Comparison of Different Motion Estimation Algorithms in Video Compression

Zar Zar Tun, Khin Htar Nwe
University of Information Technology (UIT), Myanmar
zarzarhtun@uit.edu.mm, khinhtarnwe@uit.edu.mm

Abstract

Everything that can be watched on a screen uses video coding which tends to reduce the amount of redundant video data in video compression. The idea behind video compression based on motion estimation is to save number of bits required for encoding the video. Many studies developed algorithms for storing, transmission and reducing complexity without affecting the visual quality. The more precise motion estimation results can obtain the more accurate compression technique. Actually, Motion Estimation not only can use in video compression, but also can deeply effective in different video processing applications as intelligent video surveillance, video content analysis, and video retrieval. In this paper, different Motion Estimation algorithms are described with related results which can lead to obtain effective ME algorithm.

1. Introduction

Motion information is essential for video compression and different video application systems as video content analysis, action recognition in video surveillance system, telemedicine and other video processing systems. Video coding tools are adapted for heterogeneous clients parallel in developing video communication technology. But, existing systems still require to be more compress and recognized data are to be more accurate and appearance invariant when abnormal activities are occurred. It is needed to get the most compressed Motion Estimation in video coding for reducing computational complexity, faster coding time and higher PSNR for real time applications.

Motion Estimation is the process of determining the motion vectors that describe the transformation of adjacent frames in a video sequences. The motion vectors may relate to the whole image or specific parts such as rectangular blocks, arbitrary shaped patches or even per pixel. Block matching algorithm is assumed that all the pixels within the blocks have same motion activity. Therefore, Block-based Motion Estimation technique can find the optimal motion vectors and it is faster than others. This paper can support in adaptation the block-based Motion Estimation technique to get balance trade-off in video coding without affecting the video quality. All are based on effective Motion Estimation algorithms.

This paper is organized as follows: Session 2 presents related work. Session 3 describes the background theory of motion estimation. Session 4 provides simulation experiments of different Motion Estimation algorithms and session 5 will follow the conclusion.

2. Related Work

There has been considerable effort devoted to different video processing based on video compression in the last decade. Developers adapted on Motion Estimation in different ways.

M. Ezhilarasan and P. Thambidurai [6] proposed a simplified and efficient Block Matching Algorithm for Fast Motion Estimation. Authors intended to minimize the search time on block matching. It had two steps such as prediction and refinement. The temporal correlation among successive frames and the direction of the previously processed frame for predicting the motion vector of the candidate block was considered during prediction step. Different combination of search points was considered in there finement step of the algorithm which subsequently minimize the search time. The experimental results were shown that the algorithm provided a faster search with minimum distortion when compared to the optimal fast block matching motion estimation algorithms.

X.Liyin, S.Xiuqin and Z.Shun [12] studied the low complexity ME algorithms and classified them into three categories, namely modeling the matching error surface, fast full search and reduction of searching candidate points. The aim of the review is to provide the succeeding researchers with some constructive information in design of the fast ME algorithms.

M.K.Pushpa and Dr.S.SethuSelvi [7] used two patterns for initial search and refined local search. For initial search, the proposed algorithm used a square pattern adaptively by selecting the step size based on Maximum Absolute Value of predicted motion vector. If the least error point is other than the middle point, then it becomes a new origin for subsequent refined local search with the pattern as small diamond. This is iteratively continued until the final motion vector is found.

Authors in [1] compared the existing block matching algorithms and gave their drawbacks. The applications of each algorithm were also discussed. The comparison was performed between the Exhaustive search (ES), Three Step Search (TSS), New Three Step Search (NTSS), Four Step Search (4SS), Diamond Search (DS), Hexagon-Diamond Search (HDS), Modified Diamond Search (MDS), Fast Diamond Search (FDS) and Orthogonal-Diamond Search (ODS).After review process it had been found that DS algorithm gives performance closer to the ES algorithm at minimum number of search points. Also the different variants of DS algorithms are also giving good results at an acceptable degradation in image quality.

Ms.B.Oatel, R.V.Kshirsagar and Dr.V.Nitnaware [8] presented advantages and disadvantages of different Block Based Motion Estimation Algorithms in three categories, as fast algorithms, true motion or good quality oriented methods and low computational complexity techniques algorithms. This paper tends to get the new design that can reduce the computational cost than traditional fast ME algorithms for researchers. According to comparison, Three Step Search (TTS) is less complex than other algorithms.

