Fast Block Matching Algorithm in Walsh Hadamard Domain

Ngai Li, Chun-Man Mak, and Wai-Kuen Cham

Department of Electronic Engineering, The Chinese University of Hong Kong

Abstract. A fast block matching algorithm, namely Fast Walsh Search, is proposed for motion estimation in block-based video coding. In our approach, target blocks in current frame and their candidates in reference frame are projected onto Walsh Hadamard domain, allowing early rejection of mismatch candidates to reduce computation requirement. Moreover, we introduce a new method called block pyramid matching that re-uses many previous calculations to further lessen the computation load of our approach. Experimental results show that the proposed algorithm can achieve more accurate motion estimation than the popular three-step-search and diamond search with slight increase in computation requirement only.

1 Introduction

Most video coding standards use motion compensation to reduce temporal redundancy. Motion compensation requires block matching which is to find a matching block in the reference frame that is close to the target block in the current frame. The displacement vector of the matching block is called motion vector, therefore block matching is also called motion estimation. Full search block matching (FSBM) algorithm exhaustively searches through all possible locations in the search window to obtain the matching block that has the least matching error with the target block. However, the computation requirement of FSBM is too high for real-time applications. Fast search algorithms such as three-step search (TSS) [4], four-step search (FSS) [5], new three-step search (NTSS) [6], and diamond search (DS) [7] were developed, which can reduce the computation time significantly at the cost of higher matching error. These algorithms find the minimum error using a gradient-descent approach which implicitly assumes that there is no local minimum.

Recently developed video coding standards such as H.264/AVC [8] use Walsh Hadamard Transform (WHT) to compress DC coefficients. Meanwhile, Hel-Or et al. [1, 2] proposed a real time pattern matching algorithm which works in the Walsh Hadamard (WH) domain. Their matching algorithm first computes a distance using a few WHT coefficients to perform early rejection of mismatch patterns and then focus on a small number of remaining candidates that are more likely to be a correct match of the pattern. Their proposed algorithm reduces computation overheads in WHT by an efficient pruning algorithm in which the intermediate data is effectively exploited.

Motivated by [1, 2, 8], we propose a "Fast Walsh Search" (FWS) that performs block matching in the WH domain. Although it is straightforward to perform motion estimation in spatial domain [4, 5, 6, 7], our proposed algorithm requires only slightly more computation than that of TSS and DS and achieves a more accurate block matching in terms of mean square error (MSE). The high efficiency is because of the pattern matching algorithm suggested by Hel-Or et al. [1, 2] as well as a new matching technique called block pyramid matching (BPM).

This paper is organized as follows. Section 2 introduces the proposed fast motion estimation algorithm in WH domain. Section 3 describes the proposed block pyramid matching. Experimental results and conclusions are given in the Sections 4 and 5 respectively.

2 Fast Block Matching in Walsh Hadamard Domain

2.1 Walsh-Hadamard Transform

WHT BPs contain only ±1 and so the projections of a block of pixels on 2D WH domain require additions and subtractions solely. A particular 2D WHT coefficient of a $k\times k$ block is obtained by projecting the block on the corresponding $k\times k$ WHT BPs where $k=2^n$, $n\in\mathbf{Z}^+$. In the following, we shall represent a $k\times k$ BP by a vector $\mathbf{h}_{(\mathrm{m,n})}$ in $\Re^{k\times k}$ where m and n are the number of zero-crossing in horizontal and vertical direction respectively. The BPs of an 8×8 block are shown in Fig. 1.

In our approach, we follow the same zigzag path as in [1, 2] where the projections on WHT BPs are performed in increasing sequency (the number of zero-crossings along rows and columns) order. In general, the energy of WHT coefficients of an image decreases along the zigzag order [2, 3]; therefore the projections onto the first few WHT BPs capture a large proportion of information of an image. Hel-Or et al. has utilized this energy packing property in his fast pattern matching algorithm [1,2] which is a pruning algorithm that re-uses many intermediate results to further reduce the computation requirements of the WHT. In this paper, we adopt the same idea to develop a block matching algorithm in WH domain.

