Joint Structural and Inter-frame Skipping for MPEG-2 Video

Damir Isovic

Mälardalen University, Sweden
damir.isovic@mdh.se

Abstract

In this paper, we present a combined structural and inter-frame skipping method for quality aware processing of MPEG-2 video upon overload situations. The method selects video frames both based on the structural properties, such as frame types, sizes and position in the video stream, as well as on the internal, sub-frame characteristics, i.e., the number of relevant macroblocks within a frame.

Categories and Subject Descriptors (according to ACM CCS): H.5.1 [Information Systems]: Multimedia Information Systems—Video, H.4.3 [Information Systems]: Communications Applications—Videoconferencing

1. Introduction

Today, a wide range of multimedia services has become an integral part of many industries from telecommunications to broadcasting and entertainment to consumer electronics. Distributed multimedia applications and mobile computing systems using wireless networks are becoming increasingly popular. Using such systems provide the end-users the possibility of transparently streaming multimedia content between devices of varying capabilities.

Moreover, the current trends are to move video processing from dedicated hardware to software, for reasons of cost, rapid upgradeability, and configurability. While being more flexible, software solutions are more irregular, since video processing will compete for the CPU with other applications in the system. At the same time, video is not only watched on classic TV sets, but increasingly displayed on smaller devices ranging from mobile phones to web pads, with limited system resources.

To provide smooth video playback in such heterogeneous multimedia streaming applications using software video processing, the system must be able to respond to the varying resource demands, on all the constrained devices, in such a way that the resulting quality (video or audio playback) is acceptable to all the end-users of the system. In other words, the system must be able to *adapt* the multimedia content to match the capabilities of the streaming networks and the sending/receiving devices.

Consequently, we need methods for decreasing the load introduced by media applications. There are basically two ways to do this: quality reduction, and frame skipping. With the quality reduction strategy, the decoder reduces the load by using a downgraded decoding algorithm, while frame skipping means that not all frames in a video stream are decoded and displayed, i.e., some of the frames are skipped.

In this paper, we focus on the frame skipping approach. Frame skipping can be used sparingly to compensate for sporadic high loads, or it can be used frequently if the load is structurally too high. Moreover, frames can be skipped both before sending the stream on the network, i.e., on the sending device, if the network bandwidth is restricted, and on the display device, if the processing power is limited.

However, frame skipping needs appropriate assumptions about the video stream to be effective. Skipping the wrong frame at the wrong time can result in a noticeable disturbance in the played video stream. In extreme cases, the decoding of a large and important frame might just not make it, therefore being lost and impeding quality, while simply skipping to decode a small preceding frame might have freed the resources for completion, with only slight quality reduction. In addition, skipping a frame may affect also other frames due to inter-frame dependencies. In a typical movie, a single frame skip can ruin around 0.5 seconds. Thus, frame skipping needs appropriate assumptions and constraints about streams to be effective [IFS03] .

In our previous work [IF04] we have developed a quality-aware frame skipping approach for MPEG-2 video based on realistic timing constraints for the decoding of MPEG streams. Given that not all frames can be processed, it selects those which will provide the best picture quality while matching the available resources, starting only such decoding, which is guaranteed to be completed on time. This *structural* approach to frame skipping, i.e., whole frames are skipped, is very effective with respect to making fast skipping decision at run-time, but it is not very fine grained. It does not examine the contents of the frame when selecting frames, which could play an important role. For example, the structural approach cannot provide a good comparison of two consecutive *B* frames of approximately the same size.

In this paper, we propose to use *sub-frame* selection together with the structural selection, i.e., to consider the information contained *within* a frame whenever the structural approach fails to compare two frames. As a first step towards a selection criteria on sub-frame level we have analyzed a number of MPEG-2 video streams with respect to the frame contents. We have looked into the bitstream organization (i.e., slices, macroblocks and blocks), of diverse video streams to identify the most redundant picture elements which are to be skipped first upon overload situations. Then, we proposed a new set of criteria for frame skipping based on the internal frame properties, to be used when comparing two frames on sub-frame level. Finally, we have integrated the new sub-frame skipping algorithm with the existing structural skipping method, providing a frame skipping method that both considers the whole frames and the information within single frames when making decision which frames in a video stream should be kept upon overload situations.

