Multiple Blocks Matching Pursuit Update Algorithm for Low Bit Rate Video Coding

Jian-Liang Lin, Wen-Liang Hwang, and Soo-Chang Pei, *Fellow, IEEE*

***Abstract*—Matching pursuit (MP) is a greedy algorithm that decomposes a signal into a linear combination of bases within an overcomplete dictionary. The MP algorithm is usually only approximated due to its massive computational complexity. By approximating a residual in a subspace, we propose a multiple blocks MP update algorithm for video coding that achieves a faster and better MP approximation than traditional algorithms. As the subspace is dependent on a residual, our algorithm is content-dependent. We evaluate the coding performance, including the peak signal-to-noise ratio, runtime, and subjective quality, of our algorithm and compare it with traditional MP algorithms.**

***Index Terms*—Content based video coding, fast algorithm, matching pursuit (MP), video coding.**

I. Introduction

MATCHING PURSUIT (MP), which is a frame-based algorithm, is a promising method for low bit rate video coding [11], [15]. An MP-based codec yields a better PSNR and perceptual quality than a transform-based codec, and its decoder is simpler [12]. However, it cannot be used in applications that require real time bidirectional communications, because the encoder consumes a massive amount of computational time. An MP encoder does not obtain all the coefficients in one step, but iteratively finds the frame coefficient that has the largest absolute inner product value between a residual and all the bases. The inner product value and the base from which the value is obtained are called an atom. Many approaches have been proposed to simplify the complex encoding stage. One approach approximates the codewords of a dictionary with a linear combination of simpler codewords so that the computation is easier [3], [4], [13], [14], [16]. This approach can be further developed by combining the inner product calculation and the atom finding components [9].

Another method precalculates and stores all the inner products between bases so that the encoder can update the inner products with the precalculated values of the bases, instead of recalculating the inner products between a residual and the bases at each iteration [10]. This is an efficient way to decompose a one-dimensional signal. However, it is totally unsuitable for video coding, because there are too many inner products between the bases. In the popular Gabor dictionary used in MP video encoding, there are 400 codewords, each of which is at most 35×35 pixels. Consequently, the inner products between bases need at least a 30-Gb memory (assuming four bytes for a real value).

This difficulty prevents the MP algorithm achieving its best performance. The most popular approach for finding an atom is that proposed by Neff and Zakhor [11], whereby a residual frame is divided into blocks and, at each iteration, an atom is found within the block with the highest energy. This approach is modified in [1], which gives an energy weight to each block so that the more atoms chosen from a block, the smaller the energy weight of that block will be. Therefore, the block is less likely to be chosen in later iterations. The energy-weight approach reduces the likelihood that the majority of atoms will be selected from a few blocks, and improves the peak signal-to-noise (PSNR) performance of Neff and Zakhor's algorithm.

As the above algorithms find an atom from the largest (weighted) energy block, we call them one-block algorithms. These approaches are simple and efficient, but their coding performance may be unsatisfactory. Although the performance can be improved by finding an atom from more than one block, there is still the issue of the massive number of inner products between a residual and the bases in the blocks. To solve this problem, we approximate a residual in a subspace, spanned by a small number of bases within a few blocks. The bases and the blocks are selected according to the content of the residual, while the coding performance and efficiency are determined by the number of bases and the number of blocks. Simulations show that our algorithm achieves better subjective and objective performances and requires less runtime than one-block algorithms for various sequences at low bit rates.

The remainder of the paper is set out as follows. In Section II, we review the MP update algorithm. In Section III, we propose our approach. In Section IV, the performance evaluation and comparisons are given. Finally, in Section V, we present our conclusions.

Definition: (1) A codeword is an element in a dictionary. (2) A codeword centered in a position forms a base. (3) An atom is composed of a base and its corresponding inner product value.

II. MP Update Algorithm and Atom Extraction

MP is a frame-based algorithm that represents a signal by a succession of greedy steps [10]. At each iteration, the signal is projected onto the base that approximates the signal most efficiently. Let $R^0 f(\mathbf{x}) = f(\mathbf{x})$. Instead of recalculating the inner

Manuscript received May 3, 2004; revised October 28, 2005. This work was supported in part by the National Science Council, R.O.C., under contracts NSC 94-2213-E-002-072 and NSC 93-2752-E-002-006-PAE. This paper was recommended by O. Al-Shaykh.

J.-L. Lin and W.-L. Hwang are with the Institute of Information Science, Academia Sinica, Taipei 115, Taiwan, R.O.C. (e-mail: jian@iis.sinica.edu.tw)

S.-C. Pei is with the Institute of Communication Engineering, National Taiwan University, Taipei 10617, Taiwan, R.O.C. (e-mail: whwang@iis.sinica.edu.tw).

