Performance Evaluation of Motion Estimation in DCT Transform Domain

PETRESCU CATALIN-DUMITRU, STEFANOIU DAN, LUPU CIPRIAN
Department of Automatic Systems and Computer Science
"Politehnica" University of Bucharest
Splaiul Independentei no. 313
ROMANIA
catalin@indinf.pub.ro, danny@indinf.pub.ro, cip@indinf.pub.ro

Abstract: - Motion estimation is one of the most important steps in video compression algorithms. It reduces temporal redundancy present in frame sequences and allows a better compression of video material. Most of the actual video compression algorithms use "block matching" methods which operate on the bitmap form of the frames. This paper presents a method for computing the values of DCT coefficients of a block of pixels positioned on certain coordinates over four adjacent blocks using only the DCT coefficients of these four blocks. Performance of this method is analyzed for both integer and non-integer displacements. Also, an equivalent of the full-search algorithm translated in 2D-DCT domain is presented.

Key-Words: - motion estimation, block matching, discrete cosine transform, video compression, match function

1 Introduction

Video sequences are characterized by a high level of information redundancy. Video compression algorithms use this feature of the video material to achieve high levels of data reduction. In fact, video compression is based on the similarity between successive frames.

Most of the actual video compression algorithms encode each frame in two steps [1]. In the first step (motion estimation), they try to create an as good as possible version of the current frame from other frames which were already encoded. For doing this, the current frame is divided in square blocks of *NxN* pixels. Then, for a maximum motion displacement of *R* pixels per frame, each block of pixels is matched against a corresponding block at the same coordinates but in the previous frames, within the square window of width/height *N+2R*. The best match is selected by testing a matching function like mean squared error (MSE) or mean absolute difference (MAD). In this step only the information about the position of the best matched blocks from previous frames (motion vectors) are encoded.

The searching can be performed for each possible position in the search area (Full Search algorithm). In this case the best match is found but the computing effort involved is huge. For this reason, some strategies for reducing the number or search steps were proposed like Three Step Search - TSS [2], Two Dimensional Logarithmic Search -TDL [3], Hierarchical Search – HS [4] [5], Adaptive Search [6], Fast and Robust Search [7], etc..

Because real motion in video frames is not carried out over integer pixels, motion estimation is performed at sub-pixel level [10]. This kind of search involves interpolation of pixel values and searching over an enhanced resolution images which increase the complexity of the procedure.

In the second step (motion compensation) the difference between the reconstructed frame created in the first step and the actual frame is encoded using a still image compression algorithm (usually based on discrete cosine transform, quantization and entropy coding).

Usually this search is performed on the bitmap form of the frames. Many video formats like MJPEG [8], DV25, DV50 [9], use Discrete Cosine Transform (DCT) for compressing frames.

In these cases it would be better to do the searching and matching tests directly in the transformed domain.

There are some proposed methods for motion estimation based on pseudo phase techniques [11] or phase correlation [12]. Other approaches focus on refinement of the estimations at sub-pixel level using DCT coefficients [13].

This paper presents a method for computing the values of DCT coefficients of a block of pixels positioned on certain coordinates over four adjacent blocks using only the DCT coefficients of these four blocks.

The presentation is organized as follows. The problem to solve is stated within the next section. The proposed solution for 1D and 2D cases and simulation results are succinctly described in section 3. Section 4 contains some concluding remarks and a references list completes the article.

2 Problem Formulation

The general case in a search step is the one presented in the Figure 1. We have a block from the current frame and we want to compare it with a candidate block **X** which stand over four adjacent blocks (**A**, **B**, **C** and **D**). These four main blocks are located in a previous encoded frame (reference frame). The candidate block **X** is positioned at (dx,dy) pixels relative to the bottom-left corner of the block **D**.

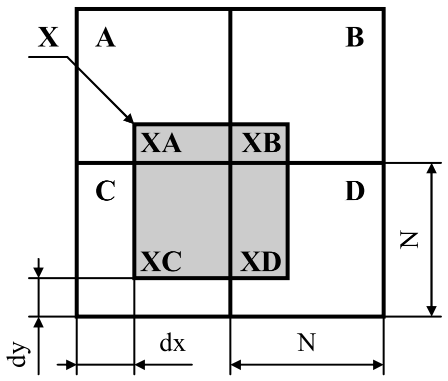

Fig.1 Position of the candidate block

The problem is to compute the DCT coefficients of the candidate block X directly from the DCT coefficients of the four adjacent blocks.

The candidate block **X** is a sum of 4 partial blocks (**XA**, **XB**, **XC** and **XD**), each of them containing parts of main blocks **A**, **B**, **C** respectively **D** like in the following figure:

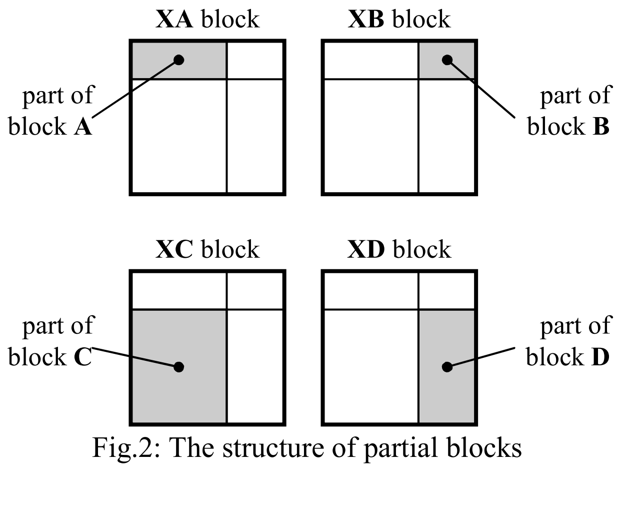

Fig.2: The structure of partial blocks

Each of the four partial blocks can be obtained from the main blocks by shifting the image inside them.

