CORE
Metadata, citation and similar papers at core.ac.uk
Provided by Elsevier - Publisher Connector

Available online at www.sciencedirect.com
ScienceDirect
Procedia Technology 11 (2013) 199 – 205
Procedia Technology

The 4th International Conference on Electrical Engineering and Informatics (ICEEI 2013)

A Comparison of Different Block Matching Algorithms for Motion Estimation

Razali Yaakob*, Alihossein Aryanfar, Alfian Abdul Halin, Nasir Sulaiman

[a]*Faculty of Computer Science and Information Technology, Universiti Putra Malaysia, Malaysia*

Abstract

In this paper, four different block matching algorithms using motion estimation are evaluated where the effects of the macro block size used will be reviewed to find the best algorithm among them is scrutinized to determine the most optimal algorithm. Four different block matching algorithms are considered and implemented. Each algorithm is evaluated using different movies from the TRANS database [11] and comparisons are made through the Peak Signal to Noise Ratio (PSNR) and search points per macro block (i.e. computation time) for different sizes of macro blocks and search areas. The results suggest that among all the evaluated algorithms, ARPS has the best PSNR based on computation time.

© 2013 The Authors. Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
Selection and peer-review under responsibility of the Faculty of Information Science & Technology, Universiti Kebangsaan Malaysia.

Keywords: block matching; motion estimation; video signal;

1. Introduction

Video signal processing an important research area and has many applications in the industry and computer science such as online monitoring of assembly processes, robot navigation, medical treatment, multimedia broadcasting, remote sensing, and military. Analyzing the motion of natural moving objects in a scene is tremendously difficult in video processing. Determining the motion vectors used to detect the transformation from

* Corresponding author. Tel.: +6-038-947-1746 ; fax:+6-038-946-6576 .
E-mail address: razaliy@fsktm.upm.edu.my

2212-0173 © 2013 The Authors. Published by Elsevier Ltd. Open access under CC BY-NC-ND license.
Selection and peer-review under responsibility of the Faculty of Information Science & Technology, Universiti Kebangsaan Malaysia.
doi:10.1016/j.protcy.2013.12.181

one sequence to another, motion estimation (ME) is utilized. ME has many applications and has been proven essential for video coding and compression [1, 2].

There are different approaches to motion estimation, and these can be categorized into region matching, gradient based methods and transform methods [1]. In this paper, selected approaches are compared to discover their inherent advantages and disadvantages. Ultimately, the effects of macro blocks size and window size are examined to?. To compute motion compensation in video coding applications, the motions related to objects have to be estimated. The most commonly used ME technique in video coding is the block matching algorithm, mainly due to its simplicity and good performance[2]. Block matching assists to choose a motion vector for each macro block instead of using a motion vector for each pixel, and only one vector per a block of pixels is sufficient [3].

Generally, block matching involves finding a candidate block within a search region in a corresponding frame that best matches the current block in the current frame. The displacement between these matching blocks is called a motion vector (MV). [4]. The important aspect in block matching is the use of intelligent search strategies to reduce the computation time [5]. Different distortion measures are used to find the best match for a desired macro block in the entire motion estimation process. Mean absolute error (MAE), mean squared error (MSE), and sum of absolute differences (SAD) are commonly used to measure the efficiency of algorithms[4].

In this paper, the base of full search (exhaustive search, ES), New Three Step Search (NTSS), Simple and Efficient Search (SES), and Adaptive Rood Pattern Search (ARPS) are discussed. The main properties of these algorithms are compared. The effect of macro blocks size on Peak Signal to Noise Ratio (PSNR) and computation time are also examined.

2. Block matching motion estimation

The foremost point about block matching motion estimation is that there is high correlation between each pixel and its neighbors. Therefore, assigning a motion vector to a block of pixels is more useful than to an individual pixel. In the block matching motion estimation process, a frame is segmented into n × n blocks. Then, for the largest motion displacement of ‘p’ pixels per frame, the current block is matched with a corresponding block in the previous frame in same coordinates. The best match on base in matching principle gives the displacement.

The choice of n will have an effect on the output result and PSNR.1shows a candidate block and its neighbors. The procedure for seeking the best match yielding minimum error is also shown in Fig. 2. In this figure, the value of ‘p’ is the pixel space of the block, which is predetermined to search within the assumed best area for finding the best matching block. In this example, it is expanded to all sides of the macro block to obtain better results, where different values for ‘p’ are considered and the averages of the result are shown.