Digital Object Identifier 10.1109/TCSVT.2006.869969

products at each iteration, Mallat and Zhang [10] developed the MP update algorithm. At the kth iteration, let

$$g_{\gamma_k} = \arg\max_{\gamma} \left|\langle R^k f, g_\gamma\rangle\right|$$

be the base of the largest absolute inner product value. The new residual signal $R^{k+1}f$ is

$$R^{k+1}f = R^k f - \langle R^k f, g_{\gamma_k}\rangle g_{\gamma_k}.$$

The inner products between $R^{k+1}f$ and the bases $\{g_\gamma\}$ can be represented by

$$\langle R^{k+1}f, g_\gamma\rangle = \langle R^k f, g_\gamma\rangle - \langle R^k f, g_{\gamma_k}\rangle\langle g_{\gamma_k}, g_\gamma\rangle. \quad (1)$$

Because $\langle R^k f, g_\gamma\rangle$ and $\langle R^k f, g_{\gamma_k}\rangle$ were calculated in the previous iteration, and if $\langle g_{\gamma_k}, g_\gamma\rangle$ is precalculated, this update operation only needs one addition and one multiplication. However, the algorithm needs a huge amount of space to store all nonzero $\langle g_{\gamma_k}, g_\gamma\rangle$ in an image and is only practical for one-dimensional signal decomposition. Thus, the MP update algorithm cannot be used in video encoding. Instead, the proposed approach in [1] and [11] divides a residual into blocks and, at each iteration, the MP algorithm is applied to the block with the highest energy. This approach is both simple and efficient, and has been implemented in many MP-based video codecs.

III. Multiple Blocks Approximation

The approach in [1] and [11] assumes there is a high probability that the current largest energy block contains the *maximum atom* (i.e., the atom with the largest absolute inner product of the current residual frame). This assumption can be developed further by measuring the probability that the maximum atom can be found within blocks with different normalized energy, as shown by the Akiyo sequence in Fig. 1. We observe from the figure that most maximum atoms are found in higher energy blocks. Thus, a fast algorithm that includes the blocks with higher energy when searching for atoms probably yields an efficient approximation of the MP algorithm.

A. Block Selection

Finding an atom from the block with the highest energy is generalized so that an atom can be found in multiple blocks of relatively high energy. Before we propose our multiple block selection algorithm, we present a condition for the optimal set of blocks for atom selection and highlight the difficulties in obtaining the optimal set in practice.

Let $\mathbb{B}$ be the set of blocks in which we search for atoms. For a block b, let $P_0(b)$ be the probability that the maximum atom is not within b, and let $P_1(b)$ be the probability that b contains the maximum atom. Hence, $P_0(b) + P_1(b) = 1$. The miss probability P_M means that the block containing the maximum atom is excluded when searching for the maximum atom

$$P_M(\mathbb{B}) = \sum_{b \notin \mathbb{B}} P_1(b). \quad (2)$$

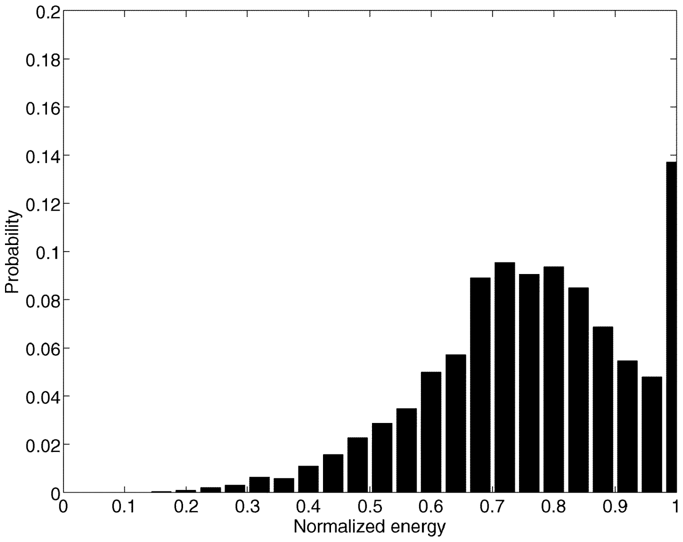

Fig. 1. Probability that the maximum atom can be found in blocks with different normalized energy. The blocks' energies are normalized so that the largest energy is equal to 1. The correlation between a block's energy and the block containing the maximum atom is high. This distribution was obtained from the Akiyo sequence encoded at 24 kbps with a frame-rate of 10 frames/s.

The false alarm probability P_F means that an atom is found in a block that does not contain the maximum atom

$$P_F(\mathbb{B}) = \sum_{b \in \mathbb{B}} P_0(b). \quad (3)$$

We define the average performance loss of selecting an atom incurred by $\mathbb{B}$ as

$$R(\mathbb{B}) = P_F(\mathbb{B})C_F + P_M(\mathbb{B})C_M$$

where the nonnegative numbers C_F and C_M represent the respective average conditional performance loss when a false alarm or a miss occurs. Because

$$\sum_b P_1(b) = 1 \quad (4)$$

from (2)–(4), we can derive

$$\begin{aligned} R(\mathbb{B}) &= \sum_{b\in\mathbb{B}} P_0(b)C_F + \sum_{b\notin\mathbb{B}} P_1(b)C_M \\ &= \sum_{b\in\mathbb{B}} P_0(b)C_F + \sum_{b\in\mathbb{B}} (1 - P_1(b))\, C_M \\ &= \sum_{b\in\mathbb{B}} (P_0(b)C_F - P_1(b)C_M) + \sum_{b\in\mathbb{B}} C_M \\ &= \sum_{b\in\mathbb{B}} (P_0(b)C_F - P_1(b)C_M) + C_M|\mathbb{B}| \quad (5) \end{aligned}$$

where $|\mathbb{B}|$ is the size of $\mathbb{B}$. Let the optimal set $\mathbb{B}^*$ be the block set that minimizes the above equation. A necessary condition for a block b in $\mathbb{B}^*$ is that $P_0(b)C_F - P_1(b)C_M \leq 0$. Otherwise, one can remove b from $\mathbb{B}^*$ and obtain a new optimal block set $\mathbb{B}^* - \{b\}$. For this block set, we have $R(\mathbb{B}^* - \{b\}) \leq R(\mathbb{B}^*)$, which contradicts the assumption that $R(\mathbb{B}^*)$ is the minimum value.

