INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2013, VOL. 59, NO. 1, PP. 9–16
Manuscript received January 16, 2013; revised March, 2013. DOI: 10.2478/eletel-2013-0001

Inter-frame Prediction with Fast Weighted Low-rank Matrix Approximation

Zhi-Long Huang and Hsu-Feng Hsiao

Abstract—**In the field of video coding, inter-frame prediction plays an important role in improving compression efficiency. The improved efficiency is achieved by finding predictors for video blocks such that the residual data can be close to zero as much as possible. For recent video coding standards, motion vectors are required for a decoder to locate the predictors during video reconstruction. Block matching algorithms are usually utilized in the stage of motion estimation to find such motion vectors. For decoder-side motion derivation, proper templates are defined and template matching algorithms are used to produce a predictor for each block such that the overhead of embedding coded motion vectors in bit-stream can be avoided. However, the conventional criteria of either block matching or template matching algorithms may lead to the generation of worse predictors. To enhance coding efficiency, a fast weighted low-rank matrix approximation approach to deriving decoder-side motion vectors for inter frame video coding is proposed in this paper. The proposed method first finds the dominating block candidates and their corresponding importance factors. Then, finding a predictor for each block is treated as a weighted low-rank matrix approximation problem, which is solved by the proposed column-repetition approach. Together with mode decision, the coder can switch to a better mode between the motion compensation by using either block matching or the proposed template matching scheme.**

Keywords—**Inter-frame prediction, template matching, block matching, low-rank matrix approximation, weighted low-rank matrix approximation.**

I. Introduction

In the modern video compression standards, a hybrid approach with block-based coding structures is adopted with great popularity. Many sophisticated intra- and inter-frame predictive coding tools, entropy coding tools in a transform domain, and adaptive filters are integrated to provide good coding efficiency.

The inter-frame predictive coding is an important technology used to remove temporal redundancy among adjacent frames. In the MPEG-4 Advanced Video Coding standard [1] and also the new High Efficiency Video Coding standard [2], motion vectors are coded into bit streams for decoders to retrieve predictors. In order to find suitable motion vectors at an encoder side, block matching algorithms are usually used to search for a similar block in a reference frame with minimal mean absolute difference. Therefore, the energy of residual data can be kept small in the hope of better coding performance.

This research was supported by the NSC of Taiwan under grant number 101-2221-E-009-086-MY3, and by the Ministry of Economic Affairs (MOEA) of Taiwan under grant number 101-EC-17-A-03-S1-214.

Z.-L. Huang and H.-F. Hsiao are with Department of Computer Science, National Chiao Tung University, 1001 University Rd., Hsinchu, Taiwan (e-mails: huangjl.cs99g@nctu.edu.tw; hillhsiao@cs.nctu.edu.tw).

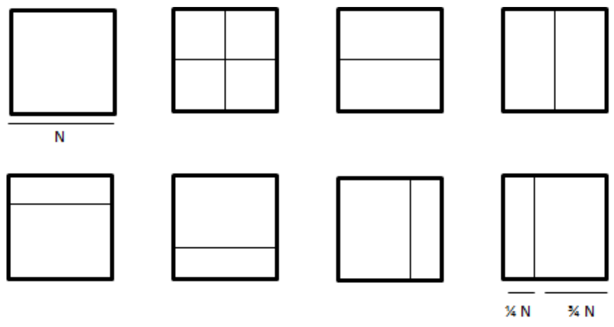

Fig. 1. Different partitions of a coding unit to form prediction units in HEVC.

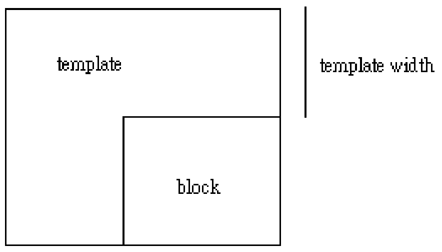

Fig. 2. Template in the shape of an upside-down L.

The block partition in HEVC [2] follows a quadtree structure. A frame can be partitioned into several largest coding units, and each coding unit can be further partitioned into four coding units, recursively. A coding unit is therefore a tree node of the quadtree. Each coding unit at the leaf nodes of the quadtree can be further partitioned into several prediction units. There are 8 kinds of prediction unit partitions, as shown in Fig. 1. A prediction unit is used as a unit for predictive coding in both intra-frame and inter-frame prediction. In inter-frame coding, either one or two motion vectors are coded into bit-stream for each partition.

In order to further improve coding efficiency by not requiring the overhead of motion vectors, a template matching prediction algorithm is proposed in [3]. The template for a block is defined as the upside-down L shape area at the upper and left sides of the block as shown in Fig. 2. At the encoder and the decoder, template matching algorithms are used to find the same block predictor so that the mean absolute difference of the template area is minimal.

