Multiresolution motion estimation for the MPEG coder

G. Calvagno, R. Rinaldo, L. Sbaiz
Dipartimento di Elettronica e Informatica
Via Gradenigo 6/a, 35131 Padova, Italy
Tel: +39-49-827 7731, Fax: +39-49-827 7699, E-mail: calvagno@dei.unipd.it

ABSTRACT

The first step of the coding technique proposed in the MPEG standard is motion compensation. It reduces the residual error energy using a fraction of the total bit rate to transmit motion information. Motion compensation is performed using a block matching approach though the algorithm to compute motion vectors is not given in the MPEG standard. Usually, an exhaustive search around the macroblock position is used. This solution (proposed in the test model) gives the lowest error but has the highest complexity. In this work we propose an algorithm that reduces the complexity of the block matching procedure while achieving comparable performance with the exhaustive search. The proposed solution is particularly attractive for the spatially scalable version of the coder when both a full resolution and a spatially downsampled sequence are transmitted. The algorithm uses a multiresolution motion compensation scheme. Exhaustive search block matching is performed in the downsampled sequence and the vector field computed is used as an estimate of the motion vectors for the full resolution sequence. Thus, only a refinement needs to be computed. This allows a consistent reduction of the computation time with respect to exhaustive search at the full resolution level, while the residual error energy increases only slightly.

Keywords: MPEG, video coding, motion estimation, multiresolution.

1. INTRODUCTION

The coding technique used in the MPEG standard[1] consists essentially of two steps. The first is motion compensation, with the purpose of reducing temporal redundancy, while the second step is the actual coding of the residual error image, i.e., of the difference between the original frame and its motion compensated prediction. Motion compensation should reduce the residual error energy, and use a fraction of the total bit rate to transmit motion information. Of course, there is a trade-off between the coding advantage that results from a small difference image energy and the necessity to reduce the rate to transmit motion information.

The choice made for motion compensation in the MPEG standard is block matching, though the algorithm to compute motion vectors is not explicitly given. A solution, proposed in the test models, is an exhaustive search around the macroblock position. Such solution obviously gives the lowest mean squared error (MSE) but has the highest complexity. In a software implementation of the coder, motion compensation may require 90% of the computation time.

In this work we propose a fully MPEG compatible motion compensation algorithm that reduces the complexity of the block matching procedure while achieving comparable performance with the full search solution. The proposed procedure is particularly attractive for the spatially scalable version of the coder,[1] i.e., when both a full resolution and a spatially downsampled sequence are transmitted. In particular, the algorithm uses a multiresolution motion compensation scheme[2] where exhaustive search block matching is performed in the downsampled

sequence. The vector field computed in this way is used as an estimate of the motion vectors for the full resolution sequence. Thus, only a refinement of the initial estimate needs to be computed for the motion vectors relative to full resolution. This allows a consistent reduction of the computation time with respect to exhaustive search at the full resolution level. In the case of spatial scalability, we consider the two possibilities of sending only the corrections or the complete motion vectors for the full resolution sequence. We remark that the proposed multiresolution motion compensation is fully compatible with the MPEG standard.

2. SPATIAL SCALABILITY IN THE MPEG CODER

In the spatially scalable coder proposed for MPEG, two signals are transmitted. The high resolution, or "up", sequence (typically an HDTV sequence) has a spatial dimension in each direction that is typically two times that of the low resolution, or "down", sequence. This assures compatibility with equipment that can decode the "down" stream but not the high resolution signal.

A general scheme for the spatially scalable coder is shown in Fig. 1.[1] In the scheme, the original high resolution sequence x is filtered and downsampled[3] by a factor two in each direction to give the "down" signal l. The low resolution signal l is then coded with a standard MPEG algorithm, including motion compensation and transform coding.

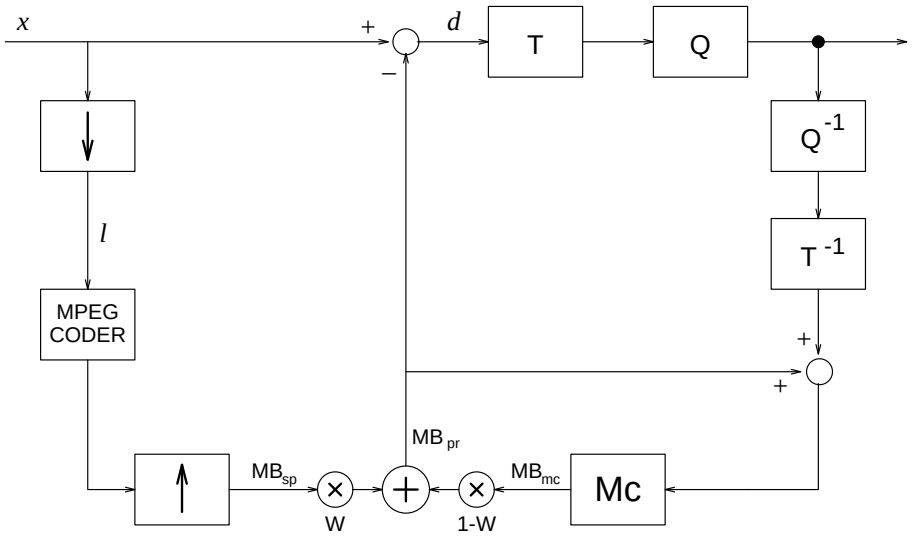

Figure 1: Spatially scalable coder scheme.