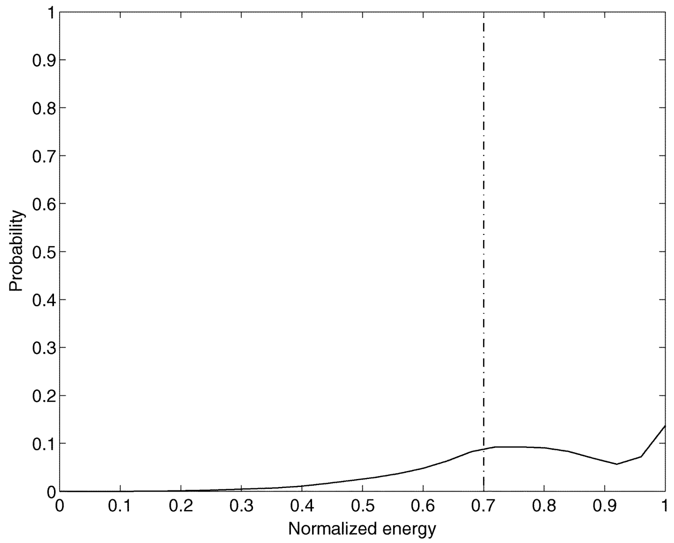

Fig. 2. Dotted line represents the chosen η. We assume that $\mathbb{B}$ is the set of blocks with normalized energy greater than η.

Let $\tilde{\mathbb{B}}$ be the set of blocks that satisfies

$$\tilde{\mathbb{B}} = \{b | P_0(b)C_F - P_1(b)C_M \leq 0\}. \quad (6)$$

Because any block in $\mathbb{B}^*$ must be in $\tilde{\mathbb{B}}$, we have

$$\mathbb{B}^* \subseteq \tilde{\mathbb{B}}. \quad (7)$$

Using $P_0(b) + P_1(b) = 1$, (6) can be rewritten as

$$\tilde{\mathbb{B}} = \{b | P_1(b) \geq \tau\} \quad (8)$$

where $\tau = (C_F/C_F + C_M)$. For a block b to be included in the optimal block set $\mathbb{B}^*$, $P_1(b)$ must be greater than the threshold τ. In practice, the optimal threshold τ^* is difficult to determine, so is the optimal set $\mathbb{B}^*$. However, a good ad-hoc procedure is to include those blocks whose P_1 is high enough to find atoms, as shown by the blocks with higher energy in Fig. 2. Thus, at each iteration of the MP algorithm, the blocks' energies are normalized so that the highest energy is equal to 1. A block b is assigned to $\mathbb{B}$ according to its normalized energy $||\tilde{b}||^2$

$$b \in \mathbb{B} \quad if \quad ||\tilde{b}||^2 \geq \eta$$

where $0 \leq \eta \leq 1$ is a threshold. An atom is then chosen from the blocks in $\mathbb{B}$.

If the MP were applied directly to the blocks in $\mathbb{B}$, as in a one-block algorithm, the complexity would increase by a factor of $|\mathbb{B}|$. The increase in complexity is an issue that needs to be addressed further.

B. *Block Content Approximation*

In low bit rate coding, which allocates a limited bit budget to each residual, we can use the MP algorithm to encode an approximation of a residual, instead of the residual itself. The approximated residual, $\tilde{f}$, is assumed to be in the subspace spanned by the union of the L bases of each block in $\mathbb{B}$. Since the MP algorithm is an efficient representation of a residual, the L bases of a block can be obtained by selecting the bases with the largest absolute inner products in the block. Let $\{g^b_{\gamma_i} | i = 1, \cdots, L\}$

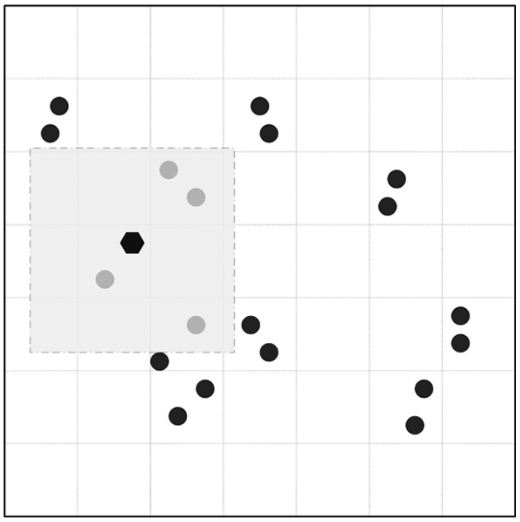

Fig. 3. After the base corresponding to the black hexagon at the center of the gray area has been selected, the inner products with the bases covered in the gray area are updated. The black dots are bases. In this example, $L = 2$ and $|\mathbb{B}| = 10$.

be the bases that have the largest absolute inner products in the block $b \in \mathbb{B}$.

$$\tilde{f} \in \text{Span}\left\{g^b_{\gamma_i} | i = 1, \cdots, L; b \in \mathbb{B}\right\}.$$

Our approach encodes $\tilde{f}$, instead of the original residual f. Fig. 3 shows an example in which $L = 2$ and $|\mathbb{B}| = 10$. If a block has $|s|^2$ pixels, and $|D|$ is the size of the dictionary D, then the total number of bases in a block is $|s|^2|D|$. Because L, the number of approximating bases, is much smaller than $|s|^2|D|$, the efficient MP update algorithm can be used to encode the approximating residual $\tilde{f}$. Our approach is explained by the following algorithm.

