RC 22095 (99054) Engineering and Technology

IBM Research Report

A Low Cost and Efficient Logo Insertion Scheme in MPEG Video Transcoding

Ligang Lu, Shu Xiao, Jack Kouloheris, Cesar A Gonzales
IBM Research Division
Thomas J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598

A Low Cost and Efficient Logo Insertion Scheme In MPEG Video Transcoding

Shu Xiao*, Ligang Lu, Jack L. Kouloheris, and Cesar A. Gonzales

Multimedia Technologies Department
IBM T. J. Watson Research Center, P.O.Box 218, Yorktown Heights, NY, 10598.
shu.xiao@skystream.com, lul@us.ibm.com

ABSTRACT

In this paper, we work on the problem of inserting logo into part of a previously compressed video sequence while transcoding it to a video stream with lower rate. In most of the transcoding architecture, it has been widely accepted to reuse the motion information extracted from the original incoming bit stream. For our transcoding and logo insertion process, we preserve the same motion information as the incoming video stream for the parts that were unaffected by logo, and propose an efficient algorithm to code logo areas. We first find the proper range of pictures that might be affected by logo insertions, analyze logo-affected parts of those pictures, and perform corresponding algorithms depending on the picture types. In addition, we develop a method to achieve better bit allocation during logo insertion. The efficiency of our algorithm is demonstrated by simulation.

Keywords: Transcoding, logo insertion, motion vector, quantization scale

1. INTRODUCTION

Video transcoding is widely used in many video communication applications. It is a process of converting a compressed stream to another one with different compression format[1,2] and/or lower bit rate.[3–6] In our study, we constrain the transcoding operation to be reducing the bit rate of a previously compressed MPEG-2 video stream. The proposed algorithms can be extended to be suitable for other common video standards, such as MPEG-1 and H.263. We proposed new rate control algorithms for MPEG-2 transcoding process in another paper.[7] One very useful and highly demanded feature in video transcoding is to insert logo into part of the bit stream. In this work, we focus on the logo insertion problem during MPEG video transcoding, and present an efficient logo insertion scheme.

We consider both cases of inserting the transparent and non-transparent logos. The non-transparent logo totally replaces the part of the image data covered by logo. The transparent logo only overlays on the original image data which still can be seen through the transparent logo. In efficient and low cost video transcoding,[1–6] it is recommended to reuse the basic video stream information obtained from the compressed input bit stream, such as picture coding types, motion vectors, and macroblock information. It reduces the computational complexity tremendously without losing much output video quality. In order for our logo insertion scheme to be applicable to as many different transcoders as possible, our scheme also reserve and reuse the information decoded from the input stream as much as possible.

In Section 2, we will first analyze what will be affected by inserting transparent and non-transparent logos respectively. Then we will develop procedures to systematically determine the affected pictures and affected area in those pictures due to logo insertion. Next we will derive solutions on how to change the macroblock prediction mode and how to modify the motion vectors of those affected areas in the affected pictures (including I, P, and B pictures) for various macroblock modes. The affected areas includes both the part covered by logo and the parts that use the image data in the logo area for motion compensated prediction. The two different affected areas are treated with different algorithms.

In Section 3, we will study how to quantize the logo area. To avoid the flicking and pulsation artifacts in the logo area and to guarantee the same visual quality of the logo, we will present an algorithm to quantize the logo

* Shu Xiao is now with SkyStream Networks, Sunnyvale, CA.

areas for both transparent and non-transparent logos. This algorithm uses different strategies to quantize the transparent and non-transparent logos in different pictures (I, P, and B pictures). For each case, the algorithm will determine whether the macroblock should be coded or skipped and how it should be coded (Intra vs. Inter) and what quantization scale should be used so that the logo area is free from visual artifacts. The algorithm also makes the rate control to prevent from the logo part to use too many bits in the low bit rate coding case and catastrophic mode occurring.