2.2 Proposed Block Matching System

Motion vector estimation is an important step in video compression. Motion vectors can be estimated by block matching algorithms that minimize a measure of matching error. Suppose the matching error between the target block at position (x,y) in the current frame F_c, and the reference block at position $(x+u,\ y+v)$ in the reference frame F_R is $E(u,v)$. The motion vector $(\hat{u},\hat{v})$ is defined as:

$$(\hat{u},\hat{v}) = \arg\min_{(u,v)\in S} E(u,v) \tag{1}$$

where $S=\{(u,v)|-R\le u,v\le R\}$ is the *candidate set*, and R is the maximum search distance. In most cases, sum-of-absolute difference (SAD) between the target block and the reference block as given in (2) is used as the matching error because of its simplicity.

$$\Phi(u,v)=\sum_{i=0}^{k-1}\sum_{j=0}^{k-1}\left|F_C(x+i,y+j)-F_R(x+u+i,y+v+j)\right| \tag{2}$$

where k is the block size.

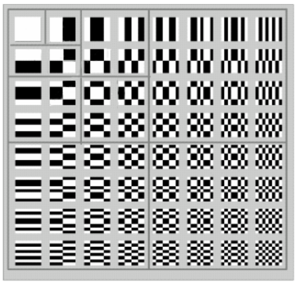


Fig. 1. BPs [1, 2] of 8×8 WHT

In this paper, we propose to perform block matching in WH domain and a *partial absolute distance* (PAD) $\Phi_p(u,v;q)$ is used as the matching error to reduce the computation requirement where q is the number of projections onto the WHT BPs. PAD may be regarded as an approximation of the SAD but requires significantly less computations. We shall show that block matching using PAD can find matching blocks of mean square error very close to that of SAD.

Suppose a $k\times k$ target block at (x,y) in current frame F_c is matched with a reference block of the same dimension at $(x+u,y+v)$ in its search area in reference frame F_R . The target block and the reference block are represented by vectors $\mathbf{b}_\mathrm{T}$ at (x,y) and $\mathbf{b}_\mathrm{R}$ at $(x+u,y+v)$ respectively in space $\Re^{k\times k}$. A difference vector $\mathbf{d}$ between $\mathbf{b}_\mathrm{T}$ and $\mathbf{b}_\mathrm{R}$ is defined as

$$\mathbf{d} = \mathbf{b}_\mathbf{T} - \mathbf{b}_\mathbf{R}\,. \tag{3}$$

The SAD d between the reference block and the target block is shown in (4) where $\|.\|_p$ is the p-norm of a vector.

$$d = \|\mathbf{d}\|_1 = \|\mathbf{b}_\mathbf{T} - \mathbf{b}_\mathbf{R}\|_1 \tag{4}$$

Let S_q be a set of index (m,n), and each of them represents the number of horizontal and vertical zero-crossing of the first q BPs along the zigzag path. Projecting $\mathbf{b}_\mathrm{T}$ and $\mathbf{b}_\mathrm{R}$ onto BPs with indices in S_q, we get sets of $c_T(x,y;m,n;k)$ and $c_R(x+u,y+v;m,n;k)$ respectively. The $\Phi_p(u,v;q)$ between $\mathbf{b}_\mathrm{T}$ and $\mathbf{b}_\mathrm{R}$ is then defined by projecting $\mathbf{d}$ onto q WHT BPs as shown in (5).