The coder of the "up" sequence makes use of the reconstructed images produced by the coder of the "down" sequence by a method called "spatio-temporal weighting". This procedure performs a weighted combination of the temporal prediction from the "up" sequence and the spatial prediction from the "down" sequence on a macroblock basis. As in a standard coder, for each macroblock the motion compensated prediction MB_{mc} is computed together with the related motion vectors. The prediction of a macroblock MB_{pr} is computed by means of a linear combination of MB_{mc} and of the macroblock MB_{sp} obtained by up conversion of the decoded sequence of the "down" level:

$$\mathrm{MB}_{pr} = w\mathrm{MB}_{sp} + (1-w)\mathrm{MB}_{mc}.$$

The weight w is chosen by the coder in a set of possible weights to obtain the minimum prediction error energy and it is coded in the bitstream with a variable length code.

One problem one has to face in the design of a scalable coder is the allocation of the total bit rate B between

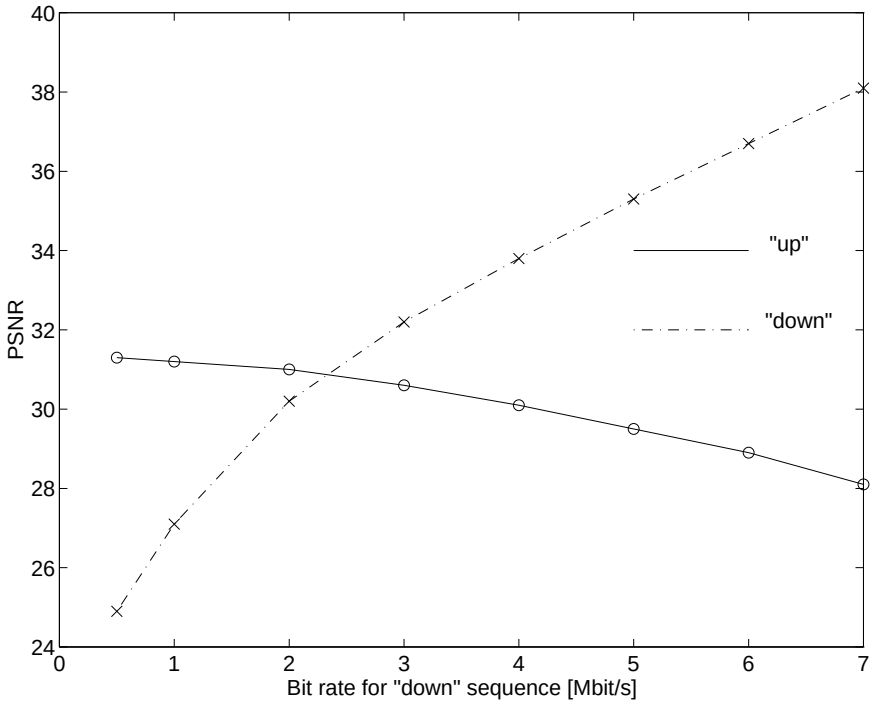

Figure 2: PSNR vs bit-rate for the "up" and "down" sequences (total bit rate B= 9 Mbit/s) for *Calendar*.

the "up" and "down" sequences. Fig. 2 shows a typical plot of the average PSNR for a total bit rate of 9 Mbit/s as a function of the rate B_l allocated to the "down" sequence. Incidentally, the PSNRs for the "up" and "down" sequences are the same when the two sequences are coded using roughly the same number of bit/pixel. Interestingly enough, although the number of bits B_h allocated to code the difference images for the "up" sequence decreases as B_l increases, the PSNR for the high resolution sequence is almost constant. A simple model that can explain this behaviour is as follows.

Suppose that $W = 1$ in Fig. 1, i.e., that the coder always chooses the upsampled "down" sequence to compute the difference image d. We can model d as the sum of a *true* difference image $\bar{d}$, that would result if the downsampled image l were not coded, and of a coding error e_l, namely

$$d = \bar{d} + e_l.$$

By supposing $\bar{d}$ and e_l uncorrelated, we have $\sigma_d^2 = \sigma_{\bar{d}}^2 + \sigma_{e_l}^2$. When we code the difference image d using B_h bit/pixel, we have a variance of the error for the high resolution sequence given by[4]

$$\sigma_{e_h}^2 = \epsilon_h^2(B_h)\sigma_d^2 2^{-2B_h} = \epsilon_h^2(B_h)\left(\sigma_{\bar{d}}^2 + \sigma_{e_l}^2\right) 2^{-2B_h}.$$

