ISSN: 2321-7782 (Online)
Volume 2, Issue 3, March 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Paper / Case Study
Available online at: www.ijarcsms.com

Video Compression Algorithm Using Motion Compensation Technique: A Survey

Muhammad Aakif Shaikh[1]
Student, Master of Engineering in Electronics and Telecommunication
G. H Raisoni College of Engeneering & Management
Amravati – India

Sagar S. Badnerkar[2]
Assistant professor in Electronics and Telecommunication
G. H Raisoni College of Engeneering & Management
Amravati – India

Abstract: Video compression plays an important role in real-time video conferencing applications. Video compression techniques are about removing and reducing redundant video data so that a video file can be easily sent over a internet and stored on removal or computer disks. Now a days there are thousands website like you-tube in that storage for videos there are required big size hard disk but it is not possible to store the large data, So we use video compression using motion compensation technique. In video compression the Motion compensation is an algorithmic technique working for the encoding of video data. In Motion compensation describes a frame in form of the transformation of a reference frame to the current frame. The reference frame maybe even from the future or previous in time. When frames can be accurately synthesised from previously stored images, the video compression efficiency can be improved. Temporal redundancy is exploited so that not every frame of the video needs to be coded independently as a new image. So in this paper, Full Search strategies are used to reduce computation. Video compression techniques are used to reduce the redundancy in video data.

Keywords: Fixed size block motion estimation, Block-based motion estimation, Peak-Signal-to-Noise-Ratio (PSNR), block matching algorithm.

I. INTRODUCTION

Now a days there are videos in high definition or high quality qualities, So it requires a large transmission bandwidth and amount of storage space. To reduce the redundant data in video, there are various strategies to employ that compress the information without negatively affecting the quality of the frames. There are some methods which lossless and lossy, but in lossless in which there is no data is lost, but most are lossy, meaning that information is thrown away that can't be retrieved. So far our discussion on compression has been on still images. In the lossy techniques try to exploit the spatial correlation that exists in a still image. When we want to compress video or sequence images we have an added dimension to exploit, namely, the temporal dimension. Generally, there is little or very little change in the spatial arrangement of objects between two or more consecutive frames in a video. Therefore, it is advantageous to send over the network or store the differences between consecutive frames rather than sending or storing each frame. The difference frame is called the residual or differential frame and may contain far less details than the actual frame itself. Due to this reduction in the details in the differential frames, compression is achieved. To illustrate the above idea, let us consider compressing two consecutive frames of a video sequence . When objects move between successive frames, simple differencing will introduce large residual values especially when the motion is large. Due to relative motion of objects, simple differencing is not efficient from the point of view of achievable compression. It is more advantageous to determine or estimate the relative motions of objects between successive frames and compensate for the motion and then do the differencing to achieve a much higher compression. This type of prediction is known as motion compensated prediction. Because we perform motion *estimation* and compensation at the encoder, we need to inform the decoder about this motion compensation. This is done by sending motion vectors as side information, which conveys the object motion in the horizontal and vertical directions. The decoder then uses the motion vectors to align the blocks and reconstruct the image [1,10].

Video compression techniques are used to reduce redundancy in video data without affecting visual quality. It mostly used in video conference and real time application. In reality, motion compensation based coding are used in video compression techniques. Such encoders make use of inter-frame correlation to provide well-organized compression. For video compression we are using motion compensation method. In this Motion compensation technique, created In the 1960s, to exploit inter-frame redundancy contained in the temporal dimension of video sequence, is implemented in three stages. The first stage estimates object motion (motion estimation-ME) between the previously reconstructed frame and the current frame. The second stage creates the current frame prediction (motion compensation - MC), using the motion estimates and the previously reconstructed frame. The final stage differentially encodes the prediction and the actual current frame as the prediction error. Block transforms used in video encoders are unitary, which means that the transform operation has an inverse operation that uniquely reconstructs the original input. The DCT successively operates on 8 x 8 image blocks. Then, the quantization stage creates a lossy representation of the DCT Coefficients. The quantizer should be well matched to the characteristics of the input in order to meet or exceed the rate-distortion performance requirements.

