PERFORMANCE ANALYSIS OF MOTION ESTIMATION ALGORITHMS BASED ON MOTION ACTIVITY IN VIDEO SEQUENCES

I. Ali, M. Muzammil* and G. Raja+

Department of Electrical Engineering, University of Engineering and Technology, Taxila
*Department of Electronic Engineering, International Islamic University, Islamabad
imdad.ali@ncp.edu.pk, m.muzammil@iiu.edu.pk, gulistan.raja@uettaxila.edu.pk

ABSTRACT: Motion Estimation (ME) is a critical part of video compression. This paper describes in-depth performance analysis of three most recent motion estimation algorithms based on motion activity in video sequences. Latest H.264/AVC video compression standard has been used for the performance analysis of three types of ME algorithms, which are: Enhances Predictive Zonal Search (EPZS), Unsymmetric-cross Multi Hexagon-grid search (UMHex) and Simplified Unsymmetric-cross Multi Hexagon-grid search (SUMHex). Performance of these algorithms is evaluated based on ME time and Mean Square Error (MSE). Results show that SUMHex has taken minimum ME time for low activity video sequences while EPZS took lowest time for the video sequences having high and moderate motion activities, whereas MSE is approximately same for all the algorithms.

Key words: Motion Estimation, H.264/AVC, EPZS, UMHex, SUMHex.

INTRODUCTION

H.264/AVC is the latest standard for video compression. H.264 video format has a wide range of application from low bit rate (internet streaming) to High Definition Television (HDTV) broadcast. Motion Estimation (ME) is a critical part of video compression which is used to identify the motion in a video. ME is a process to estimate the pixels of the current frame from a reference frame (Cheung and Po, 2002). ME algorithms are used to compute the displacement between current frame and reference frame in the video codec. Normally, the previous frame is considered as a reference. The intensity value of a pixel in the current frame and in reference frame has some correlation with its neighborhood which determines the best matching position of pixels intensity values in the reference frame. Where best match is found, difference in positions of current frame and the reference frame is calculated. This difference is defined as the displacement vector or more commonly known as the Motion Vector (MV).

There are two basic approaches to find out the ME, first one is called pixel based ME while the second is block based ME. The pixel based ME technique is also known as the optical flow method. It works on the basic assumption of brightness constancy which states that the intensity of a pixel remains constant when it is displaced in the video sequence. In this scheme, MVs are determined for every pixel in the frame. In block-based ME approach, the current frame is divided into non-overlapping blocks (for example, the seven possible modes of a block like 16×16, 16×8, 8×16, 8×8, 8×4, 4×8, 4×4 are used in H.264) (Ahmadi and Azadfar, 2008). These are called macro-blocks and for each such current frame block, one best MV is calculated in the reference frame (Pascalis *et al*., 2004). Here an inherent assumption is made that the entire block undergoes translational motion. The algorithms which use block-based ME technique are known as block matching ME algorithms. Block based ME algorithms remove temporal redundancy between two or more successive frames, and are integral part for most of the motion-compensated video coding standards.

ME is the most computationally expensive process in video compression. In case of one reference frame, the ME process takes more than 50% of total encoding time. Moreover, as the number of reference frames increase, the relative computational cost of ME process increases gradually. For example, for four reference frames computational cost of ME is about 70% of encoding cost (Barjatya, 2004). Due to such high computational cost, this field has gone through a tremendous research activity. There are many type of ME algorithms such as Three Step Search (TSS) (Koga *et al*., 1981), New Three Step Search (NTSS) (Li and Liou, 1994), Four Step Search (FSS) (Po and Ma, 1996), Simple and Efficient Search (SES) (Lu and Liou, 1997), Diamond Search (DS) (Zhu and Ma, 1997), Adaptive Rood Pattern Search (ARPS) (Nie and Ma, 2002) and Cross Diamond Search (CDS) (Lam *et al*., 2004) etc. Latest ME algorithms used in H.264/AVC are Enhanced Predictive Zonal Search (EPZS), Unsymmetric-cross Multi Hexagon-grid search (UMHex search) and Simplified Unsymmetric-cross Multi Hexagon-grid search (SUMHex search). An early termination for SUMHex search is also proposed to decrease the computational cost (Merritt and Vanam, 2007).

