Multiple Description Coding — a new technology for video streaming over the Internet

Andrea Vitali
STMicroelectronics

The Internet is growing quickly as a network of heterogeneous communication networks. The number of users is rapidly expanding and bandwidth-hungry services, such as video streaming, are becoming more and more popular by the day. However, heterogeneity and congestion cause three main problems: unpredictable throughput, losses and delays. The challenge is therefore to provide: (i) quality, even at low bitrates, (ii) reliability, independent of loss patterns and (iii) interactivity (low perceived latency) ... to many users simultaneously.

In this article, we will discuss various well-known technologies for streaming video over the Internet. We will look at how these technologies partially solve the aforementioned problems. Then, we will present and explain *Multiple Description Coding* – which offers a very good solution – and how it has been implemented and tested at STMicroelectronics.

Packet networks [1][2]

Heterogeneity adds up with errors and congestion: backbone and wired links have an increasing capacity while, at the same time, more and more low-bandwidth error-prone wireless devices are being connected.

Throughput may become unpredictable. If the transmission rate does not match the capacity of the bottleneck link, some packets must be dropped. The delivery system may provide prioritisation: the most important packets are given a preferential treatment, while the least important packets are dropped first. However, usually networks will drop packets at random. Packet loss probability is not constant; on the contrary, it can be wildly varying, going from very good (no loss) to very bad (transmission outages).

This makes the design of the delivery system very difficult. Usually there are two options:

- the system can be designed for the worst case;
- or it can be made adaptive.

If it is designed for the worst case, it will be inefficient every time the channel is better than the worst case, i.e. most of time. Conversely, if it is designed to be adaptive, it will most probably adapt too late.

Data-independent content delivery technologies

ARQ: Automatic Repeat reQuest

One of the most effective techniques for improving reliability is the retransmission of lost packets: Automatic Repeat reQuest, or ARQ. TCP-based content delivery is based on this.

If losses are sporadic, this technique is very efficient: packets are successfully sent only once. On the other hand, if losses are frequent, retransmissions can even increase congestion and also the loss rate, a vicious cycle (this is avoided in TCP-based content delivery).

Retransmission is very useful in point-to-point communications where a feedback channel is available. However, when broadcasting to many receivers, the broadcaster cannot handle all the independent retransmission requests.

The added delay of the retransmission is at least one round-trip transport time. But each retransmission can also be lost, and the delay can be arbitrarily large. This is critical for streaming video: the delay of a retransmitted packet may be much longer than inter-arrival times and, as a consequence, streaming may suffer stalls. This delay adds up in the receiver buffer which must be large enough to compensate for variation in the inter-arrival times (jitter).

FEC: Forward Error Correction / Erasure Recovery

Another very effective technique is channel coding, i.e. the transmission of redundant packets that allow recovery of erroneous / lost packets at the receiver side: Forward Error Correction / Erasure Recovery, or FEC.

If the loss rate is known, the added redundancy can be made just enough to compensate. Unfortunately, in the real world not only the amount of losses is not known, but also it is wildly time-varying. This, coupled with the fact that this technique has an all-or-nothing performance, makes its use very difficult: either errors are too much or they are less than expected.

If losses are too much, the recovery capability will be exceeded. Added redundancy will not be enough and the losses will not be recovered. Decoded quality will be very bad (cliff effect). Because of this, to be safe, broadcasters typically consider the worst case and choose to increase the amount of redundancy at the expense of the video. The video is compressed more heavily, lowering the final decoded quality.

If errors are less than expected, which is probable when the system is designed for the worst case, the losses will be recovered. The decoded quality will be guaranteed, unaffected by loss patterns. However capacity is wasted: less redundancy could be used leaving room for a higher-quality lightly-compressed video. Adaptation could be used in principle to dynamically balance the added redundancy and video compression, but it is rarely done because of the difficulty. Decoded quality is lower than it is possible to get.

The complexity can be very high: encoding and decoding of redundant packets requires memory and computational power. Efficient schemes for error correction / erasure recovery require processing of a large number of video packets. Therefore the added delay is not arbitrarily large, but it can be significant.

Data-dependent content delivery technologies

Robust source coding

The more efficient the video encoder, the more important a video packet is. When compression efficiency is very high, the loss of a packet has potentially a devastating effect. Then, a heavy recovery mechanism, such as complex FEC codes, must be used to reduce the probability of this happening.

Conversely, when the compression efficiency is low, the loss of a packet has little effect. In this case, concealment techniques do exist that can reduce or even completely hide the effect of the loss. In this case, a light recovery mechanism can be used.

Therefore, compression efficiency should be tuned *carefully*, taking into account the effect of losses, the effectiveness of concealment techniques and the effectiveness of the recovery mechanism. The available bandwidth can then be optimally split between the video data and redundant data.