By using the fact that $\sigma_{e_l}^2$ is related to the variance σ_l^2 of the "down" sequence l by

$$\sigma_{e_l}^2 = \epsilon_l^2(B_l)\sigma_l^2 2^{-2B_l},$$

we finally obtain

$$\sigma_{e_h}^2 = \epsilon_h^2(B_h)\sigma_{\bar{d}}^2 2^{-2B_h} + \epsilon_h^2(B_h)\epsilon_l^2(B_l)\sigma_l^2 2^{-2(B_h+B_l)}.$$

```
Min_FIELD_1 = MAXINT;
Min_FIELD_2 = MAXINT;
Min_FRAME   = MAXINT;
for (y = -Y_ref; y < Y_ref; y++) {
 for (x = -X_ref; x < X_ref; x++) {
  mv_fl1 = field1_mv_estimate + (x,y);
  AE_FIELD1 = AE_Macroblock( prediction(mv_fl1),
        lines_of_FIELD1_of_current_mb );
  mv_fl2 = field2_mv_estimate + (x,y);
  AE_FIELD2 = AE_Macroblock( prediction(mv_fl2),
        lines_of_FIELD2_of_current_mb);
  mv_fr = frame_mv_estimate + (x,y);
  AE_FRAME = AE_Macroblock( prediction(mv_fr), current_mb);
  if (AE_FIELD1 < Min_FIELD1) {
     MV_FIELD_1 = mv_fl1;
     Min_FIELD_1 = AE_FIELD1;
  }
  if (AE_FIELD2 < Min_FIELD2) {
     MV_FIELD_2 = mv_fl2;
     Min_FIELD_2 = AE_FIELD2;
  }
  if (AE_FRAME < Min_FRAME) {
     MV_FRAME = mv_fl1;
     Min_FRAME = AE_FIELD1;
  }
 }
}
```

Figure 3: Pseudo C code for "up" sequence motion vector computation.

Note that the shape factors ϵ_h^2 and ϵ_l^2 are mildly dependent on the bit rate when uniform quantization and entropy coding are used. For a fixed total bit rate $B = B_h + B_l$, the variance $\sigma_{\tilde{e}_h}^2$ is therefore equal to the sum of a constant term plus a smaller correction term, since $\sigma_{\tilde{d}}^2 \ll \sigma_l^2$. This explains the behaviour observed in Fig. 2.

3. MULTIRESOLUTION MOTION COMPENSATION

In a spatially scalable coder, the complexity of the block matching procedure can be reduced. The proposed algorithm uses a multiresolution motion compensation scheme[2] where exhaustive search block matching is performed in the downsampled sequence. The vectors computed in this way are used as an estimate of the motion vectors for the full resolution sequence. Thus, only a refinement is needed. The algorithm can be described by the pseudo-C code reported in Fig. 3.

The procedure shown in Fig. 3, computes the motion vectors for the "up" sequence by refining the ones computed with an exhaustive search for the "down" sequence. We denote with X_{ref} and Y_{ref} the horizontal and vertical dimensions of the refinement rectangular region, whose area results $A_{ref} = (2X_{ref} + 1)(2Y_{ref} + 1)$. With

the proposed algorithm, the times required to compute the motion vectors for P and B frames[1] are given by

$$T'_P = KA_{ref} \qquad T'_B = 2KA_{ref}, \tag{1}$$

where A_{ref} is the refining area and K is a constant depending on the implementation. The factor 2 in T'_B originates because both forward and backward motion vectors need to be computed for B frames. The computation time for the exhaustive search algorithm is $T_P = KA_P$ for P frames and $T_B = K(A_F + A_B)$ for B frames. Here, A_P denotes the searching area for the P frames, while A_F and A_B are the searching areas for forward and backward prediction in B frames, respectively. Thus the computation time reduction is

$$\frac{T'_P}{T_P} = \frac{A_{ref}}{A_P} \qquad \frac{T'_B}{T_B} = \frac{A_{ref}}{A_F + A_B}. \tag{2}$$

In the coder used for the tests we had $A_P = 91 \times 91$, while the areas A_F and A_B in B frames are 31×31 and 61×61. Assuming a refinement region $A_{ref} = 7 \times 7$, we obtain

$$\frac{T'_P}{T_P} = \frac{49}{8281} \qquad \frac{T'_B}{T_B} = \frac{98}{4682}. \tag{3}$$

The total computation times T''_P, T''_B for motion compensation of the spatially scalable coder are the sum of the times used to compute the vectors for the downsampled sequence and the times needed for the refinements T'_P, T'_B. Since the "down" sequence is downsampled, the number of macroblocks reduces to one fourth with respect to the full resolution sequence. The computation time for exhaustive search in the "down" sequence is therefore one fourth of the times T_P, T_B required in the full resolution sequence, while T'_P and T'_B are negligible. We obtain therefore

$$T''_P \approx \frac{T_P}{4} \qquad T''_B \approx \frac{T_B}{4}. \tag{4}$$

This permits to conclude that we can use the up-down conversion and the vector refinement even if we are not interested to spatial scalability. In such a case, the total computation time is mainly due to the "down" vectors. The time required for the down conversion is in fact negligible.

In summary, the proposed solution computes the vectors for the "up" sequence by adding the refinements to the vectors of the "down" sequence. In this way, we obtain two sets of vectors for the "up" and "down" sequences fully compatible with the MPEG standard. Notice that some of these vectors are not necessarily sent to the decoder, for instance in the case of intra blocks.[1] Clearly, the multiresolution algorithm can be used also in the case of a non-scalable coder in order to reduce the computation time.

