Video Sub-System of the RTMM Virtual Seminar Room

Ole Riis Jensen & Søren Forchhammer
Department of Telecommunication, Technical University of Denmark

August 12, 1999

Video coding for real-time interactive applications is a trade-off of bit-rates, quality, delay and complexity/cost. Furthermore the trade-off is influenced by the choice of constant bitrate or other constraints on bit-rate variations.

In this document we discuss some of the requirements for the video sub-system of the RTMM Virtual Seminar Room (VSR). Also discussed are implementation issues, such as the interfaces and protocols between the video codec and the host operating system (host OS) and available hardware options.

1 Overall Specifications

We aim for high quality in Demonstrator 2 and our goal is thus to be able to provide video with properties comparable to the PAL TV specifications. This involves, among other things,

- 720x576 (or 768x576) pixels resolution.
- 25 frames/sec. progressive video or 50 fields/sec interfaced video, i.e. 20 ms. per field.
- 8 bits per luminance or chrominance sample.
- raw video: 120 Mbit/s (420), 160 Mbit/s (422), 240 Mbit/s (444, e.g. RGB)
- coded video: 2-8 Mbit/sec.

Since we are used to hear a sound after we see the event that caused the sound, the requirements to video to audio synchronization is often asymmetric in high-end applications. Danmarks Radio (DR), e.g., specifies that audio must not arrive earlier than max. 10 ms. before the video, otherwice our perception of the audio-video event becomes un-natural. DR allows audio to arrive up to 30 ms after the video, to maintain, e.g., accurate lip synchroniszation.

These specifications are also required in other high-end systems like, e.g., the ATM based Distributed Musical Rehearsal Studio (DMRS) described in [1], an application which obviously requires very low delag, both one-way and roundtrip, in order for an orchestra at one site to be able to react promptly to directions and corrections from a conductor at the other site.

In the DMRS, being based on the FORE AVA-300/ATV-300 codec, two rehearsal facilities are located far apart, one near Bonn, Germany and the other in Geneva, Switzerland, interconnected with an ATM network. Total video encoding-decoding delay (only even fields are sent to save bandwidth) is about 46 ms, of which 9 ms. results from transmission delay. Audio encoder-decoder delay is about 35 ms. of which 20 ms. are accounted for by buffering to assure better synchronization between audio and video. Total round-trip delay is thus less than 100 ms. in this setup.

Suggested requirements to the source synchronization system are listed below:

```
per site, source-to-source synchronization:

  high end:
    video to audio (e.g. lip sync.) :                -10 - +30 ms

  low end:
    video to audio (e.g. lip sync.) :        better than +/- 80 ms

  video to text (e.g. subtitles) :                      +/- 240 ms
  video to other (e.g. slide show) :                    +/- 500 ms

local inter-site synchronization
  (i.e. between sites, as seen at a specific local site):

  audio to audio :                           better than +/- 120 ms


global inter-site synchronization
  (i.e. between sites, as seen from a genies position):

  audio to audio :                           better than +/- 120 ms
```

As for the synchronization requirements, acceptable delay and jitter should be specified. For each parameter a budget should be set up to clarify how much time is available for, e.g. encoding and decoding.

Very briefly our understanding is: Based on (e.g. worst-case) analysis a target delay endto-end is specified in a set-up procedure. It is the task of each presentation unit to present the data within a specified tolerance.

2 Video coding methods

For high end systems, like in RTMM VSR Demonstrator 2, we suggest to use an MPEG2 like encoding form, possibly modified such that the specifications can be met (e.g., low delay). Alternatively, e.g., to reduce encoder delay, a form of Motion-JPEG may be applied.

2.1 Notes on video for Demonstrator 1

For Demonstrator l, it may be necessary to reduce the requirements to the video codec.

An easy way to reduce the MPEG encoder delay is by only using I (intra coded) and P frames (predictive coded) and not using B frames (bi-directionally predictive coded). The reason for this is that bi-directional coding requires that the forward referenced frames must be coded (and decoded) before the "current" frame can be coded (and decoded), which requires buffering and reordering. Second, bi-directional motion estimation is usually more time consuming than uni-directional estimation.

Since encoding typically is the most time consuming part of the video codec, it may also be necessary to reduce the motion search area or to use pre-encoded and stored video for Demonstrator l.

2.2 Notes on video for Demonstrator 2

Interesting video topics improving quality-delay performance by variable bit-rate, object oriented video and scalable video. Below a few comments to these topics are given. (MPEG/H.26x terminology is used. This does not imply strict conformance.)

The coding of video for constant bit-rate clearly imposes a trade- off of image quality (variations) and delay. It is therefore obvious to look at variable bitrate.

An obvious choice is to employ 'quality'-control without explicit rate-control. This may lead to image dependent mean bit rates and large variations in bit-rates.