Said in other words, it is always useful to optimize the parameters of the source encoder and of the channel encoder jointly (a technique known as "joint source-channel coding"). In the case of multimedia communications, this means exploiting the error resilience that may be embedded in compressed multimedia bitstreams, rather than using complex FEC codes or complex communication protocols.

Video encoders use a bunch of techniques to efficiently squeeze the video: prediction (also known as motion estimation and compensation), transform, quantization and entropy coding. Prediction is one of the most important techniques from the point of view of compression efficiency: the current video is predicted from the previously transmitted video. Because of this, video packets are dependent on previous packets. If these packets have not been successfully received, then the current packet is not useful. This is known as loss propagation. Compression efficiency can be a trade-off for robustness by reducing the amount of prediction (i.e. more intra coding): dependencies will be reduced, stopping the loss propagation effectively.

Transmission delay can also be a trade-off for robustness. Video packets can be reorganized (in so-called "interleaving buffers") so that consecutive video packets do not represent neighbouring video data. This is done to delocalise the effect of losses and ease the concealment. A long burst of lost packets will affect portions of the video which are far apart from each other. Lost portions can then be concealed effectively by exploiting neighbouring video data.

Concealment is usually done blindly at the receiver side. However, the transmitter can encode hints (concealment data) that increase its effectiveness. Obviously this consumes part of the available bandwidth.

All these techniques are very effective, but it is very difficult to choose an optimal set of parameters. It is especially difficult when there are many receivers which experience different channel conditions.

Multiple Description Coding [3][4]

Multiple Description Coding (MDC) can be seen as another way of enhancing error resilience without using complex channel coding schemes. The goal of MDC is to create several independent descriptions that can contribute to one or more characteristics of video: spatial or temporal resolution, signal-to-noise ratio, frequency content. Descriptions can have the same importance (balanced MDC schemes) or they can have different importance (unbalanced MDC schemes).

The more descriptions received, the higher the quality of decoded video. There is no threshold under which the quality drops (cliff effect). This is known as "graceful degradation".

The robustness comes from the fact that it is unlikely that the same portion of the same picture is corrupted in all descriptions. The coding efficiency is reduced depending on the amount of redundancy left among descriptions; however channel coding can indeed be reduced because of enhanced error resilience. Experiments have shown that Multiple Description is very robust: the delivered quality is acceptable even at high loss rates.

Abbreviations

ARQ	Automatic Repeat reQuest	**LC**	Layered Coding
FEC	Forward Error Correction	**MD**	Multiple Description
IF-PDMD	Independent Flux - Polyphase Downsampling Multiple Description	**MDC**	Multiple Description Coding
		TCP	Transmission Control Protocol

Descriptions can be dropped wherever it is needed: at the transmitter side if the bandwidth is less than expected; at the receiver side if there is no need, or if it is not possible to use all descriptions successfully received. This is known as “scalability”. It should be noted that this is a side benefit of Multiple Description Coding which is not designed to obtain scalability; instead it is designed for robustness.

Descriptions of the same portion of video should be offset in time as much as possible when streams are multiplexed. In this way a burst of losses at a given time does not cause the loss of the same portion of data in all descriptions at the same time. If interleaving is used, the same criterion is to be used: descriptions of the same portion of video should be spaced as much as possible. In this way a burst of losses does not cause the loss of the same portion of data in all descriptions at the same time. The added delay due to the amount of offset in time, or the interleaving depth, must be taken into account.

Layered Coding

Layered Coding (LC) is analogous to Multiple Description Coding (MDC). The main difference lies in the dependency. The goal of LC is to create dependent layers: there is one base layer and several enhancement layers that can be used, one after another, to refine the decoded quality of the base layer.

Layers can be dropped wherever required but they cannot be dropped at random: the last enhancement layer should be dropped first, while the base layer must never be dropped. If the base layer is not received, nothing can be enhanced by the successive layers. Layered Coding is designed to obtain this kind of scalability.

Repair mechanisms are needed to guarantee the delivery of at least the base layer. Moreover: because of the unequal importance of layers, repair mechanisms should unequally protect the layers to better exploit Layered Coding. However not all networks offer this kind of services (prioritization).

Recovery mechanisms and Layered / Multiple Description Coding

Channel coding is needed by Layered Coding. However channel coding can also be used with Multiple Description Coding. Generally speaking, it is better to adapt the protection level of a given description / layer to its importance, a technique commonly known as “unequal error protection”.

Unequal error protection is better even in the case of equally-important descriptions (balanced MDC). In fact, armouring only one description may be more effective than trying to protect all descriptions. If this is done, there is one description which is heavily protected. If the channel becomes really bad, this description is likely to survive losses. Then the decoder will be able to guarantee a basic quality, thanks to this description.