Different parameters, like Mean Squared Error (MSE), Mean Absolute Error (MAE), Mean Absolute

Deviation (MAD), Pixel Difference Correlation (PDC) and Peak Signal to Noise Ratio (PSNR) are used to evaluate the performance of ME algorithms (Subramanya *et al*., 2004). In this paper, performance of three latest ME algorithms (EPZS, UMHex search and SUMHex search) is evaluated in consideration with motion activity of different video sequences.

OVERVIEW OF MOTION ESTIMATION ALGORITHMS

H.264/AVC is a latest standard for video compression in which intra frame (I-frame) and inter frame (P-frame) are used for generating output bit stream. I-frame predicts the information only from the current frame while Inter frame predicts information from previous and next frames using ME algorithm. When prediction is performed from the previous frames, inter frame is called P-frame (previous frame) and in case the prediction is performed from the next frame, inter frame is called B-frame (bidirectional frame). The encoder first predicts I-frame and predicts one P-frame then predicts seven B-frames, I-frame is encoded once, at the start of encoding process. Remaining frames are predicted with the sequence PBBBBBBB_PBBBBBBB_ and so on. This encoding process is continued till the whole video is encoded.

The block diagram of H.264/AVC video compression standard is shown in Fig-1 in which first frame of input video sequence is encoded directly by using entropy coding after Discrete Cosine Transform (DCT) and quantization process. This frame is I-frame which is saved after inverse quantization and Inverse DCT (IDCT) process for inter motion compensation and ME process. In ME process the saved I-frame is taken as previous frame, while the new upcoming frame from input video sequence is called current frame. The ME process is carried out in ME block by using ME search algorithms such as EPZS, UMHex and SUMHex etc. This process gives MVs for inter frame motion compensation and entropy coding block that generate coded bit stream.

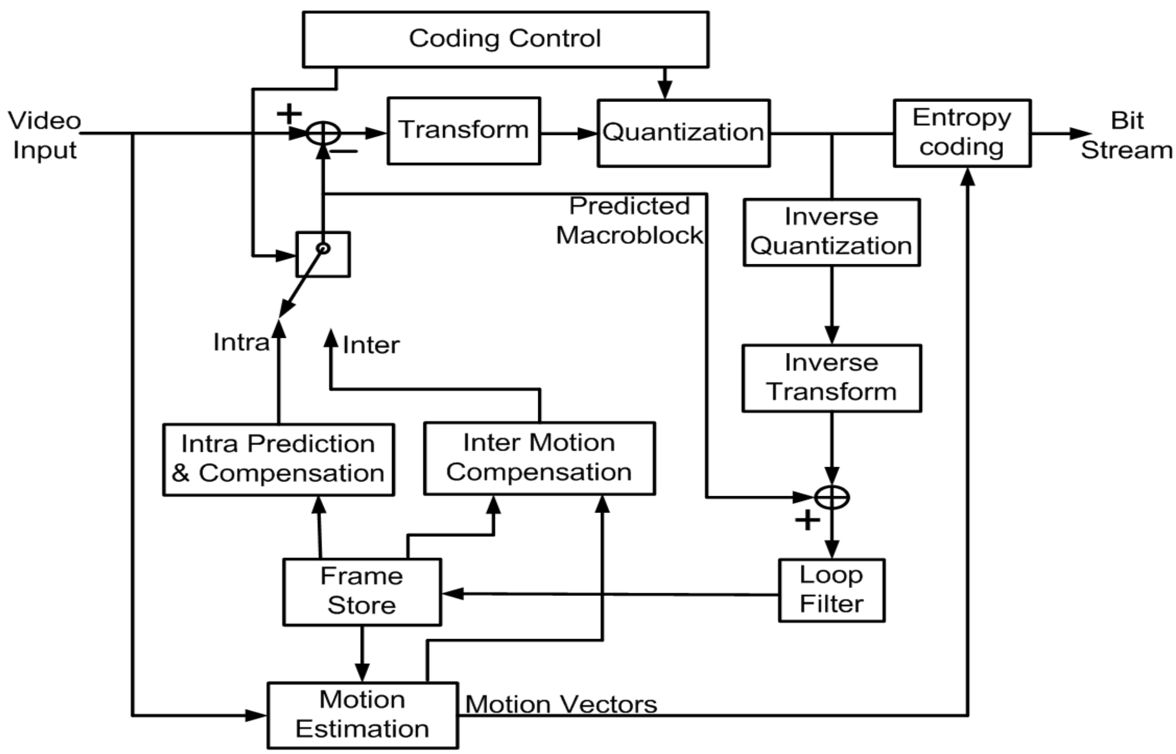

Fig-1: Block diagram of H.264/AVC standard.