If an additional WHT BP $\mathbf{h}$ is added into S_q, then PAD can be refined iteratively using (6) and becomes closer to SAD. Those reference blocks with PAD greater than a given threshold T_Φ will be rejected, and we search for the best match among the remaining candidates only.

$$\begin{aligned} d &\geq \Phi_p(u,v;q) \\ &= \sum_{(m,n)\in S_q} \frac{\left|\mathbf{h}^T_{(m,n)}\mathbf{d}\right|}{\left|\mathbf{h}_{(m,n)}\right|} = \sum_{(m,n)\in S_q} \frac{\left|\mathbf{h}^T_{(m,n)}\mathbf{b}_T - \mathbf{h}^T_{(m,n)}\mathbf{b}_R\right|}{\left|\mathbf{h}_{(m,n)}\right|} \\ &= \sum_{(m,n)\in S_q} \frac{\left|c_T(x,y;m,n;k) - c_R(x+u,y+v;m,n;k)\right|}{\left|\mathbf{h}_{(m,n)}\right|} \end{aligned} \tag{5}$$

$$\Phi_p(u,v;q+1) = \Phi_p(u,v;q) + \frac{\left|\mathbf{h}^T\mathbf{b}_T - \mathbf{h}^T\mathbf{b}_R\right|}{|\mathbf{h}|} \tag{6}$$

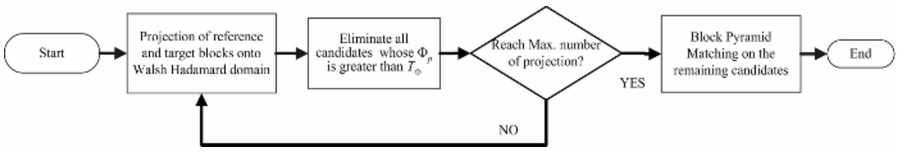

Start
Projection of reference and target blocks onto Walsh Hadamard domain
Eliminate all candidates whose Φ_p is greater than T_Φ
Reach Max. number of projection?
YES
NO
Block Pyramid Matching on the remaining candidates
End

Fig. 2. Flowchart of the proposed FWS system

The proposed motion estimation algorithm can perform fast block matching because of two reasons. Firstly, the low sequency order WHT BPs are highly probable to be parallel to the difference vector **d** so that the first few projections can acquire most of the distance between the targets block $\mathbf{b}_T$ and their candidates block $\mathbf{b}_R$. The second reason is the fast pruning algorithm of WHT, which computes the projections of the candidates in reference frames onto various BPs efficiently. We use a recursive structure of Walsh Hadamard tree [1, 2] in which the calculations applied to one candidate in reference frame or one BP projection are exploited when the projections of candidate or the projections onto another BP are computed.

The flowchart of the algorithm is shown in Fig. 2. To begin with, our algorithm computes PAD of each candidate in reference frame according to the corresponding WHT coefficients of the target blocks and reference blocks. If the PAD of a candidate is greater than a given threshold T_Φ, the location will be rejected. The remaining candidates in the reference frame are projected onto the higher sequency order WHT BPs. The PAD comparison repeats until a predefined number of projections is reached because the block matching in the WH domain is efficient only when the number of projections is small. We found that efficient block matching completely in WH domain is still possible by using a technique called pyramid block matching, which will be explained in the next section. In our implementation, only two projections are used to find the PAD. More projections require more computation but do not reduce MSE significantly.

The computation of PAD includes the transformation of frames, and the accumulation of absolute differences of WHT coefficients. Transforming reference and target frames requires about 8 operations, which include additions, subtractions, and absolute, per pixel for 2 projections 8×8 block [2, 3]. Total number of operations per pixel required to find PAD of the first and the second projections for one block is

$$N_{o,PAD} = \frac{1}{k^2}(2R+1)^2(2+3P_1) \tag{7}$$

where P_1 is the percentage of candidates remains after first projection.

