

Mobile video-streaming

M D Walker, M Nilsson, T Jebb and R Turnbull

This paper describes well-known and new techniques for streaming video to a mobile device. It demonstrates that realistic video services are attainable for GPRS-enabled mobile handsets, by using a system developed at BT called 'Fastnets'. A review of current mobile streaming systems is provided, and the advantages of the Fastnets system are also described.

1. Introduction

Mobile video has been lauded as one of the key service innovations that justified the heavy investment in 3G licences across Europe some three years ago. While 3G undoubtedly has the capacity to provide compelling video-based applications, existing 2G-based technologies such as general packet radio service (GPRS) can be used to deliver mobile video services today.

This paper considers the state of the art for mobile video streaming; it discusses the international standards that are applicable for mobile streaming; and describes a mobile video-streaming system developed by BT, which enables high-quality video-streaming using both 3G and GPRS networks.

2. Technical details for mobile streaming

2.1 IP streaming principles for mobile

Real-time video applications require media packets to arrive in a timely manner — excessively delayed packets are useless and are treated as lost. This causes the synchronisation between encoder and decoder to be broken, and errors to propagate in the rendered video. The technical implication for IP networks is that the data representing the compressed audio and video cannot be considered as ordinary data files — transfer methods such as ftp are not appropriate. The actual transfer methods for streamed media are discussed later.

By way of example, consider an audio/video clip of duration one minute, that has been compressed at an average data rate of 20 kbit/s. To stream this clip to a user, not only must the network provide an average throughput of 20 kbit/s, but the instantaneous throughput of the link must not fall below an average of 20 kbit/s, for the duration of the streaming. However, IP is non-deterministic — without quality of service (QoS) guarantees, there is no

way of knowing for sure what the future bandwidth characteristics of the network link will be.

This problem is exacerbated for delivery to a mobile device — mobile IP networks, particularly GPRS, provide very low and variable data throughput. The device itself may not have sufficient processing power to employ complex error concealment and recovery schemes. This can lead to a poor experience when viewing streamed media that has not been designed for mobile networks.

2.2 Common techniques to improve performance

Typically, streaming systems employ a number of well-known techniques to provide resilience to network congestion. These include the insertion of regular key-frames within the data-stream to provide re-synchronisation points should timeliness be lost, at the cost of reduced compression efficiency. A second common technique is to employ buffering at the player — delay is traded for packet loss and jitter: provided the average throughput requirements of the video-stream match the average available bandwidth, the receiver buffer size can be dimensioned to accommodate the expected jitter, and allow for retransmissions. In fact, buffering can also be used to artificially increase the perceived throughput of the network link, by forcing the player to buffer a large proportion of the entire clip before playback starts, but at the cost of high start-up delay.

2.3 Compression

Compression is a method for reducing the amount of data required to represent digital media. In the same way that a utility such as WinZipTM is popular for compressing documents, so there are techniques to compress audio and video data. The higher the degree of compression, the lower are the storage and transmission requirements for audio/video clips. Compression is achieved using an encoder/decoder pair, termed a 'codec'.

2.3.1 Video compression

This section gives an overview of standard video compression techniques. Standards-based video codecs fall into the category of block-based interframe prediction algorithms. Such systems are designed to reduce the data rate of video by exploiting spatial and temporal redundancies between pictures, and by removing features of the video that are less perceptible by the human visual system (HVS).

A generic block-based interframe prediction video coder has the following components:

- motion estimator — previous pictures searched for similarities to the current picture (motion tracking),
- inter-frame loop — codes differences between current and previous pictures,
- transform — picture pixel differences are transformed to frequency coefficients,
- quantiser — frequency coefficients are quantised, with the least significant bits being discarded,
- entropy coding — quantised frequency coefficients are then entropy coded.

The combination of motion tracking and inter-frame coding removes temporal redundancy. The frequency transform puts the resulting pixel differences into a better format for the removal of spatial redundancy, which is achieved by the combination of quantisation (a lossy process) and entropy coding (the lossless representation of the result of quantisation — similar in concept to morse code in that short codes are used for frequent symbols and long codes for less frequent symbols). The encoder also replicates some decoder functionality to allow the encoder to track the operation of the decoder, and produce 'previous' pictures for future motion tracking.

For a fuller description of standards-based video encoding, see Ghanbari [1].

In Fig 1, a generic inter-frame video encoder is shown; the diagram also shows which components are fixed by video codec standards, and which components are open to allow implementor value-add. An example of value-add is given by noting that a valid bitstream can be generated by only coding pictures in the 'intra' mode (i.e. with no reference to previous pictures). However, this will typically require five times as many bits to achieve the same quality as inter-frame coding with motion compensation.