Our logo insertion scheme does not require to reestimate motion vectors for the logo affected areas. Instead, it only makes simple but effective modifications on motion vectors and macroblock modes for the affected macroblocks, and quantization of the logo affected areas. In Section 4, some simulation results using several test sequences are shown to prove the efficiency of the proposed logo-insertion algorithms. Section 5 contains our conclusion.

2. MODIFICATION OF MOTION INFORMATION

As stated in Section 1, we considered two types of logos in this paper: covered and transparent logos. We furthermore made assumptions that the logo part occupies integer number of macroblocks, especially for covered logo; and the logo positions are fixed in all frames with logo inserted. There is very little difference in modifying the motion information for these two types of logos. We will describe them in one piece.

As we discussed in the last section, some of the information decoded from the compressed bit stream is preserved in most of the presented transcoders, such as the picture type and the motion information, which is the most expensive part in the whole encoding process. Since there is only a small part of the video picture sequence is affected by logo and it affects a small portion of the picture, we would like to reuse the old motion vectors during logo insertion for the part unaffected by logo. That is, we need to take care of the motion vectors of the part that is affected by logo, which includes the part covered by logo and the part outside of logo but was motion compensated from areas inside logo (only for P, B frames). We name them "logo part" and "logo-affected part" respectively in the following discussion. We will address on how to deal with these two situations separately.

To work on the part of the picture affected by logo, first we need to find out the range of video images that is possibly affected by logo. Without loss of generality, we considered open GOP structure in this paper, with a typical such sequence: BBIBBPBBP... . All the ideas can be modified for closed GOP structure also. Assume that it is required to insert logo in the range $[l, h]$ of the video sequence which starts from the sequence number zero, here, l and h are the lowest and highest frame numbers of the video sequence in display order. Thus, the possible range $[L, H]$ of the logo-affected frames is decided by the following equations:

$$L = \text{Previous reference frame} + 1 \tag{1}$$

$$H = \begin{cases} \text{Next reference frame} & \text{if next reference frame is not I frame} \\ \text{Next reference frame} - 1 & \text{otherwise} \end{cases} \tag{2}$$

Here, L is the smallest frame number of the B frame that could be affected by the smallest reference logo frame, while H is the biggest frame number that could be affected by the largest reference logo frame. It is the next reference frame if the next reference frame is not an I frame, otherwise, the possible logo affected frame count stops right before that I frame. All the frame numbers discussed are in display order. Following is an example of the relationship between logo ranges and the possible affected frame numbers. In the example, if the defined logo range is $[l, h] = [4, 11]$(or$[4, 12]$, $[4, 13]$), since the next reference frame is an I frame, the possible affected frame range is $[L, H] = [3, 13]$. On the other hand, if the defined logo range is $[5, 14]$, the next reference frame is a P frame with frame number 17, therefore, the possible affected range is $[3, 17]$. In the logo insertion process, all the frames in the range $[L, H]$ should be checked and be modified if they are affect by logo.

Frame number	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Frame coding type	B	B	I	B	B	P	B	B	P	B	B	P	B	B	I	B	B	P
Logo range				L	l							h	h	$h(H)$				
Logo range				L		l									h			H

2.1. Motion vector modification in the logo part

Once we have the possible affected frame range, we work on how to reset the motion vectors and modes for different situations. In this section, we discuss the modifications in the logo part, i.e., the macroblocks covered by the logo. Since the logo insertion range is $[l, h]$, the logo part only appears in the frame with frame number $\in [l, h]$, we considered several cases with respect to its picture coding type and frame number to correct the reference chain.