2. Related work

A server based algorithm for integrating multimedia and hard real-time tasks has been presented in [AB98]. It is based on average values for execution times and interarrival intervals. Work on predicting MPEG execution times, which is necessary to know for efficient frame skipping, has been presented in [BMP98, BA00]. A method for real-time scheduling and admission control of MPEG-2 streams that fits the need for adaptive CPU scheduling has been presented in [DA00]. The method is not computationally overloaded, qualifies for continuous re-processing and guarantees Quality-of-Service (QoS). However, no consideration on making priorities on the frame level has been done. A frame skipping pattern that makes distinction between frames has been presented in [NHW00]. However, only one skipping criterion, QoS human [NLW*02], has been applied when selecting frames, taking no consideration about frame sizes, buffer and latency requirements, or compression methods used.

Most standard video processing methods will fail to satisfy the demands of MPEG-2 upon overload situations as they do not consider the specifics of this compression standard. In our work, we consider both the structural properties of a video stream as well as the sub-frame properties of single video frames when making skipping decisions.

3. Structural frame skipping of MPEG-2 video

3.1. MPEG-2 Video Stream

The MPEG-2 standard defines three types of frames, *I*, *P* and *B*, see figure 1-a. The *I* frames or *intra* frames are simply frames coded as still images. They contain absolute picture data and are self-contained, meaning that they require no additional information for decoding. *I* frames have only spatial redundancy providing the least compression among all frame types. Therefore they are not transmitted more frequently than necessary.

The second kind of frames are *P* or *predicted* frames. They are forward predicted from the most recently reconstructed *I* or *P* frame, i.e., they contain a set of instructions to convert the previous picture into the current one. *P* frames are not self-contained, i.e., if the previous reference frame is lost, decoding is impossible.

The third type is *B* or *bi-directionally* predicted frames. They use both forward and backward prediction, i.e., a *B* frame can be decoded from a previous *I* or *P* frame, and from a *later* *I* or *P* frame. They contain vectors describing where in an earlier or later pictures data should be taken from. They also contain transformation coefficients that provide the correction. *B* frames are never predicted from each other, only from *I* or *P* frames. As a consequence, no other frames depend on *B* frames. *B* frames require resource-intensive compression techniques but they also exhibit the highest compression ratio, on average typically requiring one quarter of the data of an *I* picture.

Predictive coding, i.e., the current frame is predicted from the previous one, cannot be used indefinitely, as it is prone to error propagation. A further problem is that it becomes impossible to decode the transmission if reception begins part-way through. In real video signals, cuts or edits can be present across which there is little redundancy. In the absence of redundancy over a cut, there is nothing to be done but to send from time to time a new reference picture information in absolute form, i.e., an *I* frame. As *I* decoding needs no previous frame, decoding can begin at *I* coded information, for example, allowing the viewer to switch channels. An *I* frame, together with all of the frames before the next *I* frame, form a *Group of Pictures (GOP)*, see 1-b. The GOP length is flexible, but 12 or 15 frames is a common value. Furthermore, it is common industrial practice to have a fixed pattern (e.g., *IBBPBBPBBPBB*). However, more advanced encoders will attempt to optimize the placement of the three frame types according to local sequence characteristics in the context of more global characteristics.

