A Survey on Video Compression Fast Block Matching Algorithms

A. J. Hussain[a,*], Zaynab Ahmed[b]

[a]*Applied Computing Research Group,*
Department of Computer Science, Liverpool John Moores University, Byrom Street,
Liverpool, L3 3AF, UK
[b]*University of Baghdad*
College of Science for Women
University of Baghdad

ABSTRACT

Video compression is the process of reducing the amount of data required to represent digital video while preserving an acceptable video quality. Recent studies on video compression have focused on multimedia transmission, videophones, teleconferencing, high definition television, CD-ROM storage, etc. The idea of compression techniques is to remove the redundant information that exists in the video sequences.

Motion compensation predictive coding is the main coding tool for removing temporal redundancy of video sequences and it typically accounts for 50-80% of video encoding complexity. This technique has been adopted by all of the existing International Video Coding Standards. It assumes that the current frame can be locally modelled as a translation of the reference frames. The practical and widely method used to carry out motion compensated prediction is block matching algorithm. In this method, video frames are divided into a set of non-overlapped macroblocks and compared with the search area in the reference frame in order to find the best matching macroblock. This will carry out displacement vectors that stipulate the movement of the macroblocks from one location to another in the reference frame. Checking all these locations is called full Search, which provides the best result. However, this algorithm suffers from long computational time, which necessitates improvement. Several methods of Fast Block Matching algorithm are developed to reduce the computation complexity.

---

*Corresponding author Tel.: +44(0)1512312458, Fax: +44(0)1512074594
Email: a.hussain@ljmu.ac.uk

This paper focuses on a survey for two video compression techniques: the first is called the lossless block matching algorithm process, in which the computational time required to determine the matching macroblock of the full search is decreased while the resolution of the predicted frames is the same as for the full search. The second is called lossy block matching algorithm process, which reduces the computational complexity effectively but the search result's quality is not the same as for the full search.

1. INTRODUCTION

Digital video is a series of orthogonal bitmap digital images called frames displayed in a rapid succession at a constant rate to give the illusion of a motion picture. The applications of digital video have extended to a wide range of industrial applications, especially in the area of entertainment, communications, and broadcasting. As results of technological advances, several commercial products are becoming integral part of modern life, such as High Definition Television (HDTV), digital cinema, smart phones, and mobile devices. Huge revenue from these products and services are gained since the number of end users increases continuously. In 2013, more than one billion unique users visit YouTube each month, and video chat reaches tens of thousands of users online at any time during a day [1]. In addition, the digital video industry invests a lot in the research and development of video technology around £1.5 billion in 2013 in the UK alone [2] to ensure continuous growth in the long term. The major challenge for efficient digital video storage and transmission lies in the huge amount of data needed to display digital video, and hence large memory space is required to store video images, and equally large bandwidth for their transmission. To reduce this amount of data while preserving an acceptable video quality, various video compression techniques have been actively proposed and developed by researchers and companies since the 1980s [3-4]. The idea of these techniques is to provide efficient solutions to represent video data in a more compact and robust way so that the information can be stored or transmitted faster in videoconferencing and videophone, digital broadcasting, interactive games

(internet), etc. Well-known international video coding standards include the former MPEG series and H.26x series [5-11].

The main idea of compression techniques is to remove the redundant information that exists in video sequences. Digital video carries four types of redundancy: colour space redundancy, spatial redundancy, temporal redundancy and statistical redundancy [12-16], [32]. These redundancies are processed separately because of the differences in their characteristics. Video compression contains two systems: video encoders and video decoders. A video encoder compresses the original video for storage and transmission, after which the encoded video is decompressed by a video decoder back to the displayable video before playback and editing. A video encoder consists of three main functional units to remove redundant information: colour subsampling, a temporal model (inter-frame encoder) or a spatial model (intra-frame encoder) and an entropy encoder. Video compression efficiency is achieved by inter-frame encoder which reduces or eliminates temporal redundancy.

An inter-frame encoder exploits the high correlation that exists between successive frames in video sequences especially if the frame rate is high. This correlation leads to temporal redundancy. The goal of inter-frame encoding is to reduce this redundancy. Video coding standards share a number of common features for inter-frame encoding. Each standard assumes that after colour subsampling there are four stages of inter-frame encoding to produce the compressed bitstream, which are: temporal prediction, transform, quantisation and entropy coding [23-25].

Temporal prediction is the main tool that reduces temporal redundancy by predicting some frames from others to reduce the transmission rate of the sequence of the video images and obtain high compression. This means that the current frame could be locally modelled as a translation of the reference frames. Reference frames have to be encoded first, while a residual (difference) between current and reference frames which contain less energy will be encoded later instead of encoding the current frame [12]. To decrease this residual, the prediction can be improved by estimating the motion of the moving objects between the current and the reference frames, which is called Motion Estimation (ME) technique [110]. That is, the motion estimation used to calculate the Motion Vectors (MVs) by comparing the current frame and the reference frame. The technique that uses MVs to predict a new frame

from a reference frame is called Motion Compensation (MC). The predicted frame is known as the Motion Compensated Prediction (MCP) [13]. The first output of this process will be the difference between the current frame and the MCP, which is called the Residual Prediction Error (RPE) (or Displaced Frame Difference (DFD)); the second output will be the motion vectors. The MVs are encoded using entropy coding and RPE between the current frame and the MCP is encoded using transform coding, quantisation and entropy coding. At the decoder, the received MVs will be utilised to form an MCP from the reconstructed reference frame, and then the current frame will be reconstructed by adding the reconstructed RPE to the MCP [9], [83], [13], [110], [92], [4].

ME technique has the highest complexity among all other stages; it typically accounts for 50-80% of the total video encoder complexity. This technique has been adopted by all existing international video coding standards such as the MPEG series and the H.26x series including its latest H.265 code [5], [6], [8], [9], [10], [11]. Therefore, ME is the main challenge for implementing real-time video encoding.

It is possible to estimate the displacement for every one or two pixel positions between successive video frames. However, this is not a practical method since the calculation of these motion vectors is very computationally intensive. Moreover, the number of motion vectors is equal to or half the number of pixels [35-37]. These vectors will be sent to the decoder in order to form an MCP. As a result, a large amount of data should be transmitted [88-90]. Therefore, the most practical and widely used method is to use a group of pixels, called MacroBlock (MB) to estimate the motion of the current frame. This method is called Block Matching Algorithm (BMA) or Block Matching Motion Estimation (BMME) [121], [60], [58], [13].

BMA is the most popular technique used for motion estimation in which video frames are divided into a set of non-overlapped MBs of size N×M. Each target MB in the current frame is compared with a number of candidate macroblocks within the search area in the reference frame in order to find the best matching macroblock. The spatial difference between the two matching macroblocks will determine a set of displacement vectors that stipulate the movement of the macroblocks from one location to another in the reference frame [26], [55]. There are a number of Block Distortion Measures (BDMs) that can be used to calculate the difference between

two macroblocks, namely Mean Absolute Difference (MAD), Sum of Absolute Differences (SAD) and Mean Square Error (MSE) [110]. If a maximum displacement of p pixels/frame is allowed, then $(2p+1)^2$ locations have to be searched in order to find the best match of the current macroblock. Checking all search area locations is referred to as the Full Search (FS) algorithm. It produces the best possible match and the highest resolution MCP. However, this algorithm suffers from long computational time, which necessitates improvement. *V*arious methods of *fast block matching algorithms have been developed to decrease and improve the computational complexity* [98], [60], [34].

In this paper, we investigate two classifications of fast block matching algorithm. Lossless block matching algorithm process, in which the computational time required to determine the matching macroblock of the full search is decreased while the resolution of the predicted frames is the same as the full search. Lossy block matching algorithm process reduces the computational complexity effectively but the search result's quality is not the same as that of the full search.

2. INTRODUCTION TO VIDEO COMPRESSION

2.1 Fundamentals of Video Compression

Video compression, or video coding, has become an essential part of multimedia systems. A huge amount of information is needed in order to display a digital video, therefore a large memory space will be required to store digital video images and it will need an equally large bandwidth for transmission. Video compression is the process of reducing the amount of data required to represent digital video images while preserving an acceptable video quality. This technique provides efficient solutions to representing video data in a more compact and robust way so that the information can be stored or transmitted faster in videoconferencing and videophone, digital broadcasting, interactive games (internet), and etc [124], [56]. The balance between video quality (dependent upon frame size, frame rate and bit depth and file size) should be considered [125].

2.2 Video Coding International Standard

The existing standard of video compression techniques are developed by two public international organisations: the International Telecommunication Union–Telecommunication Standardization Sector (ITU-T), known as the Visual Coding Experts Group (VCEG), and the International Organization for Standardization/International Electrotechnical Commission (ISO/IEC).

The standards approved by the ISO/IEC are called the MPEG family, whose applications range from consumer video on CD-ROM (MPEG-1 1991) to broadcast/storage standard or high definition TV (MPEG-2 1994) and object-based representation (MPEG-4 Visual or part 2 1998). On the other hand, H.26x series of video standards published by the ITU-T focuses on improving the coding efficiency for bandwidth-restricted telecommunication applications as the number of video services increases. The ITU-T published its first video coding standard H.261 in 1990, and in 1995, it evolved H.263 video coding standards (and later enhancements of H.263 known as H.263+ and H.263++) with higher compression ratios [5], [6], [84]. The various applications for transmitting videos over the network have created great demand for efficient video coding.

VCEG and MPEG formed the Joint Video Team (JVT) in December 2001 to complete the draft of the video coding standard as H.264/AVC (MPEG-4 Part 10) in May 2003. The video coding standard H.264/AVC is reported to achieve gains in compression efficiency of up to 50% compared with its predecessor MPEG-2. However, the increasing popularity of high definition TV, video delivery on mobile devices and other multimedia applications create new demands for video coding standards. In January 2010, the Joint Collaborative Team on Video Coding (JCT-VC) was created as a group composed of VCEG and MPEG to develop a new-generation video coding international standard. In February 2012, JCT-VC introduced the committee draft video compression standard called High Efficiency Video Coding (HEVC), which is also known as H.265 and MPEG-H Part 2. The final draft international standard appeared in January 2013 [11]. HEVC code (without reduction in visual quality) has improved the video compression ratio by at least 50%, compared with H.264, across various applications such as videoconferencing, digital

storage media, television broadcasting, internet streaming and communication [128], [84], [99], [33].

2.3 Redundant Information

For all the standard video compression techniques, video coding can be obtained by taking advantage of the redundant information in any video [74], [4], [93], [5], [110], [44]. These include the following:

1. **Colour Space Redundancy**

Human Visual System (HVS) is more sensitive to luminance components than to chrominance components. Therefore, colour subsampling can reduce the resolution required to represent chrominance components. The first of several steps in compression is to transfer the information in the picture into the frequency domain.

2. **Spatial Redundancy**

This redundancy comes from the spatial correlation in an image, where a block of an image can be predicted from its neighbouring pixels, which is called intra-frame compression, as shown in Figure 1. There are several spatial compression algorithms that are proposed for this purpose; the most common uses are predictive coding, transform coding such as Discrete Cosine Transform (DCT), quantisation and entropy coding.

3. **Temporal Redundancy**

Adjacent frames are highly correlated; that is, most of the time, the image frame looks similar to the frame before it. This redundant information can be removed using *inter-frame* compression.

