Video Compression With Dense Motion Fields

Soo-Chul Han and Christine I. Podilchuk, *Member, IEEE*

***Abstract*—We propose a motion-compensated video coding system employing dense motion fields. The dense motion field is calculated at the transmitter, and the motion information is efficiently encoded and transmitted along with the residual frame. The motion estimation is performed by existing techniques in the literature, while we focus on the coding of the motion field and the displaced frame difference (DFD) frame. The dense motion field formulation leads to several novel and distinct advantages. The motion field is encoded in a lossy manner to make the motion rate manageable. The more accurate and precise motion description allows us to predict where the DFD energy will be significant, thus leading to a more efficient DFD encoder compared to applying traditional still-image coding techniques. Furthermore, the dense motion field framework allows us to refine and tailor the motion estimation process such that the resulting DFD frame is easier to encode. Simulations demonstrate superior performance against standard block-based coders, with greater advantages for sequences with more complex motion.**

***Index Terms*—Markov random field, motion compensation, motion field, video compression.**

I. Introduction

RESEARCH in digital video compression has received considerable attention over the last decade and has led to several international standards. Most, if not all, video coding schemes utilize motion compensation as a means of reducing the temporal redundancy between neighboring frames, and it has been widely established that motion compensated prediction (MCP) improves the overall performance of video coding. Many have investigated a wide variety of motion estimation and compensation techniques in the hopes of improving coding efficiency, computational complexity, scalability, and/or other factors.

Among the many techniques introduced, the block-based motion estimation and compensation method has proven to be the most popular due to its simplicity and minimal overhead information. Indeed, improvement and refinement of the block-based scheme has been a focal point of research and development in the video compression community. However, there are several inherent deficiencies with the block-based schemes, some of which have been well publicized, and some which have not. First, the block-based model fails to capture the true motion in natural video. A number of researchers have pointed out this problem, proposing alternative solutions such as variable-size block motion [1] and arbitrarily-shaped parametric motion representations [2].

Another intrinsic problem with existing motion-compensated predictive coders that is just beginning to draw attention is the coding of the residual frame or displaced frame difference (DFD). Typically, the DFD frame is encoded by applying transform coding techniques such as the discrete cosine transform (DCT), which theoretically and by design work well on still images. However, such methods are quite inefficient on the DFD frame, which consists predominantly of high-frequency data. This is especially objectionable when one considers that a good majority of the overall rate is spent on coding the DFD frame in most applications of existing block-based coders.

Finally, few have studied the issue of relating the motion estimation process and the DFD coding process. In most cases, the motion estimation and compensation is performed, *and then* the DFD coding is done. In other words, the motion estimation is performed without explicit regard as to how the resulting DFD frame will be encoded. For example, in the popular block-based predictive video coders, the motion estimation is performed to minimize a prediction error criterion such as the mean-square error (MSE). Thus the motion estimation is merely trying to get a best prediction in the mean-square sense, not taking into account how the subsequent error frame will be encoded nor whether it can be encoded efficiently. It would be beneficial to somehow tailor the motion estimation such that the ensuing DFD coding process is taken into account.

More recently, several authors have introduced forward motion-compensated video coders (i.e., the motion information is calculated at the encoder and transmitted) employing a *dense* motion field [3]–[6]. However, they had limited success in achieving good overall video coding performance, mainly because they could not code the motion field down to the truly densest level. All of them concentrate more on the process of modeling and estimating the motion field itself, while relatively simple and conventional schemes are used in the motion and DFD coding. In [3], a dense motion field together with a segmentation of the motion is found by Markov random field (MRF) modeling. The dense motion is then represented by fitting a parametric model to each region. No overall coding results were given. In [4], a variation of Horn and Schunk's optical flow field [7] is applied to video coding with some success. However, the proposed scheme have trouble going down to the densest (pixel) level, and the resulting DFD is encoded by intra-frame techniques. The authors in [5] also use optical flow fields and apply morphology to reduce the dense motion to a manageable scale. In [6], a dense motion field based on Konrad and Dubois' MRF formulation [8] is used. Again, the motion rate becomes inhibitive to encode when going down to the densest level.

Manuscript received January 4, 2000; revised July 1, 2001. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. A. Enis Cetin.

S.-C. Han is with C-Cube Microsystems, Milpitas, CA 95132 USA (e-mail: shan@c-cube.com).

C. I. Podilchuk is with Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974 USA (e-mail: chrisp@bell-labs.com).

Publisher Item Identifier S 1057-7149(01)09356-3.

In this paper, we propose an overall video coding system that employs dense motion fields. While we obtain the dense motion field using the well-known MRF formulation of [8], [6], we concentrate mainly on the problem of coding the motion and the subsequent DFD frame. Since a dense motion field takes up a much greater percentage of the overall rate, it is imperative to have a more efficient motion field encoder to show improved performance over traditional block-based coders. Likewise, a more accurate motion field should help us in modeling and encoding the motion-compensated residual frame. To our knowledge, there has been little or no work done in these contexts.

