Fast Color-Embedded Video Coding with SPIHT

Beong-Jo Kim and William A. Pearlman

Electrical, Computer and Systems Engineering Dept.
Rensselaer Polytechnic Institute, Troy, NY 12180, U.S.A.
Tel: (518) 276-6982, Fax: (518) 276-6261
E-mail:beongjo@ipl.rpi.edu, pearlman@ecse.rpi.edu

ABSTRACT

In this paper, we modify the 3-D SPIHT in [11], which is the 3-D extension to image sequence of 2-D SPIHT still image coding [4], in order to allow more flexibility in choosing the number of frames to be processed at one time by introducing unbalanced tree structure. Simulation shows that 3-D SPIHT with reduced coding latency still achieves coding results comparable to MPEG-2 at bit-rate around 500 Kbps – 2.53 Mbps with lower computational time, and exhibits more uniform PSNR fluctuations. In addition, extension to color video coding is accomplished without explicit rate-allocation and can be used to any color-plane representation. The results of color-embedded coding are comparable to H.263 at very low bit-rate around 30 Kbps – 60 Kbps.

1 INTRODUCTION

Video coding achieves high coding efficiency by exploiting temporal correlation among pixels in successive frames of an image sequence. Motion compensated predictive coding (MCP) is a widely accepted coding method, which has a hybrid structure whereby subband/transform coding in the spatial domain is applied to MC prediction error signals. In fact, MCP has been employed by the several international video coding standards such as MPEG-1, MPEG-2, and H.263. Another approach is 3-D subband/wavelet coding with/without MC [5, 8, 6, 7, 12]. They are straightforward extensions of 2-D subband/wavelet to temporal domain, and are different mainly in the way generating 3D subbands and coding methods applied to the subbands. A strong inter-frame correlation manifests itself as an energy compaction in the temporal low-subbands. The

advantages of 3-D subband/wavelet coding scheme are the lower computational complexity comparing to MCP method, none-recursive structure that limits error propagation within a certain size of video segment, and multiresolutional property for possible video scalability.

Inspired by excellent results of the SPIHT [2] (set partitioning in hierarchical trees) for image coding scheme by Said and Pearlman, there has been extensive research on zero-tree based coding for video compression. There are two main streams for video coding using zero-tree structure. One kind of scheme uses conventional predictive coding method with/without motion compensation to first remove inter-frame correlation. Then, it applies two-dimensional discrete wavelet transformation (2-D DWT) in each residual frame to reduce spatial redundancy and set up a 2-D hierarchical structure [10]. The other method, which seems to be more powerful and efficient, uses 3-D discrete wavelet transformation (3-D DWT) to remove spatio-temporal redundancy, compact energy in particular low spatio-temporal frequency subbands, and set up 3-D spatio-temporal hierarchical structure to better adapt to the 3-D nature of video.

Recently, 3-D extensions, 3-D IEZW (improved embedded zero-tree coding) [9], from 2-D IEZW [1] and 3-D SPIHT [11], from 2-D SPIHT [2], which have the properties of fast encoding/decoding, progressive transmission, complete adaptiveness (no training), precise rate control, and very simple implementation, reported excellent results without any motion compensation method. Those results reported were better than MPEG-2's with its complicated motion compensation in terms of PSNR (peak signal to noise ratio) as well as visual quality, especially at low bit-rate.

In previous 3-D SPIHT [11], the 9/7 bi-orthogonal Daubeachies' filter was used for 3-D DWT with the same number (3 times) of decompositions applied to the two spatial directions and to 16 frames in the temporal direction. This constraint to three levels of decomposition is the most that can be applied to the temporal direction, so that limiting the spatial levels to three may prevent further exploitation of spatial redundancy with larger size of video. Furthermore, even sixteen frame processing may cause unacceptable coding delay for transmission on some channels.

In this paper, we employ a modification of the 3-D SPIHT tree structure to have the flexibility to choose different numbers of decompositions between the temporal and spatial domains, in order to select a smaller number of frames to process at one time. In this way, we can decide what is the best trade-off between performance, coding delay, and memory requirements with the capability of using a larger number of decompositions in the spatial domain to compensate for possible loss of coding performance from reducing the number of frames processed at one time. With this smaller coding unit and structural freedom, there are several possible implementation options in terms of filter choice, and number of frames in one coding unit. In addition, we will consider embedded color video coding scheme, which allows implicit bit allocations among color planes, and generates one mixed color bit-stream so that we can stop decoding at any point of the bit-stream

and reconstruct the color video sequence of best quality at the given bit-rate.