There are several inter-frame compression methods of varying degrees of complexity, such as subsampling coding, difference coding, block-based difference coding and motion compensation [5], [93].

Figure 1: Spatial and temporal correlation in video sequence [12]

4. Statistical Redundancy

For any data, there is a minimum number of bits required to represent it without losing any information. Bit redundancy could be removed to improve intra-frame and inter-frame compression. This can be performed by entropy coding such as Run Length Coding (RLC), Huffman Coding and Arithmetic Coding [57].

3. LOSSLESS AND LOSSY COMPRESSION

In general, video coding contains two systems: video encoders and video decoders, as shown in Figure 2. A video encoder consists of three main functional units: colour subsampling, a temporal model (inter-frame encoder) or a spatial model (intra-frame encoder) and an entropy encoder. The target of the encoder is to condense the huge amount of information needed to display a video frame in order to achieve a high compression ratio as shown in Equation (1):

$$Compression\ Ratio = \frac{Original\ Video\ Size\ (Bytes)}{Compressed\ Video\ Size\ (Bytes)} \quad (1)$$

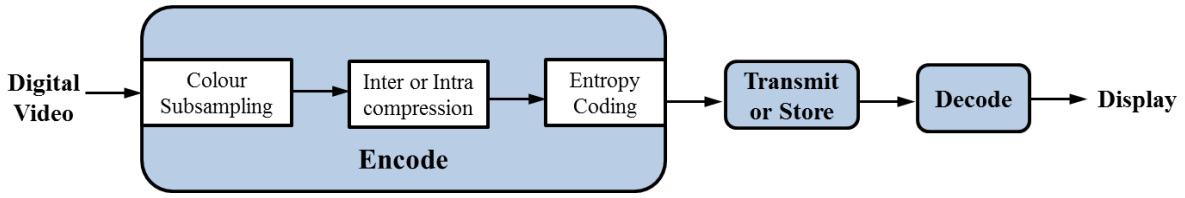

Figure 2. Encoder/ Decoder System.

The encoder can be classified into two approaches: lossless and lossy approaches. Lossless technique (which is also known as bitpreserving or the reversible method) is used to compress the statistical redundancy. This method has a low compression ratio of about *3:1* or *4:1* in the best case, but the reconstructed data is identical to the original data. Lossy technique usually achieves a high compression ratio from *50:1* to *200:1* or more, but the reconstructed data is not identical to the original data; that is, there is loss of information [13],[126].

3.1 Quality Measure in Video Coding

In video compression, lossy approach is the main method used to achieve a high compression ratio; however, this approach leads to lost information (it is called distortion) after reconstruction of the compressed video. In order to assess the quality of the reconstructed video, several methods have been developed. One of the simplest and most popular methods is to use Mean Square Error (MSE) for each frame separately and take their arithmetic mean. *MSE* is the average of the squared error measure determined according to the following equation:

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(f(i,j) - \hat{f}(i,j) \right)^2 \quad (2)$$

where *M* and *N* are the horizontal and vertical dimensions of the frame, respectively, and $f(i,j)$ and $\hat{f}(i,j)$ are the pixel values at location (i,j) of the original and reconstructed frames, respectively.

A more common form of the *MSE* measure is the Peak Signal-to-Noise Ratio (PSNR), which is defined as:

$$PSNR = 10\ log_{10} \left(\frac{(f_{max})^2}{MSE} \right) \text{ dB} \quad (3)$$

Where f_{max} is the maximum possible pixel value (for example, 255 for an 8-bit resolution component). Equation 3 shows that the PSNR measures the strength of the signal relative to the strength of the error. In application, PSNR between the original and reconstructed video sequences is measured by computing the PSNR for each frame separately and taking their arithmetic mean. A high PSNR usually indicates high quality and low PSNR usually indicates low quality. However, PSNR is an objective measure, which means that a particular value of PSNR does not necessarily equate to a subjective video quality perceived by the Human Visual System (HVS). The easy and quick calculation of PSNR makes it a very popular quality measure and it is widely used to compare the quality of the decompressed and the original videos [4], [110].

3.2 Types of Frames

Video frames are compressed using various algorithms depending on the frame type. Figure 3 shows the three major frame types used in various video coding algorithms, which consist of I-frame, P-frame and B-frame [27], [95], [13] .

I-frame 'Intra-coded frame': this type of frame is coded independently from all other frames. This frame is compressed as a still image using a still image compression technique such as transform coding, vector quantisation or entropy coding. This type of frame is the largest size in encoding but is faster to decompress than the other frames.

P-frame 'Predicted frame': an inter-coded frame, which is forward predicted from the last I-frame or P-frame, i.e. it is impossible to reconstruct it without the data of the previous frame (I or P). P-frames are typically a smaller size in encoding than I-frames.

B-frame 'Bi-predictive frame': an inter-coded frame, which is a bi-directionally predicted frame, coded based on both the previous and next I- or P- frames, but a B-frame cannot be the reference for other B-frames, i.e. there are two other frames necessary to reconstruct them. So B-frames are an effective video coding tool to improve coding efficiency. However, using B-frames for coding requires more memory in the encoder and decoder, as an extra frame (next reference) needs to be

stored during the decoding process. Furthermore, B-frames introduce extra delay (next reference send first), which is unacceptable in two-way video coding such as for a videoconferencing application; in this case, no B-frames are used [110].

3.3 Group of Pictures

Frames between two successive I-frames, including the leading I-frame, are collectively called a Group of Pictures (GOP), which is the smallest random access unit in the video sequence, as shown in Figure 3. A GOP pattern is defined by the ratio of P- to B-frames within a GOP. Common frame patterns used for DVD are IBP and IBBP. All three frame' types do not have to be used in a pattern. For example, an IP pattern can be used in two ways for video coding. Longer GOP lengths (the term long GOP refers to the fact that there are several P- and B-frames used between I-frame intervals) encode video efficiently by giving a good compression ratio. Smaller GOP patterns with shorter GOP lengths work better with video that has quick movements, but they do not compress the data as much. For television systems, an I-frame is sent typically every half second in order to enable channel surfing [95].

I-frame is often used to efficiently code frames corresponding to scene changes, i.e. frames that are different from previous frames and cannot be easily predicted. Since video sequences have variable scene durations, depending on the content, it is not possible to use a fixed GOP structure to code the video sequence effectively. This is because the position of I-frames in the sequence depends on the time that scene changes happen. For example, video coding standards allow for macroblocks which are *16×16* pixels in P- and B-frames to be intra-coded if they cannot be predicted efficiently. This means that, even if all the frames are set to be of types *P* or *B*, there may be many macroblocks in each frame that are intra-coded [122], [59].

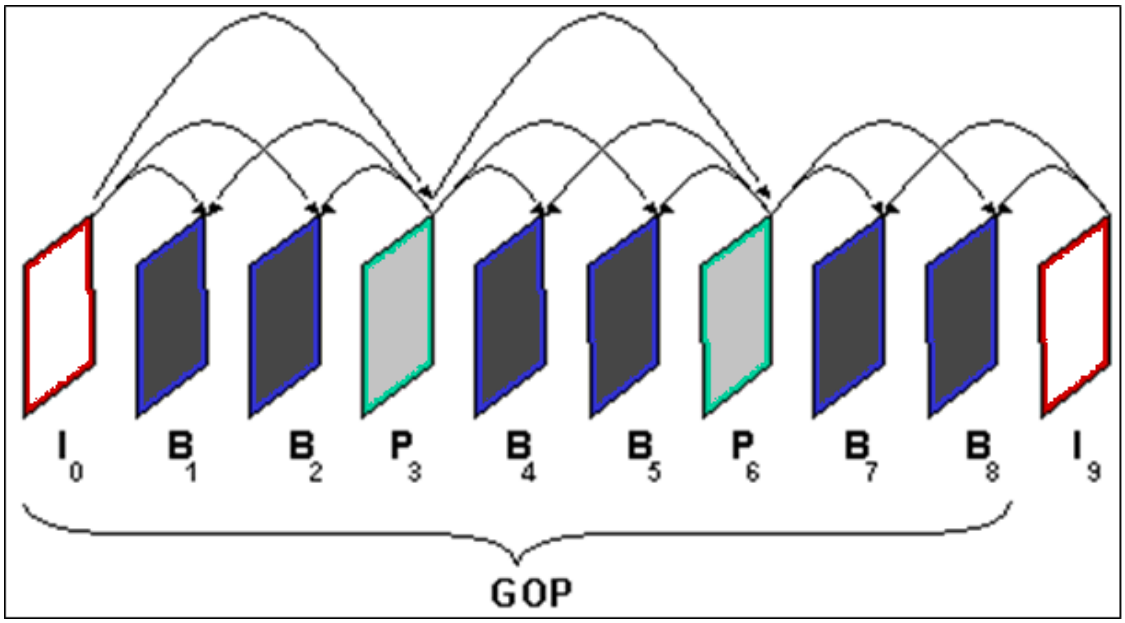

Figure 3: Types of coded frames [59]

Coding as *P*- and *B*-frames gives a higher compression rate, but it is more computationally expensive than coding an *I*-frame. This relates to the fact that coding *P*- and *B*-frames uses motion estimation and motion compensation as will be defined in Section 4.

3.4 Inter-Frame Compression

Inter-frame compression exploits the high correlation that exists between successive frames in video sequences, especially if the frame rate is high. This correlation leads to temporal redundancy [28-30]. The goal of inter-frame coding is to reduce this redundancy. Video coding standards share a number of common features, as shown in Figure 4. Each standard assumes that after colour subsampling there will be four stages of inter-frame encoding to produce the compressed bitstream: temporal prediction, transform, quantisation and entropy coding.

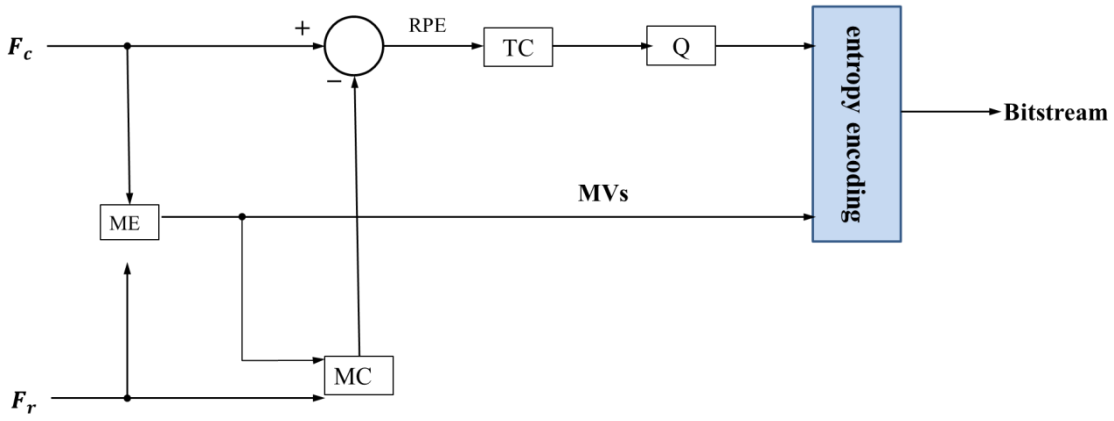

Figure 4: Inter-frame encoder

3.5 Temporal Prediction

The goal of temporal prediction is to reduce temporal redundancy coming from high correlation between successive frames. This can be done by predicting some frames from others to reduce the transmission rate of video image sequences and obtain further compression. Reference frames of type I or P could be used to predict frames of type P or B. In forward prediction, past frames in the display order are used as reference frames to the current frame; while, in backward prediction, the reference frames of the current frame are displayed in the display order in the future frames. The average of the forward and backward predictions may be used to predict frames of type B. In any prediction, reference frames have to be encoded first, while a residual (difference) between current and reference frames which contain less energy will be encoded later instead of the encoded current frame [12].