The work in [4] is proposed to improve the coding efficiency of [3]. The target template in [4] is used to find several similar templates in the reference frame. For easier discussion, the template regions searched in the reference frame are referred to as the template candidates and the block at the lower and right sides of a template candidate is a block candidate. The average of the block candidates is used as the predictor of the target block in [4]. Instead of plain average, the method in [5] predicts the target block by performing linear combination of the chosen block candidates. The weighting coefficients are determined by using the template candidates and the template of the target block. Specifically, the weighting coefficients are determined by finding the linear combination of template candidates to be close to the target template.

In addition to the upside-down L-shaped template, seven different shapes are used in [5], [6]. For each block, template matching algorithm is performed with 8 different shaped templates. The predictor of the target block is chosen from the 8 predictors according to a rate distortion optimization criterion. However, when the texture of the target template is flat, the performance of the template matching algorithm is not promising. In [7], the decimated version of a target block in flat regions is used for better performance.

In [8], the authors imply that the predictor formed by combing a number of prediction signals can be better than the predictor found solely by using the block matching algorithm. They propose to produce the predictor of a target block by linearly combining three candidates of prediction signals. These three candidates are the co-located block of the target block, the block pointed with the median of nearby motion vectors, and the block searched by using the block matching algorithm. The nearby motion vectors are defined as the motion vectors of the three neighbouring blocks of the target block: the left, top, and top-right (or top-left if top-right is not available) blocks. Because the position of the co-located block and the median of nearby motion vectors can be obtained at the decoder side, only the motion vector searched by using the block matching algorithm needs to be encoded in the bit stream.

The block size in general is rectangular; however, the shape of an object in the scene is usually irregular, and the block-based motion estimation can't predict the objects in a frame with irregular shape well. The work in [9] adapts the size and the shape of the motion estimation area to the objects in the scene to improve the performance of the overall coding process. Because the reconstruction of the previous frame is available at the decoder side, the shape of the motion estimation area is determined according to the previous frame. Therefore, the shape can be derived at the decoder side accordingly, and it is not required to be included in the bit stream.

The approaches of the low-rank approximation have been applied in many fields, such as recommendation system [10], face recognition [11], and video denoising [12]. In the field of video coding, it has been applied in the intra-frame prediction algorithm by representing image blocks as a low-rank or approximately low-rank matrix [6].

Since the conventional criteria of either block matching or template matching algorithms may lead to the generation of worse predictors, a fast weighted low-rank matrix approximation used to derive decoder-side motion vectors for inter-frame video coding is proposed in this paper. Specifically, the template matching algorithm is used to find several template candidates and the importance of each candidate is determined, followed by the fast weighted low-rank approximation approach to finding the block predictor such that the difference between the predictor and target block can be as small as possible.

The structure of the paper is as the following. Section II is the proposed fast weighted low-rank matrix approximation for inter-frame predictive coding. The simulation results and the conclusions are shown in Section III and Section IV, respectively.

II. The Proposed Fast Weighted Low-Rank Matrix Approximation Based Inter-Frame Prediction

The inter-frame predictive coding is one of the important compression tools for modern video compression standards where additional information for motion vectors is required for a decoder to obtain block predictors. Decoder-side motion vector derivation algorithms, such as template matching algorithms, have been proposed in the literature to eliminate such overhead. However, since a target block is not yet available during the decoding process, the residual information is not as close to zero as the result by using block matching algorithms. Actually, there are many cases that the residual data produced by using either block matching or template matching algorithms can be further reduced for better compression efficiency.

The framework of the proposed predictor estimation scheme is shown in Fig. 3. In the proposed method, template matching is used to find several template candidates. A template candidate and the corresponding block candidate are organized to form a column vector. The importance of each column vector is determined according to the projection of the target template on the template candidates. The predictor can then be produced through the proposed fast weighted low-rank matrix approximation. The low-rank matrix approximation problems have drawn much attention in several areas. In general, it can be regarded as the optimization problem with the restriction of rank.

Since the block size in the MPEG-4 AVC standard and the new HEVC standard can be variable, the template width for a block size needs to be determined first. Then, we find the

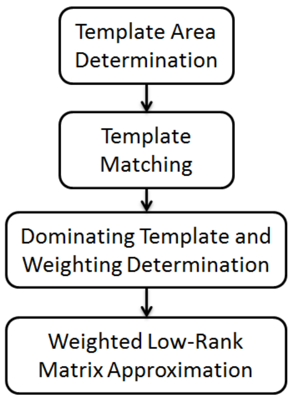
Template Area Determination
Template Matching
Dominating Template and Weighting Determination
Weighted Low-Rank Matrix Approximation

Fig. 3. Framework of the proposed inter-frame predictor estimation scheme.

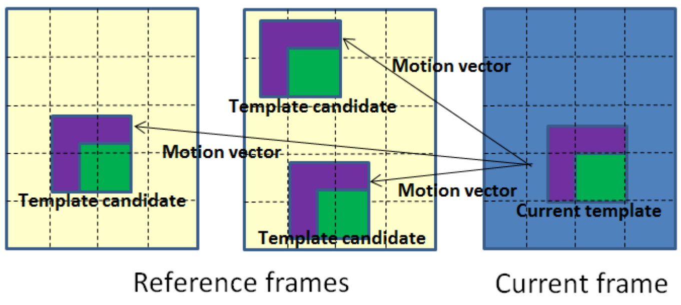

Fig. 4. Template matching between target template and template candidates.

template candidates, which are similar with the target template, by performing template matching. Each region consists of an upside-down L-shaped template and a block as shown in Fig. 2 and Fig. 4.