The organization of this paper is as follows: Section 2 reviews basic principles of SPIHT. Unbalanced trees are addressed in section 3 to allow smaller number of group of frames (GOF). Embedded color video coding scheme is discussed in section 5. Section 6 provides computer simulation results. Section 7 concludes the paper.

2 3-D SPIHT

3-D SPIHT algorithm is based on three basic concepts: (1) code/transmit important information first based on the bit-plane representation of pixels (2) ordered refinement bit-plane transmission, and (3) coding is performed along the predefined path/trees called *spatio-temporal orientation trees*, which efficiently exploit the properties of a 3-D wavelet transformed video.

3-D SPIHT consists of two main stages as in 2-D SPIHT [4]: sorting and refinement. In the sorting stage, 3-D SPIHT sorts pixels by magnitude with respect to a threshold, which is a power of two, called the level of significance. However, this sorting is a partial ordering, as there is no prescribed ordering among the coefficients with the same level of significance or highest significant bit. The sorting is based on the significance test of pixels along the spatio-temporal orientation trees rooted from the highest level of the pyramid in the 3-D wavelet transformed video.

Spatio-temporal orientation trees were introduced to test significance of groups of pixels for efficient compression by exploiting self-similarity and magnitude localization properties in a 3-D wavelet transformed video. In other words, the 3-D SPIHT exploits the circumstance that if a pixel in the higher level of pyramid is insignificant, it is very likely that its descendants are insignificant.

For practical implementation, 3-D SPIHT maintains three linked lists, the list of insignificant pixels (LIP), the list of significant pixels (LSP), the list of insignificant sets (LIS). At the initialization stage, 3-D SPIHT initializes the LIP with all the pixels in the highest level of the pyramid, the LIS with all the pixels in the highest level of pyramid except the pixels which do not have descendants, and the LSP as an empty list. The basic function of actual sorting algorithm is to recursively partition sets in the LIS to locate individually significant pixels, insignificant pixels, and smaller insignificant sets and move their co-ordinates to the appropriate lists, the LSP, LIP and LIS,respectively. After each sorting stage, 3-D SPIHT outputs refinement bits at the current level of bit significance of those pixels which had been moved to the LSP at higher thresholds. In this way, the magnitudes of significant pixels are refined with the bits that decrease the error the most. This process continues by decreasing the current threshold successively by factors of two until the desired bit-rate or video quality is reached. One can refer to [4] and [11] for more details.

3 UNBALANCED TREES

In 2-D SPIHT [4], a dyadic decomposition is used, keeping the same number of horizontal and vertical decompositions, and spatial orientation trees utilize the pyramid structure to efficiently code image. Hence, every significance test of a pixel and its descendants ends after going through the same number of levels in the trees. In other words, every tree terminates after the same level.

Having the natural extension of 2-D spatial orientation trees to 3-D spatio-temporal orientation trees, we are forced to have the same number of recursive decompositions (balanced tree) in the spatial and temporal domain. In addition, due to the structure of a node in 3-D SPIHT, it is necessary to maintain at least two frames for the same temporal lowest subband, which, coupled with number of decomposition levels, affects directly the number of frames in the GOF. For example, the 3-D SPIHT will require a minimum of four frames for one level spatio-temporal decomposition. However, for an efficient subband coding of image, it has been known that three or more recursive spatial decompositions are required depending on the image size. In general, more dyadic spatial decompositions are required for a larger image size for a better performance. The previous 3-D SPIHT [11] requires $F = 16$ frames for $l = 3$ levels of decomposition in both temporal and spatial domain. This constraint to three levels of decomposition is the most that can be applied to the temporal direction, so that limiting the spatial levels to three may prevent further exploitation of spatial redundancy for a relatively large size of video sequence. Furthermore, even 16 frame processing may cause unacceptable coding delay for interactive video application. The same problem occurs in 3-D IEZW [9].