To decrease this residual, the prediction is improved by estimating the motion of the moving objects in-between the current and the reference frames, which is called Motion Estimation (ME) technique. That is, the motion estimation has been used to calculate the Motion Vectors (MVs) by comparing the current frame and the reference frame. The technique that uses the MVs to predict a new frame from a reference frame is called Motion Compensation (MC). The predicted frame is known as a Motion Compensated Prediction (MCP). The first output of this process will be the difference between the current frame and the MCP, which is called the residual prediction error (RPE) (or displaced frame difference (DFD)); the second output will be the motion vectors.

Motion vectors are encoded by lossless compression, while RPE is encoded by lossy compression to get high compression ratio [9], [83], [13], [110], [92], [4].

3.6 Transform Coding (TC)

Transform coding is one of the most important applications of data compression, which is employed to reduce spatial redundancy. The RPE, which is the difference between the current frame and the MCP frame, has a high correlation between neighbouring pixels, as shown in Figure 5. Inter-frame compression can be coded more efficiently by exploiting these similarities and reducing the spatial redundancy. Transform coding converts the data from a spatial domain of the RPE into a transform domain to produce a set of coefficients. The energy of the transformed data (coefficients) is localised and compacted at certain areas. The transform should be reversible and can transform as much information as possible into a small number of transform coefficients [100-103].

Over the years, a variety of linear transform methods have been developed. The most popular transforms can be classified into two types: block-based transform coding and image-based transform coding [13], [71].

Block-based coding is widely used in image/video coding standards systems. In block-based transforms, an image is divided into non-overlapping macroblocks and for each macroblock the 2-D transform coding is applied. Most transform coding systems employ a macroblock size of 8×8 or 16×16. Note that both sizes are powers of 2, which reduces the computation complexity of the transform coding and requires low memory. The block-based transform coding converts the macroblock pixel information into the frequency domain where pixel correlation information is captured in a DC coefficient and pixel difference information is captured in AC coefficients.

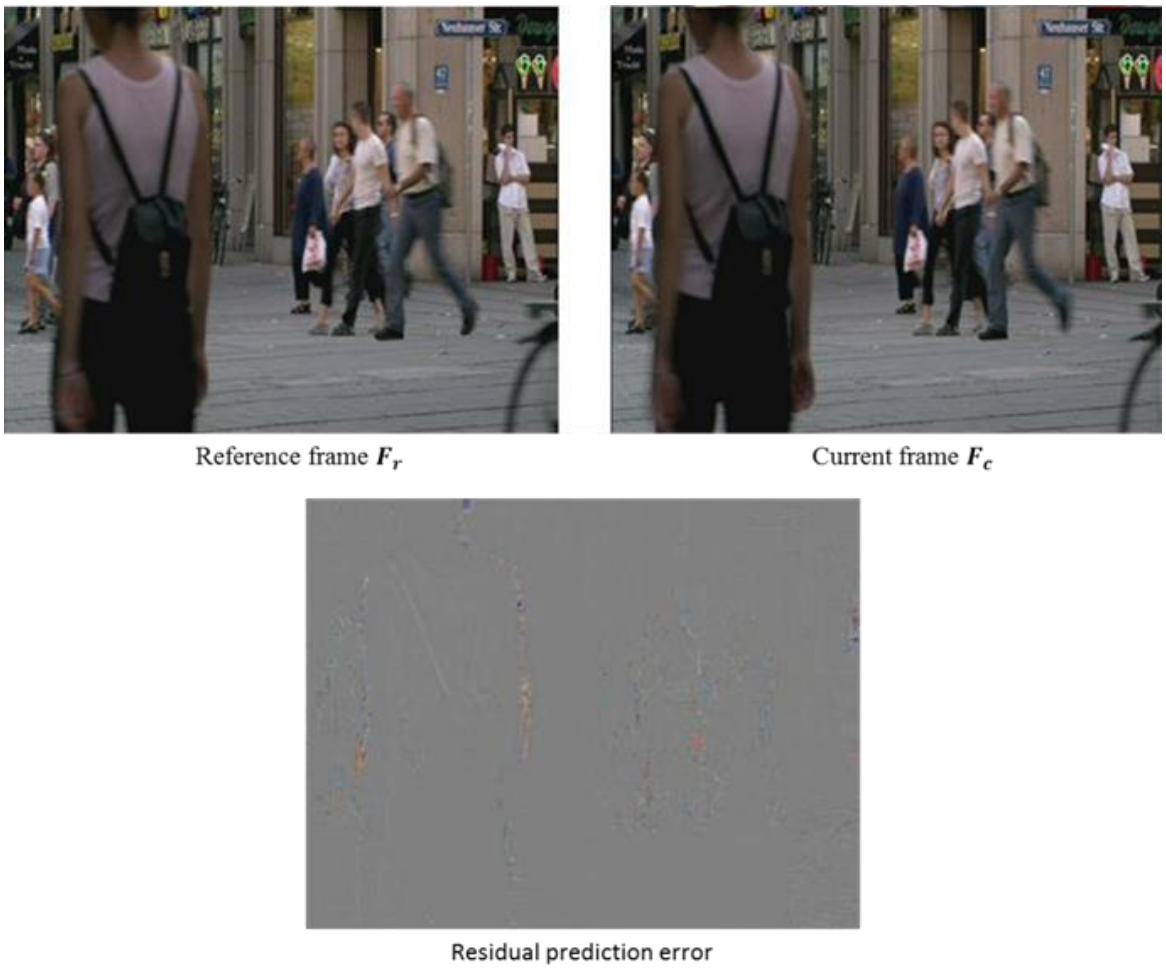

Figure 5: The similarity between neighbouring pixels of the residual prediction error

The AC coefficients normally have very small values because of the high correlation between the pixels in a macroblock. Therefore, the energy is concentrated in the DC coefficients and a small number of AC coefficients that are close to the DC coefficient. That is, the macroblock energy is usually concentrated in the low frequency region. Furthermore, block-based transform allows each macroblock to be processed according to its content in order to improve the coding performance significantly, as performed in H.264. The disadvantage of such block-based transform is that the transform can only exploit the correlations within the macroblock and hence this technique suffers from artefacts at edge macroblocks using low bit rates, which affects the coding efficiency. Popular block-based transforms include: Discrete Cosine Transform (DCT), Karhunen–Loeve Transform (KLT), and Singular Value Decomposition (SVD) [13], [31], [71], [105].

Image-based transform resolves the problem of artefacts initiated at edge macroblocks using Discrete Wavelet Transform (DWT) on the entire image or video frame. An image-based transform would provide better energy compaction, but it tends to suffer from higher computational complexity and memory requirements in comparison to block-based transform because the whole image is processed as a unit. Therefore, block-based transform is better compatible with the residual prediction error [63-66], [71], [126], [13], [31].

3.7 Quantisation (Q)

Quantisation is a mapping of a large set of possible inputs into a smaller set of possible outputs. Quantisation forms the heart of lossy compression and it is an irreversible process. The goal of this scheme is to map the data from a source into as few bits as possible such that the reconstructed data from these bits is as close to the original one as possible. There are two types of quantisation, scalar and vector. Scalar quantisation maps a single value of the input signal to one quantised output value (level). A scalar quantiser of the same step size is called a uniform quantiser, while a quantiser of different step size is called a non-uniform quantiser. If the step size is large (coarse), fewer numbers of bits are required and hence high compression ratio is achieved while the quality of the reconstructed data is reduced. However, small step size gives a larger range of quantised values and hence reduces compression efficiency and improves the reconstructed data. In each video coding standard, there exists a defined set of quantisation step size parameters that provide the best balance between decoded video quality and compression ratio for different applications.

Vector quantisation maps a group of input values (vector) (such as a block of image samples) to a group of quantised values which is the index from a “codebook”. Vector quantisation can be used alone as a method of compression and it is powerful with high computational complexity.

Scalar quantisation techniques are involved in most video coding standards with the combination of transform coding. After the transformation, the energy in both the pixel and the transform domains are equal but the transform coefficients are less

correlated than the original data. In transform domain, the majority of energy is concentrated on the low frequencies while little energy is concentrated on the high frequencies. Since human eyes are more sensitive to low frequencies compared to high frequencies, greater compression can be achieved by apply coarser quantisation step size at higher frequencies to remove insignificant coefficient values [81], [106], [104], [137], [92], [110], [13].

3.8 Entropy coding (EC)

Entropy coding is the last stage in a video encoding system. It is a lossless compression scheme used to remove statistical redundancy by determining the minimum number of bits required to represent the data without losing any information. EC converts the MVs, the quantised transform coefficients and other information from the intra-compression process into a compressed bitstream suitable for transmission or storage. The widely used entropy coding are Variable Length Coding (VLC) and Arithmetic Coding. Arithmetic coding usually provides better compression efficiency, with relatively high computational complexity. These codes are improved by Context-Adaptive VLC (CAVLC) and Universal VLC (UVLC), which are based on VLC, while Context-Adaptive Binary Arithmetic Coding (CABAC) is based on arithmetic coding. CABAC provides bit-rate savings of 9-14% compared to CAVLC but this is at the cost of higher complexity. The low complexity CAVLC entropy encoding method is utilised by the H.264 standard [128], [13], [137].

3.9 Decoding of Inter-frame compression

The decoder interprets the compressed data stream of the compressed motion vectors and compressed RPE; the process is reversed to reconstruct the original frame.

In the decoder side (Figure 6), the reference frame is reconstructed by intra-frame decoding and is ready to compensate and predict the current frame. To produce decode of residual prediction error which is denoted by $\widehat{RPE}$ in Figure 6, entropy decoding is used followed by inverse quantisation (Q^{-1}), then inverse transform

coding TC^{-1}. Note that the irreversible quantisation process means that $\widehat{RPE}$ is not identical to RPE. Finally, $\widehat{RPE}$ is added to the predicted frame to introduce the reconstructed frame.

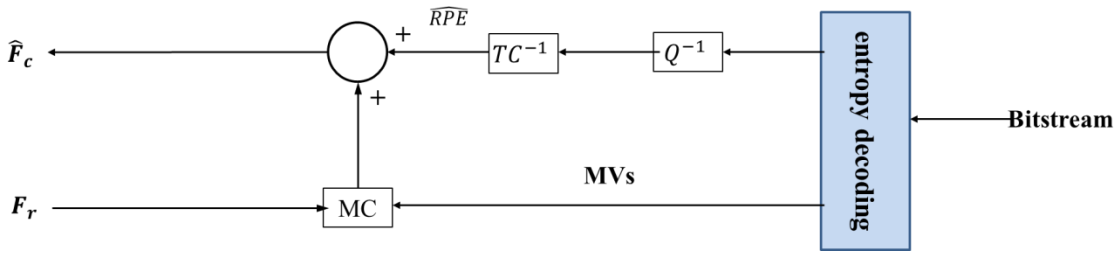

Figure 6: Inter frame decoder

4. MOTION COMPENSATION AND MOTION ESTIMATION

4.1 Motion Compensation (MC)

Motion compensation (MC) has been used as a main tool to reduce the temporal redundancy that comes from the small change in the contents from one image to another in video sequences. That is, MC is the key to achieve high compression ratio for the coding system. This technique dates back to the early 1970s and has been adopted by all of the existing international video coding standards, such as MPEG series and H.26x series including H.265 [5], [6], [8], [9], [10], [11].

