of the surface which can be fit to the available signal samples for interpolation purposes. Of course, there are many ways, other than surface fitting, that can be used for interpolation. Hence, the last column is just an indication of how powerful the interpolation method can be used in each case.

Having large $d_{\min}$ also helps when more than one packet is lost. Figure 2 illustrates the situation when two packets (packet 1 and packet 2) are lost. The sample at point $\mathbf{x}$ is in packet 1, while the samples at $\mathbf{y_1}$ and $\mathbf{y_2}$ are in packet 2. The circle shows the possible location of samples from packet 2 closest to $\mathbf{y_1}$. Suppose we want to estimate the missing sample at $\mathbf{x}$ and we want to know how many of its neighbors are also missing. Samples in packet 1 are at a distance of at least $d_{\min}$ from $\mathbf{x}$, so they are not in the immediate neighborhood. The only points in the neighborhood of $\mathbf{x}$ which may be missing are those from packet 2. With reference to Figure 2, from the triangle inequality we have $d_1 + d_2 \geq d_{\min}$, so

$$d_2 \geq d_{\min} - d_1.$$

With $d_{\min}$ fixed and sufficiently large, the lower bound on d_2 increases as d_1 decreases.

$d_{\min}$	minimal P	number of neighbors	interpolation method order
$\sqrt{2}$	2	4	1
2	4	8	1
$\sqrt{5}$	5	12	3
$\sqrt{8}$	8	20	3
3	9	24	3
$\sqrt{10}$	10	28	5
$\sqrt{13}$	12	36	5
4	15	44	5

TABLE I

THE SIZE OF AVAILABLE NEIGHBORHOOD AS A FUNCTION OF MINIMAL DISTANCE

Hence, if d_1 is small (meaning $\mathbf{y_1}$ is close to $\mathbf{x}$), then d_2 is large (meaning $\mathbf{y_2}$ is not close to $\mathbf{x}$). The argument can be extended to more than two lost packets. In simple words, we may expect that for each additional lost packet, only one additional point will be missing from the neighborhood of $\mathbf{x}$.

The following sections explain our application of this type of MD coding to robust image and video transmission.

III. Application to dispersive packetization of images and video

In this section we discuss the creation of descriptions (packets) from the lattice partition. We refer to this procedure as *dispersive packetization* [9], due to the fact that samples from any given packet are maximally dispersed across the domain of the signal. In this paper we consider subband/wavelet-transformed images and video, although concepts introduced here can be readily modified to suit other transforms or raw data.

Given the total encoding rate and the maximal allowed packet length, we can determine the suitable number of packets, say P, and construct a suitable lattice partition as described in the previous section. The next step is to use this partition to construct partitions of subbands, which we discuss next.

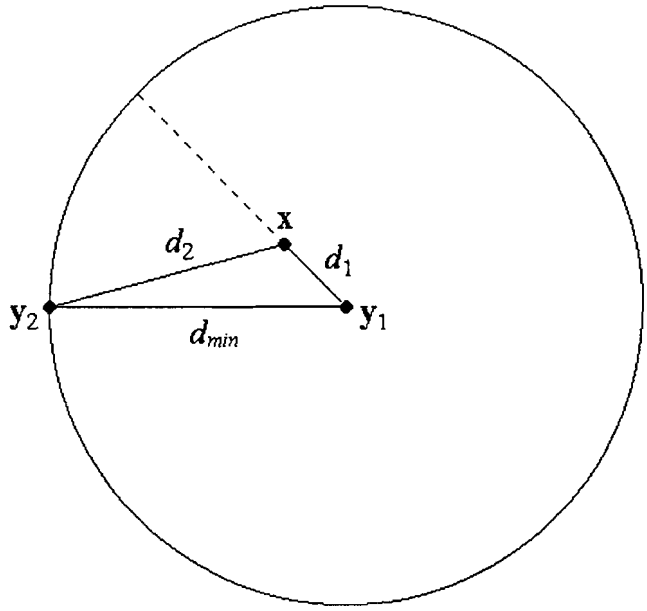


Fig. 2. The case of two lost descriptions.

A. Dispersive packetization of images

Figure 3 shows a two-level subband/wavelet decomposition of an image. One low-low (LL) band coefficient is shown in black and its neighborhood in the space-frequency domain is shown in gray. In case the sample shown in black is lost, it would be helpful to have as many of its neighbors as possible, to facilitate error concealment. Hence, we proceed in the following way.

Let the size of the LL subband be $N_1 \times N_2$. From the integer lattice divided into P partitions, we cut out the region $D_0 = \{0, 1, ..., N_2 - 1\} \times \{0, 1, ..., N_1 - 1\}$, so that each point in D_0 corresponds to one sample in the LL band. Restriction of the partitioning function $p(\cdot)$ to D_0 tells us how to packetize the LL band: if $p(i, j) = n$, then the sample at (i, j) is stored in the n-th packet. For higher frequency bands, we use modulo-shifted partitioning functions. If there are N subbands, indexed 0 through $N - 1$ in increasing order from low frequency to high frequency bands, then with k-th subband we associate the modulo-shifted partitioning function $p^{(k)} : \mathbb{Z}^2 \to \{0, 1, ..., F - 1\}$ defined as

$$p^{(k)}(\mathbf{x}) \triangleq (p(\mathbf{x}) + k) \bmod P, \qquad k = 0, 1, ..., N - 1. \tag{4}$$

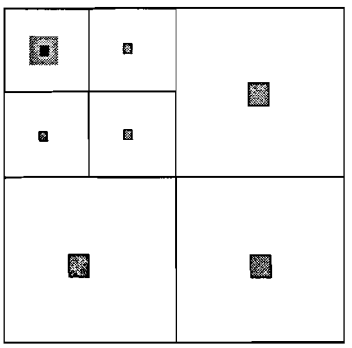
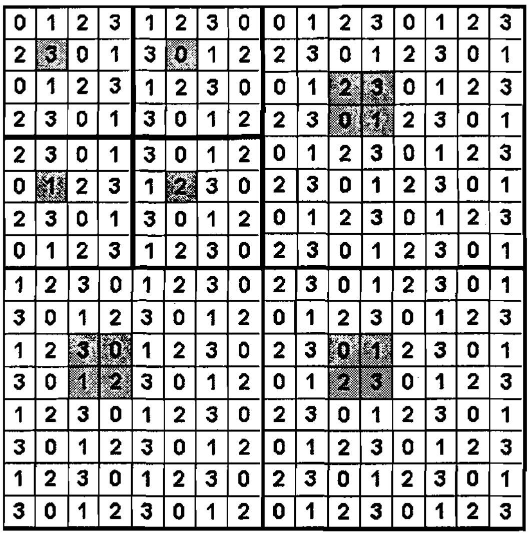


Fig. 3. Neighborhood of one low-low band coefficient in the space-frequency domain.

Equation (4) defines a one-to-one mapping from the set of partitions $\{S_0, S_1, ..., S_{N-1}\}$ into itself. The points that were assigned to S_n under $p(\cdot)$ will be assigned to $S_{(n+k) \bmod P}$ under $p^{(k)}(\cdot)$. Therefore, $p^{(k)}(\cdot)$ has the same distance properties as $p(\cdot)$ (i.e. $d_{\min}$ is the same for both functions), but their values repeat in different patterns.

For the k-th subband we cut out the region D_k (of the same size as the subband) from the lattice, and packetize according to the restriction of $p^{(k)}(\cdot)$ to D_k. The purpose of using modulo-shifted partitioning functions for different subbands is to disperse samples from the spatial orientation tree across the packets. For an M-level subband/wavelet decomposition, there are a total of 4^M samples in each tree, which is usually higher than the desired number of packets, so some of the samples from the same tree will have to appear in the same packet. Nevertheless, the construction of partitions ensures that approximately the same number of samples from each subband appear in every packet, and that no region of the image is favored by any of the packets. This makes the packets "equally important", which results in very low variation in reconstructed image quality for the fixed packet loss rate, as will be seen in the next section.

An example illustrates the above procedure for $M = 2$ level subband decomposition ($N = 7$ subbands) of a 16×16 image, partitioned into $P = 4$ partitions. In this case, seen in Figure 4, we have $d_{\min} = 2$. Numbers 0 through 3 indicate the partition to which the corresponding subband/wavelet sample belongs. Assume the subbands are ordered in a zig-zag manner from the lowest frequency band to the highest frequency band, and indexed



0	1	2	3	1	2	3	0	0	1	2	3	0	1	2	3
2	3	0	1	3	0	1	2	2	3	0	1	2	3	0	1
0	1	2	3	1	2	3	0	0	1	2	3	0	1	2	3
2	3	0	1	3	0	1	2	2	3	0	1	2	3	0	1
2	3	0	1	3	0	1	2	0	1	2	3	0	1	2	3
0	1	2	3	1	2	3	0	2	3	0	1	2	3	0	1
2	3	0	1	3	0	1	2	0	1	2	3	0	1	2	3
0	1	2	3	1	2	3	0	2	3	0	1	2	3	0	1
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3
1	2	3	0	1	2	3	0	2	3	0	1	2	3	0	1
3	0	1	2	3	0	1	2	0	1	2	3	0	1	2	3

Fig. 4. Example of partitioning 7 subbands into 4 partitions.

0 to 6. Given the partitioning scheme for the lowest frequency band, the partition for each higher frequency band is obtained by adding its index (modulo 4) to the partitioning function for the lowest frequency band, as given by (4). One spatial orientation tree of subband/wavelet samples is shown in gray. It is apparent from the figure that each description contains the same number of samples from each tree and also the same number of samples from each subband. This makes the descriptions "equally important."

After partitioning, each description (packet) is encoded independently of other descriptions. Our implementation of the domain-based MD image codec is based on the subband/wavelet layered PCM codec from the "Wavelet image compression construction kit" [10]. A set of embedded quantizers is designed for each subband and samples are quantized in a layered manner. Subband samples are first partitioned into descriptions, then quantized and finally each description (packet) is encoded using a context-based adaptive arithmetic coder. For each subband sample, the encoding context is made up of symbols for that sample from previously processed bit planes, which are encoded in the same packet. In this way context boundary matches the packet boundary and makes the packets individually decodable. Information theory tells us that conditioning does not increase entropy and we can expect that increased coding efficiency can be achieved if the context for encoding a given sample is increased to include its neighborhood in space and frequency.

But in that case, the entire context would have to be stored in the same packet, which would reduce the degree of dispersion and make error concealment more difficult. Hence, we have sacrificed coding efficiency for robustness. This coding efficiency loss due to context restriction is not very large, however, and our coder compares favorably to PZW [15] (which is a robust version of SPIHT [16]), as will be seen in the results section.

Error concealment is carried out in the three subbands (LL, HL and LH) at the lowest decomposition level, where most of the signal energy is usually concentrated. Missing samples in the LL subband at the lowest decomposition level are interpolated bi-linearly from the four nearest non-missing neighbors in the horizontal and vertical directions, and missing samples in the LH and HL subbands at the lowest decomposition level are interpolated linearly from nearest non-missing samples in the direction in which the subband has been low-pass filtered. Missing samples in the higher frequency bands are set to zero. Simulation results for dispersive packetization of images are given in part A of section IV. Here we extend domain partitioning to dispersive packetization of video.

B. Dispersive packetization of video

In the case of video, we focus on the invertible motion compensated 3-D subband/wavelet video coder (IMC-3DSBC) from [11]. A typical group-of-pictures (GOP) of this coder contains 16 frames and its structure is shown in Figure 5. The top level represents the video at full frame rate. Neighboring frames are decomposed using a motion-compensated filter bank to produce the temporal low frequency bands (solid lines) and temporal high frequency bands (dashed lines) at the next lower level. Motion vectors are shown as arrows. Low temporal frequency bands are effectively the motion-compensated (MC) averages of two neighboring frames at full frame rate, and they occur at half the frame rate. The process is repeated until we obtain the MC average of all 16 frames in the GOP which occurs at 1/16 of the full frame rate. Transmitted data in this case is naturally divided into five layers, labeled (1) through (5) in the figure, and is suitable for scalable multicast transmission. Receivers which receive layer (1) can reconstruct the video at 1/16 of the full frame rate, those which receive layers (1) and (2) can reconstruct the video at 1/8 of the full frame rate, and so on.

If we think of a video as a signal whose domain is $\mathbb{Z}^3$, then the packetization scheme

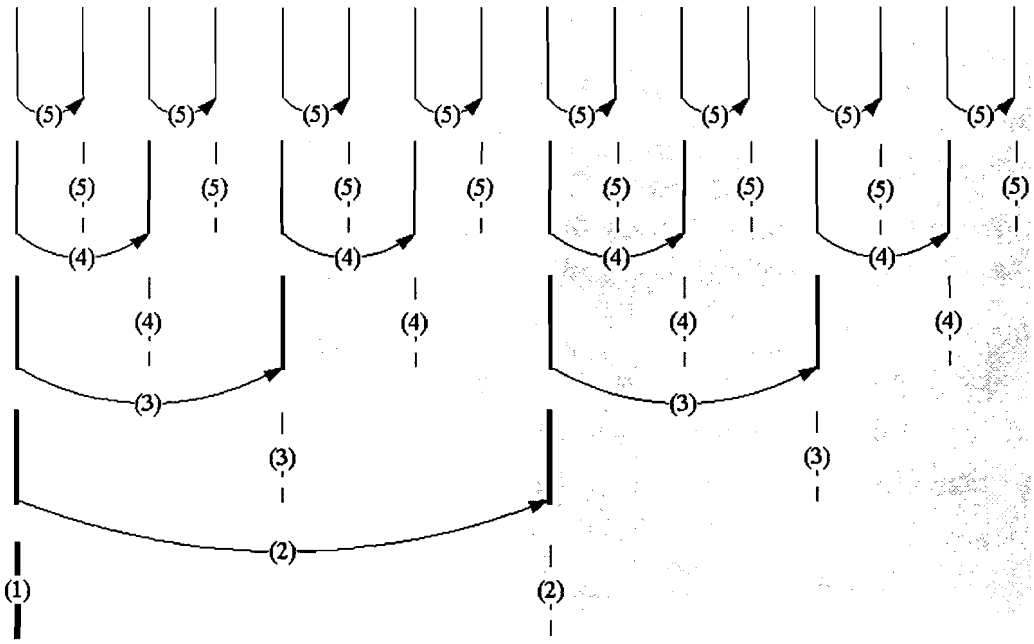


Fig. 5. The GOP structure of the MC-3DSBC video coder.

for video can be obtained by solving the sphere packing problem in $\mathbb{Z}^3$. However, besides being more difficult than the problem in two dimensions, it is not clear whether such model is appropriate in our case. In two dimensions it seems intuitively clear that the horizontal and the vertical direction are of the same importance. In the case of video, however, the importance of the temporal direction in relation to the horizontal and vertical directions depends on several parameters, such as the frame rate and the sensitivity of the human visual system to the errors in the temporal direction. The layered structure of the subband/wavelet transform brings in additional complexity. For these reasons, we have chosen a simpler method of extending the previously described 2-D packetization, wherein each layer is packetized independently.

Within each layer, data is packetized dispersively, so that each packet contains approximately equal amount of information about every frame in that layer. Layer (1) consists of a single frame and is packetized as an image, as discussed in the previous subsection. Other layers consist of a sequence of temporal high-frequency frames, with a motion vector (MV) field between each pair of frames. Suppose we constructed a packetization scheme

for one frame of a given layer as described before. The packetization scheme for the other frames and MVs in that layer can then be obtained using equation (4), where k now indicates the time-index of the frame or MV field on the temporal axis. This is a simple way of extending 2-D packetization scheme to 3-D and ensures that samples (or MVs) which are at the same spatial location, but neighboring in time, appear in different packets. This feature enables simple error concealment.

Illustration of how the packetization scheme is extended to 3-D is shown in Figure 6 for intraframe video. Two neighboring 16×16 frames, each decomposed by a two-level subband/wavelet transform, are shown in the figure. These represent the temporal high frequency frames in one of the layers. Frame 0 is partitioned into 4 descriptions in the same manner as the image in Figure 4. The partitioning scheme for the next frame (frame 1) is then obtained by adding 1 (modulo 4) to the number in each coefficient location in frame 0. The process is then repeated for the following frames. Again, one spatial orientation tree of coefficients is shown in gray in both frames, to illustrate that coefficients with the same space/frequency localization but different temporal localization appear in different descriptions (packets).

In a motion-compensated video there is a motion vector field between the neighboring frames. Motion compensated temporal subband analysis produces one low temporal frequency band and one high temporal frequency band from each pair of frames. All low temporal frequency bands from a GOP are effectively compacted into the frame at layer (1) (see Figure 5), while high temporal frequency bands together with motion vectors represent enhancement layers needed to synthesize the video at higher frame rates. Hence, at each of the layers (2) - (5) we are transmitting a sequence of high temporal frequency bands and their corresponding motion vector fields. Partitioning of this data structure is carried out in the same spirit as before - if a motion vector and a subband sample have the same spatial localization (i.e. motion vector references the subband sample), then we try to put them in a different description. The procedure is illustrated in Figure 7. The figure shows a prediction residual (i.e. high temporal frequency band) which has been further decomposed through spatial subband/wavelet transform. One spatial orientation tree of subband/wavelet samples is shown in gray. On the left side of the figure we show

the domain of the motion vector field. Within this domain, the location of the motion vectors which reference the indicated spatial orientation tree is also shown in gray. If the number of descriptions (packets) was larger than the number of the motion vectors in the shaded region plus the number of samples in the tree, we could arrange that none of them appear in the same packet. However, as mentioned above, the number of samples in the tree alone is usually larger than the number of packets we would practically want to use, so each motion vector will end up in the same packet as a fraction of the subband samples which it references. This does not seem to cause much problems in practice. Partition for the domain of the motion vector field is obtained by modulo-shifting the partitioning function of one of the subbands. Figure 7 shows the result when the partitioning function for the LL subband is shifted by 1 modulo 4. Our implementation uses the same method - shifting the LL partitioning function by 1 modulo P, where P is the number of descriptions (packets), although one could use partitioning function subbands other than LL and/or shift by any number k, $1 \leq k \leq P-1$. It should be noted that prediction residuals (high temporal frequency bands) are not natural images and we cannot know in advance which subbands contain more energy than others (i.e. which subbands are more important).

At the receiver, missing data (samples or motion vectors) are estimated from the available *spatially* neighboring data from the same subband/frame only. While it is possible to use both spatial and temporal neighborhoods in the estimation process, in this work we focus on the simpler and faster algorithms which utilize only the spatial neighborhood. This gives good results in practice, because of the nature of the packetization schemes proposed here - they have been designed to maximize the number of available neighboring samples for any missing sample in the case of a single lost packet. It was also argued that in the case of more than one lost packet, the number of available neighbors is still large. Having sufficiently many spatial neighbors around every missing piece of data improves the quality of the estimate.

Missing samples from the high temporal frequency bands are estimated as the median of however many of the 8 nearest neighbors are available. If none of the 8 nearest neighbors are available, missing sample is set to zero. Missing motion vectors are estimated using the vector median filter [12]. The estimation process for motion vectors is as follows.

First, available motion vectors in the neighborhood of the missing vector are collected, as shown in Figure 8. Let $V \triangleq \{\mathbf{v}_0, \mathbf{v}_1, ..., \mathbf{v}_{M-1}\}$ be the set of M available motion vectors in the neighborhood of the missing motion vector $\mathbf{v}$. If $M = 0$ (i.e. all vectors from the neighborhood are also lost) we set $\widehat{\mathbf{v}} = \mathbf{0}$, otherwise we proceed to find the vector median. The $p-$norm vector median of the vectors in a set V is defined as the vector $\mathbf{v}_{vm} \in V$ such that

$$\sum_{i=0}^{M-1} \|\mathbf{v}_{vm} - \mathbf{v}_i\|_p \leq \sum_{i=0}^{M-1} \|\mathbf{v}_j - \mathbf{v}_i\|_p, \quad j = 0, 1, ..., M-1. \tag{5}$$

The choice of the norm influences the result of vector median filtering. The most popular choices are $1-$norm and $2-$norm. Here we use the squared $2-$norm (i.e. squared Euclidean distance) in which case the vector median is the vector $\mathbf{v}_{vm} \in V$ that is closest to the mean of the vectors in V [13]. This choice is motivated by the speed of the resulting vector median filtering algorithm. The estimate of the missing motion vector $\mathbf{v}$ is obtained as

$$\widehat{\mathbf{v}} = \arg\min_{\mathbf{u} \in V} \|\mathbf{u} - \mathbf{v}_{mean}\|_2^2, \tag{6}$$

where $\mathbf{v}_{mean} = \frac{1}{M}\sum_{i=0}^{M-1}\mathbf{v}_i$. Alternatively, one may choose to estimate the missing motion vectors using an iterative MAP estimate from the neighboring motion vectors, or the temporal-spatial approach [14] based on finding the best match in the previous frame of the boundary of the region referenced by the missing motion vector. However, both these approaches are much more complex than median filtering and hence not suitable for simple receivers. Results in [14] indicate that they may provide about 1 dB advantage over median filtering, while in turn median filtering is about $0.5-1$ dB better than replacing the missing motion vector by the average of its neighbors. Results in [14] seem to indicate that vector median filtering provides a good trade-off between complexity and performance.

IV. Results

A. Image transmission

The process of creating domain-based multiple descriptions described above does not introduce any extra redundancy. For a given signal and desired quality under no-loss conditions, it produces bit-streams of much lower bit rate than other MD image coding methods, [2]-[5] which makes direct comparison difficult. Hence, we compare our method

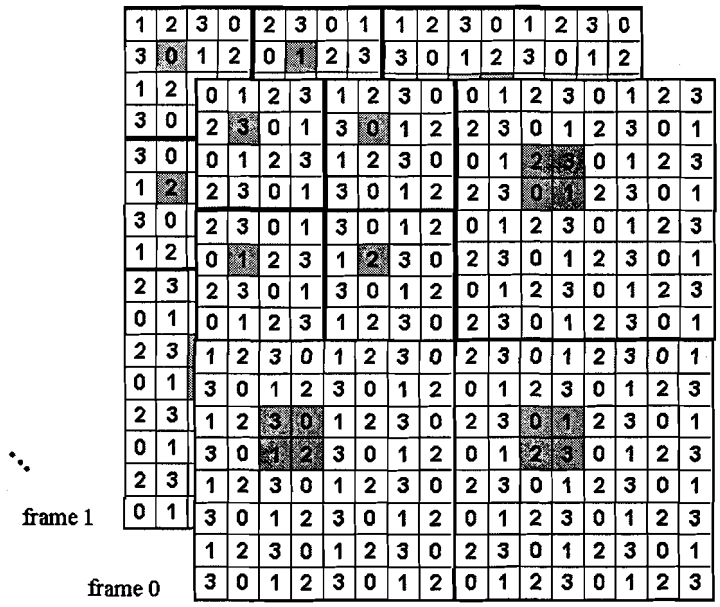


Fig. 6. Extension of dispersive packetization to 3-D.

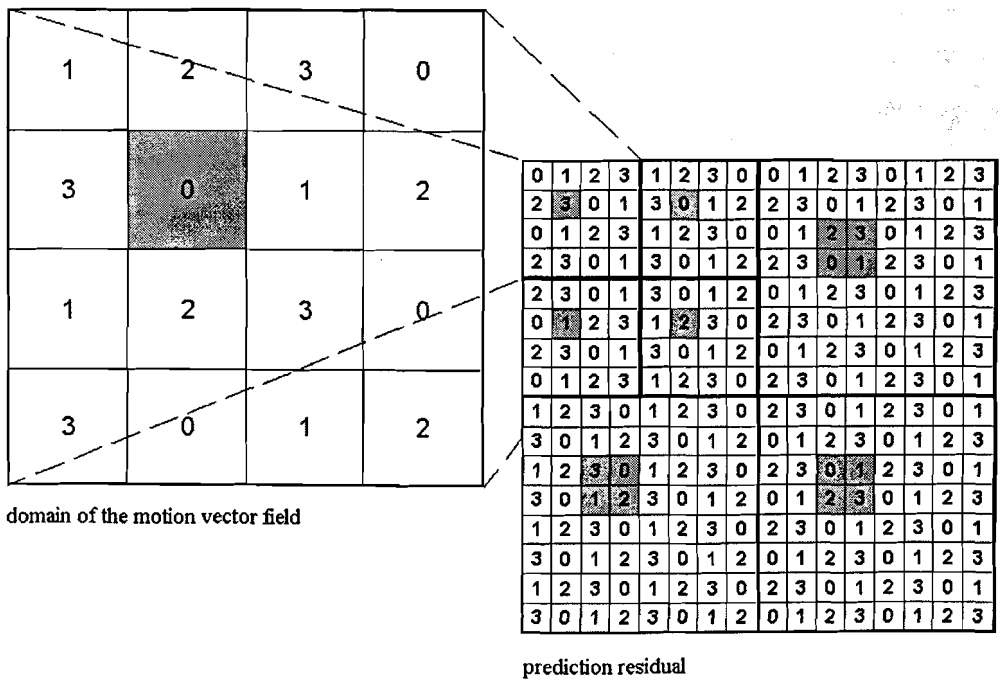


Fig. 7. Dispersive packetization of motion vectors

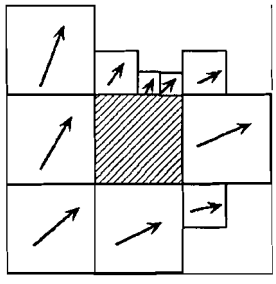


Fig. 8. Illustration of motion vector concealment - shaded area is the location of the missing motion vector and arrows indicate its neighbors.

against another similar state-of-the-art subband/wavelet method for robust image transmission, called Packetized Zerotree Wavelet (PZW) [15].

There are two major differences between our method and PZW. First, PZW is a tree-based coder (hence all subband samples from a spatial orientation tree are encoded in the same packet), while we make effort to disperse samples from tree across the packets. Second, our method is based on domain partitioning which maximizes the distance between the samples in the same packet, while the method used in PZW appears to be suboptimal in this sense. Both methods use similar error concealment algorithms for missing samples in the LL band - in our case missing samples are interpolated bi-linearly from the four nearest non-missing neighbors in the horizontal and vertical directions, and in the case of PZW they are estimated as the average of however many of the 8 nearest neighbors are available.

We carried out a set of experiments on 512×512 grayscale *Lena* and *Peppers* images. They were encoded into 16 packets, each of about 430 bytes in size, with a total rate of 0.21 bpp. In each experiment we fix the number of lost packets out of a total of 16 packets and reconstruct images for all possible combinations of lost packets.

Table II shows average PSNR results in dB and its standard deviation in brackets. Results in the table show that PSNR does not vary much at a fixed packet loss rate, as indicated by the low values of the standard deviation. This is very desirable since it enables the decoder to estimate the image quality purely from the number of received packets. It also illustrates the "equal importance" of the packets. Average PSNR results are also plotted in Figures 9 and 10, along with the PZW results from [15]. While the performance

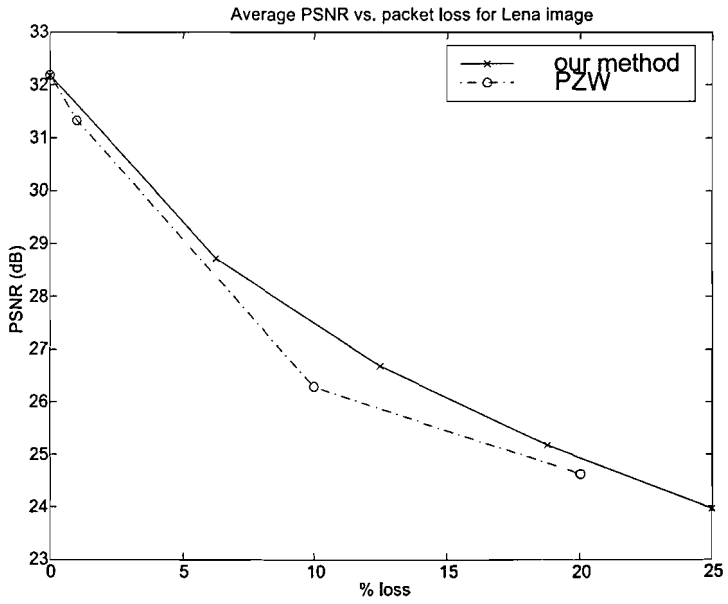


Fig. 9. Comparison of our method with PZW on *Lena* image.

of the two methods at lowest packet loss is similar, the figures show the advantage of our method over PZW of $0.5 - 1$ dB at packet loss above 10%. This gain is expected to be mainly due to a higher degree of dispersion of data across packets, which enables better error concealment.

% loss	0	6.25	12.5	18.75	25
Lena	32.2	28.7 (0.3)	26.7 (0.3)	25.2 (0.3)	24.0 (0.3)
Peppers	31.6	28.0 (0.4)	25.8 (0.4)	24.2 (0.4)	22.9 (0.4)

TABLE II

PSNR RESULTS FOR *Lena* AND *Peppers* AT 0.21 BPP

There is also a qualitative difference between images produced by our method and PZW, arising from the fact that in PZW subband trees are either completely lost or completely available, while in our case each packet usually contains at least one sample from every subband tree. This qualitative difference is illustrated in Figure 11 where we show the *Lena* image compressed at 0.21bpp and subject to 25% packet loss. Figure 11(a) shows

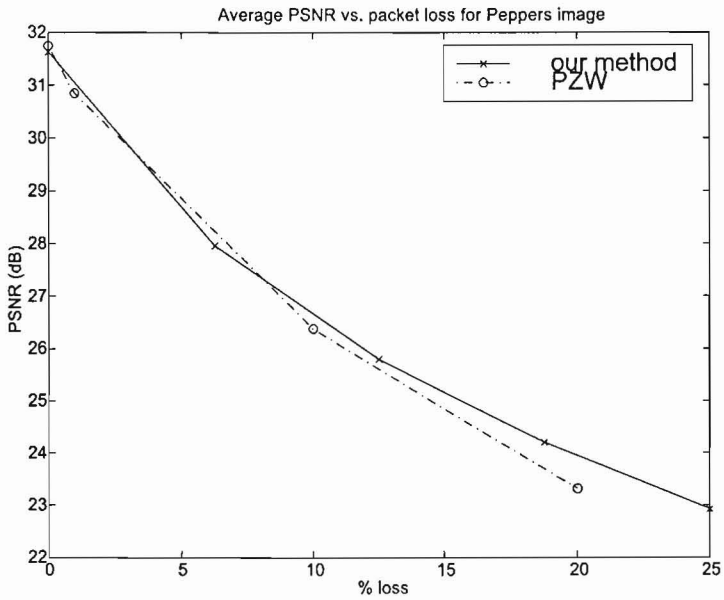


Fig. 10. Comparison of our method with PZW on *Peppers* image.

the image packetized in a manneer similar to PZW, where every subband tree is confined to only one packet, and hence completely lost if that packet is lost. Figure 11(b) shows the dispersively packetized image. In both cases, error concealment is the simple bi-linear interpolation from the four nearest non-missing neighbors in the horizontal and vertical directions, as mentioned before. The specific combination of lost packets was chosen so that the resulting images have nearly the same PSNR (up to the first decimal point), hence suffering from nearly the same mean-squared error distortion. We observe that the distortion in the image in part (a) is concentrated in the positions of the missing subband trees, and in those areas some important details (such as the left eye) are completely missing. In the image of part (b) distortion is more evenly distributed across the image, and all details, although somewhat distorted, are still visible.

B. Video transmission

In our simulations of video transmission we used a simple 2-state Markov model (Figure 12) to simulate network behavior. As discussed in [17], this model can reasonably approximate Internet transmission. In the 'good' state G, all packets are correctly received,

(a) (b)

Fig. 11. Illustration of the effects of dispersion: *Lena* image compressed at 0.21bpp with 25% packet loss (a) PZW-like method, PSNR = 22.9dB; (b) our method, PSNR = 22.9dB.

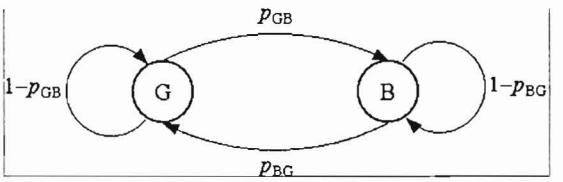


Fig. 12. A simple 2-state Markov model of network transmission.

while in the 'bad' state B, all packets are lost. Transition probabilities p_{GB} of going from G to B, and p_{BG} of going from B to G are sufficient to specify the model. In [17], several experiments were performed by sending data between US and Germany, and the above model was fitted to the results. Instead of transition probabilities, two other quantities were reported: average packet loss probability $P_B = p_{GB}/(p_{GB} + p_{BG})$, and average packet loss burst length $L_B = 1/p_{BG}$.

Results of four experiments are reported in this section. Experiments are divided into three groups. The first group (experiments 1 and 2) are based on the grayscale *Mobile calendar* sequence (SIF resolution, 30fps) encoded at 0.45 Mbps with a GOP of 16 frames and the average PSNR of 22.3 dB. The second group (experiments 3 and 4)

are based on grayscale *Football* sequence (SIF resolution, 30fps) encoded at 0.97 Mbps with a GOP of 4 frames and the average PSNR of 26.3 dB. In each group of experiments we simulated video transmission in two scenarios: first corresponding to network parameters $(P_B, L_B) = (0.1, 3)$ (experiments 1 and 3), and second corresponding to $(P_B, L_B) = (0.15, 2)$ (experiments 2 and 4). We compare our packetization scheme to the interleaved slice-based packetization scheme, which is part of the H.323 recommendation for packet-lossy environments, and is described in [18], [19]. In such a scheme, slices (which correspond to one row of DCT macroblocks) are packetized in an interleaved manner, i.e. n-th row of macroblocks is packed into $(n \bmod P + 1)$-th packet. We kept the number of packets the same as in the dispersive packetization case. The slices were generated by collecting all subband coefficients that correspond to a row of 16×16 macroblocks. In all cases, lost coefficients and lost motion vectors were estimated as described in the previous section.

Average PSNR results in dB are shown in Table III. SP stands for slice-based packetization and DP stands for dispersive packetization. Frame-by-frame PSNR is shown in Figures 13 - 16. It can be seen that our method (dispersive packetization) can provide over 1 dB advantage on average over slice-based packetization, using the same concealment algorithm. The gain is achieved simply by sending data in a different way, which enables better error concealment. In certain rare instances, for example for frames 1 to 8 in Figure 13, slice-based packetization can offer some advantage over dispesive packetization in terms of PSNR. This occurs when the lost slices are particularly easy to conceal, in which case no advantage is achieved by distributing the loss uniformly across the frame as done in DP. These events seem to be rare, however, and on average, advantage of DP over SP is on the order of 1 dB.