In paper [2], authors discussed various existing search patterns and proposed the implementation of New Combined Three Step Search pattern using Hexagon pattern, Linear pattern and Diamond Search. The simulation result shows average time saving of 50.07% of ME time compared with Three Step Search and 0.38% of ME time compared to existing Hexagon pattern in HEVC.

R.K.Akotkar and S.B.Kasturiwala [9] discussed about hybrid technique combination of two technique i.e. efficient three step search algorithm (E3SS) and cross hexagonal search algorithm (CHS). The experiment result shows that the proposal algorithm performs better than previous proposed block matching algorithms and required less computation. All reviews efficiently support to evaluate different block matching algorithms in this paper.

3. Background Theory

Multimedia processing needs large amounts of data as requiring large amounts of processing, storage, and communication resources. All requirements basically depend on the estimation of motion information in video coding.

3.1. Video Coding

Video coding is the technology behind moving digital images and it tends to reduce temporal redundancy between adjacent frames. Video coding complexity is reduced by motion estimation algorithm and therefore traditional motion estimation algorithms are adapted as new motion estimation algorithms in different ways to reduce encoding time, computational complexity and for higher Peak Signal to Noise Ratio (PSNR).

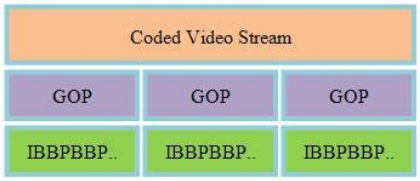

Figure 1. Group of Picture Structure

3.2 Inter Frame Prediction

In video coding, a group of pictures, or GOP structure, specifies the order in which intra and inter frames are arranged. At the partitioning stage of video encoding, an inter coded frame is divided into blocks known as macroblocks. Instead of directly encoding the raw pixel values for each block, the encoder will try to find position of the matching block on a previously encoded reference frame.

Figure 1 shows the general arrangement of GOPs that are group of I-Fame, P-Frame and B-Frame. I-Frame called intra coded frame that is the reference frame which is strictly inter coded and no need additional information to decode.

P-Frame called Predicted Frame that is used to define the forward predicted pictures from earlier reference frame and requires less coding data. P-Frame needs motion vector and transform coefficients describing prediction correction. B-Frame is the bi-directionally predicted pictures that can be interpolated from and earlier or later frame. B-Frame needs less coding time than P-Frame. B-

Frames are expressed as motion vectors and transform coefficients as P-Frame.

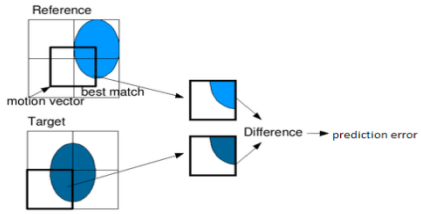

Figure 2. Motion Estimation Process

3.3 Motion Estimation

Motion estimation is the process of estimating the best match block in the previously reference frame to create the new frame. This matching block is motion vector and there is also a prediction error that is the result of difference between motion vector and transform coefficient. Figure 2 shows the process of Motion Estimation.

Motion Estimation is effective in removing temporal redundancies and it has become an integral part of all high-compression video codec. That can lead to effective widely used video applications.

3.4 Block Matching Technique

Block Matching Algorithm can estimate motion vectors on each block that can lead to faster motion estimation than pixel based. Block Matching Algorithm is also called as full-search algorithm because all candidate blocks in a search window are exhaustively searched to find best matching block.

The matching of one macro block with another is based on the output of a cost function [4]. Cost function decides the matching criteria of motion vector. The most popular less computationally expensive cost function is Mean Absolute Difference (MAD) given in equation (1):

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

Another one is Mean Squared Error (MSE) given by equation (2):

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

Where N is the size of the macro bock, C_{ij} and R_{ij} are the pixels being compared in current macro block and reference macro block, respectively. Peak-Signal-to-Noise-Ratio (PSNR) given by equation (3) which characterizes quality of motion compensated image.

$$\text{PSNR} = 10\log_{10}\frac{(\text{Peak to Peak Value of Original Data})^2}{\text{MSE}} \quad (3)$$

There are many algorithms have been developed, this paper describes some of the most basic or commonly used have been described and compared below. The parameters using in the following algorithms such as step size 'S', search parameter 'p' and location (i, j) are fixed in each algorithms.

3.4.1 Exhaustive Search (ES)

ES algorithm [1] calculates the cost function at each possible location in the search window. This leads to the best possible match of the macro-block in the reference frame with a block in another frame. The resulting motion compensated image has highest peak signal-to-noise ratio as compared to any other block matching algorithm. However this is the most computationally extensive block matching algorithm among all. A larger search window requires greater number of computations.