To achieve more flexibility in choosing the number of frames in GOF by separating number of decompositions in temporal and spatial domain, we should allow *unbalanced trees*. Suppose that we have three levels of spatial decomposition, and one level of temporal decomposition with GOF = 4. Then, a pixel with coordinate of $(i, j, 0)$ has a longer descendant tree (three levels) than that of a pixel with coordinate of $(i, j, 1)$ (one level), since any pixel with temporal coordinate of zero does not have its descendants in the temporal direction. Thus, the descendants trees in the later significance tests terminate sooner than those of the first. This modification can be made by keeping track of two different kinds of pixels in this case. One has a tree of three levels, and the other has a tree of one level.

With this smaller GOF, and structural freedom, there are several possible implementation options in terms of filter choice, and capability of a larger number of decompositions in the spatial domain to compensate for a possible loss of coding performance from reducing the number of frames in GOF. For example, it would be a better choice to use a shorter filter for short segments (4 frames) of video sequence such as S+P transform, which is known to be very efficient and uses only integer operations [3].

4 COLOR-EMBEDDED VIDEO CODING

So far, we have considered only one color plane, namely luminance. In this section, we will consider a simple application of the 3-D SPIHT to any color video coding, while still retaining full embeddedness, and precise rate control.

A simple application of the 3-D SPIHT to color video would be to code each color plane separately as does a conventional color video coder. Then, the generated bit-stream of each plane would be serially concatenated. However, this simple method would require allocation of bits among color components, losing precise rate control, and would fail to meet the requirement of the full embeddedness of the video codec since the decoder needs to wait until the full bit-stream arrives to reconstruct and display. Instead, one can treat all color planes as one unit at the coding stage, and generate one *mixed* bit-stream so that we can stop at any point of the bit-stream and reconstruct the color video of the best quality at the given bit-rate. In addition, we want the algorithm to automatically allocate bits optimally among the color planes. By doing so, we will still keep the full embeddedness and precise rate control of 3-D SPIHT. The bit-streams generated by both methods are depicted in the Fig. 1, where the first one shows a conventional color bit-stream, while the second shows how the color-embedded bit-stream is generated, from which it is clear that we can stop at any point of the bit-stream, and can still reconstruct a color video at that bit-rate as opposed to the first case.

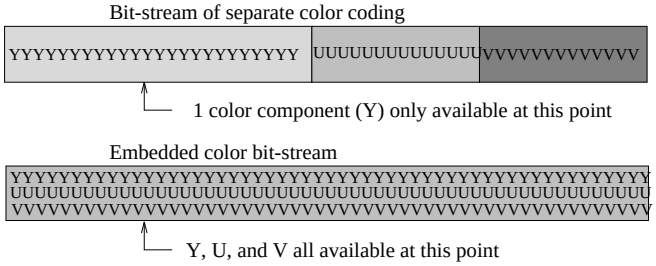

Figure 1: Bit-streams of two different methods: separate color coding (exact order of Y, U, and V) and embedded color coding (Y, U, or V selected at any point)

Let us consider a tri-stimulus color space with luminance Y plane such as YUV, YCrCb, etc. for simplicity. Each such color plane will be separately wavelet transformed, having its own pyramid structure. Now, to code all color planes together, the 3-D SPIHT algorithm will initialize the LIP and LIS with the appropriate coordinates of the top level in all three planes. Fig. 2 shows the initial internal structure of the LIP and LIS, where Y,U, and V stand for the coordinates of each root pixel in each color plane. Since each color plane has its own spatial orientation trees, which are mutually exclusive and exhaustive among the color planes, it automatically assigns the bits among the planes according to the significance of the magnitudes of their own coordinates. The effect of the order in which the root pixels of each color plane are initialized will be negligible

except when coding at extremely low bit-rate.

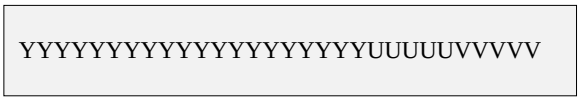

Figure 2: Initial internal structure of LIP and LIS, assuming the U and V planes are one-fourth the size of the Y plane.