As an illustration, in Figure 17 we show frame 87 of the *Football* sequence from experiment 4. Parts (a) and (b) show the original and coded version of the frame with no loss, respectively. Part (c) shows the frame reconstructed from the slice-packetized sequence, while part (d) shows the frame from the dispersively packetized sequence, both at the same packet loss rate. Parts (e) and (f) are segments of the pictures in (c) and (d), respectively. In the case of slice-based packetization, when a piece of data (either subband sample or

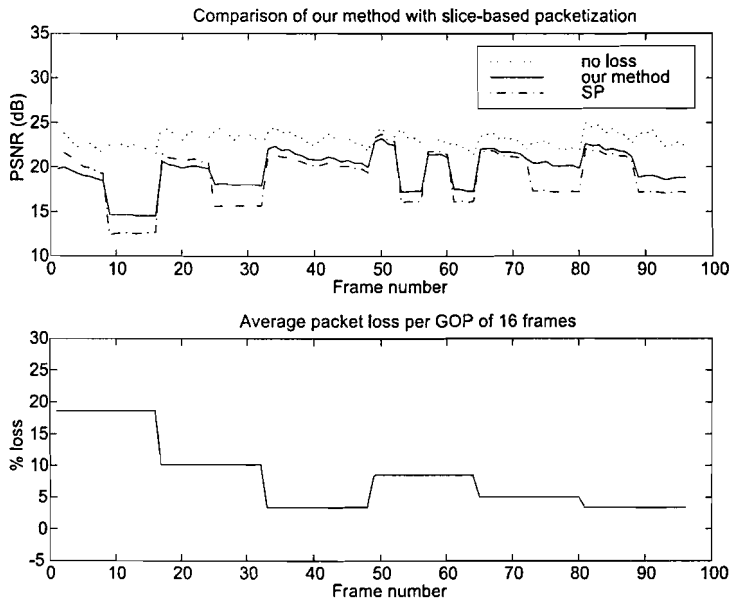


Fig. 13. Experiment 1 - transmission of *Mobile calendar* sequence at 8.2% average packet loss.

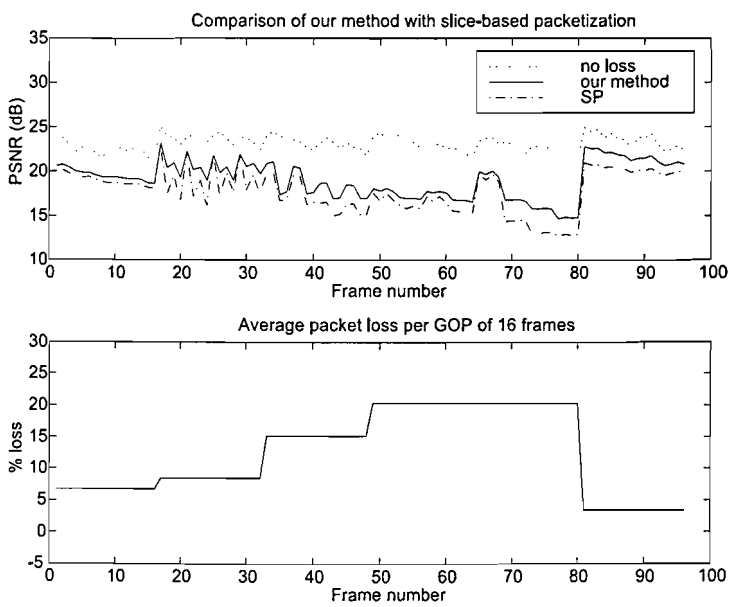


Fig. 14. Experiment 2 - transmission of *Mobile calendar* sequence at 12.4% average packet loss.

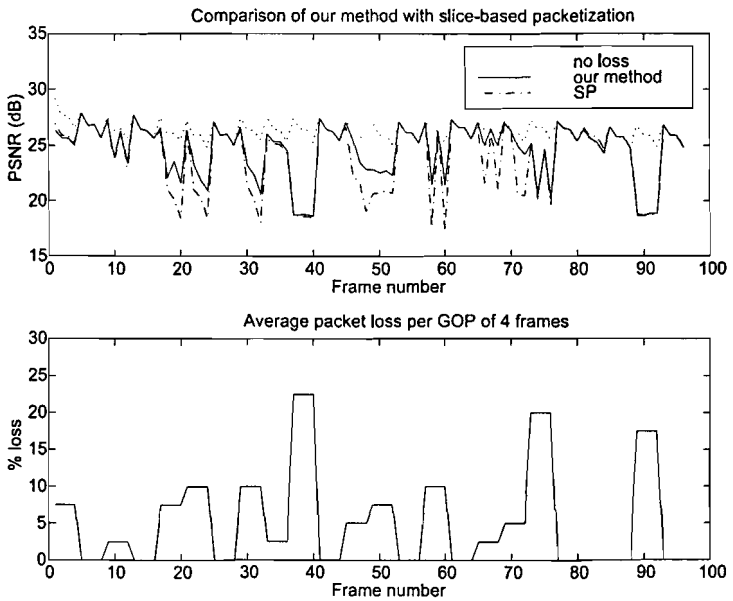


Fig. 15. Experiment 3 - transmission of *Football* sequence 5.4% average packet loss.

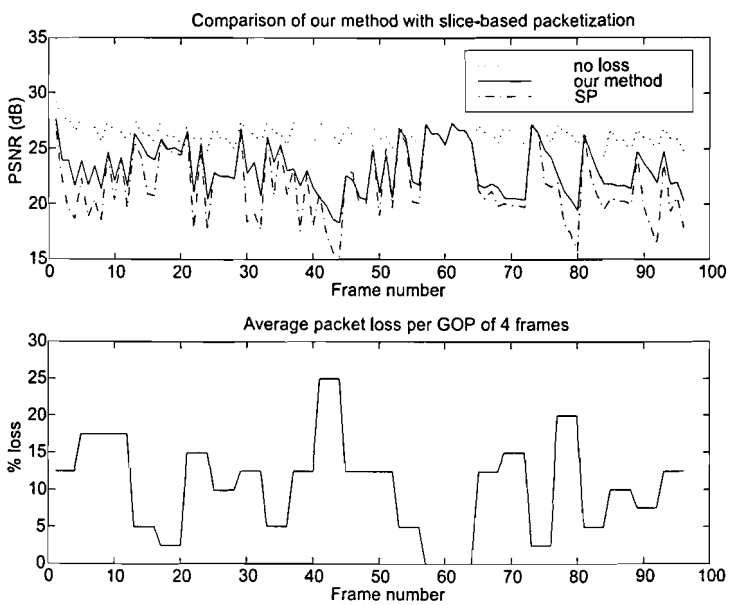


Fig. 16. Experiment 4 - transmission of *Football* sequence at 10.4% average packet loss.

Experiment	Avg. packet loss	PSNR for SP	PSNR for DP	Gain of DP over SP
1	8.2%	18.9 dB	19.7 dB	+0.8 dB
2	12.4%	17.8 dB	19.1 dB	+1.3 dB
3	5.4%	24.0 dB	24.6 dB	+0.6 dB
4	10.4%	21.8 dB	23.2 dB	+1.4 dB

TABLE III

AVERAGE PSNR RESULTS IN DB FOR SIMULATED VIDEO TRANSMISSION.

MV) is lost, its nearest neighbors in the horizontal direction are also lost, which reduces the size of the neighborhood available for concealment. This generally lowers the quality of the estimate and may cause the loss of some important details, as illustrated in the figure. The dispersively packetized frame is not free from the problems of this type, but they occur less often, and the overall quality of the frame is better, both visually and in terms of PSNR.

V. CONCLUSIONS

We have presented a method of creating domain-based multiple descriptions for robust image and video coding and transmission. The method is based on partitioning the domain of the signal into sets, such that points within each set are as far from each other as possible. This dispersive packetization of images enables simple error concealment algorithms to produce acceptable results in low to moderate packet loss scenarios, without using FEC or other forms of extra redundancy. Experimental results indicate that dispersive packetization is suitable for image and video transmission over lossy packet networks. In the case of image transmission, advantages of up to $0.5 - 1$ dB in PSNR were achieved with respect to PZW over a range of packet loss rates. In case of transmission of motion compensated video, average gains of up to 1.4 dB over the conventional slice-based packetization were observed.

The method presented here can be extended to handle higher packet loss rates, by introducing some redundancy through error correction codes and/or by combining it with MDSQ. It would be interesting to study trade-offs between different MD coding methods

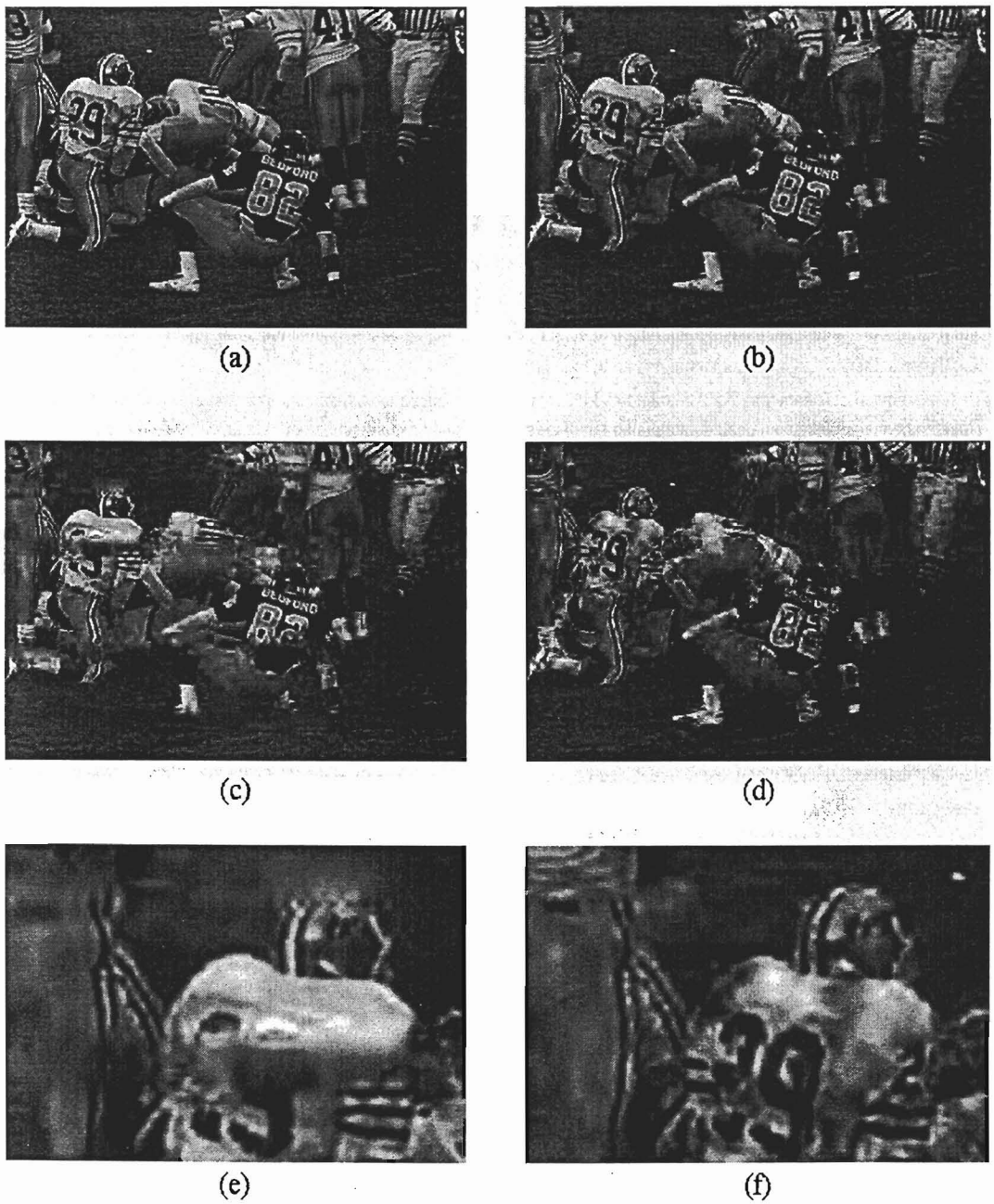


Fig. 17. Frame 87 of the *Football* sequence: (a) original; (b) coded, no loss, PSNR = 25.7 dB; (c) slice-based packetization, 10.0 % loss, PSNR = 20.2 dB; (d) dispersive packetization, 10.0 % loss, PSNR = 21.7 dB; (e) and (f) are segments of pictures in (c) and (d), respectively, showing the results of error concealment.

under various network conditions, and find ways of combining them to achieve improved robustness. These are the topics for future research.

REFERENCES

[1] V. A. Vaishampayan, "Design of Multiple Description Scalar Quantizers," *IEEE Trans. Inform. Theory*, vol. 39, no. 3, pp. 821-834, May 1993.

[2] S. D. Servetto, K. Ramchandran, V. A. Vaishampayan, and K. Nahrstedt, "Multiple Description Wavelet Based Image Coding," *IEEE Trans. Image Processing*, vol. 9, no. 5, pp. 813-826, May 2000.

[3] Y. Wang, M. T. Orchard, and A. R. Reibman, "Multiple description image coding for noisy channels by pairing transform coefficients," in *Proc. IEEE Workshop on Multimedia Signal Proc.*, pp. 419-424, Princeton, NJ, June 1997.

[4] V. K. Goyal, J. Kovačević, R. Arean, and M. Vetterli, "Multiple description transform coding of images," in *Proc. IEEE Int. Conf on Image Proc. (ICIP'98)*, Chicago, IL, Oct. 1998.

[5] R. Puri and K. Ramchandran, "Multiple description source coding using forward error correction codes," in *Proc. 33rd Asilomar Conf. on Signals, Systems and Computers*, Pacific Groove, CA, Oct. 1999.

[6] G. L. Nemhauser and L. A. Wolsey, *Integer and combinatorial optimization*, New York: John Wiley & Sons, 1999.

[7] I. V. Bajić, *Robust coding and packetization of images and intraframe-coded video*, MS Thesis, Renssealer Polytechnic Institute, Troy, NY, June 2000.

[8] I. V. Bajić and J. W. Woods, "Maximum minimal distance partitioning of the $\mathbb{Z}^2$ lattice," submitted to *IEEE Trans. Inform. Theory*, April 2001.

[9] I. V. Bajić, J. W. Woods, and A. M. Chaudry, "Robust transmission of packet video through dispersive packetization and error concealment," in *Proc. Packet Video Workshop (PV'2000)*, Cagliari, Sardinia, Italy, May 2000.

[10] G. M. Davis, "Wavelet image compression construction kit version 3.0," 1997, available at www.cs.dartmouth.edu/~gdavis/wavelet/wavelet.html

[11] S-T. Hsiang and J. W. Woods, "Invertible three-dimensional analysis/synthesis system for video coding with half-pixel-accurate motion compensation," in *Proc. SPIE 3653, Visual Communications and Image Processing'99*, January 1999.

[12] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proc. IEEE*, vol. 78, no. 4, pp. 678-689, April 1990.

[13] M. Barni, "A fast algorithm for 1-norm vector median filtering," *IEEE Trans. Image Processing*, vol. 6, no. 10, pp. 1452-1455, October 1997.

[14] P. Salama, N. B. Shroff, and E. J. Delp, "Error concealment in MPEG video streams over ATM networks," *IEEE J. Select. Areas Commun.*, vol. 18, no. 6, pp. 1129-1144, June 2000.

[15] J. K. Rogers and P. C. Cosman, "Robust Wavelet Zerotree Image Compression with Fixed-Length Packetization," in *Proc. DCC'98*, pp. 418-427, Snowbird, UT, 1998.

[16] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 6, no. 3, June 1996.

[17] B. Girod, K. Stuhlmüller, M. Link, and U. Horn, "Packet loss resilient Internet video streaming," in *Proc. SPIE Visual Comm. and Image Processing (VCIP'99)*, pp. 833-844, January 1999.

[18] S. Wenger, G. Côté, M. Gallant, and F. Kossentini, "H.263 Test model 11, revision 3," (Q15-G-16rev3), October 1999.

[19] Y. Wang, S. Wenger, J. Wen, and A. Katsaggelos, "Error resilient video coding techniques," *IEEE Signal Processing Magazine*, vol. 17, no. 4, pp. 61-82, July 2000.

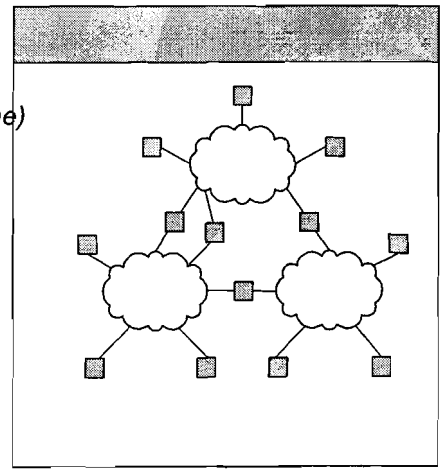
Some Communication problems in Networking

A.J. Han Vinck, University of Essen

- Introduction
- research problems
  - Optical processing
  - synchronization
  - Pulse Position Modulation

Internet = collection of networks

- access
  - *cheap (DSL, Power-line)*
- routing
  - *fast (optical), WDM*
- protocol translation
  - *standards*

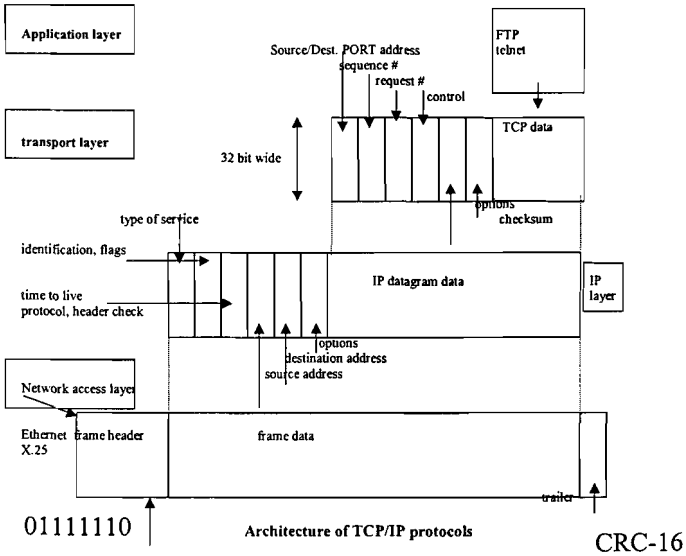


OSI problem solving layers

- Application: everything else
- Presentation: convert, compress, secure
- Session: managing users dialogue
- Transport: organize packet transmission

IP

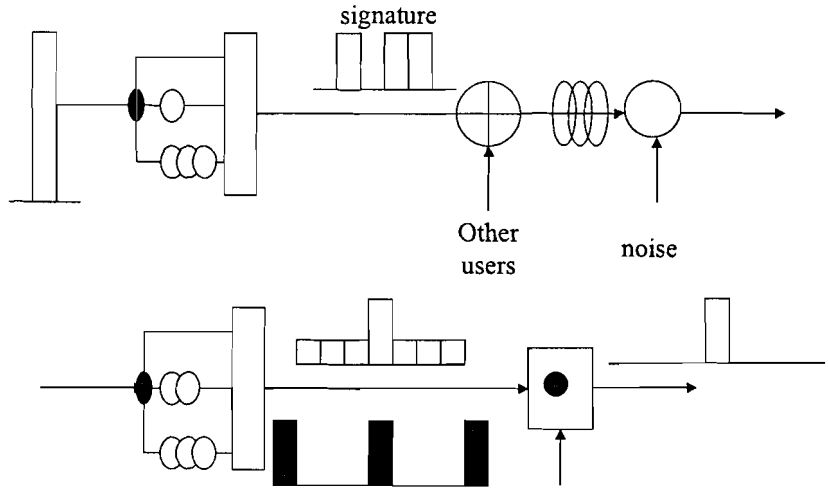
- Network: route packets
- Data link: reliable transmission
- Physical: transmit "bits"



Architecture of TCP/IP protocols

Research problems

- all optical processing:
  - *transceiver + switching*
  - *avoid going to electrical - optical memory -*
- synchronization:
  - *burst versus continuous transmission*
  - different detection rule
- pulse position modulation
  - *coded PPM for narrow and broadband noise*
  - *(optical power +)*



OPTICAL TRANSMITTER/RECEIVER

Example for two weight 3 sequences

- correct signature

000000000001011
000000000001011
000000000001011

auto-correlation 3

- in-correct signature

000001000010001
000001000010001
000001000010001

cross correlation 1

for good signatures x and y

- Property: $x, y \in \{0, 1\}$

AUTO CORRELATION	$\sum_{t=0}^{n-1} x_t x_{t+\tau} \leq 1 \quad \tau \geq 1 \quad (t+\tau) \bmod n$
CROSS CORRELATION	$\sum_{t=0}^{n-1} x_t y_{t+\tau} \leq 1 \quad \tau \geq 0 \quad (t+\tau) \bmod n$

Signature Construction problem

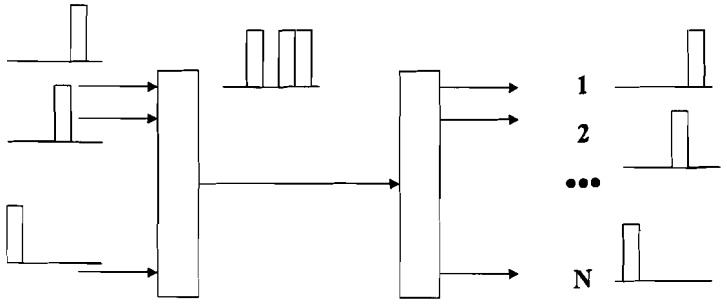
- *large set* with *low-weight* signatures

$$\frac{2n}{w^2(w-1)^2} \leq |S| \leq \left\lfloor \frac{n-1}{w(w-1)} \right\rfloor$$

- example: n = 15, w = 3, |S| = 2 OPTIMUM!
- Research problem: |S| for correlation $\lambda > 1$

Optical Switching

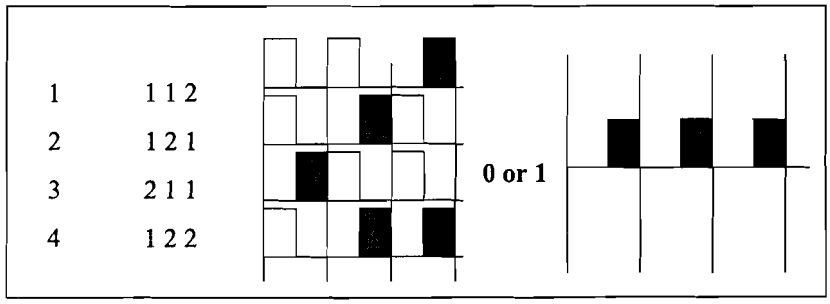
- Time division: = maximum efficiency 1



- for random access: max. efficiency = 0.69

R:= 0.69 also for random access PPM

- ex: 2 active users out of 4



- ex: receive ({1},{1,2},{2}) = 

- R(TDMA) = 2/8 R(PPM) = 2/6

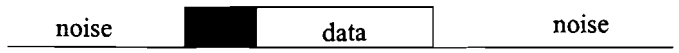
Comparisons for access

- Time Division Multiple Access (TDMA)
  - for T users maximum rate: T/N
  - needs transmitter synchronization
- Retransmit if collision (CSMA, ALOHA)
  - maximum rate 0.5 bit /dimension
  - needs feedback; needs parameter for delay
- Random Access PPM
  - maximum rate 0.69/dimension
  - no feedback needed; low density input

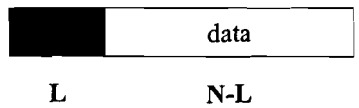
Synchronization of packets

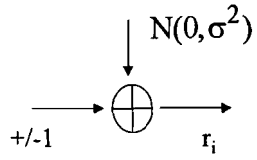
- continuous transmission

  - look for sync position in the data
  - use bit-stuffing
- bursty traffic: we need another rule



Synchronization Rules:





- Massey: continuous: *over L positions*

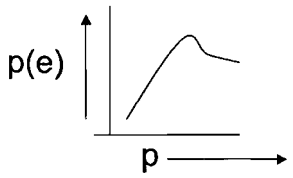
$$\boxed{\prod_{i=0}^{L-1} e^{r_i c_i/\sigma^2}} \Big/ \boxed{(e^{r_i/\sigma^2} + e^{-r_i/\sigma^2})}$$

- for bursty traffic: *over N positions*

$$\boxed{\prod_{i=0}^{L-1} e^{r_i c_i/\sigma^2}} * \boxed{\prod_{i=L}^{N-1} (e^{r_i/\sigma^2} + e^{-r_i/\sigma^2})}$$

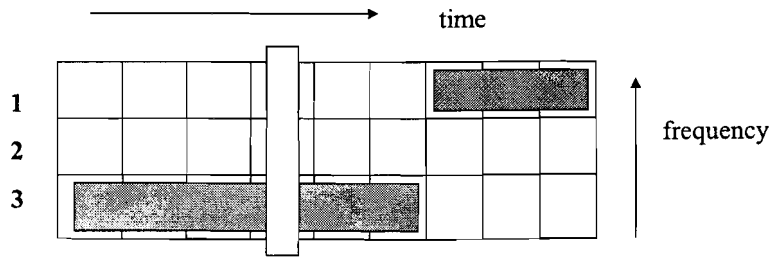
More on synchronization

- Continuous
  - bit-stuffing required
    - no sync in data
  - non-increasing behaviour



- Bursty
  - no stuffing required
  - algorithm for detection
    - gain 10log(N / L) dB

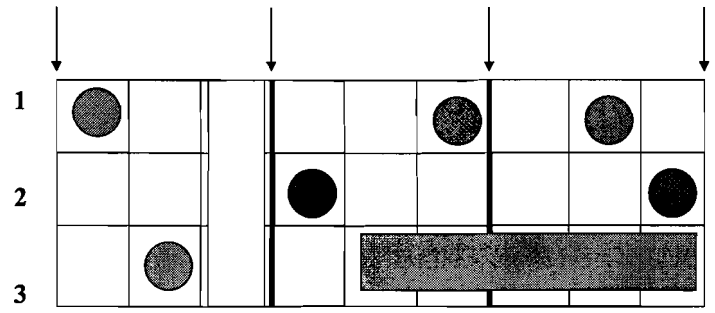
narrow-band and broad-band noise



SOLUTION: **FREQUENCY and TIME DIVISION**

PPM Code Example

- 6 code words: 123 231 312 132 213 321
- distance: = 2



General Results

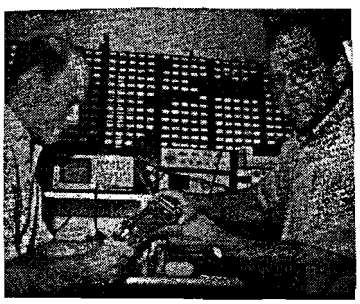
- for coded PPM :
  - distance d correction of d-1 disturbances
  - $$|S| \leq \frac{M!}{(d-1)!}$$
- application to coded OFDM, FH, WDM
- many interesting research topics left
  - code constructions, optimum detection

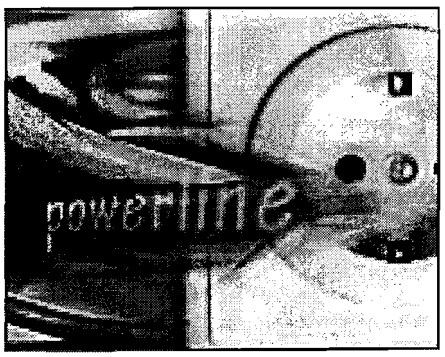
Own recent references:

- Correlation 1 codes
  - "On Optical Orthogonal Codes with Correlation 1," to be published in the *IEICE Transactions on Communications*, January 2002, Japan, with Samwel martirosyan
- Multi-Access
  - "On the Capacity of the Asynchronous T-User M-frequency noiseless Multiple Access Channel," *IEEE Trans. on Information Theory*, pp. 2235-2238, November 1996, with Jeroen Keuning.
- Synchronization
  - "On Synchronization for Burst Transmission," *IEICE, Transactions on Communications*, November 1997, pp. 2130-2135, with A. van Wijngaarden
- Power line and permutation codes
  - Coded Modulation for Power Line Communications, *AEÜ* Journal, 2000, pp. 45-49, Jan 2000.

Our goal is: Communication over 220V

in *HomeNet2Run*





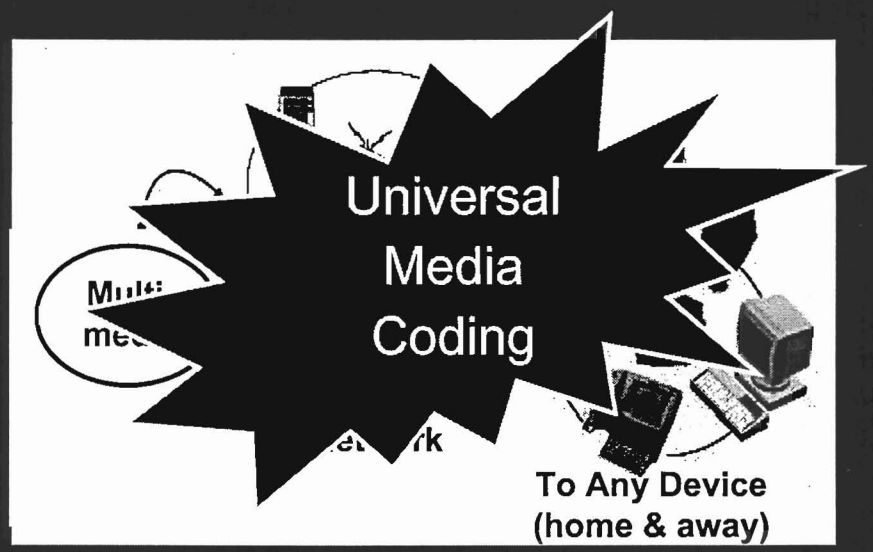
Fine-Granularity-Scalability (FGS) for wireless transmission

Mihaela van der Schaar

Outline

- FGS - general description
- Benefits & short-comings
- Adaptive Motion-Compensated FGS
- Gateway processing (Transcaling)
- Adaptive Modulated FGS
- Outlook

Universal Media Access Paradigm



- Encoding & transmission process should be separated (transmission bit-rates do not need to be specified at encoding-time)
- Decoupling of network and terminals

=> Scalable representation of content is needed for adaptation of terminal & network bandwidth

Scalability in Existent Video Standards

- MPEG-1 - no scalability
- MPEG-2 - data partitioning, temporal, spatial, SNR scalability
- MPEG-4 - most flexible scalability provided: temporal, spatial, SNR, FGS, object scalability, still-image coder (wavelet-based)

- H.261 - no scalability
- H.263+ - temporal, SNR, spatial scalability
- H.26L – temporal scalability (so far)

Conventional Scalable Coding for Transmission over Internet/wireless networks

- Alternative solution to simulcast

- Operate at a discrete set of bit-rates
- Supported bit-rates determined at *Encoding-Time*
- Overhead increases with the number of layers

- Not *ideal* for Packet Networks

An adaptive Solution for Internet Video

- Easy bandwidth adaptability
- Encoding & transmission process should be separated (transmission bit-rates do not need to be specified at encoding-time)
- Packet-loss robustness
- Low server complexity

Possible Solution: Fine-Granularity-Scalability

Fine-Granularity Scalability (FGS)

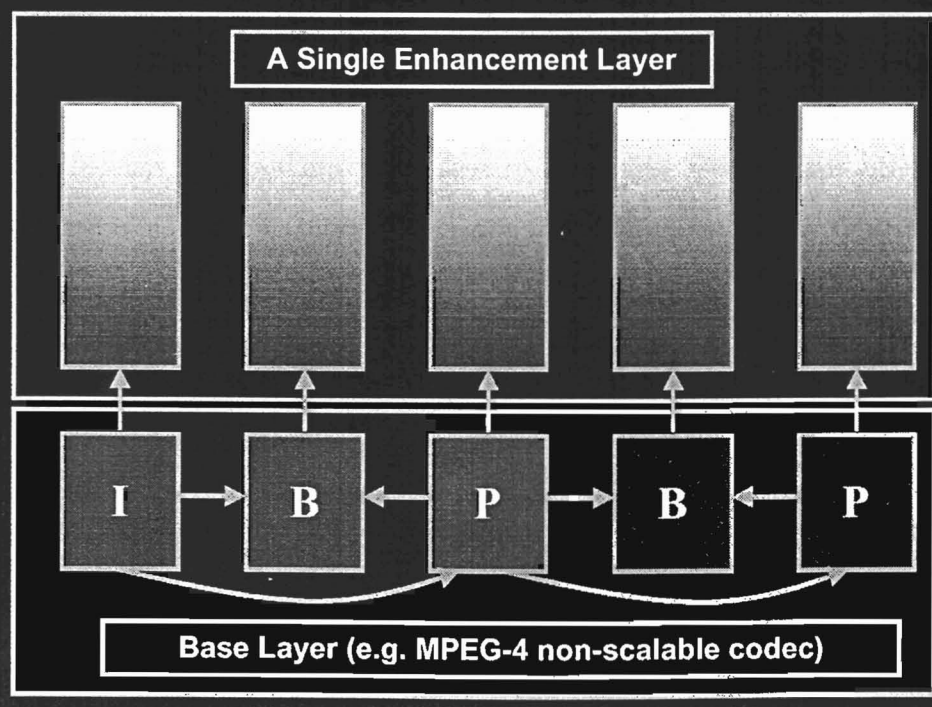
R_{Max}

R_{min}

Philips Research

PHILIPS

Fine-Granularity Scalability (FGS)