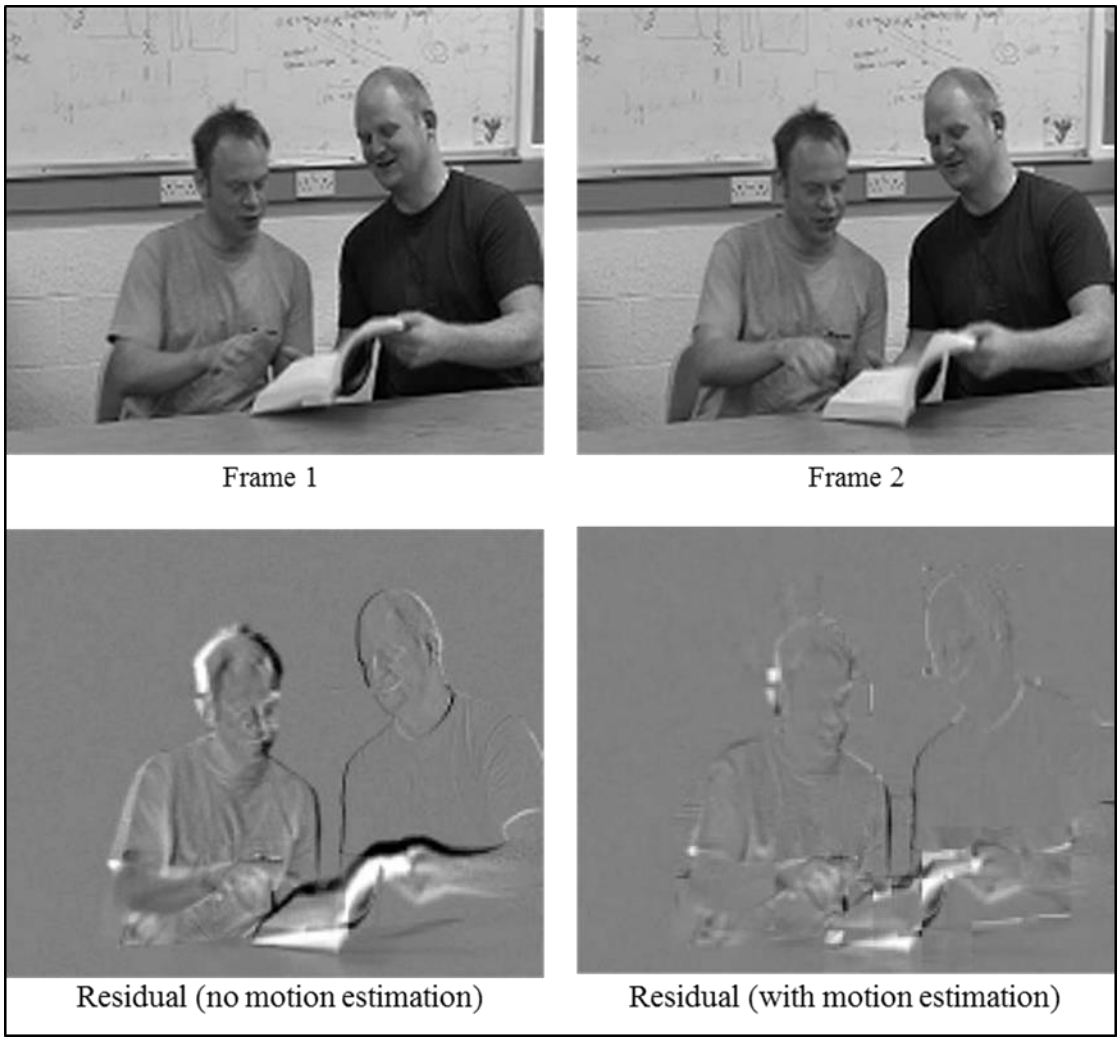

Figure 7: The residual prediction error without ME and the residual prediction error with ME [13]

Motion Compensated Prediction (MCP) assumes that the current frame can be locally modelled as a translation of the reference frames. MC uses reference frames to predict the current frame, and then encodes RPE. Normally, a P-frame is predicted from one of the previous reference frames. Similarly, a motion compensated bi-prediction or B-frame is predicted from two previous reference frames and the next frame. To achieve such a high coding efficiency, H.264/MPEG-4 AVC use Multiple Reference Frames' ME (MRFME) of up to five reference frames to predict the current frame. However, this dramatically increases the computational complexity of the encoders. Moreover, MRFME must be stored in memory until they are no longer needed for further usage, which requires a large amount of memory usage [60], [74], [121].

The simplest method of MCP is to use previous frame as the predictor for the current frame, and encode the difference between them. However, this prediction can be effective only if the two frames are similar and the residual values are close to zero. In any video, either the camera is moving or the object is moving with the fixed camera or scene lighting changes [77-79]. In all cases, the difference between successive frames will not be close to zero and a lot of energy remains in the residual frame. This means that there is still a big amount of information to compress after this stage. To achieve further compression, a better prediction of the current frame may be formed by compensating for motion between the two frames. In order to carry out motion compensated prediction, the motion of the moving objects has to be estimated first; this is known as Motion Estimation (ME). Figure 7 shows the residual prediction error with/without ME [121],[60], [137].

4.2 Motion Estimation (ME)

Motion Estimation is the first step of inter-frame compression and usually the most computationally intensive part (about 50% for one reference - 80% for five of the entire system) in a video encoder [121], [60] ,[58], [134], [13], [133]. It is possible to estimate the displacement for every pixel position between successive video frames, producing a field of pixel flow vectors known as the optical flow. The field is subsampling and hence only one vector for every two pixels is shown. However, for motion compensation, this is not a practical method since the calculation of optical flow is very computationally intensive and needs computations for each pixel. Moreover, the number of optical flow vectors is equal to or half the number of pixels. These vectors will be sent to the decoder in order to form MCP. As a result, a large amount of data should be transmitted [121], [60], [58], [13]. The practical and widely method used to estimate the motion of a group of pixels (macroblock) of the current frame is called Block Matching Algorithm (BMA).

4.3. Block Matching Motion Estimation

Block matching algorithm is the most popular technique used for motion estimation, in which the current luminance frame is divided into non-overlapped MacroBlocks (MBs) of size N×M. These macroblocks are then compared with the corresponding macroblock and their adjacent neighbours in the reference frame. This will carry out displacement vectors that stipulate the movement of the macroblocks from one location to another in the reference frame [26]. For any macroblock in the current frame, the BMA finds the matching macroblock of the same size N×M in the search area within the reference frame. The position of the matching macroblock gives the Motion Vector (MV) of the current macroblock, as shown in Figure 8. This motion vector has two parts, horizontal and vertical, which can be positive or negative. A positive value means motion to the right or motion down and a negative value means motion to the left or motion up. These motion vectors will be used to form the MCP to the current frame from the reference by block motion compensation, as shown in Figure 9. MVs will be encoded using entropy coding and the RPE between the current frame and the MCP will be encoded using transform coding, quantisation and entropy coding. At the decoder, the received MVs and RPE will be decoded and utilised to form MCP from the reconstructed reference frame in which the reconstructed RPE are used to reconstruct the current frame.

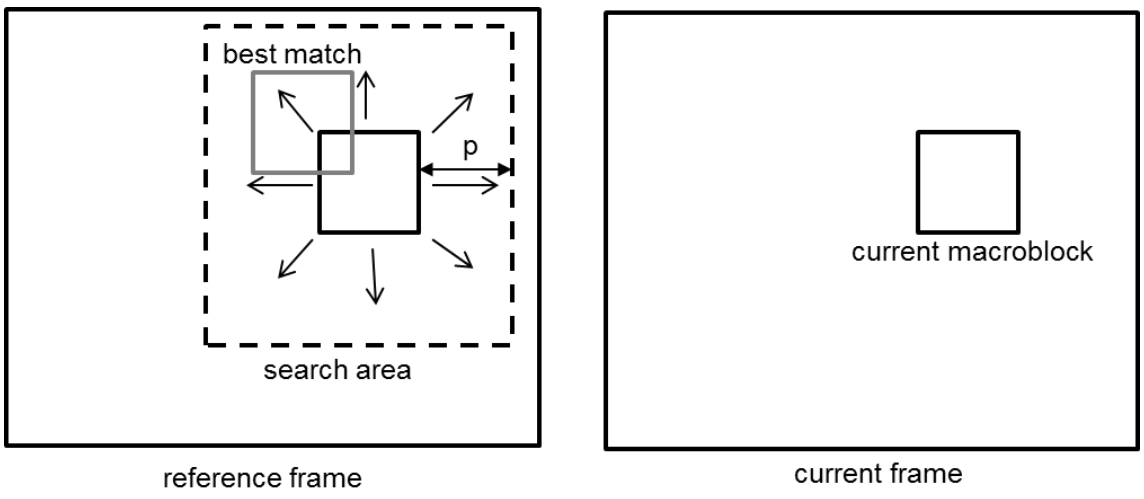

Figure 8: Block matching ME

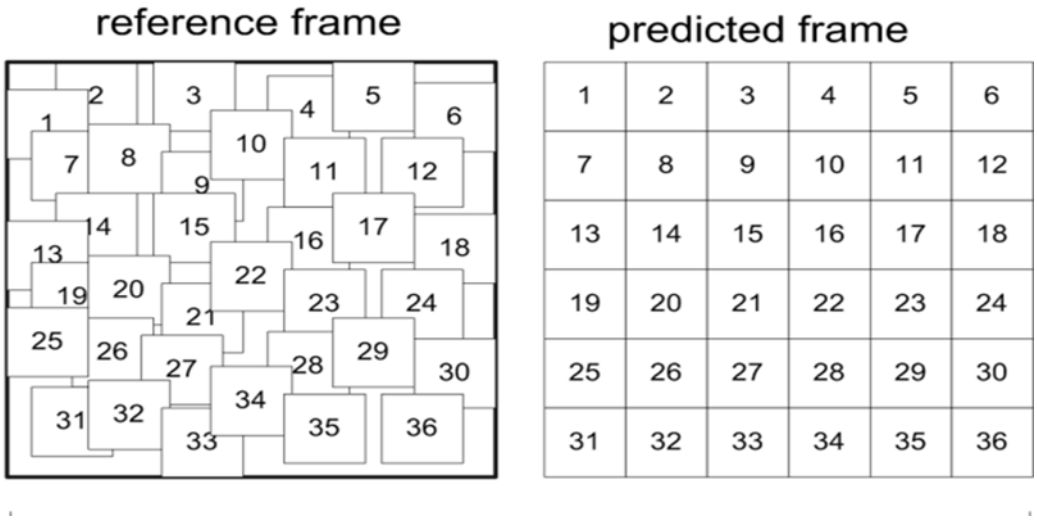

Figure 9: Block motion compensation [74]

The matching measure is usually determined using a Block Distortion Measure (BDM) like Mean Absolute Difference (MAD), or Sum of Absolute Differences (SAD) or Mean Square Error (MSE). The macroblock with the least cost is considered to be the one matching the current macroblock [93].