5 Coding results

First, we will see the effect of size of GOF on the performance of 3-D SPIHT. In general, a larger size of GOF is expected to give better rate-distortion performance. Unfortunately, temporal filtering causes inevitable coding latency. However, this poses a problem only for interactive video applications [6]. We use three different GOFs (4, 8, 16) to test with the same video sequences, and analyze in terms of specific camera movement. With GOF of 4, and 8 frames, S+P transform [3] is used for 1-level and 2-level temporal decomposition respectively, while with GOF of 16 frames, the 9/7 bi-orthogonal Daubechies' wavelet filters are used. In all cases, we performed three levels of spatial decomposition with the same 9/7 Daubechies' wavelet filters.

From frame 1 to 30, the camera is almost still, and the only movements are the bouncing ping pong ball, and hands of the player, where, as shown in Fig. 3 of "table tennis", the GOF of 16 outperforms the smaller size of GOF. Then, the camera begins to move away (or zoom out from frame 31 - 67), and the whole body of the player shows up, where GOF size does not seem to affect the performance very much. In this region, MPEG-2 obviously catches up with the camera zooming better than 3-D SPIHT. Then, there occurs a sudden scene change at frame 68 (a new player begins to show up). Note the interesting fact that the GOF of 16 is not affected very much by the sudden scene change since the scene change occurs in the middle of 16 frame segment of frames 65 - 80, which are simultaneously processed by 3-D SPIHT, while GOF of 4 suffers severely from the scene change of frame 68 which is the last frame of 4 frame segment of frames 65 - 68. With several simulations, we found that larger size of GOF is in general less vulnerable or even shows a good adaptation to a sudden scene change. After the frame 68, the sequence stays still, and there is only small local motion (swinging hands), which is effectively coded by 3-D SPIHT. 3-D SPIHT more quickly tracks up to high PSNR in the still region compared with MPEG-2 in the Fig. 3, where MPEG-2 shows lower PSNR in that region.

In Fig. 4 of "football", we again see that GOF of 16 outperforms the smaller GOFs. It has been quite a surprise that 3-D SPIHT is better than MPEG-2 for all different GOF sizes. It seems that the conventional block matching motion compensation of

Sequence	GOF=4(dB)	GOF=8 (dB)	GOF=16 (dB)	MPEG-2 (dB)
tennis	29.59	30.04	30.95	30.27
football	27.45	27.36	27.94	26.90

Table 1: Coding results with different GOFs at 0.3 bpp (760 kbits) (Average PSNR in dB)

MPEG-2 to find translational movement of blocks in the frame does not work well in this sequence, where there is much more complex local motion everywhere and not much global camera movement. This similar phenomenon has also been reported by Taubman and Zahkor [6].

Average PSNRs with different GOFs are shown in Table 1 for the performance comparison, in which we can again observe that GOF = 16 gives the best average PSNR performance at the cost of longer coding delay and more memory. However, the gain is not much in the "football" sequence with complex motion and stationary camera.

To demonstrate the visual performances with different GOFs, reconstructed frames of the same frame number (8) at bit-rate of 0.2 bpp (or 507 Kbps) are shown in Fig. 5, where GOF = 16 performs the best. With GOF = 4, 3-D SPIHT still shows the visual quality comparable to or better than MPEG-2. Overall, 3-D SPIHT suffers from blurs in some local regions, while MPEG-2 suffers from both blurs and blocking artifacts as can be shown in Fig. 5, especially when there exists significant amount of motion.

Although it might have been expected that 3-D SPIHT would degrade in performance from motion in the sequence, we have seen that 3-D SPIHT is working relatively well for local motion, even better than MPEG-2 with motion compensation on the "football" sequence. Even with the similar average PSNR, better visual quality of reconstructed video was observed. As in the most 3-D subband video coders [9], one can observe that PSNR dips at the GOF boundaries partly due to the object motion and partly due to the boundary extension for the temporal filtering with GOF = 16. However, they are not visible in the reconstructed video. With smaller number of GOF such as 4 and 8, more uniform frame-by-frame PSNR fluctuation was observed.