Since it is impractical to transmit one unique motion vector for every single pixel in an image, a variable-depth motion field is found from the original motion field (the motion encoding is thus *lossy*) [9]. This is followed by an efficient context-based entropy coding of the motion vector values [10], where the context is determined by both the magnitude and shape of the motion field in the spatio-temporal neighborhood. This allows us to obtain an accurate and detailed motion representation with a reasonable rate.

Our greatest gain in coding performance is in the modeling and coding of the DFD or residual frame, where we introduce two novel and distinct features. First, we take advantage of the dense motion information to model and encode the DFD frame. The dense motion field is a much more accurate representation of the true motion, and is available at the decoder in the forward motion-compensated framework. This information will help us predict where the significant DFD energy will be located (motion boundaries and uncovered regions). Although there has been recent work to better model and code the DFD data [11]–[14], all of them are within the block-based framework, and thus do not explicitly use the motion information as we do. Secondly, in the areas where the DFD energy is significant, we will go back and refine the motion field estimate itself with a new criterion to force the DFD values to be spatially smooth, thus making the DFD frame easier to encode although the overall MSE typically used for motion estimation actually increases. This idea of tailoring the motion estimation criteria to make the residual frame easier to encode has received little attention. In [15], the authors achieve limited success using an alternative block matching criterion which better models the subsequent DCT coding of the residual.

The outline of the paper is as follows. In Section II, the MRF-based motion estimation that we use is briefly reviewed. We especially highlight the part that will be modified and refined later in Section V as part of the DFD encoding process. Section III explains the motion encoding scheme, which consists of finding the variable-depth motion field and context-based entropy coding. The DFD encoding is detailed in Section IV, followed by the motion refinement stage in Section V. Simulation results comparing our performance against that of a block-based coder are given in Section VI. Finally, conclusions are drawn in Section VII.

II. Dense Motion Field Estimation

Although a number of dense motion field and optical flow estimation techniques are available [16], we use the multiscale motion estimation from [6] which was designed explicitly for coding purposes. The algorithm in [6] is a modified version of Konrad and Dubois' original motion estimator [8] based on Markov random fields (MRF). Namely, an extra scale dimension is added to the formulation such that the motion field is represented at multiple scales in a coarse-to-fine manner. The motion at a coarse level represents the *average* motion of the corresponding pixels at the finer level. The finest scale represents the dense (one motion vector for each pixel) level. The motion vectors are found by Markov Random Field (MRF) formulation on the four-dimensional lattice (spatio-temporal + scale) with local (in all four dimensions) smoothness constraints. Although we refer the reader to [6], [8], [16] for most of the details, it is worthwhile to note the likelihood function U_l to be

$$U_l(I_{t-1}|\mathbf{v}_t) = \frac{1}{C}\sum_{\mathbf{x}\in\mathcal{L}} |I_t(\mathbf{x}) - I_{t-1}(\mathbf{x}-\mathbf{v}_t(\mathbf{x}))| \quad (1)$$

where

- C normalizing constant;
- I_t current frame;
- $\mathbf{v}_t$ motion vector field;
- $\mathbf{x}$ pixel on the four-dimensional image grid;
- $\mathcal{L}$ set of all pixels.

This model represents the likelihood that the motion vectors minimize the displaced frame difference. In Section V, we will introduce a modification to this function that will improve the DFD coding efficiency.

III. Motion Field Encoding

The full multiscale motion field results in a motion vector at each pixel at all scales. Even though there is much redundancy due to static regions and the smoothness constraint imposed during the motion estimation, the rate to code such a full motion field is overwhelming when trying to incorporate it into a motion-compensated video coder. We thus implement a *lossy* encoding of the motion information to reduce the bit-rate. This involves two stages, the pruning of the motion field (this is the lossy part), followed by a lossless entropy coder. We use the generalized BFOS algorithm [17] for the pruning. Similar algorithms were used in [18] for variable-sized block matching and in [19] for hierarchical motion fields.

A. Pruning of Motion Field

Ideally, we would like to represent the motion on a coarse basis in static and smooth regions, and on a finer level in regions of complex motion and motion boundaries. We achieve this representation in a rate-distortion sense by observing the tree structure of the multiscale motion field and applying a pruning algorithm. The full tree is constructed from the full multiscale motion field, where each node of the tree represents a point on the multiscale grid. The tree is initially populated at each node by the rate and distortion values. The rate is estimated by the zeroth-order entropy, while the distortion at a node is taken to be the motion-compensated prediction error when using the motion vector associated with that node. The full tree is then pruned in a rate-distortion sense via the generalized BFOS algorithm until a desired rate or distortion constraint is met (see Fig. 1). Thus, areas with relatively static or smooth motion will be pruned to

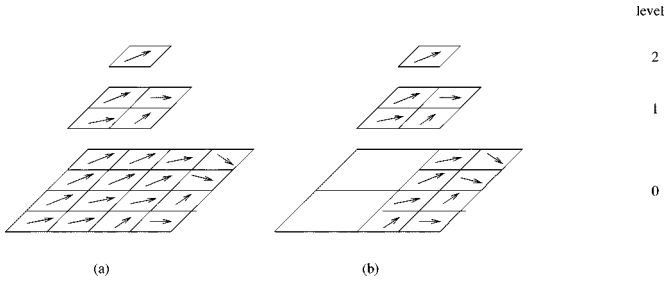


Fig. 1. Multiscale motion field: (a) full motion field and (b) pruned field.