The search area for a macroblock match is usually constrained up to p pixels on all four sides around the corresponding macroblock in the reference frame, where p is the search parameter. Larger motions require a larger p value, which demands more computational power, as shown in Figure 9.

For the current macroblock *C* of dimension *N*×*N* and the candidate macroblock *R* in the reference frame with a displacement of (v_x, v_y), SAD, MAD and MSE are defined as:

$$SAD = \sum_{i=1}^{N}\sum_{j=1}^{N}\left|C(i,j) - R(i+v_x, j+v_y)\right| \tag{4}$$

$$MAD = \frac{1}{N \times N}\sum_{i=1}^{N}\sum_{j=1}^{N}\left|C(i,j) - R(i+v_x, j+v_y)\right| \tag{5}$$

$$MSE = \frac{1}{N \times N}\sum_{i=1}^{N}\sum_{j=1}^{N}\left(C(i,j) - R(i+v_x, j+v_y)\right)^2 \tag{6}$$

where $C(i,j)$ is the pixel value of current MB at position (i,j) and $\mathrm{R}(i+v_x, j+v_y)$ is the pixel value of the reference frame with the vector (v_x, v_y) within the search range $[-p,p]$.

4.4. Block-Size Motion Estimation

Macroblock size is an important parameter of the BMA. In the BMA, increasing the size of the macroblock means that more computations are required. However, it also means that there will be fewer macroblocks per frame, so the amount of computation needed to perform motion estimation will be decreased. There is a high possibility that the big macroblock will contain different objects moving in different directions. In other words, using a larger macroblock size reduces the amount of computation; however, it provides poor prediction; while smaller macroblock size can produce better motion compensation results and hence reduces residual energy. However, smaller MB size leads to increased complexity and increase in the number of motion vectors that need to be transmitted, which may outweigh the benefit of reduced residual energy. An effective compromise is to adapt the macroblock size to the picture characteristics, for example choosing a large block size in the homogeneous and shade regions of a frame and choosing a small block size for areas of high details, edges, and complex motion, which is called Variable Block-Size Motion Estimation (VBSME) [92], [13], [110], [109].

The default block size for motion compensation is 16×16 samples for the luminance component. Fixed Block-Size Motion Estimation (FBSME) of size 16×16 or 8×8 has been used in the first-generation coding standards; while H.264\AVC utilises VBSME, which is more complicated. VBSME allows a macroblock of 16×16 samples of the luminance component to be partitioned into 4 ways, as shown in Figure 10: one 16×16 MB, two 16×8 sub-MBs, two 8×16 sub-MBs or four 8×8 sub-MBs. In addition, each of the four 8×8 sub-MB partitions within the MB can be further sub-partitioned into 3 ways: two 8×4 sub-MBs, two 4×8 sub-MBs or four 4×4 sub-MBs. These partitions and sub-partitions give around 41 MBs in total for each MB. For each type of sub-MB, a motion vector is required. Each motion vector must be coded and transmitted with the choice of partition(s). In order to get these MVs for each

MB, the computation of comparison operations was increased. To enhance these computations, a large partition size is applied for homogeneous areas of the frame and a sub-partition size may be useful for detailed areas [22], [110], [9], [129],[109].

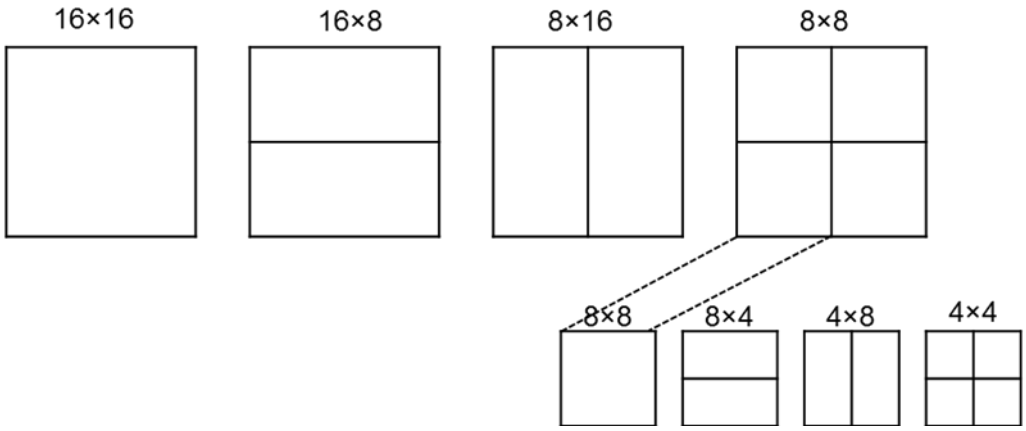

Figure 10: Macroblock partitions and sub-macroblock partitions

4.5. Full Search (FS)

The simplest algorithm which can be used for motion estimation to find motion vectors is the Full Search (FS), or Exhaustive Search (ES), which exhaustively searches for the best matching block within the search area, where the correlation window is moved to each candidate position within the search area. It can be described by:

$$SAD(m,n) = \sum_{i=1}^{N}\sum_{j=1}^{N} |c(i,j) - s(i+m, j+n)| \;;\; -p \le m,n \le p \tag{7}$$

$$MV = \{(u,v) \mid SAD(u,v) \le SAD(m,n);\, -p \le m,n \le p\} \tag{8}$$

where SAD(m, n) is the distortion of the candidate macroblock at search position (m, n), $\{c(x,y) \mid 1 \le x \le N, 1 \le y \le N\}$ means current macroblock data, $\{s(x,y) \mid -p \le x \le p+N, -p \le y \le p+N\}$ stands for search area data; the search range is $[-p, p]$, the block size is N×N.

From the above, $(2p+1)^2$ of search locations need to be examined by the FS algorithm. As a result, FS finds the best possible match and gives the highest PSNR

amongst any block matching algorithm; however, a large amount of computational complexity is involved, especially with VBSME and MRFME.

Various methods of fast block matching algorithms have been developed to decrease and improve this computational complexity. If the algorithm enhances the computation and produces the same quality results as FS then it is called lossless block matching algorithm while if the algorithm could not keep the same quality results then it is called lossy block matching algorithm [110], [121], [60].

5 FAST BLOCK MATCHING ALGORITHMS

Motion estimation shows computational complexity. Hence, the computational complexity of video coding can be reduced by efficiently coding ME. A block matching algorithm is the most common technique used for motion estimation to find the best matching macroblock for the current macroblock from the reference frame. FS is the simplest but the most computation-intensive BMA, which exhaustively tests all the search locations for the best matching macroblock within the search area. As a result, FS finds the best possible match and gives the highest PSNR. Moreover, variable block size and multiple reference frames have been involved in later video coding standards. Therefore, the required computation is highly increased and motion estimation has become a problem in many video applications, for example mobile video and real-time video coding.

In the last three decades, various methods of fast BMA have been developed to reduce such high computational complexity. Some of the fast BMA algorithms have been adopted in video coding standards [5], [6], [8] . This indicates that this is an extremely active field of research, and most fast block matching algorithms are introduced first for FBSME and then extended to VBSME [132]. The performance of each algorithm can be estimated by benchmarking with FS. The effective one minimises the RPE and saves the computational time compared with Full Search.

Fast block matching algorithms can be classified into lossy block matching algorithms and lossless block matching algorithms. Lossy BMAs reduce the computational complexity; however, the search results quality is not the same as for FS. That is, the PSNR of the decompressed video with lossy BMA is not as good as

the PSNR of the one with the full search. While lossless BMA preserves the video quality as well as speeding up the FS [98], [60], [34].

5.1. Lossy Block Matching Algorithms

5.1.1 Fixed Set of Search Patterns

Fixed set of search patterns or what is known as reduction in search positions is the most popular category in lossy block matching algorithms. These algorithms reduce search complexity by selecting a subset of the possible search candidate locations instead of all possible MBs within the search window. Most algorithms in this category state that the error decreases monotonically as the search location moves closer to the best-matching location. Therefore, the search starts with the locations coarsely spread over the search window according to some predefined uniform pattern. After that, the search is repeated with a smaller spread around the search location with the minimum BDM (error) obtained from the preceding step. Each search pattern has a specific shape (rectangle, diamond, hexagonal, cross, etc.) [4, 2002; [60].

The first algorithm initiated in this category is the Two-Dimensional Logarithmic Search (2D-LOG), which is proposed in 1981 [68]. After that, some well-known similar algorithms were proposed, such as: Three Step Search (TSS) [80], Orthogonal Direction Search (OSA) [107], New Three Step Search (NTSS) [108], Four Step Search (4SS) [82], Diamond Search (DS) [113], Simple and Efficient Search (SESTSS) [70], Cross-Diamond Search algorithm (CDS) [46], Novel Hexagon-based Search (NHS) [39], Efficient Three Step Search (ETSS) [135], Modified DS (MODS) [131], Multi-pattern-based search (TCon) [14] and many others.

Much of the research and coding was dependent on the Fixed Set of Search Patterns due to its high-speed search capabilities in comparison to other lossy BMA categories. Unfortunately, these algorithms produce significant loss in visual quality when the actual motion does not match the pattern and hence these algorithms

become trapped in a local minimum. As an example, a centre-biased search pattern cannot provide optimal motion estimation for videos with large motions [61].

5.1.2 Three Step Search (TSS), New TSS (NTSS), and Simple and Efficient TSS (SESTSS)

Three step search, new three-step search and simple and efficient three-step search come under the N-Step Search class. The steps of this class are summarised as follows: (1) Choose step size (which is usually slightly larger or equal to half of the search window). (2) Number of search points is selected at a distance of the step size as well as the centre point. The macroblock with the minimum BDM value becomes the centre of the next step. (3) Divide step size by two and select new search points at a distance of the new step size. (4) Repeat step 2 until the step size becomes one.

TSS uses a maximum of three steps in a coarse to fine search patterns. For a usual search window of parameter $p= 7$ the initial step size will be $4=(p+1)/2$; TSS utilises nine search points centred at the search area (eight points on the boundary of the search square and one centre point) to be compared in the first step search. As mentioned before, the point with the minimum BDM value becomes the centre of the next step. Therefore, there are eight search points to be compared in the second and third step searches, i.e. the total number of search points is $(9+8+8=25)$, as shown in Figure 11.