For example, the **XA** component can be obtained from the main block **A** by shifting the image inside it dx pixels to the left and N-dy pixel up like in the Figure 3 below:

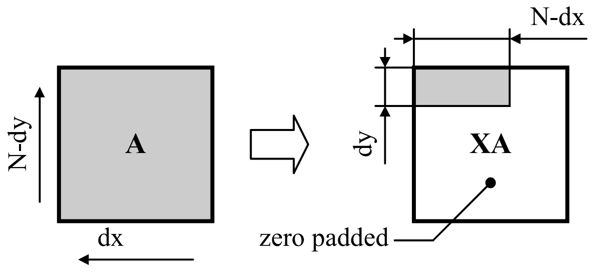

Fig.3: Obtaining **XA** from **A**

The shifts necessary for obtaining all the components are summarized in the next table:

Component	Main	Horizontal		Vertical	
		Dir.	Amount	Dir.	Amount
XA	A	Left	dx	Up	N-dy
XB	B	Right	N-dx	Up	N-dy
XC	C	Left	dx	Down	dy
XD	D	Right	N-dx	Down	dy

Table 1: Generation of candidate block components

Because Discrete Cosine Transform is linear, the transform applied to a sum of elements is equal to the sum of the transform applied to each element.

For this reason, the 2D-DCT transform of the candidate block T_x can be determined from the 2D-DCT transform of its components, by using the following equation:

$$\begin{aligned} T_X &= DCT(XA+XB+XC+XD) \\ &= DCT(XA)+DCT(XB)+ \\ &\quad + DCT(XC)+DCT(XD) \\ &= T_{XA}+T_{XB}+T_{XC}+T_{XD} \end{aligned} \tag{1}$$

It can be noticed that 2D-DCT transform of the shifted components **XA**, **XB**, **XC** and **XD** are needed.

Therefore, the problem is to determine a method for compute 2D-DCT transform of a shifted matrix directly from its own 2D-DCT coefficients and amounts of vertical and horizontal shifts.

3 Problem Solution

The first step to find the method to determine the DCT coefficients of a shifted matrix is to analyze the 1D case of this problem. Next, the 2D case will be derived from the results obtained in the fist step.

3.1 The one-dimensional case

Suppose that the values of an initial vector y are moved down by s positions. The result is a shifted vector ys. The upper-most s positions are filled with zeros like in the Figure 4:

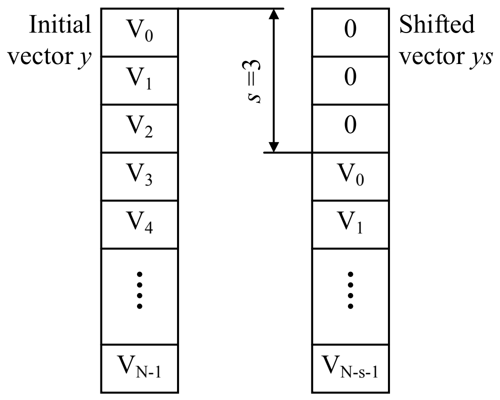

Fig.4: Initial and shifted vector

The elements of discrete cosine transform T_y applied to the initial vector y can be computed using the following equation [14]:

$$T_{y\,i} = \begin{cases} \sqrt{\frac{1}{N}} \sum_{j=0}^{N-1} y_j \cos\left(\frac{i(2j+1)\pi}{2N}\right), & i=0 \\ \sqrt{\frac{2}{N}} \sum_{j=0}^{N-1} y_j \cos\left(\frac{i(2j+1)\pi}{2N}\right), & i \neq 0 \end{cases} \tag{2}$$

This equation can be written in a matrix form like the following:

$$T_y = Q\,y \tag{3}$$

where matrix Q is defined by:

$$q_{i,j} = \begin{cases} \sqrt{\frac{1}{N}} \cos\left(\frac{i(2j+1)\pi}{2N}\right), & i=0 \\ \sqrt{\frac{2}{N}} \cos\left(\frac{i(2j+1)\pi}{2N}\right), & i \neq 0 \end{cases} \tag{4}$$

For integer values of the displacement s, the shifted vector ys can be obtained from intial vector y by:

$$ys = S(s)\,y \tag{5}$$

where the $S(s)$ matrix have 1 on the s-the subdiagonal and 0 in rest.

Initial vector y can be computed from its DCT transform T_{ys} using:

$$y = Q^T\,T_y \tag{6}$$

DCT coefficients of the shifted vector (T_{ys}) can be obtained from the DCT coefficients of the initial vector (T_y), by using:

$$\begin{aligned} T_{ys} &= Q\,y_s = Q\,S(s)\,y = Q\,S(s)\,Q^T T_y \\ &= D(s)\,T_y \end{aligned} \tag{7}$$

Analyzing the displacement matrix $D(s)$ for all possible integer values from 0 to 7, we found that its elements can be described by the next formula:

$$d_{i,j}(s) = \begin{cases} \left(1-\frac{s}{N}\right)\cos\left(\frac{\pi is}{N}\right) + \gamma_i \sin\left(\frac{\pi is}{N}\right), i=j \\ \alpha_{i,j} \sin\left(\frac{\pi is}{N}\right) + \beta_{i,j} \sin\left(\frac{\pi js}{N}\right), \quad i \neq j \end{cases} \tag{8}$$

