Block Matching Motion Estimation with Variable Search Window Size

Ionuţ Pirnog and Claudia Cristina Oprea
Telecommunications Department
"Politehnica" University of Bucharest
313, Splaiul Independentei, Sector 6, 060042
Bucharest, Romania
ionut@comm.pub.ro, cristina@comm.pub.ro

***Abstract*— Block matching algorithms for motion estimation were developed in order to obtain reasonable motion estimation efficiency with minimum computational cost. Although the gain in the computational complexity is significant these algorithms have less precision in estimation than the basic block matching motion estimation algorithm, i.e., the Full Search algorithm. The proposed motion estimation method can be used with any of the existing block matching algorithms and brings an increase of estimation precision with small increase of the global computational cost. This improvement is achieved by choosing the search window size depending of the ration between the frame size and the motion area size.**

Keywords – motion estimation, block matching, variable search window

I. INTRODUCTION

Motion information is very useful in the video compression process [2] since the development of video content retrieval applications [3]. In the video compression systems, the motion vectors are used for the representation of a video frame based on the previous frames. In the content retrieval systems, the video content can be found based on the video motion properties expressed by motion descriptors [4]. The extraction of motion vectors from a video frame based on the previous frame is known as motion estimation. There are many motion estimation methods, e.g., parametric methods, stochastic methods. The simplest and most used motion estimation method is the one based on block matching. The block matching algorithms split the current video frame into blocks and for each block a motion vector is extracted by finding the best matching block in the previous frame. The best matching block is found using a cost function that measures the similarity between two blocks. Since the best block is usually in the vicinity of the position of the current block, but in the previous frame, the search for the best matching block is not performed in the entire frame but in an area called the search window. The dimension of the search window defines the computational cost of the algorithm but also the precision of the estimation. The best block matching algorithms use a fixed dimension for the search window and show good results [5].

In this paper we present i) a group of fast block matching algorithms for motion estimation, ii) the importance of the search windows size in the precision of the estimation, iii) the algorithms that show increase in precision, and iv) a method for selecting the search windows dimensions in order to obtain the best estimation precision without an increase of computational cost. The fast block matching algorithms gain a significant decrease of the computational cost by selecting only a small set of blocks in the search windows. The current block, or the reference block, is compared only to these blocks and the best matching block is selected from this set. This means less comparisons, so small computational cost, but also the possibility that the best block is not found between the block in the search set. The main difference between the existing block matching algorithms is the search pattern, i.e., the method for selecting the blocks in the search set. We can classify these algorithms into two categories: fixed number of steps and variable number of steps. The ones in the first category have fixed number of steps and the estimation precision does not depend on the search window dimensions. The ones in the second category usually start with a search step, i.e., half the search window parameter, so the precision of the estimation depends on the search window size. The method proposed in this paper uses a variable size search window in order to obtain better motion estimation with no increase in the overall computational cost. The selection of the search window size is done in relation to the ratio between the frame size and the motion area size.

The rest of the paper is organized as follows. In Section II, we briefly describe the block matching motion estimation method; then, we present four of the best fast block algorithms, one with fixed number of steps and three with variable number of steps. In Section III, we present the proposed search windows' dimensions selection method. The comparative experimental results of the presented algorithms for different window sizes are shown in Section IV. Finally, the conclusions are provided in Section V.

II. BLOCK MATCHING MOTION ESTIMATION

Motion information is the most significant information of videos. A video can be regarded as a group of successive frames or images that together convey a specific message. Therefore, extracting features of videos can be accomplished using existing methods for extracting image features. There is a very important feature of the video that does not exist for images, namely the motion.

The concept of motion refers to the variation in time of the spatial position and applies to existing objects or the entire frame, in the case of camera motion. In the first case the background is not changing position and only the objects in the video have a variation of position over time. In the second case the whole frame changes over time due to the camera motion.

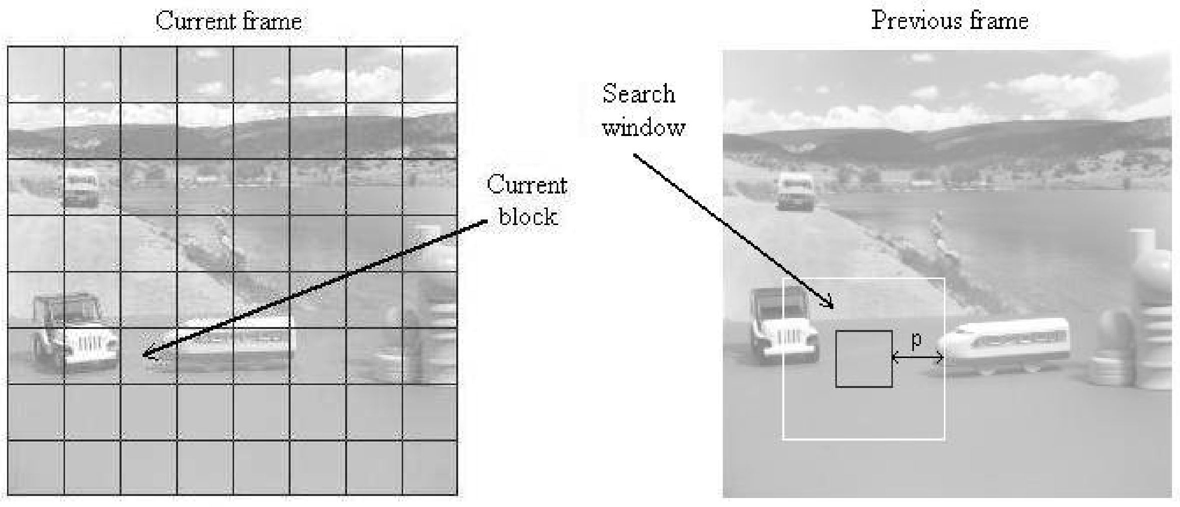


Figure 1. Schematic representation of block matching motion estimation.