Next, we provide simulation results of color-embedded coding of 4:2:0 (or 4:1:1) chrominance subsampled QCIF (176×144) sequence. We tested "hall monitor" sequence at relatively low bit-rate and compared with H.263 which is the latest test model (test model number 2.0) downloaded from the public domain (ftp://bonde.nta.no/pub/tmn/). All advanced options (-D -E -F) of H.263 were activated. Note that not many video coding schemes perform reasonably well in both high bit-rate and low bit-rate. To demonstrate embeddedness of 3-D color video coder, we reconstructed video at bit-rates of 30 Kbps and 60 Kbps, and frame-rate of 10 fps by coding every third frame, with the same color-embedded bit-stream compressed at higher bit-rate 100 Kbps. In this simulation, we chose GOF = 16, since required memory and computational time are

Bit-rate	30 Kbps (Y U V) dB	60 Kbps (Y U V) dB
3-D SPIHT	32.95 38.15 40.68	37.95 40.41 42.38
H.263	33.84 37.79 40.03	37.50 39.81 41.65

Table 2: Coding Results of "Hall monitor" sequence (frames 0 - 285) at bit-rates of 30 Kbps and 60 Kbps, and frame-rate of 10 fps

	3D-SPIHT	MPEG-2
Actual time/frame	2.44 sec	8.63 sec
Relative time/frame	1	3.5

Table 3: Coding is performed on Sun-sparc 10 machine. Total 48 frames were coded at bit-rate of 0.3 bpp (760 Kbps).

reasonable with QCIF sized video. The average PSNRs coding frame 0 – 285 (96 frames coded) are shown in the Table 2, in which we can observe that 3-D SPIHT gives average PSNRs of Y component slightly inferior to those of H.263 and average PSNRs of U and V components slightly better than those of H.263. However, visually 3-D SPIHT and H.263 show competitive performances as shown in Fig. 6. Overall, color 3-D SPIHT still preserves precise rate control, self-adjusting rate allocation according to the magnitude distribution of the pixels in all three color planes, and full embeddedness.

Lastly, we provide computational complexity in terms of encoder's running time. Absolute and relative encoding time per frame are shown in Table 3, where we obviously observe that the amount of computations required for 3-D SPIHT is much smaller than that for MPEG-2. Even comparing to H.253, we also observed 3-D SPIHT was 2.5 times faster. With 3-D SPIHT, most execution time is spent on 3-D wavelet transform, and actual SPIHT coding time takes about 4.5 % of total execution time. The decoding time is almost the same, since the 3-D SPIHT has the structure symmetric to that of the encoder.

6 CONCLUSION

In this paper, a modification of 3-D SPIHT is presented to achieve lower coding latency and flexibility of design. It should be pointed out that, by allowing the unbalanced trees, we can allow any number of recursive spatial decompositions for larger size of video regardless of the number of frames to be processed at one time. In addition, color-embedded coding was obtained and preserves the desired properties such as automatic rate allocation, precise rate control. The general conclusions of the paper are: (1) lower coding latency is achieved at the cost of small coding performance loss with 3-D SPIHT, (2) color-embedded coding is done without losing the desired properties and efficiency of 3-D SPIHT, and (3) 3-D SPIHT exhibits reproduction quality better than or at least

comparable to MPEG-2 and H.263 with considerably faster encoding time.

References

[1] A. Said, "An Improved Zero-tree Method for Image Compression", *IPL Technical Report*, TR-122, Image Processing Laboratory, Rensselaer Poly technic Institute, Nov. 1992.

[2] A. Said and W. A. Pearlman, "Image Compression Using the Spatial- Orientation Tree", *IEEE. Intl. Symp. Circuits and Systems (Chicago)*, pp. 279-282, May 1993.

[3] A. Said and W. A. Pearlman, "Reversible image compression via multiresolution representation and predictive coding", *Proc. SPIE*, vol. 2094 pp. 664–674, Nov. 1993.

[4] A. Said and W. A. Pearlman, "A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", *IEEE Transactions on Circuits and Systems for Video Technology*, vol.6 pp. 243–250, June 1996.

[5] G. Karlsson and M. Vetterli, "Three Dimensional Subband Coding of Video," *Proc. Int. Conf. on Acoust., Speech, and Signal Processing (ICASSP)*, pp. 1100 - 1103, Apr. 1988.

[6] D. Taubman and A. Zakhor, "MultiRate 3-D Subband Coding of Video," *IEEE Trans. Image Processing*, pp. 572 - 588, VOL. 3, Sep. 1994.