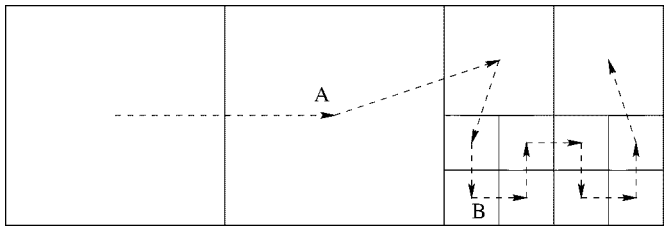


Fig. 2. "U-shaped scanning" of pruned motion field.

the coarsest scale, while fine motion vectors will be kept for regions that need them in order to improve the motion-compensated prediction. The generalized BFOS algorithm ensures that we achieve the best segmentation in a rate-distortion sense. This segmentation information is encoded as a quad-tree map and transmitted as part of the motion rate.

B. Encoding the Pruned Motion Field

Once the pruned motion field is found, we gain further compression by differentially encoding the motion vector components. Since the map information is transmitted, only the motion vectors corresponding to the leaf nodes of the pruned tree need to be encoded. The leaves are traversed in a "U-shaped scan order" to take full advantage of the spatial correlation (see Fig. 2), although a more elaborate traversal path could be used [20]. The prediction at a given leaf node is taken to be the median of the adjacent "causal" neighbors.

We also take advantage of the fact that the map information gives an indication of the spatial smoothness of the motion vectors. For instance, in Fig. 2, we expect the prediction of point A to be relatively good, while point B might be harder to predict from its neighbors since we can expect motion discontinuity. This is done by classifying the prediction into different classes based on the magnitude of the prediction as well as the geometry surrounding that point. The magnitude of the prediction, M, is compared against a threshold γ for significance, while the test for geometry is based on whether all of the adjacent neighbors from which the prediction is taken from are of the same scale. For example, the point B in Fig. 2 does *not* have all neighbors of the same scale (we call this mixed scale), while point C does (thus, uniform scale). This results in the following five classes:

1) all neighbor values are 0 and of uniform scale;
2) $M \leq \gamma$, uniform scale;
3) $M > \gamma$, uniform scale;
4) $M \leq \gamma$, mixed scale;
5) $M > \gamma$, mixed scale.

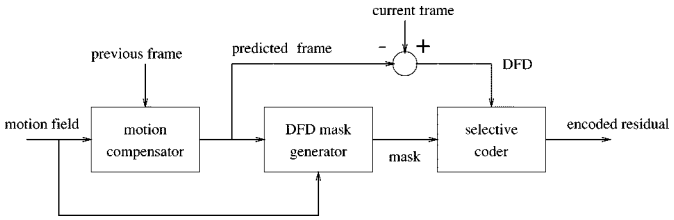


Fig. 3. DFD prediction and encoding strategy.

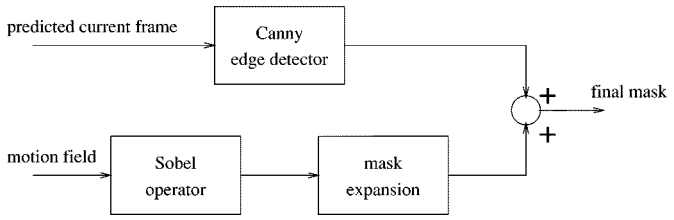


Fig. 4. DFD mask generator.

A separate adaptive arithmetic coder is then used to encode the motion vectors for each class. The main motivation for separating the data into the different classes is to model the conditional probability of symbols based on their surrounding neighborhood. A similar idea was used in encoding block-based motion vectors in [10], where only the magnitude of the motion in the causal neighborhood is considered in determining the context. With our multiscale approach, we observed significant improvements by additionally incorporating the geometrical information into the context classification, typically around 10% reduction in bit-rate in our experiments. Notice that no side information is needed (i.e., this is a backward scheme) since the context is uniquely determined from the "causal" neighbors and the transmitted map information which is available at the decoder.

IV. Encoding the DFD Frame

The basic idea of the DFD encoding scheme is to utilize the dense motion field in predicting where the DFD energy will be significant, and only coding the DFD values in these regions. The overall scheme is given in Fig. 3. First, the motion compensator builds a prediction of the current frame using the previous reconstructed frame and the dense motion field. We use the *encoded* motion vectors (using the algorithm developed in Section III) so that the exact process can be duplicated at the decoder. The DFD mask generator and the selective coder uses this information in finding and encoding the significant DFD energy.

A. DFD Mask Generator

The DFD mask generator uses both the motion information (the encoded motion field) and the spatial information (the predicted frame), as shown in Fig. 4. In other words, we expect the DFD energy to be significant around both motion-based discontinuities as well as intensity-based edges. The generator outputs a binary mask indicating where pixels with significant DFD energy are located.