First of all, if the first appeared reference frame $\in [l, h]$ is a P frame, then set the macroblock mode (mb_mode) of the macroblocks in logo area to be intra-coded (INTRA), otherwise, all the I and P frames keep the same modes. For the B frames, if it is one of the two beginning frames, i.e., frame number L or $L + 1$ (in that case, $l = L$ or $l = L + 1$), set the mb_mode to be backward predicted (BPRED). If the frame H is a B frame, which indicates that the following reference frame is an I frame, we need to take care of the B frames H or/and $H - 1$. We can either set the mb_mode of the logo part in those two B frames to be forward predicted (FPRED) or intra-coded. We used the INTRA mode for those B frames in our simulations. For the other B frames, simply set the mb_mode of the logo part to be FPRED. At last, we set the motion vectors of the non-intra logo macroblocks to be zero. This scheme guarantees an accurate reference relationship in the logo part. Equivalently, we wrote a pseudo-code as follows:

```
if (l<=pic->frame_number<=h)
{
  if (pic->picture_type == P && P is the first reference frame in [l h])
      logo_blks->mb_mode = INTRA;
  if (pic->picture_type == B)
      if (pic->frame_number == L || pic->frame_number == L+1)
            logo_blks->mb_mode = BPRED;
      else if (frame H is a B frame && (pic->frame_number == H ||
                                        pic->frame_number == H-1))
            logo_blks->mb_mode = INTRA;
      else for the non-intra logo macroblocks,
            logo_blks->mb_mode = FPRED;
  if (logo_blks->mb_mode != INTRA)
      zero_motionvectors(logo_blks);
}
```

2.2. Motion vector modification in the logo-affected part

Next, we consider how to fix the motion information in the logo-affected part, i.e., the part outside of the logo but is motion compensated from the logo area. The logo-affected part possibly includes all the macroblocks of the P and B frames in the range $[L, H]$, except those with logo inserted. Since macroblocks in I frames are intra coded, they do not belong to the logo-affected part.

For the P and B frames, we first check the reference area of each macroblock which is decided by the motion vector of that macroblock. If the reference frame belongs to $[l, h]$ and the reference area is partly/totally inside the logo area, we need to reestimate the motion vector. In our algorithm, we reestimate the motion vector by comparing the estimation errors using two coding schemes; one is to intra-code the macroblock, and the other is to motion compensate it with zero motion vector. The new motion information of the macroblock is set to be complied with the one that provides a smaller error. Here, when the zero motion compensation is performed, we adopt the same prediction direction as the input stream, for example, if the macroblock is forward/backward predicted originally, we still calculate the error of zero motion compensation of forward/backward prediction.

Since the logo area is relatively small compared to the whole picture size and the logo is usually located at the corner of a frame, normally the logo part is not frequently used for motion compensation, and the number of compensated macroblocks might not be considerable. Therefore, it is not necessary to refine the motion vectors by a complicated full search algorithm.

In practice, there are about 1%-2% macroblocks affected by the logo part. We tested on some video sequences using the perfectly reestimated motion vectors in the logo-insertion process, but the performance (in PSNR) does not increase significantly.

In pseudo-programming language, we conclude it as:

```
 if (pic->frame_number is in [L H])
 {
   for (macroblk_nmb = 0; macroblk_nmb < MAX_blk_num; macroblk_nmb++)
   {
     if (macroblk is not one of the logo_blks )
     {
       if (macroblk->mb_mode = FPRED || macroblk->mb_mode = BPRED)
       {
         if (reference area of macroblk is inside logo area)
             reestimate_mvs(macroblk);
         else if (macroblk->mb_mode = IPRED)
         {
            if (backward refer_area inside logo area
                        && forward refer_area NOT inside logo area)
              use forward motion information;
            else if (forward refer_area inside logo area
                        && backward refer_area NOT inside logo area)
              use backward motion information;
            else if (all forward/backward refer_area inside logo area)
              reestimate_mvs(macroblk);
         }
       }
     }
   }
}
```

3. QUANTIZATION SCALES

Since generally the logo is inserted at the right bottom corner of a picture which is the latest coded parts in a frame, there is no guarantee of a stable and reasonably good visual quality of the frames with logo inserted. In many of the cases, the inserted logos (especially covered logos) are the most perceptually sensitive part in a video sequence due to its motionlessness. To make the logo part look still and to assure the same quality of the logo, we need to adjust the quantization scales of the macroblocks and perform a better rate control in the macroblock level. In this section, we propose a scheme to deal with this problem. We state the modifications of the quantization scales in the logo area for covered and transparent logos separately in Sections 3.1 and 3.2.