[7] J. R. Ohm, "Three-Dimensional Subband Coding with Motion Compensation," *IEEE Trans. Image Processing*, pp. 559 - 571, VOL. 3, Sep. 1994.

[8] C.I. Podilchuk and N.S. Jayant and N. Farvardin, "Three-Dimensional Subband Coding of Video", *IEEE Trans. Image Proc.*, vol. 4, pp. 125-139, Feb. 1995.

[9] Y. Chen and W. A. Pearlman, "Three-Dimensional Subband Coding of Video Using Zero-Tree Method", *Proc. SPIE*, pp.1302-1309, Mar. 1996.

[10] S.A. Martucci and I. Sodagar and T. Chiang, "A Zerotree Wavelet Video Coder", *IEEE Trans. Video Tech.*, pp. 109-118, Feb. 1997.

[11] B.J Kim and W. A. Pearlman, "An Embedded Wavelet Video Coder Using Three Dimensional Set Partitioning in Hierarchical Trees", *Data Compression Conf.*, pp. 251-260, Mar. 1997.

[12] S. J. Choi, "Motion-compensated 3-D Subband Coding of Video," *Submitted to IEEE Trans. Image Processing*, 1997.

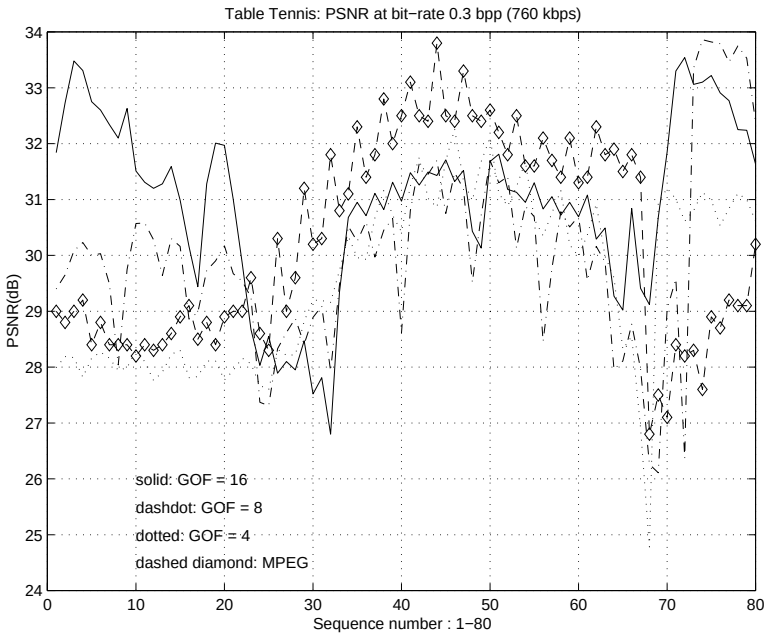

Figure 3: Comparison of PSNR with different GOFs (4, 8, 16) on table tennis sequence at 0.3 bpp (760 kbits)

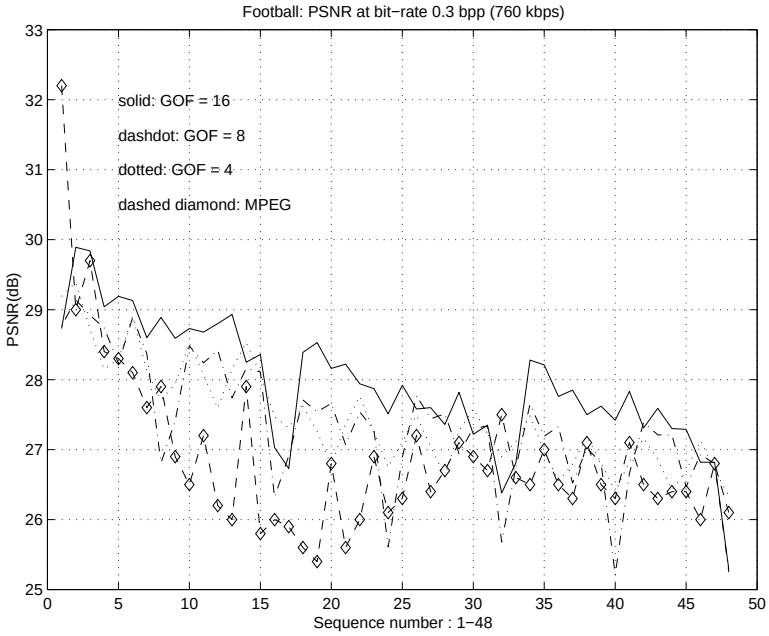

Figure 4: Comparison of PSNR with different GOFs (4, 8, 16) on football sequence at 0.3 bpp (760 kbits)