3 Block Pyramid Matching

Hel-Or et al. suggest that the best matching position is the one with the minimum sum-of-squared distances (SSD) among the remaining candidates. However, the computation requirement of SSD is heavy, and we propose to use a block pyramid matching scheme to find a distance approximating the SAD such that computation can be reduced while not affecting the MSE performance much. In the first stage of BPM, each k×k block in reference frame and current frame is decomposed into four non-overlapping $k/2$×$k/2$ sub-blocks, and the projection of each k×k block onto $\mathbf{h}_{(0,0)}$

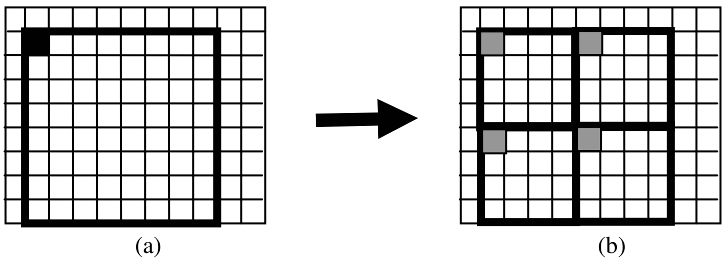


Fig. 3. Illustrate the locality relationship between (a) the WHT coefficient of the *k*×*k* block and (b) that of its corresponding *k*/2×*k*/2 sub-blocks

(shaded box in Fig. 3a) is expressed as the sum of projections of the corresponding sub-blocks onto $\mathbf{h}_{(0,0)}$ (shaded boxes in Fig. 3b). Therefore, $\Phi_p(u,v;1)$ can be formulated as (8) when k=8.

$$\Phi_p(u,v;1) = \frac{|S_{T1} - S_{R1}|}{|\mathbf{h}_{(0,0)}|} \tag{8}$$

where

$$S_{T1} = \sum_{i=0,4}\sum_{j=0,4} c_T(x',y';\frac{k}{2})$$
$$S_{R1} = \sum_{i=0,4}\sum_{j=0,4} c_R(x'+u,y'+v;\frac{k}{2})$$
$$x' = x + i$$
$$y' = y + j$$

The relationship of the coefficient of blocks and their sub-blocks is illustrated in Fig. 3. We define the first level BPM estimation based on the projection onto $\mathbf{h}_{(0,0)}$ as $E_1(0,0)$, and it is shown in (9).

$$E_1(0,0) = \frac{\sum_{i=0,4}\sum_{j=0,4}\left|c_T(x',y';\frac{k}{2}) - c_R(x'+u,y'+v;\frac{k}{2})\right|}{|\mathbf{h}_{(0,0)}|} \tag{9}$$

Because of Triangular Inequality in (10),

$$\left|\sum_{j=1}^{M} a_j\right| \le \sum_{j=1}^{M} |a_j| \tag{10}$$

where $a_j \in \mathbf{R}$ and $M \in \mathbf{Z}^+$, $E_1(0,0)$ is closer to the SAD than $\Phi_p(u,v;1)$, but is still smaller than or equal to the SAD. In other words, $E_1(0,0)$ is a more accurate estimation of the SAD than $\Phi_p(u,v;1)$, i.e.

$$d \ge E_1(0,0) \ge \Phi_p(u,v;1) \tag{11}$$

It should be noted that the projection of $k/2\times k/2$ sub-blocks onto $\mathbf{h}_{(0,0)}$ are the intermediate data in the calculation of the WHT coefficient of $k\times k$ blocks using the recursive WH tree [1,2] . As a result, evaluating $E_1(0,0)$ requires much fewer computations than that of SAD, and contribute to the success of our fast block matching algorithm.

In the second stage of BPM, each $k/2{\times}k/2$ sub-block is further decomposed into four $k/4{\times}k/4$ sub-blocks. The projection of each $k/2{\times}k/2$ sub-block onto $\mathbf{h}_{(0,0)}$ (shaded box in Fig. 4a) can be expressed as the sum of four projections of the corresponding sub-blocks onto $\mathbf{h}_{(0,0)}$ (shaded boxes in Fig. 4b), which are available when we calculate the WHT coefficient of $k{\times}k$ blocks. In this stage, the $k{\times}k$ block is divided into sixteen $k/4 \times k/4$ sub-blocks. The first level BPM estimation $E_1(0,0)$ can then be expressed as (12).

$$E_1(0,0) = \sum_{i=0,4}\sum_{j=0,4} \left| S_{T2} - S_{R2} \right| \tag{12}$$

where

$$S_{T2} = \sum_{m=0,2}\sum_{n=0,2} c_T(x'',y'';\frac{k}{4})$$

$$S_{R2} = \sum_{m=0,2}\sum_{n=0,2} c_R(x''+u,y''+v;\frac{k}{4})$$

$x'' = x+i+m$, and $y'' = y+j+n$.