3.4.2 Three Step Search (TSS)

TSS [14] is one of the earliest fast block matching algorithms. It runs as follows:

1. Start with search location at center
2. Set step size 'S' = 4 and search parameter p' = 7
3. Search 8 locations +/- S pixels around location (0,0) and the location (0,0)
4. Pick among the 9 locations searched, the one with minimum cost function
5. Set the new search origin to the above picked location
6. Set the new step size as S = S/2
7. Repeat the search procedure until S = 1

The resulting location for S=1 is the one with minimum cost function and the macro block at this location is the best match. There is a reduction in computation by a factor of 9 in this algorithm. For p=7, while ES evaluates cost for 225 macro-blocks, TSS evaluates only for 25 macro blocks.

3.4.3 Two Dimensional Logarithmic Search (TDLS)

TDLS [14] is closely related to TSS however it is more accurate for estimating motion vectors for a large search window size. The algorithm can be described as follows,

1. Start with search location at the center
2. Select an initial step size say, S = 8
3. Search for 4 locations at a distance of S from center on the X and Y axes
4. Find the location of point with least cost function
5. If a point other than center is the best matching point,
   - 5.1 Select this point as the new center
   - 5.2 Repeat steps 2 to 3
6. If the best matching point is at the center, set S = S/2
7. If S = 1, all 8 locations around the center at a distance S are searched
8. Set the motion vector as the point with least cost function

3.4.4 New Three Step Search (NTSS)

TSS uses a uniformly allocated checking pattern and is prone to miss small motions. NTSS [10] is an improvement over TSS as it provides a center biased search scheme and has provisions to stop half way to reduce the computational cost. It was one of the first widely accepted fast algorithms and frequently used for implementing earlier standards like MPEG 1 and H.261.

The algorithm runs as follows:

1. Start with search location at center
2. Search 8 locations +/- S pixels with S = 4 and 8 locations +/- S pixels with S = 1 around location (0,0)
3. Pick among the 16 locations searched, the one with minimum cost function
4. If the minimum cost function occurs at origin, stop the search and set motion vector to (0,0)
5. If the minimum cost function occurs at one of the 8 locations at S = 1, set the new search origin to this location
   - 5.1 Check adjacent weights for this location, depending on location it may check either 3 or 5 points
6. The one that gives lowest weight is the closest match, set the motion vector to that location
7. If the lowest weight after the first step was one of the 8 locations at S = 4, the normal TSS procedure follows
   - 7.1 Pick among the 9 locations searched, the one with minimum cost function
   - 7.2 Set the new search origin to the above picked location
   - 7.3 Set the new step size as S = S/2
   - 7.4 Repeat the search procedure until S = 1

Thus this algorithm checks 17 points for each macro-block and the worst-case scenario involves checking 33 locations, which is still much faster than TSS.

3.4.5 Four Step Search (4SS)

Four Step Search [5] is an improvement over TSS in terms of lower computational cost and better peak signal-to-noise ratio. Similar to NTSS, FSS also employs center biased searching and has a halfway stop provision.

The algorithm runs as follows:

1. Start with search location at center
2. Set step size ‘S’ = 2
3. Search 8 locations +/- S pixels around location (0,0) as shown in figure
4. Pick among the 9 locations searched, the one with minimum cost function
5. If the minimum weight is found at center for search window:
   - 5.1 Set the new search origin
   - 5.2 Set the new step size as S = S/2 = 1
   - 5.3 Repeat the search procedure from steps 3 to 4
   - 5.4 Select location with the least weight as motion vector
6. If the minimum weight is found at one of the 8 locations other than the center:
   - 6.1 Set the new origin to this location
   - 6.2 Fix the step size as S = 2
   - 6.3 Repeat the search procedure from steps 3 to 4. Depending on location of new origin, search through 5 locations or 3 locations
   - 6.4 Select the location with the least weight
   - 6.5 If the least weight location is at the center of new window go to step 5, else go to step 6

3.4.6 Diamond Search (DS)

Diamond Search (DS) [11] algorithm uses a diamond search point pattern and the algorithm runs exactly the same as 4SS. However, there is no limit on the number of steps that the algorithm can take.