There is also the case in which there is both camera motion and moving objects. Extracting video motion information is called motion estimation. This operation is done by comparing two by two successive video frames using different methods.

Motion estimation is used in the processes of video compression/decompression. In the compression phase the motion is estimated by comparing the current frame with the previous frame. Then, using the motion information and the previous frame, a motion compensated image of the current frame is build and the difference between the current frame and the motion compensated frame is computed. Also, instead of compressing the current frame, the motion information and the error frame is compressed. In this way higher rates of compression are achieved.

There are two classes of motion estimation methods:

- Motion flow estimation: for each pixel of the frame a motion vector is determined. The advantages of the methods in this class are high accuracy and high resolution of estimation. The main disadvantage is the computational complexity.
- Motion estimation based on blocks of pixels, known as block matching motion estimation. The basic idea of block matching algorithms is dividing the current frame is a matrix of non-overlapping macro blocks and determining motion vectors for each block of video frame (Figure 1).

The main advantage of the methods of the second class is that the size of pixel blocks can be chosen depending on the particular application. So for applications requiring high precision of motion vector estimation the size of the pixel blocks can be smaller and the computational complexity will increase, while for applications where speed is more important than the accuracy the blocks of pixels can be larger. If the size of the blocks is chosen 1x1 we obtain the motion flow estimation.

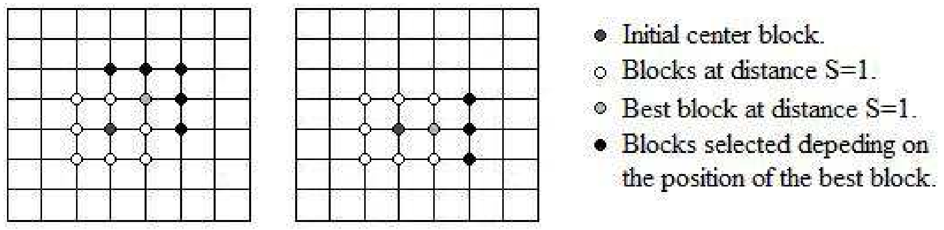
After the splitting of the frame into pixel blocks the motion of each block is estimated as follows: the block in the current frame is compared to all overlapping blocks in the search window. The search window is an area in the previous frame obtained by selecting the corresponding block, the block with the same spatial position as the current block, and adding pixels in each direction. The parameter is called the search window parameter. Its value determines the estimation precision and the computational complexity. Higher value implies higher chances of correct estimation and high number of blocks in the search window.

Determining the best block is made based on a cost function. For each block in the current frame a set of cost function values is determined by comparing it to all overlapping block in the search window. The block with the minimum cost is the best matching block. The most used cost functions are the Mean Absolute Difference (MAD) and the Mean Squared Error (MSE) given by (1) and (2), respectively:

$$MAD = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left|C_{ij} - P_{ij}\right|, \quad (1)$$

$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(C_{ij} - P_{ij}\right)^2, \quad (2)$$

where N is the block size, C_{ij} are the pixels values of the block in the current frame, and P_{ij} are the pixels values of the block in the previous frame.



● Initial center block.
○ Blocks at distance S=1.
◉ Best block at distance S=1.
● Blocks selected depeding on the position of the best block.

Figure 2. Example of blocks selected at every step for the New Three Step Search algorithm.

After the best matching block is determined the motion vector is computed as the difference between the spatial position of the current block and the spatial position of the best block. The resulting motion vector has two components for each direction, horizontal and vertical.

Based on the motion vectors of all blocks from the current frame and blocks in the previous frame the motion compensated frame is computed. The estimation precision or accuracy is determined using the Peak Signal to Noise Ratio (PSNR) between the current frame and the motion compensated frame, i.e.,

$$PSNR = 10\lg\left(\frac{Vpp^2}{MSE}\right), \tag{3}$$

where Vpp is the peak to peak value of the original data and MSE is the mean square error between the original data and the motion compensated data.

A. Full Search Algorithm

The first block matching motion estimation algorithm is called the Full Search (FS) algorithm, where all the overlapping blocks in the search window are used for determining the best matching block. Although this algorithm is the best in terms of prediction quality and simplicity, it is also the most inefficient in terms of arithmetic complexity. To assess the computational complexity of the FS algorithm we needed to determine the number of blocks in the search window compared with each reference block. For example, for 16x16 blocks and a search parameter we have 225 blocks in the search window.

To reduce the computational complexity new algorithms were developed with a higher quality complexity ratio. These algorithms are called suboptimal because they offer lower prediction quality than the algorithm above and are also called fast algorithms because they have lower computational complexity. These algorithms use only a set of blocks from the search window to determine the best matching block.

There are two classes of fast algorithms:

- Search Window Independent algorithms;
- Search Window Dependent algorithms.

B. Search Window Independent Algorithms

In order to properly classify the fast block matching algorithms it is important to explain the way these algorithms function and to identify the parameters that determine the affiliation of a certain algorithm to one of the classes defined earlier.

All of the fast block matching algorithms have an initial step in which a block from the current frame is compared to the correspondent block in the previous frame and a number of blocks at a distance S from the correspondent block. The number of blocks and their position is chosen different for every algorithm. The distance is defined in terms of number of pixels.

The algorithms in the first class start with an initial distance $S = 4$ and after one step the distance is halved. The algorithms stop when the distance reaches 1.

For this category of algorithms the size of the search window is 7 pixels in each direction, meaning that if we have block of dimension $N \times N$ then the size of the search window will be $(N+7)\times(N+7)$. We recall that the search window parameter is denoted with p. The value $p = 7$ is chosen so that the algorithms go through all their steps without reaching a block outside the search window limits.