Similar to the first level BPM estimation, we define the second level BPM estimation $E_2(0,0)$ based on the projection of $\mathbf{h}_{(0,0)}$ as

$$E_2(0,0) = \frac{\sum_{i=0,4}\sum_{j=0,4}\sum_{m=0,2}\sum_{n=0,2} \left| c_T(x'',y'';\frac{k}{4}) - c_R(x''+u,y''+v;\frac{k}{4}) \right|}{\left| \mathbf{h}_{(0,0)} \right|}. \tag{13}$$

Because of Triangular Inequality, $E_2(0,0)$ is more accurate to approximate d than $E_1(0,0)$ as shown in (14). Theoretically blocks can be decomposed further until the block size becomes one. In that case the BPM estimation becomes the SAD itself.

$$d \geq E_2(0,0) \geq E_1(0,0) \geq \Phi_p(u,v;1) \tag{14}$$

In our previous discussion, we concern with the BPM based on the projection onto $\mathbf{h}_{(0,0)}$ only. It can be shown that $E_1(0,0)$ is also the first level BPM based on the

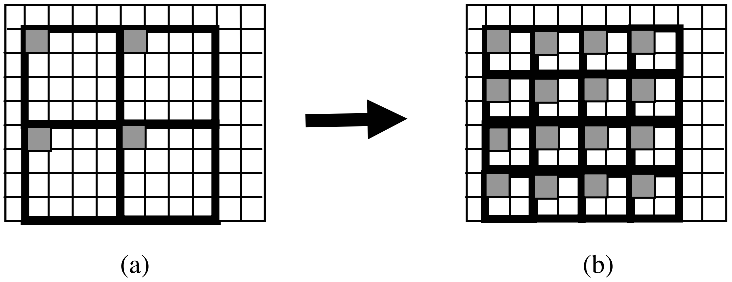


Fig. 4. Illustrate the locality relationship between (a) the WHT coefficients of the $k/2 \times k/2$ block and (b) that of its corresponding $k/4 \times k/4$ sub-blocks

projection onto $\mathbf{h}_{(a,b)}$ where $0 \le a,b \le 1$. Similarly, $E_2(0,0)$ is the second level BPM based on the projection onto $\mathbf{h}_{(c,d)}$ where $0 \le c,d \le 3$, i.e.

$$E_1(a,b) = E_1(0,0) \tag{15}$$

$$E_2(c,d) = E_2(0,0) \tag{16}$$

where $0 \le a,b \le 1$ and $0 \le c,d \le 3$.

Therefore, when we compare $E_1(0,0)$ or $E_2(0,0)$ of the target block and the reference blocks, we have already compared their $E_1(a,b)$ or $E_2(c,d)$ where $0 \le a,b \le 1$ and $0 \le c,d \le 3$ respectively. In other words, we have used the information from the projections onto higher sequency order WHT BPs to get more precise similarity evaluation when we compare the corresponding $E_1(0,0)$ and $E_2(0,0)$ of the target block and the reference blocks.