Three ME search algorithms, EPZS, SUMHex search and UMHex search are evaluated for H.264/AVC standard. These algorithms are explained below in detail.

Enhanced Predictive Zonal Search (EPZS*)* Algorithm: EPZS is an improvement of Predictive Motion Vector Field Adaptive Search Technique (PMVFAST) and Advanced Predictive Diamond Zonal Search (APDZS) (Tourapis *et al*., 2001; Tourapis *et al*., 2001-a). Predictors are used to perform the prediction process, which selects the best starting point to accelerate the process to find the best matching block. An additional set of predictors to reduce the overhead is introduced in this algorithm. EPZS uses the fact that MVs are highly correlated with the MVs of neighboring blocks in the previous frames. MV of the adjacent macro blocks, their median, the (0,0) MV, and the MV of the correlated block in the previous frame at time T-1 are used for predictor subsets. There are three types of predictor subsets which are: subset A, subset B and subset C. Predictor subsets A and B are used for PMVFAST and APDZS, while predictor subset C is used for EPZS to find the prediction values (Tourapis, 2002). The prediction values introduce complexity in ME process which is reduced by the thresholding process. Pseudo code for the determination of threshold value is given below:

Examine all predictor in subset A;
if*(current min. difference satisfies threshold T1)*
GOTO *Refinement_S;*
else
Examine all predictor in subset B;
if*(current min. difference satisfies threshold T2)*
//T2 is calculated adaptively
GOTO *Refinement_S;*
else
Examine all predictor in subset C;
if*(current min. difference satisfies threshold T3)*
//T3 is same as T2
end*;*
Refinement_S:
continue by using refinement pattern;
//apply diamond or square search patterns

After thresholding process, square search pattern is used for the calculation of minimum difference point. Square search is applied until the best match is found at the center point of square as shown in Fig-2. Square search pattern, containing nine search points is applied to determine the minimum difference point as shown in Fig-2(a). If the minimum difference point is not at the center of the search pattern, again same square search pattern is applied at the previous minimum difference point as shown in Fig-2(b). Same search pattern is applied repeatedly until the best matching point is found at the center point of the search pattern, as shown in Fig-2(c)

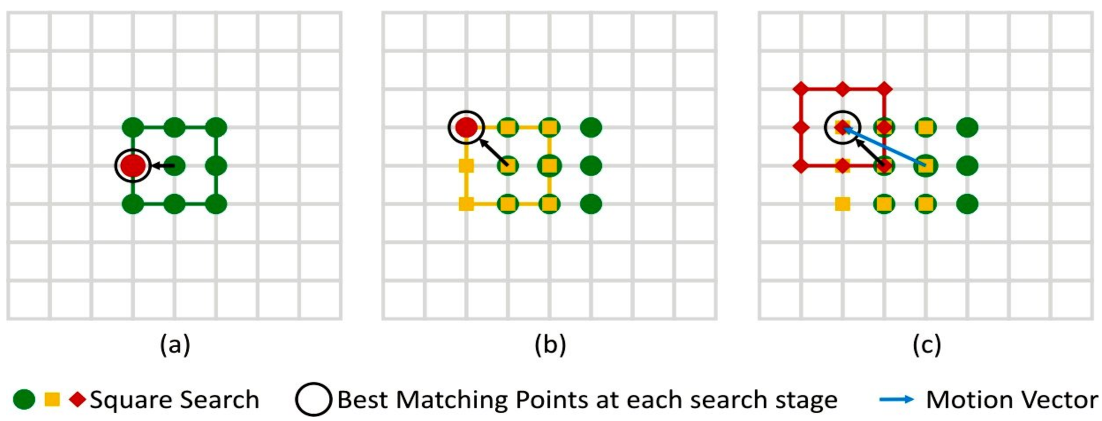

Fig-2: Square search patterns for EPZS (Tourapis, 2002).