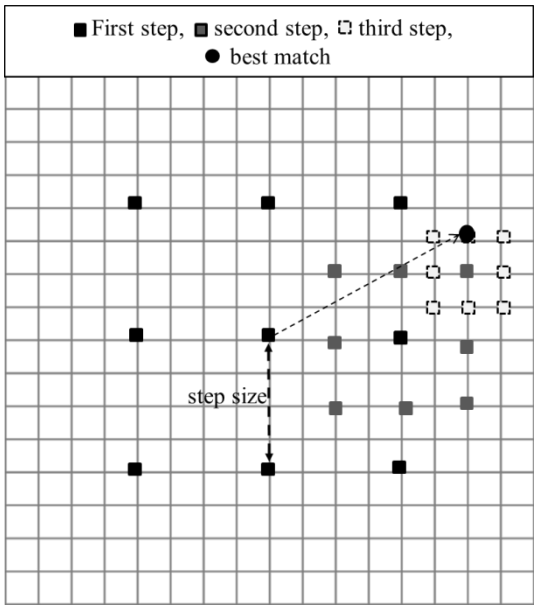

Figure 11: TSS [80]

Due to its simplicity and reasonable performance, the TSS is widely used for research purposes [45]. The drawback of the TSS is that it is not efficient with small motion video, since the search points forming the search pattern in the first step are positioned at a relatively large distance from the search centre. While 80% of the MBs in various motion video sequences can be regarded as stationary or quasi-stationary MBs, which means that 80% of MVs are centre-biased, i.e. they lie within a region of 5×5 of the central area [46]; therefore TSS is not efficient for most video sequences.

This problem was solved in 1994 by proposing a new search called NTSS [108]. NTSS provided improvement over the quality results of TSS (refer to Figure 12). Therefore, this algorithm is considered as one of the first widely accepted fast block matching algorithms. Moreover, it has been used in earlier standards like MPEG 1 and H.261 [96].

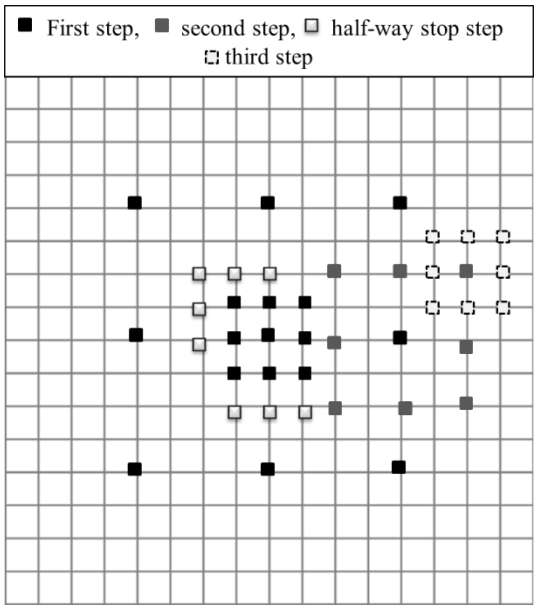

Figure 12: NTSS [108]

NTSS added a smaller search pattern of eight points at the central area to the first step of the original TSS search pattern. That is, NTSS requires more search points compared to TSS. For search windows of parameter p= 7, NTSS requires 33 search points for large motion MBs while TSS always required 25, which means more computations may be needed.

Another extension illustrated to speed up TSS was done by Simple and Efficient TSS [70]. SESTSS requires around half of the computation for TSS while keeping the same regularity and good performance. It exploits the fact that the uniform distribution search pattern in TSS is not effective since the error decreases monotonically as the search location moves closer to the best-match location. Minimum points cannot occur in two directions opposite to each other, which means that, for the search pattern in TSS, at most half of the total eight points are actually required to be searched in each step, and, thus, the computational complexity can be further reduced. Additional computation is needed to determine which directions are selected. The algorithm still has three steps like TSS but each step has two phases as follows [70]:

Step 1: first phase: compute MAD of the three locations *A, B* and *C* as shown in Figure 13. Point *A* refers to the centre location. *B* and C are located at step size = 4

away from A, towards the right-hand side and bottom. In the second phase, a few more points are added depending on the following conditions:

$$
\begin{aligned}
&If\ MAD(A) \geq MAD(B)\ and\ MAD(A) \geq MAD(C), select\ (b)\\
&If\ MAD(A) \geq MAD(B) and\ MAD(A) < MAD(C), select\ (c)\\
&If\ MAD(A) < MAD(B) and\ MAD(A) < MAD(C), select\ (d)\\
&If\ MAD(A) < MAD(B) and\ MAD(A) \geq MAD(C), select\ (e)
\end{aligned}
\tag{9}
$$

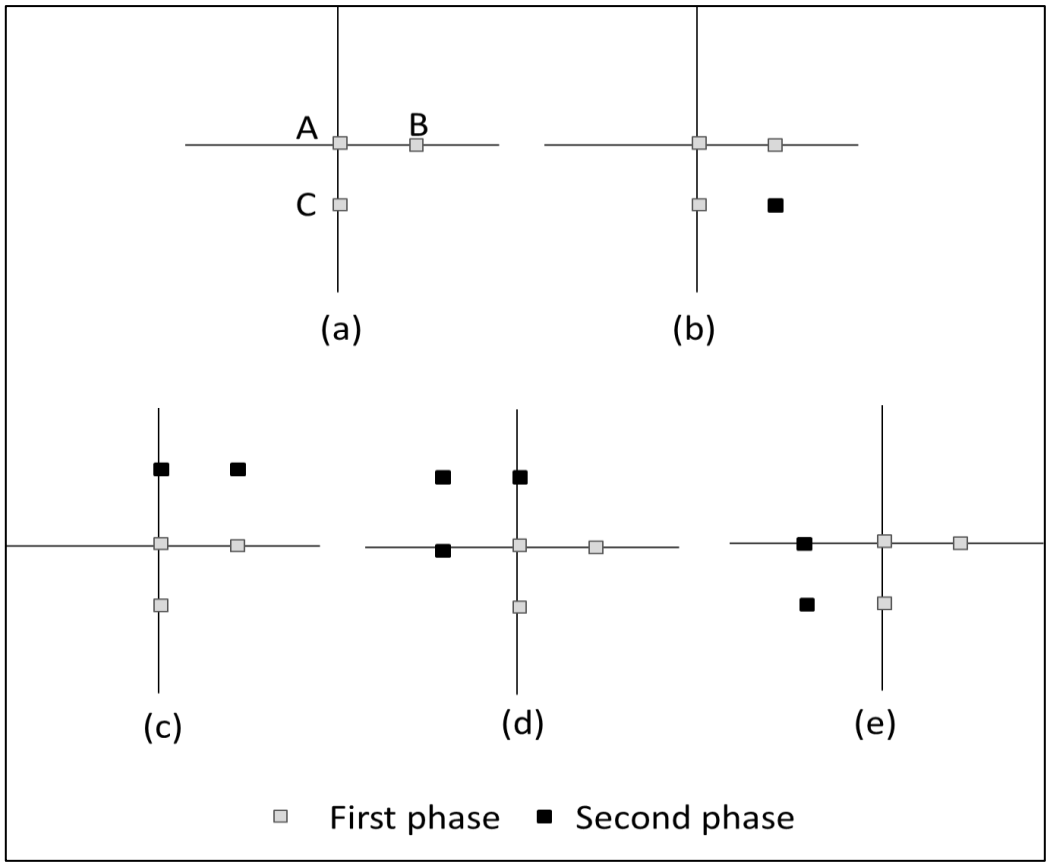

Figure13: Search patterns of SESTSS depending on MAD of A, B and C [70].

Step 2: the point with the minimum MAD value from step 1 becomes the centre of the current step and the step size will be 2. The pattern of the first phase in this step is similar to first phase in step 1.

Step 3: repeat step 2 with step size equal to 1.

Figure 14 shows an example for the SESTSS.

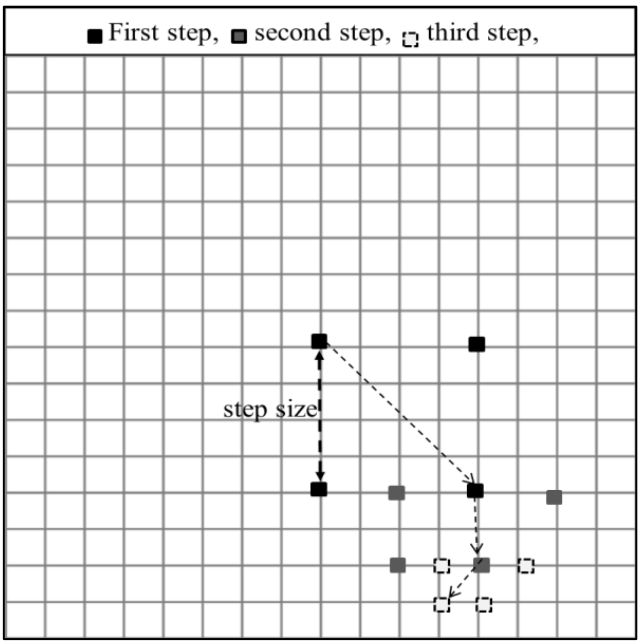

Figure 14: Example of the SESTSS search procedure [70]

5.1.3 Diamond Search (DS)

DS is one of the most common and widely used algorithms. DS requires significantly less computation by reducing the average search points while achieving acceptable performance in comparison with its prior fixed set of search pattern algorithms. Therefore, it is adopted by the reference software of MPEG-4 [7], [60].

Similar to NTSS, DS is based on the assumption that most motion vectors of typical video sequences are centre-biased. Also, it is based on the fact that the MB displacement of real-world video sequences could be in any direction, but mainly in horizontal and vertical directions [113].

This technique utilises two search patterns, a large diamond search pattern (LDSP) of 9 search points and a small diamond search pattern (SDSP) of five search points, as follows:

The matching MB is searched within the search points of the LDSP which are {(±2,0), (0, ±2), (0,0), (±1,±1)}, as shown in Figure 15. The position of the minimum BDM for the LDSP becomes the centre of the new search. If the minimum BDM is already at the centre of the LDSP, then the search pattern is switched from the LDSP to a SDSP of four points {(±1, 0), (0, ±1)}. Otherwise, the search in the next step will be performed only for three or five neighbouring points that complete the

LDSP of this new centre, as illustrated in Figure 15. The LDSP is repeatedly used in the searching procedure until the step in which the minimum BDM point stays at the centre of the LDSP. The search pattern is then switched to a SDSP. The minimum BDM point found from the SDSP will be the best matching block [139], [26], [96], [113].

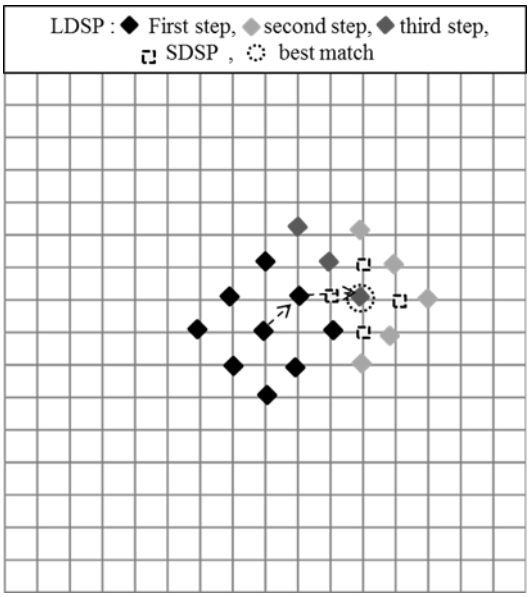

Figure 15: Diamond Search Algorithm [113]

5.1.4 Predictive Search

Predictive search technique is a lossy block matching algorithm that exploits the correlation between the current MB and its neighbouring MB. It utilises the motion information in the spatial and/or temporal neighbouring MB. The predicted MV can be obtained by selecting one of the previously-coded neighbouring MVs; for example, the predictors can be the MVs of the MBs on the left, top, and top right, as shown in Figure 16, or the MV of the collocated MB in the previous frame, as shown in Figure 17, and in the previous two frames.