Motion compensation technique engaged for video compression in the encoding of video data. Motion compensation describes the transformation of a reference frame to the current frame. The reference frame may be previous or even taken the later frames. When current frames can be accurately synthesized from previously transmitted or stored frames, the compression efficiency can be improved.

II. OBJECTIVES

A. Firstly we have to using algorithm for conversion of video into number of frames then further processes to find block matching in current frame form previous frame to develop a algorithm.

B. In motion estimation there is required to make a algorithm for searching a motion vector in frame, For compression of video for this developing algorithm in motion compensation for reduces video size.

C. At last there are developing algorithms for DCT and Quantization.

III. MOTION

Video compression is the field in Electronics engineering and computer science that works with representation of video data, for transmission and storage, for digital video. Video coding is often related with only for all type of natural video, and also applied to synthetic video, i.e. graphics. Many demonstrations take benefit of primitive features of the Human Visual System to accomplish an efficient demonstration. Using video compression, the biggest challenge is to retard the size of the video data. Due to this reason “video compression” and “video coding” are often used interchangeably by those who don’t know the difference. The finding for efficient video compression techniques dominated much of the research activity for video coding ought to 1980s, the first major milestone was H.261, from which JPEG gave the idea of using the DCT; since then other development have been made to algorithms such as motion compensation. Since ultimately 2000 the concentration has been more on Meta data and video search, resulting in MPEG-21 and MPEG-7.

IV. LITERATURE REVIEW

In Compression of video using motion compensation technique there are some methods for reducing and removing redundant video data for a digital video file can be stored in computer disk or memory. Below are methods of block matching for motion compensation technique used in previous research paper .

A. Exhaustive Search (ES)

This algorithm [1], also known as Full Search, is the most is computationally more costly for all algorithm of block matching. at each possible location in the search window calculates the cost function by this algorithm. By this method which gives the highest PSNR amongst any block matching algorithm and it finds the best possible match. To achieve the same PSNR doing as little computation as possible by Fast block matching algorithms. The obvious drawback to Exhaustive Search is that the larger the search window gets the more calculation it requires. The exhaustive search (ES) or full search algorithm gives the highest peak signal to noise ratio amongst any block-matching algorithm but requires more computational time [1]. To reduce the computational time of exhaustive search method.

B. Three Step Search (TSS)

This is one [1,8] of the previous seeking at fast block matching algorithms and dates back to mid 1980s. Technique of TSS. It begin by the finding the location at the centre and, for a usual find parameter value of 7 sets the ‘step size’ S = 4. Then it finds at eight locations +/- S pixels throughout location (0, 0). From these 9 locations found so far it put up the one giving least cost and makes it the new find origin. Then it sets the new step size S = S/2, and repeats similar finding for two more iterations until S = 1.

Where that point it finds the specific position the macro block at that location is the best match and with the smallest in cost function. The determination of motion vector is then stored for transmission. It gives a smooth reduction in computation by a factor of nine. So that for p = 7, Exhaustive Search will compute cost for 225 macro blocks whereas three step search computes cost for 25 macro blocks. The idea behind Three Step Search is due to motion in every macro block the error surface is unimodal. A unimodal surface is a round dish shaped surface such that the weights generated by the cost function makes greater monotonically from the global minimum.

C. New Three Step Search (NTSS)

NTSS [3,7] bring to more desirable state then three step search results by providing a centre biased finding scheme and having provisions for half way stop to diminish computational cost. It is first widely accepted fast algorithms and frequently used for implementing earlier standards like MPEG 1 and H.261. The three step search uses a identical allocated checking pattern for motion detection and is prone to absent small motions. 16 points are checked in the first step in addition to the find origin for lowest weight using a cost function. Of these supplementary find locations, 8 are a distance of S = 4 away (similar to TSS) and the other 8 are at S = 1 that place from the search origin. If the less cost is at the origin then the find is stopped right here and the motion vector is set as (0, 0). If the lowest weight is at any one of the 8 locations at S = 1, then we change the origin of the search to that point and check for weights adjacent to it.