As seen, the vectors of the full resolution sequence are obtained from the vectors of the downsampled sequence by adding a small correction. An interesting question concerns the possible performance gain one could achieve by sending only the corrections for the "up" sequence, instead of adding them to the "down" vectors to form complete vectors.

In the next section, we will see that sending only the corrections does not result in a significant performance gain. One problem that arises in this case is that not all the vectors computed by the "down" coder are actually supplied to the decoder. For instance, if a block is coded as an intra block in the "down" sequence, no motion vector will be transmitted. In addition, if a block is coded using frame motion compensation, field motion vectors will not be sent to the decoder. In these cases, the simplest solution is to assume a null vector in the "down" sequence.

4. EXPERIMENTAL RESULTS

Figure 4 reports the average PSNR relative to coding four groups of pictures (GOP) of the CCIR 601 test sequence *Table Tennis* at 6.75 Mbit/s. These results were obtained by setting $w = 0$ in the coder of Fig. 1, i.e.,

without using spatial prediction (this solution is sometimes referred to as "simulcast"). The motion vector field was determined with exhaustive search and with the proposed multiresolution approach for different refinement areas. The algorithm was applied to the "down" sequence obtained using the filter given by the Test Model for HDTV to TV conversion. The computation time for the motion vectors of the "up" sequence alone, using a SPARC Classic for the simulations, was 3548 s for exhaustive search and 1476 s, 67 s, 48 s to compute the corrections in the multiresolution approach with a 31×31, 7×7, 5×5 refinement area, respectively. As explained above, motion vectors of the "up" sequence are computed by adding corrections to the "down" vectors. The vectors are then coded as specified in the MPEG standard, i.e., by using prediction and variable length coding. This ensures full MPEG compatibility since information from the "down" level is used only for motion compensation at the "up" level. We will refer this solution as (a) in the following.

Note that the differences in PSNR for the various refinement areas are negligible. With a refinement area of 7×7 pixels we obtain an average PSNR of 34.40 dB, while a PSNR of 34.68 dB is obtained using full search. Thus, a dramatic reduction of the computation time is achieved at the expense of a very small degradation of the coder performance. The proposed algorithm was also tested allowing the weight w to assume the values 0, 0.25 and 1 for both fields. The results are reported in Fig. 5. Comparing Fig. 4 with Fig. 5, we note that a variable weight w gives a gain of about 0.3 dB for intra frames and 0.4 dB for P frames, while the improvement is negligible for B frames.

Figure 6 and Fig. 7 show the average PSNR for the test image sequence *Flower Garden* using the "simulcast" and the spatially scalable coder with variable w, respectively. We note that, as in the previous case, the reduction of PSNR is slight even for a small refinement area (we obtain an average PSNR of 32.66 dB using full search and 32.44 dB with the proposed algorithm with a refinement area of 7×7 pixels.)

The algorithm was also tested with the HDTV test sequences *Tamburi* and *Banchetto*, coding the "down" sequence at 4 Mb/s and the "up" sequence at 16 Mb/s. The results are given in Fig. 8, Fig. 9, Fig. 10 and Fig. 11. The results for the sequence *Tamburi* are similar to those obtained for *Table Tennis* and *Flower Garden*. The sequence *Banchetto* is interesting because, using the multiresolution approach, we obtain a PSNR that is slightly greater than that obtained with the exhaustive search algorithm. This is because the motion vector field produced by the multiresolution algorithm is smoother than in the case of the full search procedure, and therefore fewer bits are required for coding. Furthermore, the motion compensated difference image variances are not very different in the two cases. Thus, a greater PSNR can be obtained with the multiresolution approach for a given overall bit rate.

In the case of spatial scalability, we investigated the possible performance gain that results from sending only the corrections for the "up" sequence. As a matter of fact, the motion information relative to the down sequence constitutes the initial estimate of the motion field. However, it may happen that no motion vector is transmitted in the downsampled sequence, e.g., when the block is coded as an intra-block. In such a case, we take the initial estimate of the motion vector as the null vector, and send the correction vectors for the full resolution sequence. This strategy will be referred to as solution (b). One can think that solution (b), compared to solution (a), reduces the rate for motion information, but that it leads to a higher MSE because of the missing vectors from the "down" level. In case (a), we have to send for the "up" sequence the complete vectors and not just the corrections, but the refinement procedure can be more accurate.

Table 1 reports some data relative to the coding of the test sequence *Flower Garden* at 6.75 Mb/s for the second GOP (when the bit rate control reaches the steady state). The results are averages for B and P frames. The first column is relative to the exhaustive search algorithm while the second and the third to solutions (a) and (b) with a 7×7 refinement area. The first and the second rows give the average number of bits used to code a vector of a motion compensated macroblock for B and P frames. We note that this quantities decrease from the exhaustive search algorithm to the cases of solutions (a) and (b). Algorithm (a) computes motion vectors using a small refinement area, therefore vectors are more regular than with full search. This leads to a reduction of the number of bits required for their coding. Solution (b) gives an even smaller number of bits per vector because

Table 1: Some coding results for the sequence *Flower garden* at 6.75 Mb/s. Values are the averages from the second GOP.