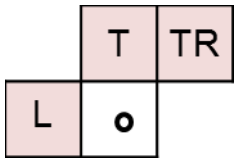

Figure 16: Current MB with the predictor MV of top (T), left (L) and top right (TR) MBs

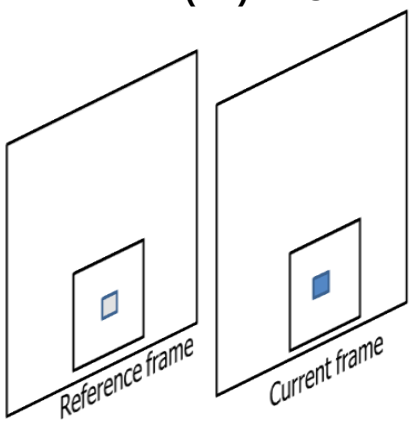

Figure 17: Current MB and the collocated MB in the previous frame

The Motion Vector Predictor (MVP) is utilised in two ways: the difference between the current motion vector and the MVP, which is called motion vector difference, is transmitted instead of the current MV itself. The MVP in this case is the median of three candidate predictors, which are the motion vectors of the three neighbouring MBs, as illustrated in Figure 17 [4].

The MVP forms an initial estimate of current MV. This type is a fast motion estimation algorithm that has low computational complexity with acceptable performance [130]. It can effectively reduce the search points and hence the computation by exploiting the target macroblock that is likely to belong to the area of the neighbouring MVs, and the initial search starts directly in this area. The MVP could be one or more of the previously-coded neighbouring MVs, or their average MVs as in Figure 17. Additional memory for storing the neighbouring MVs is needed in this method [13], [55], [39] .

This technique is used in the Adaptive Rood Pattern Search (ARPS) algorithm [98], Joint Adaptive Block Matching Search (JABMS) algorithm, Unsymmetrical Multi-Hexagon search (UMHexagonS) [136], and simplified block matching algorithm for fast motion estimation [22].

5.1.5 Adaptive Rood Pattern Search (ARPS) Algorithm

ARPS algorithm [98] based MPEG-4 Verification Model (VM) [7] showed a speed 2-3 times faster than that of the DS and maintained a fairly similar performance [138].

ARPS uses a predictive search technique to form an initial estimate of finding the global minimum point. This relates to the fact that, if the MB around the current block moves in a particular direction, there is a high probability that the current MB will also have a similar motion vector. Moreover, the step-size search pattern of this algorithm changes according to the motion vector predicted behaviour. This technique depends on the DS technique, which uses two different types of fixed patterns, the Large Search Pattern (LSP) and the Small Search Pattern (SSP), as shown in Figure 18. In addition, the MVP of this algorithm is the coded motion vector of the immediate left MB, which means one neighbouring MV needs to be recorded. This MVP is utilised to pre-determine the motion behaviour of the current MB and to define the most suitable step size to perform efficient ME.

The steps of this algorithm are as follows:

Step 1: determine the step size that refers to the distance between the centre and any vertex points in the LSP. If x and y are the horizontal and vertical components of the MVP, respectively, then the step size will be the maximum absolute value of these components determined as follows [98]:

$$step\ size = Max\{|x|, |y|\} \tag{10}$$

For the MB on the left side of the frame, the step size will be fixed as 2 pixels.

Step 2: the matching macroblock is searched first within the search points of LSP plus the search point indicated by the MVP. The point that has the least MAD becomes the origin for subsequent search steps. The new search centre directly moves to an area where there is a high probability of finding the global minimum, and the new search pattern is changed to a SSP, as shown in Figure 19.

Step 3: the matching MB found in the current step will be re-positioned as the new search centre of the next search if it is not already at the centre of the search pattern. This process will be repeated until the matching MB stays at the centre of the SSP.

A further development of this algorithm is called Adaptive Rood Pattern-Zero Motion Prejudgment (ARP-ZMP). This can be achieved by checking for zero motion prejudgment in which, if the SAD between the current MB and the MB at the same location in the reference frame (i.e., the centre of the current search window) is less than a predefined threshold, then the search is stopped and the MV will be zero [98].

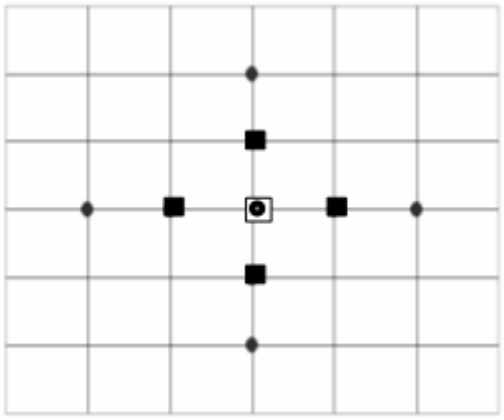

Figure 18: The sold circle points (●) are the LSP and the squares (■) are the SSP for ARPS

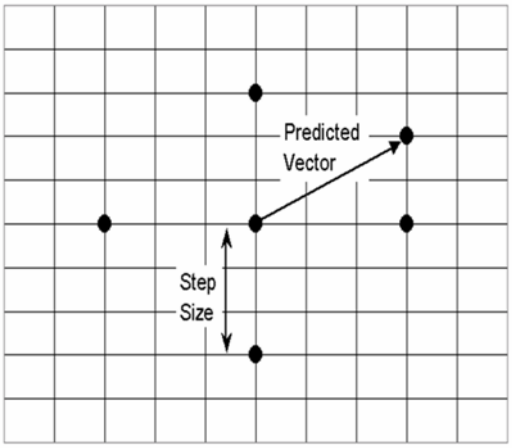

Figure 19: Adaptive Rood Pattern Search [98]

5.1.6 Hierarchical or Multiresolution Search

Hierarchical search exploits the correlation between different resolution levels that represent the same image, which is shown in Figures 20 and 21 [114]. It uses a multiresolution structure (also known as a pyramid structure) that has different image resolutions with smaller image size at the coarser level. The multiresolution structure is constructed either with simple subsampling or filtering.

Hierarchical search is based on the idea of performing motion estimation at each level successively. Thus, motion estimation is first applied at the lowest resolution level to obtain an estimate of motion vector. This MV is then passed to the next higher resolution level as an initial estimate. Motion estimation at the higher

resolution level is then used to refine this initial estimate. This process is repeated until the highest resolution level is reached. Typically, a two- or three-level hierarchical search is adopted. To reduce the complexity of calculating BDMs, small MBs are used for block matching algorithm at lower resolution levels. Moreover, smaller search ranges are used at higher-resolution levels, since motion estimation starts from a good initial estimate. This reduces the number of locations to be searched. More levels can save the amount of computation required, but it has the disadvantage of possibly being trapped in a local minimum because, when the subsampling or filtering is applied to an image, some important details will be lost. Multiresolution technique has been regarded as one of the most efficient methods in BMA and it is adopted in applications with very large frames and search areas [114], [34], [4], [98], [60].

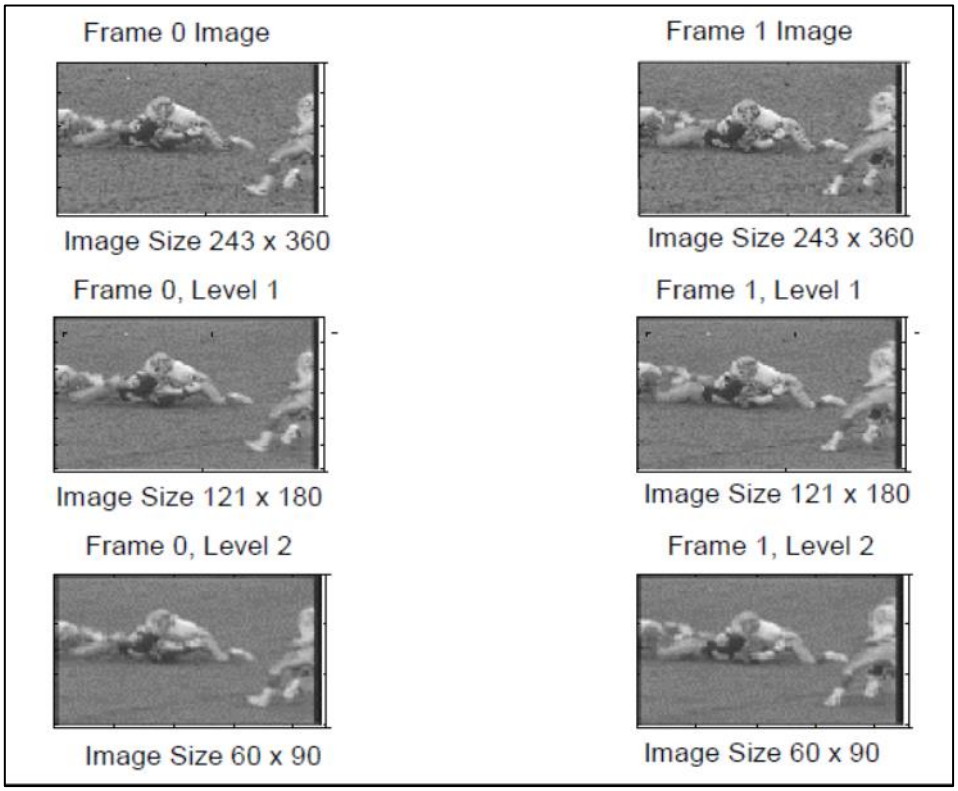

Figure 20: Two-level Hierarchical Search

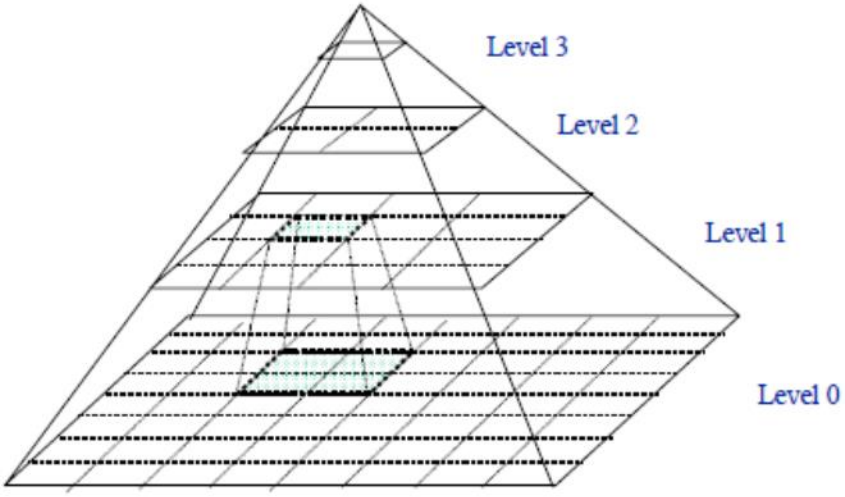

Figure 21: Hierarchical motion estimation using a mean pyramid of three levels [114]

5.2 Subsampled Pixels on Matching Error Computation

The previous three groups of BMAs can reduce the computation of ME by limiting the number of search locations. This category reduces the complexity of the BDM by decreasing the number of MB pixels in current and candidate MBs to speed up ME. In homogeneous areas, neighbouring pixels have high correlation and hence subsampling for these areas can be done without search quality regression. However, in highly textured areas the subsampling will be less accurate [115-119]. Therefore, this category does not guarantee to find the best match, hence it is lossy BMA even when checking all search area locations. Koga et al used in their work [80], a uniform subsampling pattern that performs 2:1 pixel subsampling in both horizontal and vertical directions. As a result, the total computation can be reduced by a factor of 4, as shown in Figure 22. Liu and Zaccarin in their work [87] have used a non-uniform subsampling pattern.