To alleviate this we could introduce less rigid control mechanisms than the buffer used in constant bit-rate, some examples are:

- the network may take the place of the buffer, maintaining fixed mean bit-rate at a larger temporal scale.
- using long GOPs, i.e. long spacing between I images. If necessary the coder can enforce an ETSI like sliding window (slice) up-date.
- coupled statistical multiplexing: The GOP structures could be synchronized to obtain optimal I-frame spacing.
- a distributed overall rate control mechanism based on mutual information using e.g. a map of the GOP structures and activity levels of the video streams (this may be combined with the GOP structure synchronization.)

Dynamic change of delay, e.g., a one-way and a two-way mode could provide a dynamic trade-off.

Object oriented video:

- Compose virtual scenes by objects from video from several sites
- Analyse the background beforehand to build a 3d model, extract and code moving objects.
- Face animation and other animations.

Scalability:

- One-directional B-frames could be introduced for frame-rate scalability.

3 Interface to the system host OS

The most important issues concerning the interfaces and protocols between the video sub-system and the host OS are

- how to set up a common system clock reference (SCR) for synchronization.
- the functionality of the buffering mechanism for incoming and outgoing coded video.
- the available display options for decoded video (like DMA transfer options). Additionally to these mechanisms a protocol for communication of the following issues must exist:
- initial encoder parameter setup (resolution, framerate, etc...)
- regular information updates on scaling and composition of decoded video sources.

In this section we will assume that the TriMedia board (or a board with similar capabilities) is used. For a brief discussion on the TriMedia board and other alternatives see Section 4.

3.1 Notes on the System Clock Reference

The system clock generator local to the video sub-system is expected to be set or adjusted according to SCR information provided by the host OS.

The host OS should also provide bounds on maximum and minimum transmission delays (including buffer delays, etc.).

Preferably the granularity of the master clock should be 90,000 ticks/s. (90 kHz) since this is the clock commonly used in video codecs (all practically used frame rates divides 90 kHz).

At system setup the following information must be initialized

- Master clock init
- Maximum and minimum delay from end to end (i.e., from digitization to presentation, or from buffer to buffer, ...).

The system must regularly, and when necessary, update the current delays (?).

The TriMedia supports setting the internal clock according to an absolute clock reference.

3.2 Notes on video input and output buffers

Encoded video will be provided to the system for transmission to other sites via a buffering mechanism provided by the host OS. Likewise, the video decoder sub-system expects to receive coded video streams from the remote sites via a similar buffer system, using one buffer per remote site (possibly including a "loopback" buffer for local decoding/display of the outgoing coded video from the local site).

Packet buffering mechanism:

- outgoing coded video, one buffer (length, granularity ?)
- incoming coded video, one buffer pr. incoming stream (length, granularity ?)

An MPEG pack contains a number of interleaved video and audio packets. The interleaving process is handled outside the video sub-system, e.g. by the network sub-system. The video sub-system inputs or outputs only video packs or packets, via a buffering mechanism provided by the host OS.

According to the MPEG standard the length af a pack must not exceed 0.7 sec. of data (for sypc. purposes, since the pack header contains important info for resynchronization and random access. This info is not repeated in packet headers.).

The standard MPEG-2 TS packet size is 188 bytes, which is (presumably) four times the cell size of ATM.

Frames/fields are time stamped relative to a 90 kHz system clock at the time of the first (or last) sampled bit of the frame/field.

A video frame/field may be split into several packs, and all packets belonging to the same frame/field has the same DTS and PTS. Each packet is assumed to have a unique sequence number.

To avoid playback discontinuity caused by buffer under- and overflow, the required amount of storage needed for buffers [2] is equivalent to the amount of data that may arrive in the time interval of

$$2 \cdot (\Delta_{max} - \Delta_{min})$$

where Δ_{max} and Δ_{min} are bounds on the transmission delay (including encoder delay, buffer delays etc).

A "fine granulated" buffer mechanism would adobt the 188 bytes TS packet as its unity, whereas a more coarse buffer would use a pack as its unity, e.g., one pack per 10 ms. (probably as an extreme example):

1 pack per 10 ms (1/2 field) –> 20-80 kbit/pack –> 13-55 packets/pack.

To make the first implementations easier, it is probably a good idea to start with fixed bitrates, at least for Demonstrator 1.

3.3 Options for displaying the decoded video

The available video output (display) options on the TriMedia board are:

- Onboard S-Video/Composite video out.

  The PCI bus is only used for coded video, up to 3 (or 4 with loopback enabled) ingoing and 1 outgoing coded video streams.

  Information about scaling, composition etc, should be communicated from the host OS. Based on this the TriMedia will use the on-board PAL output to display the composed video.