UMHexagon Search Algorithm: Unsymmetric-cross Multi-Hexagon-Grid Search gives a very good quality than Three Step Search (TSS), Diamond Search (DS) or hexagon based search. In UMHexagon search algorithm, prediction of initial search point is done by using different types of predictors. One of the predictors used in UMHexagon search algorithm is median MVs predictor (Toivonen and Heikkila, 2006).

After initial search, it takes following steps to find the MV. In first step, find the best point for next search using asymmetrical cross search that follows early termination scheme. In second step, uneven multihexagonal-grid search is carried out which is done by using square search pattern and then a sixteen point hexagon search is applied. After this step where minimum difference is obtained, hexagon search technique is applied until the minimum difference point is found at the center of hexagon. Finally, small diamond search is applied at the center of Hexagon to find the best match (Xu and He, 2008). These search steps are illustrated in Fig-3 and the pseudo code of UMHex is as follows:

Predict initial search point; / spatial median prediction, upper layer prediction & temporal prediction */*
Apply asymmetrical cross search;
If*(best match found)*
end*;*
else
{Apply square search; // uneven multihexagonal-grid search
Apply sixteen point hexagon search; //uneven multihexagonal-grid search
while*(best match in not at the center of hexagon)*
Keep applying hexagon search;
Apply small diamond search;}