The motion discontinuities are found initially by applying the Sobel operator to the horizontal and vertical components of

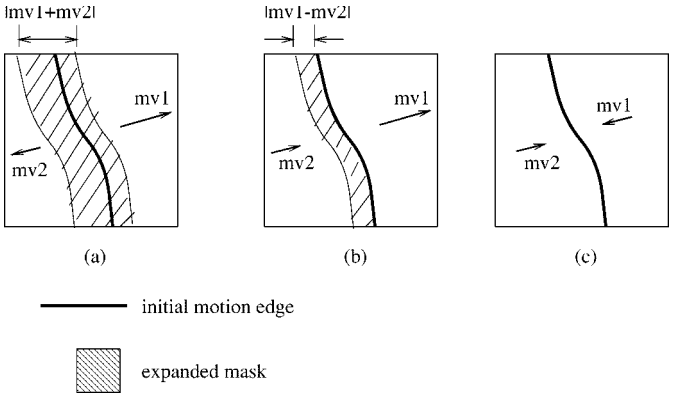


Fig. 5. Mask expansion: (a) Diverging motion, (b) concurrent motion, and (c) converging motion.

the motion field. We then possibly expand, or "thicken" these boundaries depending on the directions and magnitudes of the motion vectors at the edges. There are three types of motion discontinuities that we must differentiate, as illustrated in Fig. 5. In the case of diverging motion [Fig. 5(a)], the two objects at the boundaries are moving away from each other. Since we are predicting the current frame from the previous one, this is where we expect to find uncovered regions. Thus, the mask is grown in both directions normal to the boundary, with the thickness determined by the magnitude of the motion vector in the *opposite* direction. In the case when the two objects are moving in the same direction but at different magnitude [Fig. 5(b)], the mask is grown only in the direction opposite the larger of the two motion vectors. Finally, when two objects are converging, as in Fig. 5(c), we do not expand the mask any further.

Since by definition DFD energy represents failure in motion compensation, one would expect the DFD error to be significant only in the vicinity of motion discontinuities. However, experiments showed significant DFD error along many intensity-based edges even within regions with smooth or little motion. We compensated for this fact by finding the intensity-based edges using the current predicted frame (which is available at the decoder), and adding these edges to the DFD mask. Since the transmitted dense motion field is relatively accurate, we are able to obtain the edges from the motion-compensated predicted frame. The Canny edge detector [21] was used in detecting the edges.

B. Selective Coder

Using the DFD mask, the selective coder in Fig. 3 only encodes the DFD values within this mask. We use a backward context-based adaptive arithmetic encoder to fully utilize the motion and residual information in the spatio-temporal neighborhood [22]. Since only encoded information was used in obtaining the DFD mask, the decoder can generate the exact same mask without additional information. The decoder then reconstructs the DFD frame using this mask and the decoded DFD values by traversing through the pixels in the mask in a pre-determined order.

V. Refinement of Motion Field With DFD Smoothness Criterion

The DFD generator described in the previous section was successful in predicting most of the significant DFD energy. However, we found there was very little spatial correlation among neighboring DFD values in the mask, making it very difficult to efficiently encode them. We also observed that the motion vectors in these regions were quite unstable and highly fluctuating. This led us to believe that the motion vectors we found from our original formulation [8], [6] were not necessarily the best ones in terms of coding both the motion field and the DFD.

Based on these observations, we introduced a motion refinement stage that greatly increased the DFD encoding efficiency and the performance of the overall coding scheme. In short, instead of merely trying to minimize the DFD error in the motion estimation process, we added a constraint to make the DFD energy *spatially smooth*. In other words, we altered the motion estimation process so that the resulting DFD frame is easier to encode. Tailoring the motion estimate in order to obtain a smooth DFD which can be encoded efficiently as opposed to the more standard approach of generating a motion estimate that only minimizes DFD energy is key to the improved compression results we are able to get in this framework. Although the idea of tailoring the motion estimation such that the resulting DFD encoding is made easier seems natural, we found very little work with this line-of-thought [15].

More specifically, we add a second constraint to the likelihood function in the original Markov Random Field (MRF) formulation of [8], [6].

$$U_l(I_{t-1}|\mathbf{v}_t) = \sum_{\mathbf{x}\in L} |I_t(\mathbf{x}) - I_{t-1}(\mathbf{x}-\mathbf{v}_t(\mathbf{x}))| + \sum_{\mathbf{x}\in L}\sum_{\mathbf{y}\in N_t} |DFD(\mathbf{x}) - DFD(\mathbf{y})|. \quad (2)$$

Here, $\in N_t$ is a set of neighborhood pixels of $\mathbf{x}$. We used the first-order neighborhood in our experiments. The first term on the right-hand side of (2) is the conventional likelihood function from (1) in Section II that attempts to minimize the DFD for all pixels. However, we have added a second term in Equation (2) that simultaneously attempts to smooth the DFD values between neighboring pixels.