Summary of reviewed technologies and their characteristics

To summarize, here is an overview of the technologies that can be used for video streaming over packet networks, to compensate for losses due to errors and congestion:

Data-independent content delivery technologies

- **Automatic Repeat Request** (ARQ): suitable only for point-to-point, needs feedback, added delay arbitrarily large.
- **Forward Error Correction** (FEC): no feedback required, all-or-nothing performance (cliff effect), waste of capacity when tuned for worst case, complexity, significant added delay.

Data-dependent content delivery technologies

- **Robust Source Coding**: difficult to choose optimal parameters
- **Multiple Description Coding** (MDC): no cliff effect (graceful degradation), no prioritisation needed, allows scalability, very robust even at high loss rates
- **Layered Coding** (LC): requires prioritisation and recovery mechanisms, allows efficient scalability

It should be noted that packet networks are designed to deliver any kind of data: a data-independent technique is therefore always needed. The best option is Forward Error Correction / erasure recovery (FEC).

For multimedia data, such as video (and audio as well), several smart techniques exists. In this case the best option is Multiple Description Coding (MDC).

Standard-compatible Multiple Description Coding [6][8]

Losses due to errors and congestion do cause visible artefacts in decoded video: loss concealment techniques may help, but they are rarely effective, as can be seen in *Fig. 1*. This explains the need for an effective technique to recover losses and/or ease the concealment.

Figure 1
On the left, errors are not concealed. On the right, state-of-the-art concealment has been applied

Automatic Repeat reQuest (ARQ) is suitable only for point-to-point communications and cannot be easily scaled to broadcast scenarios; furthermore, it requires a feedback channel which may not be available. FEC is effective only if complex (which means: more power, delay, etc) and it has a threshold which yields an all-or-nothing performance (the cliff effect).

Robust source coding is difficult to use, as parameters are difficult to be tuned. Layered Coding is not designed for robustness and relies on the aforementioned recovery mechanisms. Conversely, Multiple Description Coding does not require a feedback channel and does not have an all-or-nothing behaviour: instead it has graceful degradation (more descriptions, more quality), plus it offers free scalability (to transmit as many descriptions as possible, receive as many as needed).

The question is: if Multiple Description Coding does serve the purpose well (robustness, effectiveness), then what is the price to be paid when implementing this solution (efficiency, bandwidth, quality, complexity, compatibility with legacy systems).

Standard compatibility

It is not easy to design and implement a Multiple Description video coding scheme. There are many established video coding standards deployed in the real world: e.g. MPEG-2, MPEG-4, H.263 and H.264. It is difficult to impose yet another standard which is more complex.

There are many other techniques available for creating multiple descriptions: multiple description scalar or vector quantization, correlating transforms and filters, frames or redundant bases, forward error correction coupled with layered coding, spatial or temporal polyphase downsampling (PDMD).

The best choice can be found by following this criteria:

- **Compatibility**: the possibility to use standard encoders for each description and the possibility of being compatible with legacy systems;
- **Simplicity**: minimum added memory and computational power;
- **Efficiency**: for a given bandwidth and when there are no losses, the minimum loss of decoded quality with respect to the best quality delivered by standard coding.

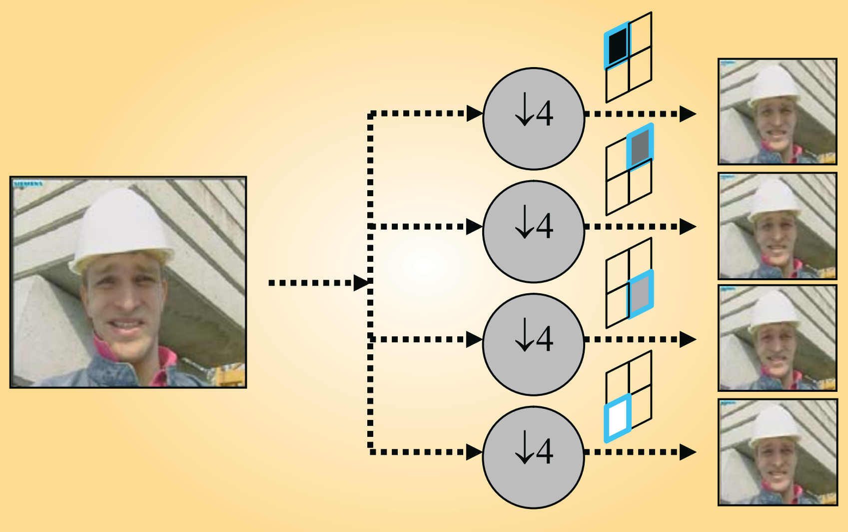

Figure 2
Pre-processing stage: downsampling in spatial domain. Odd and even lines are separated, the same is done for columns. Four descriptions are created.