Depending on which point it is we might end up value ate 5 points or 3 points .The location that gives the lowest weight is so much closest match and motion vector is set to that location. On different way if the lowest weight after the 1st step was one of the 8 locations at S = 4, then we follow the normal Three Step Search procedure. Hence although this process might need a minimum of 17 points to check every macro block, it also has the bad-case concept of 33 locations to investigate.

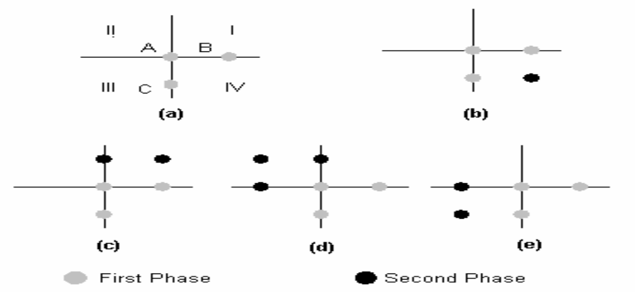

Fig. 3.1. Search patterns corresponding to each selected quadrant: (a)
Shows all quadrants (b) quadrant I is selected (c) quadrant II is selected (d) quadrant III is selected (e) quadrant IV is selected

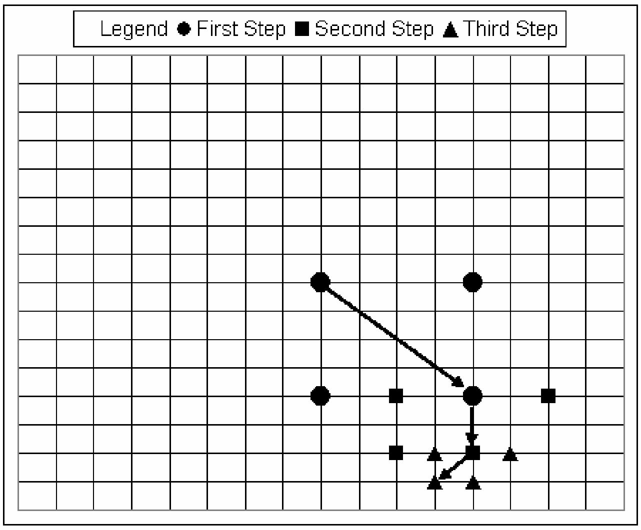

Fig. 3.2. The SES procedure. The motion vector is (3, 7) in this example

D. Simple and Efficient Search (SES)

SES [3,9] is another 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 algorithm quiet has three steps like Three Step Search, but the different introduced is that each step has further two stages. The search area is divided into four quadrants and the algorithm checks three locations A, B and C. A is at the origin and B and C are S = 4 locations some distance from A in orthogonal directions. Depending on certain weight distribution amongst the three the second phase selects few additional points. The rules for determining a search quadrant for seconds phase are as follows:

If MAD(A) _ MAD(B) and MAD(A) _ MAD(C), select (b);

If MAD(A) _ MAD(B) and MAD(A) _ MAD(C), select (c);

If MAD(A) < MAD(B) and MAD(A) < MAD(C), select (d);

If MAD(A) < MAD(B) and MAD(A) _ MAD(C), select (e);