In the proposed algorithm, after rejecting candidates in reference frame using PAD, the best K_1 % of the remaining candidates, i.e. those with the least PAD, will go through the first level BPM in which the $E_1(0,0)$ difference between the target block and the remaining candidates are computed. Then, the best K_2% candidates after first level BPM will be further examined by evaluating their $E_2(0,0)$ difference. The candidate with smallest $E_2(0,0)$ difference between the target block is elected as the best match of the target block, and will be regarded as the location pointed by the corresponding motion vector. Assuming the maximum allowed candidates are used for first and second stage of BPM, the number of operations required to find the best match per pixel is

$$N_{o,BPM} = \frac{1}{k^2}(2R+1)^2\left[K_1\left(3\left(\frac{k}{4}\right)^2 - 1\right) + K_2\left(3\left(\frac{k}{2}\right)^2 - 1\right)\right]. \tag{17}$$

4 Experimental Results

We applied the FWS to 80 frames of three standard sequences: Football, Foreman, and Stefan with k=8 and R=16. For each candidate in reference frame, the maximum number of projections allowed is two and the remaining candidates will go through BPM. The threshold T_Φ is 10. For BPM, K_1=10% and K_2=5%.

4.1 Computation Requirement

Finding the SAD of a $k \times k$ block requires k^2 subtractions, k^2 absolute operations, and k^2-1 additions; therefore, the total number of operations is $3k^2-1$. In FSBM, the number of candidate for each block is $(2R+1)^2$, and each frame with A number of pixels has number of blocks A/k^2. Then the total number of operations $N_{o,FS}$ required per pixel is

$$N_{o,FS} = \frac{1}{k^2}(2R+1)^2(3k^2-1). \tag{18}$$

With k=8 and R=16, $N_{o,FS}$ = 3245. On the other hand, TSS has only $8\log_2 R+1$ candidates for each block, therefore, the total number of operations per pixel, $N_{o,TSS}$, becomes

$$N_{o,TSS} = \frac{1}{k^2}(8\log_2 R + 1)(3k^2-1). \tag{19}$$

In our experiment, $N_{o,TSS} = 98$. Since DS has no fixed number of search points, its computation varies for different videos. According to [7], DS has a computation of about 80% of TSS. Similar to DS, the computation of FWS also varies for different videos since the numbers of remaining candidates after each projection are different. Experimental results show that around 40% of candidates remain after first projection, and 25% remains after second projection. Table 1 shows the total number of addition, subtraction and absolute operations required per pixel for FS, TSS, and FWS. The computation of the proposed FWS includes WHT of frames, PAD, and BPM computations. FWS usually requires about 20% more computation than TSS. Because the intermediate data in the recursive WH tree are reused, more memory is needed compared to FS and TSS.

4.2 MSE Performance

Experimental results show that two projections are enough and additional projections do not reduce MSE significantly, but will increase the computation time. About 75% of candidates will be eliminated after two projections. Table 2 shows the average MSE over 80 frames of the three sequences using different algorithms and Fig. 5 shows the MSE of the each frame. The performance of FWS in terms of MSE is very close to FS, but the computation required is only a little bit more than TSS. TSS and DS, while much faster than FS, produce MSE which are significantly larger than FS and FWS.

Replacing SAD by BPM after two projections can significantly reduce computations. The resultants MSE, however, are not affected much. Table 3 shows the increase in MSE when SAD is replaced by BPM. On average, the MSE is increased by merely 5%.

Table 1. Operations per pixel needed for different search methods

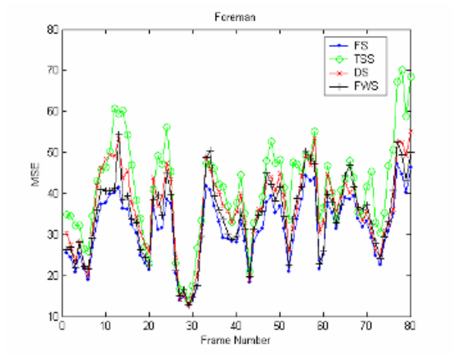
Sequence	FS	TSS	FWS
Foreman	3245	98	118
Football	3245	98	126
Stefan	3245	98	123