Although the values of constants $\alpha_{i,j}$, $\beta_{i,j}$ and γ_i can be evaluated in closed form, the manipulations are rather difficult. Therefore, the coefficients can be obtained in a simpler way, with the help of Least Squares Method (LSM) [15]. The following algorithm leads to the desired result:

```
1. For each (i,j) with $i = 0:N-1$ and $j = 0:N-1$
   1.1. For $s = 0:N-1$ (integer values)
        1.1.1. Generate initial vector $y$ using:
               $y_k = \cos\left(\frac{j(2k+1)\pi}{2N}\right),\ k = 0{:}7$
        1.1.2. Generate shifted vector $yd$ with:
               $yd_k = \begin{cases} 0, & k<s \\ y_{k-s}, & k \geq s \end{cases}$
        1.1.3. Compute 1D-DCT of the two vectors:
               $T_y = Q\,y$
               $T_{yd} = Q\,yd$
        1.1.4. Determine the value of $d_{i,j}(s)$ as:
               $d_{i,j}(s) = \frac{T_{y\ i,j}}{T_{yd\ i,j}}$
   1.2. If $i = j$, then
        1.2.1. Solve the superimposed system
               below, by using LSM:
               $\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_{N-1} \end{bmatrix} \gamma_i$
```

where:

$$b_k = d_{i,j}(k) - \cos\left(\frac{\pi\, ik}{N}\right)$$

$$a_k = \sin\left(\frac{\pi\, ik}{N}\right)$$

else

1.2.2. Solve the superimposed system below, by using LSM:

$$\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{N-1} \end{bmatrix} = \begin{bmatrix} a_{0,0} & a_{0.1} \\ a_{1,0} & a_{1,1} \\ \vdots & \vdots \\ a_{N-1,0} & a_{N-1,0} \end{bmatrix} \begin{bmatrix} \alpha_{i,j} \\ \beta_{i,j} \end{bmatrix}$$

where:

$$b_k = d_{i,j}(k)$$

$$a_{k,0} = \sin\left(\frac{\pi\, ik}{N}\right)$$

$$a_{k,1} = \sin\left(\frac{\pi\, jk}{N}\right)$$

For example, the values of these constants were determined for the case of N=8 (usual size in image/video compression applications):

$$\alpha = \begin{bmatrix} 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.4531 & 0.0000 & 0.2079 & 0.0749 & 0.0373 & 0.0207 & 0.0114 & 0.0051 \\ -0.2310 & -0.4329 & 0.0000 & 0.2452 & 0.0957 & 0.0488 & 0.0259 & 0.0114 \\ -0.1591 & -0.2517 & -0.3955 & 0.0000 & 0.2566 & 0.1008 & 0.0488 & 0.0207 \\ -0.1250 & -0.1877 & -0.2310 & -0.3841 & 0.0000 & 0.2566 & 0.0957 & 0.0373 \\ -0.1063 & -0.1560 & -0.1762 & -0.2258 & -0.3841 & 0.0000 & 0.2452 & 0.0749 \\ -0.0957 & -0.1389 & -0.1509 & -0.1762 & -0.2310 & -0.3955 & 0.0000 & 0.2079 \\ -0.0901 & -0.1301 & -0.1389 & -0.1560 & -0.1877 & -0.2517 & -0.4329 & 0.0000 \end{bmatrix}$$

$$\beta = \begin{bmatrix} 0.0000 & 0.4531 & -0.2310 & 0.1591 & -0.1250 & 0.1063 & -0.0957 & 0.0901 \\ 0.0000 & 0.0000 & 0.4329 & -0.2517 & 0.1877 & -0.1560 & 0.1389 & -0.1301 \\ 0.0000 & -0.2079 & 0.0000 & 0.3955 & -0.2310 & 0.1762 & -0.1509 & 0.1389 \\ 0.0000 & 0.0749 & -0.2452 & 0.0000 & 0.3841 & -0.2258 & 0.1762 & -0.1560 \\ 0.0000 & -0.0373 & 0.0957 & -0.2566 & 0.0000 & 0.3841 & -0.2310 & 0.1877 \\ 0.0000 & 0.0207 & -0.0488 & 0.1008 & -0.2566 & 0.0000 & 0.3955 & -0.2517 \\ 0.0000 & -0.0114 & 0.0259 & -0.0488 & 0.0957 & -0.2452 & 0.0000 & 0.4329 \\ 0.0000 & 0.0051 & -0.0114 & 0.0207 & -0.0373 & 0.0749 & -0.2079 & 0.0000 \end{bmatrix}$$

$$\gamma = \begin{bmatrix} 0.0000 \\ -0.3266 \\ -0.1768 \\ -0.1353 \\ -0.1250 \\ -0.1353 \\ -0.1768 \\ -0.3266 \end{bmatrix}$$

For this case (N=8), the maximum error in approximation of the real $d_{i,j}(s)$ coefficients was about 7.4e-16 (using double precision arithmetic).

The matrix $D(s)$ is used only for down-shift case. For the up-shift ($s<0$) another matrix ($U(s)$) must be used. Elements of the $U(s)$ matrix are defined by:

$$u_{i,j}(s) = -1^{i+j}\, d_{i,j}(s) \tag{9}$$

3.2 The bi-dimensional case

Assume that the values of an initial matrix M are moved down by dy positions and right by dx positions. The result is a shifted matrix Ms like in the Figure 5 below:

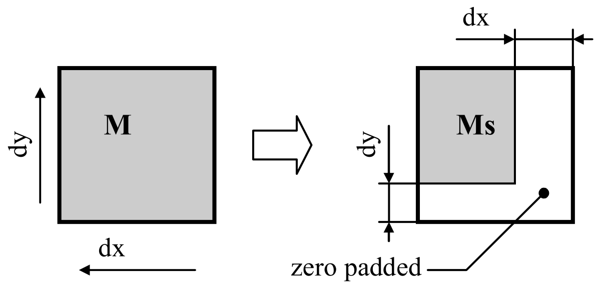

Fig.5: Initial and shifted matrix

The 2D discrete cosine transform applied to the initial matrix is:

$$T_M = Q\,M\,Q^T \tag{10}$$

The 2D-DCT coefficients of shifted matrix (T_{Ms}) can be obtained from the 2D-DCT coefficients of the initial matrix (T_M) by using:

$$T_{Ms}(dx,dy) = V(dy)\, T_M\, H(dx) \tag{11}$$

The $V(dy)$ matrix corresponds to the vertical shift and the $H(dx)$ matrix to the horizontal. These two matrices are defined by:

$$V(dy) = \begin{cases} D(dy)\,, & dy \ge 0 \\ U(-dy)\,, & dy < 0 \end{cases} \tag{12}$$

and

$$H(dx) = \begin{cases} D^T(dx)\,, & dx \ge 0 \\ U^T(-dx)\,, & dx < 0 \end{cases} \tag{13}$$

3.3 Determination of 2D-DCT coefficients of the candidate block

Now a method for determination of 2D-DCT coefficients of the candidate block **X** can be designed (see again Figure 1).

From equation (1), it follows:

$$T_X = T_{XA} + T_{XB} + T_{XC} + T_{XD} \tag{14}$$

Matrices XA, XB, XC and XD are shifted versions of matrices A, B, C respectively D. The amounts of shift on horizontal and vertical directions needed for each matrix are given in Table 1.

Using this information and equation (11), the four components of T_X can be determined as follows:

$$\begin{aligned} T_{XA} &= V(dy-N)\, T_A\, H(-dx) \\ T_{XB} &= V(dy-N)\, T_B\, H(N-dx) \\ T_{XC} &= V(dy)\, T_C\, H(-dx) \\ T_{XD} &= V(dy)\, T_D\, H(N-dx) \end{aligned} \tag{15}$$

It can be noticed that we need only the 2D-DCT transforms of the four initial blocks (T_{XA}, T_{XB}, T_{XC} and T_{XD}) and the values of displacements on the two directions dx and dy.

3.4 Tests

The accuracy of the proposed method was evaluated using several tests.

Generation of DCT coefficients of a shifted vector from DCT coefficients of the initial vector described by equations (7) and (8) do not require any accuracy evaluations for integer values of the displacement. Its accuracy was already tested when $\alpha_{i,j}$, $\beta_{i,j}$ and γ_i coefficients were determined using the Least Squares Method.

The first test was performed to evaluate accuracy of determination for 2D-DCT coefficients of the candidate block described by equations (14) and (15).

For this test, the next image was considered (size 16x16 pixels):

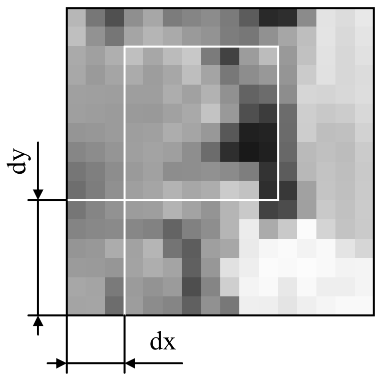

Fig.6: The test image

This image is a part of a larger size real image (luminance component) and pixel values are integer numbers between 0 and 255.

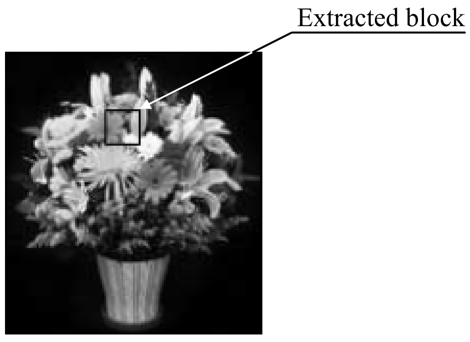

Fig.7: Source image for the test block

The 2D-DCT coefficients of block with size of 8x8 located at (dx,dy) pixels from the bottom-left corner of the image were computed by the following methods:

- applying 2D-DC transform over the image block extracted directly from initial bitmap;
- using equations (14) and (15) applied to the 2D-DCT coefficients of the four sub-blocks.

The all 64 coefficient values obtained by the two methods were compared and the maximum absolute difference was found.

This test was performed for all possible integer pairs (dx,dy) between (0,0) and (8,8). The maximum absolute differences noticed for each case were arranged in a matrix where the element from line i and column j correspond to the maximum absolute difference noticed for a vertical displacement $dy=i$ and a horizontal one $dx=j$. This matrix is:

$$\varepsilon = \begin{bmatrix} 1.85 & 1.99 & 2.27 & 2.84 & 2.56 & 2.84 & 3.98 & 3.69 & 3.41 \\ 2.27 & 2.56 & 1.99 & 3.41 & 2.56 & 3.41 & 3.98 & 2.84 & 3.13 \\ 1.99 & 1.99 & 1.99 & 3.13 & 2.77 & 3.13 & 3.69 & 2.84 & 3.13 \\ 1.85 & 2.27 & 2.27 & 2.84 & 2.84 & 3.69 & 3.98 & 3.41 & 2.56 \\ 1.99 & 2.27 & 2.27 & 2.56 & 2.98 & 3.41 & 3.69 & 3.41 & 2.56 \\ 1.42 & 2.56 & 3.13 & 2.70 & 2.84 & 2.56 & 3.41 & 3.13 & 3.13 \\ 1.71 & 2.42 & 1.99 & 2.56 & 2.27 & 2.56 & 2.84 & 2.84 & 2.84 \\ 2.56 & 3.13 & 2.27 & 3.13 & 3.13 & 2.98 & 3.69 & 2.84 & 3.13 \\ 1.71 & 1.99 & 1.71 & 2.56 & 2.56 & 1.99 & 2.84 & 3.69 & 3.13 \end{bmatrix} 10^{-13}$$

The values are very small and are very likely caused by approximation errors inherent in floating point operations.

