Fast Overlapped Block Motion Compensation with Checkerboard Block Partitioning

Tien-ying Kuo and C.-C. Jay Kuo

***Abstract*—The overlapped block motion compensation (OBMC) scheme provides an effective way to reduce blocking artifacts in video coding. However, in comparison with the traditional block motion compensation (BMC), its complexity of motion estimation increases significantly due to the interdependency of motion vectors. In this work, a new fast motion estimation algorithm is proposed for OBMC based on the checkerboard block partitioning and grouping in encoding. This algorithm, called the grouped OBMC (GOBMC), effectively decorrelates dependency among OBMC motion vectors and facilitates the search process. In comparison with traditional iterative OBMC motion search, GOBMC performs motion estimation only once per macroblock at the encoder, and reaches a local optimal solution with degradation of 0.05–0.1 dB. At the decoding end, we also propose a fast OBMC reconstruction scheme which reduces the complexity of multiplication to 38% of that of traditional OBMC reconstruction while preserving the same visual quality as obtained via BMC search with OBMC reconstruction.**

***Index Terms*—H.263, motion estimation, overlapped block motion compensation (OBMC), video coding.**

I. Introduction

THE MAIN shortcoming of block motion compensation (BMC) is the blocking artifact observed in the reconstructed image frame when motion vectors in the neighborhood are significantly different and/or the coding bit rate becomes very low. The overlapped block motion compensation (OBMC) scheme [1]–[3], which has been adopted in the advanced mode of the H.263 recommendation, proposes the use of overlapped blocks to reduce the blocking artifact. The efficiency of pixel prediction is also increased by employing multiple pixel values in OBMC. That is, OBMC predicts the pixel value using more than one motion vector, which come from the target macroblock as well as neighboring macroblocks. The major problem associated with OBMC is the increase of the computational complexity in estimating motion vectors at the encoder end and in reconstructing the image frame at the decoder end.

Motion search for OBMC is difficult since each OBMC motion vector affects pixel values in more than one macroblock. In other words, motion vectors are interdependent. The search of optimal motion vectors for OBMC is a noncausal problem with no closed-form solution. Watanabe and Singhal [1] used the classical BMC motion search, and applied OBMC only in reconstruction. This scheme improves the PSNR value at the expense of a higher computational complexity required by frame reconstruction. In comparison with OBMC which considers interdependence between motion vectors, the decoupled motion search with BMC is only suboptimal. Furthermore, boundaries of reconstructed blocks may be distorted in this scheme since all pixels in each block are equally weighted in BMC motion search while pixel values are weighted differently in OBMC reconstruction.

In an effort to get more accurate OBMC motion vectors, Ohta and Nogaki [2] proposed a cost function of weighted sum of absolute difference (SAD) for motion search. The procedure can be simply described below. The intensity difference of every pixel in an enlarged target block (i.e., the overlapped block) and the corresponding search region is calculated at each search displacement. The difference of the enlarged block is then multiplied by a window support function so that a higher weighting is assigned to a pixel near the center of the enlarged block. Finally, the SAD value of this weighted enlarged block is computed, and the motion candidate which gives the minimum cost function (i.e., the one with the lowest weighted SAD value) is chosen as the target motion vector. The main drawback of this method is that it requires one OBMC window multiplication at each search position, thus resulting in a very high computational complexity. Besides, this method only takes the target motion into account. That is, the cost function of motion search is only the function of the target motion, and no neighboring motion is involved. Thus, the interdependency of OBMC motion vectors is ignored during the search process.

In order to find a better set of OBMC motion vectors, Orchard and Sullivan (ORCH) [3] proposed an iterative OBMC motion search algorithm to solve this problem. They initialized block motion vectors with a certain motion field, e.g., the zero motion field, and then iteratively refined these vectors block by block with the window support function of OBMC, conditioned on neighboring motion vectors, until the vector field converges. Since the cost function of the motion search is not convex, the solution may not give the optimal OBMC motion field. Nevertheless, the iterative solution usually converges to a good local minimum. The major shortcoming of this method is the extremely high computational complexity required by iteration.

Rajagopalan *et al.* [5] proposed a two-pass algorithm to compute the OBMC motion field. Basically, an exhaustive BMC motion search is performed for every macroblock in the first pass. The second pass searches OBMC motion vectors based on an important partial set of BMC vectors obtained

Manuscript received June 20, 1997; revised July 1, 1998. This work was supported by Rockwell and the Integrated Media Systems Center, a National Science Foundation Engineering Research Center, with additional support from the Annenberg Center for Communication at the University of Southern California. This paper was recommended by Associate Editor W. Li.

The authors are with the Integrated Media Systems Center and the Department of Electrical Engineering—Systems, University of Southern California, Los Angeles, CA 90089-2564 USA.

Publisher Item Identifier S 1051-8215(98)07874-4.

in the first pass. Since the motion search procedure for each block could be more than once, the two-pass motion search algorithm is still expensive.

In the decoder, the procedure of OBMC frame reconstruction is not as simple as classical BMC. Additional multiplications are required for pixel value weighting so that the decoder's complexity increases. Sullivan and Orchard [4] attempted to reduce the OBMC decoder complexity by considering a new hardware design.