	Exhaustive algorithm	Case (a)	Case (b)
bits per vector, frames B	7.70	7.37	6.54
bits per vector, frames P	10.21	10.06	6.92
% inter MB in B frames	99.94	99.49	96.72
% inter MB in P frames	99.05	95.01	94.19
residual error variance B frames	76.96	91.37	98.89
residual error variance P frames	203.59	237.10	235.70
PSNR B frames (dB)	32.59	32.32	32.09
PSNR P frames (dB)	32.92	32.79	32.59
bpp B frames	0.387	0.383	0.406
bpp P frames	1.022	1.058	1.010

only the refinements are coded. On the other hand, solutions (a) and (b) give a larger residual error variance and an increased number of intra macroblocks. Examining the PSNRs we conclude that the bits saved for coding of the motion vectors do not compensate for the larger error variance. Hence solution (a) is effective for the reduced complexity while solution (b) does not give significant advantages.

5. CONCLUSIONS

In this paper we proposed a multiresolution approach for the computation of the motion vectors in the MPEG coder. Experimental results confirm that a considerable reduction of the computation time can be achieved at the expense of a negligible degradation of the overall performance. In the case of a spatially scalable coder, we compared the two solutions of sending full vectors for the "up" sequence or only the refinements of the "down" vectors. We found that the former solution gives the best performance still maintaining full compatibility with the MPEG standard.

6. ACKNOWLEDGEMENT

The authors would like to thank Centro Ricerche RAI, Torino, Italy, for kindly supplying the HDTV sequences.

7. REFERENCES

[1] "Coded Representation of Picture and Audio Information," ISO/IEC JCT1/ SC29/ WG 11 MPEG, Test Model 5, Draft Revision 2, pp. 21–23, 7 April 1993.

[2] M. Bierling, "Displacement estimation by hierarchical block-matching," in *SPIE Conference on Visual Communications and Image Processing.*, Boston, pp. 942–951, Nov. 1988.

[3] G. Cortelazzo, G.A. Mian, S. Verri, "Multistage SDTV/HDTV scanning rate converters", *IEEE Trans. Circuits and Systems for Video Technology*, August 1995, pp. 278–297.

[4] N.S. Jayant, P. Noll, *Digital Coding of Waveforms*, Englewood Cliffs, NJ: Prentice Hall, 1984.

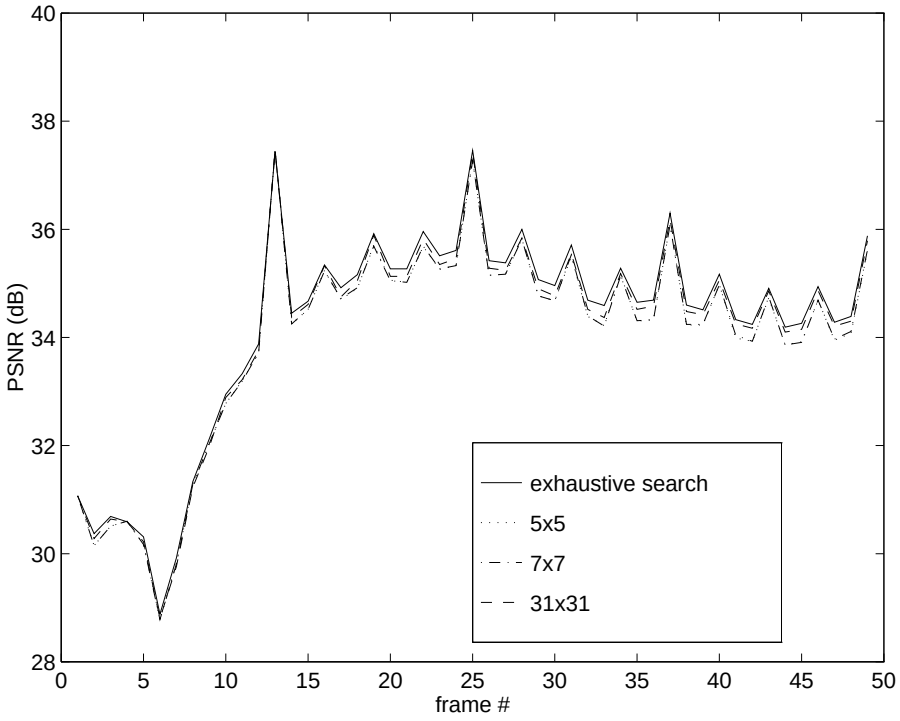

Figure 4: PSNR for the first 4 GOPs of *Table Tennis* and different search techniques using the "simulcast" coder.