Another category of tests were focused on accuracy in the case of non-integer displacements. This kind of test is difficult to be done in 2D or using real images. For this reason, we performed the following tests only in 1D case:

- accuracy evaluation of shifted vector DCT coefficients obtained by equations (7) and (8);
- accuracy evaluation for a 1D version of the candidate block DCT coefficients determination described by equations (14) and (15).

The second test assumes the following situation:

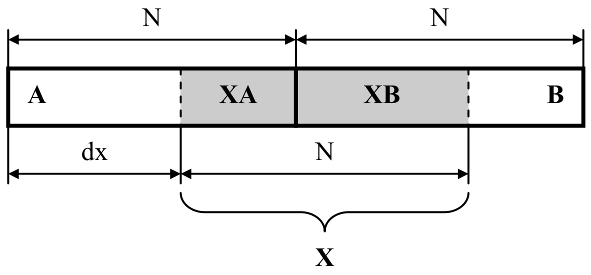

Fig.8: The 1D case of the candidate block DCT coefficients determination

We have the 1D-DCT transform of the two main vectors and is necessary to determine the 1D-DCT transform of the **X** vector.

The candidate vector **X** is a sum of 2 partial vectors (**XA** and **XB**), each of them containing parts of main vectors **A** respectively **B** like in the following figure:

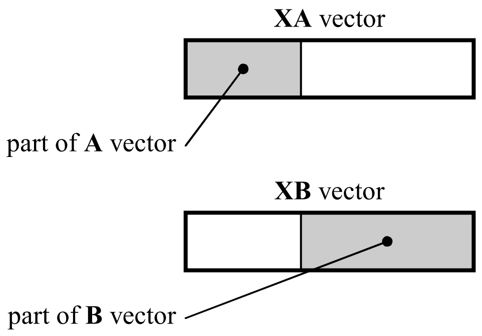

Fig.9: The structure of partial blocks

The shifts necessary for obtaining the two components are summarized in the next table:

Component	Main	Horizontal	
		Direction	Amount
XA	A	Left	dx
XB	B	Right	N-dx

Table 2: Generation of candidate block components

Like in the 2D case, DCT transform of the candidate vector T_x can be determined from the DCT transform of its components, using a correspondent equation:

$$T_X = T_{XA} + T_{XB} \tag{16}$$

where:

$$\begin{aligned} T_{XA} &= T_A\, H(-dx) \\ T_{XB} &= T_B\, H(N-dx) \end{aligned} \tag{17}$$

Both tests will be performed using some test-vectors chosen for an easy evaluation of the DCT coefficients determination accuracy.

Evaluation will be performed on the non-transformed vectors obtained from the DCT coefficients after applying the inverse transform.

3.4.1. Results of shift evaluation tests

The first test vectors tried were the 8 base functions which define the DCT transform. The values of the *K* base vector are defined by:

$$y_i^K = \cos\left(\frac{k(2i+1)\pi}{16}\right), i = 0:7 \tag{18}$$

In the ideal case, shifting these vectors to the right by s positions will produce the following values for elements:

- for $s \le 0.5$:

$$y_i^K(s) = \cos\left(\frac{k(2i+1-2s)\pi}{16}\right) \tag{19}$$

- for $s > 0.5$:

$$y_i^K(s) = \begin{cases} 0 & , i = 0 \\ \cos\left(\frac{k(2i+1-2s)\pi}{16}\right) & i \ne 0 \end{cases} \tag{20}$$

All these vectors were tested for displacements to the right between 0 and 1 position with a resolution of 0.05.

To not exceed the page limit, in this paper, will be presented only the results for the extreme values of the *K* (1 and 7) and for displacements by 0.25, 0.5, 0.75 and 1 positions.

In the next plots, the dotted line represent the ideal shifted vector determined by equations (19) and (20) and the solid line represent the vector obtained performing shift in transformed domain followed by inverse transform:

- for *K*=1:

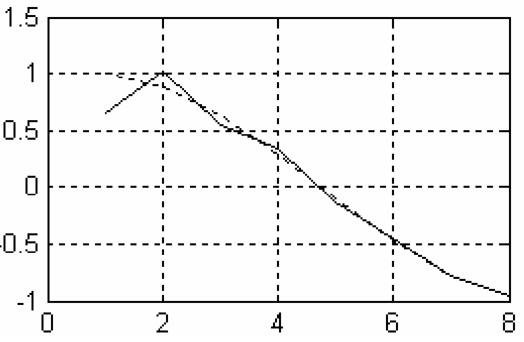

Fig.10: *K*=1, shift by *s*=0.25 positions

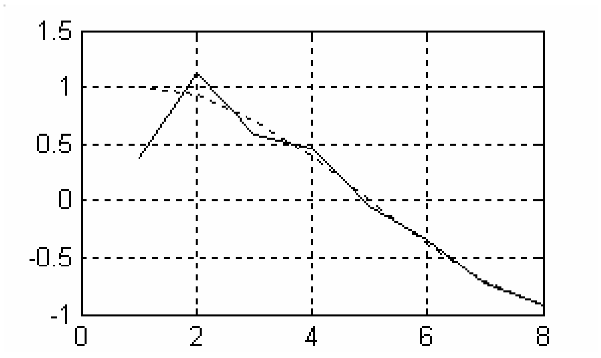
Fig.11: K=1, shift by s=0.5 positions

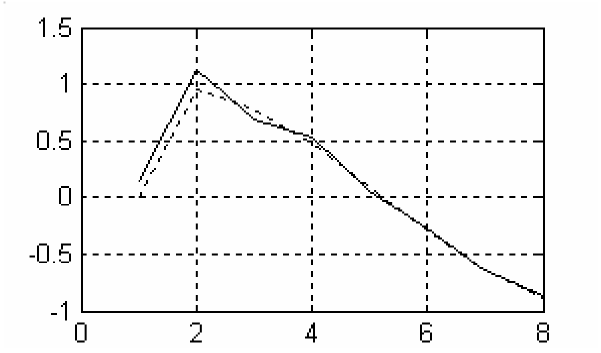
Fig.12: K=1, shift by s=0.75 positions

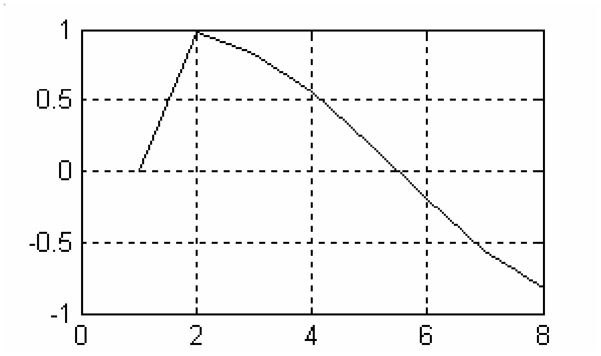
Fig.13: K=1, shift by s=1 position

- for K=1:

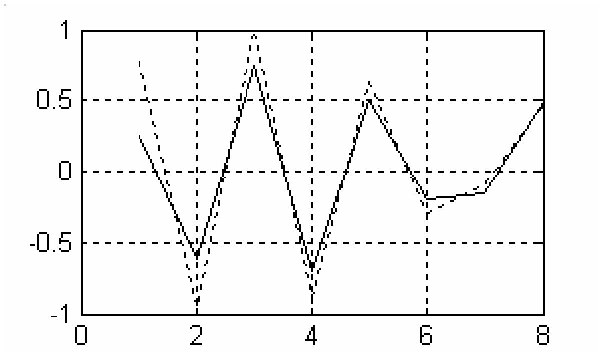
Fig.14: K=7, shift by s=0.25 positions

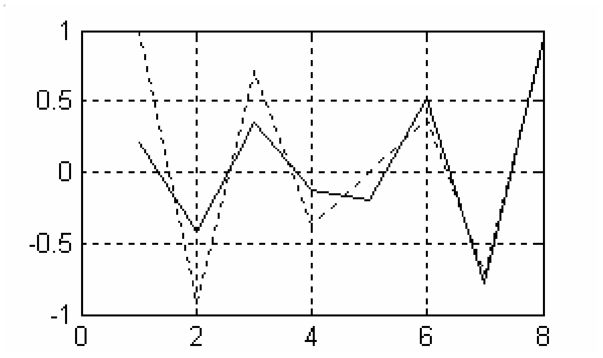
Fig.14: K=7, shift by s=0.5 positions

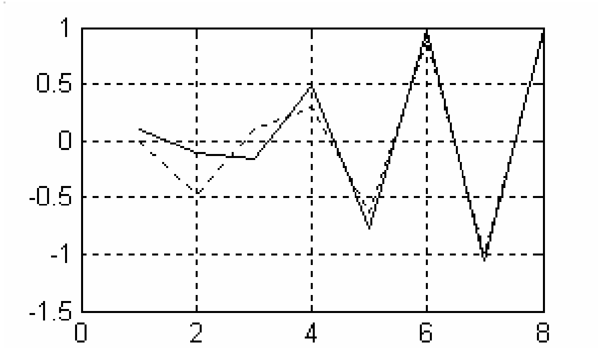
Fig.15: K=7, shift by s=0.75 positions

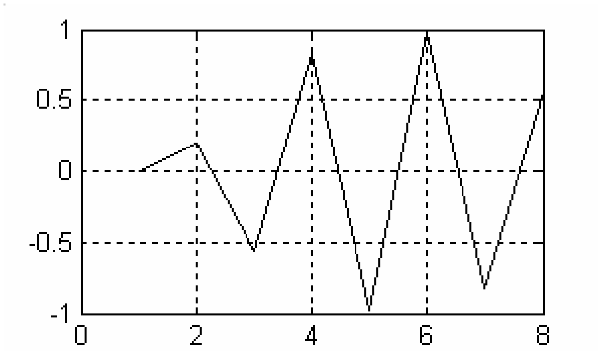
Fig.16: K=7, shift by s=1 position

It can be noticed that there are some errors in determination of shifted vectors especially for the displacement values around 0.5 positions, the magnitude of these errors increasing as K become higher.

These errors are more visible for the elements located on the left side of the vector. This side corresponds to the place where the values of the vector are not defined for displacements greater than 0.5 positions and zero padding is performed.

Another test vector which was tried is a linear ramp defined by:

$$y_i = i \tag{21}$$

In the ideal case, shifting these vectors to the right by s positions will produce the following values for elements:

- for $s \le 0.5$:

$$y_i(s) = i - s \tag{22}$$

- for $s > 0.5$:

$$y_i(s) = \begin{cases} 0 \ , i = 0 \\ i - s \quad i \ne 0 \end{cases} \tag{23}$$

This vector was tested for displacements to the right between 0 and 1 position with a resolution of 0.05 but, like in the base vector case, only the results for 0.25, 0.5, 0.75 and 1 values are presented in the following plots.