As mentioned above, *B* frames are predicted from two *I*

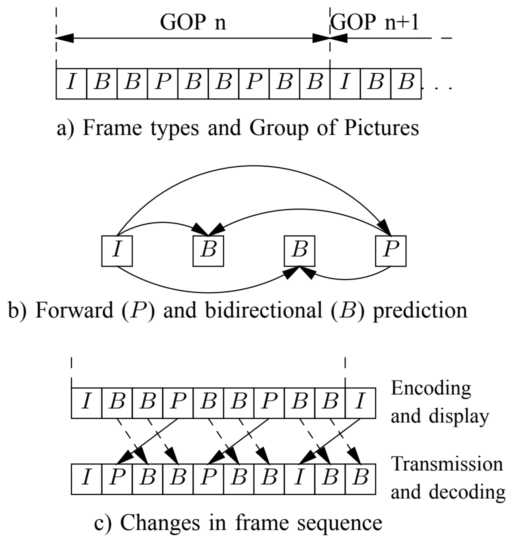

Figure 1: *MPEG-2 video stream*

or P frames, one in the past and one in the future. Clearly, information in the future has yet to be transmitted and so is not normally available to the decoder. MPEG gets around the problem by sending frames in the "wrong" order. The frames are sent out of sequence and temporarily stored. Figure 1-c shows that although the original frame sequence is *I BB P* ..., this is transmitted as *I P BB* ..., so that the future frame is already in the decoder before bi-directional decoding begins. Picture reordering requires additional memory at the encoder and decoder and delay in both of them to put the order right again. The number of bi-directionally coded frames between I and P frames must be restricted to reduce cost and minimize delay, if delay is an issue.

3.2. Structural skipping

In our previous work [IF04] we have identified a number of criteria that are to be applied when making skipping decisions. According to the frame type criterion, the I frame is the most important one in a GOP. If we lose the I frame in a GOP, then the decoding of all consecutive frames in the GOP will not be possible, since all other frames in the GOP depend directly or indirectly on the I frame. B frames are the least important ones because they are not reference frames.

Frame position criterion is applied on P frames. Skipping a P frame will cause the loss of all its subsequent frames, and the two preceding B frames within the GOP. For instance, skipping the first P frame (P_1) would make it impossible to reconstruct the next P frame (P_2), as well as all B frames that depends on both P_1 and P_2. And if we skip P_2 then we cannot decode P_3 and so on. Hence, the closer to the start of the GOP the more important P frame.

Frame size criterion applies mainly to B frames. According to our analysis [IF02], there is a relation between frame size and decoding time, and thus between size and gain in display latency. The purpose of skipping is to increase display latency. So, the bigger the size of the frame we skip, the larger display latency obtained. However, skipping large B frames might not always be the best option. Small B frames might exploit complex compression techniques which minimize frame size, but are more expensive to decode, in terms of needed processing power. Frame prediction from reference frames is found to be most computationally expensive [MP93]. Hence, if the network bandwidth is limited, then large B frames should be skipped first, and if the objective is to decrease the CPU load, small, more compressed frames should be skipped.

Skipping distribution criterion says that with the same number of skipped B frames, a GOP with *evenly* skipped B frames will be smoother than a GOP with uneven skipped B frames, e.g if we have a GOP=*IBBPBBPBBPBB* then even skipping $I-BP-BP-BP-B$ will give smoother video than uneven skipping $I--PBBPBB--$, since the picture information loss will be more spread [NLW*02].

Please refer to our previous work [IF04] for details on all identified structural criteria and the structural frame skipping method.

4. Sub-frame skipping

A picture frame consist of a number of *macroblocks*, which are 16x16 arrays of luminance pixels, or picture data elements. Macroblocks in a frame can be coded as intra and non-intra, where the first type does not need a reference to be decoded, while the second one is either forward-predicted, backward-predicted or forward-and-backward predicted from other macroblocks. The macroblocks within an I frame are coded as intra. Most of the macroblocks in P and B frames are non-intra. However, some of them are coded as intra, which is a way to refresh the video information without introducing an extra I frames. This guarantees that after a certain number of frames all macroblocks in a frame have been intra updated. This will stop the error propagation when a part of a video frame is lost during transmission.

4.1. Intra macroblocks

Intra-macroblocks are the most interesting to keep when making skipping decisions because they are used as reference in other frames. The more intra-macroblocks in a frame, the more dependencies on the frame.