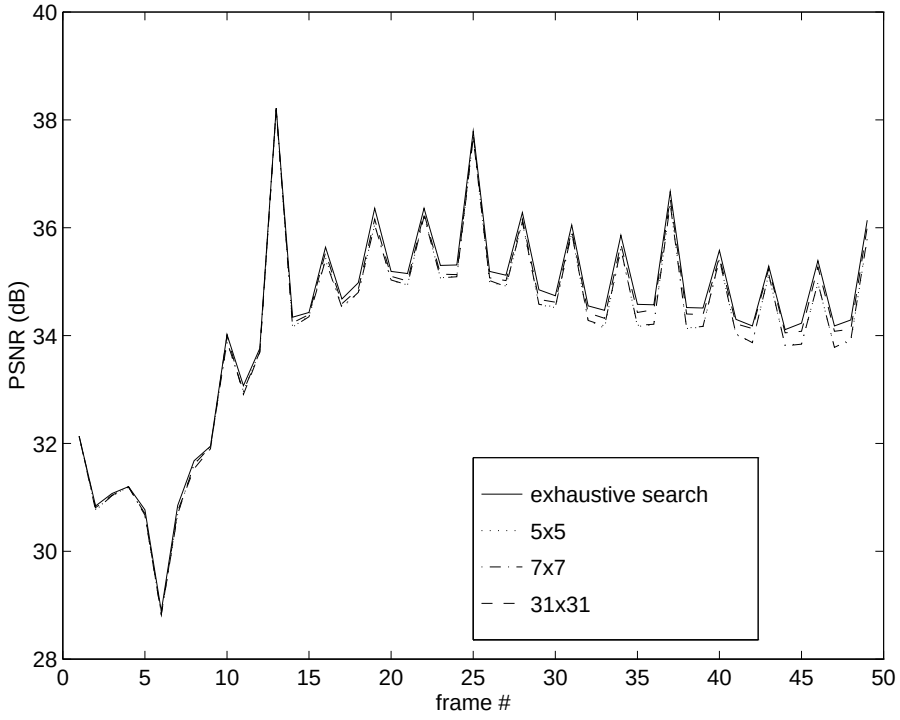

Figure 5: PSNR for the first 4 GOPs of *Table Tennis* and different search techniques using spatial prediction.

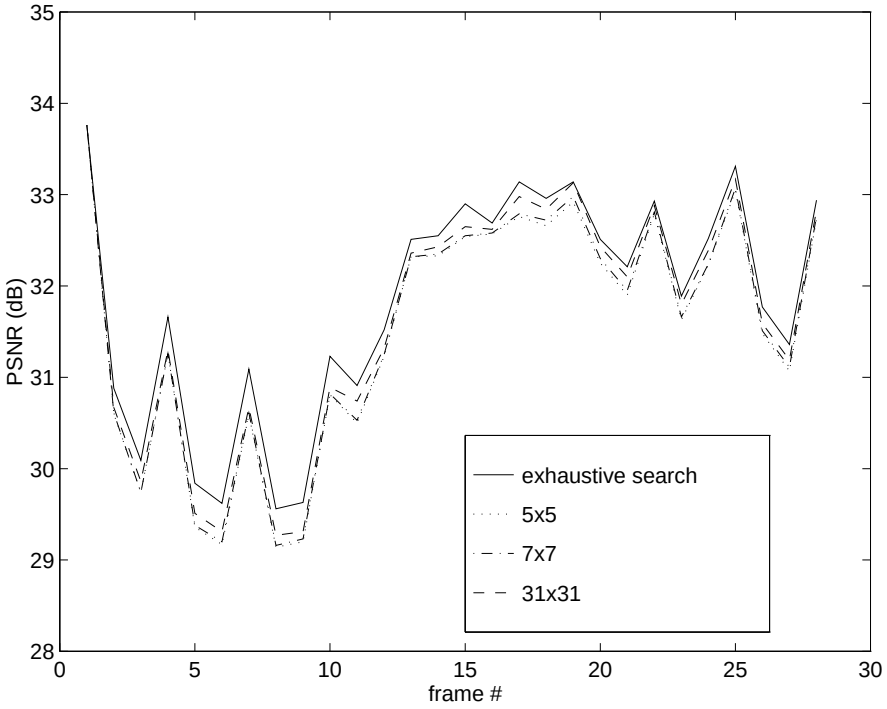

Figure 6: PSNR for the first 28 frames of *Flower Garden* and different search techniques using the "simulcast" coder.