This can be best illustrated with an example in Fig. 6. With the conventional likelihood function of (1), the motion vector for the current pixel in Fig. 6 would be chosen such that the DFD for that pixel is minimized. We can clearly see that this would be suboptimal, and that a better candidate would be one that would make the DFD value similar to its neighbors. In some sense, we are deliberately attempting to *worsen* the motion-compensated prediction in terms of mean-square error, but the resulting residual frame will be much easier to encode.

In our overall scheme, this refinement was only carried out for the motion vectors within the DFD mask, making the increase in computation minimal. The pruning and coding of the refined motion field is performed as in Section III. As can be expected, the actual prediction error *increased* in terms of mean-square error. However, this was more than offset by the fact that the resulting DFD was much smoother, thus making the context-based

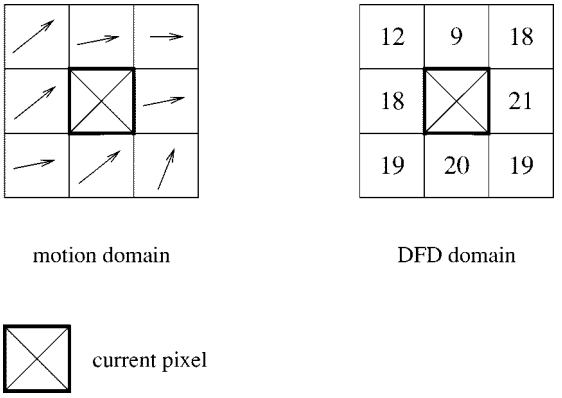


Fig. 6. Refinement of motion vectors.

coder much more effective. We also found that the rate needed to encode the refined motion vectors was similar to encoding the unrefined motion vectors.

VI. Experimental Results

Simulations were performed on two test sequences, *manya* and *Football*, both at SIF resolution (352 × 240) and 30 frames/sec. *Manya* is a typical head-and-shoulders sequence with limited movement, while *Football* has much more complex motion. We present in order results of the motion estimation, motion vector encoding, and the DFD coding, followed by overall video coding performance.

A. Motion Estimation and Encoding

The MRF-based motion estimation as outlined in Section II was carried out using deterministic relaxation [23] to make the complexity manageable. The trade-off between the likelihood function and the smoothness constraint was empirically found by experiment. The estimation between two frames took about 20 minutes on an UltraSparc-1 machine. Fig. 8 shows the horizontal and vertical components of the motion field between frame 3 and frame 4 of *Football* (Fig. 7). Here, the magnitudes were scaled for display purposes, with the darker shade representing negative motion, and the lighter color representing positive motion. We feel this method is better in conveying our motion field results more accurately, as opposed to the "arrows" illustrations commonly used which can only describe the motion in a general sense.

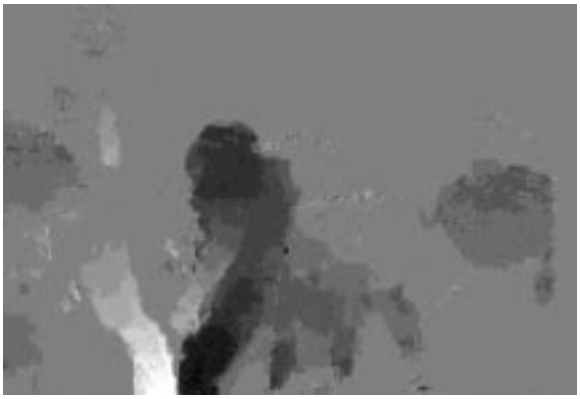
This motion field was coded (the pruning makes this a *lossy* motion vector encoder) using the procedure described in Section III, and the coded motion field is shown in Fig. 9. The resulting rate to encode this particular motion field, with the lossy step followed by the context-based arithmetic coder, was 5590 bits. The rate of the original uncoded motion vector field of Fig. 8 was 18 856 bits, using lossless differential arithmetic coding. For comparison, Fig. 10 shows the analogous motion field when using fixed-size (16 × 16) block matching. The standard lossless encoding (spatial differencing followed by entropy coding) of this motion field took 1790 bits.

We can see from Figs. 8 and 9 that the proposed motion vector encoder does a good job of retaining most of the details of the original dense motion field while at the same time reducing the

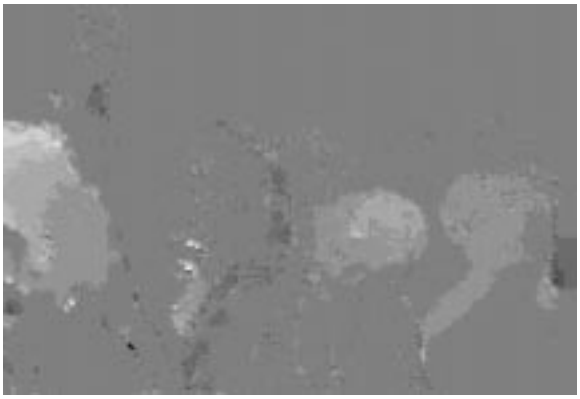
(a)

(b)

Fig. 7. Frames (a) 3 and (b) 4 of *Football*.