For each possible template in the reference frames, the mean absolute difference (MAD) between the possible template candidate and target template is computed as shown in (1).

$$MAD(T_t, T_c) = \frac{1}{N}\sum_{i=1}^{N} |p_i - p_i'|, \quad (1)$$

where T_t stands for the target template and T_c stands for a possible template candidate. p_i and p_i' stand for the pixel values at the same position within T_t and T_c, respectively. N is the number of pixels within the template. The template candidates are determined by selecting m_1 regions with smaller MAD values. For each template candidate i and the corresponding block candidate, their pixels are rearranged to form a column vector V_i according to the vertical scanning order as shown in Fig. 5.

Similarly, the pixels of the target block and the target template can form a target column vector V_s with the same scanning order shown in Fig. 5. For a conventional low-rank matrix approximation problem, matrix D is defined as (2). Because the criterion of selecting those vectors in this paper is to find the candidates with smaller MAD, there is a good chance that the matrix D has the property of low rank and the predictor of target block can be estimated with good accuracy. One of the solutions to a low-rank matrix approximation problem can be found in [13].

$$D = [V_s, V_1, V_2, \cdots, V_{m1}]. \quad (2)$$

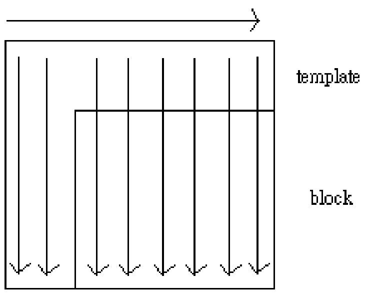

Fig. 5. Vertical scanning order to form a column vector.

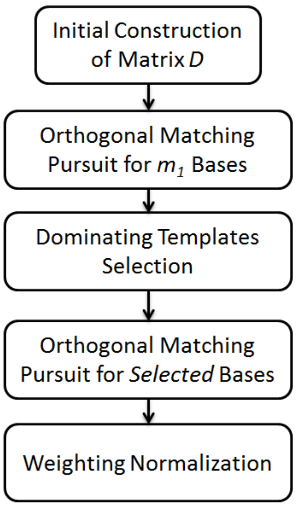
Initial Construction of Matrix D
Orthogonal Matching Pursuit for m1 Bases
Dominating Templates Selection
Orthogonal Matching Pursuit for Selected Bases
Weighting Normalization

Fig. 6. The selection of dominating template candidates and the corresponding weighting values.

However, the MAD between each template candidate and the target template is most likely different. The selection of template candidates and their contribution to form the block predictor shall be adjusted accordingly, which is described as follows.

A. Dominating Template Candidates and Corresponding Weightings

When we produce the matrix D in the low-rank matrix approximation problem, the number of the regions to construct the matrix can affect the performance of the prediction. In reference [5], the template candidates are viewed as the basis and the orthogonal matching pursuit algorithm [14] is used to find the coefficients for each basis, so that the linear combination of the bases is regarded as the predictor of the target template. In our method, the orthogonal matching pursuit algorithm is also used. However, the role of the orthogonal matching pursuit algorithm is to find a suitable set of dominating column vectors as well as their importance factors. This procedure of selecting dominating template candidates and weightings is shown in Fig. 6.

Initially, the matrix D is obtained by constructing V_s and column vectors V_i as described earlier, for $1 \le i \le m_1$. m_1 is

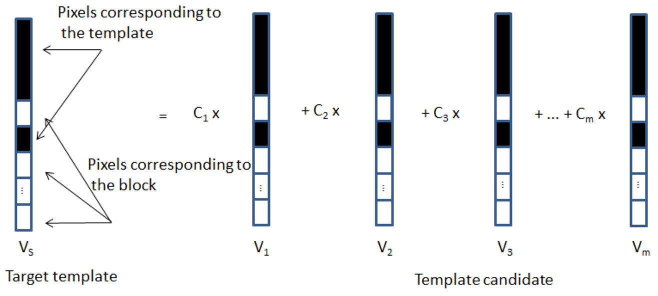

Fig. 7. Orthogonal matching pursuit for target template and candidate template.

equal to 15 in this paper. Including V_s, there are 16 column vectors in the matrix D.

In the second step of Fig. 6, the template parts of the m_1 vectors are treated as the initial bases. The orthogonal matching pursuit algorithm is used to find the coefficients c_i to represent the target template with the choice of vectors. The schematic diagram is shown in Fig. 7 where each bar stands for a column vector in the matrix D, and the regions in black stands for the pixels from template region.