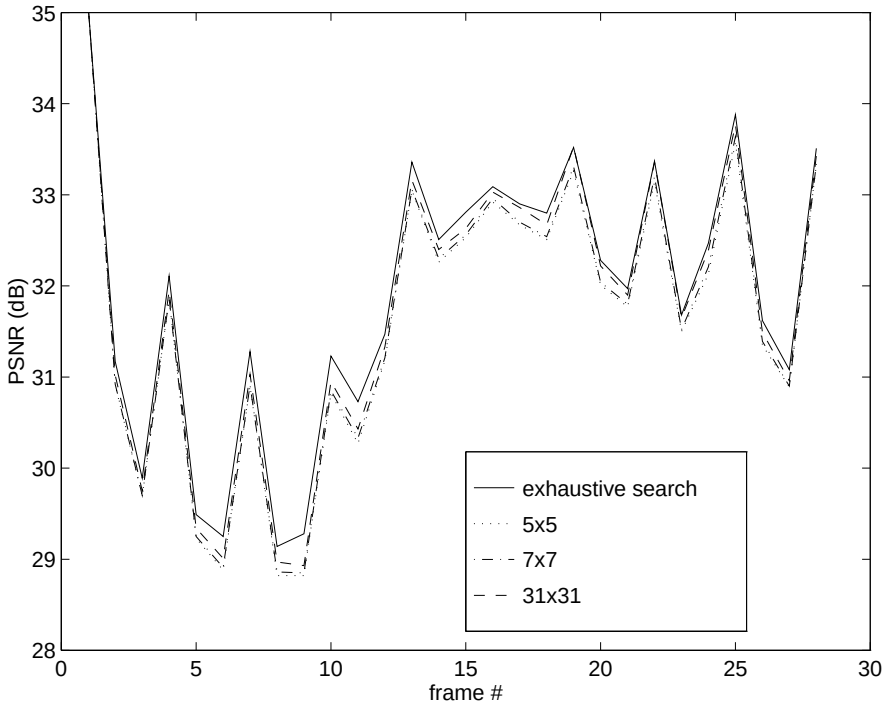

Figure 7: PSNR for the first 28 frames of *Flower Garden* and different search techniques using spatial prediction.

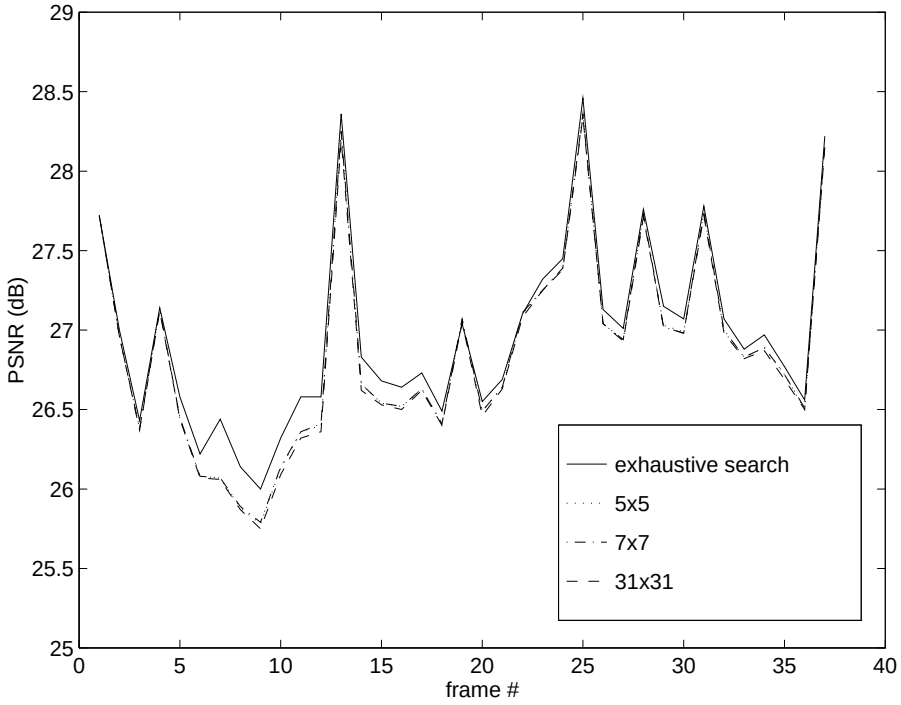

Figure 8: PSNR for the first 3 GOPs of *Tamburi* and different search techniques using the "simulcast" coder.

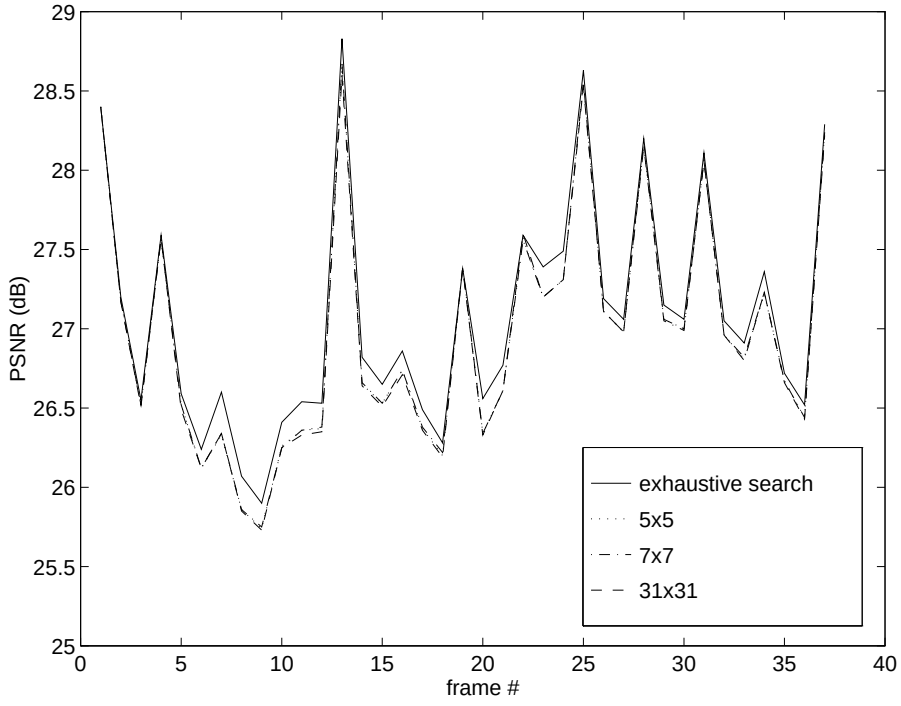

Figure 9: PSNR for the first 3 GOPs of *Tamburi* and different search techniques using spatial prediction.