Among the aforementioned techniques, polyphase downsampling is particularly interesting as it is very simple and it can be easily implemented using standard state-of-the-art video encoders.

The sequence to be coded is subdivided into multiple subsequences which can then be coded independently. This is done in a pre-processing stage *(Fig. 2)*. At the decoder side, there is a post-processor stage *(Fig. 3)* in which decoded subsequences are merged to recreate the original one. This simple scheme is also known as “Independent Flux Polyphase Downsampling Multiple Description” coding (IF-PDMD).

This scheme is completely independent of the underlying video encoder.

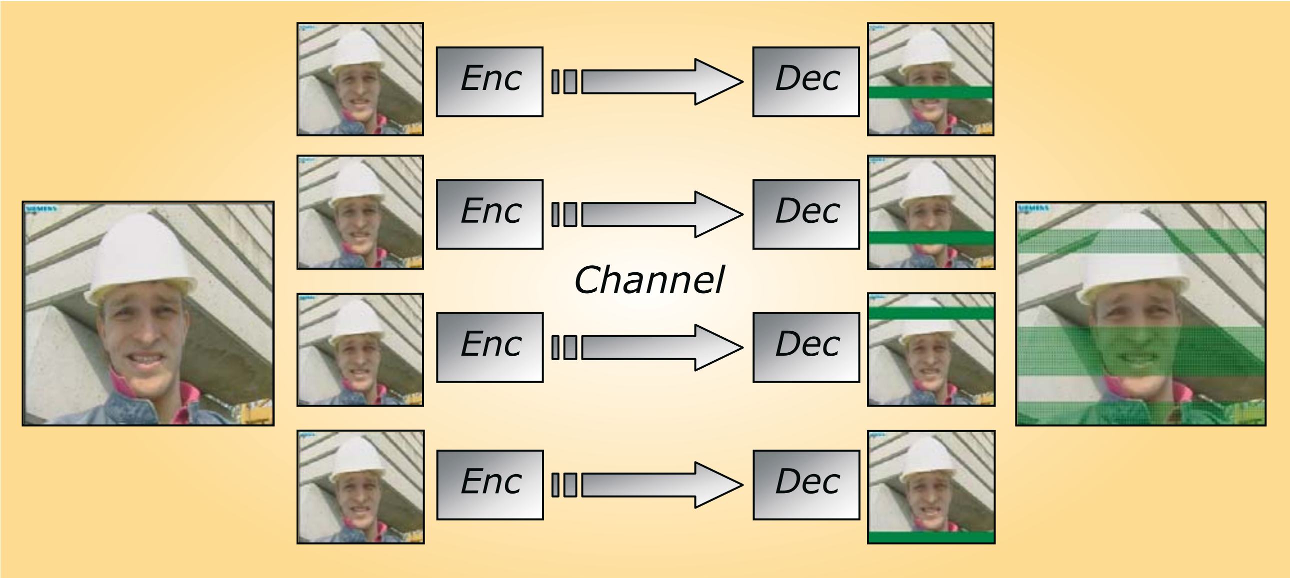

Figure 3
The whole chain: pre-processing, encoding, transmission, decoding, post-processing

Subdivision to create descriptions can be done along the temporal axis (e.g. by separating odd and even frames) or in the spatial domain (e.g. by separating odd and even lines). As encoding of each description is independent from others, there can be slight differences in the decoded quality. When temporal subdivision is used a potentially annoying artefact may arise: the difference among odd and even frames may be perceived as “flashing”.

On the contrary, when spatial subdivision is used *(see Fig. 4)*, a potentially pleasant artefact may arise: the difference between descriptions may be perceived as *dithering*, a known technique applied in graphics to hide encoding noise.

Figure 4
Dithering effect as a result of spatial downsampling: 4 descriptions are created by separating odd/even lines and taking every other pixel. As encoding of each description is independent from others, the decoded quality may differ slightly.

Spatial subdivision has two more advantages:

- Two descriptions can be created by separating odd and even lines: interlaced video is then reduced to two smaller progressive video streams which may be easier to encode.
- Four descriptions can be created by separating odd and even lines, and then separating odd and even columns: high definition video (HDTV) is then reduced to four standard definition video streams which can be encoded using existing encoders.

It should be noted that keeping Multiple Description Coding decoupled from the underlying codec prevents it from giving its best. To get maximum quality and to encode the descriptions with least effort, joint or coordinated encoding could be used. Also, to exploit the redundancy and to maximize the error resilience, joint Multiple Description decoding is recommended.

As an example, video encoders can share expensive encoding decisions (motion vectors) instead of computing them; also they can coordinate encoding decisions (quantization policies) to enhance the quality or resilience (interleaved multi-frame prediction policies, intra-refresh policies). Decoders can share decoded data to ease error concealment; also they can share critical internal variables (anchor frame buffer) to stop error propagation due to prediction.