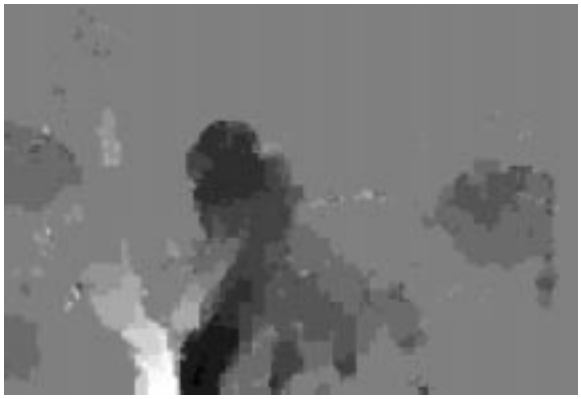


(a)

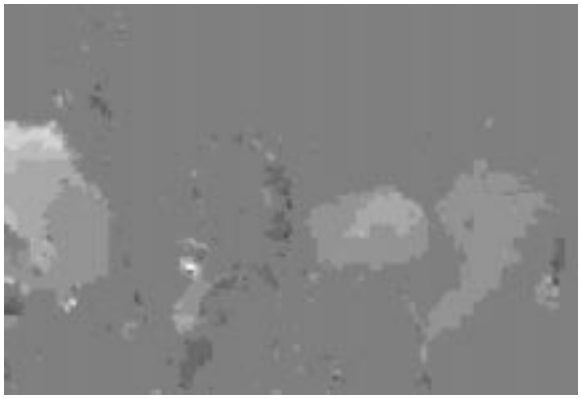


(b)

Fig. 8. Dense motion vector field (18856 bits).

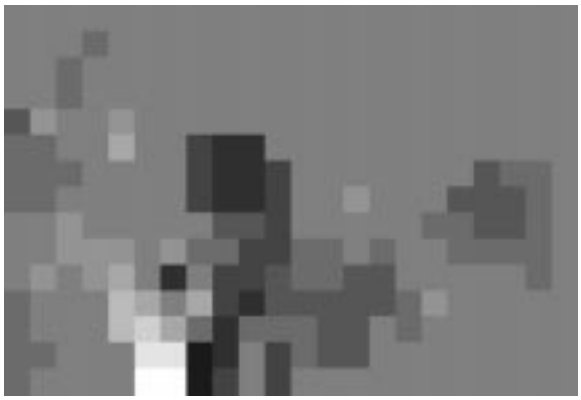


(a)

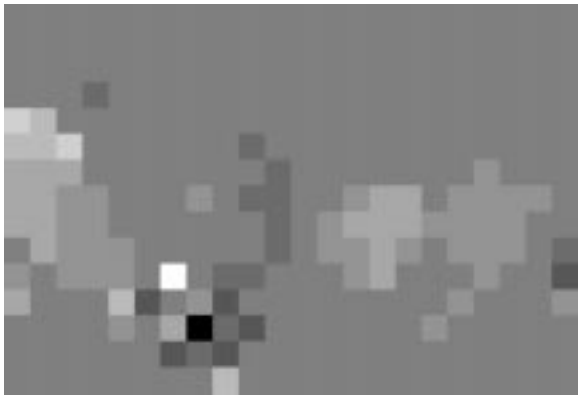


(b)

Fig. 9. Encoded dense motion vector field (5590 bits).

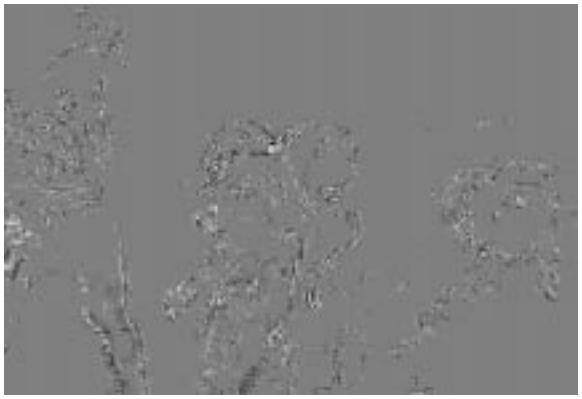


(a)

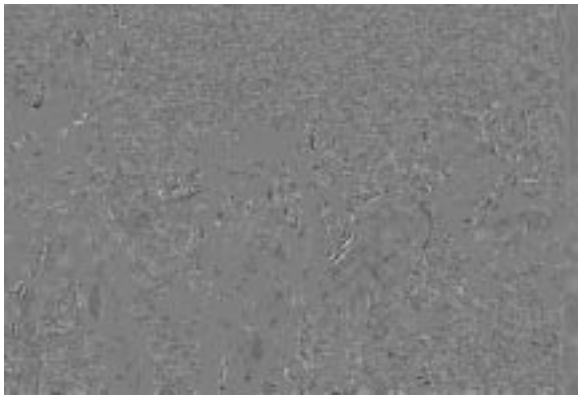


(b)

Fig. 10. Block-based motion vector field (1790 bits).



(a)



(b)

Fig. 11. (a) DFD values in significant mask and (b) remaining DFD values outside mask.

bit-rate by a factor of more than 3 to 1. Also, comparing Figs. 9 and 10, the dense motion field formulation followed by lossy coding results in motion vectors that are much more accurate and true to the real motion as compared with results of the block-based approach.