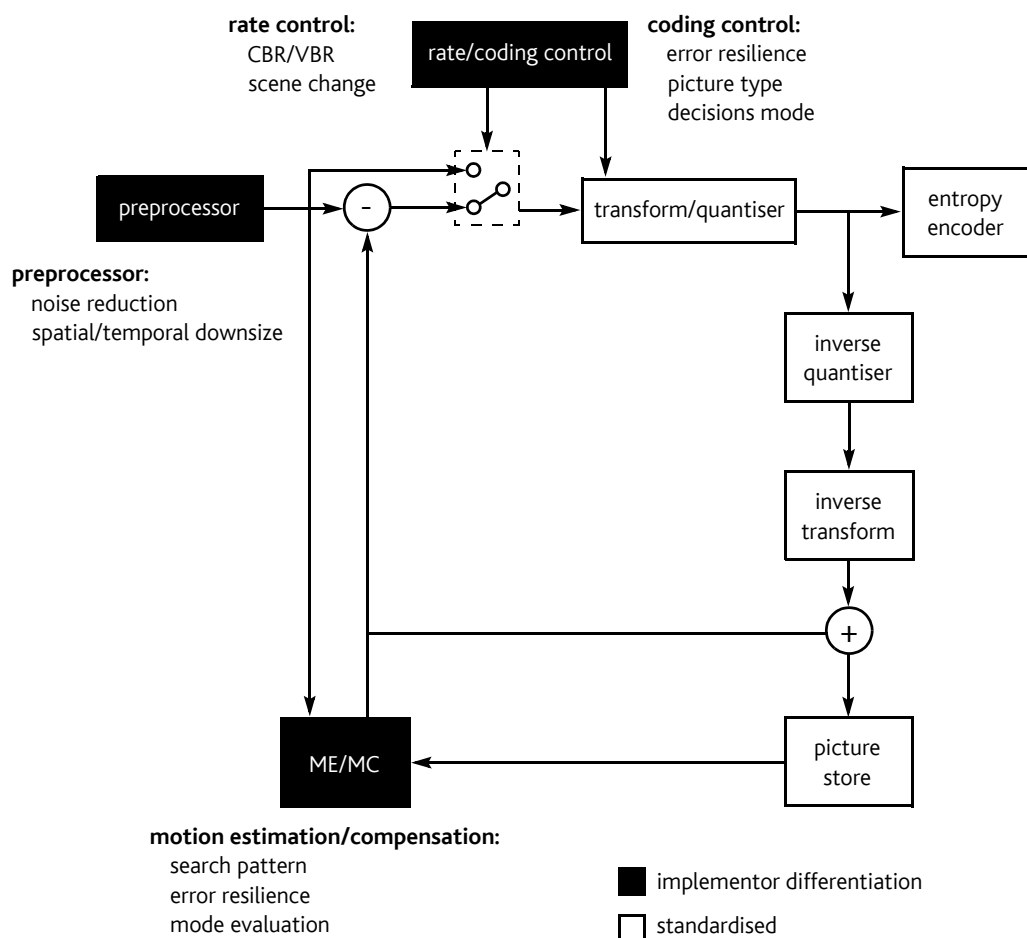


Fig 1 Standard and non-standard components of a video encoder.

Figure 2 shows the components of a video decoder. While the majority of picture-quality gains can be made at the encoder, the implementation at the decoder can include intelligent error masking, and post-processing to increase the quality of the decoded pictures.

Consequently, a vendor's implementation of a video standard may be good, bad or indifferent, depending on the nuances of their coding decisions and pre-/post-processing options.

Video compression standards

International standards on video compression are essential for interoperability. There are two well-known standardisation bodies that have developed video compression standards: the International Telecommunication Union (ITU-T), which has developed the H.26x series of Recommendations, and the International Standards Organisation (ISO), whose Moving Picture Experts Group (MPEG) has developed the MPEG series of standards. The ITU-T has produced the following Recommendations:

- H.261 (also called 'P times 64'), the first digital video coding standard with practical success,
- H.262 (common text with MPEG-2), for high-quality videoconferencing,
- H.263 (and its updates: 'H.263+' and 'H.263++') for videoconferencing and streaming.

MPEG has produced the following standards:

- MPEG-1 (ISO/IEC 11172-2) for video CD applications;
- MPEG-2 (ISO/IEC 13818-2) — used widely in DVD, broadcast, satellite, and cable DTV,
- MPEG-4 Visual (ISO/IEC 14496-2), for streaming applications (including compatibility with the H.263 baseline profile).

For mobile streaming applications, it is normally recommended that H.263/MPEG-4 are used. The similarity between the two coding schemes is acknowledged on the MPEG Web site [2]:

'Although the schedules and requirements for MPEG-4 video and ITU-T's video coding efforts are different, their requirements do overlap in many ways, for example both efforts seek at least to define a standard that can efficiently code natural video at bit rates in the range of 24 Kbit/s to 64 Kbit/s. Because of this overlap in requirements, many participants in MPEG-4 Video also participate in the ITU-T's low bit-rate coding (LBC) effort. Through these common members and through formal liaison statements the MPEG-4 Video and ITU-T frequently share their results with each other in hopes of improving their standards and achieving a high degree of interoperability between their standards'.

H.263 and MPEG-4 for 3GPP streaming

3GPP specifies H.263 and MPEG-4 may be used for video streaming, with compatibility between the two achieved at baseline. While this baseline compatibility is mandatory for 3GPP compliance, the baseline implementations only provide basic functionality. Both standards can be enhanced by use of different profiles/annexes. The implementor chooses which optional extras to add. For example, with MPEG-4 streaming, the scalability modes are popular as they enable mechanisms to provide an adjustment of the quality of video received by a user both efficiently and gracefully, depending on the perceived performance of the access network. It should be noted that the scalability mode on its own does not provide efficient and graceful degradation to changes in network performance — the application vendor must still measure the network performance, and then make decisions on the use of the scalability modes to adjust the quality of video being transmitted.