Two different types of fixed patterns are used for search,

- Large Diamond Search Pattern (LDSP)
- Small Diamond Search Pattern (SDSP)

The algorithm runs as follows:

- LDSP:

1. Start with search location at center
2. Set step size 'S' = 2
3. Search 8 locations pixels (X,Y) such that (|X|+|Y|=S) around location (0,0) using a diamond search point pattern
4. Pick among the 9 locations searched, the one with minimum cost function
5. If the minimum weight is found at center for search window, go to SDSP step
6. If the minimum weight is found at one of the 8locations other than the center, set the new origin to this location
7. Repeat LDSP

- SDSP:

1. Set the new search origin
2. Set the new step size as S = S/2 = 1
3. Repeat the search procedure to find location with least weight
4. Select location with the least weight as motion vector

Diamond Search algorithm has a peak signal-to-noise ratio close to that of Exhaustive Search with significantly less computational expense.

3.4.7 Adaptive Rood Pattern Search (ARPS)

ARPS [13] consists of two sequential search stages: 1)initial search and 2) refined local search. For each macroblock (MB), the initial search is performed only once at the beginning inorder to find a good starting point for the follow-up refined localsearch. By doing so, unnecessary intermediate search and the riskof being trapped into local minimum matching error points couldbe greatly reduced in long search case. For the initial search stage,an adaptive rood pattern (ARP) is proposed, and the ARP's sizeis dynamically determined for each MB, based on the available motion vectors (MVs) of the neighboring MBs. In the refined localsearch stage, a unit-size rood pattern (URP) is exploited repeatedly,and unrestrictedly, until the final MV is found.

3.4.8 Simple and Efficient Search on TSS (SES_TSS)

SES_TSS [3] extension to TSS and exploits the assumption of unimodal error surface. The main idea behind the algorithm is that for a unimodal surface there cannot be two minimums in opposite directions and hence the 8 point fixed pattern search of TSS can be changed to incorporate this and save on computations.

The search area is divided into four quadrants and the algorithm checks three locations as A, B and C. A is at the origin and B and C are S = 4 locations away from A in orthogonal directions. Depending on certain weight distribution amongst the three,the second phase selects few additional points.

Once the points have been selected to check for in second phase, find the location with the lowest weight and set it as the origin. Then change the step size similar to TSS and repeat the above SES procedure again until reach S = 1. The location with the lowest weight is then noted down in terms of motion vectors and transmitted.

4. Simulation Experiment

This paper compares the experimental results on ES, DS, 4SS, TSS, NTSS, ARPS and SESTSS. For experiment, macroblock size of 16*16 is used in a search range of 7 for each algorithm. The search range is assumed half of the macroblock size. Figure 3 shows the reference search range and current block in experiments.

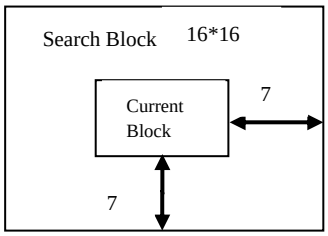

Figure 3. Search Range and Current

Figure 4 shows the sample sequences used in experiments. Figure 5 shows the comparison of resulted motion vectors for each algorithm between two adjacent current frame and reference frame. X

axis represents 2 frames as 1 and 2. Y axis represents motion vector values.

Table 1 shows the average number of searched points and processing time in seconds between the reference and current macroblock in decreasing order. According to experiment, Exhaustive Search algorithm can get maximum average number of searched points but it takes maximum processing time. Simple and efficient three step search (SES-TSS) can get the least number of searched points but with minium processing time.

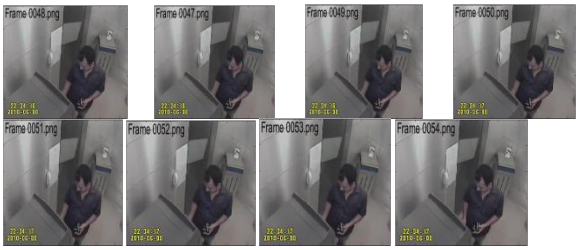

Figure 4. Sample Frame Sequences

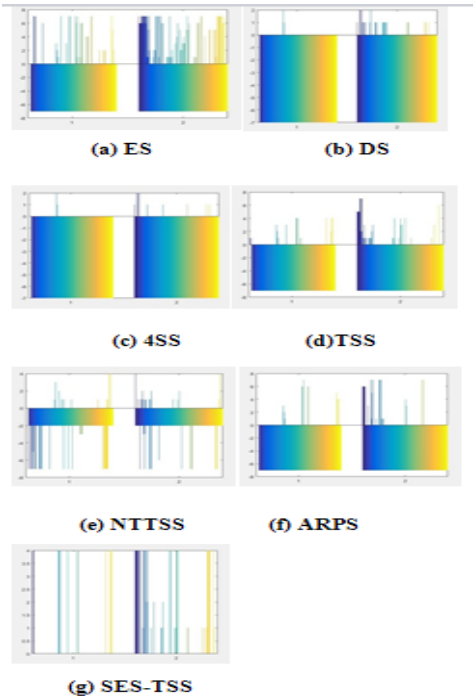

Figure5. Experimental Results of Motion Values: (a) ES (b) DS (c) 4SS (d) TSS (e) NTSS (f) ARPS (g)