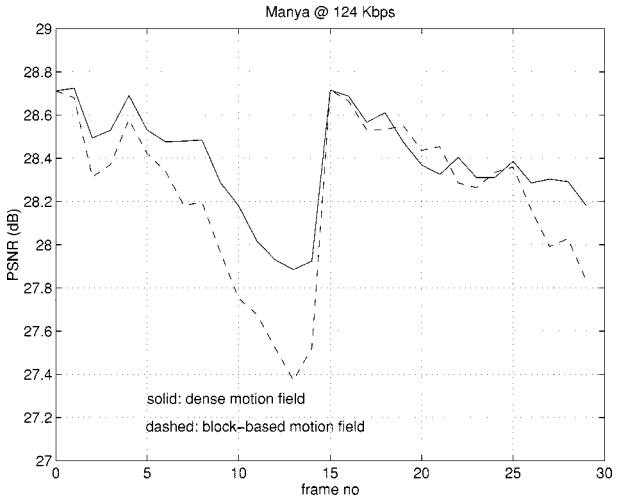
B. DFD Encoding

Using the encoded motion field and the predicted current frame, the DFD encoding stage amounts to finding the DFD mask and coding the DFD values inside the mask. Fig. 11 illustrates this for the same frame of *Football*. Typically, the significant DFD values occupied less that 8% of the entire frame while capturing over 95% of the total DFD energy. Finally, to increase the spatial correlation among the DFD values inside the mask, the motion field was refined as described in Section V and re-encoded. This lead to a 20–30% reduction in DFD coding rate while the increase in motion vector rate was minimal.

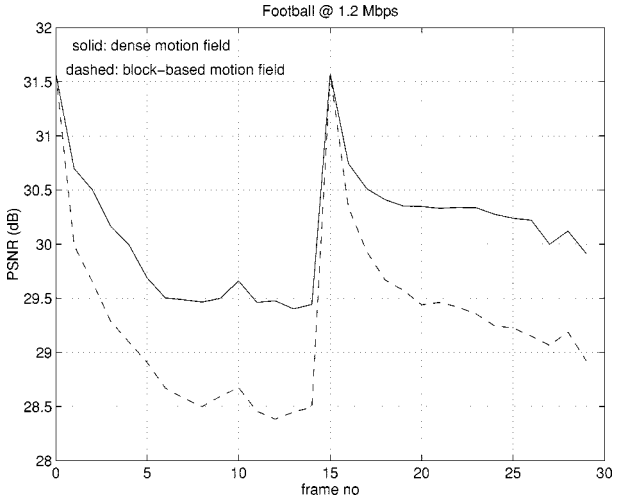
C. Overall Coding Results

We compared our overall coding results against a coder with motion vectors found by fixed-size block matching (half-pel accuracy) and encoded using standard entropy coding, followed by 8×8 DCT coding of the residual. For both cases, a group of pictures (GOP) of 15 was used, with an initial I frame (encoded by a wavelet coder) followed by 14 P frames.

Fig. 12 shows PSNR plot comparing the two methods, while Tables I and II summarize the average PSNR and the relative



(a)



(b)

Fig. 12. PSNR comparison.

TABLE I
PSNR RESULTS FOR *FOOTBALL* AT 1.2 Mbps

	MV %	PSNR(dB)
Block-based	14 %	31.2
Proposed	40 %	34.8

TABLE II
PSNR RESULTS FOR *MANYA* AT 124 kbps

	MV %	PSNR(dB)
Block-based	11 %	27.9
Proposed	25 %	29.2

motion vector rates. We can see the superiority of our proposed video coder employing the dense motion field, especially for *Football*, which has relatively large and complex motion. The advantage is more dramatic as the number of consecutive P frames grows, where the dense motion field continues to provide accurate prediction. Furthermore, note that the motion vector information takes up a much larger percentage of the overall rate (denoted by MV% in Tables I and II), especially for *Football*.

VII. CONCLUSIONS

We have introduced a forward motion-compensated video coding scheme employing dense motion fields as an alternative to the popular block-based methods. The high-quality dense motion field is found by existing techniques, while we concentrate on the coding of the motion information and the motion-compensated residual. The motion field is encoded in a lossy fashion such that the fine motion information is retained only where it is beneficial in the rate-distortion sense. The motion information is then used to improve the residual frame coding by predicting where the significant DFD energy will be located. Furthermore, the motion field itself is refined in these regions such that the DFD is smoother and thus easier to encode. We feel this last step is an important ingredient in the overall framework and can be regarded as a step toward bridging the motion estimation process and the DFD coding process.