Figure 5
30% packet loss;
left: **the output of a standard decoder, not aware of Multiple Description, has been instructed to see descriptions as replicas of the same packet (fake standard encoding);**
right: **the output of a Multiple Description decoder.**

It is worth mentioning that, if balanced descriptions are properly compressed and packed, any losses can be recovered before the decoding stage. In this case, decoders are preceded by a special processor in which lost packets are recovered by copying similar packets from other descriptions. Similar packets are those that carry the same portion of video data.

The scheme is also compatible with systems not aware of Multiple Descriptions *(see Fig. 5)*.

In fact, each description can be decoded by a standard decoder, which need not to be MD-aware in order to do this. Of course, if spatial MD has been used, the decoded frame has a smaller size ... while if temporal MD has been used, the decoded sequence has a lower frame rate.

Moreover, MD encoding can even be beneficial. In fact, multiplexed descriptions can be marked so that old decoders believe that they are multiple copies of the same sequence.

As an example, when four descriptions are transmitted, the old decoder will believe that the same video packet is transmitted four times. Actually, they are four slightly different packets, but this does not matter. The decoder can be instructed to decode only the first copy and, if this copy is not received correctly, it can be instructed to decode another copy.

Why use Multiple Description Coding?

Firstly: increased error resilience. Secondly: we get scalability for free.

Robustness

Multiple Description Coding is very robust, even at high loss rates *(see Fig. 6)*. It is unlikely that the same portion of a given picture is corrupted in all the descriptions. It's as simple as that!

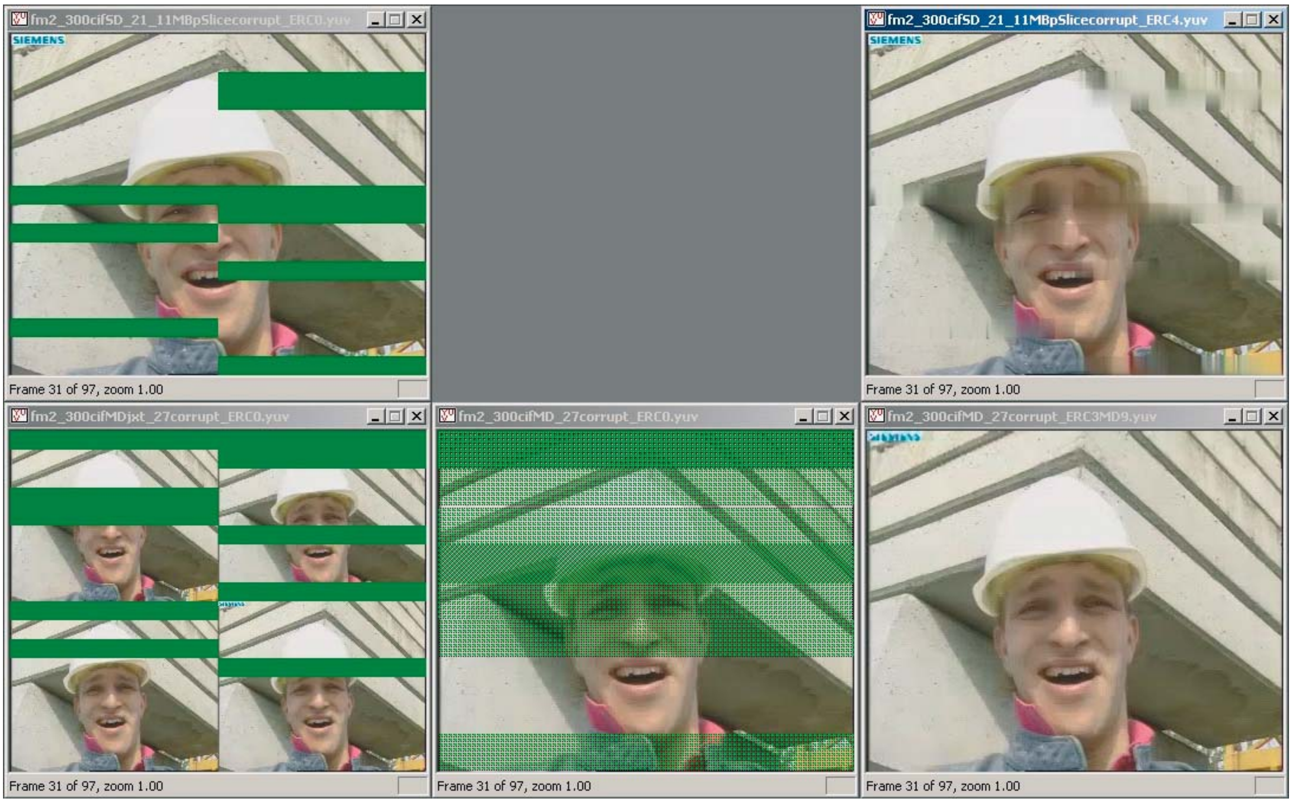

Figure 6
Same aggregate bandwidth, number of packets and average packet size, and with 30% packet loss rate.
Top row: **standard coding.** *Bottom row:* **four multiple descriptions generated by separating odd/even lines and taking every other pixel; before and after concealment.**