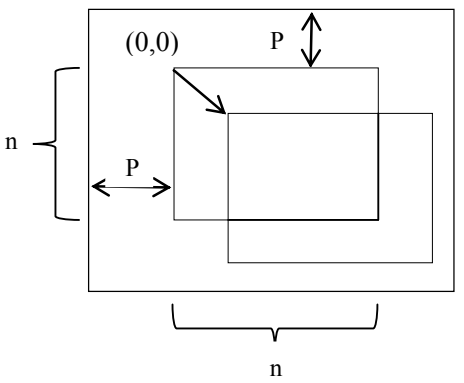

Fig. 1. A sample of candidate block and its neighbors (P pixels)

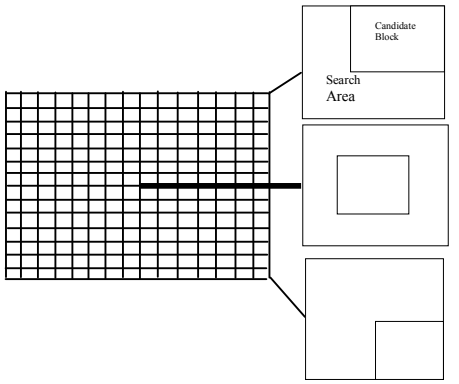

Fig. 2. Scanning process for candidate block in assumption area in different point

When the value of n is increased, the number of total blocks that needs to be processed in each frame will decrease. Therefore, the computational complexity decreases as well. However, finding a valid match for the given block is difficult, which also reduces the PSNR.

In general, the search area is constrained up to 'p' pixels on all four sides of the corresponding macro block in the frame to find the best match. The value of 'p' is defined as the search parameter. Large motions need a larger value for 'p', which makes searching for parameters more computationally expensive. In terms of block matching, two important subjects are considered, namely the matching criteria and the search algorithm.

3. Matching criteria

When parts of the image are examined pixel by pixel for image matching, block matching plays an important role to improve the efficiency. The goal of image matching is to determine the comparison between the images or portions of images. The similarity measure or correlation measure is a key element in the matching process. Finding the minimum dissimilarity or matching error rather than finding the maximum similarity or correlation, is a proficient way in block matching. Several matching criteria have already been proposed such as the Mean Square Error (MSE), Mean Absolute Differences (MAD), and Peak Signals to Noise Ratio (PSNR), which are represented in Equations. 1-3, respectively. "Eq. (1-3)", PSNR is most popular, and it determines the motion compensated image and is calculated based on motion vectors and macro blocks from the original frame [6].

$$MSE = \frac{1}{n^2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\left(C_{ij} - R_{ij}\right)^2 \tag{1}$$

$$MAD = \frac{1}{n^2}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\left|C_{ij} - R_{ij}\right| \tag{2}$$

$$PSNR = 10Log_{10}\left[\frac{255^2}{MSE}\right] \tag{3}$$

4. Searching algorithms

There are diverse methods available for searching. In this paper, four different algorithms are discussed.

4.1. Full search Exhaustive Search (ES)

Among the many block matching algorithms, the ES algorithm is the most computationally expensive. However, but the best match is found, so the highest PSNR will be produced compared to other methods. In this algorithm, the cost function at each possible location within the search area is calculated. There are total of (2P+1) × (2P+1) positions that should be investigated. The maximum similarity or minimum dissimilarity gives the finest match, and good searching accuracy providing the optimal match is delivered. However, because of the lengthy computation time, it is unsuitable for real-time video coding.

4.2. New Three-Step Search (NTSS)

The Three Step Search (TSS) algorithm has been established to reduce the number of candidate blocks within a search window. This algorithm uses a 3× 3 grid at each step, and when the minimum MAD is found, it is used as the motion vector. TSS has some problems because it uses a uniformly allocated search pattern in its first step, which it is not efficient at identifying small motions. As a result, the New Three Step Search (NTSS) technique was proposed.

NTSS assumes a center-biased point pattern to check and incorporate a halfway-stop technique. The algorithm has eight extra points compared to TSS; they are the eight neighbors of the search window center. The halfway-stop technique is used for stationary and quasi-stationary blocks. For this case, if in the first step search for the search window center, the minimum MAD occurs, then the search is stopped, and if one of the eight neighbors was the minimum MAD point during the first step, the search will be executed for eight neighboring points of the minimum and then stops the search [8].