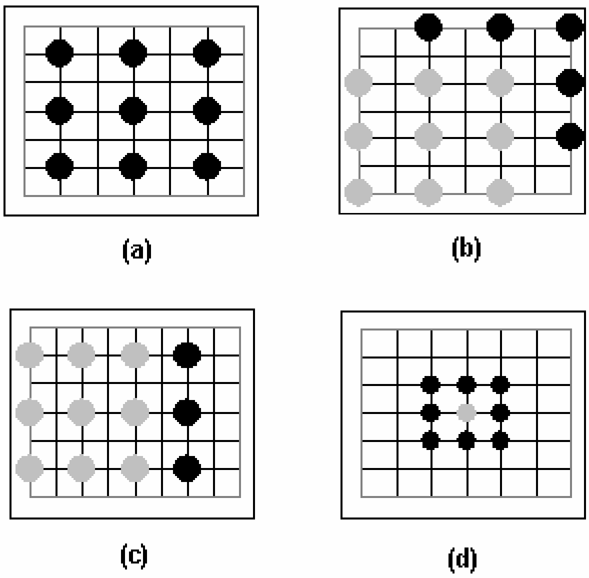

Fig. 4.1. Search patterns of the FSS. (a) First step (b) Second/Third step (c)Second/Third Step (d) Fourth Step

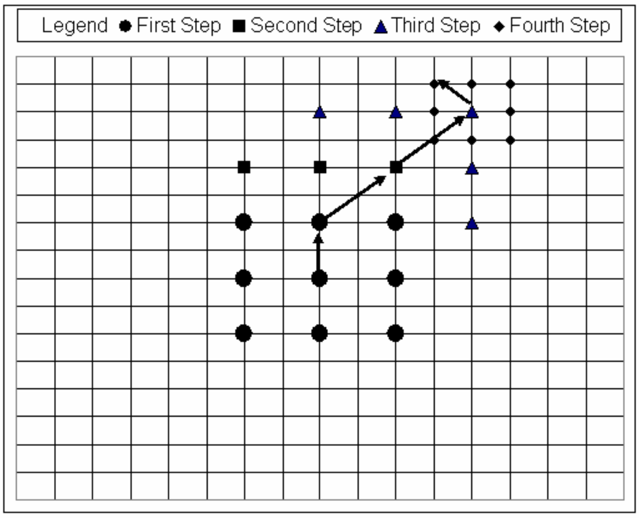

Fig. 4.2. Four Step Search procedure. The motion vector is (3, -7).

Once we have selected the points to check for in second phase, we find the location with the lowest weight and set it as the origin. We then change the step size similar to TSS and repeat the above SES procedure again until we reach S = 1.

The location with the lowest weight is then noted down in terms of motion vectors and transmitted. An example process is illustrated in Fig 6. Although this algorithm saves a lot on computation as compared to TSS, it was not widely accepted for two reasons. Firstly, in reality the error surfaces are not strictly unimodal and hence the PSNR achieved is poor compared to TSS. Secondly, there was another algorithm, Four Step Search, that had been published a year before that presented low computational cost compared to TSS and gave significantly better PSNR.

E. Four Step Search (4SS)

Same method to NTSS, 4SS [7,9] has a halfway stop provision and also employs centre biased searching. 4SS sets a fixed pattern size of S = 2 for the first step, no matter what the search parameter *p* value is. Thus it looks at 9 locations in a 5x5 window. If the least weight is found at the centre of search window the search jumps to fourth step. If the least weight is at one of the eight locations except the centre, then we make it the search origin and move to the second step. The search window is still maintained as 5x5 pixels wide.

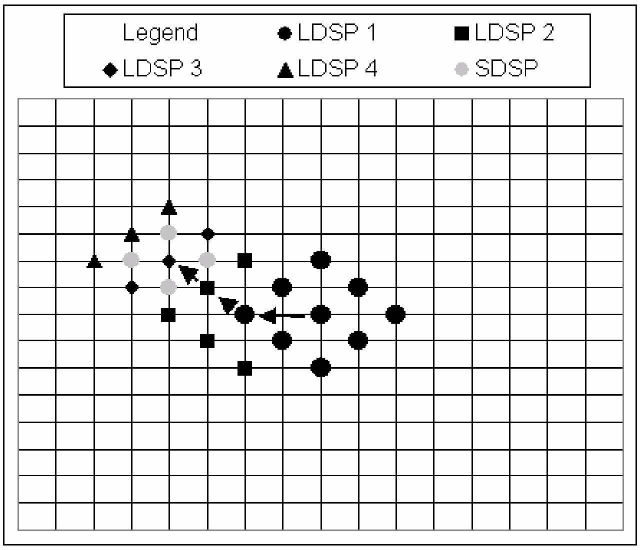

Fig. 5.1. Diamond Search procedure. This figure shows the large diamond search pattern and the small diamond search pattern. It also shows an example path to motion vector (-4, -2) in five search stepsfour times of LDSP and one time of SDSP.