We have analyzed diverse MPEG-2 streams for the amount of intra-coded macroblocks per P and B frames, see

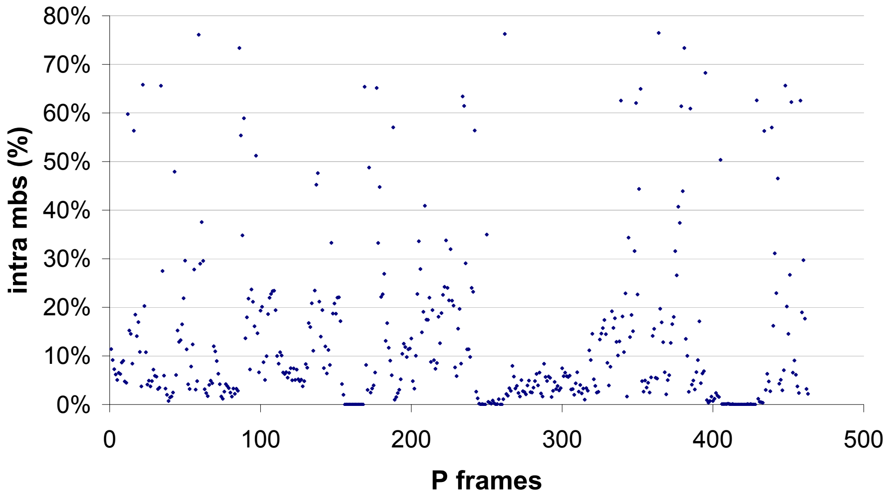

Figure 2: *Intra macroblocks (mbs) per P frames in an example MPEG-2 video stream*

figures 2 and 3for an example. We can see from the figure that the number of intra macroblocks vary a lot between different frames. Hence, we can propose a new criterion for selecting frames on inter-frame level: the more intra macroblocks per a frame, the higher priority will be assigned. Once again, observe that this is valid on the decoder side, i.e., the stream is either stored locally or, in the case of video streaming, the GOP has been transmitted to the decoder. On the other hand, if we perform stream adjustment before sending the stream on the network, we might need to actually do the opposite, i.e., keep the frames with less intra macroblocks, since they have lower bit sizes.

4.2. Skipped macroblocks

Furthermore, we have analyzed the percentage of skipped macroblocks, i.e., macroblocks for which no data is encoded. Skipped macroblocks are used to achieve higher compression ratio. When a macroblock is skipped, it is implicitly defined by the standard in the following way: in a *P* frame, a skipped macroblock is a direct copy of the corresponding macroblock from the previous *I* or *P* frame. In a *B* frame, a skipped macroblock is reconstructed by assuming the motion vectors and motion type (i.e., forward, backward, or bidirectional) are the same as the last encoded macroblock. In this case, skipped macroblocks can not follow intra-coded macroblocks because then there would be not motion type or motion vectors defined.

The average numbers of skipped macroblocks per frame type for some example streams are presented in table 1. We do not present skipped macroblocks for *I* frames simply because we could not identify any. According to the MPEG standard [MPE96], even *I* frames can have skipped macroblocks (that use only spatial redundancy), but we could not find any skipped *I* macroblocks in any of the analyzed streams. We conclude that macroblocks are seldom skipped in *I* frames. Skipped macroblocks per *B* frames for an example video stream are shown in figure 4.

MPEG-2 Video Streams	*P* frames	*B* frames
Drama movie, 720x576	19%	15%
Action movie, 354x240	7%	10%
Philharmonic, 720x576	25%	22%
Cartoon, 354x240	12%	27%

Table 1: *Average skipped macroblocks per P and B frames*

The more skipped macroblocks per frame, the more similar it becomes to some other frame. Hence, frames with a lot of skipped macroblocks should be given lower priority, since some of the adjacent frames contain the same video information. Besides, no other macroblocks are reconstructed from skipped macroblocks.