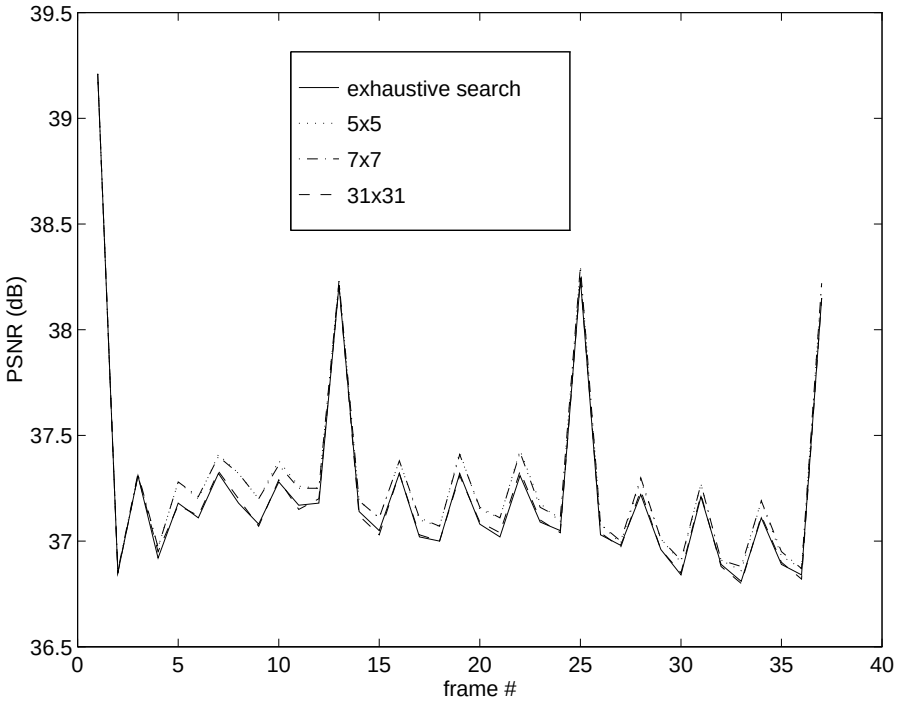

Figure 10: PSNR for the first 3 GOPs of *Banchetto* and different search techniques using the "simulcast" coder.

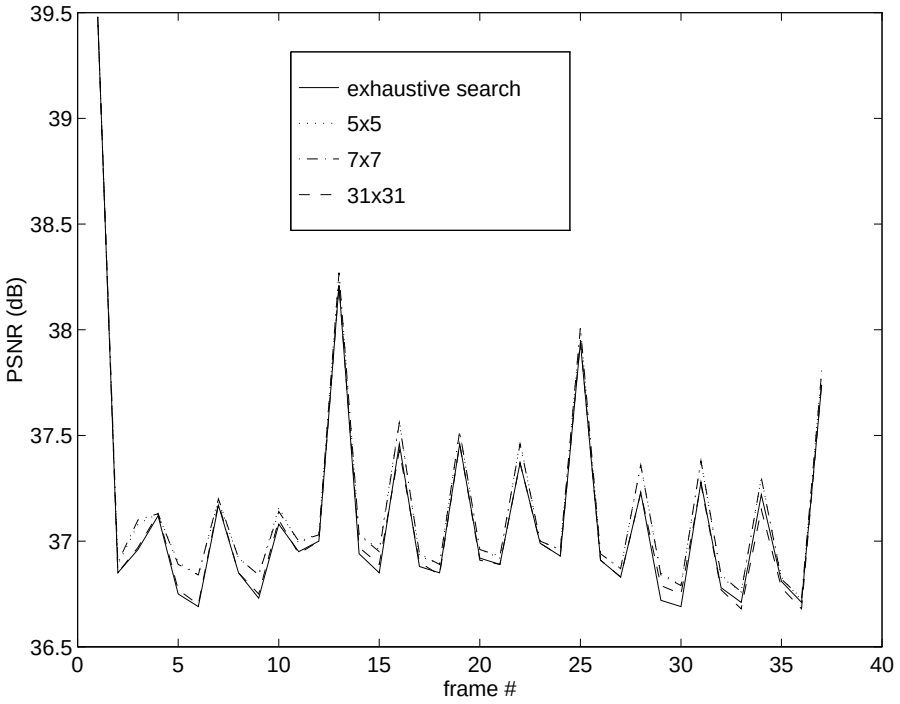

Figure 11: PSNR for the first 3 GOPs of *Banchetto* and different search techniques using spatial prediction.