A new approach to OBMC search complexity reduction at the encoder end based on the checkerboard block grouping is investigated in this work. It is called the grouped OBMC (GOBMC). Only one motion estimation operation is required for each macroblock in GOBMC at encoding since the obtained motion vector set nearly reaches a local optimal solution at the first iteration. The encoding complexity is significantly reduced. The distortion measures, both in terms of PSNR and visual quality, remain about the same as those obtained from the iterative OBMC motion search (ORCH). In comparison with the work of Rajagopalan *et al.* [5], the OBMC motion search is performed only once for each macroblock in our algorithm. It is important to emphasize that, with our new approach, the OBMC motion search problem follows the same framework as BMC motion search as given in [6]. Thus, fast algorithms for BMC motion search described in [6] can be easily extended to the OBMC case.

At the decoding end, we also propose a fast OBMC reconstruction scheme which reduces the complexity of multiplication to 38% of traditional OBMC reconstruction while preserving the same visual quality as obtained via BMC search with OBMC reconstruction. No overhead bit is required by the decoder with this new scheme.

The paper is organized as follows. In Section II, we briefly review the OBMC concept and technical barriers. Then, the proposed GOBMC algorithm is described in Section III. Two GOBMC reconstruction schemes are examined in Section IV. Experimental results are provided in Section V. Concluding remarks are given in Section VI.

II. Brief Review of OBMC

The OBMC scheme is briefly reviewed in this section. As shown in Fig. 1, there are nine numbered squares, each of which represents a 16 × 16 macroblock. Let block 5 be the current (or target) block. The other eight blocks are its neighboring blocks. In BMC image reconstruction, each macroblock is associated with one motion vector, and pixel prediction is obtained by copying and pasting the 16 × 16 block from the previous frame offset by the motion vector. No block overlapping occurs in BMC during the copy-and-paste procedure. For the OBMC case, we use enlarged blocks to copy and paste. The enlarged block is represented by the dotted line in Fig. 1. The enlarged block, called the domain of window support, is of size 32 × 32. For block 5, its region is covered partially by the window support functions of the eight neighboring blocks and fully by its own window support. Therefore, there are nine motion vectors involved in predicting values in block 5. They are called the "motion set" of the target

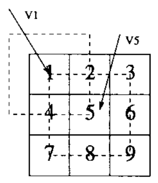

Fig. 1. Illustration of overlapped block motion compensation.

block. To get the correct pixel prediction in the overlapped part, we need to multiply the copy-and-paste block by the window support function, and provide the correct weighting after overlapping. OBMC can be viewed as a linear estimator of pixel intensity. It estimates a pixel value by using the weighted sum of pixel intensities obtained from the motion vector of the target block as well as those from neighboring blocks.

To reduce the computational cost, the following bilinear weighting window of size 32 × 32 is adopted in our work:

$$\mathcal{W}(x, y) = \mathcal{W}_x \cdot \mathcal{W}_y, \qquad \text{where}$$
$$\mathcal{W}_x = \begin{cases} \frac{1}{16}(x + \frac{1}{2}), & \text{for } x = 0, \cdots, 15 \\ \mathcal{W}_{31-x}, & \text{for } x = 16, \cdots, 31 \end{cases}$$

as shown in Fig. 3(d). Even though other window support functions can also be used, we observe that the difference in the PSNR performance is not sensitive to the choice of different window shapes.

For a given target block, there are nine blocks involved in the OBMC process, including the target block and its eight surrounding blocks. To set a mathematical framework of OBMC motion search, let us adopt the following notation. The nine blocks involved in OBMC are indexed by variable i, $1 \leq i \leq 9$, as shown in Fig. 1. The OBMC linear estimator for a predicted pixel can be written as

$$\overline{I}(x) = \sum_{i=1}^{9} [W_i(x_b) \cdot \hat{I}(x - u_i)]$$

where x is the absolute pixel position with respect to the frame coordinates, x_b is the pixel position relative to the coordinates of the current block within a block, i is the macroblock number, W_i and u_i are, respectively, the weighting coefficient and the motion vector of block i, $\overline{I}$ is the predicted OBMC pixel intensity, and $\hat{I}$ is the previous reconstructed pixel intensity. Note that x and x_b refer to the same location, but with a different reference point. W_i is of size 16 × 16, which is derived from 32 × 32 window support $\mathcal{W}$.

One barrier of OBMC motion search results from noncausality in the search pattern. The noncausality property can be explained from the fact that (1) is a function of motion variables u_1–u_9. Motion vectors of OBMC are interdependent because each pixel is affected by more than one motion vector. Existing OBMC work was described in Section I. In Section III, we propose a new fast motion search algorithm for OBMC based on the concept of checkerboard block partitioning.

III. Fast OBMC Motion Search with Checkerboard Partitioning

To develop the fast OBMC search algorithm, we have to solve the motion interdependency problem first. Two assumptions are made in this work to decorrelate the dependency of motion vectors. The first assumption is that the motion vector of the target block is primarily correlated with those of its eight neighboring blocks. Since motion vectors of neighboring blocks are also correlated with those of their own neighboring blocks, the correlation should propagate through every block in the whole frame in principle. This propagation property complicates the motion search problem. To simplify the problem, we assume that the target motion is only correlated with the motion vectors within a certain range. That is, motion vectors associated with blocks separated by one block apart are uncorrelated with each other in the derivation of fast OBMC motion search.