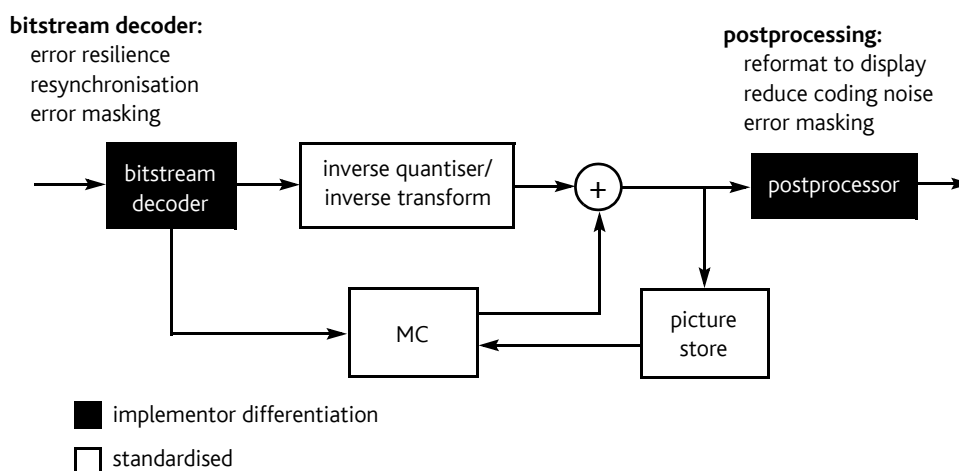


Fig 2 Standard and non-standard components of a video decoder.

H.264 video compression — the future of video compression

The ITU-T and ISO are currently creating a new video codec that is being heralded as a key enabling technology for mainstream use of video applications. The project is being developed by the JVT, and has proved very successful — standardisation is now almost complete. The standard will be known as both H.264 and MPEG-4 part 10. Many results for the new codec show that it requires about half the data-rate of H.263 and MPEG-4, and as little as a third of the data-rate for MPEG-2, to achieve the same picture quality [3, 4]. The H.264 design is architecturally equivalent to previous codecs as detailed above; however, by applying a 'back-to-basics' principle, the JVT has managed to refine each component to achieve this remarkable performance. The major differences between H.264 and H.263/MPEG-4 are described below.

- Motion Compensation

The motion compensation process differs from that of prior standard designs in its support of a wider variety of block shapes and sizes and in its fractional-sample interpolation filtering. Further enhancements are achieved by motion displacement accuracy of $\frac{1}{4}$ of a pixel¹ — meaning that motion is much more accurately tracked. In addition, multiple previously encoded and transmitted pictures are available for motion prediction.

- Transform

H.264 uses a 4×4 transform, in contrast with the 8×8 shape typically found in prior designs. The inverse transform is specified in terms of precise integer operations, rather than as a rounding tolerance relative to a real-valued ideal transform as found in most prior designs. In some cases the 4×4 shape is extended by applying an additional 2×2 or 4×4 transform to the DC coefficients of a first-stage transform.

- Entropy coding

Entropy coding is performed either by using context-adaptive variable length coding (CAVLC), or by use of context-based adaptive binary arithmetic coding (CABAC).

- Loop filter

A strong deblocking filter is used within the prediction loop.

The new codec standardisation process is due to be complete in mid 2003. As reported in IT Week in April 2003 [5]:

¹ Pixel: a picture element (also known as 'pel').

'... trial implementations and system tests have begun in earnest. Perhaps the most impressive demonstration was given by BT's research labs when it showed an H.264 decoder running on a Nokia 7650 mobile phone. BT's implementation uses Quarter Common Intermediate Format (QCIF), which offers a resolution of 176×144 pixels, and can decode at about 14 frames per second. The video bit rate was about 20 kbit/s, which can be handled by either GPRS or HSCSD mobile data technology.'

However, while the joint H.264/MPEG-4 part 10 coding scheme may become the *de facto* standard in the next few years, the 3rd Generation Partnership Programme (3GPP) [6] standardisation body currently specifies H.263 and MPEG-4 for video compression.

2.3.2 Audio compression standards

3GPP specify two codecs for the coding of speech and audio for a transparent end-to-end packet-switched streaming service. The codecs achieve compression by making assumptions about the signals such that they can exploit the redundancies and irrelevancies pertinent to speech or music.

Speech codecs are explicitly based on a model of speech production. The encoder uses a mathematical model of the human vocal system and by analysing the input signal, produces parameters describing the activity of the glottis and vocal tract. These parameters are transmitted to the decoder, where they are used to re-synthesise the original speech. This method is very effective at coding speech signals and can produce toll quality at bit rates as low as 4.8 kbit/s (e.g. AMR mode 0). 3GPP have specified that the AMR decoder shall be supported for narrowband speech [7] and the AMR wideband [8] operating at 16 kHz sampling for wideband speech.

The algorithms used for speech codecs rely on assumptions about the source signal. They are therefore not suitable for the effective coding of non-speech signals such as music. For this reason 3GPP have specified the use of the MPEG-4 AAC codec [9], which has been developed by MPEG. Like previous MPEG audio codecs such as the popular MP3 codec, it uses a perceptual noise shaping scheme, which uses the signal-masking properties of the human ear in order to achieve compression. AAC supports sample rates between 8 kHz and 96 kHz and any number of channels between 1 and 48, although 3GPP have specified an upper limit of 48 kHz and either mono or stereo operation only.

3. Streaming techniques

Current IP networks are not well suited to the streaming of multimedia content as they exhibit packet loss, delay, jitter and variable throughput, all of which can detract from the

end-user's enjoyment. Although quality of service schemes can reduce the severity of these effects, they are generally not widely deployed.