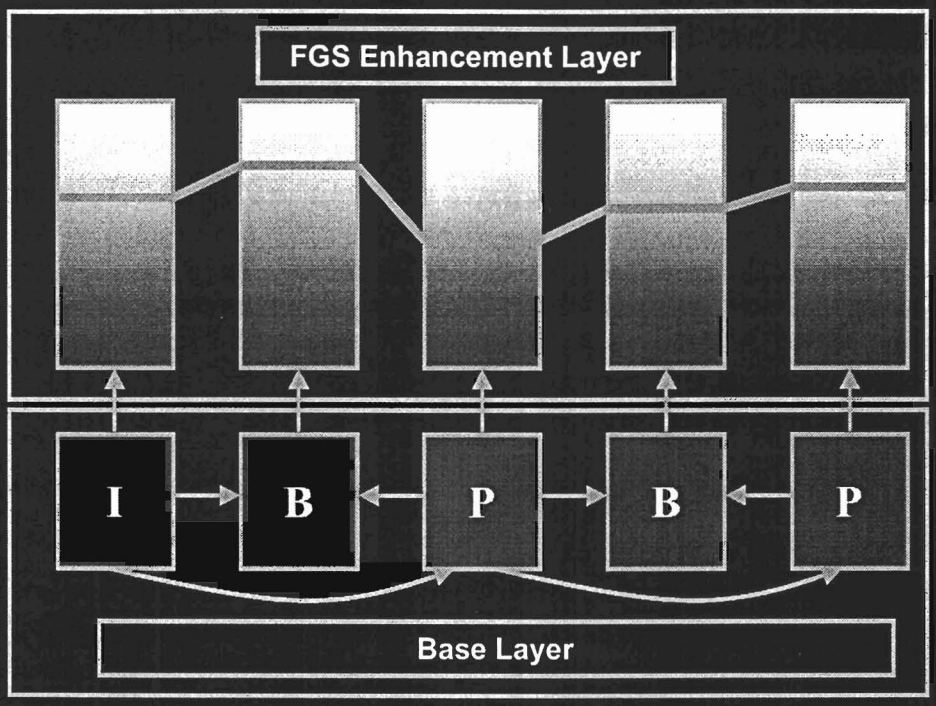
R_{Max}

$R_{BL} < R_{min}$

Philips Research

PHILIPS

Internet Video Streaming with FGS - Server Side



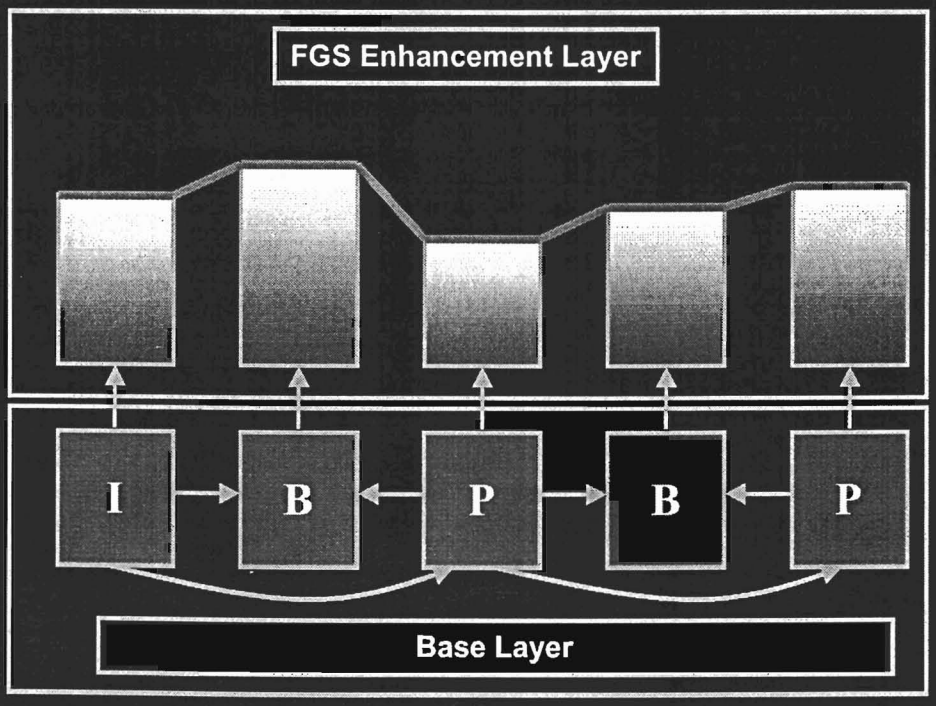
R(t)

FGS EL-portion transmitted in real-time

Philips Research

PHILIPS

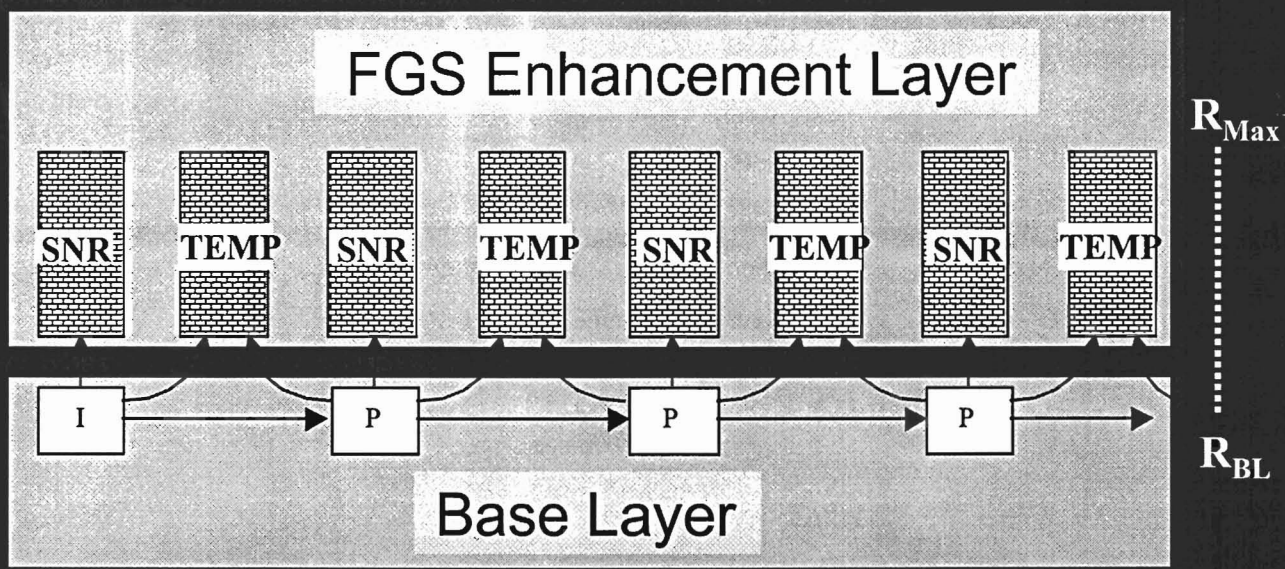
Internet Video Streaming with FGS - Decoder Side



Philips Research

PHILIPS

An all FGS Temporal-SNR Scalability



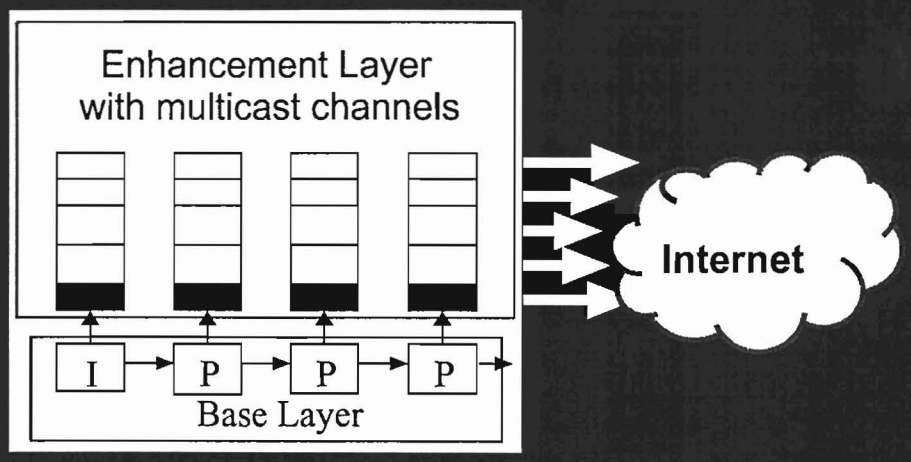
Advantages of the FGS structure

- Fine granular scalable enhancement layer
- Encoding & Transmission processes - separated
- Easy rate-control on multiple-streams for VOD
- Resilience to packet-losses

- Efficient for both unicast & multicast

Multicast scenario using FGS

At the streaming server

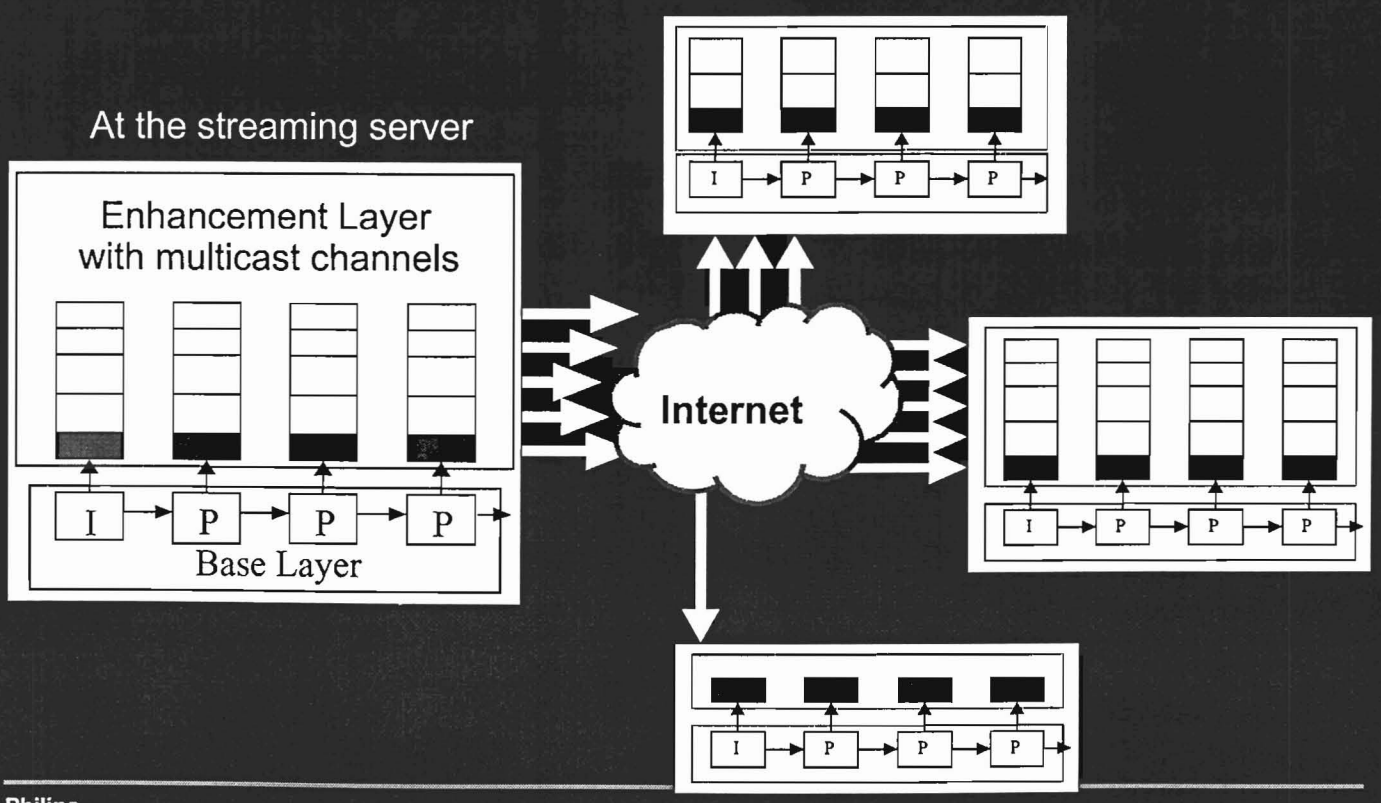


Philips
Research

PHILIPS

Multicast scenario using FGS

At the streaming server



Philips
Research

PHILIPS

Fine-Granular-Scalability (FGS) in MPEG-4

- **FGS - most recently standardized scalable video compression standard**
- July '98 **MPEG-4 approved an FGS core-experiment**
- Dec '98 **FGS Requirements formally established**

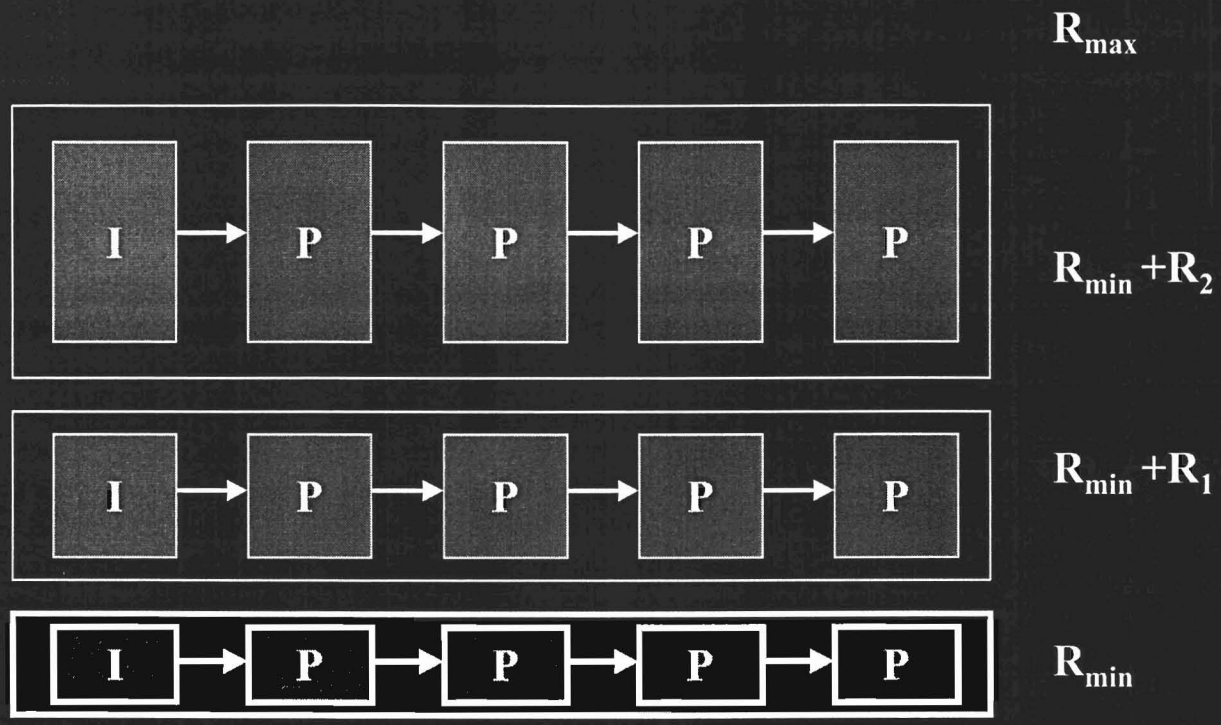
  ... .
- July '00 **FGS reaches FPDAM-status**

  ... .
- March '01 **FGS is an IS** (MPEG-4 v4)

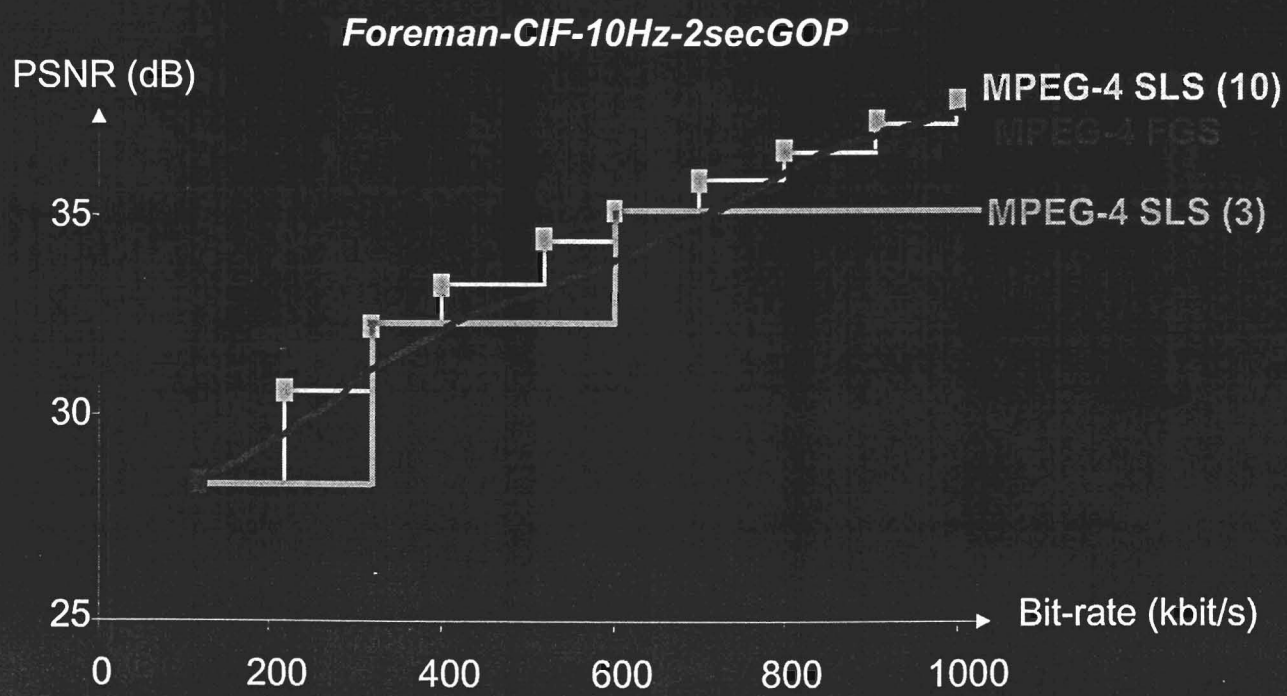
- **AFGS for MPEG-21 video?**

How does FGS compare with alternative streaming solutions?

Single-Layer Switching (SLS)



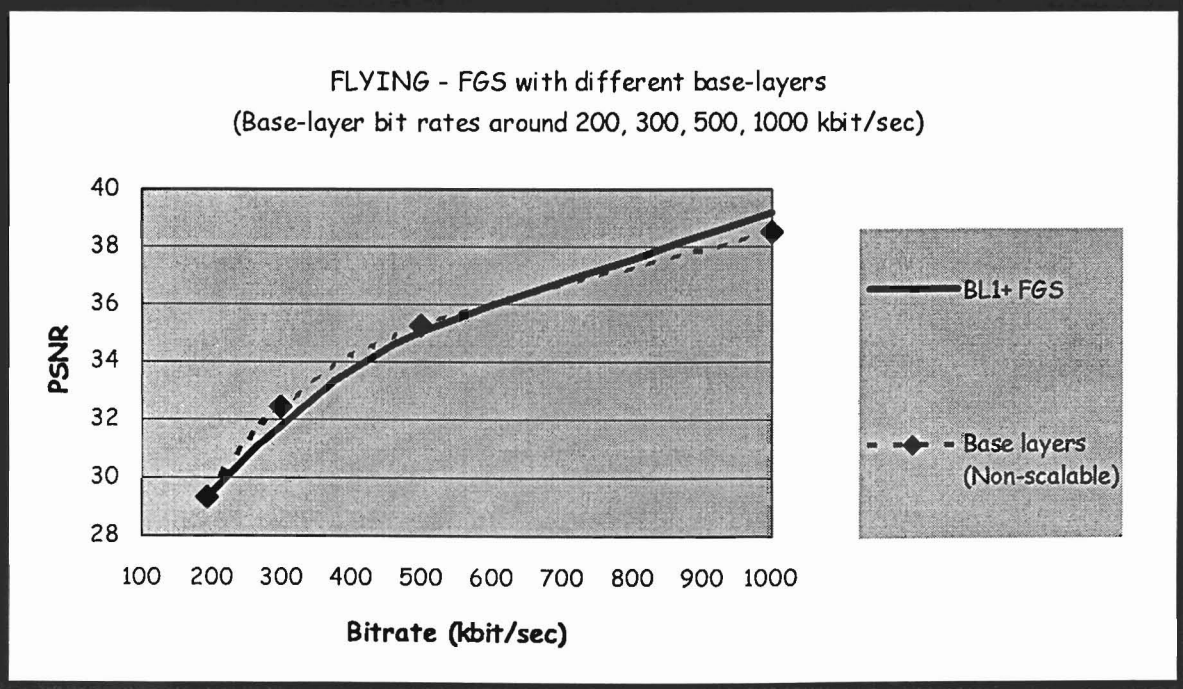
Performance of MPEG-4 FGS vs.SLS under Lossless Conditions



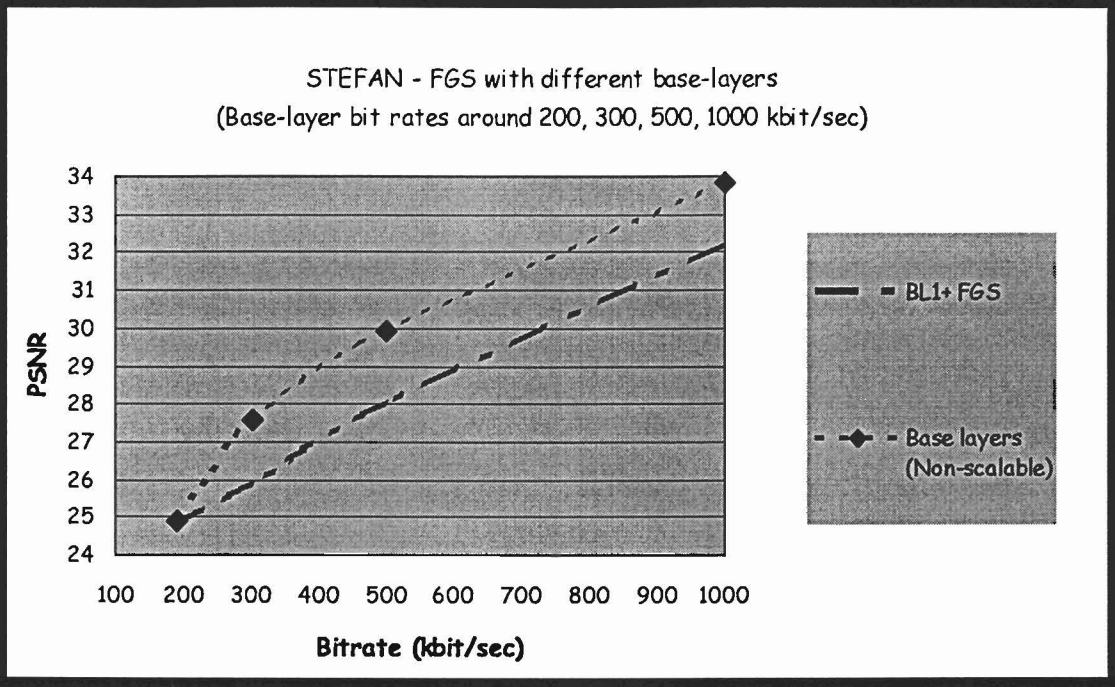
Why does FGS perform worse than SL?

- No MC within the enhancement-layer

FGS vs. SL



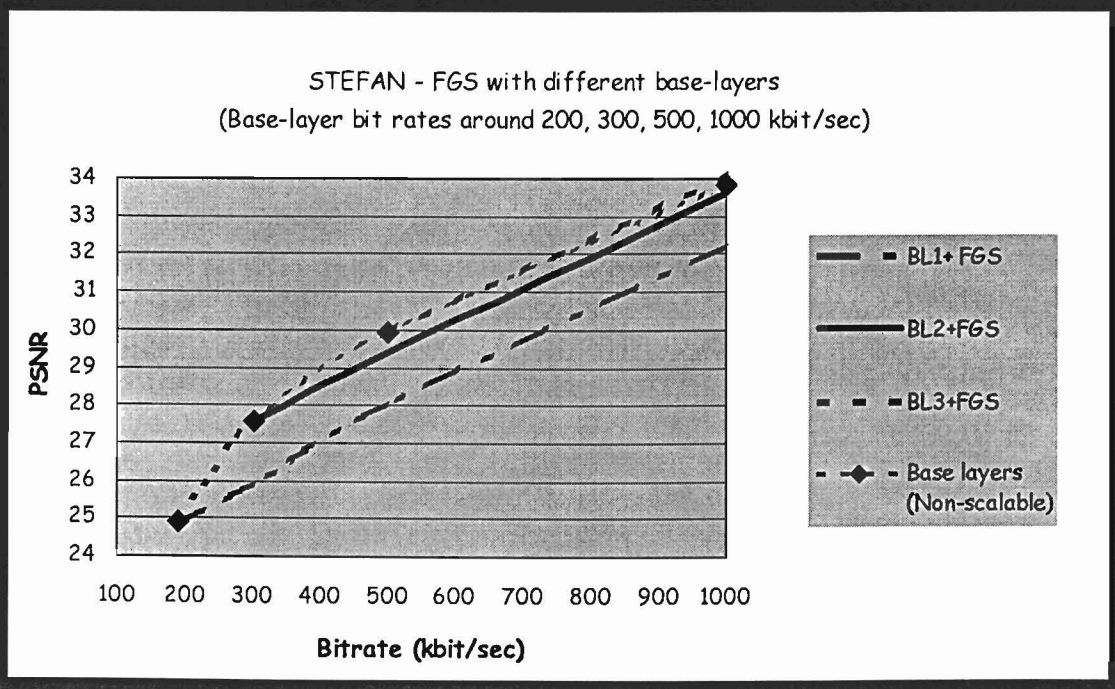
FGS vs. SL



Philips Research

PHILIPS

FGS vs. SL

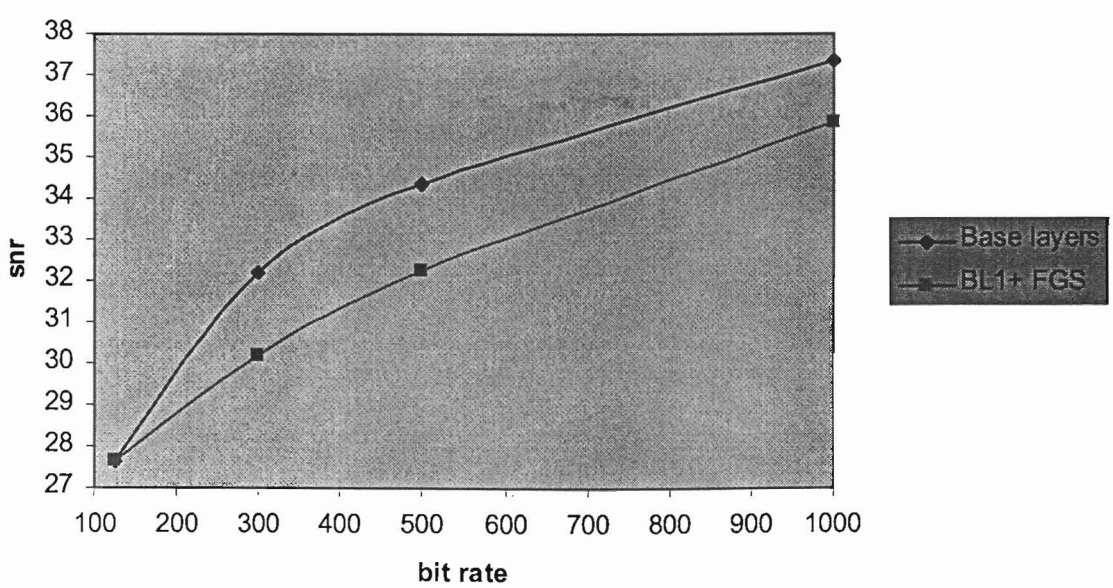


Philips Research

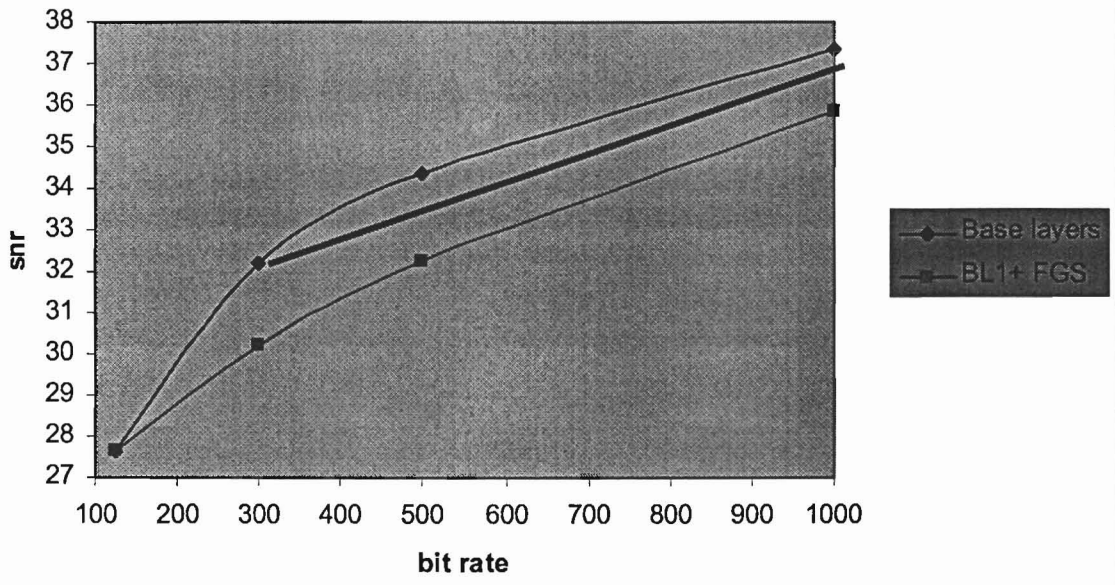
PHILIPS

How to minimize gap between FGS and single-layer coding?