The second assumption is to let unknown motion vectors of neighboring blocks take the same motion value as that of the target block. During a noncausal motion search process, we may not know motion vectors of neighboring blocks with respect to a desired target block. By taking the strong spatial correlation property observed among the motion vectors of neighboring blocks and that of the target block, we approximate unknown neighboring motion vectors with the motion value of the target block. Therefore, if the motion vector of the target block is given, all eight neighboring motion vectors are assumed to be available to perform OBMC motion search.

Without loss of generality, we consider the QCIF sequence, which is of size 144 × 176 with the macroblock size 16 × 16, as an example. There are totally 9 × 11 macroblocks inside one image frame. To solve the motion interdependency problem, we divide macroblocks in an image frame into three groups with the checkerboard pattern. The three groups are labeled by 1, 2, and 3 as shown in Fig. 2. In each frame of QCIF video, we have 30 macroblocks of Group 1 (G1), 20 macroblocks of Group 2 (G2), and 49 macroblocks of Group 3 (G3). This checkerboard pattern partitioning is closely related to the first assumption, i.e., motion vectors associated with blocks separated by one block distance are uncorrelated with each other. It is clear that macroblocks of Group 1 are separated by one block, thus satisfying the first assumption. To perform OBMC motion search, we search all motion vectors for Group 1 (G1) macroblocks first under assumption 1. Then, motion vectors for Group 2 (G2) macroblocks are searched based on G1 motion vectors obtained earlier. Finally, motion vectors of Group 3 (G3) are determined based on surrounding G1 and G2 motion vectors available after the above two steps.

To derive the new search algorithm, we use the block numbering shown in Fig. 1, where block 5 is the target block. The local cost function for the OBMC motion search can be written as

$$J = \min_{u_5} \left(\sum_{x \in \text{ target block } (i=5)} |I(x) - \overline{I}(x)| \right) \tag{2}$$

where I is the desired target pixel intensity, $\overline{I}$ is the OBMC predicted pixel intensity given by (1), and variable u_5 is the unknown motion vector of the target block which is embedded in $\overline{I}$. By using (1), the cost function can be expressed as

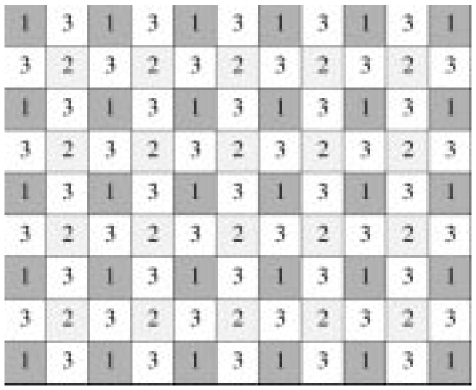

Fig. 2. Macroblocks with checkerboard partitioning, where blocks are classified into three different groups.

$$J = \min_{u_5} \left(\sum_{x \in \text{ target block}} \left| I(x) - \sum_{i=1}^{9} [W_i(x_b) \cdot \hat{I}(x - u_i)] \right| \right). \tag{3}$$

Our goal is to obtain the motion vector u_5 of the target block to minimize the above function. However, (3) is also a function of u_1–u_9. In other words, local cost functions are, in fact, coupled. It becomes difficult to solve the minimization problem with many unknown variables. We can apply assumption 2 to simplify the problem and remove the number of unknown neighboring vectors in the cost function. We know from Fig. 2 and the group search order that, for G1 macroblocks, all neighboring motion vectors (four G2 blocks and four G3 blocks) are unknown so that we simply set the neighboring vectors to the same value as the target vector G1. For G2, the neighboring four G1 motion vectors are available so that we only have to set the four G3's to the same value as the target vector G2. A similar rule can be also applied to G3 by setting four unknown surrounding G3 motion vectors the same value as the target motion vector G3.

To summarize, we can divide motion search into three steps, and determine motion vectors for G1, G2, and G3 blocks in each step sequentially. For G1 blocks, the size of the motion set involved in (3) is one, which is the unknown motion vector of the target block. For G2 blocks, the size of the motion set involved in (3) is five, consisting of one unknown motion vector of the target block and four known motion vectors of neighboring G1 blocks. For G3 blocks, the size of the motion set involved is also five, consisting of one unknown motion vector of the target block, two known motion vectors of neighboring G1 blocks, and two known motion vectors of neighboring G2 blocks. In terms of mathematics, let v_i denote the known vector for block i, we have

$$\text{Step 1: G1} \quad u_i = u_5, \qquad i = 1, \cdots, 9$$

$$\text{Step 2: G2} \quad u_i = \begin{cases} v_i, & i = 1, 3, 7, 9 \\ u_5, & i = 2, 4, 5, 6, 8 \end{cases}$$

$$\text{Step 3: G3} \quad u_i = \begin{cases} v_i, & i = 2, 4, 6, 8 \\ u_5, & i = 1, 3, 5, 7, 9. \end{cases}$$