The window independent algorithms have two important properties:

- The maximum number of verified blocks is known.
- The precision of estimation is independent of the search window dimensions, so that if the motion has a high amplitude the increase of the search window dimensions will have no effect on the estimation.

The most efficient fast block matching algorithm in this category is the New Three Step Search (NTSS) algorithm. It has good precision of estimation and low computational complexity. It does not fall into the category of interest for the proposed method but for comparison reasons we present it in the following.

C. New Three Step Search Algorithm

The NTSS algorithm compares the block in the current frame to the center block, eight blocks at distance $S = 4$ and eight blocks at a distance $S = 1$ on the horizontal and vertical axes, in the previous frame [6].

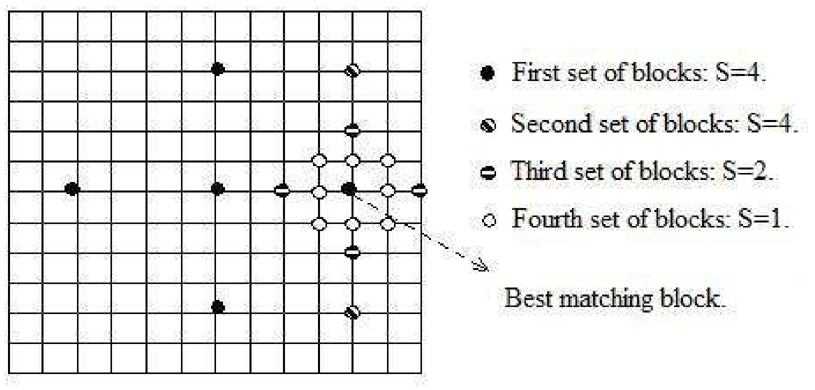

● First set of blocks: S=4.
Second set of blocks: S=4.
Third set of blocks: S=2.
○ Fourth set of blocks: S=1.
Best matching block.

Figure 3. Example of blocks selected at every step for the Two Dimensional Logarithmic Search algorithm.

The best block from these initial 17 blocks is determined based on the cost function values. Depending on the positions of the best block we have three situations:

1) If the best block is the one in the centre of the search window, then the algorithm stops.

2) If the best block is one of the blocks located at a distance $S = 1$, then its neighbors are compared with the current block and the best is determined as the block with the minimum cost function value.

3) If the best block is one of the blocks located at a distance $S = 4$, then the block is set as the new center, the distance is halved and all eight blocks at distance S are verified. The algorithm stops when the distance is one.

To decrease the number of blocks compared and to eliminate the re-evaluation of some blocks, the neighbors selected in the second case (when the best block is one of the blocks located at distance $S = 1$) depend on the position of the best block as shown in Figure 2.

D. Search Window Dependent Algorithms

As compared to the algorithms in the first class the algorithm in this class start with an initial search distance equal to half the search window parameter p. The selection of the blocks that are used for computing the cost values depends on the algorithm.

It is obvious that compared to the first class the algorithms in the second class have a maximum number of verified blocks that depends on the search window dimensions. Also, if the search window size increases the precision of estimation will increase in the case with high amplitude motion.

In the following, we present the most important algorithms in this class.

E. Two Dimensional Logarithmic Search Algorithm

Two Dimensional Logarithmic Search (TDLS) algorithm selects at every step the center block and four blocks at a distance S on the horizontal and vertical axes. The initial distance is chosen as half the search window parameter $p = 7$. If the search parameter is an odd number then S is chosen as the rounded value of $p/2$.

After the selection of the initial distance, the block from the current frame is compared to the center block, i.e., the corresponding block in the previous frame, and the four blocks at distance S on the horizontal and vertical axes. The comparison is done by computing the cost functions. The block that gives the lowest cost function is selected for the next step.

If the block selected at the first step is the center block then the search distance S is halved, else the selected block is set as the new center and the first step is repeated.

When the search distance becomes equal to one the center block and all its neighbors are compared to the block in the current frame and the best matching block is selected as the block with the minimum cost function value.

F. Orthogonal Search Algorithm

The Orthogonal Search (OS) algorithm is a combination of the TDLS algorithm and the Three Step Search (TSS) algorithm. The TSS algorithm is the first fast block matching algorithm and is independent of the search window dimensions, so it belongs to the first class of algorithms. The similarity between OS and TSS algorithms is the number of steps.

The initial search distance is chosen as half the search window parameter. The OS algorithm has the following 3 steps:

1) The block from the current frame is compared to the center block and the two blocks at distance S on the horizontal axis. The block with the minimum cost function value is set as the new center.

2) The center block and the two blocks at distance S on the vertical axis are verified, and the new center is selected as the block with the lowes cost.

3) If the distance parameter S is bigger than one then the distance is halved and steps 1 and 2 are repeated. Else, the last center block is the best matching block.

G. Adaptive Rood Pattern Search Algorithm

The Adaptive Rood Pattern Search (ARPS) algorithm uses the motion information of the neighboring block in the left. This is helpful if the current block and its neighbor on the left belong to the same object in the frame; in this case, their motion is similar [7]. The steps of the ARPS algorithm are:

1) The block from the current frame is compared to the center block, four blocks at distance S on the horizontal and vertical axes, and the block indicated by the motion vector of the neighbor block in the left. The initial search distance S is selected as the maximum between the absolute values of the neighboring motion vector.

2) The block with the minimum cost function value is set as the new center. The search distance is set to 1 and the centre block together with its four axis neighbors are evaluated.

3) If the block with the minimum cost is in the centre, then the algorithm stops; consequently, this is the best block. Else, step 2 is repeated.

III. Proposed Search Window Selection

The proposed method is based on the simulation results that showed, as the theory stated, that by increasing the size of the search window the precision of estimation increases. The increase of the search window dimensions has an unwanted effect, i.e., an increase of the number of verified blocks.