A more sophisticated point of view is to note that descriptions are interleaved. In fact, when the original picture is reconstructed, descriptions are merged by interleaving pixels. A missing portion in one description, will results in scattered missing pixels. These pixels can easily be estimated by using neighbouring available pixels.

It is assumed that errors are independent among descriptions. This is true only if descriptions are transmitted using multiple and independent channels. If one single channel is used instead, descriptions have to be suitably multiplexed. If this is done, error bursts will be broken by the demultiplexer and will look random, especially if the burst length is shorter than the multiplexer period.

Scalability

There are many scenarios where scalability can be appreciated. With mobile terminals in mind, when standard coding is used, the whole bitstream should be decoded and downsized to adapt it to the small display. Power and memory are wasted. Conversely, when Multiple Description is used, a terminal can decode only the number of descriptions that suits its power, memory or display capabilities.

Also, when the channel has varying bandwidth, it would be easy to adapt the transmission to the available bandwidth. Descriptions may simply be dropped. Instead, a non-scalable bitstream would require an expensive transcoding (re-encoding the video to fit the reduced available bitrate).

This kind of scalability should be compared to the scalability provided by Layered Coding: think about losing the base layer while receiving the enhancement. It happens that the received enhancement is useless and bandwidth has been wasted. Usually, in order to avoid this, the base layer is given a higher priority or is more protected than the enhancement layer.

When MD coding is used, there is no “base” layer. Each description can be decoded and used to get a basic quality sequence. More decoded descriptions lead to higher quality. There is no need to prioritise or protect a bitstream.

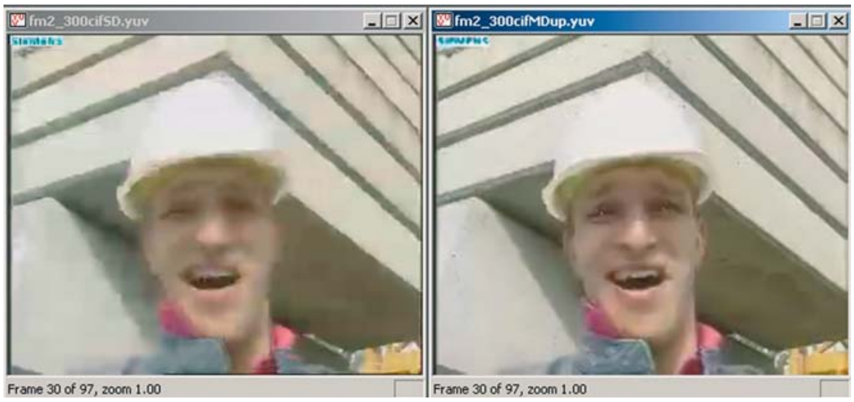

Figure 7
Site foreman, CIF resolution (352x288 pixels) at 155 kbit/sec using MPEG-4/10 encoder
Standard coding on the *left*, one out of four multiple descriptions on the *right*.

Finally, it must be noticed that at very low bitrates the quality provided by Multiple Description Coding is greater than that provided by standard coding. This happens because the low bitrate target can easily be reached by simply dropping all descriptions except one. On the contrary, with standard coding a rough quantization step must be used. Artefacts introduced by heavy quantization are more annoying than artefacts introduced by dropping descriptions *(see Fig. 7)*.

Why not use Multiple Description Coding?

At a given bitrate budget, there is a quality loss with respect to standard (single description) coding. The loss depends on the resolution (the lower the resolution, the higher the loss) and on the number of descriptions (the more the descriptions, the higher the loss).

Descriptions are more difficult to encode. Prediction is less efficient. If spatial downsampling is used, pixels are less correlated. If temporal downsampling is used, motion compensation is not accurate because of the increased temporal distance between frames.

Also, syntax is replicated among bitstreams. Think about four descriptions. There are four bitstreams. Each holds data for a picture which has 1/4th the original size. When taken all together, the four bitstreams hold data for the same quantity of video data as the single description bitstream. The bit-budget is the same. However, the syntax is replicated, therefore there is less room for video data.

However, it must be noted that it is not fair to compare the decoded quality of Multiple Description Coding with standard (single description) coding – when there are no losses.