4.3. Zero-motion macroblocks

By performing the sub-frame analysis we could make an interesting observation: the total number of macroblocks for some *P* frame is not equal to the sum of all intra macroblocks and forward-predicted macroblocks for the frame. *P* frames do not exploit backward prediction, i.e., they do not contain any backward-predicted macroblocks, hence the total sum of all macroblocks per *P* frame should be the sum of all intra and forward-predicted macroblocks. The explanation is that in *P* frames (and only *P* frames), there are some macroblocks which are not intra (i.e., motion compensation is in use) but also do not define any forward motion vectors. By definition, these macroblocks are interpreted as using mo-

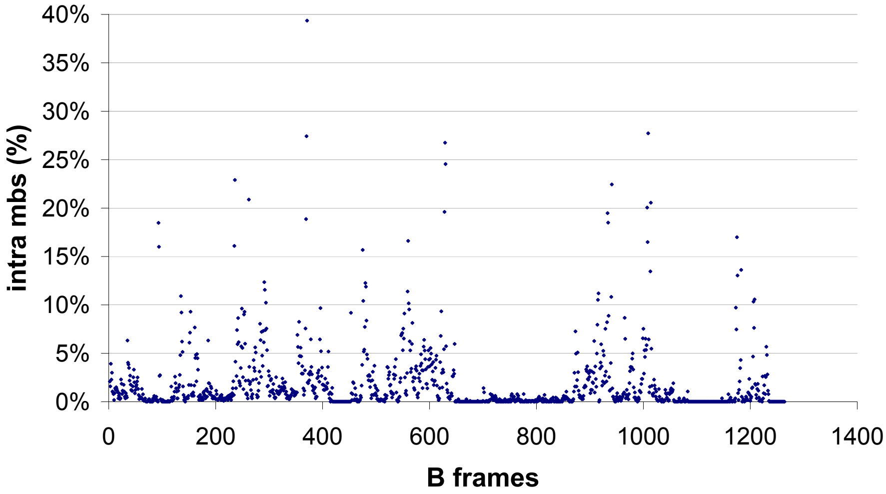

Figure 3: *Intra macroblocks (mbs) per B frames in an example MPEG-2 video stream*

tion compensation with a motion vector defined as (0,0). It is a special case in the standard because it happens so often. That what makes it interesting is those macroblocks are very good candidates for skipping on sub-frame level, since no other macroblocks depend on them. So, an additional inter-frame criterion for *P* frames is: the more (0,0) macroblocks in a *P* frame, the lower priority should be assigned.

5. Combined skipping algorithm

The new skipping algorithm that we propose uses the following set of structural and sub-frame skipping criteria:

1. *Frame type* – Assign highest priority to *I* frames, and lowest priority to *B* frames.
2. *GOP position* – Assign higher priority to *P* frames that are closer to the start of the GOP.
3. *Skipping distribution* – Assign priority to *B* frames such that the skipped frames are distributed more evenly in the GOP.
4. *Frame size* – Assign higher priority to larger *B* frames.
5. *Intra macroblocks* – Assign higher priority to *P* and *B* frames with larger number of intra-coded macroblocks in the frame.
6. *Skipped macroblocks* – Assign lower priority to *P* and *B* frames with larger number of skipped macroblocks per frame.
7. *Zero motion macroblocks* – Assign lower priority to *P* frames with more zero motion macroblocks.

When deciding the relative importance of frames for the entire GOP, we assign priorities to frames according to all criteria collectively applied, rather than applying a single criterion. Since the criterion 1 is the strongest one, the *I* frame will always get the highest priority, i.e., all frames in the GOP depend on it. For *P* frames, in our previous work, we always kept the *P* frames closer to the start of the GOP. Here, we make better, more fine-grained decisions on which frames to keep, based on the internal structure of the frames, i.e., the number of intra, skipped and zero motion macroblocks. Similarly, we used to apply only the frame size criterion for *B* frames, but in the new algorithm we look deeper into the structure of the frame to make better skipping decisions.