  It is possible to have the TriMedia board overlay information, locally generated or provided by the host OS, such as text, alpha channels, chroma keying, backgrounds, etc, before video is shown on the on-board video out port.

- VGA overlay to Windows NT/Linux via shared TriMedia and host PC PCI bus.

  The PCI bus will additionally to the coded video (as well as other data from the system) also have to carry the decoded video stream(s). To maintain minimum delay and minimum host OS and PCI bus activity, the decoded video should be overlayed to the host PC VGA graphics adapter via DMA transfer, if possible.

  Two approaches to displaying are possible. One is to have all (remote) sites appear in the same window on the VGA adapter. The other, more flexible method, would be to have a separate window for each site, allowing the user to rescale windows and to move windows around on the desktop.

  In the first alternative, all scaling and composition should be done on the TriMedia board such that only one decoded video stream should be overlayed to the host system.

  The second alternative implies that several video streams, possibly scaled to different resolutions, should be overlayed to the VGA adapter simultaneously and shown at different positions. Whether this is at all possible in real time, or a prioritised tradeoff between the windows is necessary (e.g., based on window sizes), depends on the possiblities and characteristics of DMA transfers under the chosen OS, among other things.

  (Otherwise, if composition of the four video sources from the four sites is supposed to be done by the host OS and then overlayed to the VGA graphics adapter, the TriMedia board will store decoded video in host PC ram via DMA transfer. The host OS must then perform tomposition and overlay to the VGA adapter. The PCI bus will in the worst case have to carry up to, e.g. $4 \cdot 160$ Mbit/s unscaled uncompressed video (though, probably only one unscaled), plus up to $4 \cdot 8$ Mbit/s. compressed video.)

4 Alternative hardware for video digitization and coding

At least four different approaches to video grabbing and coding are currently available (but are not limited to) at TELE. These options are:

1. TriMedia board, with analog S-Video/Composite video in/out and a multi-purpose multimedia enhanced DSP. Video format is YUV 422 or RGB.

   Extensive platform independent documentation available.
   Extensive driver support and software available for Windows NT/9x.
   No (or at least very early alpha) support for Linux.

2. Pinnacle / MiroVideo DC30 board, with analog S-Video/Composite in/out and a dedicated MJPEG encoder/decoder DSP. Video format is (AFAIK) RGB only (not confirmed).

   Limited driver support and software available for Windows NT/9x.
   Beta level driver and support software available for Linux (note, that the enhanced DC30+ board is not supported !).

   (TELEs piece of this hardware may be defective)

3. Low-cost Bt848 and Bt878 based TV cards With analog S-Video/Composite video in. Video format is YUV or RGB.

   Limited driver support and software available for Windows NT/9x.
   Extensive driver support and software available for Linux.

   Video encoding and decoding must be done on a separate DSP board or using the host OS/CPU (preferably running on a PIII which has the necessary multimedia (MPEG) enhancements).

4. Digital Video (DV or DVCAM format) via an IEEE 1394 "Firewire" interface card.

   Limited driver support and software available for Windows NT/9x.
   No (AFAIK) driver support or software available for Linux.

   Decoding must be performed in software by the host OS, or by a separate DSP board like the TriMedia.

Currently, our main interest is with the TriMedia board, since its multimedia enhancements makes it capable of performing both digitizing, encoding and decoding. It may be necessary to use two separate boards, one for encoding of the outgoing video stream and one for decoding multiple incoming video streams.

All the above mentioned hardware options include support for audio grabbing. However, since the audio and video sub-systems are completely separated in the RTMM VSR system, and thus handled by different hardware, we will not comment further on the audio capabilities in this document.

5 Concluding remarks

What needs to be discussed are details about

- System Clock Reference initialization.
- Estimation of acceptable end-to-end delays together with a budget for the different parts of the system.
- Buffering mechanisms.
- How to display decoded video.

References

[1] D. Konstantas, Y. Orlarey, S. Gibbs and O. Garbonel, "Design and Implementation of an ATM based Distributed Musical Rehearsal Studio", Lecture Notes in Computer Science, Multimedia Applications, Services and Techniques - ECMAST '98. Proceedings, May 1998, pp. 326-339.

[2] P. Venkat Rangan, S. S. Kumar and S. Rajan, "Continuity and Synchronisation in MPEG", IEEE Journal on Selected Areas in Comm., Vol 14, No. 1, January 1996, pp. 52-60.

Legend / Abbreviations

```
SCR   System Clock Reference
DTS   Decoding Time Stamp
PTS   Presentation Time Stamp
TS    Transport Stream (MPEG2)
GOP   Group of Pictures (MPEG)
I     Intra coded picture/frame/field (MPEG)
P     (Uni-directionally) predicted frame (MPEG)
B     Bi-directionally predicted frame (MPEG)
```