3.1. Quant-scale modification for covered logo

For covered logos, it should look still in the motion picture sequence. Therefore, we set the quantization scale of all the intra blocks in logo area to be a same constant, and we skip all the logo macroblocks of P frames. For the B frames, since there is no skip mode in the original codec, we set the quantization to be maximum if the macroblock is not coded as INTRA, so that the motion compensation in the logo area of B frames is minimized. In MPEG-2 standard, the biggest quantiser scale code for both linear and nonlinear quantizations is 31. We write:

```
mquant_index = 32/output_bit_rate(in MB) for intra logo_blks
mquant_index = 31 for nonintra B frame logo_blks
skip logo_blks for nonintra P frames
```

3.2. Quant-scale modification for transparent logo

The quant-scale adjustment for transparent logo is similar to that of the covered logo. In the case of covered logo, no macroblock difference in the logo area needs to be coded; while the transparent logo is placed on the original image data and the logo parts for different frames are not identical, therefore, we need to code the motion compensated macroblocks.

In order to make the logos in different frames look approximately the same, we set quantization index of intra-coded blocks to be the same constant as in Section 3.1. Besides, since the compensated error of the the transparent logo area is not negligible, we set the quantization index of the inter coded macroblocks to be a value slightly bigger than that of the intra quant-scales.

```
mquant_index = 32/output_bit_rate(in MB) for intra logo_blks
mquant_index = 48/output_bit_rate(in MB) for inter logo_blks
```

The reason we make the quantization index inverse-proportional to the output bit rate is that we do not want to use too many bits on logo parts when the bit rate is low. Otherwise, a lot of macroblocks might be forced to catastrophic mode.

Since we make the logo part look constantly clear by fixing the quatization scales, the average bits consumed in the area might be more than that of the other parts in the picture. Since the quantization scale of the logo part is fixed and logo area normally belongs to the latest coded portion in a picture, it should be guaranteed that enough bits are left when we code the logo macroblocks, which is not realistic for the encoder. To fix this problem, one simple way is that we can first pre-encode the logo part (or count the bits needed for logo part), deduct that amount of bits from the total target bits, and then perform bit allocation to other macroblocks in the picture with what left from the target bits. It assures a good visual quality in the logo area, and prevents other macroblocks from catastrophic mode in the low bit rate coding case.

4. SIMULATION RESULTS

To demonstrate the effectiveness of our low cost logo-insertion algorithm, we tested it in a transcoder on some test sequences and compared with the full motion reestimation method. The two schemes use the same rate control algorithm proposed in [7]. We present part of the comparison results with three different sequences: cheer-leader, table-tennis and basketball.

In our simulation, the bit rates of input streams are 8 Mb/s. We inserted logos into part of the bit stream while reducing the bit rate to 6 Mb/s. All the codes are based on the state-of-the-art IBM SW MPEG-2 encoder. We calculated the PSNR (peak signal-to-noise ratio) between 150 pairs of output and input frames and plot them in Figure 1 and Figure 2. Here, PSNR is calculated as $10\log_{10}\frac{255\times255}{D}$ with D being the MSE between the logo-inserted frames and the original input pictures with logo added at the corresponding position on the required pictures. The lines with "*" in the figures are the results of using our proposed algorithms:

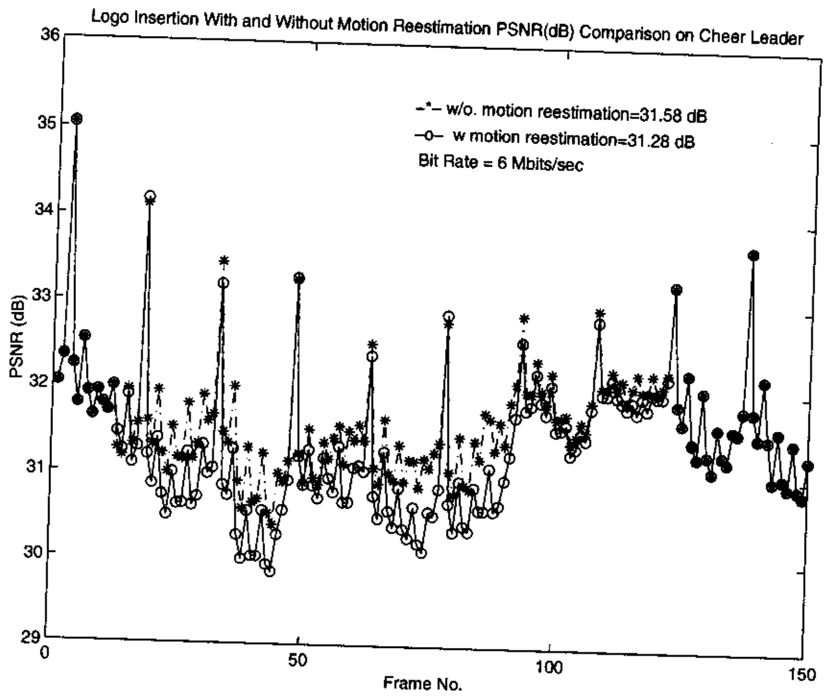

Figure 1: Comparison of Logo Insertion with vs. without Motion Reestimation on Cheer Leader

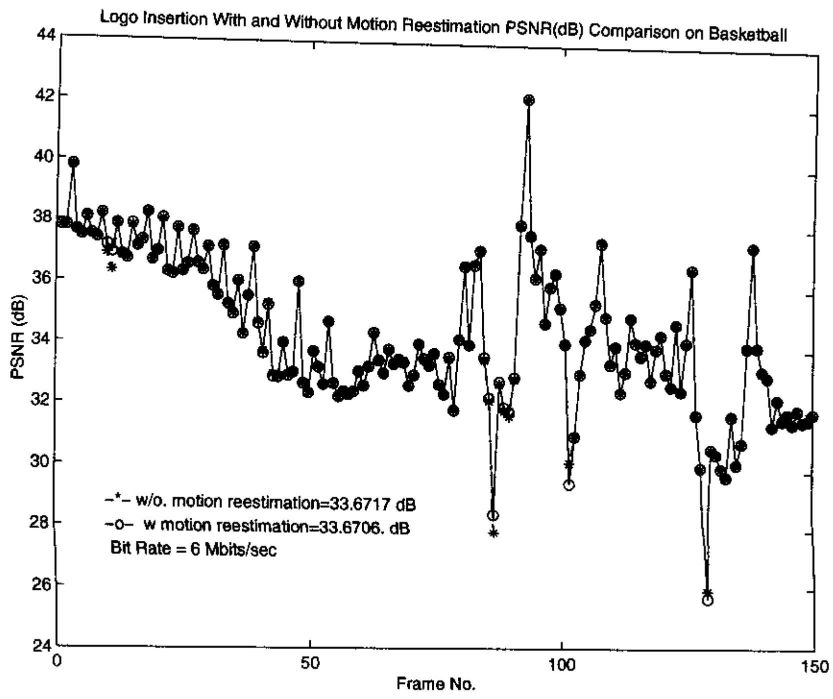

Figure 2: Comparison of Logo Insertion with vs. without Motion Reestimation on Basketball