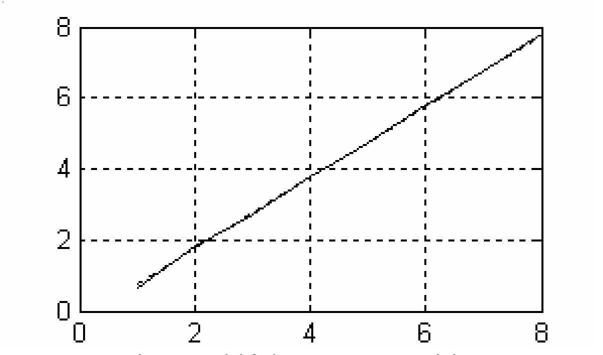
Fig.17: shift by s=0.25 positions

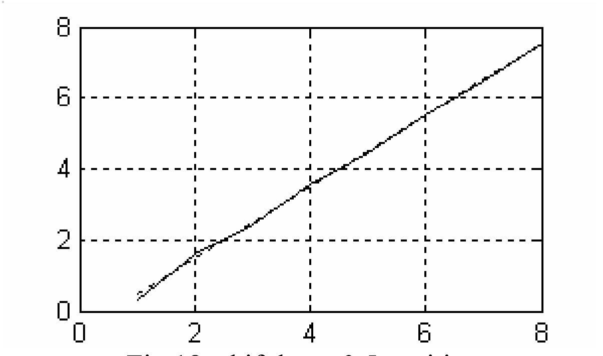
Fig.18: shift by s=0.5 positions

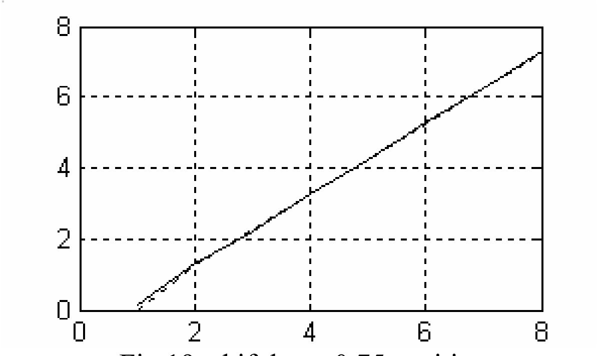
Fig.19: shift by s=0.75 positions

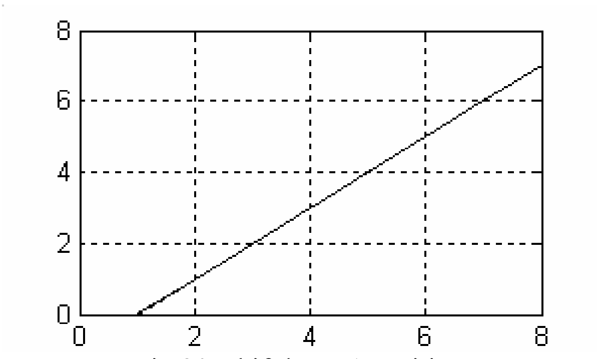
Fig.20: shift by s=1 position

For this vector, errors are still present but their magnitudes are smaller than in the base vector cases.

3.4.2. Results of candidate vector determination evaluation tests

The first test vectors were the same 8 base functions which define the DCT transform. These base vectors were used as the main vectors **A**. For continuity reasons, the base vectors **B** were chosen to be the extensions of the base functions as follows:

$$\begin{aligned} a_i^K &= \cos\left(\frac{k(2i+1)\pi}{16}\right) \quad , i = 0:7 \\ b_i^K &= \cos\left(\frac{k(2(i+8)+1)\pi}{16}\right) , i = 0:7 \end{aligned} \tag{24}$$

In the ideal case, the candidate vector **X** obtained for a displacement dx will have the following values:

$$x_i^K(dx) = \cos\left(\frac{k(2i+1+2dx)\pi}{16}\right) \tag{25}$$

Testing these vectors for displacement values between 0 and 1 position with a resolution of 0.05 a very small error (about $10^{-15}\dots10^{-14}$) was noticed.

This fact leads to the conclusion that shift estimation errors corresponding to the **A** and **B** main vectors have almost same values but with opposite signs and canceling themselves.

The second test vectors were a linear ramp extended to both **A** and **B** vectors as follows:

$$\begin{aligned} a_i &= i \qquad , i = 0:7 \\ b_i &= 8 + i \ \ , i = 0:7 \end{aligned} \tag{26}$$

In the ideal case, the candidate vector **X** obtained for a displacement dx will have the following values:

$$x_i = i + dx \tag{27}$$

Testing for displacement values between 0 and 1 position with a resolution of 0.05, an error of maximum 0.0479 was noticed. Although this error is bigger than in the base vectors case, its value is small enough to be considered almost null.

3.5 Motion estimation

In the motion compensation step, the difference between the "reconstructed" frame and the actual frame is encoded using a still image compression algorithm (usually based on discrete cosine transform, quantization and entropy coding). Classic matching functions like mean squared error (MSE) or mean absolute difference (MAD) do not guarantee that the difference frame is easy to compress because they try to minimize the energy of the difference frame in the bitmap domain not in transformed one.

Using the proposed method for the estimation of transformed coefficients of the candidate block, the matching functions can evaluate the energy of the DCT coefficients which are actually encoded.

One idea of matching function likely to be used in the transformed domain is:

$$F\left(X(dx,dy),Y\right)=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\frac{\left|x_{i,j}(dx,dy)-y_{i,j}\right|}{h_{i,j}} \qquad (28)$$

where:

- X is the matrix of 2D-DCT coefficients of the candidate block from the reference frame;
- Y is the matrix of 2D-DCT coefficients of the current block;
- $h_{i,j}$ are the quantization coefficients used in the motion compensation step for compressing the difference frame.