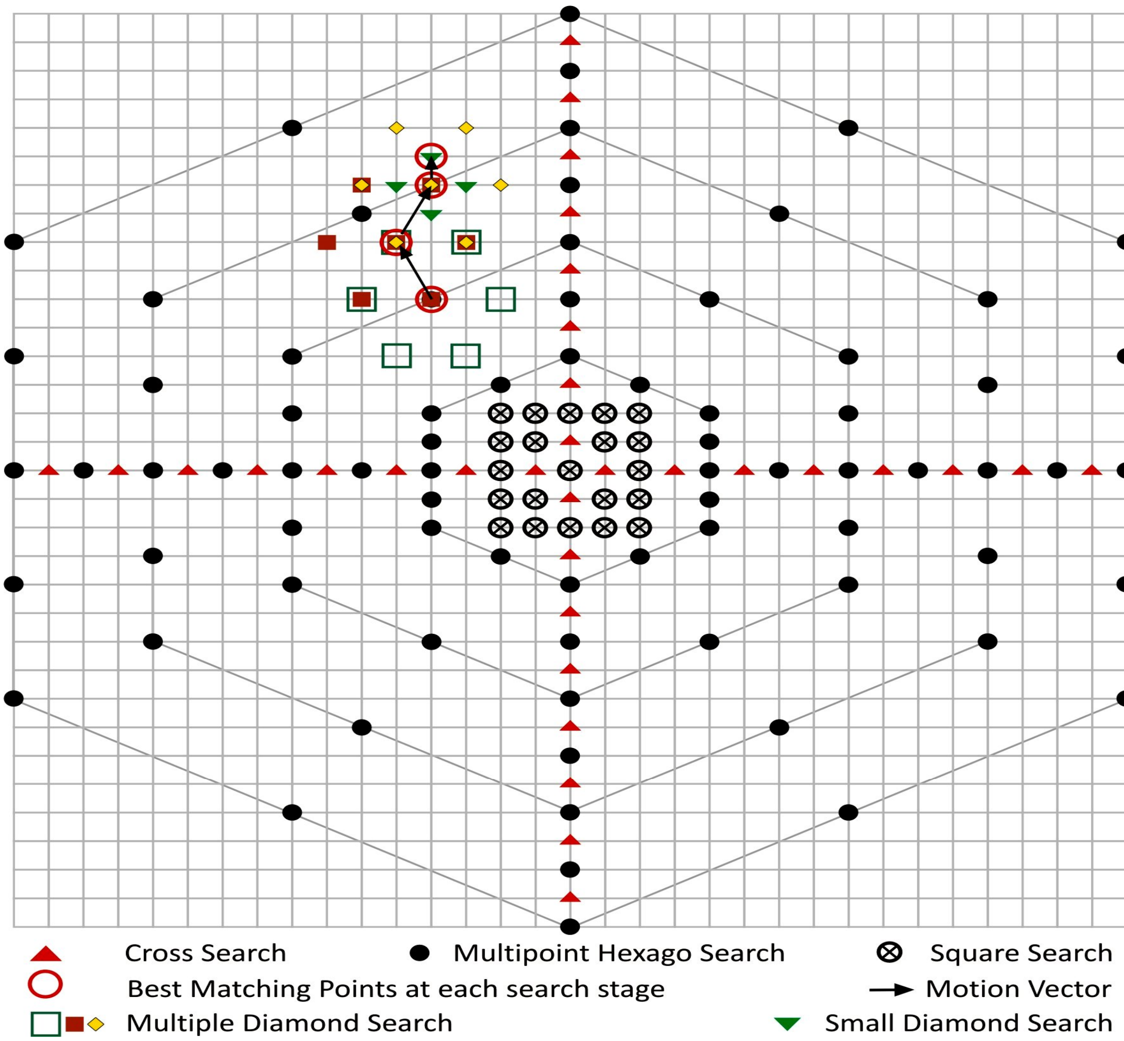

Fig-3: Search process of UMHex search algorithm (Toivonen and Heikkila, 2006).

SUMHexagon Search Algorithm: The simplified unsymmetric-cross multi-hexagon-grid search algorithm uses only up-layer predictor for MV prediction. After predicting the MV, it checks for the convergence condition; if this condition is fulfilled, convergence step is performed to find the minimum difference point, otherwise cross search is applied. Hexagon and multi-big hexagon searches are applied if intensive search condition is satisfied. Then up-layer prediction search and small local search are applied and again convergence condition is checked. In case convergence condition is not satisfied, extended hexagon search, extended diamond search and convergence search are applied (Xu and He, 2005). The SUMHexagon Search Algorithm adopts the following pseudo code to find the minimum difference point:

Check predictors; *//only up-layer predictor is use*
if*(Converge condition is true)* *//Simple threshold value used*
apply Convergence Search;
else
*{****if****(intensive search condition is true)*
{ apply cross search;
apply hexagon & multi-big hexagon search;
Check_Conv_Condition:
apply up-layer prediction search;
apply small local search;
if*(converge condition is true)*
apply convergence search;
else
{apply extended hexagon search;
apply extended diamond search;
apply convergence search;}
}
else
{apply up-layer predictor search;
GOTO *Check_Conv_Condition;*
}
}

SIMULATION SETUP AND RESULTS

JM 17.2 open source software (Karsten, 2011) for H.264/AVC is used to find the ME time and MSE of the video sequences having different amount of motion activity (QCIF video, 2011). Specifications of the computer used for the simulation are: Intel® 2.1GHz Core™2 Duo Processor, 2MB L2 cache, 1GB 667MHz DDR2 RAM, 128MB memory of video card and 32bit Windows®7 operating system. Parameters set in the configuration file of the encoder of JM17.2 are given in the Table-1.

Table-1. Parameters of JM17.2 used for simulation

Parameter	Value
Number of frames encoded	100
Image format	QCIF (176 x 144)
YUV format	4 : 2 : 0
Frame rate	30 fps
Bit rate	45020 bps
Group of Picture (GOP) structure	I P B B B B B B B
Search range	32
Entropy mode	CABAC
Rate distortion optimization	Low complexity mode

To acquire better results after the simulation, 100 frames are used to encode each video sequence. Quarter Common Intermediate Format (QCIF) videos are used because these have smaller frame size thus fewer search blocks for ME. For the enhanced compression of colored YUV video format, YUV 4:2:0 format is used. ME algorithms are analyzed according to motion activity in video sequences and the motion activity is compared according to Table-2.

Table-2. Classification of motion activity in video sequences (Raja *et al*., 2008)

Video Sequence	Classification of Motion Activity	Motion Direction
Container	Low	Unidirectional
Akiyo	Low	Unidirectional
Miss America	Low	Unidirectional
Claire	Low	Unidirectional
Bridge Close	Low	Unidirectional
Hall	Low	Unidirectional
Mother Daughter	Moderate	Unidirectional
Coastguard	Moderate	Bidirectional
Car Phone	Moderate	Unidirectional
Mobile	Moderate	Multidirectional
Foreman	Moderate	Multidirectional
Soccer	High	Multidirectional
Football	High	Multidirectional

Motion Estimation Time: Simulation results given in Fig-4 show the ME time required to encode different video sequences. For video sequences having high and moderate motion activity, EPZS takes noticeable less time as compare to other two ME algorithms. For high motion activity video sequences, EPZS takes approximately 45% less time than the UMHex and SUMHex which reduces the computational cost extensively. For video sequences having moderate motion activity, EPZS consumes 20% less time as compare to UMHex and SUMHex.