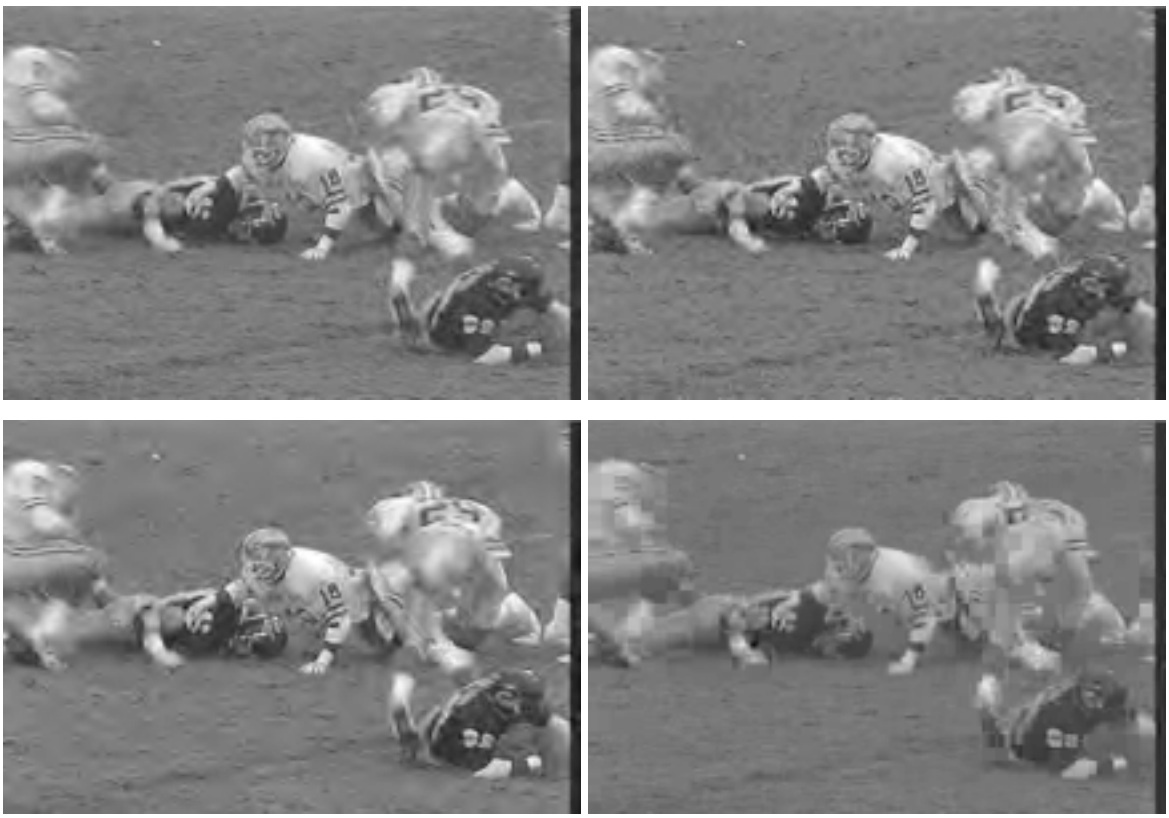

Figure 5: Comparison of visual performance of 3-D SPIHT with GOF = 16 (top-left), GOF = 8 (top-right), GOF = 4 (bottom-left) and MPEG-2 (bottom-right) at bit-rate of 0.2 bpp

Figure 6: Comparison of visual performance (luminance or Y component) of 3-D SPIHT with GOF = 16 (top) and H.263 (bottom) at bit-rate of 60 Kbps and frame-rate of 10 fps