As IP networks generally have no connection admission control they can become congested causing a sustained drop in throughput. If transmission continues at the same rate, the result is high levels of packet loss and 'unfairness' to other network users — it is usual, and 'polite', to reduce transmission rate during congestion.

This section considers, in more detail, the international standards applicable to mobile streaming, and some well-known techniques for ameliorating the effects of network throughput variation and packet loss.

3.1 Video-streaming — congestion control

This section examines some of the popular methods of providing video-streaming over general IP networks. It should be noted that, while a good implementation of a video codec can provide value-add in terms of better picture quality for the same bit rate, the video packets must still be transmitted over a network to a receiver. It is this transmission that can differentiate a good implementation from a bad implementation.

A generic video-streaming architecture is shown in Fig 3.

The role of the multimedia server is to provide the player with multimedia data that has been transmitted across the network. Most server/client applications operate by employing an end-to-end congestion control mechanism, coupled with a buffering strategy at the player. This enables the player to supplement data received from the network with data that has been built up in this buffer. This provides better utilisation of the network, and better performance. However, it is not enough for the congestion control algorithm to simply measure the available network resources — the compression stage must include mechanisms to transmit the video at varying data rates, without compromising the requirement for timeliness of delivery. A streaming system may need to be able to reduce its transmission rate requirements, which

necessitates reducing the compressed rate of the media to be streamed, for which we use the term 'elastic'.

The use of real-time encoding would allow the bit rate to be varied dynamically, but is not practical due to the high processing power requirements which prevent such a solution from scaling to support many users. So to achieve a system that is scalable, media must be compressed without knowledge of the state of the network at transmission time, but, as soon as they are compressed, they become inelastic, and need to be transmitted at their encoded bit rates.

Layered encoding [10] is a well-known technique for creating elastic video sources, but such techniques are inefficient in terms of compression [11], and significantly increase the processing power required at the client which is undesirable for streaming to limited capability mobile devices.

Transcoding [12] is another technique for creating elastic video sources, which can be designed to have much lower computational complexity than video encoding. However, the computational complexity is not negligible, and so would have an impact on the scalability of the architecture.

3.2 Streaming protocols

The transport mechanism for streaming media is not adequately catered for [13] by either the transmission control protocol (TCP) [14], or the user datagram protocol (UDP) [15]. While it is clear that TCP is the protocol of choice for file-transfer type applications such as e-mail, and that UDP is the favoured transport protocol for conversational applications, there is no clear choice for streamed media. The standards bodies do not give many clues either — both UDP and TCP are supported by 3GPP. However, most 3GPP applications use UDP for streaming — even though research has shown that TCP may provide acceptable quality, at a lower application complexity [13].

3.2.1 UDP streaming

UDP is often the favoured transport protocol for delivery of audio and video data over IP networks. However, as UDP

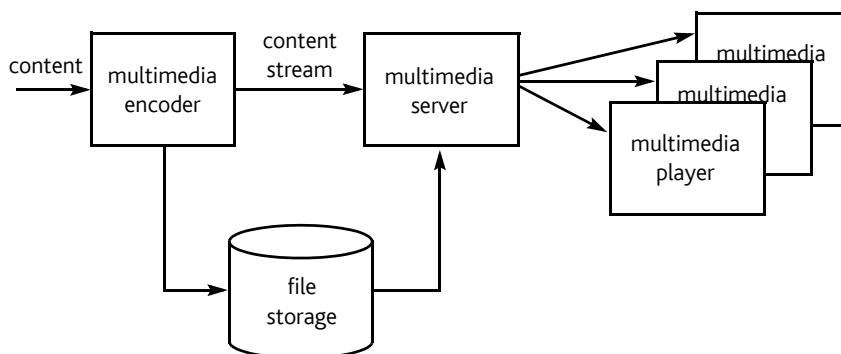


Fig 3 Generic end-to-end video transmission.

provides no guarantee of delivery, a video streaming application needs to be provided with extra information to enable the detection of lost, delayed and unordered packet delivery. UDP is usually supplemented with framing and feedback information, such as that provided by the real-time transport protocol (RTP) and the associated RTP control protocol (RTCP) [16].

RTP provides framing and timing structure to audio and video streams and allows inter-stream synchronisation. RTP provides additional information, such as payload type identification, sequence numbering and delivery monitoring, which can be used by the streaming application to detect, and act upon, transmission errors. It should be noted that RTP itself does not provide any mechanism to ensure timely delivery or provide other QoS guarantees, but relies on lower-layer services to do so. RTCP provides a mechanism for providing feedback to the source.

3.2.2 TCP streaming

While UDP is often the transport protocol of choice for many streaming applications, TCP can be preferred because of its relative implementation simplicity. For example, although TCP can offer no timeliness guarantees, there is now no requirement for extra framing information such as RTP and RTCP — with TCP, packets are guaranteed to eventually arrive. With TCP streaming, the application need only ameliorate the effects of delay, which is usually through the use of novel buffering techniques.

There is still much research comparing UDP to TCP for streaming applications [13] — but, with other schemes (see section 3.2.3) still in their infancy, the mix of solutions is likely to continue for the foreseeable future.

3.2.3 TCP-friendly streaming

While TCP and UDP can both provide some benefits to streaming technologies, each has drawbacks. There has recently been effort in the research community to define a 'half-way' solution — termed 'TCP-friendly' [17]; this congestion control mechanism is designed to be used for non-TCP applications. The goal of this research is to develop a congestion control scheme that is 'friendly' to other TCP traffic (i.e. a TCP-friendly source will fairly share network resource with TCP traffic) yet is more efficient and responsive than TCP.