For the video sequences having low motion activity, SUMHex performs slightly better than EPZS and UMHex. For video sequence having low motion activity in more than one direction e.g. “Bridge Close.YUV”, EPZS performs better than the remaining two algorithms. For any type of motion activity, video sequences having motion in more than one direction, UMHex performs better than SUMHex e.g. "Mobile.YUV" and "Foreman.YUV".

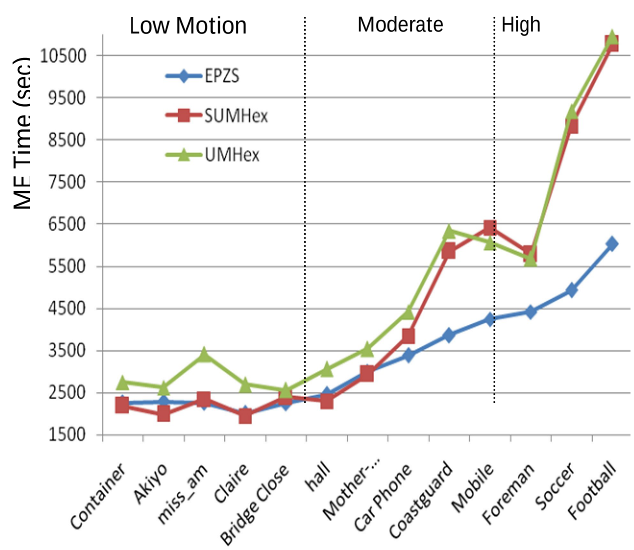

Fig-4. Motion Estimation time required to encode 100 frames

Mean Square Error: Simulation results given in Fig-5 show the average MSE calculated by encoding 100 frames of each of different video sequences. For video sequences having motion activity in more than one direction, MSE becomes high for all ME algorithms. MSE can be calculated by using following Eq. (1):

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

Where N is the number of pixels for one side of block, C_{ij} is the specific pixel of current frame and R_{ij} is the specific pixel of reference frame.

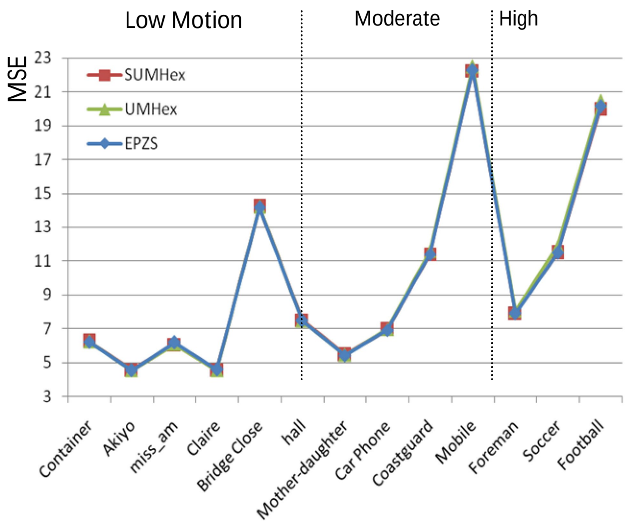

Fig-5. Average MSE of 100 frames

Conclusion: For the simulation results given in the preceding section, we can conclude that MSE of all the three ME algorithms is approximately equal. However, difference in the ME time for different type of video sequence is significant. EPZS takes less ME time to encode the video sequences having moderate, high and multidirectional motion activities. UMHex performs better in video sequences having low multidirectional motion activity. Table-3 shows the recommended motion estimation algorithms for different types of motion activity video sequences.

Table-3. Recommended motion estimation algorithms to be used for various video sequences

S No.	Type of Motion Activity	Recommended ME Algorithm
1	Low, unidirectional	SUMHex
2	Moderate, unidirectional	EPZS
3	High, unidirectional	EPZS
4	Moderate, multidirectional	EPZS or UMHex
5	High, multidirectional	EPZS

REFERENCES

Ahmadi, A., and M. M. Azadfar. Implementation of fast motion estimation algorithms and comparison with full search method in H.264. Computer Science & Network Security J., 8:139-143 (2008).