Multiple Blocks Update Algorithm:

Step 1) **Initialization** ($k = 0$): The residual f is first divided into blocks, and the blocks' energies are normalized so that the highest energy is equal to 1. A block whose normalized energy is greater than η is assigned to $\mathbb{B}$, and the inner products between the residual and the bases of the block are calculated. We then record the L bases that have the largest absolute inner products and assign them to B_L. We also obtain an approximated residual $R^0\tilde{f} = \tilde{f}$.

Step 2) **Apply MP Update** (**at** k**th iteration**): Find the base in B_L that yields the largest absolute inner product value in $R^k\tilde{f}$, and extract the atom to obtain $R^{k+1}\tilde{f}$. We then update the inner products according to the selected base and (1).

Step 3) **Update** $\mathbb{B}$: For a block that is not in $\mathbb{B}$, if its normalized energy is larger than η, we include it in $\mathbb{B}$. We then calculate the inner products between $R^{k+1}\tilde{f}$ and the bases within this block, and record the best L bases in B_L, as in Step 1.

Step 4) **Next iteration**: $k = k + 1$. If $k < n$, go to Step 2.

Step 1 uses all the bases to compute the inner products for each block in $\mathbb{B}$. From each block in $\mathbb{B}$, we then select the L bases that produce the largest absolute values. Let B_L be the union of the bases of all blocks. Because we assume that the approximating residual is in the subspace spanned by B_L, Step 2 applies the MP update algorithm to the bases in B_L to obtain new inner products. After an atom has been extracted from a block in Step 2, the energy of some blocks changes. Thus, new blocks are added to $\mathbb{B}$ in Step 3, and more new bases are obtained from those blocks. Because $|\mathbb{B}|$ is dependent on a residual, this algorithm is content-dependent. Also, $\mathbb{B}$ and B_L may grow as the number of iterations increases.

Step 1 calculates $|s|^2|D|$ inner product values for each block in $\mathbb{B}$. After the inner products have been obtained, the complexity of $|s|^2|D|$ for each block is required to extract the L bases with the largest absolute inner products from them [2]. Although the complexities of calculating and sorting the inner products are of the same order, inner product calculations are more expensive than sorting inner product values. Thus, in the analysis below, we omit the complexity of sorting inner product values. Since Step 1 applies MP to $|\mathbb{B}|$ blocks, it requires totally $|s|^2|D||\mathbb{B}|$ inner products. In Step 2, after the atom with maximum inner product has been selected, the inner products between the new residual and the bases in B_L must be updated. Since selecting the atom only affects part of the residual, according to (1), Step 2 only needs to update the inner products of the bases that have nonzero inner products with the newly selected atom. Fig. 3 shows an example where the number of inner products to be updated is 4.

As Step 3 may add new high energy blocks to $\mathbb{B}$, the L bases in the blocks are selected and assigned to B_L. For each newly added block, the computation cost will be the same as Step 1. Let m be the average number of bases that must be updated at each iteration. If a residual performs n iterations on average, we require

$$|s|^2|D||\mathbb{B}|\alpha + (n-1)m\beta \tag{9}$$

complexity to approximate a residual, where α and β, respectively, are the average complexity for calculating an inner product and performing an MP update operation. The first term in (9) is the sum of the costs of Steps 1 and 3, while the second term is the cost of applying $n-1$ times Step 2.

To achieve better efficiency than the one-block algorithm, we require that, after n iterations

$$|s|^2|D||\mathbb{B}|\alpha + (n-1)m\beta < |s|^2|D|n\alpha \tag{10}$$

where the term $|s|^2|D|n$ is the total number of inner products needed to find n atoms using the one-block algorithm. Equation (10) can be coarsely approximated by $\alpha > \beta$ and $m \ll |s|^2|D|$ to yield

$$|\mathbb{B}| < n. \tag{11}$$

Equation (11) stipulates the condition that makes our approach more efficient than traditional approaches. For a low bit rate (24 kb) QCIF video, a typical n is 200 and $|\mathbb{B}|$ is at most 99, if the block size is 16 $\times$16. Thus, the condition is usually satisfied and leads to an increase in the computation speed of our algorithm. In theory, satisfying the condition does not necessarily yield a better PSNR than one-block algorithms because each block is approximated. However, in practice, our simulation results indicate that the PSNRs of our algorithm for all the test sequences are better than those of the one-block algorithms. Next, we evaluate the performance of our algorithm.