SL vs. FGS



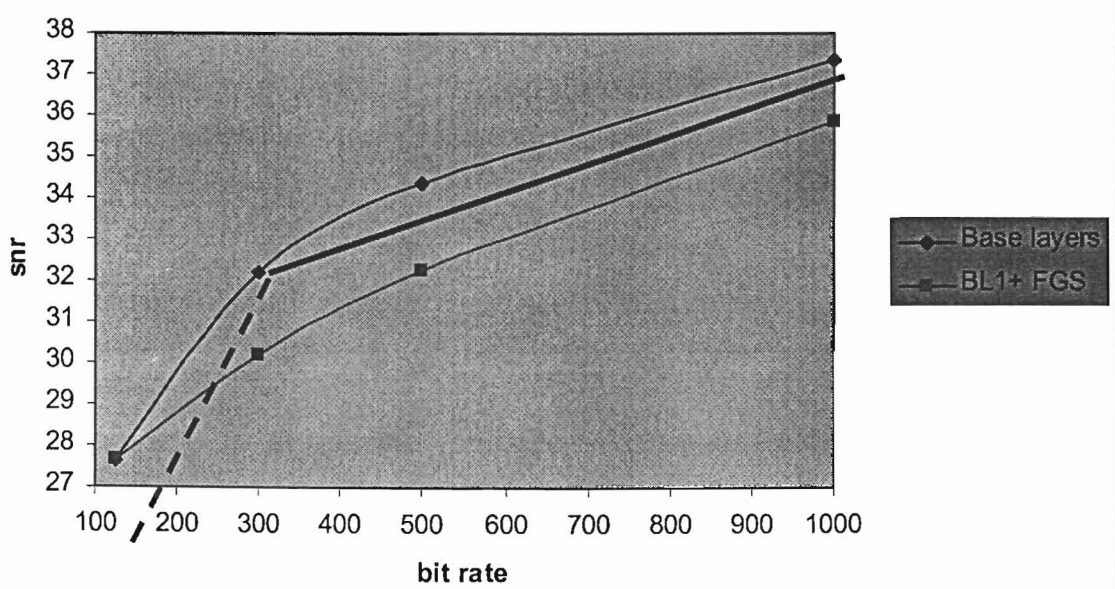
SL vs. FGS



Philips
Research

PHILIPS

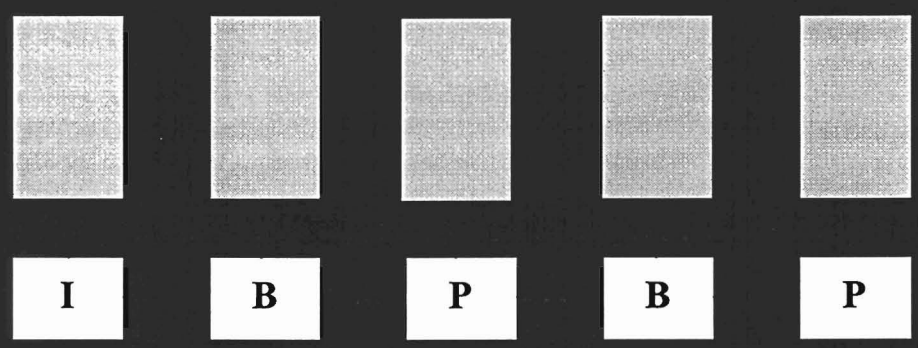
SL vs. FGS



Philips
Research

PHILIPS

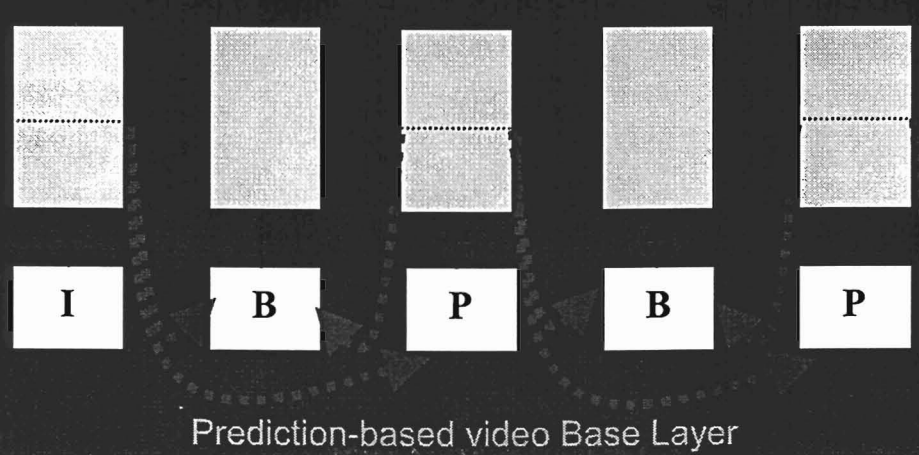
Original FGS Scheme



Philips Research

PHILIPS

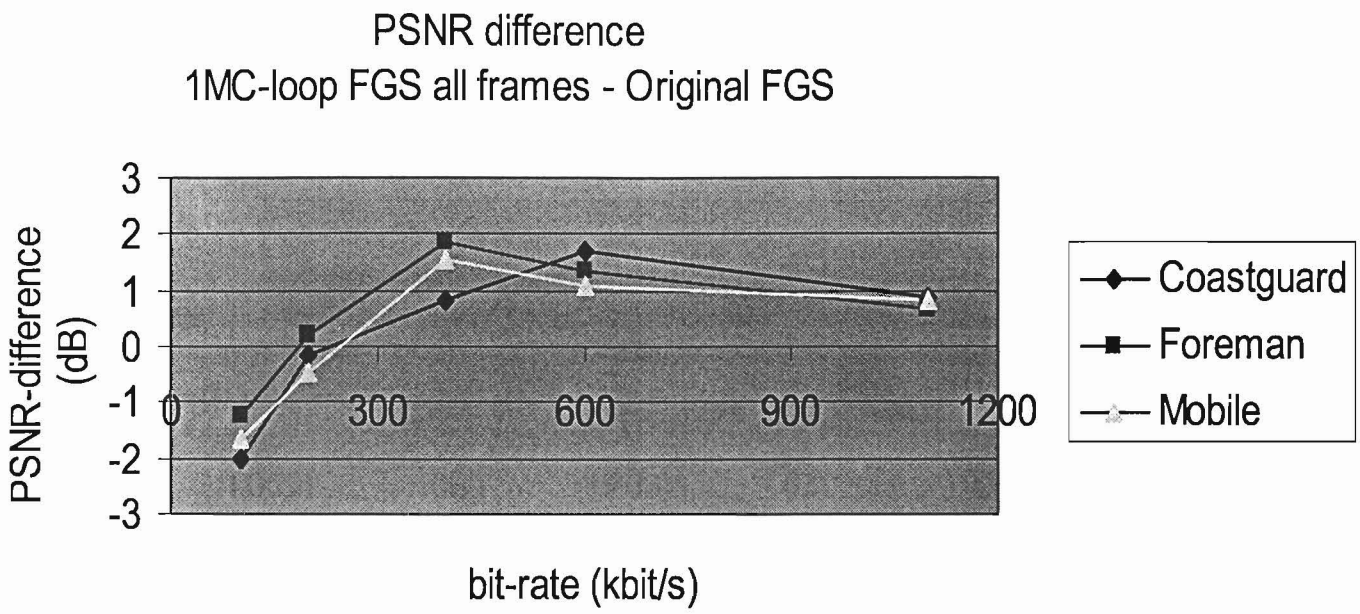
Single-Loop MC-FGS for all frames



Philips Research

PHILIPS

Results

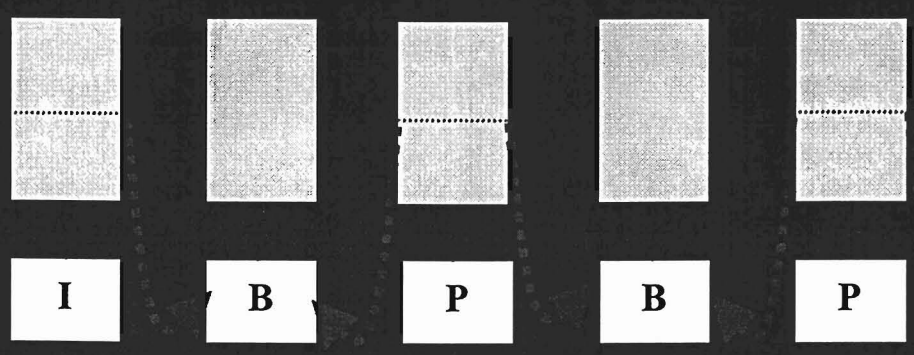


Philips Research

PHILIPS

Single-Loop MC-FGS for B-Frames

Prediction-based Fine granular scalable Enhancement Layer

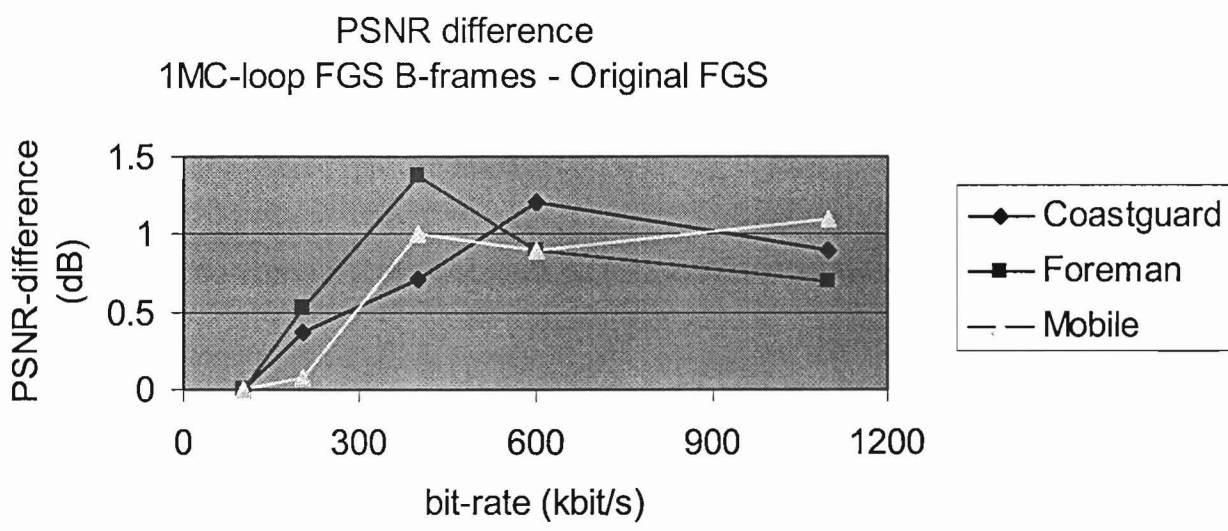


Prediction-based video Base Layer

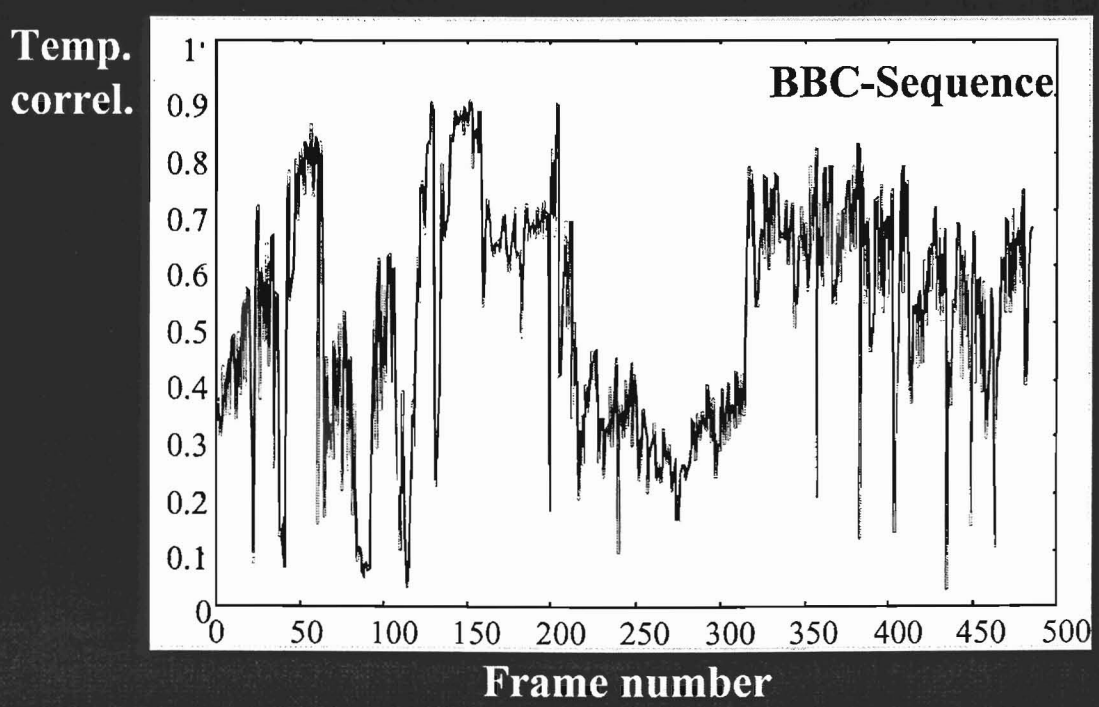
Philips Research

PHILIPS

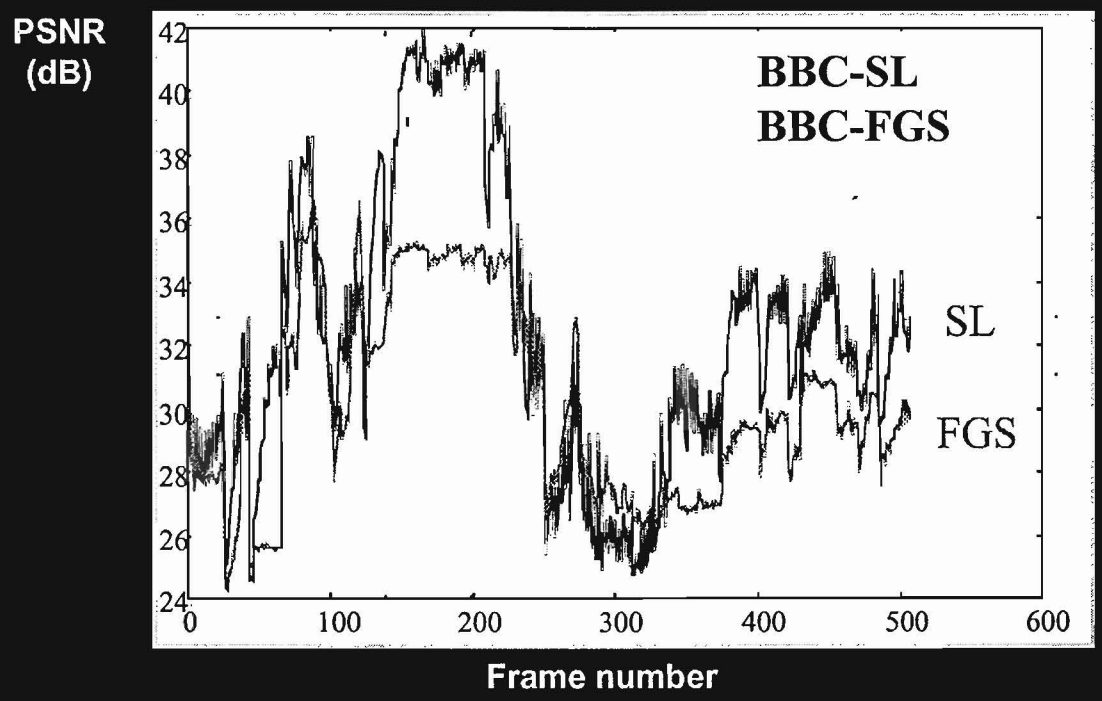
Results



Determination of FGS Quality Gap Based on Sequence Temporal Correlation



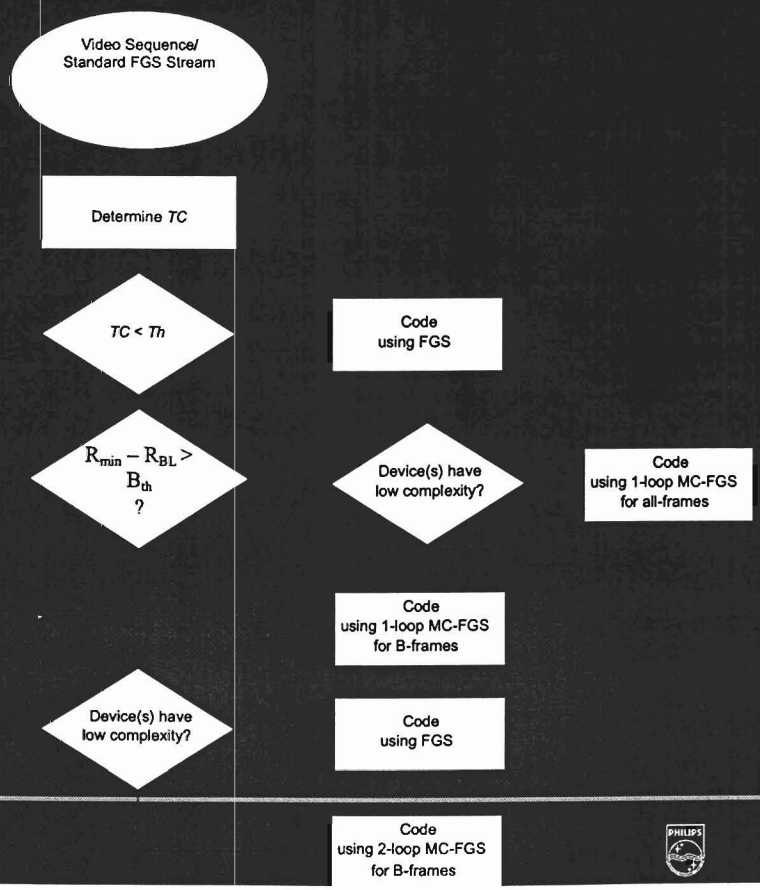
Determination of FGS Quality Gap Based on Sequence Temporal Correlation



Philips Research

PHILIPS

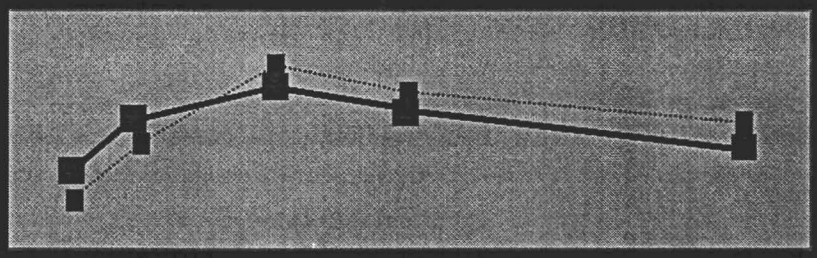
Decision Adaptive MC-FGS



Philips Research

PHILIPS

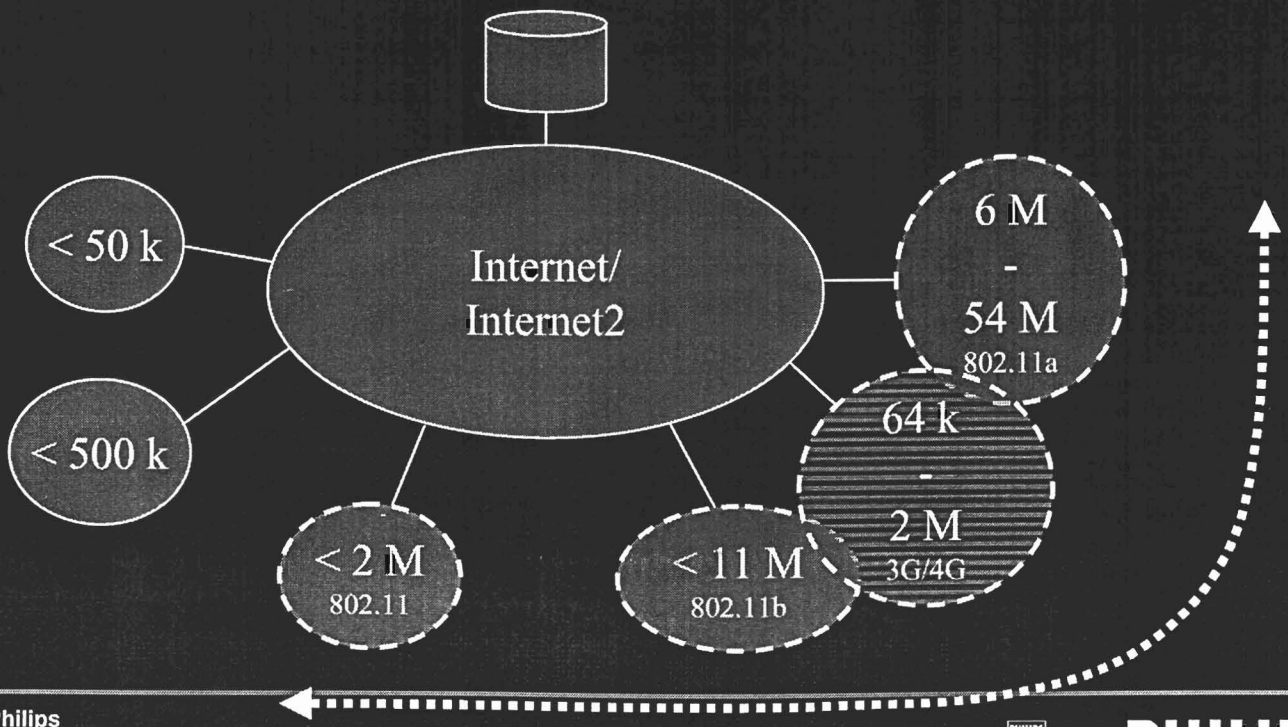
Adaptive MC-FGS



1-loop MC-FGS all frames

Switching between 1-loop MC-FGS all frames and "basic" FGS

Outlook - heterogeneous networks



Outlook - heterogeneous networks

- Convergence of different wireless and mobile networks with the Internet is creating a whole new level of heterogeneity
- This is leading to wider bandwidth variation
- Consequently this emphasizes the need for adaptive and scalable multimedia solutions for the wireless Internet

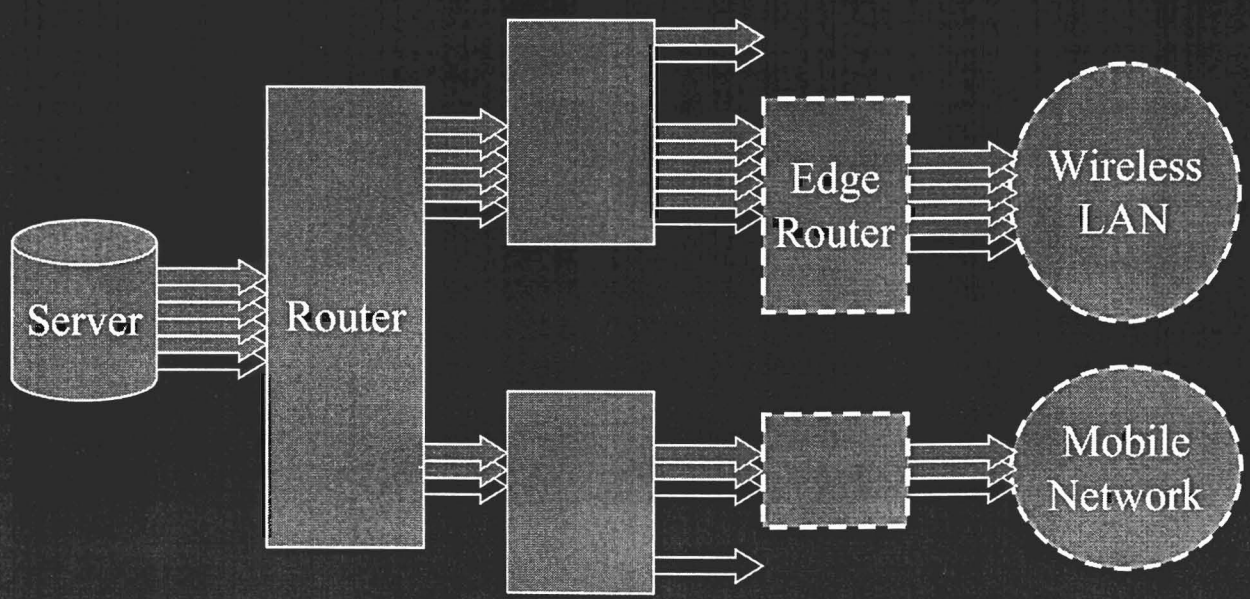
Gateway processing: Motivation

- Higher level of scalable coding leads to lower overall video quality
- Solutions are needed to:
  - Provide highly scalable video
  - Maintain good video quality over the whole bandwidth range needed

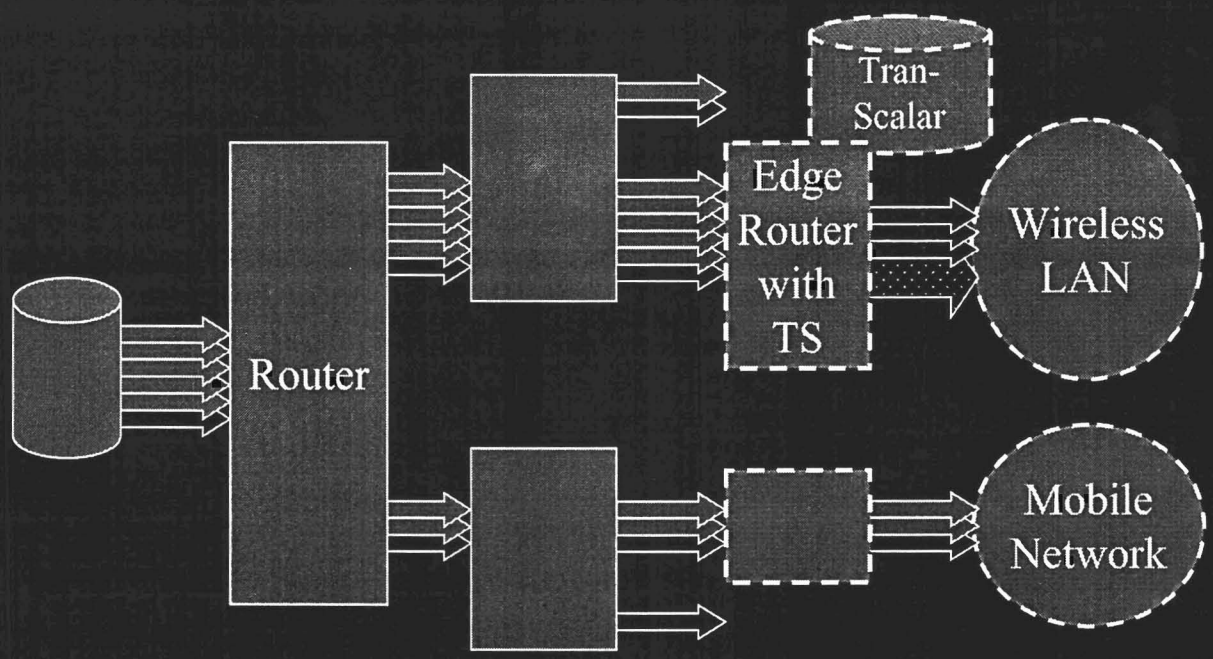
Example of gateway processing: TranScaling

- TranScaling is a mapping of a scalable stream into one or more scalable streams
- TS is a generalization of (non-scalable) transcoding
- TS provides new design options and a level of flexibility that does not exist with non-scalable transcoding

Example: Receiver-driven Layered Multicast (RLM)



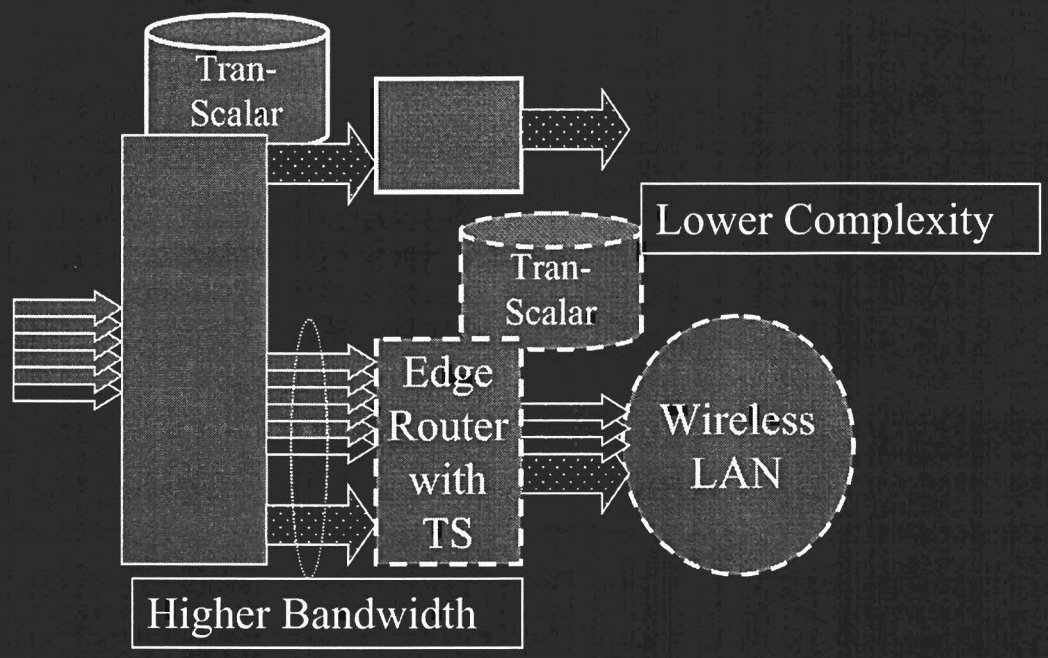
TranScaling based RLM



TranScaling Attributes

- Distributed nature of TS
- The "Fall back" option – Hierarchical TS
- TS with multiple scalable streams
- Classes of TS

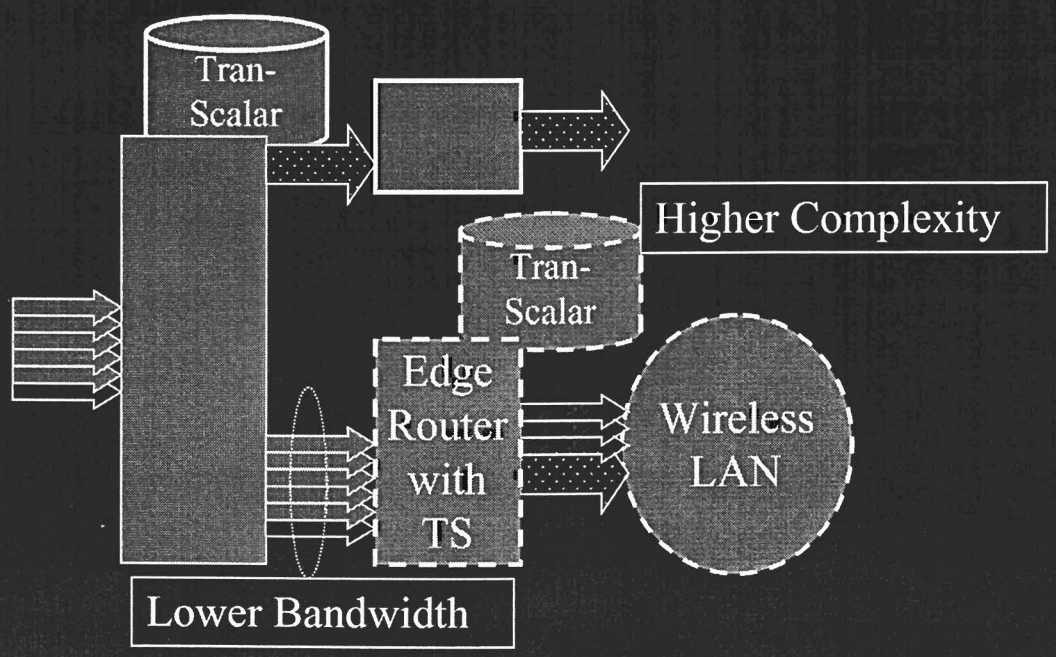
Distributed TranScaling



Philips
Research

PHILIPS

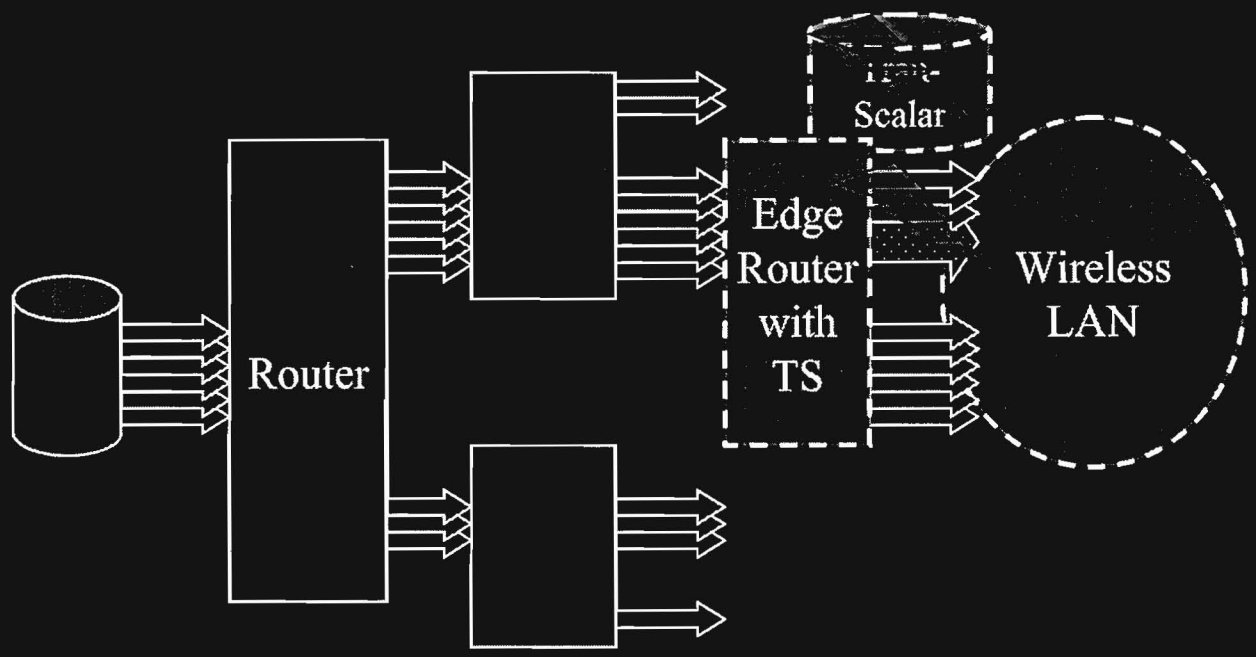
Distributed TranScaling



Philips
Research

PHILIPS

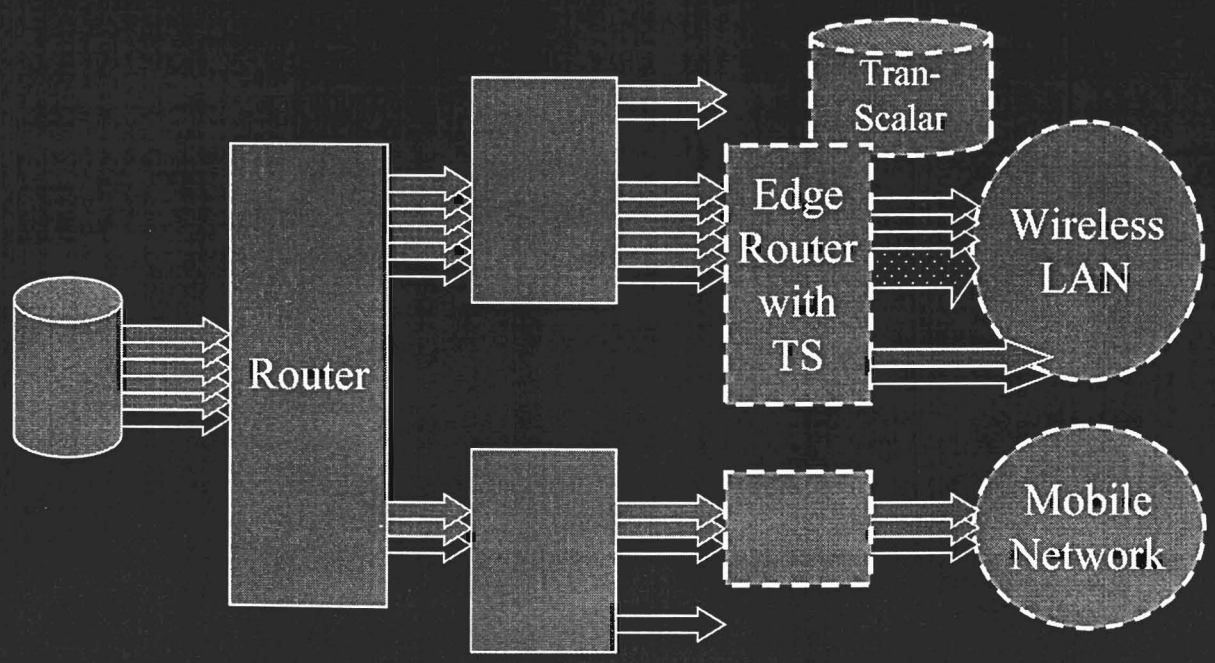
TranScaling with the Fallback Option



Philips
Research

PHILIPS

TranScaling with Multiple Scalable Streams



Philips
Research

PHILIPS

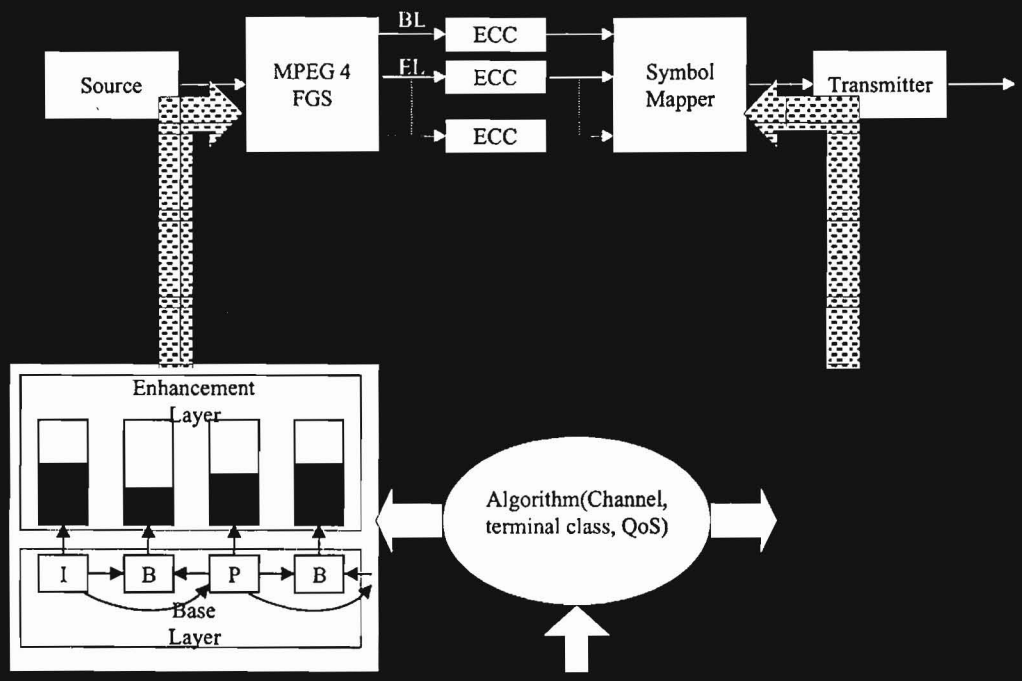
TranScaling: Conclusions

- TranScaling is a mapping among scalable streams
- TS provides a framework for improving content quality while maintaining a high-level of scalability
- TS can be done with a distributed architecture – trading off complexity versus bandwidth
- Initial results with MPEG-4 FGS are promising

Gateway processing: Hierarchically modulated FGS (HFGS)

- Scalable coding + Adaptive Modulation -> Adaptation based on the channel conditions
- Simultaneous Transmission: The base layer data can be mapped to the MSBs of the transmitted signal and the enhanced layers can be mapped to the LSBs. The channel coding on the different layers can be different (i.e., more robust coding on the BL).
- Time Division Multiplexing: The BL could be mapped to a lower level constellation than the enhancement layer (i.e., 4 level QAM for the BL and 16 level QAM for the EL). The channel coding again would be different for each layer.