Barjatya, A., Block matching algorithms for motion estimation. Technical Report, Utah State University (2004).

Cheung, C. H., and L-M Po. A novel cross-diamond search algorithm for fast video coding and video conferencing applications. IEEE Trans. on Circuits & Systems for Video Technology, 12:1168-1177 (2002).

Karsten, S. H.264/AVC software coordination, joint model software version 10.2. Fraunhofer Institute for Telecommunications, Heinrich Hertz Institute, Germany (2011).

Koga, T., K. Iinuma, A. Hirano, Y. Iijima and T. Ishiguro. Motion compensated interframe coding for video conferencing. Proc. Nat. Telecom. Conf., New Orleans, USA: G5.3.1-G5.3.5 (1981).

Lam, C. W., L-M. Po and C. H. Cheung. A novel kite-cross-diamond search algorithm for fast block matching motion estimation. Proc. IEEE Intl. Symp. Circuits & Systems, 3:729-732 (2004).

Li, R., B. Zeng and M. L. Liou. A new three-step search algorithm for block motion estimation, IEEE Trans. on Circuits & Systems for Video Technology, 4:438–442 (1994).

Lu, J., and M. L. Liou. A simple and efficient search algorithm for block-matching motion estimation, IEEE Trans on Circuits & Systems for Video Technology, 7:429–433 (1997).

Merritt, L., and R. Vanam. Improved rate control and motion estimation for H.264 encoder, IEEE Intl. Conf. on Image Processing, San Antonio, TX, USA: V: 309–312, (2007).

Nie, Y., and K. K. Ma. Adaptive rood pattern search for fast block-matching motion estimation, IEEE Trans. on Image Processing, 11:1442–1449 (2002).

Pascalis, P. D., L. Pezzoni, G. A. Mian and D. Bagni. Fast motion estimation with size-based predictors selection hexagonal search in H.264/AVC encoding. Proc. 12th European Signal Processing Conf., Vienna, Austria: 273-276 (2004).

Po, L-M., and W. C. Ma. A novel four-step search algorithm for fast block motion estimation, IEEE Trans. on Circuits & Systems for Video Technology, 6:313–317 (1996).

QCIF Video. YUV test video sequences, video trace library, Arizona State University, USA (2011).

Raja, G., M. J. Mirza and T. Song. H.264/AVC deblocking filter based on motion activity in video sequences. IEICE Electronics Express J., 5:809-814 (2008).

Subramanya, S.R., H. Patel and I. Ersoy. Performance evaluation of block-based motion estimation algorithms and distortion measures, Proc. Intl. Conf. on Information Technology: Coding & Computing, Las Vegas, USA, 2: 2–7 (2004).

Toivonen, T., and J. Heikkila. Improved unsymmetric-cross multi-hexagon-grid search algorithm for fast block motion estimation. Proc. IEEE Intl. Conf. on Image Processing, Atlanta, GA, USA: 2369-2372 (2006).

Tourapis, A. M., Enhanced predictive zonal search for single and multiple frame motion estimation. Proc. SPIE Conf. on Visual Communications and Image Processing, San Jose, CA, USA: 1069-1079 (2002).

Tourapis, A. M., O.C. Au and M.L. Liou. Predictive motion vector field adaptive search technique (PMVFAST) – enhancing block based motion estimation. Proc. Visual Communications & Image Processing, San Jose, USA: 883-892 (2001).

Tourapis, A. M., O.C. Au and M.L. Liou. New results on zonal based motion estimation algorithms – advanced predictive diamond zonal search. Proc. IEEE Intl. Symp. on Circuits & Systems, Sydney, Australia, 5: 183–186 (2001-a).

Xu, X., and Y. He. Improvements on fast motion estimation strategy for H.264/AVC. IEEE Trans. on Circuits & Systems for Video Technology. 18:285–293 (2008).

Xu, X., and Y. He. Comments on motion estimation algorithms in current JM software. Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6 17th Meeting), Nice, FR: (2005).

Zhu, S., and K. K. Ma. A new diamond search algorithm for fast block matching motion estimation, Proc. Intl. Conf. on Information, Communications and Signal Processing, Singapore: 292–296 (1997).