To highlight these observations we present in Table I the PSNR between the current frame and the motion compensated frame for all, of the above presented, fast block matching algorithms. Also, in Table II we present the total number of blocks verified. The simulations were done for different search window dimensions, 8x8 pixel blocks and the computer generated video sequence “Motion.”

From Table I it can be observed that for the search window dependent algorithms the estimation precision increases with the increase of the search window.

In Table II it can be seen that along with the increase of the PSNR there is also on increase of the number of verified blocks.

Based on the results presented in both tables we make two observations:

1) First of all, for the search window dependent algoritms the estimation precision parameter for larger search windows exceeds both the FS algoritm and the NTSS.

2) Second, although for all the search window dependent algoritms the number of verified blocks increases, the computational complexity remains significantly smaller than for the FS algoritms, and in some cases, like ARPS, even smaller than for the NTSS algorithm.

TABLE I. PSNR Values for Different Window Sizes

Algorithm	Search Window Parameter			
	p=7	*p=15*	*p=31*	*p=63*
FS	30.78	-	-	-
NTSS	30.49	30.49	30.49	30.49
TDLS	29.3	32.49	33.8	34.69
OS	28.88	31.54	32.18	33.45
ARPS	29.47	31.8	32.73	33.21

TABLE II. Overoall Number of Verified Blocks for Different Window Sizes

Algorithm	Search Window Parameter			
	p=7	*p=15*	*p=31*	*p=63*
FS	921600	-	-	-
NTSS	79898	79898	79898	79898
TDLS	74349	93676	111607	128330
OS	53248	69632	85988	102300
ARPS	28548	30038	30462	30642

There is also one disadvantage in increasing the search window. For sequences with low motion amplitude there is small or even zero increase in the estimation precision but the increase of the number of verified blocks remains. So the primary concern is determining a way of obtaining the best estimation precision with the lowest computational cost.

The goal of proposed method is the optimum search window size selection. This is done in three simple steps:

1) Motion area detection.
2) Search window parameter computation.
3) Motion estimation.

The detection of the area where motion exists is done by simple difference between the current frame and the previous frame and two morphological operations to eliminate the misdetection of motion due to variations of pixel intensity.

The search window parameter is computed based on the ratio between the entire frame and the motion area, as:

$$p = 2^{r+1} - 1, \tag{4}$$

$$r = \left[\frac{A_x \times A_y}{F_x \times F_y} \right], \tag{5}$$

where $[\cdot]$ is the round operator, A_x and A_y are the motion area dimension, F_x and F_y are the frame dimensions, and p is the search window parameter. The parameter p is chosen as a power of 2, minus one, so that when computing the search parameter S (as half the search window dimension) it will be also a power of 2.

By selecting the search window size this way we obtain the best PSNR with the lowest computational complexity. This means that if the ratio r is high the PSNR will be higher without increasing to much the computational complexity. If the ratio is close to one the window size will be low, with $p = 7$, the usual value for the window size independent fast block matching algorithms.

Also by applying the motion estimation algorithms only to the area where motion exists there will be a significant decrease of the computational complexity with small loss of estimation precision.

IV. SIMULATION RESULTS

In this section we present the comparative results of the presented fast block matching algorithms for fixed search window size and variable search window size selected with the proposed method.

The fast block matching algorithms presented in section II were implemented using Matlab and we used a set of video sequences containing monochrome videos and color videos of different sizes, some of the videos artificially generated and some from the real life. All the videos were obtained from test sequences databases commonly used for motion estimation.

We present the results for the test sequence "Motion:" a computer generated monochrome sequence with 10 frames, 512x512 pixels and high amplitude motion.

In concordance with step 3 of the proposed search window selection method presented in Section III, in this section the comparative results are split into two distinct scenarios: one to compare the results for the estimation when the improved method is applied to the entire frame, i.e., Variable Search Window – Full Frame (VSW-FF), and one to compare the results for the estimation when the improved method is applied only to the area where motion is detected, i.e., Variable Search Window – Motion Area (VSW-MA). The results for the two scenarios are compared to the results of the original segmentation method with fixed search window dimensions, i.e., Fixed Search Window (FSW).

For the first scenario, after the detection of the area with motion and the selection of the search window dimension, the algorithms are applied only to the area where motion is present. In this case, if the ratio between the entire frame and the area with motion is one then the PSNR will be slightly smaller than the one obtained with the initial motion estimation method but the overall number of blocks verified decreases. If the search window is bigger then the PSNR increases if the motion has high amplitude and or decreases if the motion has low amplitude. The decrease of the PSNR is due to the small intensity differences of the pixels from the areas of the frame where the algorithms are not applied.

Also, along with the decrease of the PSNR we obtain a decrease of the overall number of blocks verified. For this reason it is important to see the comparative result of the PSNR and Nb ratio.

For the second scenario the modified motion estimation method uses the selection of the search window dimension by detecting the area where motion is present. If the ratio between the entire frame and the area with motion is high then the search window size is bigger, according to equation (4). In this case, the precision of estimation will increase and also the overall number of verified blocks will increase.

To compare the results we present the PSNR between the current frame and the motion compensated frame of the initial motion estimation method using the FS, NTSS, TDLS, OS, and ARPS block matching algorithms, and of the proposed estimation method with variable search window size. We also present the overall number of blocks verified all the algorithms in the two situations.