TABLE I
RUNTIME FOR ENCODING THE AKIYO SEQUENCE BY DIFFERENT METHODS AT 24 kbps, 10 FRAMES/S FOR 3.3 S IN QCIF FORMAT. KEY: TOTAL TIME IS THE TIME REQUIRED TO OBTAIN MOTION VECTORS AND ENCODE RESIDUALS. NEFF IS THE ALGORITHM IN [11], AND OSAMA IS THE ALGORITHM IN [1]. OUR ALGORITHMS ARE IN THE THIRD AND THE FOURTH COLUMNS WITH $L = 100$ AND $L = 400$, RESPECTIVELY. $T_{\rm ip}$ IS THE RUN-TIME FOR INNER PRODUCT COMPUTATION; $T_{\rm sort}$ IS THE RUNTIME FOR FINDING AN ATOM IN THE FIRST TWO COLUMNS, OR L LARGEST INNER PRODUCT VALUES IN THE LAST TWO COLUMNS. AND $T_{\rm up}$ IS THE RUNTIME SPENT UPDATING INNER PRODUCT VALUES. NOTE THAT THE NEFF AND OSAMA ALGORITHMS DO NOT USE AN MP UPDATE ALGORITHM

Algorithm	Neff	Osama	L=100	L=400
Total time (seconds)	321.20	287.44	183.89	205.33
T_{ip}	185.94	160.65	56.67	67.34
T_{sort}	9.23	7.76	2.85	3.51
T_{up}	0	0	3.45	15.43

IV. PERFORMANCE EVALUATIONS AND COMPARISONS

To evaluate the performance of our algorithm, we compared it with the popular algorithms in [1] and [11] that search for an atom in the block with the highest energy. The index of a codeword is encoded by an adaptive arithmetic code. We set $\eta = 0.5$ in all our experiments. The inner product value is encoded by a bit plane-based approach and the position of a base is located by a quadtree and quadtree representation [8]. Other MP atom encoding methods can be used, but they change the average number of bits needed to encode an atom. For motion estimations, the basic mode in the standard H.263 [5], [6] is used to obtain motion vectors. The first frame of a video sequence is an intra-frame (I-frame) encoded by DCT; all other frames are inter-frames (P-frames) encoded by MP.

Table I shows the total runtime taken by various search algorithms to encode the Akiyo sequence. In our test platform, the CPU speed was 2.4 GHz/s. The total time is the overall runtime needed to encode the sequence for 3.3 s, and includes motion vector estimation and motion residual encoding. Our algorithm comprises three operations: computing the inner products between a residual and bases ($T_{\rm ip}$); sorting the largest L bases for each block ($T_{\rm sort}$); and updating the inner products of atom candidates ($T_{\rm up}$). The runtime of each operation is also shown in Table I. Our algorithm computes the inner products at the first iteration, and updates them in the following iterations. Because updating the inner products is relatively faster than computing new inner products, the overall runtime of our algorithm is constrained by $T_{\rm ip}$.

Figs. 4 and 5 are the plots of the average runtime versus the PSNR of various sequences at different bit rates. Our sequences include the following slow-motion sequences: Akiyo, Sean, Miss America, Container, Mother and Daughter, Salesman; and the fast-motion sequences: Carphone, Foreman, and Stefan. The PSNR of our method improves as L increases. This implies

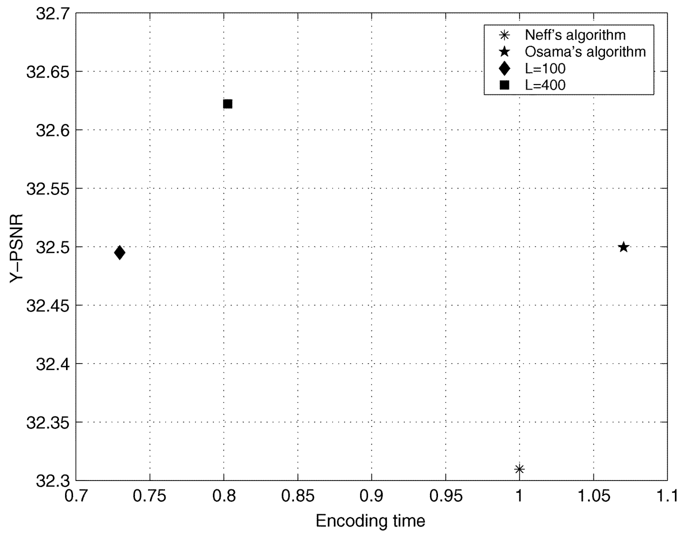

Fig. 4. Plot of the average runtime versus the PSNR of various methods for the following sequences: Akiyo, Sean, Container, Miss America, Mother and Daughter, Salesman, Carphone, Foreman, and Stefan. They were all run at 24 kbps, 10 frames/s for 3.3 s in QCIF format. In the legend, the last two items are our algorithms with different Ls.

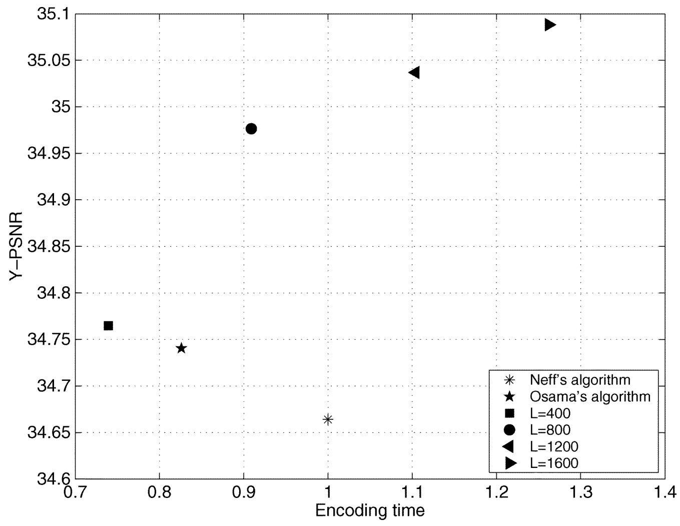

Fig. 5. Plot of the average runtime versus the PSNR of various methods for the following sequences: Akiyo, Sean, Container, Miss America, Mother and Daughter, Salesman, Carphone, Foreman, and Stefan. They were all run at 44 kbps, 10 frames/s for 3.3 s in QCIF format. In the legend, the last four items are our algorithms with different Ls.