Here is the the pseudo-code for the frame selection algorithm that takes a set of frames, e.g., a GOP, as an input and assigns importance values to the frames based on the identified criteria:

Let:

N	= GOP length
M	= distance between reference frames
$\mathcal{P}$	= a set containing all P-frames in the GOP
$\mathcal{B}$	= a set containing all B-frames in the GOP
$v(f)$	= importance value of frame f
ESC_i	= i^{th} even-skip chain of *B* frames

Step 1: Assign the highest value to the I-frame (equal to the number of frames in the GOP).

$$v(I) = N$$

Step 2: The set $\mathcal{P}$ contains all P-frames, sorted according to their position in GOP. The longer the distance from the I-frame, the lower the importance value.

$$\forall P_i \in \mathcal{P}, 1 \leq i \leq |\mathcal{P}|$$
$$v(P_i) = N - i$$

Step 3: Reassign priorities of P-frames according to the intra-macroblock criterion. Apply skipped macroblock and zero motion criteria to break ties.

$$\forall P_i \in \mathcal{P}, 1 \leq i \leq |\mathcal{P}|$$

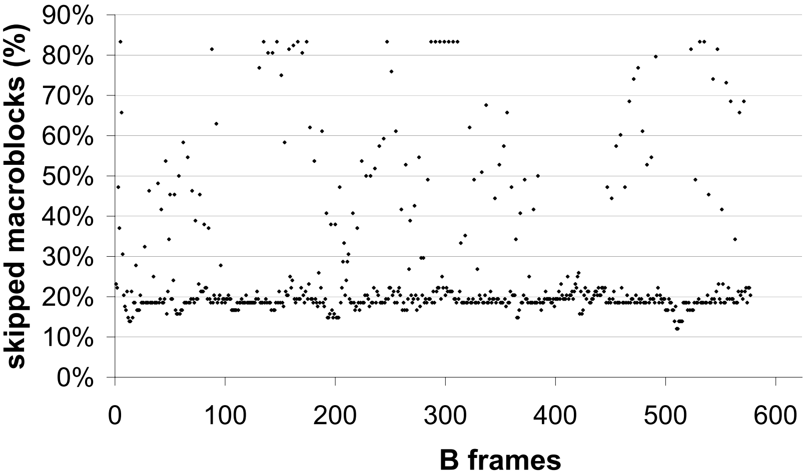

Figure 4: *Skipped macroblocks in B frames*

$$
\begin{array}{l}
if\ (intra(P_i) > intra(P_{i-1})) \\
\quad swap(P_i, P_{i-1}) \\
elseif(intra(P_i) == intra(P_{i-1})) \\
\quad if\ (skipped(P_i) < skipped(P_{i-1}) \\
\quad\quad swap(P_i, P_{i-1}) \\
\quad elseif(skipped(P_i) == skipped(P_{i-1})) \\
\quad\quad if\ (zero(P_i) < zero(P_{i-1})) \\
\quad\quad\quad swap(P_i, P_{i-1})
\end{array}
$$

Step 4: Initially set all values for B-frames to the lowest P-value.

$$
\begin{array}{l}
\forall B_k \in \mathcal{B}, 1 \leq k \leq |\mathcal{B}| \\
\quad v(B_k) = min[v(P_i) \mid 1 \leq i \leq |\mathcal{P}|] - 1
\end{array}
$$

Step 5: Identify all “even-skip” chains for B-frames and sort them according to the total byte size. Decrease the importance values of the B-frames, depending on which chain they belong to.

$$
\begin{array}{l}
ESC_1 = \{B_1\} \cup \{B_{1+j*M} \mid 1 \leq j \leq \frac{N}{M-1}\} \\
\forall i, 2 \leq i \leq |\mathcal{B}^*| \\
\quad ESC_i = \{B_i\} \cup \{B_{i+j*M} \mid 1 \leq j \leq \frac{N}{M-1}\} \\
\quad if\ \ sum(ESC_i) > sum(ESC_{i-1})) \\
\quad\quad swap(ESC_i, ESC_{i-1}) \\
\quad \forall B_k \in ESC_i \\
\quad\quad v(B_k) = v(B_k) - |ESC_{i-1}|
\end{array}
$$

Step 6: Within each chain, apply sub-frame skipping criteria to assign unique priorities to the B-frames in the chain.