For the first scenario, VSW-MA, and the computer generated sequence "Motion" we observe the following:

- First of all, as expected the NTSS algorithm results are independent of the search window size so the PSNR between the current frame and the motion compensated frame decreases for the proposed method. This happens because the algorithm is applied only to the motion area and due to small changes in the frames that are not determined by motion. The overall number of blocks verified decreases for the same reason above.
- For the other three algorithms if we compare the results presented in Figure 4, for the fixed size search window and for the variable size search window and motion area, we observe that there are two situations depending of the existence of camera motion or the occlusion of objects.
- In the first case, for frames 2-7 and 9, there is no camera motion and no occlusion. We see from Figure 4a-c that the PSNR between the current frame and the motion compensated frame obtained from the previous frame and the motion vectors increases for all of the search window dependent algorithms, TDLS, OS, and ARPS.
- From Figure 4d we observe that the highest increase in PSNR is obtained for the TDLS algorithm and that the precision of estimation for TDLS and ARPS exceeds the results of the FS algorithm. For the OS algorithm, although there is an increase of the PSNR, for some frames the precision of the motion estimation is lower that the one of the FS algorithm. In terms of PSNR between the current frame and the motion compensated frame we can conclude that the best results are obtained by the TDLS algorithm.
- Regarding the computational complexity, evaluated through the overall number of blocks verified, we observe from Figure 5a that for frames 1-7 and 9 there is a decrease in computational complexity because all the algorithms are applied only to the area where the motion is detected. For a better observation of the results regarding the computational complexity we have not represented the results for the FS algorithm, results that are constant for all the frames and are equal to 10^5 .

- In Figure 5b is represented the computational complexity for the three search window dependent algorithms. We can observe that the TDLS algorithm, that has the highest PSNR, has also the highest number of verified blocks. The OS and ARPS algorithms have lower computational complexity. In terms of the overall number of verified blocks we can conclude that both OS and ARPS algorithms have good results.
- So in the first discussed case, for the frames without camera motion, we can conclude that all of the three algorithms show good results with increase of the PSNR and a decrease of the overall number of verified blocks and that the windows parameter selection method has good results.
- In the second case, for frame 1, the existence of occlusion leads to smaller PSNR even if the search window increases. For frame 8 the existence of camera motion leads to the detection of a motion area almost equal to the entire frame. In this case the ratio between the motion area and the entire frame is close to 1 and the search window parameter is set to $p = 7$. In this case there is no increase in the PSNR for none of the algorithms and a small decrease in computational complexity due to the fact that the detected motion area is smaller than the entire frame.
- In this case the TDLS and ARPS algorithms have similar results in terms of PSNR but the ARPS algorithm has lower computational complexity.

As a conclusion for this scenario of the proposed motion estimation method we can state that:

- The computational complexity, evaluated through the overall number of blocks verified, decreases.
- The estimation precision, evaluated through the PSNR, increases in case of large amplitude motion or decreases slightly in case of small amplitude motion.

In the second scenario, i.e., VSW-FF, the results are presented in Figures 6 and 7. Based on these results we make the following observations:

- The NTSS algorithm has the same results for the proposed method as the initial method both in terms of PSNR and number of blocks verified. This was expected because the NTSS algorithm is independent of the search window.
- All of the three search window dependent algorithms show an increase in PSNR, Figure 6a-c, and also an increase in the overall number of verified blocks, Figure 7a. From the same figures we observe that for the proposed method in the case of the TDLS and ARPS algorithms the PSNR is higher than the PSNR for the FS algorithm. Also, even if the number of verified blocks increases is significant smaller that the number of verified blocks for the FS algorithm.
- From Figure 6d we observe that in terms of PSNR between the current frame and the motion compensated frame the algorithm with the best results is the TDLS algorithm, followed by ARPS and OS.
- From Figure 7b it can be seen that in terms of the overall number of verified blocks the ARPS algorithm has the best results, followed by OS and TDLS.

In conclusion, for the variable search window parameter method applied to the entire frame, the results show that all of the algorithms have an increase of PSNR, when there is no camera motion and the motion area is at least two times smaller than the entire frame. If there is camera motion then the value for the search window parameter will be equal to the value for the original method and the results the same.

For applications that require high precision of estimation and no constraints in computational complexity the proposed method can be applied to the entire frame. For applications that require low computational complexity the method can be applied only to the area where motion is detected with little loss or significant gain in PSNR.

In Tables III and IV, we represented the mean value of the PSNR and the total number of verified blocks, for the video sequence "Motion," for the initial motion estimation method with fixed search window parameter and the two variations of the proposed variable search window method.

TABLE III. MEAN PSNR VALUES FOR SEQUENCE "MOTION"

Algorithm	**Search Window Dimension Type**		
	FSW	***VSW – MA***	***VSW – FF***
FS	31.34	-	-
NTSS	31.06	30,67	31.06
TDLS	30.05	32,59	32.69
OS	29.74	31,24	31.39
ARPS	30.32	31,84	31.96

TABLE IV. OVEROALL NUMBER OF VERIFIED BLOCKS FOR SEQUENCE "MOTION"

Algorithm	**Search Window Dimension Type**		
	FSW	***VSW – MA***	***VSW – FF***
FS	921600	-	-
NTSS	87817	27399	87817
TDLS	79923	31154	100573
OS	53248	19363	69629
ARPS	34547	16851	35256

The second set of results is for the video sequence "Hall Monitor," a real life color video sequence with 144x176 pixels. In Tables V and VI, we presented the results for the original motion estimation method with fixed search window and the two variations of the proposed method. Based on these results we can make the following observations:

- In the first case, when the variable search window method is applied to the area where motion is detected, the PSNR between the current frame and the motion compensated frame decreases for all the presented algorithms but also the overall number of verified blocks decreases.
- In the second case, when the variable search window method is applied to the entire frame, the PSNR also decreases for all of the algorithm and the overall number of verified blocks increases.
- The results can be explained by the existence of low amplitude motion, this means that a good precision can be obtained using a small search window, and by the existence of illumination variations, which lead to an increase of the MSE and a decrease of PSNR.