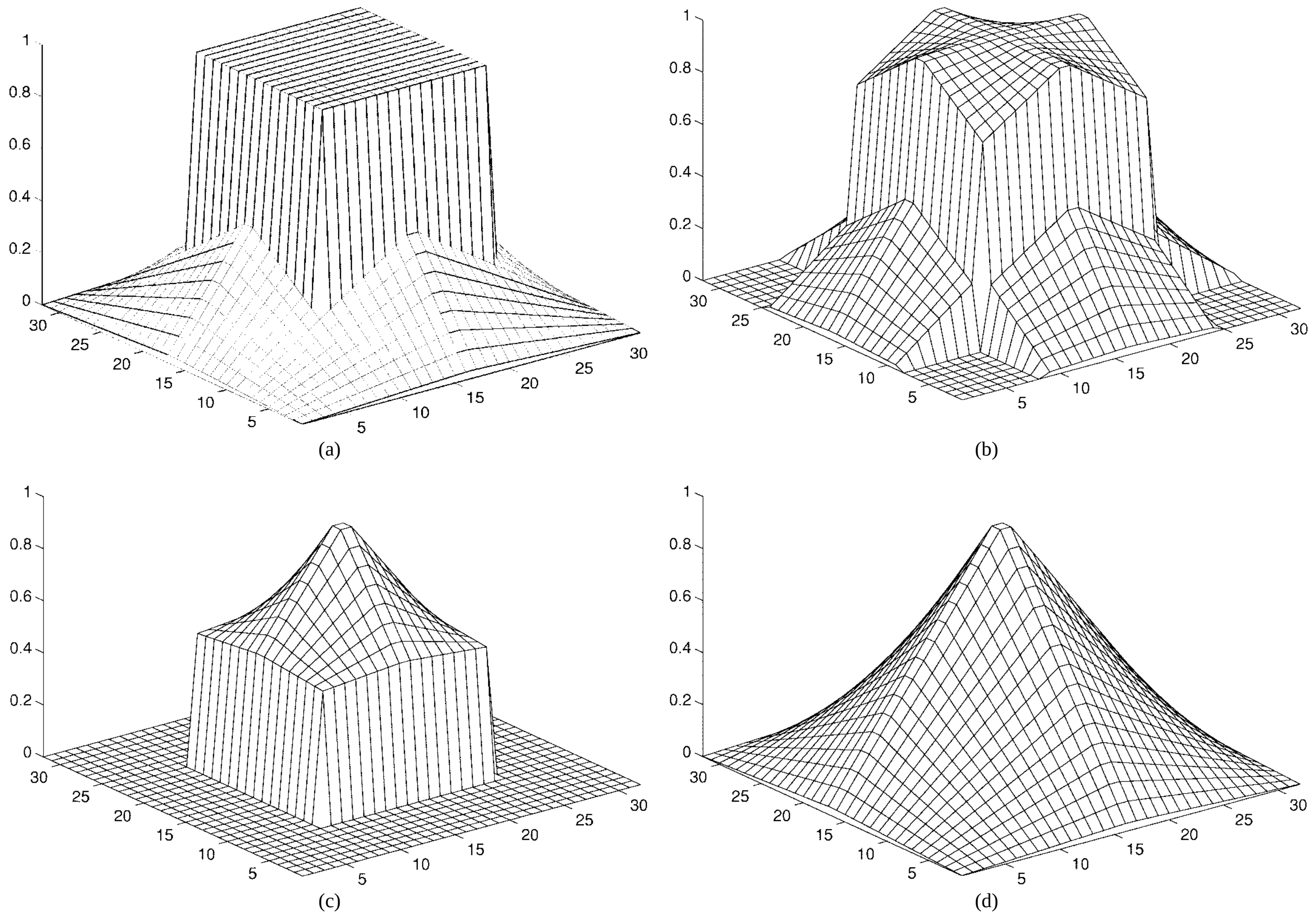

Fig. 3. Derived bilinear window for Groups (a) G1, (b) G2, and (c) G3. The window support (d) is the bilinear weighting window.

By combining the above formulas with (3), we have the following results for the three groups with the same form, i.e.,

$$J_{Gj} = \min_{u_5} \left(\sum_{x \in \text{target block}} |I_{Gj}(x) - W_{Gj}(x_b) \cdot \hat{I}(x - u_5)| \right) \tag{4}$$

$$= \min_{u_5} \left(\sum_{x \in \text{target block}} \left| \frac{I_{Gj}(x)}{W_{Gj}(x_b)} - \hat{I}(x - u_5) \right| \right), \quad \text{for } j = 1, 2, 3 \tag{5}$$

where

$$I_{G1}(x) = I(x) \tag{6}$$

$$I_{G2}(x) = I(x) - \sum_{i=1,3,7,9} [W_i(x_b) \cdot \hat{I}(x - v_i)] \tag{7}$$

$$I_{G3}(x) = I(x) - \sum_{i=2,4,6,8} [W_i(x_b) \cdot \hat{I}(x - v_i)] \tag{8}$$

$$W_{G1}(x) = 1 \tag{9}$$

$$W_{G2}(x) = \sum_{i=2,4,5,6,8} W_i(x_b) \tag{10}$$

$$W_{G3}(x) = \sum_{i=1,3,5,7,9} W_i(x_b). \tag{11}$$

In deriving the GOBMC cost function, we see that (4) requires the multiplication of W_{Gj} with every reconstructed block $\hat{I}$ for each search displacement. The resulting computational cost is high. This cost can be significantly reduced by dividing (4) with the nonzero coefficients W_{Gj} as given in (5). That is, the number of the window support multiplication is reduced to only once per motion vector search for all search displacements since the term $I_{Gj}(x)/W_{Gj}(x_b)$ is independent of the search position u_5, and has to be computed once only.