Graceful Wireless Video Transmission using HFGS



Hierarchically modulated FGS - Results

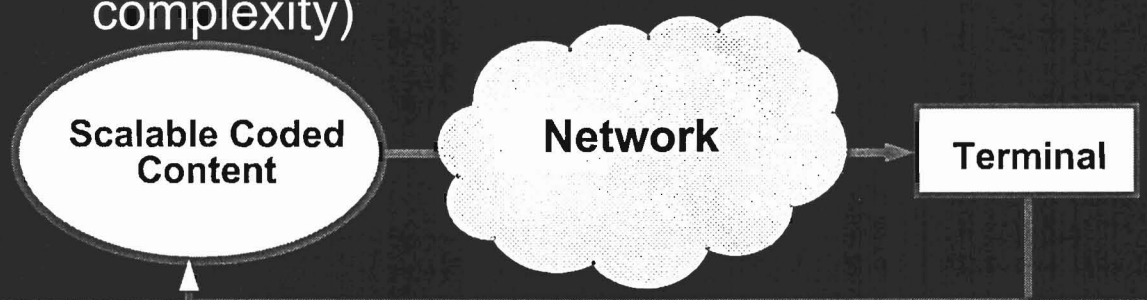
Best joint source-channel coding trade-off to be found

Bit-rates (mbps) 1.5	PLR (18db)	18 dB	25 dB
1.5	10-4	34.75, 30.90	34.80, 30.95
1.5+3	10-4,10-3	37.94, 33.04	38.25, 33.26
1.5+9	10-4,3*10-1	35.58, 31.30	39.36,33.88

Quality improvement

Outlook

- Universal Media Access paradigm:
  code once and then access content
  - "Anytime"
  - from "Anywhere" (i.e. using any access network - wireless, internet etc.)
  - and by "Anyone" (i.e. with any terminal complexity)

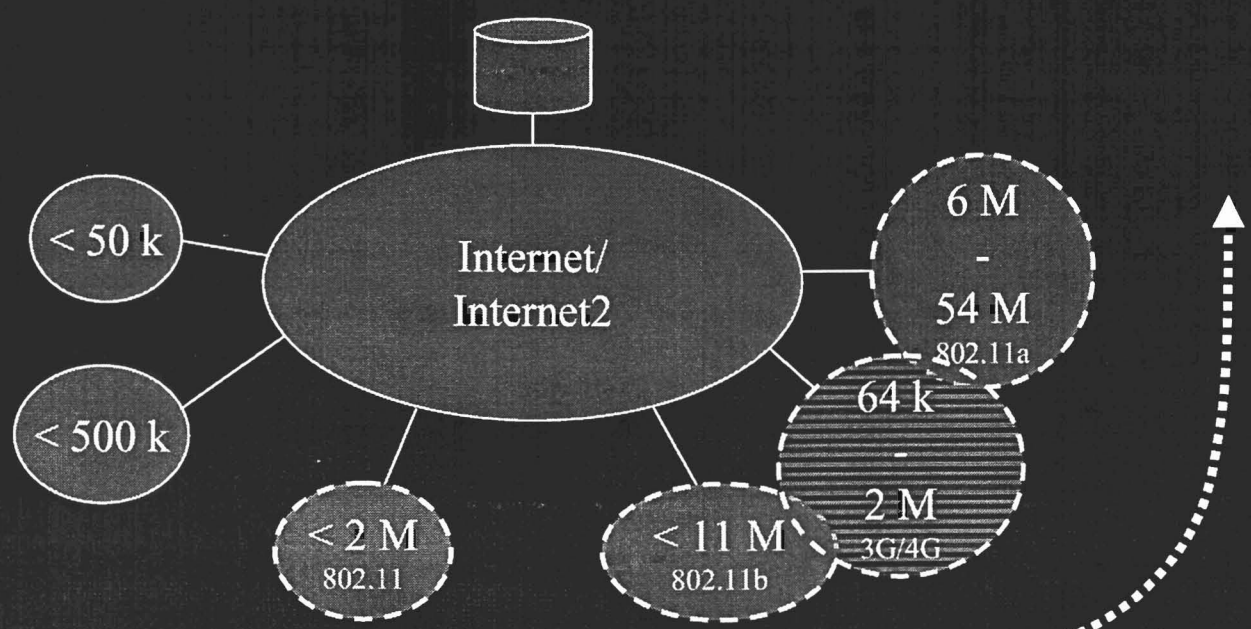


Philips Research

PHILIPS

Outlook

Scalable representation of content is desirable for heterogeneus networks and different-capability terminals



Philips Research

PHILIPS

Outlook

- Development of efficient scalable technology
  - is one research frontier of video coding
  - Constraint : Scalability shall not cost much in terms of data rate and complexity
  - Popular video coding techniques like MC prediction bear inherent problems, mainly due to drift
- What is required ?
  - Flexible switching between all scalable modes – e.g. spatial resolution, temporal resolution, SNR, localization for content
  - Scalability of rate and complexity over wide range
  - Efficient embedded scalability of spatial and temporal resolution is key issue to achieve this

Outlook – Prospective Technologies

- Solutions to the drift problem in MC prediction
  - Limitation of drift within the loop
  - Optimum tradeoff between drift and coding efficiency
  - More efficient solutions for multiple-loop encoders
- Quantization and entropy coding
  - Bitplane approach has certain deficiency due to integer precision of bit planes
  - Possible solutions to achieve "fractional" bit planes
    – Development of relevance criteria for sequel of encoding
    – Vector quantization
  - Scalable entropy coding
- Wavelet coding!

Internet Streaming
Transporting Continuous Media

Warner ten Kate

ABSTRACT

In this document we summarize the main problems encountered in implementing streaming capabilities in the Internet. We look at the problem from the perspective of the end-to-end transport. We discuss the functions needed to support streaming transport, where we assume the internet's best-effort service. This print contains part of the complete paper.

Table of Contents

Data & Streaming - The Problem

> *"At the March, 1992 meeting of the Internet Engineering Task Force (IETF) in San Diego, live audio from several sessions of the meeting was 'audiocast' using multicast packet transmission from the IETF site over the Internet to participants at 20 sites on three continents spanning 16 time zones. This experiment was not only the first sizeable audio multicast over a packet network, but also significant for the size of the IP multicast network topology itself. [..snip..] Unlike listening to a radio broadcast, the remote participants could also talk back." [Audiocast].*

The Internet has been designed for ***data*** traffic. Data traffic is characterized by flowing in ***bursts*** and being satisfied with ***elastic*** transport, i.e. the data is submitted in irregular patterns and it may be transmitted at varying speeds. In each burst a certain amount of information is transmitted and the average throughput at which the entire information is transmitted is the figure of merit.

The delivery of audio, or, more generally, the transmission of content for media presentations, extends the Internet design to support for ***streaming*** traffic. The streams are characterized by ***flowing*** with a fixed time profile and requiring ***real-time*** throughput guarantees such as instantaneous forwarding capacity. They require the network to preserve ***time integrity***.

The Internet is a ***packet switching*** network. In packet switching the network resources (buffers, routers, link capacity) are shared dynamically. This provides for greater speed and flexibility in setting up connections, and for a more efficient use of the resources after the connections have been established. Hosts can be "always on" the network. Packet switching also facilitates the interconnection of networks with different architectures through relying on the minimal service abilities of the networks. It provides, however, little control over the packet delay at the switches. If the user demands exceed the network capacity, ***congestion*** can occur if no measures are taken to control the flow.

The opposing approach is that of a ***circuit switching*** network. In circuit switching the network resources are allocated to a user for the duration of a session. Hosts connect to the network after passing admission control. Circuit switching eases the accommodation for QoS provisioning and control, but reduces the flexibility in connecting over various network architectures and in adapting for varying resource allocations.

In order to support for streaming traffic three major types of functions have to be added to the traditional Internet:

1. **Transport for streams.**
   A packetizing scheme is needed to enable the transport by IP. The main functions are to provide for packet delivery, keeping track of their order, and to enable for synchronization of these packets, i.e. to keep track of their temporal order. This includes the control of the reliability of the delivery, the timely delivery and the adaptation of the delivery rate to the available bandwidth. Secondary functions concern the identification of the session, like administration of media source and destination, media type and media encoding. *(Note that the packet flow doesn't need to be necessarily a one-to-one correspondence with the media representation stream. For example, the instantaneous packet rate may differ from the instantaneous coding rate.)*
2. **QoS guarantees.**
   Because of the real-time characteristics the transport needs guarantees on the timely delivery of the packets and the capacity available for transport. A consequence of providing such guarantees is that additional functions for admission control and resource reservation are to be added to the network. *(An alternative approach is to ensure the network is overprovisioned, such that resources are always available.)*
3. **Management & Control.**
   The additional functions require ways to manage and control them. MIB schemes are needed to store and interchange the configured settings. Policy signaling and enforcing schemes are needed to configure and provision for QoS settings. Last, but not least, protocols are needed to announce and control streaming sessions. The session set-up information needs to be announced, as well as ways are needed to control the course of a streaming delivery (play, pause, continue and stop).

In adding these functions to the Internet it is ***the challenge and objective*** to keep the original Internet design philosophy [DesignPhil]. The ***datagram***-based ***best-effort*** transmission reflects the core of this philosophy. Important characteristics are the absence of state information in the network, and the push of complexity towards, or into, the ***end nodes*** [EndArg]. Noteworthy, this concerns the control of the transfer. The ***connectionless*** characteristic of datagram-based transmission is an important component in providing robustness, as well as in providing link layer independence, the so-called ***minimal link assumption*** [DesignPhil].

An important operational aspect concerns the notion of ***fairness***. Fairness requires that all users of a resource get a same share of that resource. For data traffic, which operates under elastic transport, TCP implements a distributed protocol for fair utilization of the network resources. The introduction of different transport types requires a generalization of that concept. Commonly, TCP behavior is used as the yardstick to measure fairness. Traffic that does conform to the TCP behavior is called *TCP-friendly*. (More strictly, it suffices that the other, TCP-friendly, traffic does not push out the TCP flows.)

Two other classes of conditions to be observed concern matters like the *scalability* and distributed operation and the support for secure and accountable operation. In particular scalability and security apply not only to the data plane but also to the control plane of a solution.

In the following sections we briefly summarize some of the fundamentals of the Internet design. In the subsequent sections we discuss the major solutions for providing streaming transport and Quality of Service as being developed in the Internet community.

The Internet

The Internet - Its Basic Design Philosophy

The Internet is a **network of networks** [InternetModel]. It connects networks, yielding global connectivity such that applications running on one ***host*** that is connected to one local network can communicate with applications running on another host that is connected to another local network. The different networks are connected to each other through ***relays***. The local networks may implement a different communication technology, providing different types of service. The name "Internet" (capital "I") stems from the term ***internetworking***, that is used to indicate the connection between two or more networks [CerfKahn]. Figure 1. depicts the concept of internetworking.

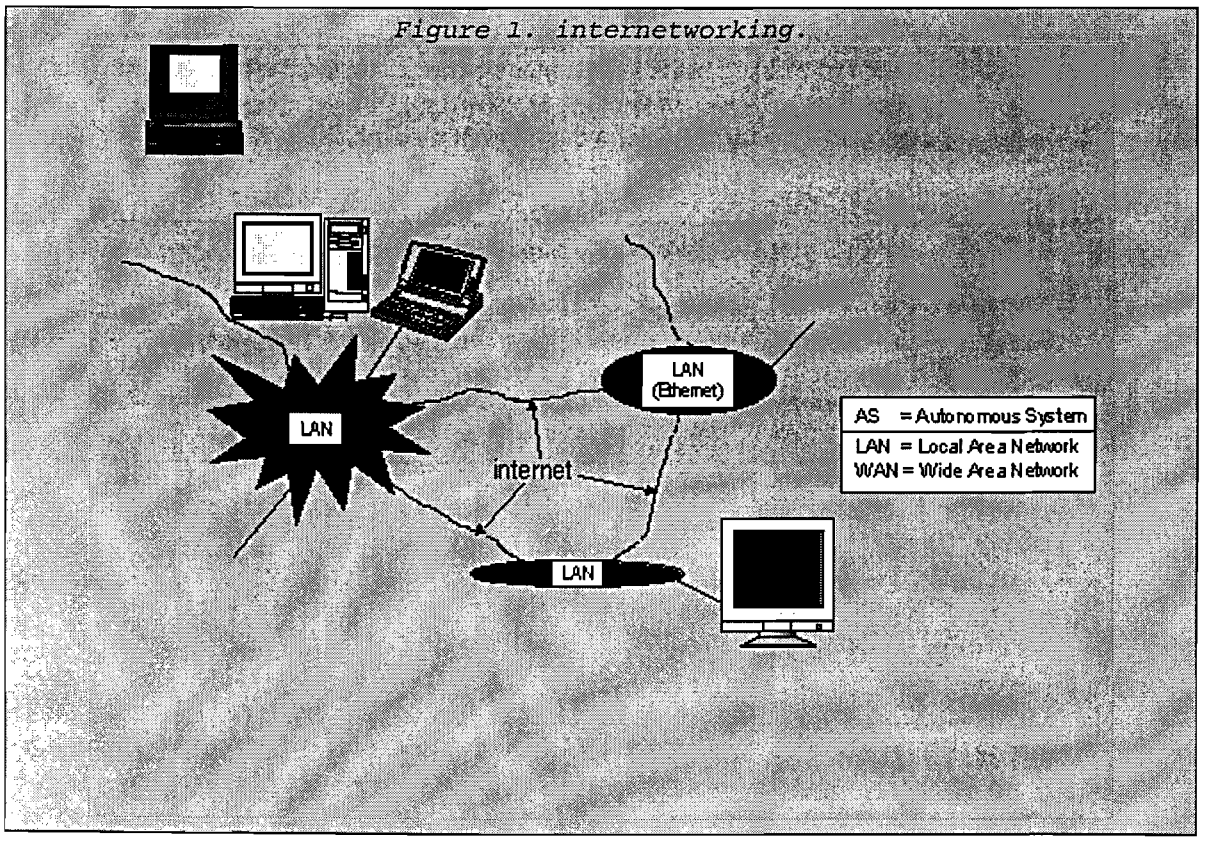


Figure 1. internetworking.

There exist several Local Area Networks, ***LAN***, which are connected through a Wide Area Network, ***WAN***, that is performing the internetworking. In the Internet the WAN typically performs the function of ***backbone***, and the LAN is a LAN indeed, like an Ethernet network, or is an access network, like a PPP telephone or cable link.

Each of these subnets is called an ***Autonomous System*** (AS). They associate with ***two important basic characteristics***. The first is that each of these networks is administrated and managed by a different, individual party, leading to different policies in operating the networks. The second is that each of them may operate a different communication technology and protocol, providing different types and levels of service. Examples of an Autonomous System are a university campus network, a backbone, an ISP's network, and an Intranet.

Given this environment of autonomous and heterogenous networks, expressed through these two basic

characteristics, the Internet was designed to connect them, such as to provide [DesignPhII]

> *An effective technique for multiplexed utilization of interconnected networks.*

This has conducted to the principle of a ***datagram*** based network design.

Interconnected

The networks to be interconnected are heterogeneous. A ***minimum link service assumption*** has to be made, such that the interconnection protocol runs over *any network*. Any link service like QoS or connection-oriented support cannot be relied upon. The datagram is the largest common unit that any link connection can transport. The datagrams are transported through a store & forward protocol operating at a layer on top of the local network protocol.

Multiplexed

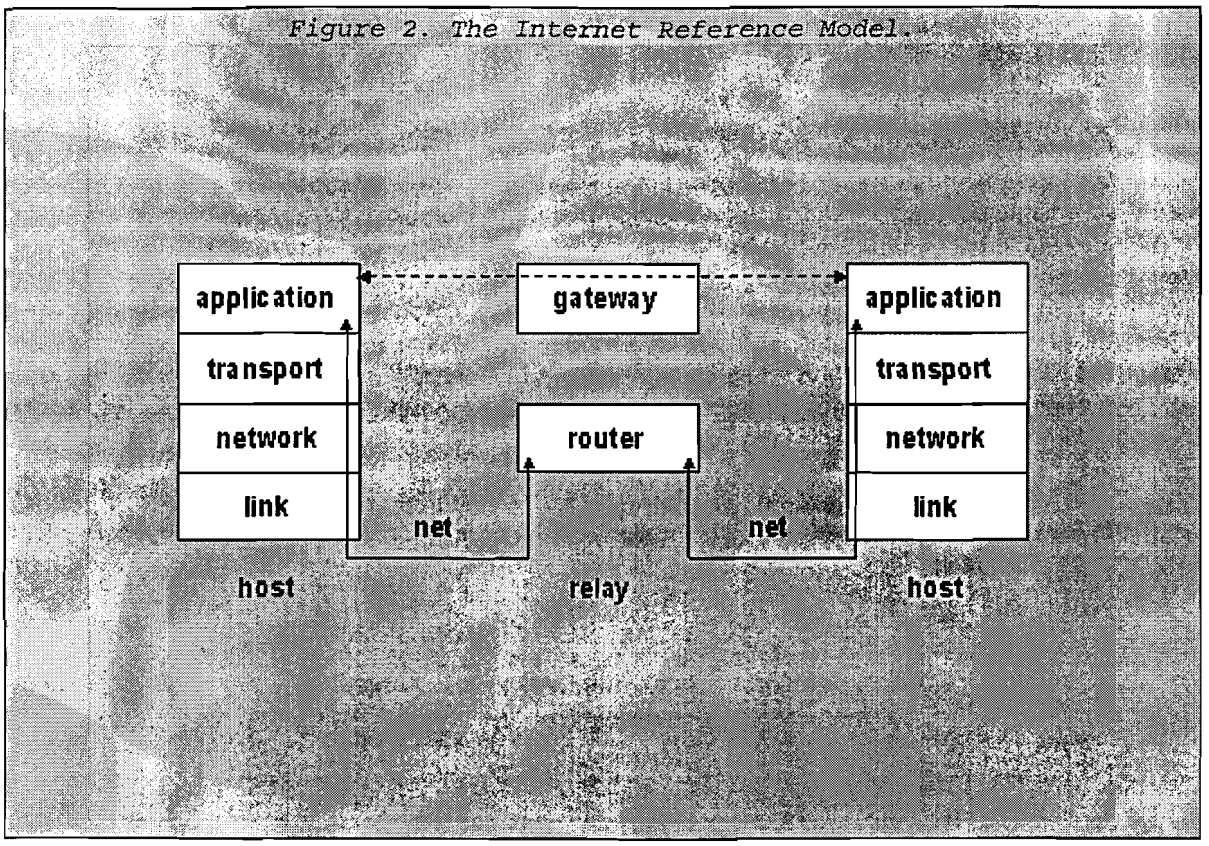
A packet switched network design enables for the multiplexing. The alternative would have been to adopt the circuit-switched approach as known from telephony communication services. The use of packet switching, however, is more natural to bursty accessing, "always on" type of applications.

Effective

Robustness is a fundamental requirement for the network, a.o. because of its (highly expanded) distributed character of ownership and management. The transport must provide good reliability against node and link failure. The network must support multiple types of communication service. The datagram is a basic transport building block to all service types, including the request-reply transaction.

These goals have lead to the layered protocol stack for the Internet as depicted in Figure 2 [CerfKahn, InternetModel].

> *In their paper Vinton Cerf and Robert Kahn address the internetworking concept, i.e. the communication over heterogeneous (packet switching) networks [CerfKahn]. They present the architecture of network layer and transport layer, by proposing (the first form of) the TCP/IP protocol pair. In fact, their concept that the Internet is a network of networks builds on the assumptions that the "networks" are heterogeneous packet networks and that the "network" provides a connectionless datagram service at the internet layer [InternetModel].*



The ***link layer*** represents the subnets, both local and backbone. As they belong to the various Autonomous Systems they are heterogeneous and they are only required to be able to forward (single) datagrams.

The ***network layer*** performs the forwarding of the datagrams, routing them to their destination host. It assumes a single (global) logical address space [RFC2775]. The forwarding is a best-effort delivery: the order of delivery or even the successful delivery of the packets is not guaranteed. The protocol runs on top of the link layer protocols, i.e. it is using their (minimal) service but is staying independent of them by not integrating with them. It is the famous ***Internet Protocol*** (IP) that operates at this layer [IP]. IP is essentially designed to connect over multiple networks.

The ***transport layer*** turns the best-effort datagram service of the network layer into the end-to-end transport service that the communication application is requiring. The protocol runs essentially at the end hosts; the network's protocol communications are not part in this. As RFC 1958 state it [RFC1958]:

> *The goal is connectivity, the tool is the Internet Protocol, and the intelligence is end to end rather than hidden in the network.*

It is the ***Transmission Control Protocol*** (TCP) that mainly operates at this layer [TCP]. TCP segments the data to be transmitted into IP packets and numbers the bytes. Using these numbers the receiver of a TCP connection reorders the received packets, in case IP has delivered them out of order. It further sends an ***ACK*** packet (acknowledgement) back to the sender per received packet of which the preceding packets have also been received. The TCP sender uses these ACKs to determine which packets need to be resend in order to repair for losses, and by that to turn the IP best-effort delivery into a reliable end-to-end transport. A packet is retransmitted after a timeout period having been expired in which an expected ACK has not been received, or when its previous packet has been ACK-ed three times. The timeout period is based upon an estimate of the ***Round Trip Time*** (RTT). Duplicate ACKs indicate that packets succeeding the missing one do have been received, which in turn is an indication that there is no further congestion in the transport channel. *(Other strategies to inform the sender on missing packets is by sending NAKs, Negative ACKs, or SACKs, Selective ACKs.)*

TCP also uses the ACKs to determine its transmission rate. This latter follows as the number of outstanding packets that need acknowledgement, the ***window***, relative to RTT. The window is increased upon successful

transmissions, while lost packets reduce it again. In this way TCP adapts its transmission rate to the available bandwidth, as is explained in the section on congestion control.

Communication applications that cannot build on TCP, can make use of the ***User Datagram Protocol*** (UDP) that is a providing the application with a minimal transport layer interface, viz. the datagram service [UDP]. Multicast and streaming are typical applications in this class. A more traditional example is the request-reply type of application, although T/TCP has been designed for those purposes. *(Also HTTP, which is of the request-reply type, has been built on top of TCP.)*

The ***application layer*** provides the communication service tailored to the particular application (domain). The traditional examples are file transfer (FTP), email (SMTP), and remote login (Telnet). Since the Web's invocation HTTP has been added. RTP is the champion in the streaming application domain. Less known is the range of control and management protocols which make use of the services offered by TCP to exchange their control commands and data.

It is clear that the network and transport layer form the core of the Internet protocol stack. IP/TCP forms the base of the Internet's transport system. Table 1. compares the network layer and transport layer in their complementary characteristics.

TABLE 1: THE COMPLEMENTARY CHARACTERISTICS OF NETWORK AND TRANSPORT LAYER OF THE INTERNET'S IP/TCP TRANSPORT SYSTEM.

network layer:	*transport layer:*
IP	TCP
hub-to-hub	end-to-end
best effort (datagram forwarding)	reliable, fair (ARQ, throughput control)
connectionless (addressing, routing)	connection-oriented (session, order)
no state in network (a router is unaware of the transport connections passing through it)	state contained in end hosts (end-hosts are unaware of the path used by their packets)

Other principles that have guided the design of the Internet are the ***end-to-end arguments*** [EndArg], and, more recently, the ***application level framing*** and ***integrated layer processing*** [ALF].

The end-to-end arguments conclude that any functionality that treats an effect that can (also) occur at an using function, is better implemented at that using layer. This may improve performance and efficiency. For example, receiving and sending hosts need to verify the reliable transport of their data, and thus reliability at the link level does not contribute. *(This example needs some nuance: a link layer is assumed not to corrupt the content of the packets it does transport.)* More generally stated, a service should only be implemented in the network if the network can provide the full service, and if the service is useful for all the applications. *(As the example above indicates, the end-to-end arguments are not intended as an absolute. There are functions that can only be implemented in the core of the network, such as policing against selfish user behavior, and issues of efficiency and performance that may motivate core-located features, such as assuring some form of QoS.)*

ALF and ILP are design principles on which Internet streaming is based. The principles derive from the conclusion that for certain applications it is better to leave some lower layer decisions to the upper layer, since the decisions are strongly dependent on the type of application. In case of streaming applications, for example, it is better to leave decisions on loss and out-of-order reception, which are traditionally associated with transport layer functionality, to the application layer process. (The function may still be *executed* at the transport layer, however.) The performance optimization for a human-human communication will emphasis low delay, while in a one-way broadcast reception delay can be traded for reliability and quality enhancements. The ALF principle states that it is better to packetize the transmission segments along application-related boundaries, and to let each packet bear an absolute address in the total data space. It is the application layer that performs the segmentation, rather then the transport layer. This allows for

independent processing of the packets, including error concealment measures and moving the received data into the storage/presentation space. The ILP principle combines the processing steps from the stacked protocol layers, yielding significant gain in (end-to-end) efficiency. For example, while packetizing an audio frame, the addition of each protocol layer's header in separate processing steps introduces latency related to the subsequent copying of the data. In ILP this copying is saved by adding the headers in an integrated manner. Moreover, a function like retransmission is combined over the (traditional) application and transport layer.

THE END-TO-END ARGUMENTS.

Citing [EndArg]:

> *It is fashionable these days to talk about layered communication protocols, but without clearly defined criteria for assigning functions to layers. Such layerings are desirable to enhance modularity. End-to-end arguments may be viewed as part of a set of rational principles for organizing such layered systems.*

[ALF] distinguishes the data transfer functions in data manipulation and transfer control. The former include operations like moving, buffering, and retransmitting the data to/through/from the network; the latter include operations like flow/congestion control, timestamping, acknowledgement, and framing. It concludes that layering is a powerful approach for the network layer and below, where intermediate relay entities must participate in some aspects of the communications process. These entities can operate at one or more layers without regard to the semantic content of the symbols being exchanged at the upper and lower layers. At the transport layer and above, where symbols are exchanged on an end-to-end basis, layered isolation might be less suitable, and at least depends on the application.

The end-to-end arguments are summarized in RFC 2775 [RFC2775]:

> *This is an argument first described in [EndArg] and reviewed in [RFC1958], from which an extended quotation follows:*
>
> *"The basic argument is that, as a first principle, certain required end-to-end functions can only be performed correctly by the end-systems themselves. A specific case is that any network, however carefully designed, will be subject to failures of transmission at some statistically determined rate. The best way to cope with this is to accept it, and give responsibility for the integrity of communication to the end systems. Another specific case is end-to-end security.*
>
> *"To quote from [EndArg], 'The function in question can completely and correctly be implemented only with the knowledge and help of the application standing at the endpoints of the communication system. Therefore, providing that questioned function as a feature of the communication system itself is not possible. (Sometimes an incomplete version of the function provided by the*

communication system may be useful as a performance enhancement.)'

"This principle has important consequences if we require applications to survive partial network failures. An end-to-end protocol design should not rely on the maintenance of state (i.e. information about the state of the end-to-end communication) inside the network. Such state should be maintained only in the endpoints, in such a way that the state can only be destroyed when the endpoint itself breaks (known as fate-sharing). An immediate consequence of this is that datagrams are better than classical virtual circuits. The network's job is to transmit datagrams as efficiently and flexibly as possible. Everything else should be done at the fringes."

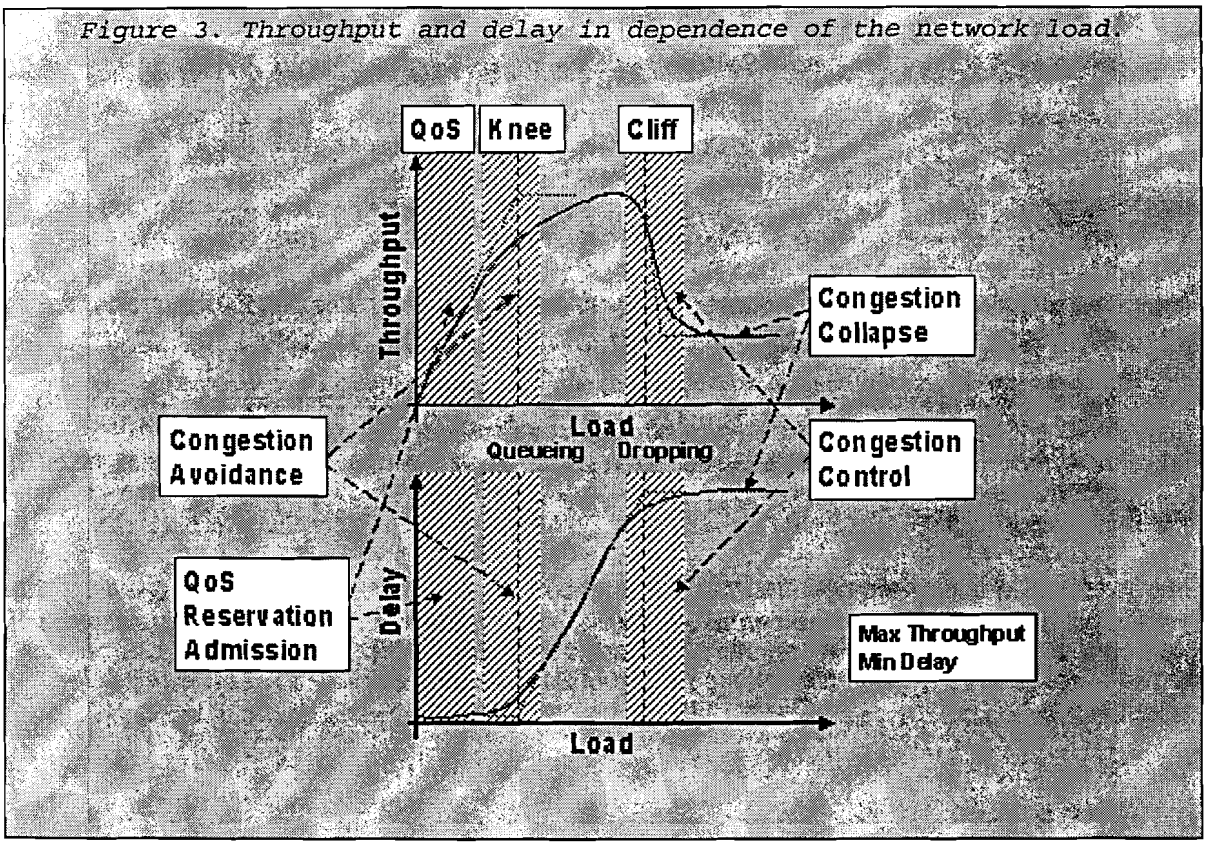
End-to-end performance more or less assumes an end-to-end address transparency, i.e. a single logical address space [RFC2775]. Aside from the source and destination addresses to be unique labels for the end systems, this enables packets to traverse the network essentially unaltered. This (un)alteration of packets is not absolute; fragmentation and hop count limiting, for example, are not obstructing the end-to-end principle. The essence relates to the ***fate-sharing*** aspect of the end-to-end principle. Alterations that require ***per flow state*** information to be kept at the intermediate nodes violate this aspect (aside from the scalability issues that it would raise).

The Internet - Congestion Control

he Internet Protocol is a ***best-effort*** protocol. The network accepts all packets that are submitted and does he best it can in delivering these. However, it does not guarantee the delivery, the latency it consumes for elivery, and the order in which it delivers the packets. In other words, there are no *QoS guarantees*. Any party an start a communication at any time. There is no *admission control*. Since the rate at which a source can ubmit packets and the number of sending sources are unbounded, the network may become overloaded, over ven congested, and a congestion control mechanism is needed.

stable congestion control mechanism operates according to the principle of negative feedback: the ubmission rate is reduced when the network signals a load exceeding a reference value, while it is increased hen the load drops below that target value. Next to stability, it is usually required that the congestion control echanism converges to a ***fair*** distribution of resource usage, that is, all sessions converge to the same rate sing an equal share of the network's resources. The fairness problem can be understood as to maximize an llocation objective function, derived from a fairness policy, under the constraint that no links in the network are eing congested [Fairness].

Figure 3. depicts schematically how throughput and delay are affected by the network load.



In the first phase the network is unloaded and the throughput increases proportional with the submitted data load. The delay stays fairly constant and is mostly determined by the routing processes and the link propagation speeds.

In the second phase, at the *Knee*, the network becomes loaded and the throughput increases less fastly. Part of the data load is absorbed by the queueing buffers in the routers instead of being in transmission across a link. The waiting time in the queueing buffers becomes significant and is added to the total delay. Occasionally, a packet may be dropped when it arrives at a fully filled queue.

In the third phase, at the *Cliff*, the network becomes congested and the throughput saturates or even may collapse. Packets in the queues are dropped and the delay explodes. Congestion collapse is a stable condition, assuming fair dropping treatment by the network.

Congestion collapse is believed to be due to the fact that packets newly submitted to the network are retransmissions of packets which have been dropped earlier. These retransmitted packets get dropped as well, and effectively there is no throughput traffic through the network. Congestion collapse occurs when a packet is more than once in the network, or has already been received, i.e. when the `retransmission timeout < RTT`.

There exist other forms of collapse. One arises when bandwidth is wasted by submitting packets that get dropped before arrival. This can happen by flooding the network, or, even more severe, when the flooding is increased in an attempt to compensate for unsuccessful arrivals, e.g. through the addition of (more) redundancy to a corrupted audio stream. This type of collapse is not stable, in the sense that it recovers when the load is released.

Note, the subtle difference between users submitting packets at a too a high rate and too many users attempting to access the network. In the first case the resulting congestion can be solved by applying admission control and/or flow control, in the second case only admission control will help (or, the network needs to be provisioned with increased

capacity).