TABLE V. MEAN PSNR VALUES FOR SEQUENCE "HALL MONITOR"

Algorithm	**Search Window Dimension Type**		
	FSW	***VSW – MA***	***VSW – FF***
FS	31.01	-	-
NTSS	30.82	29.89	30.82
TDLS	30.25	29.35	30.11
OS	29.88	28.99	29.66
ARPS	29.85	28.01	28.64

TABLE VI. OVEROALL NUMBER OF VERIFIED BLOCKS FOR SEQUENCE "HALL MONITOR"

Algorithm	**Search Window Dimension Type**		
	FSW	***VSW – MA***	***VSW – FF***
FS	89100	-	-
NTSS	8403	2804	8403
TDLS	7424	3031	9642
OS	5148	2177	7084
ARPS	2956	951	2628

V. CONCLUSIONS AND FUTURE WORK

In this paper we have evaluated the importance of the search window dimensions for fast block matching algorithms for motion estimation and a method for selecting the search window parameter depending of the area where motion is detected.

By evaluating twelve fast block matching algorithms for motion estimation, with different block sizes and search window dimensions, we concluded in [1] that the existing fast block matching algorithms can be split into two categories: fixed number of steps and thus independent of the search window dimensions, and variable number of steps and thus dependent on the search window dimensions. The results presented in [1] showed that for increased search windows some of the algorithms in the second category show an increase of the PSNR between the current frame and the motion compensated frame obtained from the previous frame and the estimated motion vectors.

We presented four fast block matching algorithms for motion estimation, one with fixed number of steps, and thus independent of the search window dimensions, and three with variable number of steps that depend of the search window dimensions. As shown in Tables I and II for the three algorithms with variable number of steps by increasing the search window parameter we obtain an increase of the motion estimation precision but also an increase of the computational complexity.

The basic idea behind the method proposed in this paper is to select the search window dimensions in such a manner that lead to good precision of estimation and low computational complexity.

The proposed motion estimation method uses a variable search window dimension depending on the detection of the area where motion is present. The detection of motion is done by simple differencing between the current frame and the previous frame and two morphological operations. The search window parameter, that defines how many pixels the search window is extended around the current block, is computed according to the ratio between the size of the entire frame and the size of the area with motion, as shown in equations 4 and 5.

We have evaluated the proposed method in two cases: when the algorithms are applied only to the area where motion is present and when the algorithms are applied to the entire frame. The first set of simulation results were obtained for a computer generated video sequence with high amplitude motion.

In the first case we observed that, although for some frames a small decrease of the PSNR may occur, the mean PSNR for the entire sequence increases and the overall number of verified blocks decrease significantly. The simulation results show that some algorithms have better results in estimation precisions, the TDLS algorithm, while others show a more significant decrease in computational complexity, the ARPS algorithm. Depending of the application we can use one or another.

In the second case we observed that the PSNR increases for all the frames, when the search window parameter increases, but also the overall number of verified blocks increases. Similar to the result in the first case, the TDLS algorithm shows higher increase in PSNR compared to the OS and the ARPS algorithms, and the ARPS algorithm shows lower increase of computational complexity compared to the TDLS and the OS algorithms. A very important observation is that all of the three window size dependent algorithms obtain better precision of estimation that the first block matching motion estimation algorithm, the FS algorithm, and with significant lower computational complexity.

The algorithms using the proposed search window selection method we were also applied to real life video test sequences. The simulation for these test sequences also included the two variations described above.

From this case of our simulation results we drew the following conclusion:

- The search window parameter chosen by detection of the motion area leads to two situation.
- First, when the sequences contain camera motion of significant illumination variations, the value of the search parameter is low and equal to the value recommended for the fast block matching algorithms. In this case there the results are the ones from which we started.
- Second, when there is no camera motion, the illumination variations are low enough and the objects in motion occupy an area much smaller than the entire frame, the search window parameter value is higher than the one in the first case.
- In this case, for the first scenario, when the algorithms are applied to the entire frame, the results for the real life test sequences show a small decrease in estimation precision and an increase in computational complexity. This is explainable by the existence of low amplitude motion that means no increase in estimation precision when the search window parameter increases, by the fact that in the areas without motion there are changes in pixel values due to illumination, changes that can be compensated by motion estimation with a low search parameter, and also by the search pattern used by the algorithms. The computational complexity increases because in the areas without motion many blocks are verified even if is not necessary.
- For the second scenario, when the algorithms are applied only to the motion area, the estimation precision decreases slightly but the computational complexity decreases significantly. The decrease in estimation precision is explainable by the illumination variations in the areas not used, variations that can compensated by motion estimation with a low search window parameter. The decrease in computational complexity is high and it is a very important aspect that can be exploited.
- As an overall conclusion of the presented method we can definitely say that the proposed search window parameter method show good results in estimation precision when the test sequences contain objects with high amplitude motion and also good results in computational complexity for the second scenario presented.

For future work we consider the idea of using the motion area detection for selecting the search window parameter value, by applying the algorithms with the selected parameter value only to the motion area and by applying the algorithms with the lowest parameter value for the areas without motion. Also we consider evaluating the results for the proposed method and the presented algorithms for different block dimensions.

ACKNOWLEDGMENT

This work was supported by the UEFISCDI Romania under the Grant PN-II-RU-TE no. 7/05.08.2010.

REFERENCES

[1] I. Pirnog, C. Anghel, A. A. Enescu, and C. Paleologu, “Evaluation of Fast Algorithms for Motion Estimation,” AICT 2011, The Seventh Advanced International Conference on Telecommunications, pp. 107-111, Mar. 2011.

[2] Z. Chen, “Efficient Block Matching Algorithm for Motion Estimation,” International Journal of Signal Processing, 5(2):133-137, 2009.