Coefficients W_{Gj} of the group window support function can be precalculated before the actual OBMC motion search. Their values are plotted as the central 16 $\times$ 16 matrix of the 32 $\times$ 32 window supports in Fig. 3(a)–(c) for the bilinear cases. As far as the term I_{Gj} is concerned, it is completely known since it only involves known neighboring motion vectors. Thus, in this minimization problem, the only unknown variable in (5) is variable u_5, which is the desired target motion vector. From (5), the cost function of GOBMC motion search looks similar to that of the BMC, except that the term I is replaced by I_{Gj} and an additional multiplication with $(1/W_{Gj})$ is needed. Therefore, we can apply BMC motion search methods such

TABLE I
PSNR PERFORMANCE COMPARISON WITH RESPECT TO SEVEN DIFFERENT METHODS WITH THE BILINEAR WINDOW

PSNR	BMC	WATA	ORCH	E-GOBMC-R1	E-GOBMC-R2	F-GOBMC-R1	F-GOBMC-R2
Miss America(150)	40.79	41.54	42.12	42.00	41.28	42.04	41.29
Suzie(150)	35.32	35.97	36.54	36.43	35.87	36.39	35.84
Salesman(448)	39.23	39.78	40.09	39.95	39.63	39.96	39.63
Carphone(382)	30.08	30.56	30.79	30.72	30.48	30.73	30.47
Claire(494)	41.91	42.75	43.15	43.08	42.34	43.08	42.30
Foreman(400)	30.77	31.26	31.66	31.55	31.19	31.57	31.18
Akiyo(300)	42.68	43.45	44.08	43.94	43.28	43.93	43.28

TABLE II
COMPARISON OF THE COMPUTATIONAL COMPLEXITY OF ALGORITHMS FOR THE SUZIE SEQUENCE AS INDICATED BY THE AVERAGE NUMBER OF BLOCK OPERATIONS PER MACROBLOCK, WHERE i IS THE NUMBER OF ITERATION

	Encoder		Decoder	
Method	Addition/Subtraction	Multiplication	Addition/Subtraction	Multiplication
BMC	782.6	0	0	0
WATA	782.6	0	8	9
ORCH	$790.6 \times i$	$9 \times i$	8	9
E-GOBMC-R1	785.4	3.5	8	9
E-GOBMC-R2	785.4	3.5	2.8	3.5
F-GOBMC-R1	151.7	3.5	8	9
F-GOBMC-R2	151.7	3.5	2.8	3.5

as the exhaustive search or various fast search algorithms to (5) to obtain OBMC motion vectors. Depending on whether exhaustive or fast BMC search algorithms are adopted, we can get exhaustive GOBMC (E-GOBMC) or fast GOBMC (F-GOBMC) search methods. In the implementation, we incorporate a fast BMC motion search algorithm proposed in [6] to speed up the motion search process and to further reduce the complexity of motion search. This fast BMC motion search utilizes the spatial and temporal correlations of motion field to predict the target motion vector based on motion vectors from neighboring blocks or previous frames and then perform the local refinement. If the refined motion vector is still not accurate, then full search is applied. Compared to the E-GOBMC search method, F-GOBMC motion search only requires around 20–35% of macroblocks for full search on the average.

IV. IMAGE RECONSTRUCTION: GOBMC-R1 AND GOBMC-R2

Regarding image reconstruction in encoding or decoding, we propose two new reconstruction algorithms called GOBMC-R1 and GOBMC-R2 based on the following observations. First, given BMC vectors, OBMC reconstruction can give a better PSNR performance than the BMC reconstruction even though motion vectors are optimized for the BMC case. Second, the window support with a larger motion set can give a better reconstructed image. In GOBMC-R1, we use the traditional OBMC reconstruction, which adopts a motion set of nine motion vectors and 32 $\times$ 32 window support with the bilinear window shape for every group.

In order to reduce the complexity of OBMC reconstruction in the decoder, we also consider an alternative in which we do not use all nine motion vectors for every macroblock, but simply apply those OBMC configurations of each group as in the encoder to the decoder. This is called algorithm GOBMC-R2. For each block in Group 1, no OBMC reconstruction is used so that no block multiplication is required. As for pixels in Groups 2 and 3 compared with GOBMC-R1, the block weighting multiplication is reduced into 5/9 and 5/9, respectively, because we can sum up the five out of nine W_i into W_{G2} and W_{G3} in advance as given in (10) and (11). For example, for each OBMC reconstruction of blocks in Group 2, we only have to perform the block multiplication of $(1/W_{G2})$ in (5) as well as the block multiplications of W_1, W_3, W_7, and W_9 in (7). The multiplication complexity of GOBMC-R2 reconstruction is reduced to about 38% of GOBMC-R1 (or OBMC) based on the following computation:

$$0 \cdot \frac{30}{99} + \frac{5}{9} \cdot \frac{20}{99} + \frac{5}{9} \cdot \frac{49}{99} \approx 0.38$$

where numbers 30, 20, and 49 are block counts in one image frame for each group, respectively, and 99 is the total block number within one image frame. Note that the above complexity reduction result is valid for any QCIF sequence.

V. EXPERIMENTAL RESULTS

Seven QCIF videoconferencing image sequences are used in the experiment to illustrate the performance of the proposed algorithms. The first column of Table I gives the test video title, and the number in parentheses is the total number of frames used to generate the average results. For each video, we perform a forward predictive coding with OBMC for all frames (called the P frames), except for the first frame which is encoded as the I frame, and compute the average PSNR value for decoded P frames. The experiments are conducted based on the Telenor TMN5 H.263 environment with a fixed quantization step of size 2. The size of the macroblock is 16×16. The motion search displacement is in the range from -16 to 16. All motion vectors are in integer-pixel precision. Bilinear window support is used.