$$
\begin{array}{l}
\forall B_k \in ESC_i \\
\quad if\ (intra(B_k) > intra(B_{k-1})) \\
\quad\quad swap(B_k, B_{k-1}) \\
\quad elseif(intra(B_k) == intra(B_{k-1})) \\
\quad\quad if\ (skipped(B_k) < skipped(B_{k-1}) \\
\quad\quad\quad swap(B_k, B_{k-1})
\end{array}
$$

The presented algorithm skips small B-frames first. If the objective is to utilize limited network bandwidth, then the "even-skip" chains above should be sorted in acceding order, i.e., large B-frames should be skipped first. Moreover, in this case, we would also keep the frames with less intra macroblocks, since they have lower bit sizes. The algorithm for opimizing the stream before sending it over network is very similar to the one presented above, and hence, omitted in this paper.

6. Conclusions

In our previous work, we proposed a structural approach for quality-aware frame skipping of MPEG-2 video, which selects frames according to a set of criteria on the frame level, such as frame types, positions and sizes. In this paper, we extended it by considering the internal structure of the frames when making skipping decisions.

First, we analyzed a number of MPEG-2 video streams with respect to the frame contents, to identify the most redundant picture elements within a single frame. Then, we proposed a set of new, sub-frame skipping criteria, such as intra, skipped and zero macroblock, used for for intra-frame skipping. Finally, we integrated the sub-frame skipping with the structural skipping, and proposed a joint structural and sub-frame skipping approach for MPEG-2 video.

We have previously evaluated our frame skipping method based on structural skipping, by using both subjective and objective quality measurements. Currently, we are extending it to the joint structural and inter-frame skipping approach presented in this paper. Furthermore, looking into how we can apply similar methods on MPEG-4 video.

References

[AB98] ABENI L., BUTTAZZO G. C.: Integrating multimedia applications in hard real-time systems. In *Proceedings of the 19th IEEE Real-Time Systems Symposium* (Madrid, Spain, 1998). 2

[BA00] BURCHARD L. O., ALTENBERND P.: Estimating decoding times of mpeg-2 video streams. In *Proceedings of International Conference on Image Processing (ICIP 00)* (Vancouver, Canada, September 2000). 2

[BMP98] BAVIER A., MONTZ A., PETERSON L.: Predicting mpeg execution times. In *Proceedings of ACM International Conference on Surement and Modeling of Computer Systems (SIGMETRICS 98)* (Madison, Wisconsin, USA, June 1998). 2

[DA00] DITZE M., ALTENBERND P.: Method for real-time scheduling and admission control of mpeg-2 streams. In *The 7th Australasian Conference on Parallel and Real-Time Systems (PART2000)* (Sydney, Australia, November 2000). 2

[IF02] ISOVIC D., FOHLER G.: Analysis of mpeg-2 streams. In *Technical Report at Malardalen Real-Time Research Centre,Vasteras, Sweden* (March 2002). 3

[IF04] ISOVIC D., FOHLER G.: Quality aware MPEG-2 stream adaptation in resource constrained systems. In *ECRTS* (Catania, Italy, July 2004). 1, 3

[IFS03] ISOVIC D., FOHLER G., STEFFENS L. F.: Timing constraints of mpeg-2 decoding for high quality video: misconceptions and realistic assumptions. In *Proceedings of the 15th Euromicro Conference on Real-Time Systems* (Porto, Portugal, June 2003). 1

[MP93] MAYER-PATEL K.: Performance of a software MPEG video decoder. In *ACM Multimedia Conference* (1993). 3

[MPE96] Iso/iec 13818-2: Information technology - generic coding of moving pictures and associated audio information, part2: Video. 4

[NHW00] NG J. K.-Y., HUI C. K.-C., WONG W.: A multi-server design for a distributed MPEG video system with streaming support and QoS control. In *Proceedings of the 7th International Conference on Real-Time Systems and Applications* (Cheju Island, South Korea, December 2000). 2

[NLW*02] NG J. K., LEUNG K. R., WONG W., LEE V. C., HUI C. K.: Quality of service for mpeg video in human perspective. In *Proceedings of the 8th Conference on Real-Time Computing Systems and Applications (RTCSA 2002)* (Tokyo, Japan, March 2002). 2, 3