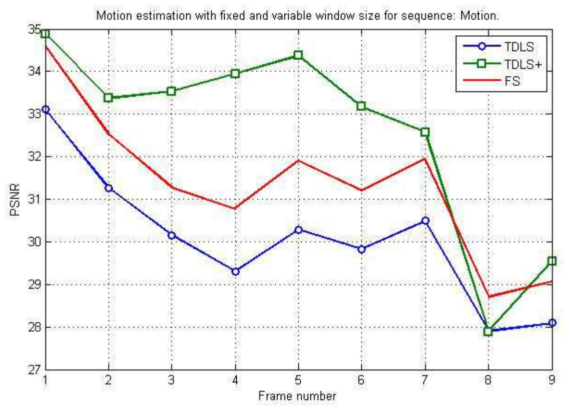
[3] ISO/MPEG N4358, “Text of ISO/IEC Final Draft International Standard 15938-3 Information Technology - Multimedia Content Description Interface - Part 3 Visual,” MPEG Video Group, Sydney, July 2001.

[4] A. Barjatya, “Block Matching Algorithms For Motion Estimation,” Final Project Paper, 2004.

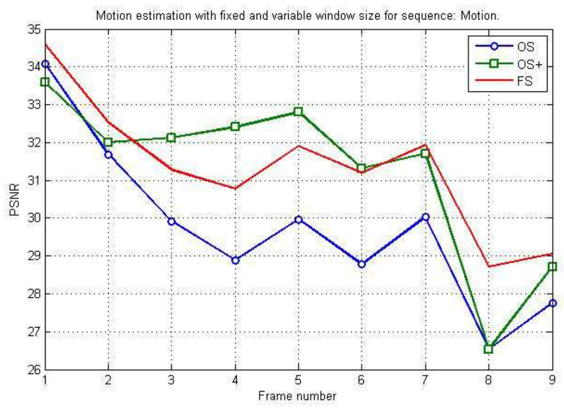
[5] Y. C. Lin and S. C. Tai, “Fast Full-Search Block-Matching Algorithm for Motion-Compensated Video Compression,” IEEE Transactions on Communications, 45(5):527-531, 1997.

[6] R. Li, B. Zeng, and M. L. Liou, “A New Three-Step Search Algorithm for Block Motion Estimation,” IEEE Trans. Circuits and Systems for Video Technology, vol 4., no. 4, pp. 438-442, Aug. 1994.

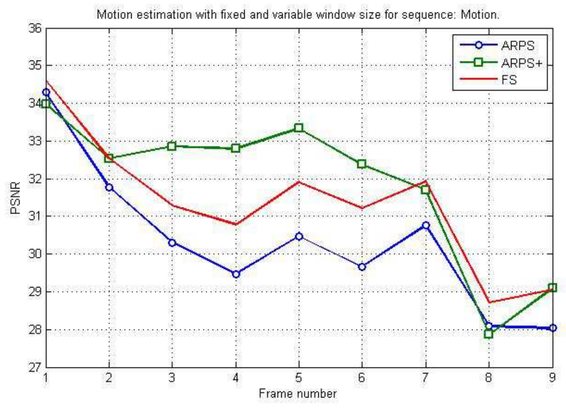
[7] S. Jamkar, S. Belhe, S. Dravid, and M. S. Sutaone, “A comparison of block-matching search algorithms in motion estimation,” Proceedings of the 15th International Conference on Computer Communication, pp. 730 – 739, Mumbai, India, 2002.



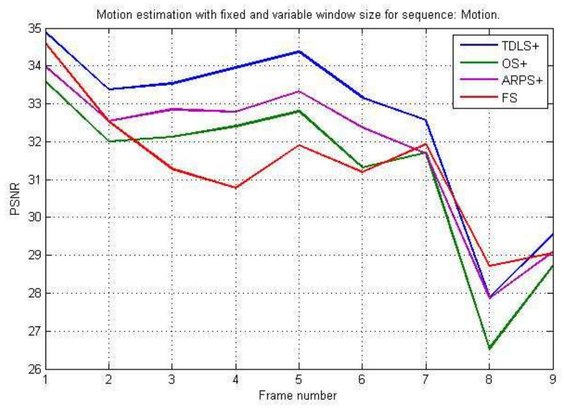
a)



b)

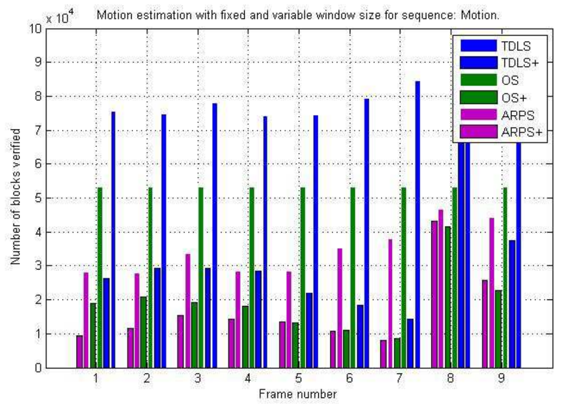


c)



d)

Figure 4. Comparative simulation results for video sequence “Motion” for 8x8 pixel blocks with fixed and variable search window size and motion area. a) Two-Dimensional Logarithmic Search. b) Orthogonal Search. c) Adaptive Rood Pattern Search. d) Comparative results of the three algorithms with variable search window size.