4.3. Simple and Efficient Search (SES)

SES tries to solve some problem of TSS, and exploits the assumptions which are seemed for some types of errors. It is impossible to have two minimums in two opposite directions for a uni-modal surface, so it saves the computation time [10]. Like TSS, this algorithm has three steps. The difference is that the steps are further divided into two phases. In the first phase, a search quadrant is selected, and in the second phase, the location of the minimum error in the selected quadrant is found (Fig 3 and Fig 4).

fzTT2 1 Tf00 Td()TjGS1 gsTT-( )Tj

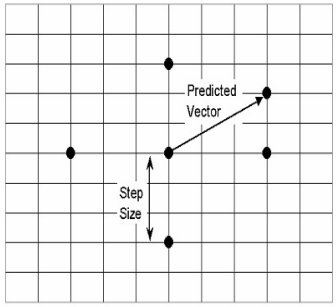

Fig. 5. The predicted motion vector is (3,2) and the step size is 3

5. Experimental results

To compare the efficiency of the mentioned algorithms, each has been implemented in MATLAB. Three different video clips from the TRAN database [8] were used for evaluation, which are WAVING, JUMPING, and RUNNING. Because the value of the macro blocks size and 'p' may have effects on the results, the implementations used macro blocks of different sizes namely 8, 16, 32, 64 and 'p' by the values of 7, 9, 11, 13, 15. The following results are the averages of the results by the different values for 'p' and macro block size. Search points per macro block have been used to represent the computation time. The computation time results for clips 'WAVING' and 'JUMPING' are shown in Fig 6 and Fig 7.In these figures, the computation time for ES has been eliminated for better visibility of the other results since this value is a constant number of approximately 13.8. The results were the same for the third video clip.

As shown in the Fig 6 and Fig. 7, the ES has the highest computation time, and ARPS has the lowest computation time. Another point is that the exact values of the search points per macro blocks depend on the movie but in the same movie the SE has the worst value and ARPS has the best value.

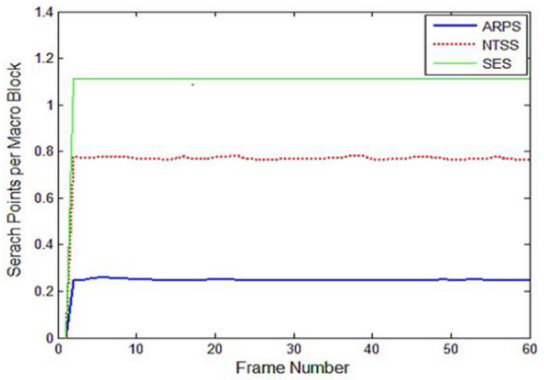

Fig. 6. Different computation time for movie 'waving'

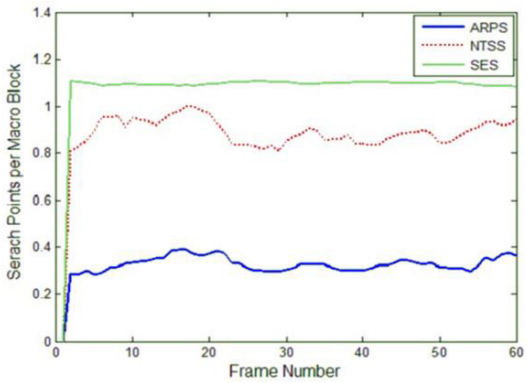

Fig. 7. Different computation time for movie 'jumping'

The simulation results for the PSNR are shown in the Fig. 8 and Fig. 9. It is seen that the computation time for ARPS is the best and ES is the worst. Another point is that the exact values for PSNR depend on the content of the frames and the rate of motion in the movie, but in different movies, the algorithms have almost the same results. Considering the computation time ARPS has the best result.

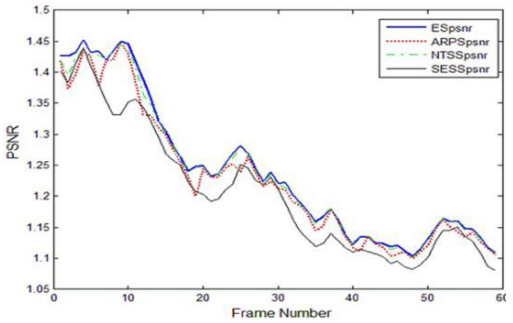

Fig. 9. PSNR results for different algorithms on the movie 'waving'

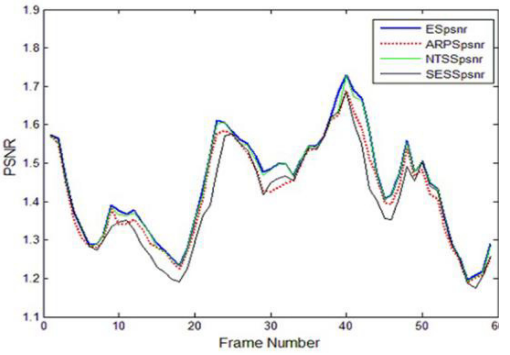

Fig. 8. PSNR results for different algorithms on the movie 'jumping'

The effects of size of macro blocks and the value of 'p' on PSNR and computation time are shown in Tables 1 and 2, respectively. Table 1 shows increased macro block size decreases PSNR. Considering 'P=7', when the size of the macro block (MB) is 8, the PSNR for NTSS is 37.38 and for ARPS is 37.36. When we change the size of the MB into 16, the PSNR for NTSS and ARPS are decreased to 36.50 and 36.54, respectively. Hence, by increasing the size of MB to 32, the PSNR decreases to 35.82 and 35.84.