Table 2. Average mean-squared-error of 80 frames

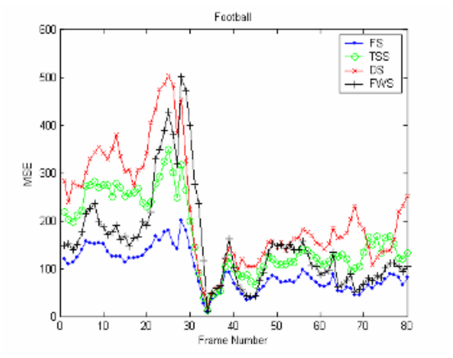
Sequence	FS	TSS	DS	FWS
Foreman	31.5	41.1	36.4	34.5
Football	94.4	167.0	220.0	155.4
Stefan	142.5	341.1	308.0	181.3

Table 3. MSE comparison of SAD and BPM

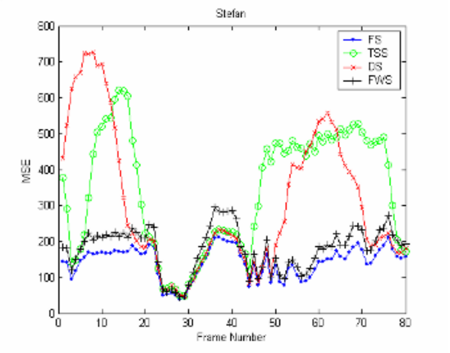
Sequence	MSE		
	SAD	BPM	MSE
Foreman	32.8	34.5	+5.4%
Football	148.8	155.4	+4.4%
Stefan	169.2	181.3	+7.1%



(a)



(b)



(c)

Fig. 5. MSE plots for sequence (a) Foreman (b) Football (c) Stefan

5 Conclusions

A fast block matching method, FWS, which is based on a pattern matching algorithm in Walsh Hadamard domain, is proposed in this paper. The computation requirement is similar to the three-step-search, but the accuracy is comparable with the full-search method. Efficient projection scheme is utilized for fast WHT. Furthermore, we exploit the intermediate results in WHT calculation to reject candidate blocks that are unlikely to be a good match. Both measures significantly reduce computations in the block matching process. Experimental results show that the performance of FWS in terms of MSE is very close to that produced by full search algorithm.

References

[1] Y. Hel-Or; H. Hel-Or; "Real time pattern matching using projection kernels", Proc. of Ninth IEEE International Conference on Computer Vision, Vol. 1, pp. 1486 – 1493, Oct. 2003.

[2] Y. Hel-Or; H. Hel-Or; "Real time pattern matching using projection kernels", IEEE Trans. Pattern Analysis and Machine Intelligence, Vol. 27, No. 9, Sept 2005.

[3] W. K. Cham; R. J. Clarke; "Application of the principle of dyadic symmetry to the generation of orthogonal transforms", IEE Proc. F, Commun., Radar & Signal Process., Vol. 133, no.3, pp.264-270, June 1986.

[4] T. Koga; K. Iinuma; A. Hirano; Y. Iijima; T. Ishiguro; "Motion compensated interframe coding for video conferencing," in Proc. Nat. Telecommun. Conf., New Orleans, LA, Nov. 29-Dec. 3 1981, pp. G5.3.1-5.3.5.

[5] Lai-Man Po; Wing-Chung Ma; "A novel four-step search algorithm for fast block motion estimation," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 6, No. 3, June 1996, pp. 313 - 317.

[6] Reoxiang Li; Bing Zeng; Liou, M.L.; "A new three-step search algorithm for block motion estimation," IEEE Transactions on Circuits and Systems for Video Technology, Vol. 4, No. 4, Aug. 1994, pp.438 - 442.

[7] Shan Zhu; Kai-Kuang Ma; "A new diamond search algorithm for fast block-matching motion estimation," IEEE Transactions on Image Processing, Vol. 9, No. 2, Feb. 2000, pp. 287 - 290.

[8] T. Wiegand; G. J. Sullivan; G. Bjontegaard; A. Luthra, "Overview of the H.264/AVC video coding standard," IEEE Trans. Circuits Syst. Video Technol, Vol. 13, pp. 560-576, July 2003.