We compare the performance of seven methods as shown in Tables I–II and Figs. 4–5. The first three methods are existing methods, while the last four are variations of the proposed

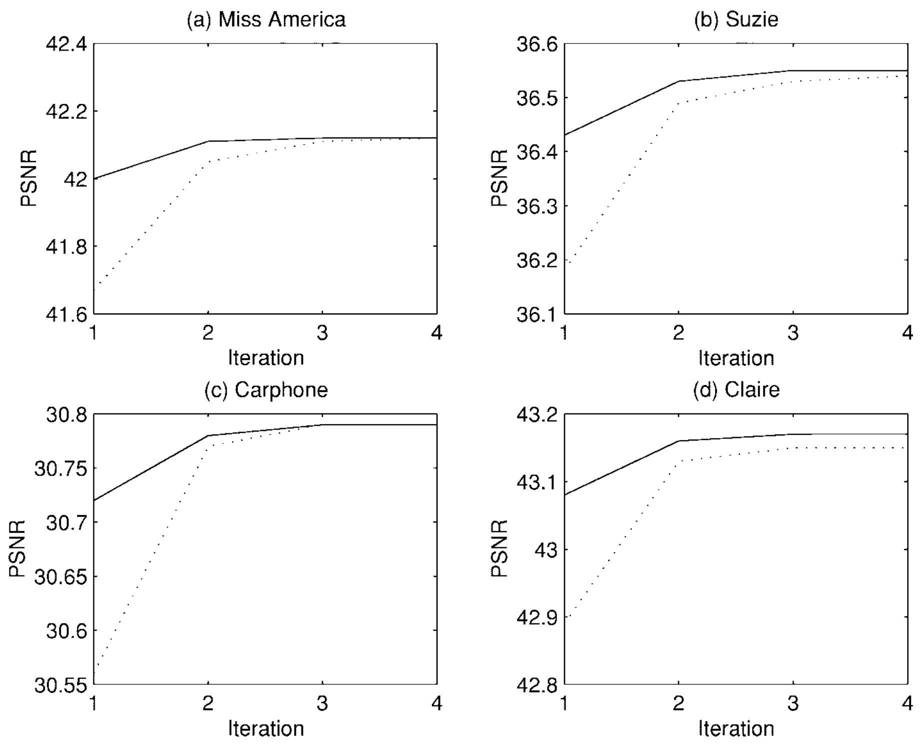

Fig. 4. Comparison of the average PSNR values as a function of the iteration number for iterative OBMC (ORCH) and E-GOBMC-R1 with the bilinear window. The ORCH and proposed E-GOBMC-R1 are denoted by a dotted line and a solid line, respectively.

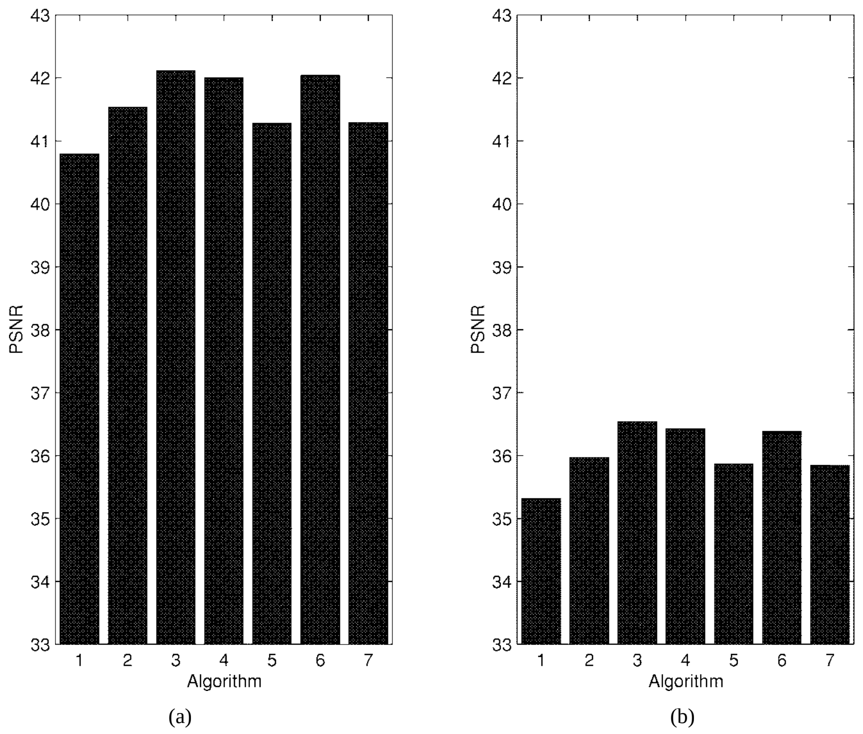

Fig. 5. PSNR performance comparison with respect to seven different methods with the bilinear window. The seven algorithms are: 1) BMC, 2) WATA, 3) ORCH, 4) E-GOBMC-R1, 5) E-GOBMC-R2, 6) F-GOBMC-R1, and 7) F-GOBMC-R2.