Although a number of research papers have been published showing the benefits, this scheme is still relatively immature, and is not widely adopted.

4. Fastnets

This section provides an overview of the Fastnets² system, developed by BT. Fastnets is a multimedia streaming system designed to overcome the characteristics of IP

networks, and in particular mobile IP networks, to provide users with continuous multimedia of consistent quality with minimal start-up delay. H.264 video compression is used to achieve acceptable video quality over GPRS to mobile devices, such as the Nokia 7650.

4.1 Introduction

Our underlying philosophy when designing the Fastnets system was to develop technology that would enhance the end-user experience of streamed mobile video. In particular, Fastnets was designed to achieve 'Fast-start' streaming (reducing the delay normally associated with buffering) and to provide the best compression efficiency, without losing the ability to adapt to changes in network throughput. While the Fastnets system is an end-to-end streaming architecture, each technology component has been designed to 'plug-in' to an existing, standards-based streaming system, to potentially provide 'turbo-charging' of the existing video-streaming system.

4.2 Key technology components

Fastnets has a number of key technology components that each provide enhanced streaming, and together enable Fastnets to achieve high-quality streaming over 2.5G networks such as GPRS:

- uses H.264 for compression (also runs with H.263/MPEG-4),
- features fast-start, adaptive video — starts in 2 sec for 3G networks, and 5 sec for GPRS,
- uses CBR and VBR streaming (VBR provides up to 30% better compression),
- uses both TCP, and UDP operating a TCP-friendly transmission algorithm,
- uses bit-rate switching for adaptation — efficient and effective,
- fully featured with 'trick-modes' such as fast-forward, random access,
- includes in-band bandwidth detection for optimal streaming.

4.3 System architecture

The key attributes of the overall system are:

- decoupling of the transmission rate from the media encoding rate,
- varying the transmission rate in a network-friendly manner,
- building up a buffer of data at the client without incurring a start-up delay,

² Fastnets: FAST-start, NETwork-friendly Streaming — <http://mobile.fastnets.com>

- smoothing short-term variations in network throughput by use of client buffering,
- adjusting long-term average bandwidth requirements to match the available resources in the network by switching between streams encoded at different bit rates.

The Fastnets architecture, shown in Fig 4, consists of a Fastnets encoder for encoding live or stored multimedia content for either onward transmission via a Fastnets server, or for storing into a Fastnets file, for subsequent transmission using a Fastnets server.

4.3.1 Fastnets encoder

The Fastnets encoder encodes multimedia content into an elastic representation. Audio is encoded using AMR encoding at 4.8 kbit/s, and hence is inelastic, but as it requires a lower bit rate than video, the combined encoding can be considered to be elastic if the video is encoded in an elastic fashion.

Video is encoded using the H.264/MPEG-4 Part 10 standard into a hierarchy of independent video streams, with means to link between them.

The video hierarchy includes an 'intra stream', to allow random access, and one or more 'play streams', for ordinary viewing, each being encoded at a different bit rate, thus making the hierarchy elastic. The hierarchy also contains switching streams, which allow switching from the intra stream to play streams, and between play streams.

A hierarchy with two play streams and one intra stream is illustrated in Fig 5. The purpose of the hierarchy is to allow the server to transmit play streams to a client to achieve an optimal balance between building up a buffer of received data to provide resilience to network degradation, and providing video at the highest rate that the network instantaneously supports.

The intra stream consists of a series of intra pictures for providing random access and recovery from severe error conditions. The play streams consist of predictive and bi-predictive pictures, and periodic access pictures. The switching streams consist of a series of linking pictures.

Access and linking pictures are encoded as SP pictures, as defined in the H.264/MPEG-4 Part 10 standard. The concept underlying the encoding of SP pictures is shown in Fig 6. The source picture and the motion-compensated

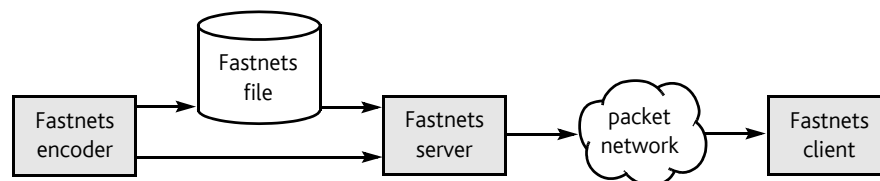


Fig 4 The Fastnets architecture.