If there are n bases, the orthogonal matching pursuit algorithm needs n iterations to derive the coefficients with respect to all the bases. For the first iteration, orthogonal matching pursuit selects the basis having the highest correlation with the column vector corresponding to the target template. The coefficient with respect to the selected basis and the residual are then determined. The residual is derived by subtracting the vector corresponding to target template from its component on the selected basis. This component means the quotient of the selected basis and its coefficient. For the k-th ($k > 0$) iteration, the algorithm selects the basis that has the highest correlation with the residual from the bases not selected in previous iterations, and then calculates the corresponding coefficient. Then the orthogonal matching pursuit algorithm updates the coefficients that have been collected so that the newly derived residual is orthogonal to not only the immediately selected basis, but also all the bases selected at previous iterations.

The magnitude of each coefficient is used to represent the importance of the corresponding template candidate. In the step of selecting dominating templates, partial normalization of coefficients is introduced as follows.

For each k that satisfies $1 \leq k \leq m_1$, the sum of the normalized coefficients in (3) is computed.

$$\begin{aligned} c'_{i,k} &= |round(c_i/c_k)|, \\ \text{sum}_k &= \sum_{i=1}^{m_1} c'_{i,k}. \end{aligned} \tag{3}$$

The task in this step is to find the largest c_k such that sum_k is larger than or equal to m_1. The dominating templates are defined as the template candidates whose $c'_{i,k}$ are not 0. The dominating templates are the selected template candidates that will be used to produce the predictor.

Based on the choice of dominating templates, the orthogonal matching pursuit algorithm is applied again to find the coefficients to represent the target template based on the selected dominating template candidates. The coefficients are then normalized so that the summation of the integral coefficients is m_1.

B. *Fast Weighted Low-Rank Matrix Approximation*

Conventionally, the solutions to weighted low-rank matrix approximation problems require a lot of computation effort. In the proposed fast weighted low-rank matrix approximation scheme, a normalized coefficient represents the importance of the corresponding column vector which will be repeated for that amount of times to form a new matrix D. Then the weighted low-rank approximation problem is transformed to a low-rank approximation problem, which can be solved much faster.

The definition of the low-rank matrix approximation problem is shown below.

$$\begin{gathered} \min \text{rank}(A), \\ \text{subject to } A_{ij} = D_{ij}(i,j) \in \Omega. \end{gathered} \tag{4}$$

The matrix A is the matrix with unknown samples recovered. Ω is the set of index of known samples inside the matrix D. The definition of the problem is to find the unknown samples inside the matrix D such that the rank of the matrix can be minimized. Furthermore, in [15], the authors showed that the unknown samples in matrix D could be found by solving the following problem.

$$\min ||A||_* \text{ subject to } A_{ij} = D_{ij}(i,j) \in \Omega. \tag{5}$$

Equation (5) means that finding the unknown samples inside the matrix D such that the nuclear norm of the matrix can be minimized. The nuclear norm of a matrix is the sum of the absolute values of the singular values of the matrix. The formula in (5) can be further expressed as the following formula:

$$\min ||A||_* \text{ subject to } A + E = D, P_\Omega(E) = 0. \tag{6}$$

P_Ω is a linear operator that keeps the samples in the set Ω unchanged and sets those samples outside the set Ω zeros.

The matrix E is the matrix used to compensate the unknown samples of matrix D, and the unknown samples of the matrix D are initialized as 128. The problem can then be solved by using one of the many existing algorithms, such as the inexact augmented Lagrange multiplier (IALM) method [13]. The augmented Lagrange multiplier method approaches the approximation problem by minimizing the nuclear norm of the incomplete matrix, and it is shown to be much faster than the exact augmented Lagrange multiplier method. The partial augmented Lagrangian function [16] of (6) is described as follows.

$$L(A,E,T,\mu) = ||A||_* + <Y, D-A-E> + \frac{\mu}{2}||D-A-E||_F^2. \tag{7}$$

Where μ is the parameter of IALM which is defined in [13]. The IALM is described as the following.

The IALM algorithm minimizes the partial augmented Lagrangian function by updating A, E, and Y iteratively. At the beginning of each iteration, the algorithm updates A by minimizing the partial augmented Lagrangian function with respect to A, while Y and E remains fixed. The method to update A is subtracting a constant from each singular value of A. This constant is a parameter that can influence the performance of the algorithm [13]. Then, the IALM algorithm updates E by setting its elements corresponding to the known samples as zeros. At last, the amount of violation of the constraint $A + E = D$ is used to update Y. More details and parameter selection about the implementation of the IALM method can be found in [13].

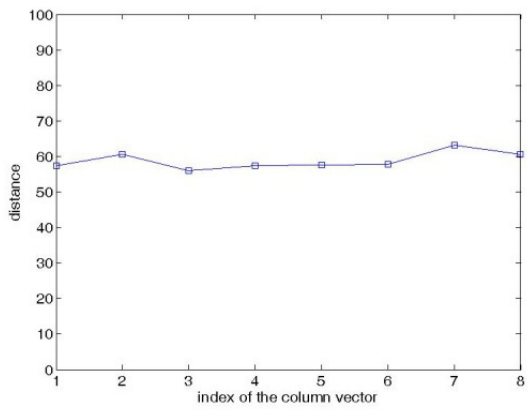

Fig. 8. Distance between the target column vector and the other column vectors.