Standard coding has been designed for efficiency while Multiple Description Coding has been designed for robustness. If there are no losses, this increased error resilience is useless. A fair comparison would be to compare error-resilient standard coding with Multiple Description Coding. As an example, the standard bitstream can be made more error resilient by reducing the amount of prediction (increased intra refresh).

The intra refresh should be increased until the quality of the decoded video is equal to the quality of decoded Multiple Description. Then it would be possible to evaluate the advantage of using Multiple Description by letting the packet loss rate increase and see which coding is better.

Experiments have shown [5] that Multiple Description is still superior when compared to error-resilient standard coding, even if the packet loss rate is very low (~1%). Simulations have been done at the same aggregate bitrate and same decoded quality using one of the most efficient FEC schemes: Reed-Solomon (R-S) codes *(see Fig. 8)*.

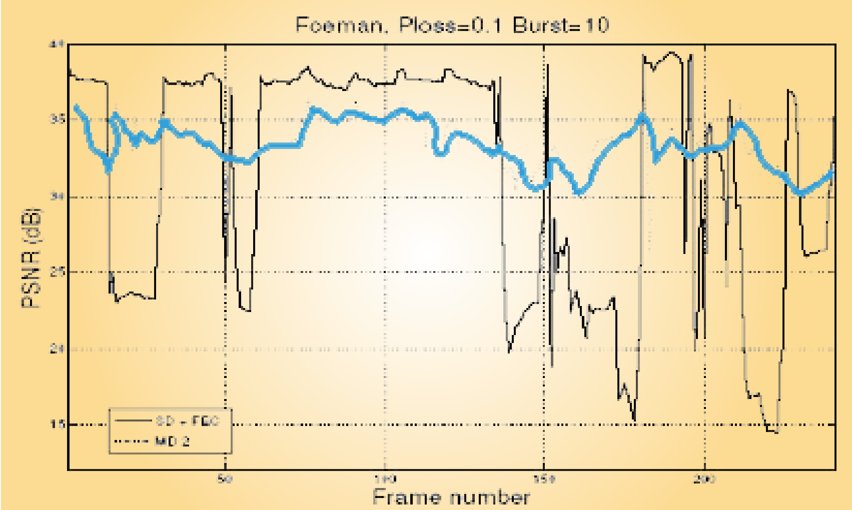

Figure 8
Quality frame-by-frame: the black line corresponds to standard coding protected by Reed-Solomon forward error correction (all-or-nothing behaviour), the blue line corresponds to two Multiple Descriptions (slightly lower average quality, but much lower variance).

From a higher point of view, we might decide to reduce channel coding and use part of its bit-budget for Multiple Descriptions bitstreams, therefore increasing the quality of the decoded Multiple Descriptions.

Foreseen applications of Multiple Description Coding

- Divide-and-rule approach to HDTV distribution: HDTV sequences can be split into SDTV descriptions; no custom high-bandwidth is required.
- Easy picture-in-picture: with the classical solution, a second full decoding is needed plus downsizing; with MDC/LC, it is sufficient to decode one description or the base layer and paste it on the display.
- Adaptation to low resolution/memory/power: mobiles decode as many descriptions/layers as they can – based on their display size, available memory, processor speed and battery level.
- pay-per-quality services: the user can decide at which quality level to enjoy a service, from low-cost low-resolution (base layer or one description only) to higher cost high-resolution (by paying for enhancement layers / more descriptions).
- Easy cell hand-over in wireless networks: different descriptions can be streamed from different base stations exploiting multi-paths on a cell boundary.
- Adaptation to varying bandwidth: the base station can simply drop descriptions/layers; more users can easily be served, and no trans-coding process is needed.
- Multi-standard support (simulcast without simulcast): descriptions can be encoded with different encoders (MPEG-2, H.263, H.264); there's no waste of capacity as descriptions carry different information.
- Enhanced carousel: instead of repeating the same data over and over again, different descriptions are transmitted one after another; the decoder can store and combine them to get a higher quality.

Application to P2P (peer-to-peer) networks

In P2P networks users help each other to download files. Each file is cut into pieces. The more popular a file is, the greater the number of users that can support a given user by transmitting the missing pieces.

Streaming however is a different story. The media cannot easily be cut into pieces, and in any case the pieces should be received in the correct order from a given user to be useful for the playout.

Also, a typical user has greater downlink capacity than uplink capacity. Therefore (s)he is not able to forward all the data (s)he receives and cannot help other users that are willing to receive the same stream.

One of the most effective solutions for live streaming has been implemented by Octoshape [7]. This is their scheme:

- A video that would require 400 kbit/s is split into four streams of 100 kbit/s each.
- Therefore N redundant 100 kbit/s streams are computed, based on the original four streams; the user is able to reconstruct the video given any four streams out of the available streams (the four original and the N redundant streams) – this can be done using an (N,4) Reed-Solomon FEC.