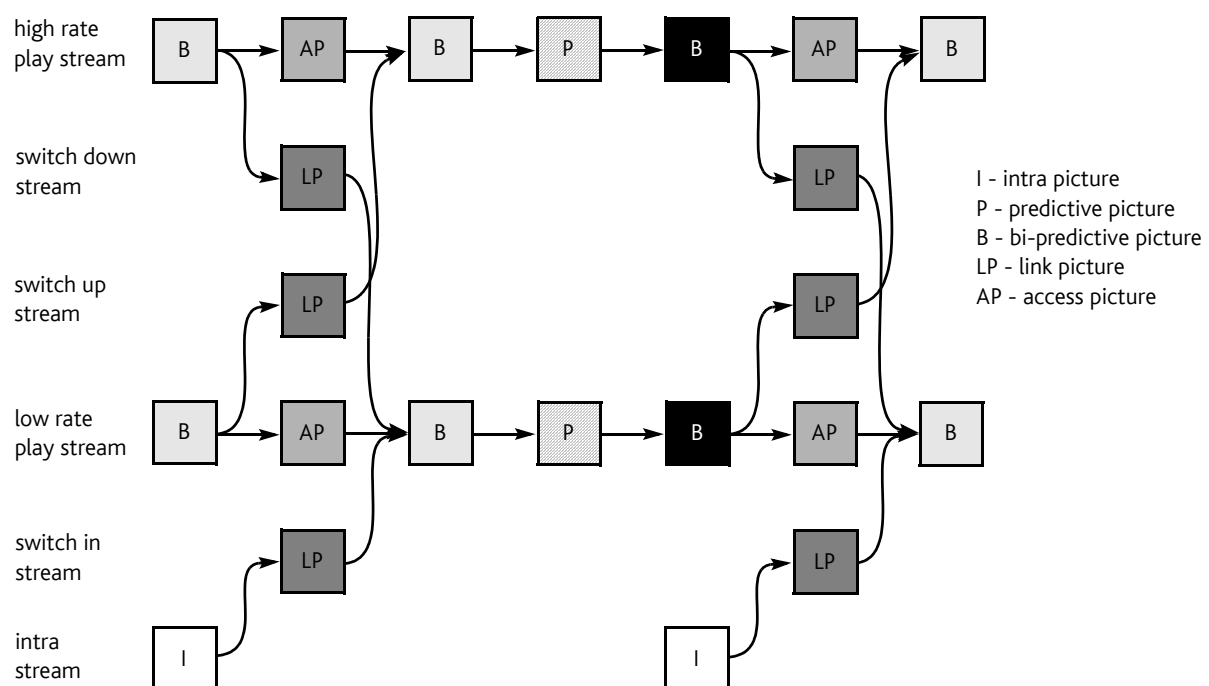


Fig 5 An example video hierarchy.

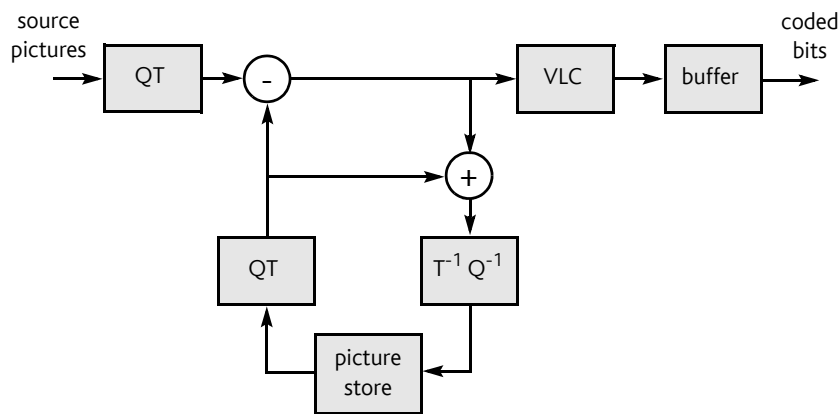


Fig 6 Concept diagram for encoding SP pictures.

prediction are independently quantised with the same quantiser and transformed, before being subtracted and entropy coded. The reconstructed picture is simply a quantised version of the source, independent of the motion-compensated prediction. Hence identical reconstructions can be obtained even when different reference pictures are used, and hence mismatch-free switching is achieved. The motion compensated prediction is relevant, as it reduces the entropy of the signal to be variable-length encoded, and hence reduces the number of bits produced by encoding a picture.

4.3.2 Fastnets server and client

The Fastnets server receives encoded multimedia content either from the Fastnets encoder or from a Fastnets file, and serves this content to one or more clients. The server scales to support many clients accessing many pieces of content independently as it performs little processing, just selecting packets for onward transmission. No encoding or transcoding of media is performed in the server.

The server works in the same way for both live streams from the encoder, and for pre-encoded streams from file. This paper therefore focuses on the details of serving live content. Figure 7 shows the main features of the server and client.

The server consists of a number of circular buffers, one for each intra, play and switching stream for each piece of live content. For each client there is one instance of packet transmitter which determines when and from which buffer the next packet is read, and sends it to the client. The

transmission ensures that all packets are received, and that the transmission rate is controlled to be 'network friendly'. Fastnets currently uses TCP, but there is also a variant that is more suited to multimedia streaming, by using a different transmission-rate-adaptation algorithm that allows a higher average throughput to be achieved.

4.3.3 Start-up

When a client connects to the server, the server locates an appropriate intra picture from the circular buffer, and sends this to the client. It then sends the linking picture to switch to the play stream with the lowest bit rate, and continues to serve from that play stream.

The transmission of packets to the client is an independent process, with the rate of transmission depending on the state of the network. The intention is that initially the transmission rate is greater than the bit rate of the play stream with the lowest bit rate. This allows the client to start decoding and presenting media to the user immediately, while also building up data in its decoding buffer.

4.3.4 Client buffering

The decoding buffer at the client reduces the impact of network jitter, packet loss and variable throughput on the quality of media presented to the user. By de-coupling the transmission rate from the media encoding rate, the client's decoding buffer can be filled when network conditions are favourable, providing resilience for less favourable times.