The proposed fast weighted low-rank matrix approximation is to transform the problem to generic low-rank matrix approximation through column repetition. To observe the effect of the column repetition on the predictors, we design a series of experiment. Supposed that a matrix A_1 consists of nine column vectors, V_1 to V_9. $A_1 = [V_1\ V_2\ V_3\ V_4\ V_5\ V_6\ V_7\ V_8\ V_9]$ and it is shown in (8).

$$A_1 = \begin{bmatrix} 104 & 131 & 151 & 108 & 145 & 135 & 119 & 130 & 146 \\ 147 & 139 & 126 & 151 & 103 & 142 & 107 & 149 & 141 \\ 120 & 146 & 148 & 129 & 105 & 151 & 135 & 151 & 112 \\ 119 & 148 & 116 & 111 & 103 & 127 & 110 & 140 & x \\ 124 & 114 & 149 & 113 & 115 & 149 & 110 & 144 & x \\ 129 & 131 & 111 & 114 & 149 & 104 & 139 & 106 & x \\ 109 & 139 & 138 & 109 & 137 & 127 & 123 & 144 & x \\ 150 & 111 & 151 & 144 & 115 & 116 & 128 & 131 & x \\ 131 & 136 & 119 & 106 & 144 & 150 & 106 & 120 & x \end{bmatrix} \tag{8}$$

V_9 is the target column vector and the samples marked as x are the unknown samples. After the unknown samples of matrix A_1 are recovered by treating it as a low-rank matrix approximation problem and applying IALM algorithm, the transpose of the resulting target column vector is: [146 141 112 109.09 120.49 111.46 131.38 127.65 131.15]. The distance between the resulting target column vector and the other column vectors, V_1 to V_8, is shown in Fig. 8. The value at the x-axis is the index of the column vector used to calculate the distance from the resulting target column vector. From Fig. 8, we can find that for different column vector V_i, where i is between 1 and 8, their distances are close to each other.

Then, we modify matrix A_1 by repeating the column vector V_1 a number of times, and observe the effect of the repetition of column vector V_1 on the resulting target column vector obtaining by solving the low-rank matrix approximation problem. The result is shown in Fig. 9 where the value at the x-axis is the number of repetition times of column vector V_1, and the value at the y-axis is the distance between the column vector V_1 and the resulting target column vector. From Fig. 9, we can find that the distance decreases when the number of the repetition times increases.

The decrease of the distance between the column vector V_1 and the target column vector means that the unknown samples in the target column are closer to the corresponding samples in the vector V_1 when the number of repetition times is larger. Therefore, by repeating the column vectors of the matrix, we can adjust the degree of similarity between the unknown samples in the target column vector and the corresponding samples in the other column vectors.

The size of a column vector is determined by the size of block and also the size of template. The size of the template can affect the performance of a template-matching algorithm and the accuracy of predictors. In order to study for a proper template size, the following experiment is designed.

The generic low-rank matrix approximation based inter-frame predictive coding is used to encode the video sequence *suzie*. The results of the mean absolute difference (MAD) between the target block and its predictor with respect to different template sizes are shown in Fig. 10. Two different block sizes, 4x4 and 8x8, are simulated. The template sizes are adjusted by changing the width of template illustrated in Fig. 2. From the experiment results, we find that the MAD is smaller when the pixel number of a template is about 5 times of the pixel number of a block.

When we solve the low rank matrix approximation problem to predict the target block, the performance shall be better with

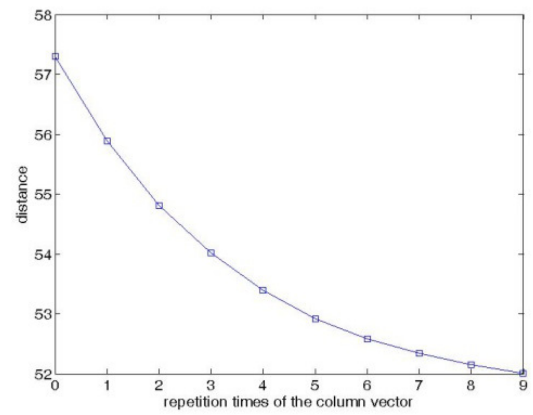

Fig. 9. Distance between the target column vector and the other column vectors, where matrix A_1 is modified by using column repetition.

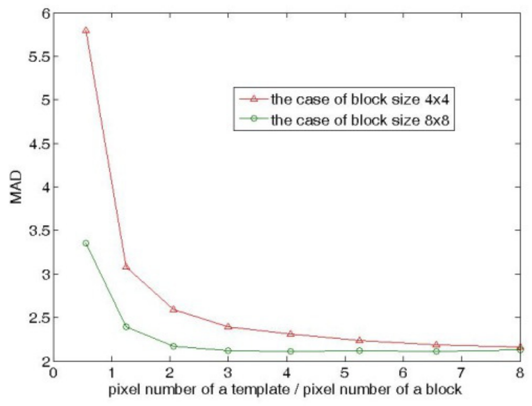

Fig. 10. MAD performance with respect to different template sizes.