It is also interesting to note that, unlike the rate-based transmission schemes, the window-based schemes, such as TCP, are self-limiting: The source stops submitting packets once it has a full window of data standing out. This property makes the window-based schemes intrinsically stable.

As said, a control mechanism is needed to prevent the network from a congestion collapse occurring. Because the Internet Protocol provides a best-effort service, it is the protocol at the transport layer, or otherwise the protocol at the application layer, that is assumed to take care of this. TCP belongs to the first category; applications using UDP belong to the second.

The approach fits with the Internet design principle of pushing complexity to the end hosts. There is no per flow reservation or control by the network. It does introduce the assumption of non-selfish, cooperative behavior, however. The sending host is expected to adjust its data submission rate according to the sensed state of network load, such in a fair manner. Another implication is that the control mechanism must allow for a distributed implementation.

A control mechanism operating at the *"cliff"*, i.e. the network enters a collapse and the control mechanism recovers, is referred to as ***congestion control***, while one that operates at the *"knee"* is referred to as ***congestion avoidance***. The other approach is to reserve the network resources for the sessions at hand, and to require new sessions to gain access through an ***admission control*** procedure. See the section on streaming QoS.

FAIRNESS.

Fairness provides an objective function to resource allocation policies. Because of the stateless nature of the network, the allocation cannot be based on resource reservations. However, depending on the network policy, the definition of fairness can be refined, such as to account for the differences in traffic types. Commonly though, fairness indicates that all users obtain an equal share of the available resources.

A frequently cited measure of fairness is the ***fairness index*** [AIMD]:

$$f = (\Sigma_n x_i)^2 / (n \Sigma_n x_i^2)$$

where x_i is the resource allocation (throughput) to the ith connection, and n the total number of connections through the bottleneck link. The fairness index ranges from $1/n$ to 1. It is maximum when all users receive the same allocation, it is minimum when one user receives all the allocation. The index is k/n when k users equally share the resource, and the other $n-k$ users receive zero allocation.

Another measure is the ***min-max ratio***, defined as

$$m = \min_{\{i,j\}} (x_i / x_j)$$

that ranges between 0 and 1. When $n=2$ there is a one-to-one mapping between m and f. f reflects the fairness in resource allocation of the total system; m captures the fairness as perceived by an individual user, viz. the one with worst allocation. For $m=1$, f also equals 1. For $m=0$, f ranges between $1/n$ and $(n-1)/n$, depending on wether $n-1$ users or "only" one user has not been allocated a resource.

A generic measure is based on ***majorization*** [Majorization]. In majorization the allocations x_i of a feasible distribution are ordered in increasing size. The (most) fair allocation distribution {X} is that one that, compared that all other feasible allocation distributions {Y},

$$\Sigma_{i=1..k}\ x_i >= \Sigma_{i=1..k}\ y_i$$

for all 1<=k<=n, and where equality holds for k=n. The number of users and their totally received allocation are assumed to be the same in all the feasible distributions. As the comparison is on the sum of allocations from the smallest x_i onwards, it follows that fairness is increased by increasing the small and decreasing the large allocations. The maximal fair distribution achievable is that one of which all x_i are equal (an obvious property of all measures of fairness). In other words, the contribution of x_i to the fairness decreases with its rank in the distribution {X}, and the most majorized allocation distribution is that one that maximizes $\Sigma\ g(x_i)$ for any concave function g. ***This concave property of the "fairness value" function g has an interesting parallel with the utility function mentioned above and the (economic) law of diminishing returns (revenues). Those are also of concave shape, and, thus, are also maximized with fairness.***

All measures of fairness lead to the same allocation distribution in case the resources are located at a single node. Fairness can be refined by weighting the allocations per connection. For example, to assign priorities for pricing and balancing between the typical rates needed by applications. Without weighting, users with a small demand are likely to receive their complete demand be allocated - it depends on the policy whether this is considered to be fair.

There are various approaches to realize "fairness". The most common form is that of Additive Increase Multiplicative Decrease (***AIMD***), as TCP implements and discussed below. Each strategy leads to another (fair) distribution of resource allocations. The most often cited forms are ***max-min*** [DataNetworks], and ***(max) proportional*** fairness [PropFair]. Others include ***max throughput*** and ***min (potential) delay*** [MinDelFair]. In [Majorization] several other measures are listed.

In ***max-min fairness*** all connections get the same share of the bottleneck. If a connection cannot use all of its share, e.g. because of another bottleneck, the excess capacity is shared among the remaining connections. In other words, bandwidth is allocated first to the connection with the minimum throughput. The minimum throughput is maximized; a user is always able to send up to the rate of equally shared bandwidth.

In ***proportional fairness*** the principle of diminishing returns is applied: connections are assigned additional bandwidth such that the relative increase in bandwidth is maximized. A connection may not receive all of its share, say obtaining x% less, because that would provide another connection with a larger gain, larger than x% of that other connection. If the flows pass through one single bottleneck the proportional fairness solution is also max-min fair.

The AIMD type of flow control tends to distribute rates according to proportional fairness [Fairness]. Note, that the primary objective of AIMD is to protect against congestion collapse, providing stability; the secondary objective being to do that in a fair manner. (Other objectives, such as providing fast convergence and small oscillations at equilibrium, are not part of the original approach of AIMD.)

In other words, AIMD rather leads to a certain allocation distribution, than being the result of realizing a fairness objective, such as optimizing the utility function leading to proportional fairness. For example, asymmetry in the feedback, in particular differences in feedback delay (RTT), leads to another distribution in the allocations. This is a consequence of the AIMD implementation and, depending on the actual configuration of flows, is not necessarily in line with the optimal solution that would derive from the proportional fairness policy [FairBWSharing]. In fact, a rate(window)-based flow control scheme is independent on RTT in case its fairness is rate(window)-oriented, but will depend on the RTT in case of a window(rate)-oriented fairness [WindowRateControl].

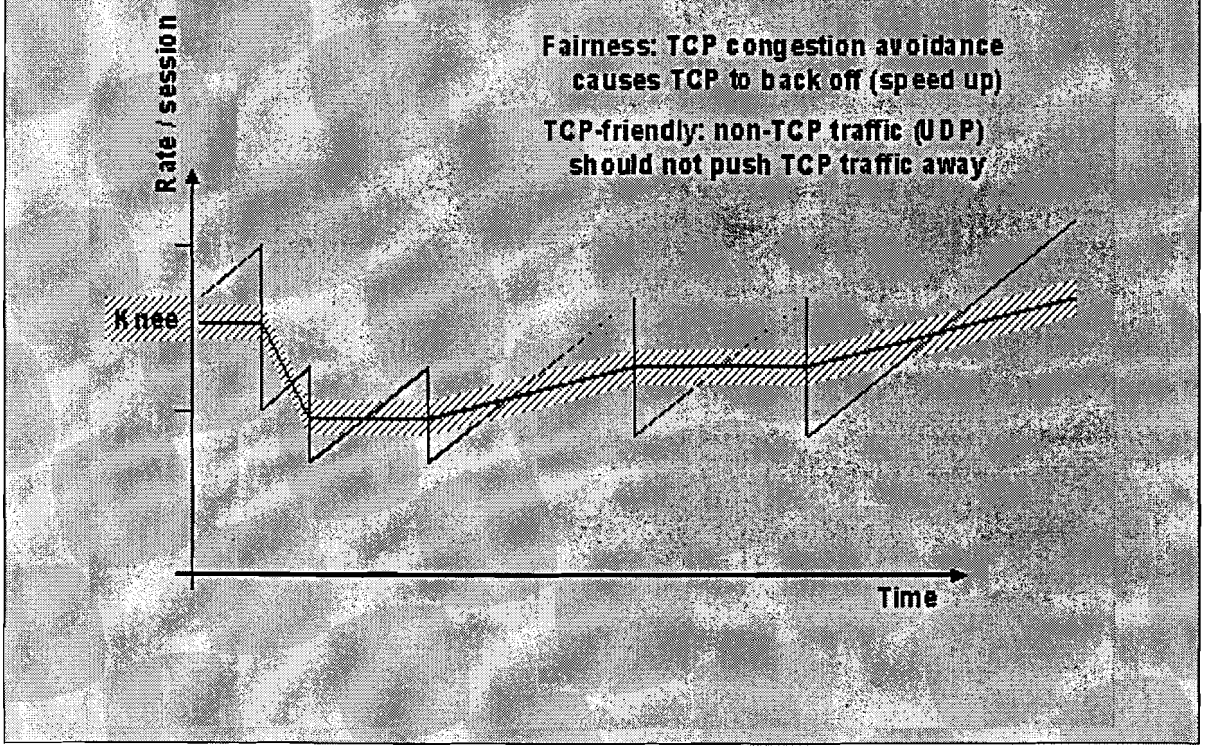
The basic algorithm to realize fair submission rates in a distributed manner while maximizing the network utilization, is given by the principle of ***Additive Increase Multiplicative Decrease (AIMD)*** [AIMD]. The

sending host increases its submission rate in an additive way up to when it detects congestion. Upon reception of such a congestion signal it decreases the submission rate in a multiplicative way, after which it starts increasing the rate additively again. *(The terminology "additive" and "multiplicative" refer to the type of adaptation in window or rate. When expressed in terms of the changes over time, the adjectives would have been "linear" and "exponential", respectively.)*

This additive-increase/multiplicative-decrease process causes the submission rates of different hosts to converge to an equal, i.e. fair, rate. This can be understood as follows. Assume the rates differ. Upon each decrease this difference is reduced by the multiplication factor, while during the increasing phase the difference stays constant.

A consequence of the AIMD algorithm is the resulting sawtooth kind of data transmission rate, as illustrated by Figure 4.

Figure 4. TCP transmission rate.



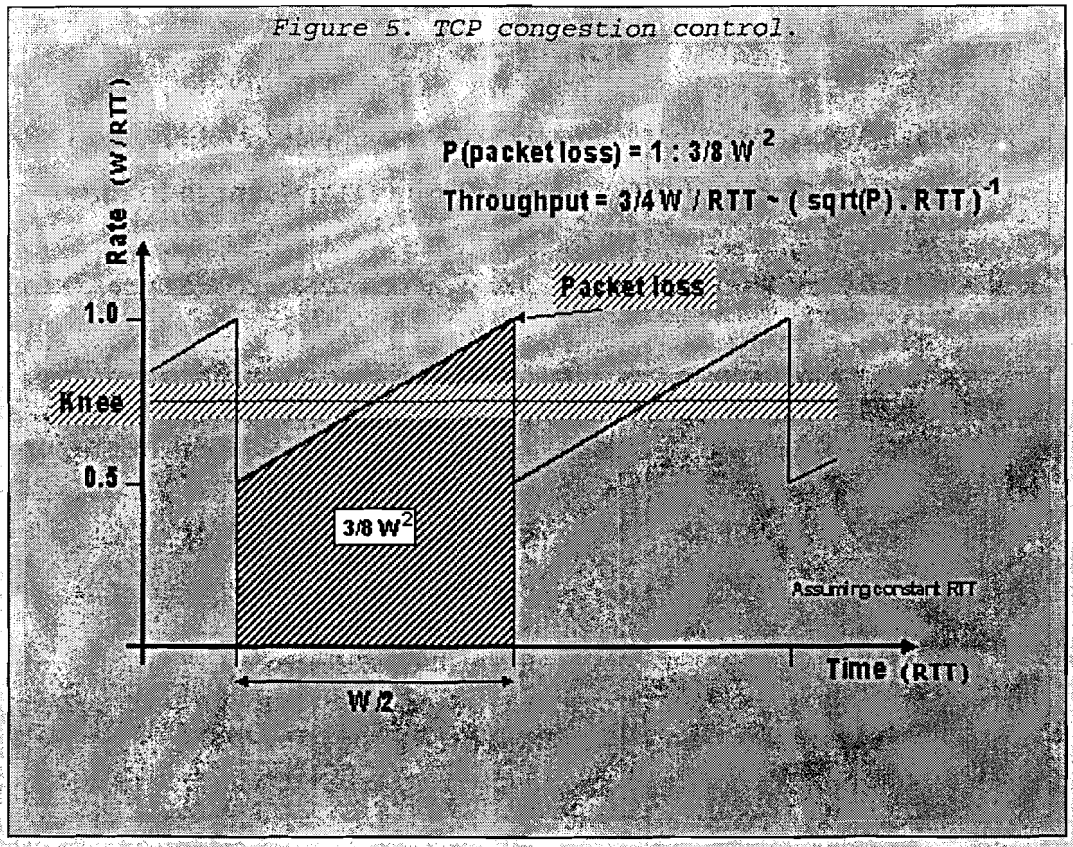
In case of elastic traffic this does not pose a problem, but in case of streaming traffic the algorithm is less suitable. On the other hand, AIMD has resulted from the requirement for congestion prevention by control of the submission rates to a best-effort network in a distributed, fair manner. Some form of admission control, associated with reservation of network resources, is implied otherwise.

TCP CONGESTION CONTROL.

Basically, TCP congestion control operates according to the AIMD principle. There are additional algorithms to accelerate and maintain the convergence at the fair transmission rate. These are called ***slow start***, ***congestion avoidance***, ***fast retransmit***, and ***fast recovery***. The transmission rate is controlled by the window size of not yet acknowledged packets. By increasing that window more packets can be in transfer at the same time, effectively increasing the transmission rate.

Figure 5. displays the most basic situation [TCPRate]. Steady-state is observed and it is assumed that the round trip time, RTT, between a packet being send and the reception of its acknowledgement, ACK, is constant. For simplicity it is further assumed that all packets are of equal, unit size. Thus, a window size w corresponds to w outstanding packets.

Figure 5. TCP congestion control.



Upon every received ACK, TCP increases its window by `1/w`. Approximately, in one RTT there are `w` ACKs received and the window will be increased by 1. This shows the additive increase of AIMD. Also, since `w` packets are transmitted, the transmission rate follows as `W/RTT`.

The window is increased until congestion occurs, what causes the buffers to drop packets. The packet loss is used by TCP to detect congestion and subsequently to halve its window. This shows the multiplicative decrease of AIMD.

Figure 5. shows the triangular shape of the resulting transmission rate. One cycle period is commonly referred to as an ***epoch***. At the beginning of the epoch, just after a packet loss has occurred, the window has been halved to `w/2`, i.e. the transmission rate has dropped from `W/RTT` to halve that value (here, `w` is used to indicate the window size at its maximum). Since the window increases by 1 each RTT, it takes `w/2` RTTs to return to the full size `w` again. So, the average throughput rate is `0.75 W/RTT`. The number of packets transmitted during an epoch is `w/2 RTT` times `0.75 W/RTT`, or `3/8 w2`. There is one packet drop per epoch, and thus the packet loss probability `P` follows as `1 : 3/8 w2`. (More generally, `P` is to be understood as the congestion notification rate.) Eliminating `w` from the throughput rate shows that the throughput rate is inversely proportional to the square root of `P`. Because of the queueing delays induced by the associated larger network load, the RTT will increase with the drop probability. Effectively it causes the throughput to decrease more than (inversely) proportional to the square root of `P`. This can be accounted for by adopting the RTT to the value as measured by the connection.

The original version of TCP did not include congestion control [CerfKahn]. It did provide for flow control, however. ***Flow control*** *concerns the transmission rate of a single connection between two hosts, such as to balance with the processes and buffers at either side of the connection.* ***Congestion control*** *concerns the submission rate to the network, such as to balance the network's load. (Rate control relates to flow control, usually indicating the use of a rate- i.s.o. a window-based control scheme, and being applied on streaming, the transmission of continuous media.)*

Streaming

Streaming - Transmission of Continuous Data over the Internet

Basically, there are two approaches to deliver continuous media over the Internet. The first approach uses the existing Store & Forward method of data delivery. The media is downloaded before being reproduced. Figure 6. depicts the scenario in a Web environment.

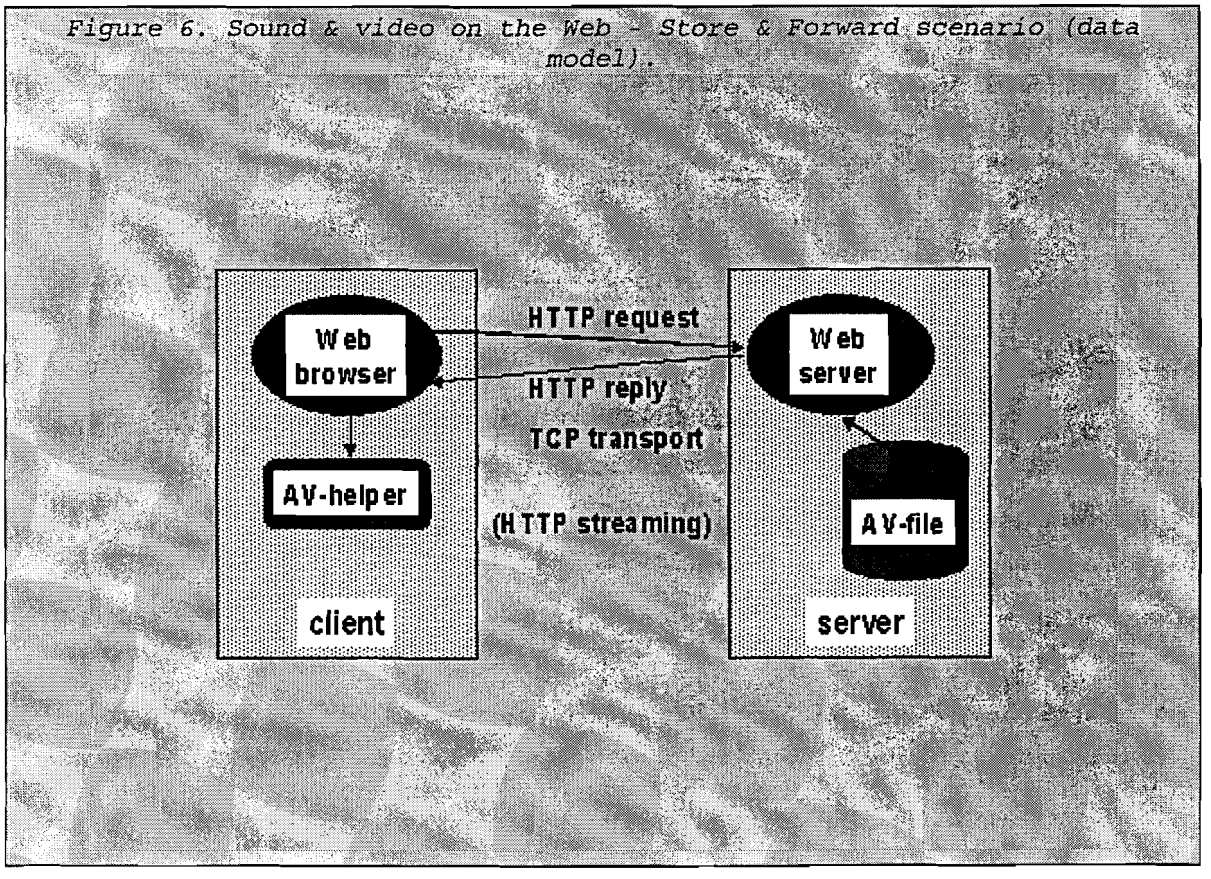


Figure 6. Sound & video on the Web - Store & Forward scenario (data model).

The media data are delivered through a HTTP request-reply sequence. TCP is used as the transport layer protocol, which ensures the reliable delivery. Because the media is first delivered before being reproduced, this scenario is only useful in case of pre-recorded material that is being stored in files of small size *(small is relative to the available bandwidth).*

Assuming that the average throughput of the TCP connections stays constant during the session, the reproduction can be advanced by starting to reproduce when a certain amount of the data has been received. This is commonly referred to as ***HTTP streaming***. Because the transport is using TCP, there is poorly control to optimize for the streaming performance. TCP is designed for elastic traffic, and does not support synchronization. TCP adapts to congestion in an autonomous way, which disables throughput control optimized for streaming. *(Alternative solutions must still satisfy the fairness criterion.)* TCP provides reliability based on ARQ and is order constrained: TCP delivers packets to the application in correct, i.e. transmission, order. This causes additional delay in packet reception, and, for example, an audio segment received in time may be withhold by TCP, awaiting arrival of a previous packet, possibly causing both packets to be delivered to the application after their deadline.

In fact, streaming delivery implies new functions to be added to the Internet. These include:

1. A streaming transport protocol for packetizing and synchronizing the data segments.
2. A protocol to control the course of streaming delivery (pause/resume type of functions).
3. Mechanisms to control the delivery performance (QoS): rate control, reliability and latency control/concealment.

The second approach to deliver continuous media over the Internet is to use a transport method that is optimized for streaming delivery according to these new requirements. Figure 7. depicts the scenario in a Web environment.

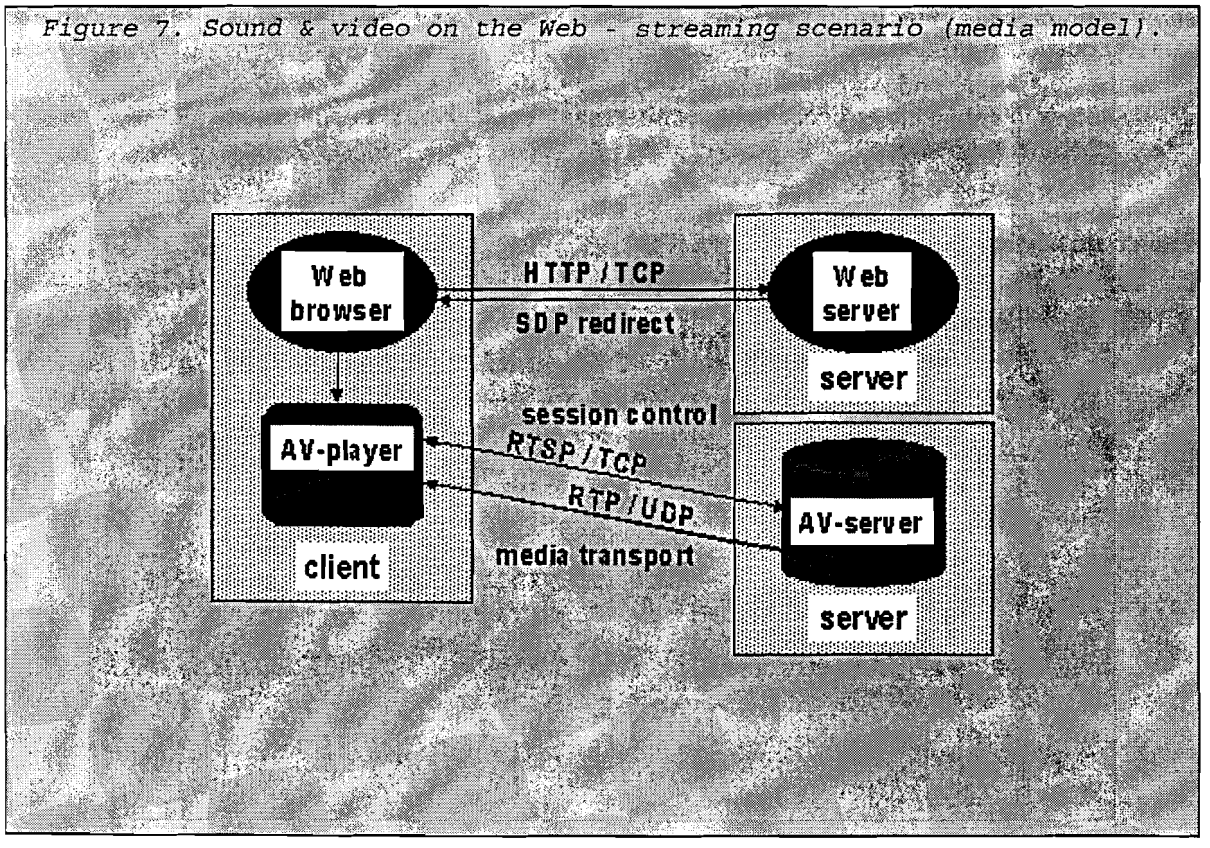


Figure 7. Sound & video on the Web - streaming scenario (media model).

Again, the media data are requested through an HTTP request. However, the HTTP reply contains a redirect, where a.o. the client is instructed to start a streaming session. *(Instead of HTTP, RTSP can also be used in this step.)* The redirect initiates a RTSP request-reply sequence between the streaming client and the streaming server. RTSP is used to control the course of the streaming. The first request (reply) is to start (successfully) the streaming; the final pair stops the streaming. Intermediate requests include pausing and seeking type of control. A consequence is that the server and client maintain state in an RTSP session. In a sense, the continuous nature of the streaming delivery has implied for a generalization of the "discrete" request-reply pair into something continuous. Another consequence is that the actual data transport is (commonly) not part of the "reply" message, but through a separate delivery protocol (such next to the implications imposed by the packetizing requirements).

Streaming - Transporting the Continuous Data

As said, in general, when the reply on a request for continuous data is streamed to the client, the data transport is separated from the request-reply (course of stream) control protocol. RTSP is the protocol designed for the control of the course of the stream. ***RTP***, the Real-time Transport Protocol, has been designed for the streaming transport [RTP].

TCP is the transport protocol for unicast, elastic traffic. It is not suitable for multicast, and not optimal for streaming traffic. UDP is the alternative transport protocol. UDP adds port addressing to the IP protocol, i.e. it adds multiplexing of applications on the host *(IP addressing provides multiplexing of host addresses on the Internet)*. UDP enables the application layer to build its own transport service on top of the network layer's datagram-based, best-effort service *(which has been identified as the common denominator to all types of traffic)*. RTP builds on UDP, using its port addressing for associating the host's media streaming processes within the total multiplex of communication traffic.

RTP extends UDP with generic packetizing and synchronization functionality. It adds sequence numbering and timestamping for identifying packet order and sampling order, respectively. It is the responsibility of the

application to act on these order informations to realize reliability and congestion control, and (re) synchronization and buffer management. In order to assist the application in this control task, RTP is augmented with a secondary protocol, ***RTCP***, the RTP Control Protocol, to monitor the performance of the transport: At regular intervals the participants in the streaming session report, among other data, their observed performance in terms of packet loss and jitter. Work is under way to also use RTCP as a general format for requesting retransmission of lost packets.

Figure 8. shows the resulting protocol stack for streaming media over the Internet, and Figure 9. displays the header format of an RTP packet.

Figure 8. Internet streaming stack.

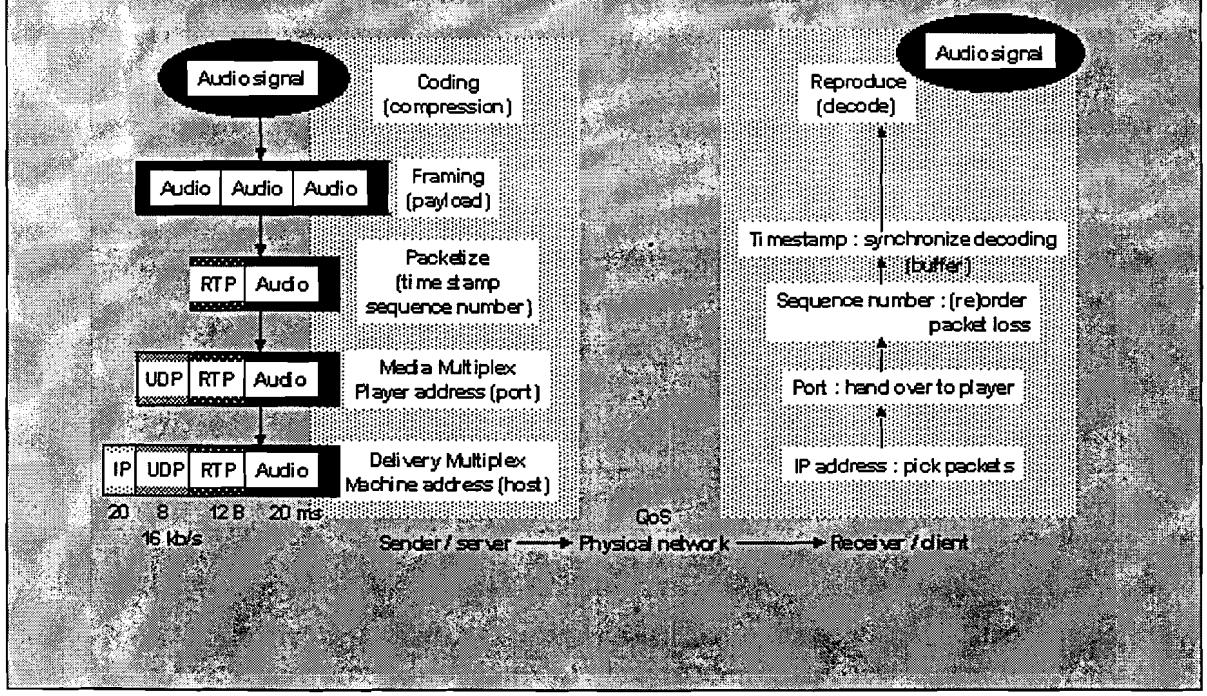
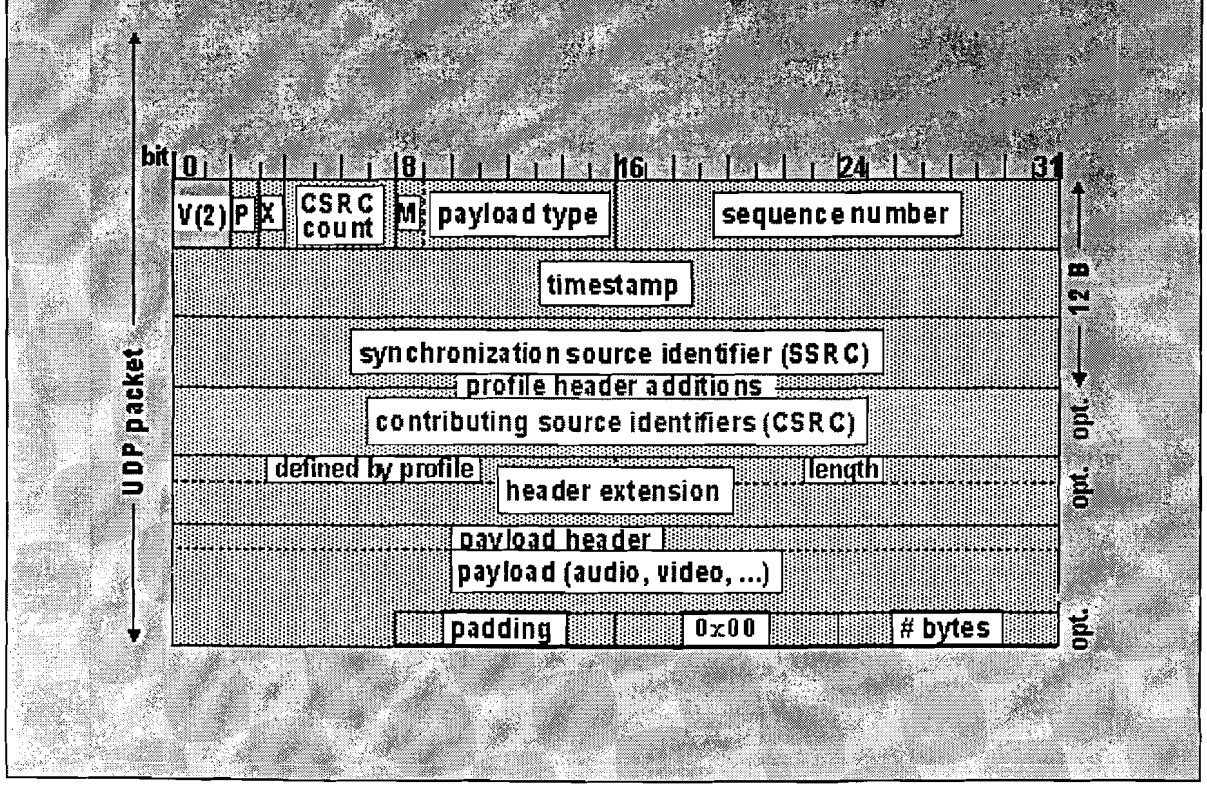


Figure 9. RTP packet header format.



The packet sequence numbers are incremented by one for each consecutive packet. They enable the detection of congestion and packet loss. They also allow the application to (re)order received packets; in particular, when the packets share the same timestamp, which, for example, might be the case in a video frame's encoding that has been segmented over multiple packets. Note, that the timestamps may increment non monotonously with packet order. For example, if video frames are predicted from future frames, the latter can be sent first, such that the decoder can operate more efficiently. Also, the pattern of timestamp increments might be irregular. For example in the case of speech transmission, packet traffic can be absent

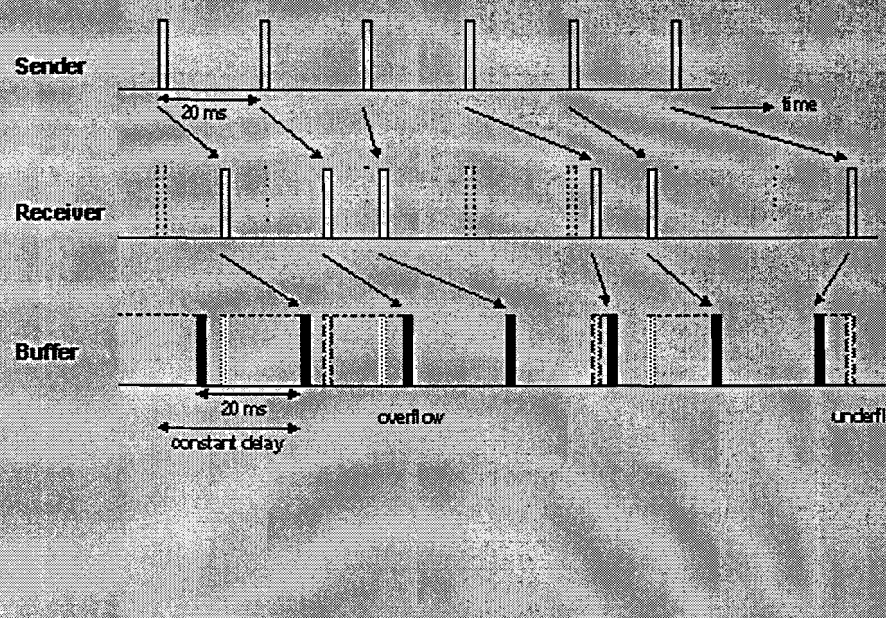
during intervals of silence.