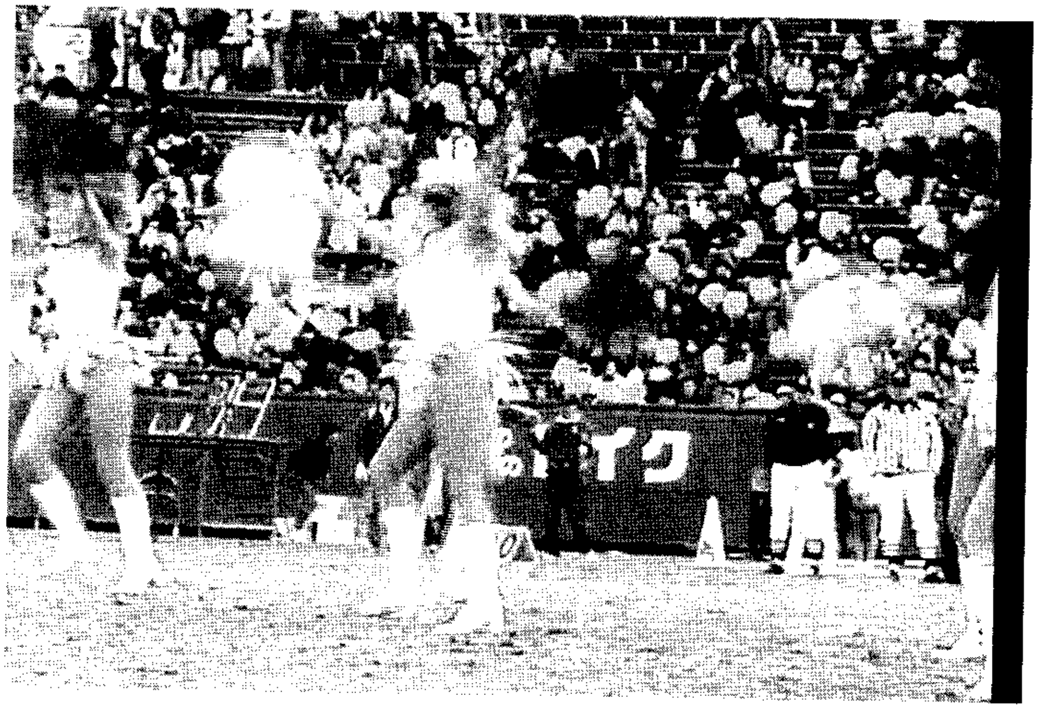

Figure 3: A Frame with Inserted Transparent Logo in Cheer Leader

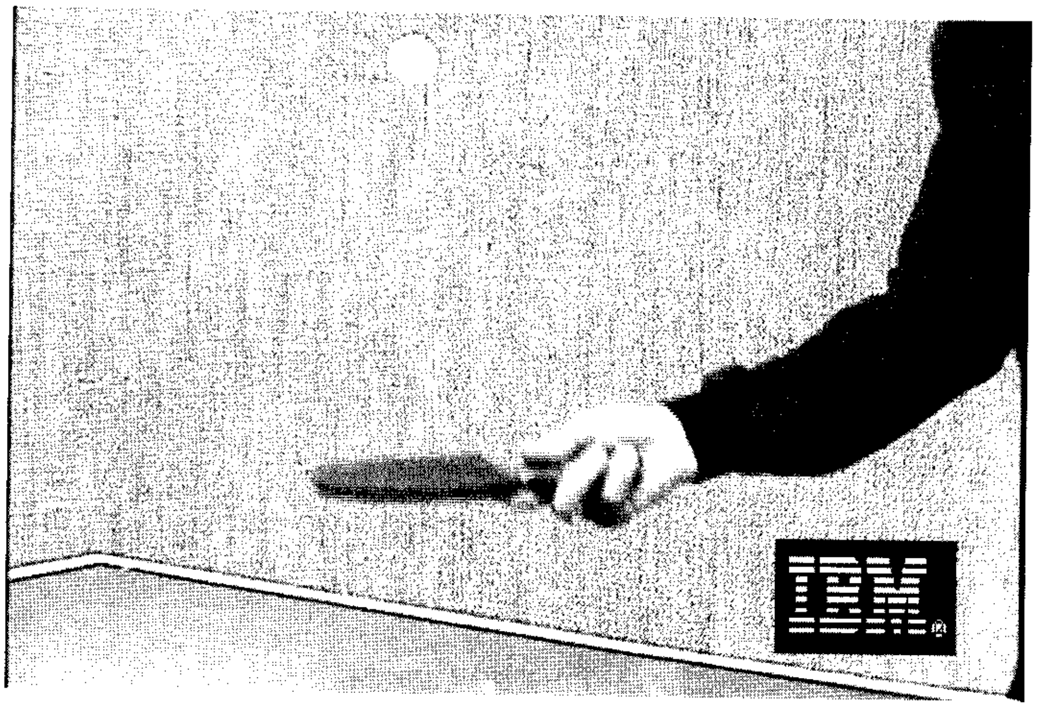

Figure 4: A Frame with Inserted Non-transparent Logo in Table-tennis

logo insertion without motion reestimation, and the lines with "o" are plotted from the results using motion reestimation.

The plots indicate that the performance of our scheme is slightly better: it outperforms the motion-reestimation scheme by 0.3 dB on cheer-leader sequence and 0.001 dB on basketball sequence averagely. This is because the motion reestimatiton is based on the decoded pictures which in general result in worse motion estimation.

Besides the very low computational cost, our logo insertion scheme has also achieved constant visual quality on both transparent and non-transparent logos. Figure 3 and Figure 4 are some frames of inserted transparent and non-transparent logos in the video clips: cheer-leader and table-tennis.

5. CONCLUSION

In this paper, we proposed a low-cost logo-insertion algorithm during transcoding process. We first analyzed the possible frames that might be affected by the inserted logo. Then we described the strategies of modifying the motion vectors in the logo areas and the logo-affected parts separately. To assure good perceptual video quality, we furthermore modified the quantization scales in the logo area. The presented algorithm was demonstrated by simulation results and was compared to the motion-reestimation scheme with the same picture-level rate control algorithm. Our scheme provides compatible video quality, while reducing the computational complexity significantly.