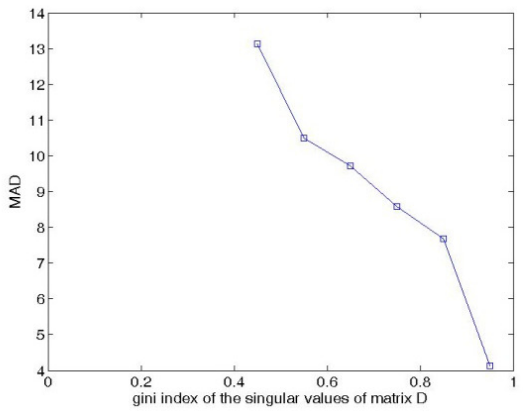

Fig. 11. MAD performance with respect to different gini values of singular values of the matrix D.

lower rank of the matrix. In addition, if the rank of the matrix is lower, the singular values of the matrix are more centralized.

We design another experiment to see the relation between the degree of concentration of the singular values of the matrix D and the performance of the algorithm. We use the value of gini index [17] to quantify the degree of the concentration of the singular values of the matrix D. Given a ordered vector $f = [f(1), f(2), \ldots, f(N)]$, $|f(1)| \le |f(2)| \le \cdots < |f(N)|$, the calculation of gini index is shown in (9).

$$GI(\underline{f}) = 1 - 2\sum_{k=1}^{N} \frac{f(k)}{|(\underline{f})|_1}\left(\frac{N-k+\frac{1}{2}}{N}\right), \tag{9}$$

where $|(\underline{f})|_1$ is the norm 1 of $\underline{f}$. The value of gini index is normalized, and its range is from 0 to 1. If the value of gini index is larger, it means that the data set $(f(1), f(2), \ldots, f(N))$ is sparser. The video sequence in this experiment is mobile and the search range is +-15. From the results shown in Fig. 11, we can find that the MAD decreases when the degree of the concentration of singular values of matrix D increases.

Because the template matching is to search for the most similar template candidates in the reference frame to the template around the target block, the MAD performance of the decoder-side template matching may not be always better than the performance of block matching. Therefore, for each block, template matching with the fast weighted low-rank matrix approximation and block matching algorithm are considered at the same time. If the fast weighted low-rank matrix approximation is turned off for a block, motion vectors are imbedded into bit-stream. Otherwise, if the fast weighted low-rank matrix approximation-based template matching is used, the overhead of motion vectors is not required. The rate-distortion optimization according to the Lagrangian cost function J in (10) can be used to choose a better mode M to operate at the encoder side.

$$\min_{M}\{J = D_c + \lambda \cdot R\}, \tag{10}$$

where α is the Lagrange multiplier. D_c and R in (10) are video distortion and bit-rate, respectively.

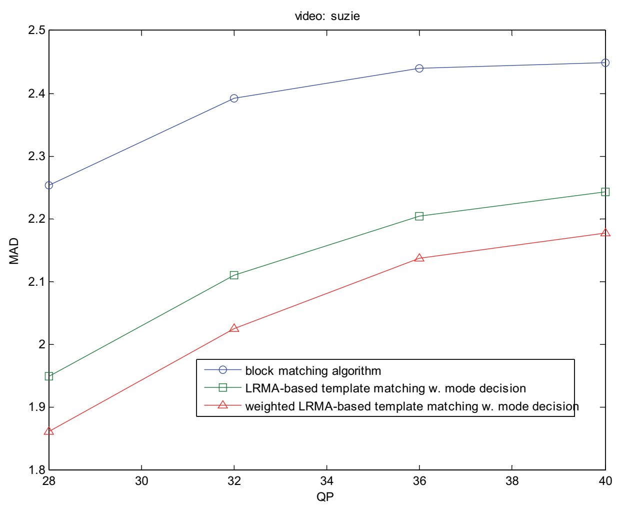

Fig. 13. The MAD performance comparison for the video sequence *suzie*.

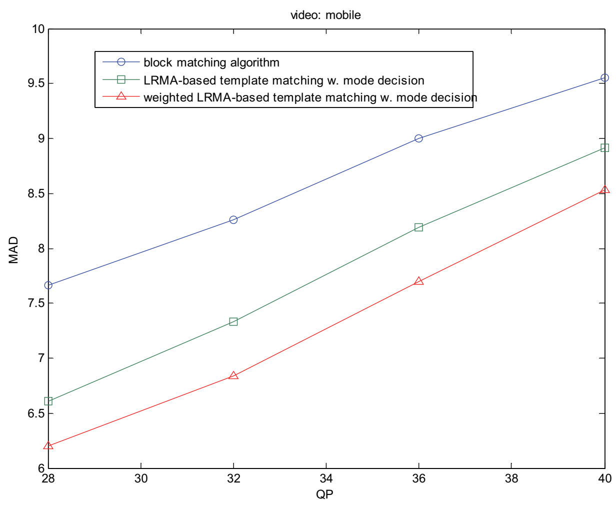

Fig. 12. The MAD performance comparison for the video sequence *mobile*.