REFERENCES

[1] M. H. Chan, Y. B. Yu, and A. G. Constantinides, "Variable size block matching motion compensation with applications to video coding," *Proc. Inst. Elect. Eng.*, vol. 137, pp. 205–212, Aug. 1990.
[2] H. Musmann, M. Hotter, and J. Ostermann, "Object-oriented analysis-synthesis coding of moving images," *Signal Process.: Image Commun.*, vol. 1, pp. 117–138, Oct. 1989.
[3] C. Stiller, "Object-oriented video coding employing dense motion fields," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, vol. V, Adelaide, Australia, 1994, pp. 273–276.
[4] P. Moulin, R. Krishnamurthy, and J. W. Woods, "Multiscale modeling and estimation of motion fields for video coding," *IEEE Trans. Image Processing*, vol. 6, pp. 1606–1620, Dec. 1997.
[5] A. Deever and S. Hemami, "Dense motion field reduction for motion estimation," in *Proc. Asilomar Conf. Signals, Systems, Computers*, Pacific Grove, CA, 1998.
[6] S. Servetto and C. Podilchuk, "Stochastic modeling and entropy constrained estimation of motion from image sequences," in *Proc. IEEE Int. Conf. Image Processing*, vol. 3, Chicago, 1998, pp. 591–595.
[7] B. K. P. Horn and B. Schunk, "Determining optical flow," *Artif. Intell.*, vol. 17, pp. 185–203, 1981.
[8] J. Konrad and E. Dubois, "Bayesian estimation of motion vector fields," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 910–927, Sept. 1992.
[9] S. Han and C. Podilchuk, "Efficient encoding of dense motion fields for motion-compensated video compression," in *Proc. IEEE Int. Conf. Image Processing*, Kobe, Japan, 1999.
[10] W. Jiang and A. Ortega, "Forward/backward adaptive context selection with applications to motion vector field encoding," in *Proc. IEEE Int. Conf. Image Processing*, Santa Barbara, CA, 1997.
[11] K. Ratakonda, S. Yoon, and N. Ahuja, "Coding the displaced frame difference for video compression," in *Proc. IEEE Int. Conf. Image Processing*, vol. 1, Santa Barbara, CA, 1997, pp. 353–356.
[12] B. Tao and M. Orchard, "Prediction of second-order statistics in motion-compensated video coding," in *Proc. IEEE Int. Conf. Image Processing*, Chicago, IL, 1998.
[13] O. Al-Shaykh, E. Miloslavsky, T. Nomura, R. Neff, and A. Zakhor, "Video compression using matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 172–186, Feb. 1999.
[14] Y. Wang and M. Orchard, "Coding of motion compensation residuals using edge information," in *Proc. IEEE Int. Conf. Image Processing*, Kobe, Japan, 1999.
[15] A. Fuldseth and T. Ramstad, "A new error criterion for block based motion estimation," in *Proc. IEEE Int. Conf. Image Processing*, Washington, DC, 1995, pp. 188–191.
[16] A. M. Tekalp, *Digital Video Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1995.
[17] E. Riskin, "Optimal bit allocation via the generalized BFOS algorithm," *IEEE Trans. Inform. Theory*, vol. 37, pp. 400–402, Mar. 1991.
[18] S. Choi and J. Woods, "Motion-compensated 3-D subband coding of video," *IEEE Trans. Image Processing*, vol. 8, pp. 155–167, Feb. 1999.
[19] H. Bi and W. Chan, "Rate-distortion optimization of hierarchical displacement fields," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 18–24, Feb. 1998.

[20] G. Schuster and A. Katsaggelos, "An optimal quadtree-based motion estimation and motion-compensated interpolation scheme for video compression," *IEEE Trans. Image Processing*, vol. 7, pp. 1505–1523, Nov. 1998.
[21] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 679–698, 1986.
[22] W. Jiang and A. Ortega, "Backward context adaptive interframe coding," in *Proc. SPIE Conf. Visual Communications Image Processing*, San Jose, CA, 1998.
[23] J. Besag, "On the statistical analysis of dirty pictures," *J. R. Statist. Soc.*, vol. 48, pp. 259–279, 1986.

Soo-Chul Han was born in Seoul, Korea, and received the B.S. degree in electronic enginerring from Yonsei University, Seoul, in 1990. He received the M.S. degree in electrical engineering, the M.S. degree in mathematics, and the Ph.D. degree in electrical engineering in 1992, 1996, and 1997, respectively, all from Rensselaer Polytechnic Institute, Troy, NY.

He served in the Korean Army in 1993–1994. From 1998 to 2000, he was a Member of Technical Staff, Visual Communications Research Department, Bell Labs, Lucent Technologies, Murray Hill, NJ. He is currently an R&D Engineer with C-Cube Microsystems, Milpitas, CA. His research interests include signal processing, image/video communications, and its applications to software and hardware systems.

Christine I. Podilchuk (S'83–M'88) received the B.S., M.S., and Ph.D. degrees in electrical engineering from Rutgers University, New Brunswick, NJ, in 1984, 1986, and 1988, respectively.

Currently, she is with the Visual Communications Research Department, Bell Labs, Lucent Technologies, Murray Hill, NJ. Her research interests are in the general area of image processing and computer vision. Her work includes video compression, source/channel coding for wireless networks, digital watermarking for security applications, and image recognition.

Dr. Podilchuk has served as Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING and is a Member of the IEEE Image and Multidimensional Signal Processing Technical Committee.