method. It is shown in Fig. 4 that our proposed algorithm can be used in a noniterative fashion. In Fig. 4, two methods are compared. The first method is ORCH [3], denoted by the dotted line. It sets all initial motion vectors to zero and performs the OBMC motion search at the first iteration, and then refines OBMC motion field iteratively in the remaining iterations. The second method E-GOBMC-R1 is denoted by the solid line, which uses the proposed exhaustive GOBMC scheme for motion search at the first iteration. Then, for the remaining iterations, it uses the same bilinear window support for all three groups as the ORCH method, while the motion search is performed in the grouping order. As shown in Fig. 4,

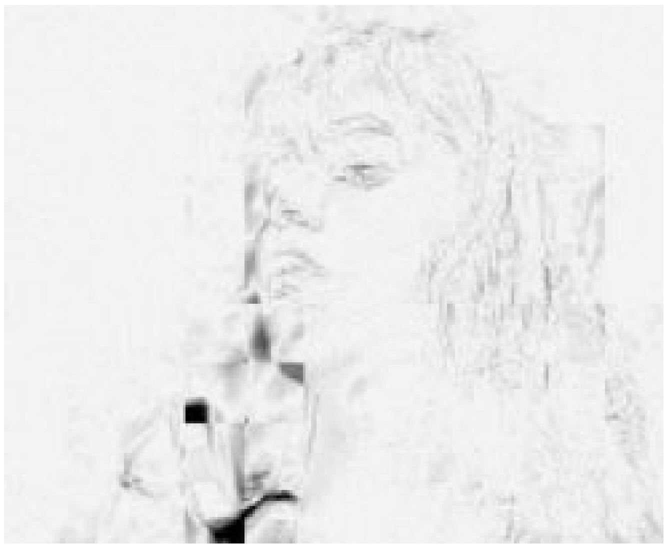
Fig. 6. Fifty-second DFD of block motion compensation (BMC) for the Suzie sequence with PSNR = 30.33 dB.

E-GOBMC-R1 converges much faster than ORCH in test video. In fact, GOBMC-R1 reaches nearly the local optimal solution at the end of the first iteration. This justifies the termination of GOBMC-R1 at the end of the first iteration. It is also worthwhile to point out that the difference in the converged PSNR values of ORCH and GOBMC-R1 is very small as given in the fourth iteration.

The average PSNR results of seven algorithms are summarized in Table I and Fig. 5. The WATA method [1] is good enough to improve the PSNR value, even without OBMC motion search. The PSNR value of E-GOBMC-R1 and F-GOBMC-R1 approaches that of iterative OBMC search (ORCH) [3] with 0.05–0.1 dB degradation. The PSNR value of GOBMC-R2 is not as good as GOBMC-R1, but it still preserves about the same PSNR value as the WATA method with a lower complexity in the decoder. If we use BMC motion vector for GOBMC-R2 reconstruction, the performance is not better than GOBMC-R2 reconstruction with GOBMC motion search. The complexity saving of GOBMC-R2 in the decoder without much PSNR degradation can be explained by the fact that GOBMC motion search provides a suitable motion field for GOBMC-R2 reconstruction. We also see from this table that GOBMC with fast search (F-GOBMC) leads to a set of good OBMC motion vectors which keep the same PSNR as GOBMC with exhaustive search (E-GOBMC) at a lower computational cost.

To show the visual effect and the removal of blocking artifacts, three displaced frame difference (DFD) images of Suzie are shown in Figs. 6–8, corresponding to the fifty-second frame difference of the original and reconstructed frames by using BMC, WATA, and E-GOBMC-R1, respectively. In these figures, a darker gray level is used to indicate a larger DFD value. As shown in Fig. 6 (BMC), one can see clearly the blocking artifacts and two regions with a large DFD value in the phone receiver and the collar of Suzie. Blocking artifacts in Fig. 6 (BMC) are removed in the reconstructed images of Figs. 7 (WATA) and 8 (E-GOBMC-R1), which are DFD images of OBMC with different search methods. However, the two large DFD regions in the phone and the collar still remain. The result of our proposed algorithm is shown in Fig. 8. Errors in the two high DFD regions are substantially reduced and spread out more uniformly to their neighborhood as the result of efficient GOBMC motion search and prediction. Other parts, e.g., the left eye of Suzie, are also improved in Fig. 8 in comparison with Figs. 6 and 7. Note that the predication error in Fig. 8 is lower than the ones in Figs. 6 and 7 because of the lowest average value of PSNR as indicated in the figure captions. In addition to the above examples, it is generally true that the proposed E-GOBMC-R1 is subjectively better since it has a lower and more uniform error distribution in comparison with the WATA method.

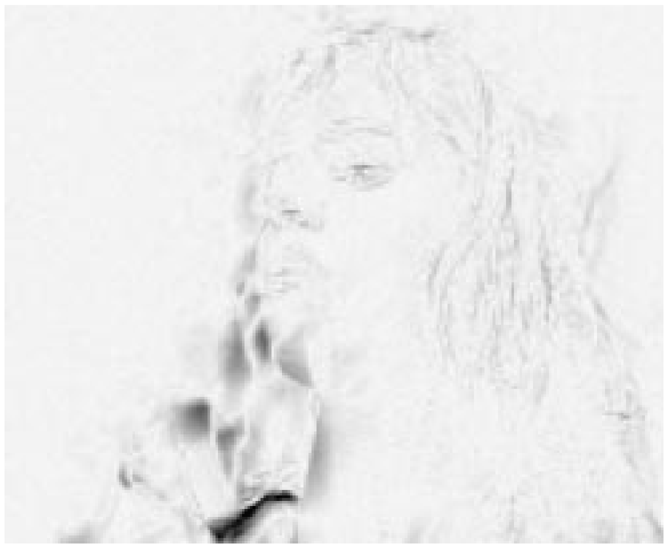
Fig. 7. Fifty-second DFD of BMC vectors with OBMC reconstruction for the Suzie sequence with the bilinear window and PSNR = 31.51 dB.