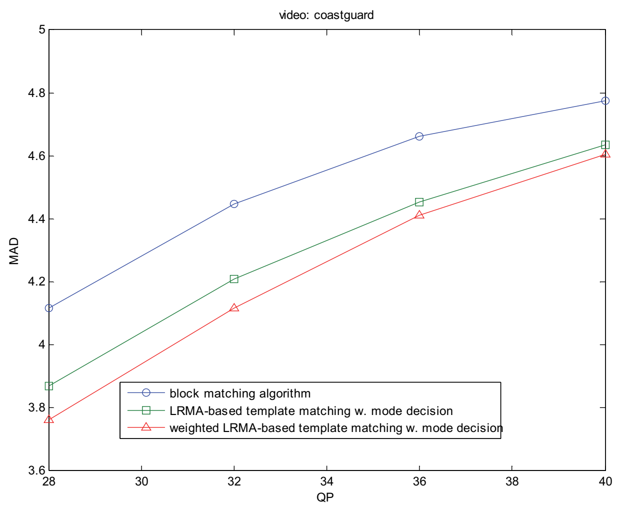

Fig. 14. The MAD performance comparison for the video *coastguard*.

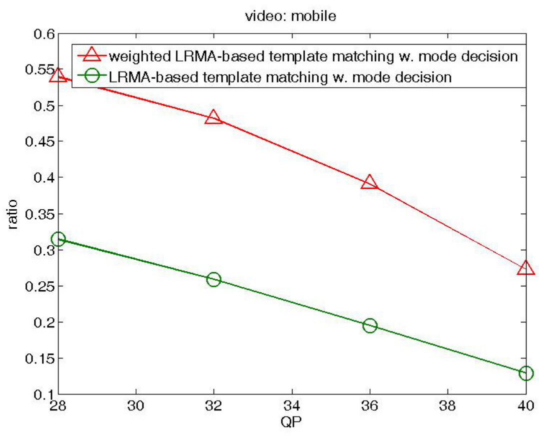

Fig. 15. The ratio of block numbers used by using template matching for the video sequence *mobile*.

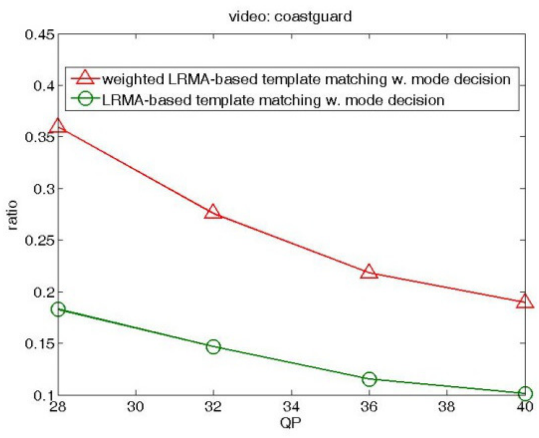

Fig. 17. The ratio of block numbers used by using template matching for the video sequence *coastguard*.

III. Simulation Results

In this section, the coding performance in terms of mean absolute difference of the target blocks is presented. Three video sequences (*mobile*, *suzie*, and *coastguard*) at QCIF resolution are coded by the proposed method: the fast weighted low-rank matrix approximation-based template matching (weighted LRMA-based template matching). Switching mechanism between the template matching and the block matching is also used, where the value of α is zero in this simulation. It means that only the distortion is considered. If the method is implemented on the HEVC reference software, the developed method shares the same Lagrange multiplier with the RDO-based block mode decision.

Two other methods are simulated as well for comparison. One is the generic low-rank matrix approximation-based template matching with switching mechanism; another is the traditional encoder-side block matching algorithm. The MAD values of the three methods at different quantization parameters (QPs) are shown from Fig. 12 to Fig. 14. From the results, the MAD performance of the proposed weighted LRMA-based template matching with switching mechanism is better than the other two approaches clearly. The ratio of the number of blocks predicted by the proposed method and the number of total blocks at different quantization parameters (QPs) are shown from Fig. 15 to Fig. 17.

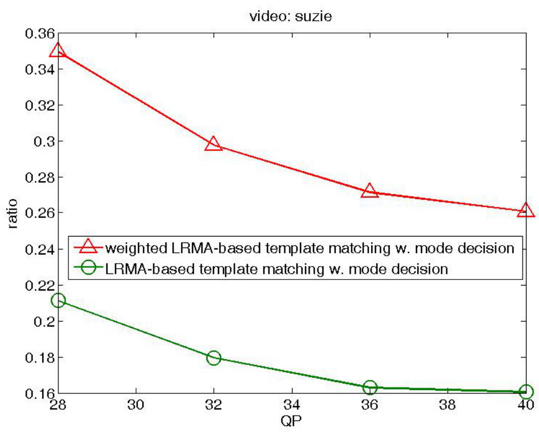

Fig. 16. The ratio of block numbers used by using template matching for the video sequence *suzie*.