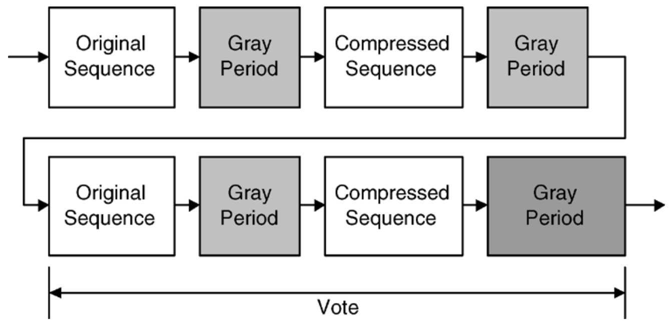

Fig. 6. Presentation structure of test sequences.

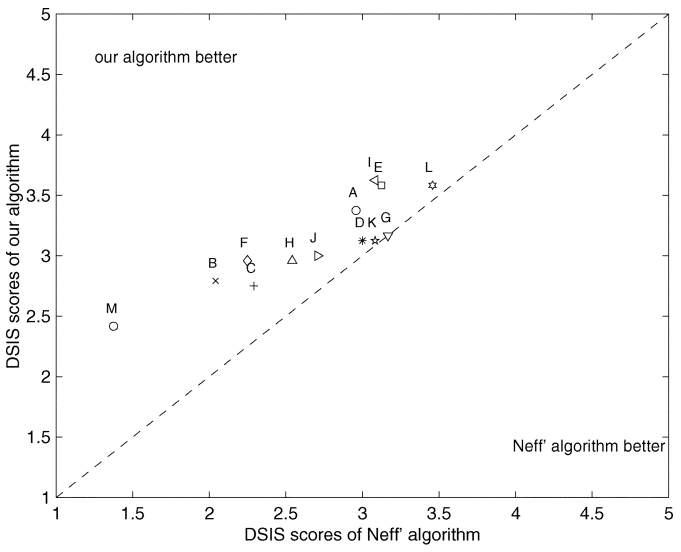

Fig. 7. Perceptual comparison of Neff's algorithm and our algorithm with $L = 400$ encoded at 24 kb/s for the following sequences: (A) Akiyo, (B) Carphone, (C) Mother and Daughter, (D) News, (E) Claire, (F) Foreman, (G) Hall monitor, (H) Salesman, (I) Sean, (J) Coast guard, (K) Container, (L) Miss America, and (M) Stefan.

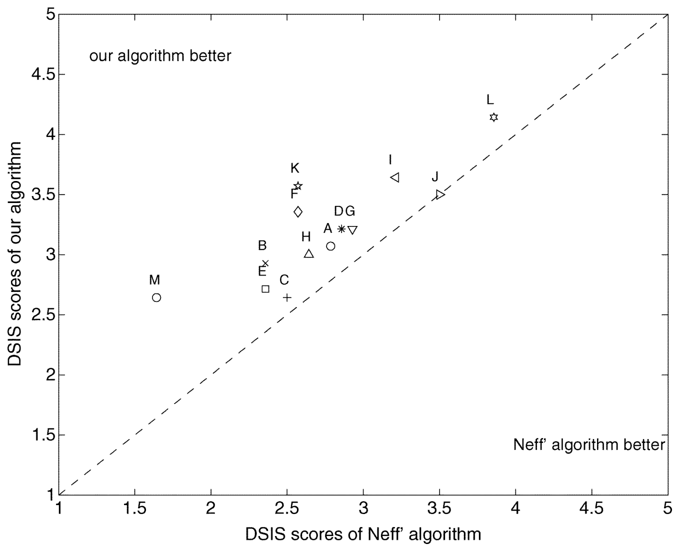

Fig. 8. Perceptual comparison of Neff's algorithm and our algorithm with $L = 800$ encoded at 44 kb/sec for the following sequences: (A) Akiyo, (B) Carphone, (C) Mother and Daughter, (D) News, (E) Claire, (F) Foreman, (G) Hall monitor, (H) Salesman, (I) Sean, (J) Coast guard, (K) Container, (L) Miss America, and (M) Stefan.

that using more bases to approximate a residual yields a better PSNR; however, the overall runtime also increases. The data in Fig. 4 shows that our approach with $L = 400$ achieves the best average PSNR performance of all methods at 24 kbps, and requires less runtime than both one-block algorithms. The PSNR improvement derived by this parameter $(L = 400)$ over Neff and Zakhor's one-block algorithm is, on average, 0.3–0.4 dB. We normalized the runtime of Neff and Zakhor's algorithm to 1, so that the comparison was not affected by the speed of the CPU. The runtime of our algorithm with $L = 400$ is 0.8 that of the one-block algorithms. Fig. 5 shows that the overall runtime increases linearly as a function of L. For L between 800 and 1200, our method achieves a PSNR gain of 0.4–0.5 dB over Neff and Zakhor's one-block algorithm at 44 kbps.