The synchronization source identifier SSRC, see Figure 9., associates with the input device at the senders terminal (microphone and camera). There is one SSRC per input device. The SSRC identifies a timing and sequence number space within and across the IP-address/UDP-port space. For example, an accompanying FEC stream or the various layers in a layered encoding can be collected and synchronized via the SSRC. *(In principle, the UDP-port also provide for identification, since per port one media type is streamed. RTP also supports the notion of mixers, where several (speech) streams are multiplexed in a new stream. In that case the SSRC identifies the mixed stream, while the CSRC fields identify the original SSRC.)*

TIMESTAMPS.

The timestamps represent the sampling times of the packet's data at the sending host. They enable the receiving host to synchronize the received streams. Figure 10. shows the principle.

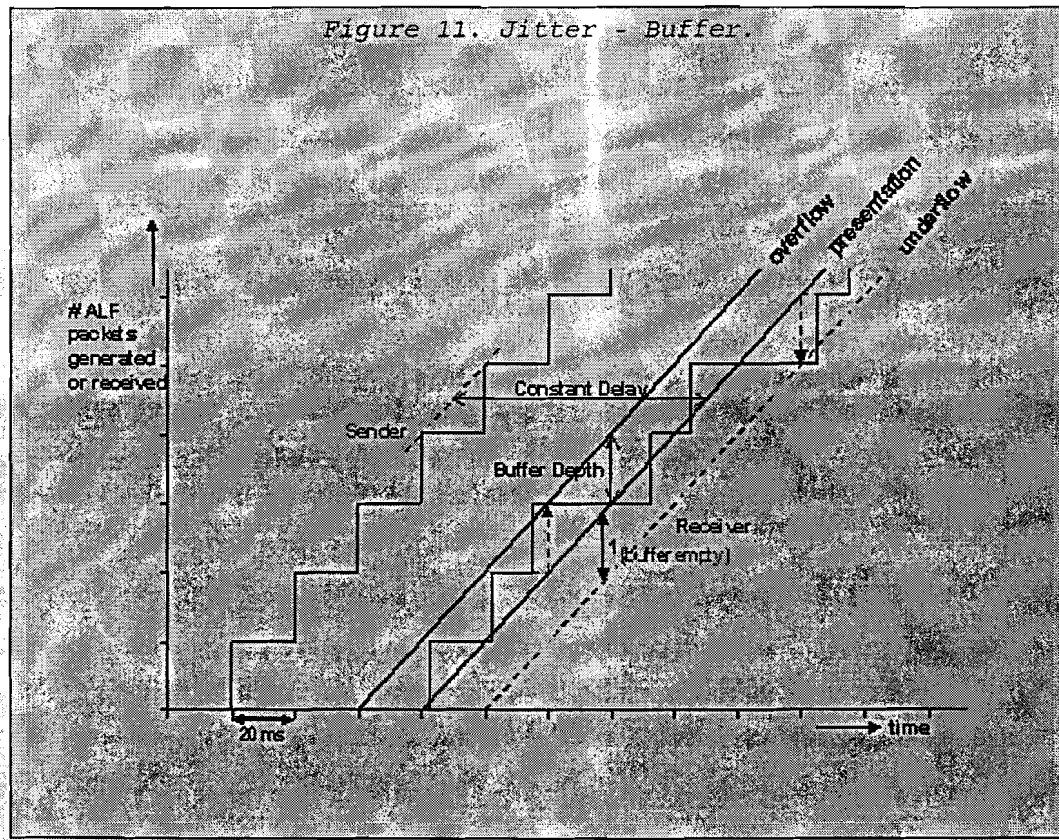
Figure 10. Jitter - Timestamps.



In the figure, the packets are transmitted at regular time intervals of 20 ms. Due to the transmission delay and in particular the variation in this delay, the packets arrive at varying times at the receiver. The receiver stores the packets in a buffer from which they are fetched at the regular time intervals, according to their associated timestamps. The buffer adds delay such that the overall delay becomes constant.

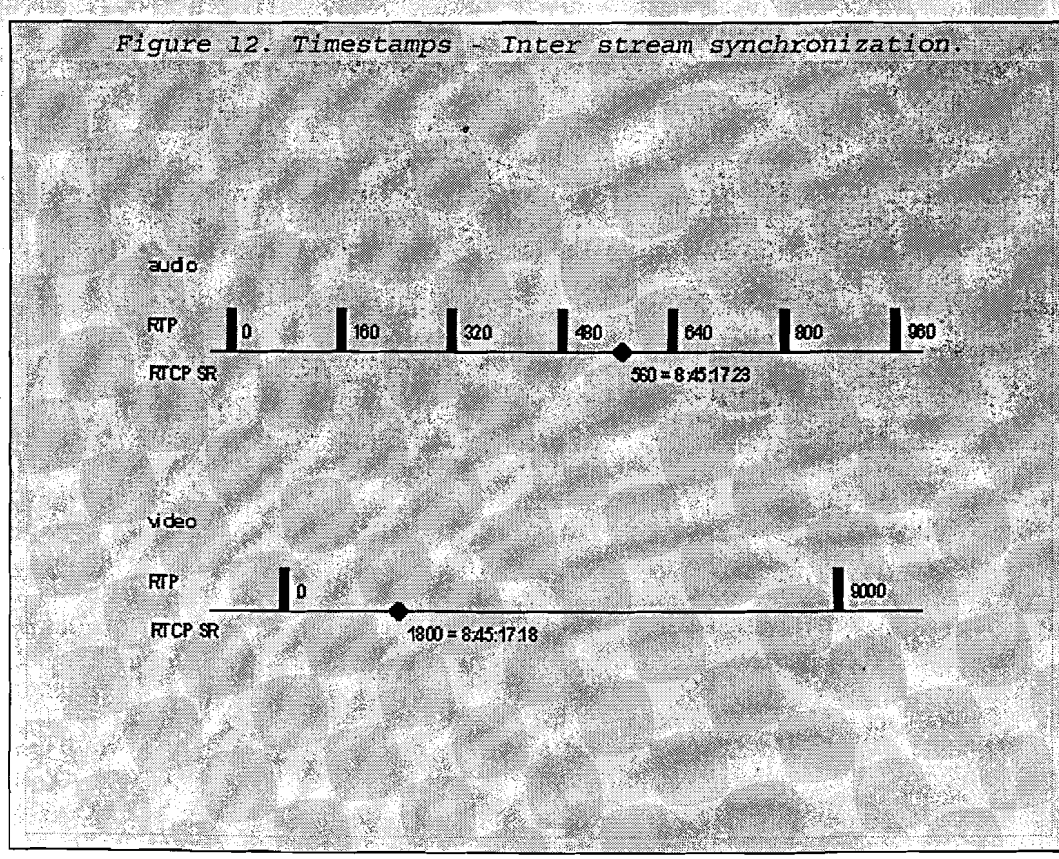
It might happen that a packet arrives too early such that the previous packet has not yet been fetched from the buffer. This is the ***overflow*** situation: the packet is discarded leading to a loss at its scheduled moment for reproduction (unless a retransmission has occurred in time). The dual situation is the ***underflow***: the packet arrives too late, not being available at its scheduled moment of reproduction. It is clear that the buffer depth must be designed and operated according to the jitter exhibited by the network. Operating the buffer at a large overall delay prevents underflow, but may introduce unnecessary (or intolerable) delay. It implies a large buffer depth (which, in turn, represents additional costs). A small buffer depth may not be able to circumvent the jitter, leading to overflow, and/or, depending on the buffer's operation, underflow. Figure 11. reflects this trade-off.

Figure 11. Jitter - Buffer.



The timestamps can also be used for inter stream synchronization, as illustrated in Figure 12. In its RTCP reports the sender incorporates an association between the sampling times of each media stream and its (system) clock, using the NTP format [NTP]. The NTP values in the sender reports on the different media streams provide a reference to synchronize the timestamps of those media streams. ***(These NTP values are not necessarily meant to synchronize the system clocks between sender and receiver.)***

Figure 12. Timestamps - Inter stream synchronization.



CLOCK SYNCHRONIZATION.

Jitter is not special to Internet transport, but a general aspect of packetized transport networks, where multiple traffic flows are served. Commonly, a network provides one or more levels of QoS including guarantees on the jitter bounds. Under the assumption that the clocks at the transmitting side and the receiving side are synchronized, it becomes possible to specify buffer sizes together with data submission policies, which, given the network's QoS guarantees, will prevent any occurrence of buffer overflow or buffer underflow at the receiving side. This yields a hiccup-free presentation of the streaming data.

There are two types of clock synchronization between systems. One is where all systems aim to share the same time, e.g. UTC. That system time is used by the applications running on the system. The NTP protocol serves that purpose [NTP]. The other is where the clocks in a session on all systems but one slave to the master on the remaining system. This second type differs from the first in that the aim is directed on the locking of the clock speeds in the first place, rather than on the value of those clocks (which also need to synchronize, of course). Moreover, the synchronization is needed for the duration of a session within an application. It is this second type that is needed for streaming transport. Of course, solutions to the first could serve this as well, depending on the accuracy they are providing. We discuss the second type further.

A straightforward approach to realize clock synchronization is by monitoring the receiver's buffer filling depth and maintaining that at a fixed level on an average. This approach assumes some sort of a constant delivery rate. For instance, the data are being encoded at a constant bit rate and the amount of data in the buffer is monitored, or the packets are being transmitted at a constant rate and the number of packets in the buffer is monitored (corrected for the emptying rate). This raises a requirement on the spread in arrival times of the consecutive data/packets in the buffer. A large spread implies a long averaging time for accurate clock synchronization, i.e. a low pass filter of narrow bandwidth in the feedback loop controlling the presentation clock. This, in turn, extends the time to convergence, which leads to requiring additional buffer space, i.e. to longer overall delays.

Another approach is to use a separate communication connection for establishing the clock synchronization. That releases the data transmission from being bounded by the clock synchronization requirements, as well as leaves its rate of transmission open to what the application suites best. It is, of course, possible to combine both communications, e.g., by piggybacking the clock samples with the data packets. However, those packets are subject to both required transmission bounds.

A common way to realize clock synchronization, using a separate connection, is by means of a Phase Locked Loop (PLL) [NTPPLL]. This is shown in Figure 13.

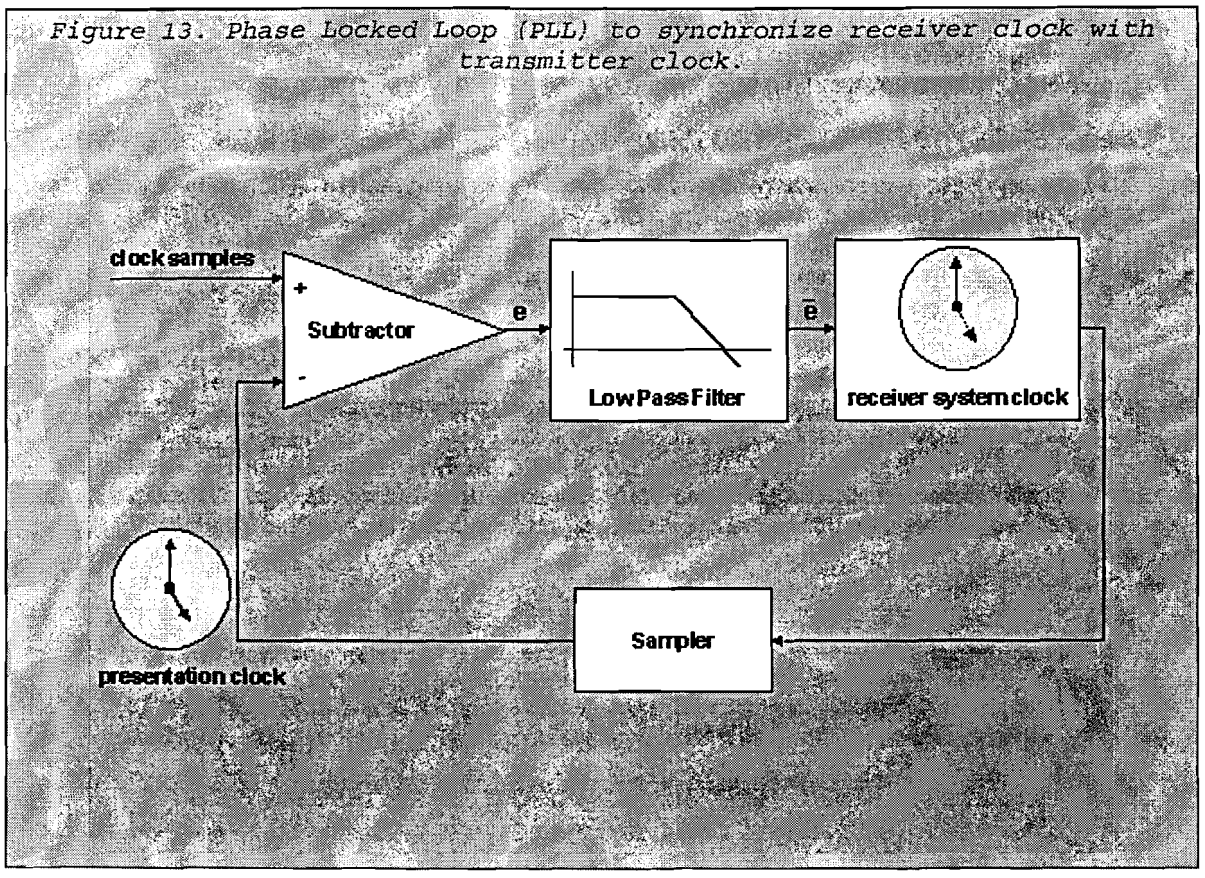


Figure 13. Phase Locked Loop (PLL) to synchronize receiver clock with transmitter clock.

The system clock at the transmitter is sampled and these sample values are sent to the receiver. The receiver's clock is running at about the same rate as that of the transmitter. At initiation its value is set to the first value received from the transmitter (not shown). The subsequently received values are compared with those of the receiver clock, and the difference is used to adjust the receiver's clock rate. By low-pass filtering this difference signal the adaptation speed and accuracy is controlled, i.e., the time before the receiver clock gets in synchronization on the one hand is traded with the sensitivity to jitter in the clock arrivals on the other hand. In terms of the buffer control the PLL adapts for the ***wander*** or long term variations in the clock, whereas the buffer adapts for the ***jitter*** or short term variations in the clock [NetSynch].

RTP is a generic protocol, providing the functions any type of streaming transport needs. Some of its semantics is left open and is required to be specified by a *profile*. Within a profile a further specification is required to detail for the particular media type that is being transported. These are the *payload* specifications. A payload specification usually adds additional header fields, and possibly dedicated transport semantics.

Currently, there exists one profile specification, which is the ***RTP/AVP***, Audio Video Profile [AVP]. The AVP profile specifies that the marker field takes one field, and the payload type field seven, see Figure 9. For speech media types the marker bit is set at the first packet of a talkspurt, in case during silence periods the packet transmission is suppressed or occasionally sends a packet containing data for comfort noise generation. It helps recognition of (non)lost packets. The marker bit can be used to start a new synchronization period of the jitter buffer, where the silence periods are used to adapt the buffer's operating point. In case no silence suppression, i.e. packet transmission suppression, is applied the marker bit must not be set. For video media types the marker bit is set at the last packet of a video frame, i.e. the last packet sharing the same timestamp, such that decoders can use that to synchronize their (frame) decoding (all packets of the frame received; decoding can start). The bit is cleared otherwise. In case of audio media types, the target packetization interval is 20 ms or one frame, whichever is longer. Within AVP many payloads have been and are being specified.

Streaming - Providing the Quality of Service

The support of streaming transport has lead to new performance requirements, collectively referred to as ***Quality of Service (QoS)***:

- Guaranteed throughput rate (average over a small time interval)
- End-to-end in finite time (response time, jitter)
- Reliable (recover erasures in finite time)
  (collapses with the requirement on finite latency: loss is a special case of late arrivals.)

A boundary condition is the requirement of fairness under congestion, where TCP's behavior provides the reference. Applications that adapt in a similar way as TCP, i.e. that do not push out the TCP flows, are called *TCP-friendly*. An approach is TCP Friendly Rate Control (TFRC) [TFRC]. TFRC estimates the equivalent rate to which a TCP connection would converge, and adapts accordingly. TFRC yiels a more continuous rate, but adapts slower to network state. The estimation is using an equation that models the TCP rate [TCPRate], see also the section on TCP congestion control.

As explained in the section on congestion control, traditional IP provides a best effort service, and TCP has been designed to add reliability and congestion control to that. Because of its real-time, non-elastic nature, streaming introduces these new requirements on throughput, delay/jitter and reliability, which relate to the operations congestion/rate control, buffer control and erasure recovery, respectively. RTP does not solve for these, but assumes the application to adapt the sending rate (in a fair manner), to buffer packets, and to conceal errors.

The RTP specification does not include a mechanism for congestion control. However, it does require congestion control, which should be specified in the profiles. Further, RTCP does provide monitoring feedback such that the application can adapt. The Audio Video Profile states that

- *if RTP is operated on a network with QoS support, the RTCP monitor should be used to ensure that the service was received (and if not, assume the network is a best-effort and behave accordingly), and,*
- *if RTP is operated on a best-effort network, RTCP should be used to monitor the loss and to reduce the transmission rate if the average throughput is greater than what TCP would get (or stop transmitting if the loss rate is unacceptably high).*

The user may want an RTP stream to get more than its share of the bandwidth of a narrow link near the user. It is expected that QoS support should be used in that case so that appropriate service can be requested, as in the first case listed above.

Another approach is to solve these QoS requirements by adding the corresponding functionality to the network layer, as explained in the section on QoS. However, it is ***the challenge to preserve the Internet design philosophy***, as explained in the section on the basic design philosophy, and in particular ***to preserve the separation between data manipulation and transfer control***, as mentioned in the section on the end-to-end arguments. This suggests that only basic QoS assurance is added at the network layer (bandwidth and delay bound), while the full optimization happens at the transport and/or application layer (reliability and rate/jitter).

The IP philosophy is based on the end-to-end arguments [EndArg] and is characterized by stateless datagram transport. This philosophy has shown the Internet to be robust, flexible, scalable, and simple. Complexity is pushed as much as possible to the edges of the network. At the network layer the routers perform the routing and forwarding functionality, at the transport layer the end hosts construct the service.

Conclusion

We discussed the basic principles of Internet transport. The Internet has been designed for data transport over multiple, heterogeneous networks. At the network layer IP provides the packet forwarding, in a best effort manner. At the transport layer TCP adds reliability and congestion control, in a fair manner.

For streaming both layers need extensions. At the transport layer the RTP, Real-time Transport Protocol, is added. Next to providing a packetizing format its main function is to provide synchronization information.

Congestion control and reliability are devised to the application layer. At the network layer Differentiated Services is added. It provides QoS in the form of service assurance (bandwidth or loss probability).

Given the level of QoS, including best-effort, the problem of adding streaming capabilities to the Internet falls apart in managing the throughput and the end-to-end delay in a real-time, i.e. instantaneous, manner. Delay includes the aspects of delay variation (jitter) and packet loss (erasure). Managing these can be through adding control and through creating adaptation mechanisms.

Managing the throughput can be characterized in terms of the following control-aspect pairs:

1. congestion control - fairness, AIMD
2. rate control - TCP-friendly
3. admission control - utility, selfish independent users

TCP/IP are designed in concert for data traffic. The challenge is to divide the support for streaming and QoS over both layers also in concert, and to tune the streaming with the data delivery. It implies that the network is not solely responsible in supporting the streaming delivery, and that the end hosts contribute in realizing the required performance.

References

[AIMD]
D-M. Chiu and R. Jain, *Analysis of the Increase and Decrease Algorithms for Congestion Avoidance in Computer Networks*, Comp. Netw. ISDN Syst. 17 (1989) 1-14.

[ALF]
D.D. Clark and D.L. Tennenhouse, *Architectural Considerations for a New Generation of Protocols*, Proc. ACM SIGCOMM 1990, Philadelphia, USA, ACM Comp. Comm. Rev. 20 No.4 (Sep. 1990) 200-208.

[Audiocast]
S. Casner and S. Deering, *First IETF Internet Audiocast*, ACM Comp. Comm. Rev. 22 No.3 (Jul. 1992) 92-97.

[AVP]
H. Schulzrinne, *RTP Profile for Audio and Video Conferences with Minimal Control*, RFC 1890, Jan. 1996.
H. Schulzrinne, S.L. Casner, *RTP Profile for Audio and Video Conferences with Minimal Control*, draft-ietf-avt-profile-new-11.txt, Jul. 2001.

[BEResUtil]
L. Breslau and S. Shenker, *Best-Effort versus Reservations: A Simple Comparative Analysis*, Proc. ACM SIGCOMM 1998, Vancouver, Canada, ACM Comp. Comm. Rev. 28 No.4 (Sep. 1998) 3-16.

[CerfKahn]
V.G. Cerf and R.E. Kahn, *A Protocol for Packet Network Intercommunication*, IEEE Trans. Commun. 22 (1974) 637-648.

[Cidon93]
I. Cidon, A. Khamisy, and M. Sidi, *Analysis of Packet Loss Processes in High-Speed Networks*, IEEE Trans. Inf. Th. 39 No.1 (Jan. 1993) 98-108.

[CL]
J. Wroclawski, *Specification of the Controlled-Load Network Element Service*, RFC 2211, Sep. 1997.

[Collapse]
J. Nagle, *Congestion Control in IP/TCP Internetworks*, RFC 896, Jan. 1984.

[CRTP]
S. Casner and V. Jacobson, *Compressing IP/UDP/RTP Headers for Low-Speed Serial Links*, RFC 2508, Feb. 1999.

[DataNetworks]
D.P. Bertsekas and R.G. Gallager, *Data Networks*, Prentice-Hall, Inc., NJ, USA, 1992.

[DCP]
E. Kohler, M. Handley, S. Floyd, J. Padhye, *Datagram Control Protocol (DCP)*, draft-kohler-dcp-00.txt, Jul. 2001.

[DECBit]
K.K. Ramakrishnan and R. Jain, *A Binary Feedback Scheme for Congestion Avoidance in Computer Networks*, ACM Trans. Comp. Syst. 8 No.2 (May 1990) 158-181.

[DesignPhil]

D.D. Clark, *The Design Philosophy of the DARPA Internet Protocols*, Proc. ACM SIGCOMM 1988, Stanford, USA, ACM Comp. Comm. Rev. 18 No.4 (Aug. 1988) 106-114.
(reprinted in ACM Comp. Comm. Rev. 25 No.1 (Jan. 1995) 102-111.)

[DiffServ]
S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, *An Architecture for Differentiated Services*, RFC 2475, Dec. 1998.
Will be updated with
D. Grossman, *New Terminology for Diffserv*, draft-ietf-diffserv-new-terms-06.txt, Oct. 2001.

[DSFIELD]
K. Nichols, S. Blake, F. Baker, and D. Black, *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*, RFC 2474, Dec. 1998.
Will be updated with
D. Grossman, *New Terminology for Diffserv*, draft-ietf-diffserv-new-terms-05.txt, Aug. 2001.

[ECN]
K. Ramakrishnan and S. Floyd, *Explicit Congestion Notification (ECN)*, RFC 2481, Jan. 1999.

[EndArg]
J.H. Saltzer, D.P. Reed, and D.D. Clark, *End-To-End Arguments in System Design*, ACM Trans. Comp. Sys. 2 (1984) 277-288.

[ErrorConcealment]
C. Perkins and O. Hodson, *Options for Repair of Streaming Media*, RFC 2354, Jun. 1998.

[FairBWSharing]
D.M. Chiu, *Some Observations on Fairness of Bandwidth Sharing*, Proc. ISCC2000, Antibes-Juan les Pins, France, Jul. 2000.

[Fairness]
M. Vojnovic, J.-Y. Le Boudec, and C. Boutremans, *Global Fairness of Addtive-Increase and Multiplicative-Decrease with Heterogeneous Round-Trip Times*, Proc. IEEE INFOCOM 2000, Haifa, Israel, 1303-1312, Mar. 2000.

[FairShare]
S. Shenker, *Making Greed Work in Networks: A Game-Theoretic Analysis of Switch Service Disciplines*, Proc. ACM SIGCOMM 1994, London, UK, ACM Comp. Comm. Rev. 24 No.4 (Oct. 1994) 47-57.

[Floyd94]
S. Floyd and V. Jacobson, *The Synchronization of Periodic Routing Messages*, IEEE/ACM Trans. Networking 2 No.2 (Apr. 1994) 122-136.

[GS]
S. Shenker, C. Partridge, and R. Guerin, *Specification of Guaranteed Quality of Service*, RFC 2212, Sep. 1997.

[IIAD]
D. Bansal and H. Balakrishnan, *TCP-friendly Congestion Control for Real-time Streaming Applications*, MIT Technical Report, MIT-LCS-TR-806, May. 2000.

[ICMP]
J. Postel, *Internet Control Message Protocol (ICMP)*, RFC 792, Sep. 1981.

[InternetModel]
V.G. Cerf and E. Cain, *The DoD Internet Architecture Model*, Comp. Netw. 7 (1983) 307-318.

[IntServ]
R. Braden, D. Clark, and S. Shenker, *Integrated Services in the Internet Architecture: an Overview*, RFC 1633, Jun. 1994.

[IP]
J. Postel (Ed.), *Internet Protocol (IP)*, RFC 791, Sep. 1981.

[ISRSVP]
J. Wroclawski, *The Use of RSVP with IETF Integrated Services*, RFC 2210, Sep. 1997.

[Jacobson88]
V. Jacobson, *Congestion Avoidance and Control*, Proc. ACM SIGCOMM 1988, Stanford, USA, ACM Comp. Comm. Rev. 18 No.4 (Aug. 1988) 314-329.
(reprinted in ACM Comp. Comm. Rev. 25 No.1 (Jan. 1995).)

[Majorization]
R. Bhargava, A. Goel, A. Meyerson, *Using Approximate Majorization to Characterize Protocol Fairness*, Message to the IETF end-to-end list, Nov. 2000.

[MinDelFair]
L. Massoulié and. J. Roberts, *Bandwidth sharing: objectives and algorithms*, Proc. IEEE INFOCOM 1999, New York (NY), USA, 1395-1403.

[NetSynch]
J.C. Bellamy, *Digital Network Synchronization*, IEEE Commun. Mag. 33 No.4 (Apr. 1995) 70-83.

[NTP]
D.L. Mills, *Network Time Protocol (Version 3) Specification, Implementation and Analysis*, RFC 1305,

Mar. 1992.
[NTPPLL]
D.L. Mills, *Internet Time Synchronization: The Network Time Protocol*, IEEE Trans. Commun. 39 No.10 (Oct. 1991) 1482-1493.
[PDB]
K. Nichols and B. Carpenter, *Definition of Differentiated Services Per Domain Behaviors and Rules for their Specification*, RFC 3086, Apr. 2001.
[PDBAR]
N. Seddigh, B. Nandy, J. Heinanen, *An Assured Rate Per-Domain Behavior for Differentiated Services*, draft-ietf-diffserv-pdb-ar-01.txt, Jul. 2001.
[PDBBH]
B. Carpenter and K. Nichols, *A Bulk Handling Per-Domain Behavior for Differentiated Services*, draft-ietf-diffserv-pdb-bh-02.txt, Jan. 2001.
[PDBVW]
V. Jacobson, K. Nichols, and K. Poduri, *The 'Virtual Wire' Per-Domain Behavior*, draft-ietf-diffserv-pdb-vw-00.txt, Jul. 2000.
[PHBAF]
J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski, *Assured Forwarding PHB Group*, RFC 2597, Jun. 1999.
Will be updated with
D. Grossman, *New Terminology for Diffserv*, draft-ietf-diffserv-new-terms-05.txt, Aug. 2001.
[PHBEF]
V. Jacobson, K. Nichols, and K. Poduri, *An Expedited Forwarding PHB*, RFC 2598, Jun. 1999.
B. Davie (ed.), A. Charny, F. Baker, J. Bennett, K. Benson, J.-Y. Le Boudec, A. Chiu, W. Courtney, S. Davari, V. Firoiu, C. Kalmanek, K.K. Ramakrishnan, and D. Stiliadis, *An Expedited Forwarding PHB*, draft-ietf-diffserv-rfc2598bis-02.txt, Sep. 2001.
A. Charny (ed.), F. Baker, J. Bennett, K. Benson, J.-Y. Le Boudec, A. Chiu, W. Courtney, B. Davie, S. Davari, V. Firoiu, C. Kalmanek, K.K. Ramakrishnan, and D. Stiliadis, *Supplemental Information for the New Definition of the EF PHB*, draft-ietf-diffserv-ef-supplemental-00.txt, Feb. 2001.
G. Armitage, A. Casati, J. Crowcroft, J. Halpern, B. Kumar, and J. Schnizlein, *A Delay Bound alternative revision of RFC2598*, draft-ietf-diffserv-efresolve-01.txt, Apr. 2001.
[PropFair]
F. Kelly, *Models for a Self-Managed Internet*, Philosophical Transactions of the Royal Society A358 (2000) 2335-2348.
[QoSOverview]
X. Xiao and L.M. Ni, *Internet QoS: A Big Picture*, IEEE Network 13 No.2 (Mar./Apr. 1999) 8-18.
[QoSPolicy]
R. Rajan, D. Verma, S. Kamat, E. Felstaine, and S. Herzog, *A Policy Framework for Integrated and Differentiated Services in the Internet*, IEEE Network 13 No.5 (Sep./Oct. 1999) 36-41.
[RED]
B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, and L. Zhang, *Queue Management and Congestion Avoidance*, RFC 2309, Apr. 1998.
[RFC1958]
B. Carpenter (Ed.), *Architectural Principles of the Internet*, RFC 1958, Jun. 1996.
[RFC2775]
B. Carpenter, *Internet Transparency*, RFC 2775, Feb. 2000.
[RSVP]
R. Braden (Ed.), L. Zhang, S. Berson, S. Herzog, and S. Jamin, *Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification*, RFC 2205, Sep. 1997.
[RTP]
H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, RFC 1889, Jan. 1996.
H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, draft-ietf-avt-rtp-new-10.txt, Jul. 2001.
[RTSP]
H. Schulzrinne, A. Rao, and R. Lanphier, *Real Time Streaming Protocol (RTSP)*, RFC 2326, Apr. 1998.
[SAP]
M. Handley, C. Perkins, and E. Whelan, *Session Announcement Protocol*, RFC 2974, Oct. 2000.
[SCTP]
R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, *Stream Control Transmission Protocol*, RFC 2960, Oct. 2000.
R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson, *Stream Control Transmission Protocol*, draft-ietf-tsvwg-rfc2960-bis-00.txt, May 2001.
[SDP]

M. Handley and V. Jacobson, *SDP: Session Description Protocol*, RFC 2327, Apr. 1998.
M. Handley, V. Jacobson, and C. Perkins, *SDP: Session Description Protocol*, draft-ietf-mmusic-sdp-new-03.txt, Jul. 2001.
D. Kutscher, J. Ott, C. Bormann, and I. Curcio, *Requirements for Session Description and Capability Negotiation*, draft-ietf-mmusic-sdpng-req-01.txt, Apr. 2001.
D. Kutscher, J. Ott, and C. Bormann, *Session Description and Capability Negotiation*, draft-ietf-mmusic-sdpng-01.txt, Jul. 2001.

[SIP]
M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, *SIP: Session Initiation Protocol*, RFC 2543, Mar. 1999.
M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, *SIP: Session Initiation Protocol*, draft-ietf-sip-rfc2543bis-04.txt, Jul. 2001.

[TCM]
J. Heinanen and R. Guerin, *A Single Rate Three Color Marker*, RFC 2697, Sep. 1999.
J. Heinanen and R. Guerin, *A Two Rate Three Color Marker*, RFC 2698, Sep. 1999.

[TCP]
J. Postel (Ed.), *Transmission Control Protocol (TCP)*, RFC 793, Sep. 1981.

[TCPCongestCtrl]
M. Allman, V. Paxson, and W. Stevens, *TCP Congestion Control*, RFC 2581, Apr. 1999.

[TCPRate]
M. Mathis, J. Semke, J. Mahdavi, and T. Ott, *The Macroscopic Behavior of the TCP Congestion Avoidance Algorithm*, ACM Comp. Comm. Rev. 27 No.3 (Jul. 1997).
J. Padhye, V. Firoiu, D.F. Towsley, and J.F. Kurose, *Modeling TCP Reno Performance: A Simple Model and Its Empirical Validation*, IEEE/ACM Trans. Networking 8 No.2 (Apr. 2000) 133-145.

[TCRTP]
B. Thompson, T. Koren, and D. Wing, *Tunneling Multiplexed Compressed RTP (TCRTP)*, draft-ietf-avt-tcrtp-03.txt, Jul. 2001.

[TFRC]
B. Thompson, T. Koren, and D. Wing, *TCP Friendly Rate Control (TFRC): Protocol Specification*, draft-ietf-tsvwg-tfrc-03.txt, July 2001.

[TSWTCM]
W. Fang, N. Seddigh, and B. Nandy, *A Time Sliding Window Three Colour Marker (TSWTCM)*, RFC 2859, Jun. 2000.

[UDP]
J. Postel, *User Datagram Protocol (UDP)*, RFC 768, Aug. 1980.

[WindowRateControl]
S.J. Golestani and K.K. Sabnani, *Fundamental Observations on Multicast Congestion Control in the Internet*, Proc. IEEE INFOCOM 1999, New York (NY), USA, 990-1000.

Compression for Reduction of Off-chip Video Bandwidth

E.G.T. Jaspers[a] and P.H.N. de With[b]

[a]Philips Research Laboratories, Eindhoven, The Netherlands
[b]CMG Eindhoven / University of Technology, Eindhoven, The Netherlands

ABSTRACT

The architecture for block-based video applications (e.g. MPEG/JPEG coding, graphics rendering) is usually based on a processor engine, connected to an external background SDRAM memory where reference images and data are stored. In this paper, we reduce the required memory bandwidth for MPEG coding up to 67% by identifying the optimal block configuration and applying embedded data compression up to a factor four. It is shown that independent compression of fixed-sized data blocks with a fixed compression ratio can decrease the memory bandwidth for a limited set of compression factors only. To achieve this result, we exploit the statistical properties of the burst-oriented data exchange to memory. It has been found that embedded compression is particularly attractive for bandwidth reduction when a compression ratio 2 or 4 is chosen. This moderate compression factor can be obtained with a low-cost compression scheme such as DPCM with a small acceptable loss of quality.