Following this scheme, the typical user is able to fully use the uplink capacity even if it is smaller than the downlink capacity. Each user computes and forwards as many redundant streams as possible, based on the capacity of its uplink.

A very similar scheme can be implemented using Multiple Description Coding:

- Four descriptions can be created by separating odd and even lines and taking every other pixels; each subsequence is encoded in 1/4th of the bitrate that would have been dedicated to full resolution video
- Redundant descriptions can be created by further processing video data; as an example: averaging the four aforementioned descriptions, and so on. This is known as frame expansion.

Frame expansion can easily be explained by this simple example: 2 descriptions can be generated by separating odd and even lines as usual; a 3rd description can be generated by averaging odd and even lines. It is clear that perfect reconstruction (except for quantization noise) is achieved if any 2 descriptions out of 3 are correctly received. Frame expansion can be seen as equivalent to a Forward Error Correction code with rate 2/3: one single erasure can be fully recovered (except for the quantization noise). However, unlike FEC, there is no threshold: if there is more than one erasure, received descriptions are still useful. Moreover, the redundancy can be controlled easily by quantizing the third description more heavily.

Conclusions

Two data independent content delivery techniques have been presented: Automatic Repeat reQuest (ARQ) and Forward Error Correction (FEC). The latter is preferable as it does not require a feedback from receivers and is then suited to broadcast. However this technique has an all-or-nothing performance: when the correction capability is exceeded the quality of decoded video drops.

Three data dependent content delivery techniques have been presented: robust source coding, Multiple Description Coding (MDC) and Layered Coding (LC). The latter is also known as Scalable Video Coding (SVC) as it allows efficient scalability: layers can be decoded one after another, starting from the base layer; layer have different importance and require prioritisation which may not be available in the network. Robust source coding exploit the resilience that can be embedded in the bitstream by tuning coding parameters; however it is very difficult to optimize. Multiple Description Coding allows scalability (transmit or decode as many descriptions as possible), does not

Andrea L. Vitali graduated in Electronics at the Milan Polytechnic in 1999. He then joined STMicroelectronics as a designer of digital multi-standard decoders for analogue TV. In 2000 he moved to the company's System Technology labs to work on real-time hardware prototyping for video algorithms. He has also worked on non-standard still picture / bayer-pattern compression and on Multiple Description video coding, and has published several papers on these topics. He holds patents granted in Europe and the USA in digital video processing, still picture compression, digital modulators and silicon sensors (automotive market).

Mr Vitali is now working in the field of robust source coding, joint source channel coding, adaptive multimedia play-out, metadata for multimedia signals, and graphical interfaces. He gave lectures on Digital Electronics at Pavia Polytechnic in 2002 and, since 2004, has also been an external professor at Bergamo University, Information Science department, where he is teaching Microelectronics.

require prioritisation, it is very robust (it is unlikely to lose all descriptions) and has no all-or-nothing behaviour (decoded descriptions all contribute to decoded video quality).

A standard-compatible Multiple Description Coding scheme has been presented: descriptions are created by spatial downsampling in a pre-processing stage prior encoding, they are merged after decoding in a post-processing stage. MDC performance has been compared to standard coding protected by state-of-the-art FEC: peak quality of decoded video is lower but it is much more stable (absence of cliff effect). Several foreseen applications have been listed, including applications in peer-to-peer networks.

References

[1] F. Kozamernik: **Webcasting – the broadcasters' perspective**
EBU Technical Review No. 282, March 2000

[2] F. Kozamernik: **Media Streaming over the Internet – an overview of delivery technologies**
EBU Technical Review No. 292, October 2002.

[3] V.K. Goyal,: **Multiple Description Coding: Compression Meets the Network**
IEEE Signal Processing Magazine, September 2001.

[4] N. Franchi, M. Fumagalli, R. Lancini and S. Tubaro: **Multiple Description Video Coding for Scalable and Robust Transmission over IP**
PV conference 2003

[5] R. Bernardini, M. Durigon, R. Rinaldo and A. Vitali: **Comparison between multiple description and single description video coding with forward error correction**
MSP 2005.

[6] A. Vitali, M. Fumagalli, draft-vitali-ietf-avt-mdc-lc-00.txt: **Standard-compatible Multiple-Description Coding (MDC) and Layered Coding (LC) of Audio / Video Streams**
www.ietf.org, July 2005.

[7] S. Alstrup and T. Rauhe: **Introducing Octoshape – a new technology for streaming over the Internet**
EBU Technical Review No. 303, July 2005.

[8] A. Vitali, A. Borneo, M. Fumagalli and R. Rinaldo: **Video over IP using standard-compatible Multiple Description Coding: an IETF proposal**
PV conference 2006.