We evaluated and compared the subjective performance of our method to that of Neff's one-block algorithm, using the methodology for subjective assessment in [7]. We adopted the

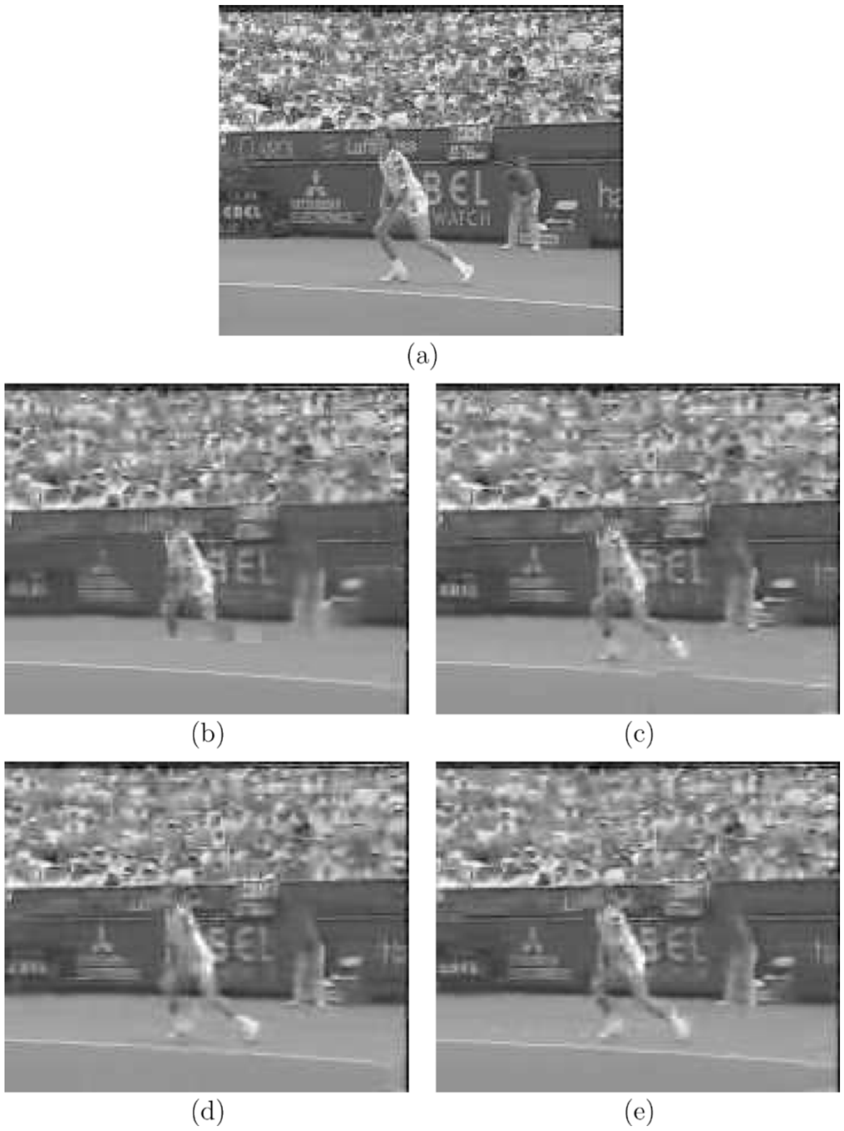

Fig. 9. Perceptual quality for Frame 81 of the QCIF Stefan sequence encoded at 44 Kbit/sec, 10 frames/sec. (a) Original frame. (b) Neff's algorithm. (c) Osama's algorithm. (d) Our algorithm with $L = 400$. (e) Our algorithm with $L = 800$.

double-stimulus impairment scale (DSIS) to evaluate the subjective quality. In this test procedure, participants were shown multiple sequence pairs consisting of an original and a compressed sequence, both of which were rather short. The original sequence was presented first, followed by a gray period, then the compressed sequence was presented. Both sequences were presented twice, as shown in Fig. 6. The participants were required to vote using a five-grade impairment scale: imperceptible (5), perceptible, but not annoying (4), slightly annoying (3), annoying (2), and very annoying (1). The subjective evaluation results of our update algorithm and one-block algorithm for thirteen test sequences encoded at various bit rates are shown in Figs. 7 and 8. The DSIS grading scores of 40 participants were averaged. The evaluation results show that our proposed update algorithm not only achieves a superior PSNR performance and runtime, but also has better perceptual quality. The snapshots of video sequences encoded by different methods at various bit rates are shown in Fig. 9. Clearly, our algorithms yield a better perceptual quality in encoding the feet and border lines below the feet.

V. Conclusion

In contrast to traditional video encoding approaches, in which an atom is chosen from the block with the highest (weighted) energy, we approximate a residual in a subspace spanned by a few MP bases in multiple blocks. From this approximation, we obtain a new MP atom finding algorithm that uses multiple blocks for atom searching, and the MP update algorithm to update the inner product values. The analysis and simulations show that our proposed algorithm outperforms one-block algorithms in terms of PSNR, runtime, and perceptual quality. The performance of our method depends on two parameters: the number of blocks and the number of bases in each block. Adaptation of the parameters for different video sequences to achieve the best performance is an issue worthy of further study.

References

[1] 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, no. 1, pp. 123–143, Feb. 1999.