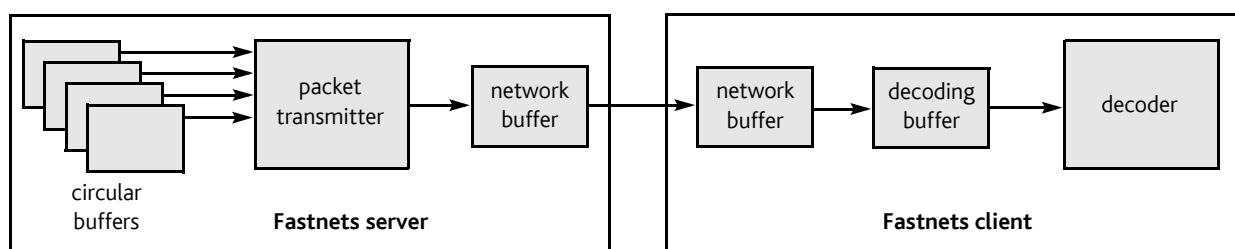


Fig 7 The Fastnets server and client.

The accumulation of tens of seconds of data in the decoding buffer allows jitter of the same magnitude to be masked from the user and provides time for the retransmission of lost packets before they are needed for decoding, thus allowing recovery from most instances of packet loss, including short periods of severe loss where the network throughput drops to zero, without affecting decoded media quality.

4.3.5 Streaming

The server attempts to send packets as quickly as possible. Initially a number of packets are sent back-to-back regardless of the network capacity, as they are simply building up in the network buffer. When the network buffer becomes full, the rate at which packets can be sent to the network buffer matches the rate of transmission over the network.

As the transmission is reliable, the server knows how much data has been received by the client — this is all of the data that has been written to the network buffer less what is still in it, which is calculated assuming it is full.

By assuming that the first transmitted data was decoded as soon as it was received, the server knows, by dead-reckoning, how much data has been decoded at any later time, and so knows how much data is in the client's decoding buffer.

When the amount of data buffered in the client's decoding buffer reaches a threshold, say 30 seconds, the server restricts the transmission rate to maintain this level of fullness.

The network throughput is estimated by counting bytes that have been sent and dividing by the time over which the measurement is made. This technique works satisfactorily provided the server is attempting to stream as quickly as possible. But when the amount of data in the decoding buffer exceeds a threshold, the server restricts its transmission rate to maintain a constant buffer fill. In this case, to make an estimate of the network throughput, the server will periodically ignore the client decoding-buffer-fullness threshold, and stream at full rate for a given period of time. So no feedback is required from client to server, beyond that needed by TCP, for the server to know how much data has been received by the client and the instantaneous network throughput.

4.3.6 Play stream switching decisions

The server initially streams the play stream with the lowest bit rate, to allow the client to decode and present media to the user immediately while also building up the level of the decoding buffer to provide resilience to network impairments. If the network has sufficient capacity to support transmission of a higher rate, the server switches

at an appropriate moment to streaming a higher rate play stream.

The server makes use of its knowledge of the amount of data buffered in the client's decoding buffer and the instantaneous network throughput to select the play stream to transmit to the client.

We have considered two criteria for determining when to switch to a higher rate stream. Firstly, the client should have sufficient data in its decoding buffer to be able to continue decoding and presenting media for a specified period of time. Secondly, the network throughput that has been achieved in the recent past should be sufficient to sustain streaming of the higher rate play stream indefinitely for live streaming, and until the end of the file when streaming from file, taking into account the data buffered at the client and the amount of data in the file still to be transmitted.

The aim is to avoid frequent switching as this can be more annoying to the user than constant quality at the lower rate.

Also to achieve this aim, the switching-down decision includes hysteresis relative to the switching-up decision. The Fastnets system considers just one criterion — switching down to the next lower bit rate play stream when the client no longer has sufficient data in its decoding buffer to be able to continue decoding and presenting media for a specified period of time. In the case of a configuration with three or more play streams, and the currently streamed play stream being the third or even higher rate play stream, this strategy does not result in an immediate drop to the bottom of the hierarchy. As access pictures only occur periodically, and the decoding buffer fullness would normally recover after an initial switch down, a second switch down would not be necessary.

4.3.7 Streaming of live content

In the case of live streaming, it is not possible to build up data in the client's decoding buffer if the first data is sent to the client with minimal delay from the time it was captured. Hence, the first data that is sent to the client is 'old' data, that is, data representing events that took place some time before the client connected. Then, as the decoding buffer fills, the most recent data in it becomes more and more recent, while the media presented to the user remains at a constant delay from the actual time of its occurrence.

The system is only acceptable for the streaming of live events where users are prepared to view the media with a constant delay of tens of seconds — users will generally accept such delays for the majority of uni-directional media streamed over IP networks.

4.3.8 Streaming from file

Streaming from file is simpler than live streaming. There is no need for circular buffers since data can be read from file as and when needed. The server, however, uses the same techniques to fill up the decoding buffer at the client and to switch between play streams. Trick modes, such as fast forward, fast reverse and random access are achieved using the intra stream.

Live streamed content can be made available for streaming at a later time by writing 'old' data in the circular buffers to file just before being overwritten. In addition to allowing users to view the content later, this functionality also allows a client to pause the presented media for an indefinite period of time, and continue streaming afterwards. It would also allow the user to fast forward after such a pause to catch up with the live stream.

4.3.9 Packetisation

We have implemented a proprietary packetisation derived from TPKT, with the header specifying the length of the packet, an RTP timestamp, and a stream identifier to allow audio and video to be multiplexed into a single TCP connection. This ensures synchronisation of audio and video transmission. If separate TCP connections were used, it is possible that they would respond slightly differently to network characteristics and achieve different throughputs, which would result eventually in vastly different amounts of data in the decoding buffers. Although these differences could be managed, the issue is totally avoided by using a single TCP connection.