Table 1. Comparison of Different Motion Estimation Algorithms

Block Matching Algorithms	ES	DS	4SS	TSS	NTSS	ARPS	SES-TSS
Searched Points	216.9273	31.9669	26.2868	24.5356	23.2761	15.0620	12.0718
CPU Processing Time (in seconds)	1.2231e+03	179.4480	147.7641	138.5913	120.4952	84.0065	67.6264

5. Conclusion

There is complex combination of translation and rotational motion in real video scenes. Large amount of processing is required to estimate vigorous motions for video coding that can lead to the growing market for video compression products and video processing applications. Motion Estimation is effective in removing temporal redundancy to get robust video compression technique. This study analyses and describes the nature of Motion Estimation algorithms and their respective suitable applications. It is hope that this study will become an applicable point for the researchers. As further research, this paper tends to find the most relevant block matching algorithm for different purpose video processing applications.

References

[1]. C.Pandey and D.Pandey , "Literature Review on Block Matching Motion Estimation Algorithms for Video Compression", *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, Volume 4, Issue 5, May 2015.

[2]. Davis P and S.Marikkannan , "Implementation of Motion Estimation Algorithm for H.265/HEVC", *International Journal of Advanced Research in Electrical Electronics and Instrumentation Engineering*, Volume 3, Issue 3, April 2014.

[3]. Jianhua Lu, and Ming L. Liou, "A Simple and Efficent Search Algorithm for Block-Matching Motion Estimation", *IEEE Trans. Circuits And Systems For Video Technology*, vol 7, no. 2, pp. 429-433, April 1997

[4]. L. Tao, Y.-ying, S, Gaopeng,"An improved three-step search algorithm with zero Detection and vector filter for motion estimation",*IEEE International Conference on Science and Software Engineering*, 2008.

[5]. Lai-Man Po, and Wing-Chung Ma, "A Novel Four-Step Search Algorithm for Fast Block Motion Estimation", *IEEE Trans. Circuits And Systems For Video Technology*, vol 6, no. 3, pp. 313-317, June 1996.

[6]. M. Ezhilarasan and P. Thambidurai, "Simplified block Matching Algorithm for Fast Motion Estimationin Video Compression", *Journal of Computer Science 4 (4)*, 282-289, 2008.

[7]. M.K.Pushpa and Dr.S.SethuSelvi, "Adaptive Square-Diamond Search(ASDS) Algorithm for Fast Block Matching Motion Estimation", *International Journal of Computer Science And Information Technologies*, Vol. 3 (5), 2012, 5247-5253, 2012.

[8]. Ms.B.Oatel, Dr.R.V.Kshirsagar and Dr.V.Nitnaware, "Review and Comparative Studey of Motion Esitmation Techniques to Reduce Complexity In Video Compression", *International Journal of Advanced Research in Electrical Electronics and Instrumentation Engineering*, Volume 2, Issue 8, Aug2013.

[9]. R.K.Akotkar and S.B.Kasturiwala, "Hybrid Approach for Video Compression Using BlockMatching Motion Estimation", *IEEE Sponsored World Conference on Futuristic Trends in Research and Innovation for Social Welfare (WCFTR'16)*, March 2016.

[10]. 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, Volume 4., no. 4, pp. 438-442, August 1994.

[11]. Shan Zhu, and Kai-Kuang Ma, " A New Diamond Search Algorithm forFast Block-Matching MotionEstimation", *IEEE Trans. ImageProcessing*, vol 9, no. 2, pp. 287-290, February 2000.

[12]. X.Liyin, S.Xiuqin, Z.Shun, "A Review of Motion Estimation Algorithms for Video Compression", *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*, Oct 2010.

[13]. Yao Nie, and Kai-Kuang Ma, "Adaptive Rood Pattern Search for FastBlock-Matching Motion Estimation", IEEE Trans. Image Processing,vol 11, no. 12, pp. 1442-1448, December 2002.

[14]. https://en.wikipedia.org/wiki/Block-matching_algorithm.