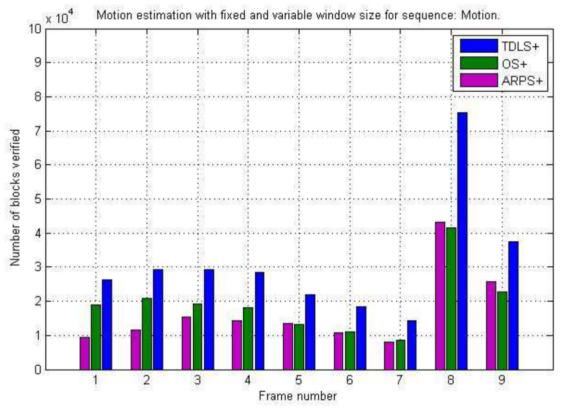
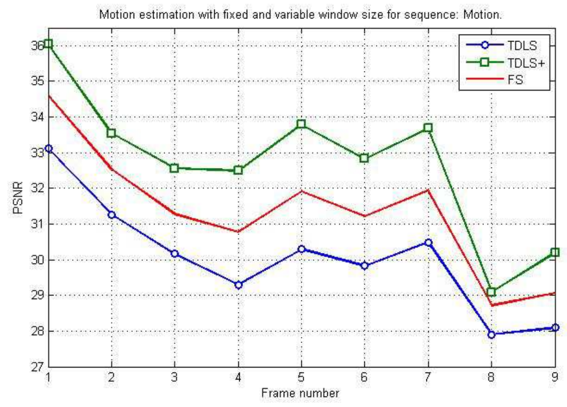
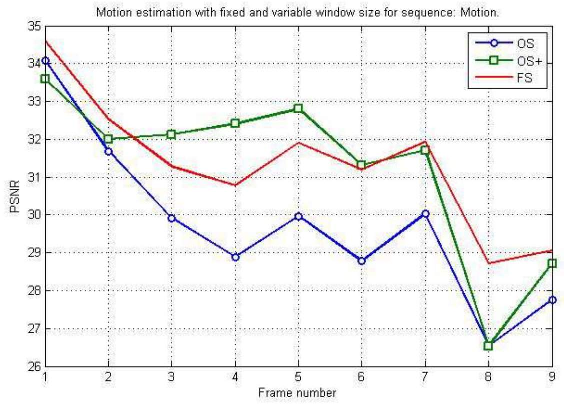


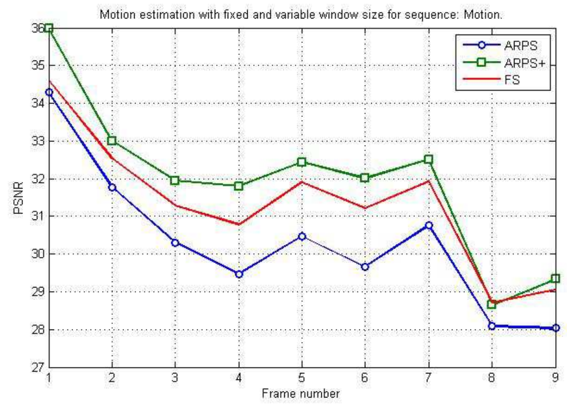
Figure 5. Overall number of verified blocks for block matching motion estimation for the video sequence “Motion” with 8x8 pixel blocks. a) Comparative results for fixed and variable search window size. b) Comparative results for the three algorithms in the case of variable search window size.



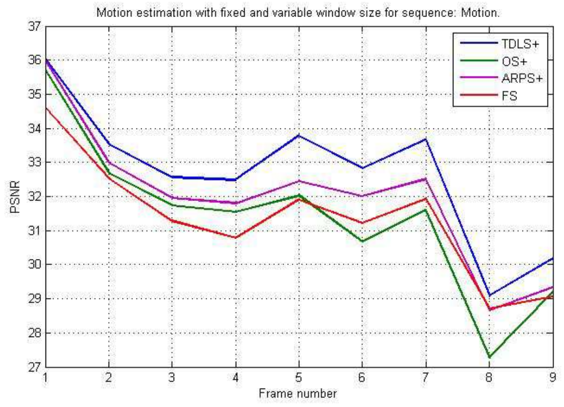
a)



b)

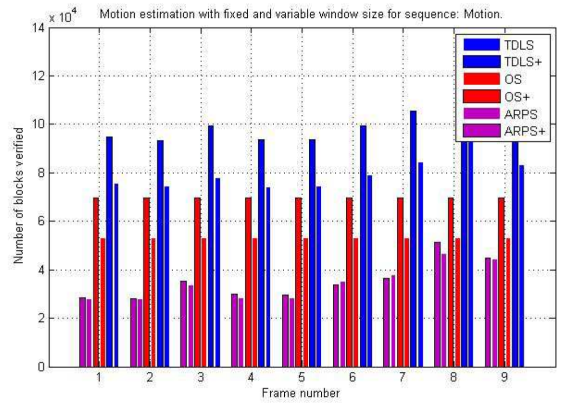


c)



d)

Figure 6. Comparative simulation results for video sequence “Motion” for 8x8 pixel blocks with fixed and variable search window size and full frame. a) Two-Dimensional Logarithmic Search. b) Orthogonal Search. c) Adaptive Rood Pattern Search. d) Comparative results of the three algorithms with variable search window size.



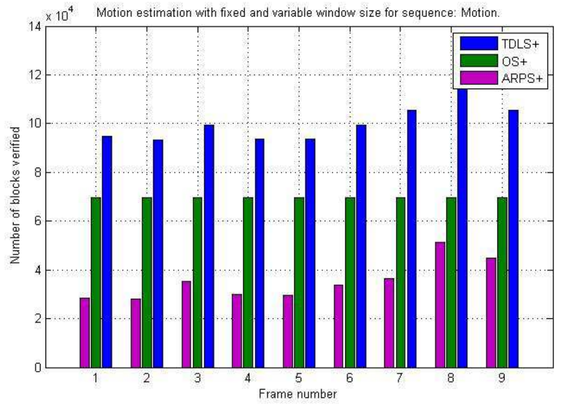


Figure 7. Overall number of verified blocks for block matching motion estimation for the video sequence “Motion” with 8x8 pixel blocks and full frame. a) Comparative results for fixed and variable search window size. b) Comparative results for the three algorithms in the case of variable search window size.