4.4 Fastnets system performance

A demonstration system of the Fastnets is described in this section.

A desktop Pentium PC is used to run the Fastnets encoder and server. The client is a Nokia 7650 mobile telephone, shown in Fig 8. In a typical configuration, to a telephone that supports four time-slots in the downlink, two switching streams are used, with bit rates of 10 kbit/s and 20 kbit/s.



Fig 8 Fastnets on the Nokia 7650.

The system performs as expected. Transmission starts with the intra stream and then switches to the 10 kbit/s play stream, where it stays for some time, accumulating data in the client as a result of actually transmitting faster than 10 kbit/s. Then when sufficient data has been accumulated, and the short-term average receiving rate is more than 20 kbit/s, it switches to the higher rate play stream.

At times during a lengthy session, occasional switches back to the lower rate play stream occur as a result of reduced network throughput. And very rarely, media presentation is interrupted because of a significant period during which the network could deliver no data to the client.

The overall effect during most sessions is that the user can view continuous media presentation, with occasional changes in quality, but no distortions of the type usually associated with bit errors and packet loss. Only very rarely are complete pauses in media presentation observed as a result of severe network impairments and loss of throughput.

5. Conclusions

This paper has discussed a number of well-known and standardised techniques for audio-/video-streaming. The paper has also described the Fastnets multimedia streaming architecture for mobile IP networks. The system makes use of a hierarchy of independent video streams encoded according to ITU-T Recommendation H.264 and a bit-rate adaptive transmission technique to match the transmitted multimedia data to the instantaneous network capability. The system allows users to access multimedia content over GPRS networks, and in the near future 3G networks, using devices such as the Nokia 7650, and to experience continuous multimedia of consistent quality with minimal start-up delay.

References

- 1 Ghanbari M: 'Video coding — an introduction to standard codecs', IEE Telecommunications (1999).
- 2 MPEG — <http://mpeg.telecomitalia.com/faq/mp4-vid/>
- 3 Topiwala P, Sullivan G, Joch A and Kossentini F: 'Overview and performance evaluation of the Draft ITU-T H.26L video coding standard', Proc SPIE Appl Dig Im Proc (August 2001).
- 4 http://standards.pictel.com/ftp/q6-site/0109_San/VCEG-N18.doc
- 5 IT Week: 'Codec frees video telephony', — <http://www.itweek.co.uk/Analysis/1139947>
- 6 3rd Generation Partnership Programme — <http://www.3gpp.org>
- 7 3GPP TS 26.071: 'Mandatory speech codec speech processing functions, AMR Speech Codec; general description' — <http://www.3gpp.org/>

- 8 3GPP TS 26.171: 'AMR speech codec, wideband; general description' — <http://www.3gpp.org/>
- 9 ISO/IEC 14496-3 (1999): 'Information technology — coding of audio-9 objects — Part 3: audio' — <http://www.iso.org/>
- 10 Ghanbari M: 'Two-layer coding of video signals for vbr networks', IEEE Journal on Selected Areas in Communications, 7, No 5, pp 771—781 (June 1989).
- 11 Walker M and Nilsson M: 'A study of the efficiency of layered video coding using H.263', The Ninth International Packet Video Workshop, New York (April 1999).
- 12 Kasai H, Nilsson M, Jebb T, Whybray M and Tominaga H: 'The development of a multimedia transcoding system for mobile access to video conferencing system', IEICE Trans Communications special issue on Mobile Multimedia Communications, E00-B, No1 (January 2002).
- 13 The case for streaming multimedia with TCP — <http://www.cse.ogi.edu/~krasic/krasic-idms2001.pdf>
- 14 Transmission control protocol — <http://www.ietf.org/rfc/rfc0793.txt?number=793>
- 15 User datagram protocol — <http://www.ietf.org/rfc/rfc0768.txt?number=768>
- 16 Realtime Transfer protocol (RTP) and RTP control protocol (RTCP) — <http://www.ietf.org/rfc/rfc1889.txt?number=1889>
- 17 TCP friendly unicast rate-based flow control — http://www.psc.edu/networking/papers/tcp_friendly.html



Mike Nilsson received the BA degree in Engineering Science from Exeter College, Oxford in 1988. He is a chartered engineer currently working in the Advanced Video Services team within BT Exact at Adastral Park.

He represents BT in the ITU-T, where he is the editor of Recommendation H.245. His interests include video compression and video transcoding, and the provision of video services over heterogeneous networks.



Tim Jebb joined BT in 1984. Since then he has worked on various aspects of computer communication, including SNA networks, network management, OSI, and distributed relational and object-oriented databases.

For the last four years he has worked in the video encoding area at Adastral Park, and has contributed to the design and development of the Fastnets system, and to BT's implementation of the H.264 standard.



Matthew Walker joined BT in 1997 having graduated from Nottingham University with a BSc in Mathematics. His career with BT has involved many aspects of video compression, storage and transmission.

He currently leads the Multimedia Streaming Group within the Content and Coding Lab, at Adastral Park. His project interests include the delivery of video over mobile IP networks, and optimisation of decoder applications for mobile devices.



Rory Turnbull graduated from York University in 1994 with an MEng in Electronic Systems Engineering. He then joined BT at Adastral Park and started working in the field of speech and audio coding.

He has recently widened his interests to include video coding and currently works on real-time multimedia streaming and its application on both fixed and wireless network technologies. He is a member of TCN and the IEEE.