For the proposed method, the switching mechanism by the rate-distortion optimization is operated at the encoder side. This one-bit overhead for mode decision is needed. The template-matching approaches do not need the overhead of motion vectors, when compared with the traditional motion compensation predictive coding by using the block matching algorithm. When the QP is smaller, the percentage of choosing the fast weighted LRMA-based template matching is higher, and the MAD reduction is more significant.

IV. Conclusion

The conventional criteria of either block matching or template matching algorithms are used to find a matched predictor. However, the predictors found by such approaches still leave much room for improvement. A fast weighted low-rank matrix approximation approach to deriving decoder-side motion vectors for inter frame video coding is proposed in this paper by identifying dominating block candidates first. The importance of each candidate is determined by using the orthogonal matching pursuit algorithm. The template-matching problem is formulated as the weighted low-rank approximation problem. Through the technique of repeating column vectors, the predictors can be found by using the inexact augmented Lagrange multiplier method in the literature, which is shown to be computationally efficient. The results have shown the advantages of the proposed method by reducing the MAD consistently. As a future work, further investigation has been conducted to see if the mode decision can be made at the decoder side as well, so as to remove the overhead and therefore improve the coding efficiency.

References

[1] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, “Video coding with H.264/AVC: tools, performance, and complexity,” *IEEE Circuits and Systems Magazine*, pp. 7–28, 2004, First Quarter.

[2] G. J. Han, J. Gary, J. R. Ohm, W. J. Han, and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," in *IEEE Transactions on Circuits and Systems for Video Technology*, September 2012, p. 1.

[3] K. Sugimoto, M. Kobayashi, Y. Suzuki, S. Kato, and S. B. Choong, "Inter frame coding with template matching spatio-temporal prediction," in *IEEE International Conference on Image Processing*, October 2004, pp. 24–27.

[4] Y. Suzuki, C. S. Boon, and T. K. Tan, "Inter frame coding with template matching averaging," in *IEEE International Conference on Image Processing*, 16 September – 19 October 2007.

[5] M. Turkan and C. Guillemot, "Sparse approximation with adaptive dictionary for image prediction," in *IEEE International Conference on Image Processing*, Cairo, Egypt, November 2009, pp. 25–28.

[6] J. Wang, Y. Shi, W. Ding, and B. Yin, "A low-rank matrix completion based intra prediction for h.264/AVC," in *Multimedia Signal Processing (MMSP), 2011 IEEE 13th International Workshop*, 17–19 October 2011.

[7] Y. Suzuki, C. S. Boon, and S. Kato, "Block-based reduced resolution inter frame coding with template matching prediction," in *IEEE International Conference on Image Processing*, 8-11 October 2006, pp. 1701–1704.

[8] K. H. Ng, L. M. Po, K. W. Cheung, X. Y. Xu, and K. M. Wong, "A new motion compensation method using superimposed inter-frame signals," in *IEEE International Conference on Speech and Signal Processing (ICASSP)*, 25-30 March 2012, pp. 1213–1216.

[9] S. Milani, "Segmentation-based motion compensation for enhanced video coding," in *IEEE International Conference on Image Processing (ICIP)*, 11-14 September 2011, pp. 1649–1652.

[10] J. Bennett and S. Lanning, "The netflix prize," in *Proceedings of KDD Cup and Workshop*, 2007.

[11] S. Ma, D. Goldfarb, and L. Chen, "Robust principal component analysis?: Recovering low-rank matrices from sparse errors," in *IEEE Sensor Array and Multichannel Signal Processing Workshop (SAM)*, October 2010, pp. 201–204.

[12] H. Ji, C. Liu, Z. Shen, and Y. Xu, "Robust video denoising using low rank matrix completion," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2010, pp. 1791–1798.

[13] Z. Lin, M. Chen, L. Wu, and Y. Ma, "The augmented lagrange multiplier method for exact recovery of a corrupted low-rank matrix," University of Illinois at Urbana-Champaign, Tech. Rep. #UILU-ENG-09-2215, October 2009.

[14] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad, "Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition," in *Proceedings of The 27th Annual Asilomar Conference on Signals, Systems and Computers*, 1–3 November 1993, pp. 40–44.

[15] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *In Magazine Communications of the ACM CACM Homepage*, vol. 55, no. 6, pp. 111–119, June 2012.

[16] D. Bertsekas, *Constrained optimization and lagrange multiplier method*. Academic Press, 1982.

[17] D. Zonoobi, A. A. Kassim, and Y. V. Venkatesh, "Gini index as sparsity measure for signal reconstruction from compressive Samples," *IEEE Journal on Selected Topics in Signal Processing*, pp. 927–932, September 2011.