Depending on where the least weight location was, we might end up checking weights at 3 locations or 5 locations. Once again if the least weight location is at the centre of the 5x5 search window we jump to fourth step or else we move on to third step. The third and second steps are exactly same. In the fourth step the window size is dropped to 3x3, i.e. S = 1. The location with the very small weight is the best matching macro block and the motion vector is set to point o that location. This search algorithm has the best case of 17 checking points and worst case of 27 checking points.

F. Diamond Search (DS)

DS [6] algorithm is exactly the same as 4SS, but the search point pattern is changed from a square to a diamond, and there is no limit on the number of steps that the algorithm can take.

DS uses two different types of fixed patterns, one is Large Diamond Search Pattern (LDSP) and the other is Small Diamond Search Pattern (SDSP). These two patterns and the DS procedures are illustrated in Fig. 9. Just like in FSS, the first step uses LDSP and if the least weight is at the centre location we jump to fourth step. The consequent steps, except the last step, are also similar and use Large Diamond Search Pattern, but the number of points where cost function is checked are either 3 or 5 and are illustrated in second and third steps of procedure shown in. Its location with the smallest weight is the best match and the last step uses SDSP around the new search origin and the. As look out the pattern is neither too big nor too small and the fact that there is no limit to the number of steps, this algorithm also find on global minimum very accurately. The end result should see a PSNR close to that of ES while computational expense should be significantly less.

G. Adaptive Rood Pattern Search (ARPS)

ARPS [10,13] algorithm apply for the common motion in a frame is usually related, i.e. if the macro blocks around the current macro block displaced in a particular direction then there is a more chances that the current macro block will also have a same motion vector. The motion vector of the macro block to it's suddenly left to predict its own motion vector uses this algorithm. The predicted motion vector points to (3, -2). In addition to checking the location pointed by the predicted motion vector, it also clarify at a rood pattern distributed points, where they are at a step size of S = Max (|X|, |Y|). X and Y are the x-coordinate and y-coordinate of the gives the motion vector. This rood pattern search is always the first step. It directly puts the search in an area where there is a high probability of finding a good matching block. The point that has the least weight becomes the origin for subsequent search steps, and the search pattern is changed to SDSP. The procedure keeps on doing SDSP until least weighted point is found to be at the center of the SDSP. A further small improvement in the algorithm can be to check for Zero Motion Prejudgment [8], using which the search is stopped half way if the least weighted point is already at the centre of the rood pattern. The main advantage of this algorithm over DS is if the gives motion vector is (0, 0), it utilize computational time in performing LDSP, it directly starts using Small Diamond Search Pattern. Moreover, if the motion vector that are predicted is far away from the origin, then again Adaptive Rood Pattern Search save on computations by directly goes to that neighbour sides and using Small Diamond Search Pattern, whereas DS takes its time doing Large Diamond Search Pattern. Carefully mandate that it has not repeat the computations at that points that were evaluate before. Care also needs to be taken when the predicted motion vector turns out to match one of the rood pattern locations. We have to avoid double computation at that point. Rood pattern step size is fixed at 2 pixels for macro blocks in the first column of the frame.

V. CONCLUSION

The A review of the various block matching based motion estimation algorithms has been presented. These algorithms are classified into three categories, namely, fast algorithms, true motion or good quality oriented methods and low computation complexity VSBME techniques algorithms. The features, advantages and disadvantages of these algorithms have been presented. In addition, we have also briefly reviewed the various matching, criterion such as MAE, MSE, or SAD. Algorithms are reviewed, in terms of both coding efficiency and computational complexity. Additionally, in order to design a low complexity ME algorithm, the researchers should jointly use the powerful techniques that are listed above to reduce the computational cost in different aspects. When used well together, the new design will significantly reduce the computational cost, as compared with the traditional fast ME algorithms.