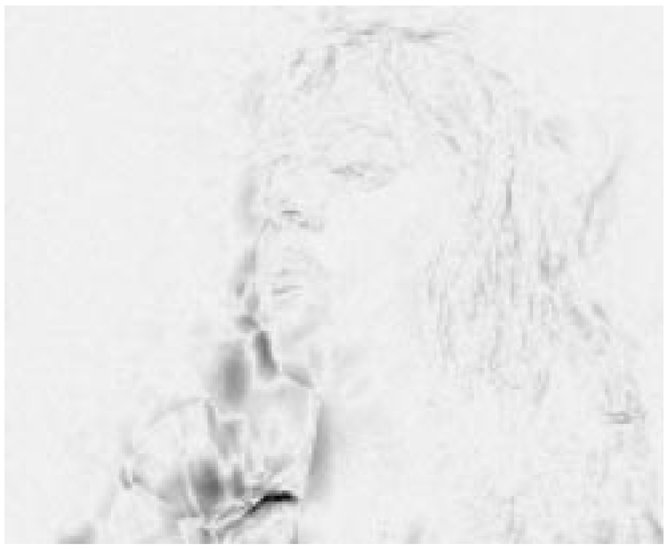
Fig. 8. Fifty-second DFD of the proposed noniterative exhaustive GOBMC-R1 for the Suzie sequence with the bilinear window and PSNR = 32.79 dB.

The complexity comparison based on the average result for the Suzie sequence is shown in Table II. For motion estimation in the encoder and frame reconstruction in the decoder, we can measure the complexity by the number of addition/subtraction and multiplication per macroblock. We count it as one block operation for a 16 $\times$ 16 matrix (macroblock size) adding or multiplying with a 16 $\times$ 16 matrix. For example, the $W_i \cdot \hat{I}(x - v_i)$ in (7) is counted as "one block operation" of multiplication for each i. As an example in F-GOBMC-R2 of Table II, each macroblock requires 151.7 block operations of addition/subtraction and 3.5 block operations of multiplication on the average in the encoder. We observe that F-GOBMC-R1 and F-GOBMC-R2 have the lowest complexity in the encoder. For the Suzie sequence, it takes only one-fifth of the exhaustive motion search method. Furthermore, F-GOBMC-R2 saves many computations in the decoder. Therefore, F-GOBMC-R1 and F-GOBMC-R2 are the best algorithms in term of complexity among the seven methods. One thing we have to point out is that E-GOBMC-R1 still has a lower complexity than iterative OBMC (ORCH) [3], even though the iteration number i of ORCH is set to one.

VI. Conclusions

In this work, we focused on the complexity reduction for OBMC in both motion vector search at the encoder end and image frame reconstruction at the decoder end. The proposed motion search scheme GOBMC-R1 determines a set of good motion vectors while giving almost the same PSNR performance as iterative OBMC search (with 0.05–0.1 dB degradation). Complexity reduction due to one motion search per macroblock and fast search in the encoder make the new algorithm GOBMC-R1 more attractive. By implementing GOBMC-R1 with fast search algorithms, the complexity can be reduced to 20%–35% of the exhaustive GOBMC search. In block motion search, instead of applying one block multiplication per motion search displacement, we only perform the block multiplication once per block motion search with GOBMC-R1. Furthermore, GOBMC-R2 has the same motion search complexity as GOBMC-R1 while with fewer block multiplications. The multiplication complexity of OBMC reconstruction can be reduced to 38% of that of GOBMC-R1 in the decoder by using GOBMC-R2. A good visual quality is still preserved by GOBMC-R2, which has a similar performance as that of BMC motion search with OBMC reconstruction.

References

[1] H. Watanabe and S. Singhal, "Windowed motion compensation," in *Proc. SPIE Visual Commun. Image Processing*, vol. 1605, Nov. 1991, pp. 582–589.

[2] S. Nogaki and M. Ohta, "An overlapped block motion compensation for high quality motion picture coding," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 1992, pp. 184–187.

[3] M. T. Orchard and G. J. Sullivan, "Overlapped block motion compensation: An estimation theoretic approach," *IEEE Trans. Image Processing*, vol. 3, pp. 693–699, Sept. 1994.

[4] G. J. Sullivan and M. T. Orchard, "Methods of reduced-complexity overlapped block motion compensation," in *Proc. IEEE Int. Conf. Image Processing*, Nov. 1994, pp. 957–961.

[5] R. Rajagopalan, E. Feig, and M. T. Orchard, "Motion optimization of ordered blocks for overlapped block motion compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 119–123, Apr. 1998.

[6] J. Chalidabhongse and C.-C. J. Kuo, "Fast motion vector estimation using multiresolution-spatio-temporal correlations," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 477–488, June 1997.