It can be noticed that the differences between DCT coefficients of the current and candidate block are weighted by quantization coefficients. The reason is to make the matching function less sensitive to the high frequency components which are however reduced in the quantization step.

Another reason behind this weighting can be the fact that noise tend to affect more the high frequency components, hence, this weighting will reduce the sensitivity of the matching function to the noises too.

Using this matching function defined above, its equivalent in transformed domain for the full-search algorithm can be developed.

Suppose that the current block is known by its 2D-DCT transform Y. The search area cover four adjacent blocks (see Figure 1). All 2D-DCT transforms of these four blocks are known as T_A, T_B, T_C and T_D. The result is a motion vector (vx,vy) which gives the relative displacement of the best matched block from the reference frame.

```
1. $fmin=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\frac{|y_{i,j}|}{h_{i,j}}$, $vx$=-1, $vy$=-1
2. For $dx$=0:8 (allowed in fractionar steps)
   2.1. For $dy$=0:8 (allowed in factionary steps)
        2.1.1. determine 2D-DCT transform of
               the candidate block $X(dx,dy)$ using
               equations (11) and (12)
        2.1.2. evaluate the match function:
               $f=F\left(X(dx,dy),Y\right)$ using equation (13)
        2.1.3. if $f<fmin$ then
               2.1.3.1. $fmin=f$
               2.1.3.2. $vx=dx$
               2.1.3.3. $vy=dy$
```

At the end, vx and vy will contain the horizontal and vertical components of the motion vector. If vx and vy will have -1 value, this means that no candidate block from the search area would offer a smaller coefficients energy of the difference than the actual coefficients of the current block.

4 Conclusion

The method for motion estimation in DCT transformed domain has some advantages over classic methods:

- if the video material is already compressed with a DCT based algorithm (like MJPEG or DV25) it is not necessary to perform conversions from transformed to bitmap domain before motion estimation and then another conversion back for the difference block;
- this method allows to test candidate blocks shifted by non-integer amount of pixels, hence pixel interpolation and searching with greater resolution are avoided;
- it allows using of some match functions more appropriated to the motion compensation step.

It can be noticed that number of operations needed by the full-search algorithm version presented here is much bigger than for classic version. This fact leads to the need of extending the research for finding efficient versions of the search step.

Another weak point of this method is the presence of the errors in the non-integer displacement cases. However, the tests showed that the magnitude of these errors is small enough to be considered important.

References:

[1] L. Hanzo, P. J. Cherriman and J. Streit, *Video Compression and Communications - >From Basics to H.261, H.263, H.264, MPEG4 for DVB and HSDPA-Style Adaptive Turbo-Transceivers (Second Edition)*, John Wiley & Sons, Ltd, 2007

[2] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, T. Ishiguro *Motion-compensated interframe coding for video conferencing*, Proceedings NTC'81 (IEEE), p G.5.3.1 - G.5.3.4

[3] Jaswant R. Jain, Anil K. Jain *Displacement measurement and its application in interframe image coding*, IEEE Transactions on Communications, Volume COM-29, Number 12, p 1799 - 1808, December 1981

[4] Kwon Moon Nam, Joon-Seek Kim, Rae-Hong Park, Young Serk Shim, *A fast hierarchical motion vector estimation algorithm using mean pyramid*, IEEE Transactions on Circuits and Systems on Video Technology, Vol.5, No.4, pp 344-351, (1995)

[5] M. Tun, K. K. Loo, J. Cosmas, *Semi-Hierarchical Based Motion Estimation Algorithm for the Dirac Video Encoder*, WSEAS Transactions on Signal Processing, Issue 5, Volume 4, May 2008

[6] C. L. Lin, J. J. Leou, *An Adaptive Fast Search Motion Estimation Algorithm for H.264*, WSEAS Transactions on Communications, Issue 7, Volume 4, July 2005, pp. 396-406.

[7] H. Nam, S. Lim, *A New Motion Estimation Scheme Using a Fast and Robust Block Matching Algorithm*, WSEAS Transactions On Information Science & Applications, Issue 11, Volume 3, November 2006, pp. 2292-2299.

[8] ISO/IEC 10918-1, *Information technology - Digital compression and coding of continuous-tone still images - Part 1: Requirements and guidelines*, International Electrotechnical Commission, 1994

[9] IEC 61834-2, *Recording - Helical-scan digital video cassette recording system using 6,35 mm magnetic tape for consumer use (525-60, 625-50, 1125-60 and 1250-50 systems) - Part 2: SD format for 525-60 and 625-50 systems*, International Electrotechnical Commission, 1998

[10] ISO/IEC 14496-10, *Information technology - Coding of audio-visual objects - Part 10: Advanced Video Coding*, International Electrotechnical Commission, 2005

[11] Ut-Va Koc and K. J. R. Liu, *DCT-based motion estimation*, IEEE Trans. On Image Processing, Vol. 7, July 1998, pp.948-965

[12] Li, M.; Biswas, M.; Kumar, S.; Truong Nguyen, *DCT-based phase correlation motion estimation*, International Conference on Image Processing ICIP'04, Volume 1, 24-27 Oct. 2004 Page(s):445 - 448 Vol. 1

[13] Ut-Va Koc and K. J. R. Liu, *DCT-based subpixel motion estimation*, Acoustics, Speech, and Signal Processing, 1996 (ICASSP-96), Conference Proceedings., 1996 IEEE International Conference on, 1930-1933, 1996

[14] Proakis J.G., Manolakis D.G., *Digital Signal Processing. Principles, Algorithms and Applications*, third edition, Prentice Hall, Upper Saddle River, New Jersey, USA, 1996.

[15] Stefanoiu D., Culita J., Stoica P., *Foundation of Systems Modeling and Identification*, Printech Press, Bucharest, Romania, 2005.