Acknowledgement

The authors would like to thank Sagar S. Badnerkar for giving an idea about the development of system. His continuous guidance and motivation helps us for the development of system. Also we are gratefully acknowledging the support of all the IEEE papers which focuses on different parts of system.

References

1. Renxiang Li, Bing Zeng ; Liou, M.L, "A New Three-Step Search Algorithm For Block Motion Estimation", Circuits And Systems For Video Technology, Ieee Transactions On (Volume:4 , Issue: 4), Aug 1994.
2. Gary J.Sullivan and Thomas Wiegand, "Rate-Distortion Optimization for Video compression",IEEE Signal processing magzine, November 1998.
3. Sheckler, Y. Ozturk, and H. Abut, "Variable Size Block Motion Estimation",Signals, Systems & Computers, 1998. Conference Record of the Thirty-Second Asilomar Conference on (Volume:1), November 1998.
4. Omair Ahmad, M. N. S. Swamy, "Quadtree Structured Region-Wise Motion Compensation for Video Compression", IEEE Transactions On Circuits And Systems For Video Technology, Vol. 9, No. 5, August 1999.
5. X.Q. Gao, C.J. Duanmu, and C.R. Zou, "A multilevel successive elimination algorithm for block matching motion estimation," IEEE Transactions on Image Processing, vol. 9, no. 3, pp. 501-504, Mar. 2000.
6. Shan Zhu And Kai-Kuang Ma, "A New Diamond Search Algorithm For Fast Block-Matching Motion Estimation", IEEE Transactions On Image Processing, Vol. 9, No. 2, February 2000.
7. M. Brunig and W. Niehsen, "Fast full-search block matching," IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 2, pp. 241-247, Feb. 2001.
8. Dongkyun Park, Young Jang ; Jonghwa Lee, "A New Fast Three Step Search Motion Estimation Algorithm In H.264", Strategic Technology, 2007. IFOST 2007. International Forum On, 3-6 Oct. 2007.
9. Hui Zhao, Xin-Bo Yu ; Jia-Hong Sun ; Chang Sun, An Enhanced Adaptive Rood Pattern Search Algorithm For Fast Block-Matching Motion Estimation, Image And Signal Processing, 2008. CISP '08. Congress On (Volume:1), 27-30 May 2008.
10. Tao Zhang, Jianyu Wang, Pengfei He, "A New Motion Estimation Method and Its Application in Motion Compensation Deinterlacing Algorithms", Applied Informatics and Communication Communications in Computer and Information Science Volume 225, 2011, pp 344-351.
11. S. M. Reza Soroushmehr,, Shadrokh Samavi, Shahram Shirani,, "Simple and efficient motion estimation algorithm by continuum search", Multimedia Tools and Applications , Springer US, December 2012.
12. K.Leela Bhavani, R.Trinadh, "Architecture For Adaptive Rood Pattern Search Algorithm For Motion Estimation", International Journal of Engineering Research & Technology, Vol.1 - Issue 7 (September - 2012).
13. Yun Cheng, Tiebin Wu,Minlei Xiao , "A Modified Hexagon-Based Search Algorithm For Fast Block Matching Motion Estimation", Frontier And Future Development Of Information Technology In Medicine And Education Lecture Notes In Electrical Engineering Volume 269, 2014, Pp 1095.
14. K. S. Thyagarajan, "Still Image And Video Compression With Matlab", A John Wiley & Sons, Inc., Publication.
15. Jayanta Mukhopadhyay, "Image And Video Processing In The Compressed Domain", © 2011 By Taylor And Francis Group, Llc.
16. John C. Russ, "The Image Processing Handbook, Third Edition", Copyright © 1996-2000 Earthweb Inc, Pub Date: 07/01/98.

AUTHOR(S) PROFILE

Muhammad Aakif Shaikh received the BE degree in Electronics and Telecommunication and ME degrees in Electronics and Telecommunication from GH Raisoni Amravati, respectively.