[2] A. V. Aho, J. E. Hopcroft, and J. D. Ullman, *The Design and Analysis of Computer Algorithms*: Addison-Wesley, 1974, pp. 97–100.
[3] P. Czerepiński, C. Davies, N. Canagarajah, and D. Bull, "Matching pursuits video coding: dictionaries and fast implentation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, no. 7, pp. 1103–1115, Oct. 2000.
[4] B. Jeon and S. Oh, "Fast matching pursuit with vector norm comparison," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 4, pp. 338–342, Apr. 2003.
[5] *Video Coding for Audio Visual Services at Px64 kbit/s*, ITU-T Recommendation H.263, 1995.
[6] *Video Coding for Low Bit Rate Communication*, ITU-T Recommendation H.263, Sep. 1997.
[7] *Methodology for the Subjective Assessment of the Quality of Television Pictures*, ITU-R Recommendation BT.500-11, 2002.
[8] J. L. Lin, W. L. Hwang, and S. C. Pei, "SNR scalability based on bitplane coding of matching pursuit atoms at low bit rates: fine-grained and two-layer," *IEEE Trans. Circuits Syst. Video Technol*, vol. 15, no. 1, pp. 3–14, Jan. 2005.
[9] ——, "A combined dictionary approximation and maximum atom extraction design for matching pursuit speed-up," IEEE Trans. Circuits Syst. Video Technol, submitted for publication.
[10] G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *IEEE Trans. Signal Process.*, vol. 41, no. 12, pp. 3397–3415, Dec. 1993.
[11] R. Neff and A. Zakhor, "Very low bit rate video coding based on matching pursuits," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 1, pp. 158–171, Feb. 1997.
[12] R. Neff, T. Nomura, and A. Zakhor, "Decoder complexity and performance comparison of matching pursuit and DCT-based MEPG-4 video codecs," in *Proc. IEEE Int. Conf. Image Process.*, 1998, pp. 783–787.
[13] R. Neff and A. Zakhor, "Matching pursuit video coding-part I: dictioanry approximation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 12, no. 1, pp. 13–26, Jan. 2002.
[14] D. W. Redmill, D. R. Bull, and P. Czerepiński, "Video coding using a fast non-separable matching pursuits algorithm," in *Proc. IEEE Int. Conf. Image Processing.*, 1998, pp. 769–773.
[15] M. Vetterli and T. Kalker, "Matching pursuit for compression and application to motion compensated video coding," in *Proc. IEEE Int. Conf. Image Process.*, 1994, pp. 725–729.
[16] C. De Vleeschouwer and B. Macq, "Subband dictionaries for low-cost matching pursuits of video residues," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 7, pp. 984–993, Oct. 1999.

Jian-Liang Lin was born in I-Lan, Taiwan, R.O.C., in 1975. He received the B.S. degree in electronic engineering from Fu Jen Catholic University, Taipei, Taiwan, R.O.C., in 1997, and the M.S. degree in electrical engineering and the Ph.D. degree in communication engineering from National Tsing Hua University, Hsinchu, Taiwan, R.O.C., in 1999 and 2006, respectively.

He was a Visiting Student at the University of Washington, Seattle, from March 2004 to March 2005. Currently, he is a Postdoctoral Fellow at the Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C., where he served as a Research Assistant from 1999 to 2005. His research interests include matching pursuits, image and video processing, as well as multimedia compression and transmission.

Wen-Liang Hwang received the B.S. degree in nuclear engineering from National Tsing Hua University, Hsinchu, Taiwan, R.O.C., the M.S. degree in electrical engineering from the Polytechnic Institute of New York, and the Ph.D. degree in computer science from New York University, New York, in 1993.

He was a Postdoctoral Researcher with the Department of Mathematics, University of California, Irvine, in 1994. In January 1995, he became a member of the Institute of Information Science, Academia Sinica, Taipei, Taiwan, R.O.C.m where he is currently a Research Fellow. He is co-author of the book *Practical Time-Frequency Analysis* (Academic, 1998). His research interests include wavelet analysis, signal and image processing, and multimedia compression and transmission.

Dr. Hwang was awarded the Academia Sinica Research Award for Junior Research in 2001.

Soo-Chang Pei (SM'98–F'00) was born in Soo-Auo, Taiwan in 1949. He received B.S.E.E. degree from National Taiwan University (NTU), Taipei, Taiwan, R.O.C., in 1970 and the M.S.E.E. and Ph.D. degrees from the University of California, Santa Barbara (UCSB), in 1972 and 1975, respectively.

He was an engineering officer in the Chinese Navy Shipyard from 1970 to 1971. From 1971 to 1975, he was a Research Assistant at UCSB. He was a Professor and Chairman in the Electrical Engineering Department, Tatung Institute of Technology, from 1981 to 1983, and at NTU from 1995 to 1998 . Presently, he is a Professor in the Electrical Engineering Department at NTU. His research interests include digital signal processing, image processing, optical information processing, and laser holography.

Dr. Pei received the National Sun Yet-Sen Academic Achievement Award in Engineering in 1984, the Distinguished Research Award from the National Science Council, R.O.C., from 1990 to 1998, the Outstanding Electrical Engineering Professor Award from the Chinese Institute of Electrical Engineering in 1998, the Academic Achievement Award in Engineering from the Ministry of Education in 1998, the IEEE Fellow in 2000 for contributions to the development of digital eigenfilter design, color image coding and signal compression, and electrical engineering education in Taiwan, the Pan Wen-Yuan Distinguished Research Award in 2002, and the National Chair Professor Award from the Ministry of Education in 2002. He was President of the Chinese Image Processing and Pattern Recognition Society in Taiwan from 1996–1998, and is a member of Eta Kappa Nu and the Optical Society of America.