Keywords: multimedia systems, embedded compression, transfer overhead, adaptive DPCM, data burst, bandwidth reduction, MPEG, memory communication

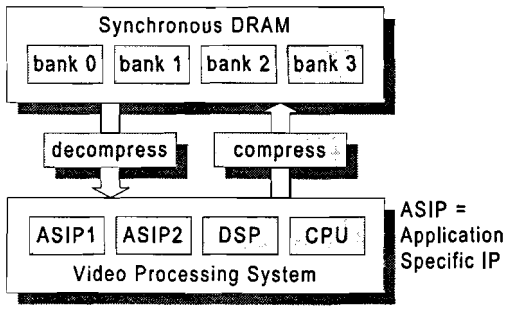


Figure 1: A video system with embedded compression.

1. INTRODUCTION

The processing architecture for block-based video processing functions such as compression standards (MPEG /JPEG), 3-D graphics rendering and field-rate conversion, is usually based on a processor engine consisting of various processors, which is connected to an external background SDRAM memory for the storage of images. This system represents a flexible multimedia platform with a shared memory, that enables simultaneous execution of several high-throughput stream-oriented video processing tasks (see Fig. 1). Due to the fast increase of required computational power of consumer systems, the data communication to and from the off-chip memory has become the bottleneck in the overall system performance (memory wall problem).

In recently developed systems, the memory communication and bandwidth problem was combated by communicating to several memory devices in parallel. However, this leads to the use of multiple memory controllers, more data connections requiring expensive chip packages, and therefore increased system costs. In this paper, we propose the combination of two techniques for substantially reducing the required memory bandwidth, thereby

[a] egbert.jaspers@philips.com, [b]peter.de.with@cmg.nl

keeping system costs low. First, we apply the newly developed technique for optimal mapping of the video into the memory by analyzing the actual memory accesses [1][2]. Second, we exploit an embedded compression technique in combination with our optimized mapping with the aim to further reduce the memory bandwidth.

Let us briefly consider why memory bandwidth is so scarce in video coding and processing systems. Most bandwidth-consuming tasks in multimedia platforms such as high-level MPEG coding and 3-D graphics, can be classified as video processing tasks with block-based pixel access at more or less random positions in the (off-chip) memory. These tasks, requiring large amounts of pixel data and intensive memory access, result in a large bandwidth overhead, due to the burst-oriented data transfer for currently available SDRAM devices. Particularly when the burst size approaches or exceeds the size of the required data block, such as e.g. a macroblock (MB) for MPEG coding, the overhead can be significant and double or even triple the required bandwidth. This paper shows the feasibility of an *embedded compression* technique to reduce the communication bandwidth and combines this with an optimal mapping of pixels into data bursts.

Although many compression techniques have been introduced, their usage for reducing the data bandwidth of the memory is not trivial. For example, a problem in computer systems is that the throughput of an application can degrade significantly when the working space of that application does not fit in the main memory. This results in an increased number of memory page faults so that the background memory on hard disk is accessed more often. As a possible solution, Roy *et al.* [3] implemented a concept of compressed memory pages to reduce the amount of disk accesses for the computer system. Although this may be true, the compressed memory pages need to be decompressed and written back into the memory when they are requested by the application. This process consumes extra memory access, so that the potential bandwidth reduction is influenced in a negative way. Moreover, it will be shown in the next section that a large grain size of compressed data packets such as the proposed memory pages, does not result in bandwidth reduction. Another application for embedded compression is presented in [4]. This approach applies lossless compression to relatively large data packets of 1 Kbyte in a high-complexity memory-controller chip to increase the memory capacity and is mainly intended for general-purpose computer servers. This functionality is located between the main memory and a large cache memory. Because the data traffic between the cache and the main memory is usually based on relatively large-grain packets. For the same reasoning as in the previous example, the compression gives a penalty in bandwidth, but because the large packet size there may be sufficient net gain. In this paper, where we concentrate on stream-based media processing, caching techniques offer only limited improvement because streaming data is commonly used only once within a limited time interval. Our objective is to develop an inexpensive solution without the need for substantial cache memory.

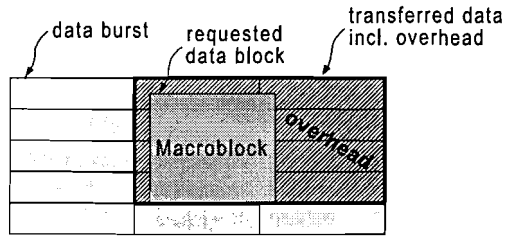


Figure 2: Memory access of a macroblock including the transfer overhead.

The use of compression for embedded memory applications has been studied primarily for reducing the required memory space. The usage of such techniques for memory bandwidth reduction, is not straightforward. For example, in [5], segments of nine macroblocks (MBs) were compressed into a fixed-sized block. Consequently, additional memory accesses are necessary to address individual MBs. In [6], a simple embedded DPCM technique is applied in HDTV decoding for memory capacity compression. Although this paper presents the measurement of bandwidth to derive the utilization memory bus, it does not further analyze the reduction in bandwidth. Van der Schaar *et al.* [7] proposed similar low-cost compression algorithms without bandwidth optimization and

showed later [8] that the obtained compression factor only partially aids in bandwidth reduction. In our paper, we quantify results on the feasibility of low-cost compression schemes (e.g. [9]) for reduction of the memory *communication*. Furthermore, we propose an optimization technique to find all feasible compression factors that reduce the bandwidth between the memory and the video processing. This reduction which can be as high as 67% for a compression ratio of four, can be exploited to enhance the system quality, reduce costs and/or add extra functionality.

Recent SDRAM devices use relatively large data bursts (e.g. 8 words) to improve the effective bandwidth at the expense of *overhead*, particularly when the aforementioned burst size approaches or exceeds the size of the required data block, e.g. a MB for MPEG coding. The overhead is caused by fetching more data than strictly required, i.e. all underlying data bursts (see Fig. 2). An embedded compression technique can be used to store more video pixels into a single data burst. However, as already mentioned, the reduction in bandwidth is not proportional with the compression ratio. Since the compression technique increases the amount of pixels contained in a single data burst, an additional overhead is introduced which has to be compensated by the compression ratio. The results in this paper show that the bandwidth can be substantially reduced for newly developed media systems with a wider memory bus ($\geq$ 32 bits). We have obtained a reduction of the memory bandwidth by 38% in an MPEG decoding system without sacrificing visual picture quality (limited compression ratio two).

The paper is divided as follows. Section 2 discusses the behaviour of SDRAM memory and explains how pixels can be mapped into the memory. Section 3 addresses the compression algorithm and proves the feasibility of embedded compression. Section 4 briefly outlines our new mapping optimization technique [1][2] which is extended for use with embedded compression. Section 5 explores all possible solutions. Finally, we will conclude with the results.

2. MAPPING OF PIXELS INTO THE MEMORY

The efficiency of the memory access depends on two principal elements: the burst-oriented communication and the memory segmentation into separated banks. With respect to the first element, the utilization can be optimized when video data is accessed at the grain size of data bursts (e.g. a 64-bit bus and a burst length of eight words, leading to 64 bytes). These data bursts represent non-overlapping memory blocks that contain pixel data and which can only be accessed as an entity, referred to as *data units*. Consequently, small data-block accesses result in a large amount of pixel overhead, because complete data units need to be fetched from memory. Moreover, the amount of pixel overhead also depends on the aspect ratio of the data units. For example if a MB of 16×16 samples is requested, data units of 64×1 result in more overhead than data units of 8×8, although the size is equal. In [10], a technique is proposed to determine the optimal aspect ratio of the data units by analyzing the application software model only, without considering data dependencies. In our initial work [1][2] an optimal mapping of the video into the memory is determined by analyzing the actual memory accesses, so that data dependencies are taken into account. Note that for our objective, the size S of a data unit does not

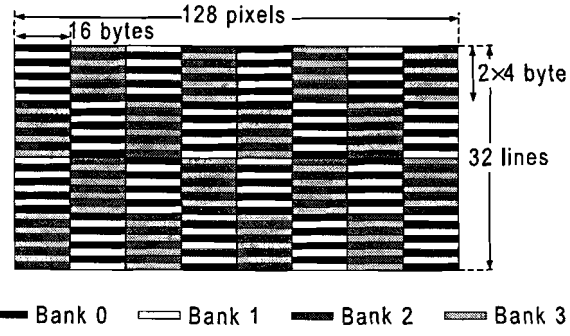


Figure 3: Mapping of video onto memory data units considering both progressive and interlaced accesses.

coincide with the size B of a data burst. For example, data units of 32×4 pixels can be compressed into a 64-byte data burst.
The second principal element for bandwidth efficiency is the organization into memory banks, commonly applied in all modern memory devices. Efficient communication can only be achieved when the memory banks are accessed alternately. Consequently, adjacent video pixels are interleaved in the memory banks. Furthermore, the successive odd and even lines of an interlaced video signal are mapped in different banks of the memory to prevent additional overhead when both progressive and interlaced video data blocks are requested by the video processing. The resulting mapping strategy evaluated for an MPEG-2 decoder, using 16×4 data units, is shown in Fig. 3. Note that an interlaced field requires interleaved access to all banks. Hence, alternating addressing of the memory banks for both progressive and interlaced video is achieved. A more detailed overview of pixel mapping into banks is given in [1][2].

3. EMBEDDED COMPRESSION

The previous section discussed how burst-oriented memory communication influences the optimal mapping of pixels onto the address space. This section will show that such a constraint is exploited to make embedded compression feasible for reduction of memory bandwidth.

Since compression techniques exploit spatial correlation between pixels, it is usually applied for a group of pixels. For example, when nine MBs are encoded as one entity [5], the entity has to be partially decoded before the value of a certain pixel can be determined. For retrieval of an arbitrary MB in the picture, several compressed data entities may have to be decoded because they all contain part of the requested MB. Obviously, this does not help in the reduction of the memory bandwidth, particularly if random access to the memory is required.
Let us consider an example that shows why embedded compression for bandwidth reduction is not easily obtained. We have measured the memory bandwidth of an MPEG-2 decoder.

If groups of 16×8 pixels are clustered as an access entity, 113% more data is transferred than was strictly required for the decoding process. Consequently, a compression ratio of 2.13 is necessary to accomplish the break-even point and even more for a net bandwidth reduction. However, when using an SDRAM-based memory, an overhead is already present due to burst-oriented storage. Table 1 indicates this overhead, dependent on the size of the communicated data bursts. For these numbers, the aspect ratio of the data-units are optimized for minimum transfer bandwidth. The bandwidth percentages show the transfer bandwidth relative to the bandwidth that is strictly required for the decoding. In the sequel of this paper we call these numbers the *relative transfer bandwidth* (RTB). Since the data-burst size is given by the architecture, compression becomes

Table 1: RTB requirement for data units with increasing sizes and the optimal dimensions.

size [bytes]	optimal dimension	requested data [%]	transferred data $\bar{o} \cdot 100\%$
16	(16×1)	100	131
32	(8×4)	100	149
64	(16×4)	100	172
128	(16×8)	100	213

more feasible: only the additional overhead due to compression needs to be compensated. For example, if data units of 16×8 are compressed into 64-byte data bursts, the RTB increases from 172% to 213%. Therefore, the break-even compression ratio changes to $2.13/1.72 = 1.24$, which is feasible.

Up to this point, we discussed the properties of an SDRAM memory that result in block-based (bursts) storage of data and therefore increase the bandwidth requirement significantly. Furthermore, we have explained how

For the bandwidth calculation, the set of possible data-block requests V has to be divided into a subset of progressive data block requests V_p and a subset of interlaced data block requests V_i, such that $V = V_i \cup V_p$. This separation is necessary because the calculations for both contributions are slightly different. We denote the average RTB of progressive data-block requests by $\bar{o}_p(M, N, V_p)$ and for interlaced data-block requests by $\bar{o}_i(M, N, V_i)$. Both contributions are already probability weighted, so that the total average RTB is:

$$\bar{o}(M, N, V) = \bar{o}_i(M, N, V_i) + \bar{o}_p(M, N, V_p). \tag{2}$$

To achieve the minimal bandwidth requirements for a given application, the dimensions of the data units (M, N) are optimized.
In this paper we consider an MPEG-2 decoder as an example for our experiments. For this application the set of data blocks is:

$$\begin{aligned} V_p &= \{(16 \times 16), (17 \times 16), (16 \times 17), (17 \times 17), (16 \times 8), (18 \times 8), (16 \times 9), (18 \times 9)\} \\ V_i &= \{(16 \times 16), (17 \times 16), (16 \times 17), (17 \times 17), (16 \times 8), (18 \times 8), (16 \times 9), (18 \times 9), \\ &\quad (17 \times 8), (17 \times 9), (16 \times 4), (18 \times 4), (16 \times 5), (18 \times 5), (M \times N)\} \end{aligned}$$

The large variety of data-blocks requests is caused by the MPEG standard and depends on field/frame prediction, luminance/ chrominance data and the sub-pixel accuracy. In our experiments, we consider the reading of prediction data for motion compensation, the writing of the motion-compensated result, $(16 \times 16) \in V_p$, and the reading for interlaced display of the video, $(M \times N) \in V_i$. For the last aspect, it is assumed that the display unit contains line memories to read the block-based data units of $(M \times N) \in V_i$ and to display the video lines sequentially. To acquire representative results for an average MPEG-2 bit stream, a large set of test bit streams, containing in total 732 frames, is used to derive the optimum.

To determine $P(B_x \times B_y)$, the amount of occurrences of each type of data-block, all requests are measured at the memory interface, thereby feeding one of the data dependencies into the model.
The probability distribution of the positions of the data blocks $P_{B_x \times B_y}(m, n)$, is defined as the probability that the upper-left corner pixel of a requested data block $B_x \times B_y$ is positioned at any location (x, y), satisfying the condition: $(x \bmod M = m)$ AND $(y \bmod N = n)$. Hence, a low-complexity bookkeeping of the occurrences at position $(x \bmod M, y \bmod N)$ is used to determine $P_{B_x \times B_y}(m, n)$. This probability distribution highly depends on the dimensions of the data units. Large differences may occur in the result, due to the 16×16 MB grid for MPEG and the high probability of the zero-motion vectors. For example, if $(M, N) = (32, 2)$, the probability that $(x \bmod 32 = 0)$ AND $(y \bmod 2 = 0)$ is relatively high. However, for data-unit dimensions that are not aligned with the MB grid, e.g. $(M, N) = (12, 5)$, the probability distribution of the positions of the data-blocks is totally different (see both examples in Fig. 4).

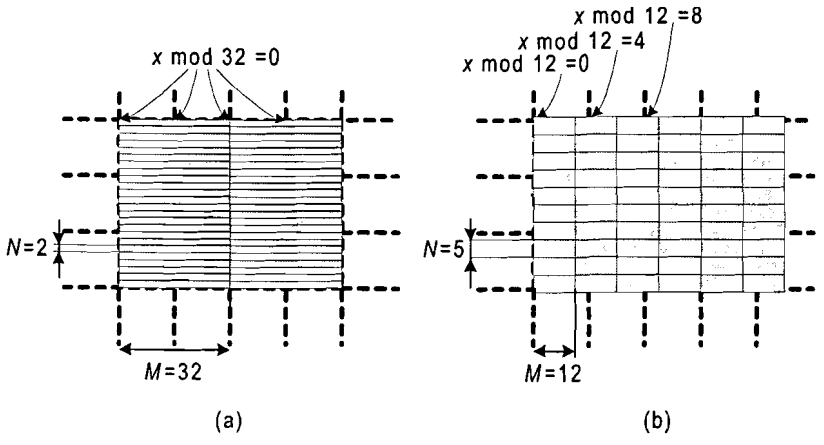


Figure 4: 32×2 (a) and 12×5 data units (b) overlaid on a MB grid.

this block-based storage can be exploited to reduce the bandwidth requirement by means of compression. However, the suitability of compression schemes is bounded by the constraints we have developed so far. Firstly, we found that the burst size determines the size of the compressed data entity. Secondly, the data must be easily accessible at regular address positions, thereby leading to a fixed compression ratio. Hence, this leads to fixed-sized input data units (data-units size S) and output blocks (data-burst size B). Let us now discuss a compression algorithm that satisfies the aforementioned constraints, presented by Bayazit *et al.* [9]. This technique is based on an adaptive DPCM algorithm, which can achieve a 50% memory compression ratio with no distinguishable loss of visual quality. The paper describes the independent compression of one data block consisting of one luminance block of 16×2 samples and two chrominance blocks of 8×1 samples, thus 48 bytes in total. Because our system assumes separate storage of the luminance and chrominance data, it is required to independently compress luminance and chrominance components. This requirement does not limit the suitability of the algorithm and no performance degradation is expected for this reason. Another difference for the applicability of the proposed algorithm is the size of the data units. Data units of 48 bytes are rather small for obtaining sufficient compression. Section 5 will show that most suitable data-unit sizes for 32-byte data bursts ($B = 32$) are $S = 64$ bytes (and larger). For larger data bursts the most suitable size is even larger. Consequently, the algorithm will be able to exploit more correlation when applied for our purpose.

The experiments as described in the above-mentioned paper show that the compression scheme was not matched to the size of the memory bursts. The 48-byte data blocks were compressed with a factor 1, 1.33, 1.6, and 2, resulting in compressed data entities of 48, 36, 30, and 24 bytes, respectively. Because the sizes of these data entities do not match with the alignment grid of the data bursts, these compression ratios lead to sub-optimal solutions; i.e. more data bursts are transferred to access a compressed data entity. The results of the experiments show a bandwidth reduction of 12.5% for a compression ratio of two. The corresponding results on picture quality show a degradation of 0.97 dB for a MP@ML MPEG-2 video decoder, when decoding a high-quality 9−Mbps bitstream. For a bitstream that was coded at 4 Mbps, the quality degraded with only 0.34 dB. Subjective tests revealed high-quality video with imperceptible artifacts. For our system we target to decorrelate larger data blocks and hence expect an even better picture quality. Moreover, the results in the next section will show a significantly higher bandwidth reduction than in [9].

4. BANDWIDTH CALCULATIONS

In our experiments we used the implementation of an MPEG-2 decoder to statistically analyze the behaviour of the data communication. In this section we elaborate on the communication dependencies for computing the overhead. To derive the memory bandwidth, the calculation model as proposed in [1][2] is used. The calculated result from this model represents the transfer bandwidth relative to the bandwidth that is strictly required for the decoding (RTB). However, in this model, embedded compression is not considered. Consequently, the size S of a data unit is assumed to be equal to the data-burst size B ($S = B$). In the following, we extend the above-mentioned model for the use of embedded compression. For this purpose, we introduce the compression ratio c_S, where S stands for the data-unit size. The value of the compression equals the ratio between the data-unit size and the data-burst size, thus:

$$c_S = \frac{S}{B}. \tag{1}$$

Below, we list the parameters on which the calculations for the RTB depend, including the compression ratio c_S:

- the dimensions of the requested data blocks, $B_x \times B_y$;
- the dimensions of the data units, (M, N);
- the interlace factor of the requested data blocks;
- the probability of their occurrence, $P(B_x \times B_y)$;
- the probability distribution of their positions, $P_{B_x \times B_y}(m, n)$;
- the compression ratio, c_S.

At this stage, we have discussed the parameters that are necessary for the calculations and can derive the RTB. For the set of interlaced data-block requests, the following equation applies:

$$\bar{o}_i(M,N,V_i) = \frac{\sum\limits_{B_x \times B_y \in V_i} P(B_x \times B_y) H(M,N,V_i)}{c_S \cdot \sum\limits_{B_x \times B_y \in V_i} P(B_x \times B_y) \cdot B_x \cdot B_y}, \tag{3}$$

with

$$H(M,N,V_i) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} P_{B_x \times B_y}(M,N) \cdot \left(1 + \left\lfloor \frac{B_x + m - 1}{M} \right\rfloor\right) \cdot \left(1 + \left\lfloor \frac{B_y + n - 1}{N} \right\rfloor\right).$$

The summation in numerator in Eq. (3) represents the amount of transferred pixels including the overhead, whereas the summation in the denominator represents the amount of pixels that is strictly required for the decoding without the overhead. c_S indicates the compression ratio. Note that the calculation without c_S resembles the amount of transferred pixels relative to the amount of pixels that are strictly required for the decoding. Since this is directly proportional to the bandwidth, the equation with c_S points to the effective bandwidth relative to the bandwidth that is strictly required for decoding without compression. The RTB calculation for the set of progressive data-block requests is similar to Eq. (3) but V_i becomes V_p and $H(M,N,V)$ is defined according to:

$$H(M,N,V_p) = \sum_{m=0}^{M-1} \sum_{n=0}^{2N-1} P_{B_x \times B_y}(M,N) \cdot \left(1 + \left\lfloor \frac{B_x + m - 1}{M} \right\rfloor\right) \cdot \left(2 + \left\lfloor \frac{\lceil B_y/2 \rceil + \lfloor n/2 \rfloor - 1}{N} \right\rfloor + \left\lfloor \frac{\lfloor B_y/2 \rfloor + \lceil n/2 \rceil - 1}{N} \right\rfloor\right).$$

5. EXTRACTION OF FEASIBLE SOLUTIONS

The results of the previous section present a calculation model for the memory communication, featuring the data dependencies and the usage of embedded compression. Although the feasibility of embedded compression for reduction of the memory bandwidth is already proven by the example in Section 3, this section uses the calculation model to identify all feasible solutions. The method to systematically determine these solutions comprises of the following steps:

- determination of the optimal data-unit configuration for zero compression;
- determination of the optimal data-unit dimensions for an incremental value of the compression ratio;
- pruning of the non-feasible data-unit configurations.

The outcomes of the first step is already presented in Table 1. For example, the optimal dimensions (M_B, N_B) for 64-byte data bursts ($B = 64$) and a data-unit size $S = B$, are formalized as follows:

$$(M_B, N_B) = \left(\exists_{(m,n)} : m \in [1..B], n \in [1..B] :\downarrow \bar{o}(m,n,V) \wedge m \cdot n = B\right), \tag{4}$$

where $\downarrow$ denotes the minimum. For example, the minimum RTB for data units containing 64 pixels equals $\bar{o}(M_B, N_B, V) = 172\%$.

In the second step, the optimal data-unit dimensions are determined for increasing compression ratio. Note that the compression ratio c_S is given by Eq. 1. Thus, for a fixed data-burst size ($B = 64$) the data-unit size is incrementally increased from the size of the data burst up to four times the burst size. Note that this resembles an incremental increase of the compression factor. For each $S \in (B..4B]$ the optimal data-unit dimensions can be determined by:

$$(M_S, N_S) = \left(\exists_{(m,n)} : m \in [1..S], n \in [1..S] :\downarrow \bar{o}(m,n,V) \wedge m \cdot n = S\right). \tag{5}$$

As a result from this calculation the RTB value for the optimal dimensions (M_S, N_S) for each S is $\bar{o}_t(M_S, N_S, V)$. Fig. 5 shows these values as function of the data-unit size. Because most solutions do not result in less data transfer than in the case without compression, they have been removed for an improved visualization of the results. Consequently, all shown solutions satisfy the following equation:

$$\bar{o}_t(M_S, N_S, V) < \bar{o}_t(M_B, N_B, V) \text{ with } S \in (B..4B]. \tag{6}$$

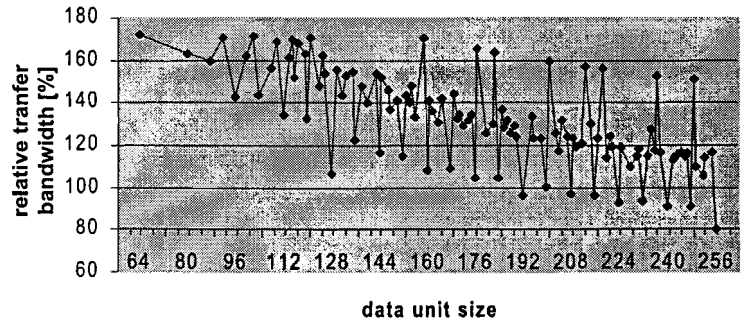


Figure 5: Minimal RTB as function of the data-unit size S and a burst size of $B = 64$ bytes.

In the third and final step, pruning of the found solutions can be applied. This pruning means in Fig. 5 that, starting at the left side and going to the right, only those solutions are adopted that give a lower RTB than the previously selected point. This means that all solutions with a larger compression ratio than other solutions while having less bandwidth-reduction gain, are removed. Since the picture quality as function of the compression ratio is generally a monotonic decreasing function, it can be concluded that the removed solutions have a lower picture quality than the remaining solutions while consuming more bandwidth. After the pruning process all remaining solutions satisfy the following condition:

$$\left(\forall_{\bar{o}_t(M_{S_1},N_{S_1},V),\bar{o}_t(M_{S_2},N_{S_2},V)} : S_1 \in (B..4B], S_2 \in (S_1..4B] : \bar{o}_t(M_{S_1}, N_{S_1}, V) > \bar{o}_t(M_{S_2}, N_{S_2}, V)\right). \tag{7}$$

The remaining data-unit configurations represent feasible solutions and are shown in Fig. 6. The label at each point indicates the optimal data-unit dimension giving the result. This picture enables a tradeoff between compression ratio and bandwidth reduction. Note that data units of 16×8, reduce the relative transfer bandwidth from 172% to 106%. This compression ratio provides the largest amount of bandwidth reduction per unity of compression. Moreover, the data-unit dimensions are powers of two, which makes it even more attractive from an implementation point of view.

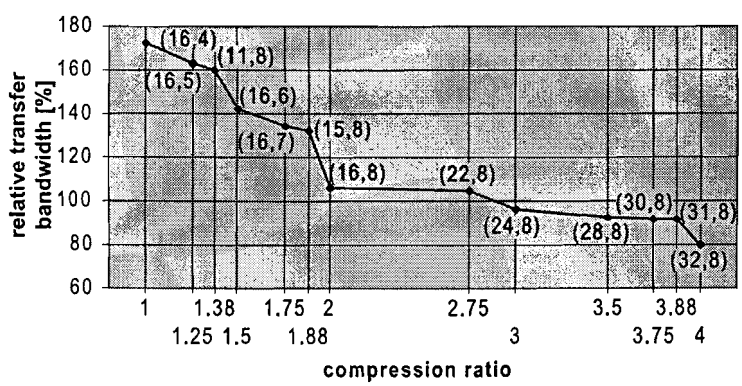


Figure 6: Feasible data-unit configurations for compression into 64-byte data bursts.

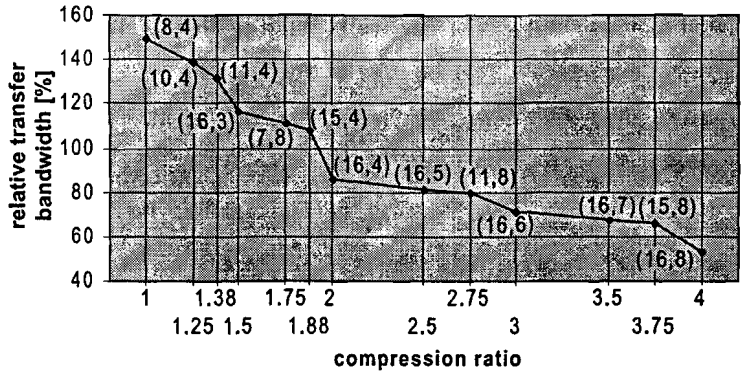


Figure 7: Feasible data-unit configurations for compression into 32-byte data bursts.

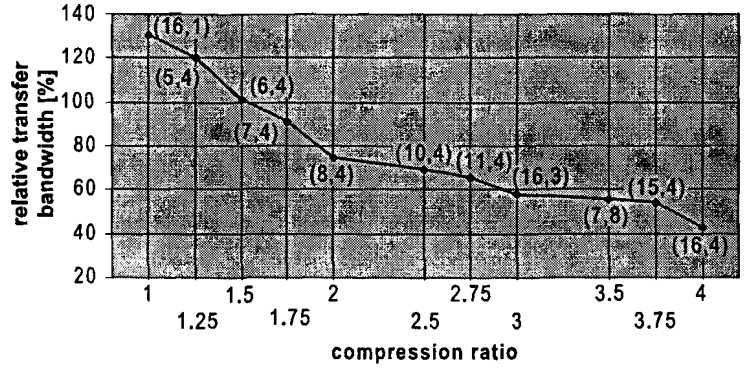


Figure 8: Feasible data-unit configurations for compression into 16-byte data bursts.

To explore the complete design space, the above-described method is also applied to an MPEG-2 decoder system with a external memory having 32-byte and 16-byte data bursts; i.e. $B = 32$ and $B = 16$, respectively. The outcomes are depicted in Fig. 7 and Fig. 8. Notice that the graphs in Fig. 6, 7 and 8 look very similar. For example, in all figures, the maximum amount of bandwidth reduction per compression unit is positioned at compression ratio two. For this compression ratio, the optimal data-unit dimensions are aligned with the MB grid in both horizontal and vertical direction, thereby enabling memory requests with low overhead at relatively high probability. Moreover, also the size of the data unit is an important parameter for feasibility. A data unit of 16×8 can easily be compressed with a factor two, while maintaining a high picture quality. For data units of 8×4 this is less straightforward.

6. RESULTS AND CONCLUSIONS

Many current multimedia systems intended for applications such as e.g. MPEG coding and 3-D graphics rendering, feature double-data-rate (DDR) SDRAM with bus widths up to 64 bits. These expensive memory configurations are adopted to obtain sufficient communication bandwidth, which is demanded by the continuous increase of computational power. This paper shows the feasibility of embedded compression for reducing the previously mentioned costly bandwidth bottleneck.

Our experiments with an MPEG-2 decoder show that the amount of data transferred to and from the memory is 172% of the data that is strictly required for the decoding, using the optimal mapping of groups of pixels (data units) to be compressed into data bursts. However, in most currently used systems a straightforward linear mapping of 64×1 pixels into data bursts is applied, resulting in a relative transfer bandwidth of even 341% [1][2]. Due to the trend of increasing bandwidth requirements, the memory data bus is becoming wider. Consequently, the size of the data bursts grows, leading to even more transfer overhead. Fortunately, larger

blocks can be decorrelated more efficiently, which makes the use of embedded compression for memory bandwidth reduction increasingly attractive.

Unfortunately, it has been found that compression not necessarily leads to reduction of the memory bandwidth. Due to the block-based storage of compressed data entities, the bandwidth may even be increased instead of reduced. This phenomenon limits the amount of feasible solutions for using embedded compression. Moreover, it has also been found that certain data-unit sizes offer less bandwidth reduction than other solutions, while giving less picture quality. Obviously, these solutions are therefore not attractive and have been omitted for feasibility. This has resulted in a limited amount of feasible compression ratios corresponding with a limited amount of optimal data-unit dimensions.

For our consumer application, it is required to provide a low-cost compression scheme to reduce bandwidth without sacrificing visual picture quality. For a compression factor four, a bandwidth reduction is established of 53%, 64% and 67% for 64-byte, 32-byte and 16-byte data bursts, respectively. However, for all three burst sizes, a compression ratio of two gives the largest amount of bandwidth reduction per unit of compression. For this compression factor an algorithm from Bayazit *et al.* [9] that is based on adaptive DPCM has been proven to be feasible. We propose to slightly modify the algorithm by matching the dimensions of the independent data units to the properties of the burst-oriented SDRAM memory, thereby enhancing the bandwidth reduction from 12.5% to approximately 40%. More precisely, for a compression ratio of two, the bandwidth is reduced with 38%, 42% and 43% for 64-byte, 32-byte and 16-byte data bursts, respectively. Summarizing, our data-dependent communication model decreases the memory bandwidth of the MPEG decoder from 341% to 172% and the embedded compression accounts for a further reduction to 106%, thereby gaining a total bandwidth reduction of 69%.

With respect to picture quality, the following conclusion applies. For data-burst sizes of 64 and 32 bytes, we compress 128 and 64 bytes, instead of the 48 bytes as presented in [9]. Therefore, we can expect the picture-quality degradation in the MPEG decoder caused by the embedded compression to be less than 0,34 dB as indicated in the same paper.

A side benefit of using an embedded compression scheme with a fixed compression ratio is that it reduces the required memory size proportionally to the compression ratio. Thus where usually three picture memories are required for MPEG decoding, only half of it is necessary when applying embedded compression with a factor two.

The proposed techniques for reducing the bandwidth bottleneck of external memory can be applied to a broad class of block-based video processing applications. When adopting embedded compression for decoding of high-definition video pictures, the bandwidth reduction is as high as 165 or 275 MB/s for a compression ratio of two or four, respectively. Since this substantial improvement in memory bandwidth is guaranteed, it can be exploited easily for reducing the system cost, to improve the picture quality, or to extend the functionality.

REFERENCES

1. E.G.T. Jaspers and P.H.N. de With, "Bandwidth optimization for video-based consumer systems," in *Digest of Technical Papers of IEEE Int. Conf. Cons. Electr.*, pp. 72–73, June 2001.
2. E.G.T. Jaspers and P.H.N. de With, "Bandwidth optimization for video-based consumer systems," *IEEE Trans. Cons. Electr.* , to appear in Nov. 2001 / Febr. 2002.
3. S. Roy, R. Kumar and M. Prvulovic, "Improving system performance with compressed memory," in *Proc. of Int. Parallel & Distr. Proc. Symposium - IPDPS*, April 2001.
4. R.B. Tremaine *et al.*, "Pinnacle: IBM MXT in a memory controller chip," *IEEE Micro* **21**, pp. 56–68, March-April 2001.
5. P.H.N. de With, P.H. Frencken and M. v.d. Schaar-Mitrea, "An MPEG decoder with embedded compression for memory reduction," *IEEE Trans. Cons. Electr.* **44**, pp. 545–555, Aug. 1998.

6. J. Tajime, *et al.*, "Memory compression method considering memory bandwidth for HDTV decoder LSIs," in *Proc. of IEEE Int. Conf. on Image Proc.*, **2**, pp. 779–782, Oct. 1999.
7. M. v.d. Schaar-Mitrea and P.H.N. de With, "Novel embedded compression algorithm for memory reduction in MPEG codecs," in *Proc. SPIE Vis. Commun. & Image Proc.*, **3653-2**, pp. 864–873, Jan. 1999.
8. M. v.d. Schaar-Mitrea and P.H.N. de With, "Near-lossless embedded compression algorithm for cost reduction in DTV receivers," *IEEE Trans. Cons. Electr.* **46**, pp. 923–933, Nov. 2000.
9. U. Bayazit, L. Chen, and R. Rozploch, "A novel memory compression system for MPEG-2 decoders," in *Digest of Technical Papers of IEEE Int. Conf. Cons. Electr.*, pp. 56–57, June 1998.
10. H. Kim and I. Park, "Array address translation for SDRAM-based video processing applications," in *Proc. SPIE Vis. Commun. & Image Proc.*, **4067-2**, pp. 922–931, June 2000.