Increasing the value of 'p' will lead to decreasing the value of PSNR. For example, considering 'MB = 8', increasing 'p' leads to increased PSNR. For the values of 'P = 7', 'P = 9','P = 11' and 'P = 13', the result of PSNR for the ARPS is 37.36, 37.54, 47.62, 37.67. Based on the results in Table 2, when we increase the size of MB, the computation time will decrease. For example, considering the 'P=7', when the size of MB is 8, the computation time for NTSS is 17.00 and for ARPS is 6.38. When the size of MB is increased to 16, the computation time for NTSS and ARPS are decreased to 16.94 and 5.29, respectively.

Table 1. Effects of size of macro block on PSNR for different methods

Mb	P	ES	SES	NTSS	ARPS
8	7	38.09	36.68	37.38	37.36
	9	38.43	36.68	37.38	37.54
	11	38.67	36.68	37.38	37.62
	13	38.84	36.68	37.38	37.67
16	7	36.82	36.79	36.50	36.54
	9	37.02	36.79	36.50	36.67
	11	37.18	36.79	36.50	36.75
	3	37.31	36.79	36.50	36.81
32	7	35.94	35.91	35.82	35.84
	9	36.01	35.91	35.82	35.89
	11	36.07	35.91	35.82	35.92
	13	36.11	35.91	35.82	35.94

Table 2. Effects of size of macro block on computation time PSNR

Mb	P	ES	SES	NTSS	ARPS
8	7	217.56	23.75	17.00	6.38
	9	348.30	23.75	17.00	6.38
	11	508.84	23.75	17.00	6.38
	13	699.72	23.75	17.00	6.39
16	7	213.84	21.11	16.94	5.29
	9	342.92	21.11	16.94	5.30
	11	502.34	21.11	16.94	5.30
	3	692.09	21.11	16.94	5.30
32	7	206.50	17.37	16.86	5.03
	9	330.97	17.37	16.86	5.04
	11	484.66	17.37	16.86	5.04
	13	667.57	17.37	16.86	5.04

6. Conclusion

In this paper, four search algorithms for block matching (ES, ARPS, NTSS, and SES) are compared from the aspects of PSNR and computation time. ES has the highest computation time and PSNR. This is because ES searches all probabilistic similarity. ARPS has logical computation time and PSNR compared to other algorithms. The effects of the value of 'p' and macro block size are also examined, and it is shown that by increasing the size of the macro blocks, the PSNR will be decreased. For future work, we will study the artificial search methods to improve the PSNR without increasing computation time.

References

[1] Dufaux, F. and J. Konrad, *Efficient, robust, and fast global motion estimation for video coding.* Image Processing, IEEE Transactions on, 2000. **9**(3): p. 497-501.

[2] Dufaux, F. and F. Moscheni, *Motion estimation techniques for digital TV: A review and a new contribution.* Proceedings of the IEEE, 1995. **83**(6): p. 858-876.

[3] Celebi, A., et al., *An all binary sub-pixel motion estimation approach and its hardware architecture.* Consumer Electronics, IEEE Transactions on, 2008. **54**(4): p. 1928-1937.

[4] Huska, J. and P. Kulla. *Trends in block-matching motion estimation algorithms*. 2008.

[5] Moshe, Y. and H. Hel-Or, *Video block motion estimation based on gray-code kernels.* Image Processing, IEEE Transactions on, 2009. **18**(10): p. 2243-2254.

[6] Bhaskaran, V. and K. Konstantinides, *Image and video compression standards: algorithms and architectures*. 1997: Springer.

[7] KHAMMAR, M., *Evaluation of different block matching algorithms to motion estimation.* IJVES, 2012: p. 148-153.

[8] Lins, R., E. Lima, and S. Melo. *Low Discrepancy Sequences Applied in Block Matching Motion Estimation Algorithms*. 2011: IEEE.

[9] Lu, J. and M.L. Liou, *A simple and efficient search algorithm for block-matching motion estimation.* Circuits and Systems for Video Technology, IEEE Transactions on, 1997. **7**(2): p. 429-433.

[10]Hui, Z., et al. *An Enhanced Adaptive Rood Pattern Search Algorithm for Fast Block-Matching Motion Estimation*. in *Image and Signal Processing, 2008. CISP '08. Congress on*. 2008.

[11]Tran, D. and A. Sorokin, *Human activity recognition with metric learning.* Computer Vision–ECCV 2008, 2008: p. 548-561.