REFERENCES

1. S. Acharya and B. Smith, "Compressed domain transcoding of mpeg," in *Proc. IEEE International Conference on Multimedia Computing and Systems (ICMCS)*, pp. 295–304, (Austin, Texas), June 1998.
2. T. Shanableh and M. Ghanbari, "Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats," *IEEE Trans. on Multimedia* **2**, pp. 101–110, June 2000.
3. P. N. Tudor and O. H. Werner, "Real-time transcoding of mpeg-2 video bit streams," *International Broadcasting Convention (IBC 97)* , pp. 286–301, (Amsterdam, Netherlands), Sept. 12-16 1997.
4. N. Björk and C. Christopoulos, "Transcoder architectures for video coding," *IEEE Trans. on Consumer Electronics* **44**, pp. 88–98, Feb. 1998.
5. P. A. A. Assunção and M. Ghanbari, "Transcoding of single-layer mpeg video into lower rates," in *IEE Proceedings - Vision Image and Signal Processing*, pp. 377–383, Dec. 1997.
6. P. A. A. Assunção and M. Ghanbari, "A frequency-domain video transcoder for dynamic bit-rate reduction of mpeg-2 bit streams," *IEEE Trans. on Circuits and Systems for Video Technology* **8**, pp. 953–967, Dec. 1998.
7. L. Lu, S. Xiao, J. L. Kouloheris, and C. A. Gonzales, "Efficient and low cost video transcoding," in *SPIE Proceedings – VCIP02 (to appear)*, (San Jose, CA), Jan. 2002.