Figure 23 shows a block of 8 × 8 pixels with each pixel labelled $a, b, c,$ and d in a regular pattern. If only the pixels of the pattern that consists of all the $\boldsymbol{a}$ pixels are used for block matching, then the computation is reduced by a factor of 4. To reduce the drawback that ¾ of the pixels do not enter into the matching computation, all four subsampling patterns are using in a specific alternating manner.

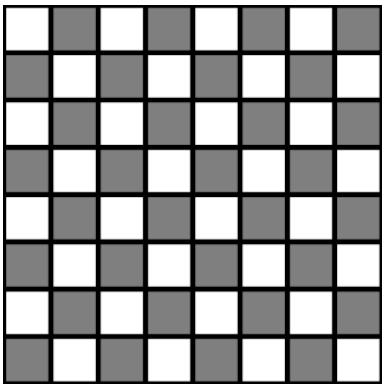

Figure 22: Uniform subsampling pattern 2:1 [80]

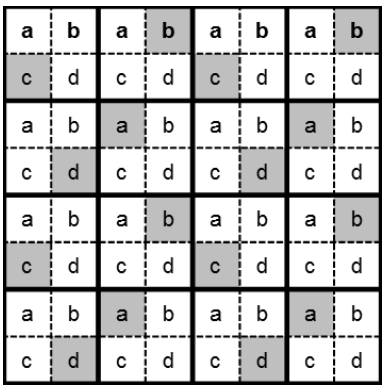

Figure 23: Non-uniform subsampling pattern 4:1 [87]

To enhance the quality of a non-uniform subsampling, Yui-Lam and Wan-Chi (Yui-Lam and Wan-Chi, 1995) changed the number of pixels in the subsampling pattern according to block details. That is, for shade MBs fewer pixels are used and more pixels are involved for high-activity MBs. Such a computation reduction method can be incorporated into other BMAs to achieve higher computational gain.

5.3 Bitwidth Reduction

In a luminance frame, each pixel is represented with 8 bits resolution. This search technique reduces the original 8 bits resolution to less bits width in order to reduce the hardware cost and power consumption and then applies normal ME search strategies. The first algorithm proposed in this category was Bit-Plane Matching (BPM), which indicates whether a pixel is edge or not [69]. The MB mean is used as the threshold to satisfy a One–Bit Transformation (1BT), and the bit plane of an image frame is constructed in the form of:

$$B(i,j) = \begin{cases} 1 & if\ I(i,j) \geq t_{bm} \\ 0, & otherwise \end{cases} \quad (11)$$

where t_{bm} is the threshold value that is set equal to the MB mean, $I(i,j)$ shows the $(i,j)th$ pixel of the image frame and $B(i,j)$ shows the corresponding bit-plane value.

The other common transformation maps a frame of multi-valued pixels to a frame of binary-valued pixels by comparing the original frame with their multi-bandpass filtered versions to construct *1BT* representations [96]. Each frame *I* is filtered with a *17 ×17 kernel K* which is given as in Equation 15. The filtered frame I_F is compared with the original frame *I* to create a one-bit frame *B*, as in Equation 16 [48].

$$K(i,j) = \begin{cases} 1/25 & i,j \in [0,4,8,12,16] \\ 0, & otherwise \end{cases} \quad (12)$$

$$B(i,j) = \begin{cases} 1 & if\ I(i,j) \geq I_F(i,j) \\ 0, & otherwise \end{cases} \quad (13)$$

where $I_F(i,j)$ is the filtered form of the image frame $I(i,j)$.

To find the best matching MB for the current MB, a full search can be used. The error between current and candidate MBs will be calculated as the Number of Non-Matching Points (NNMP), which is measured by the exclusive-or (XOR) operation as follows [48]:

$$NNMP(m,n) = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(B^t(i,j) \oplus B^{t-1}(i+m, j+n) \right) \quad (14)$$

$$-s \leq m,n \leq s-1$$

where (m,n) shows the candidate displacement, $B^t(i,j)$ and $B^{t-1}(i,j)$ are the one-bit planes for the current and reference frame, respectively, s determines the search range, and $\oplus$ is the XOR operation [94].

In Erturk [48], a Two-Bit Transformation (*2BT*) was proposed to improve motion estimation accuracy compared with 1BT. The first bit plane of 2BT is constructed using the mean value ($\mu = E[I_{tw}]$) of the threshold window surrounding the current MB. The second bit plane is constructed using the square root of the variance value ($\sigma^2 = E[I_{tw}^2] - E^2[I_{tw}]$) as follows:

$$B_1(i,j) = \begin{cases} 1 & if\ I(i,j) \geq \mu \\ 0, & otherwise \end{cases} \qquad (15)$$

$$B_2(i,j) = \begin{cases} 1 & if\ I(i,j) \geq \mu + \sigma\ \ or\ \ I(i,j) \leq\ \mu - \sigma \\ 0, & otherwise \end{cases}$$

where $B_1(i,j)$ and $B_2(i,j)$ represent the 2BT, while the number of non-matching points is defined as:

$$NNMP(m,n) = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} \{B_1^t(i,j) \oplus B_1^{t-1}(i+m, j+n)\} \| \{B_2^t(i,j) \oplus B_2^{t-1}(i+m, j+n)\} \qquad (16)$$

$$-s\ \leq\ m,n\ \leq\ s-1$$

where (m,n) shows the candidate displacement, $B_{1,2}^t(i,j)$ and $B_{1,2}^{t-1}$ are the two-bit planes for the current and reference frame, respectively, s represents the search range, and $\oplus$ is the XOR operation. The operation || denotes the Boolean OR operation.

Some other algorithms were proposed to enhance and modify the 2BT , all these algorithms save hardware costs and power consumption but are run at the risk of losing too much quality and hence they are classified as lossy block matching algorithms [49-53].

5.4 Lossless Block Matching Algorithms (Fast Full Search)

A lossless algorithm attempts to improve the time to determine the matching MB without affecting the quality of the FS. However, many studies have indicated that the quality of the produced compressed videos is not as good as that of the ones produced by FS [60]. Usually, the ideas of this category are borrowed from the fast search of Vector Quantisation (VQ) [40].

5.4.1. Partial Distortion Elimination (PDE) Algorithm

This algorithm is the earliest algorithm in this category that has been widely used to reduce the computational complexity efficiently. It is employed in the FS algorithms in H.263 and H.264 [76], [86]. It uses the halfway-stop technique in the BDM calculation. In other words, the partial sum of matching distortion between current MB and candidate MB is stopped as soon as the matching distortion exceeds the current minimum distortion, meaning that the remaining computation is avoided. The conventional top-to-bottom kth partial SAD matching scan is determined as follows:

$$\sum_{i=1}^{k}\sum_{j=1}^{N}\left|C(i,j)-R\left(i+v_x,j+v_y\right)\right| \quad , \ k=1,2,\dots N \qquad (17)$$

where N represents MB size. If k is smaller than N and the summation exceeds the current SAD_{min}, then the remaining summation can quit and move to the next candidate MB.

The speed-up problem in this algorithm depends on: (1) fast searching, that is, how fast the global minimum in a given search range is detected; (2) fast matching error, that is, how to stop the calculation of the matching error early in the comparison process, which means finding the k value in Equation (20) faster to stop the partial sum [111-112].

The *fast searching* can be satisfied by applying the PDE algorithm with a spiral-ordered search starting at the centre of the search area since the best match location is usually centre-biased, then moving outward in a spiral design. This was employed in Telenor's H.263 codec [4].

The *fast matching* can be satisfied by eliminating the average number of rows examined per MB as well as the operations required. PDE employs SAD as a BDM to avoid more multiplication when calculating the matching error using MSE. Moreover, instead of the ordinary top-to-bottom matching scan, there are different scanning orders that improve performance of block matching. Kim et al. proposed various types of matching scan [76], [75], [72] depending on the relationship between block matching error and the spatial complexity of the reference MB, which is based on the concept of representative pixels. That is, the representative pixels are examined earlier than other pixels to detect the impossible candidates faster and reject them to obtain the reduction of computation in the block-matching algorithm.

This algorithm is called *adaptive matching scan algorithm based on gradient magnitude*. It utilises four directions: top-to-bottom, bottom-to-top, left-to-right, right-to-left. It uses gradient magnitude to measure the image complexity due to performance and computational complexity. In general, the gradient points in the direction of the maximum increase of a function. The gradient magnitude *G* can be calculated as follows:

$$|G[f(x,y)]| \approx |G_x| + |G_y| \approx |f(x,y) - f(x+1,y)| + |f(x,y) - f(x,y+1)| \quad (18)$$

The gradient magnitudes are calculated in four 8×8 sub-blocks of the candidate MB, as shown in Figure 24, and then make a sum of gradient magnitudes in sub-blocks, which are in four cases: *(1)+(2), (3)+(4), (1)+(3), (2)+(4)*. The maximum value of these sums points to the direction of matching scan; for example, the direction of matching scan is from top-to-bottom when the sum of gradient magnitudes *(1)* and *(2)* is maximum, as shown in 28 which describes this algorithm. The sub-block may be 4×4, i.e. there are 16 sub-blocks as in Jong-Nam et al. [72]. The matching scan order will also be according to the local complexity of the sub-block.

If the matching scan order is well arranged then the probability of eliminate the average number of rows examined increases.

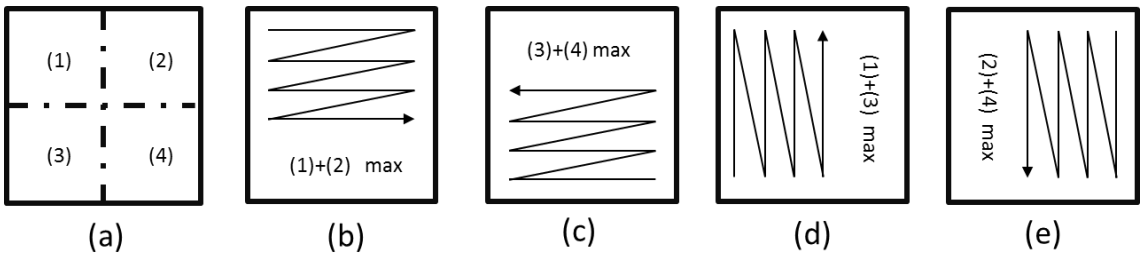

Figure 24: Adaptive matching scan based on representative pixels: (a) gradient magnitudes of sub-block division, (b) (top-to-bottom) matching scan when (1)+(2) is maximum, (c) bottom-to top matching scan when (3)+(4) is maximum, (d) left-to right when (1)+(3) is maximum, (e) right-to left when (2)+(4) is maximum [76]

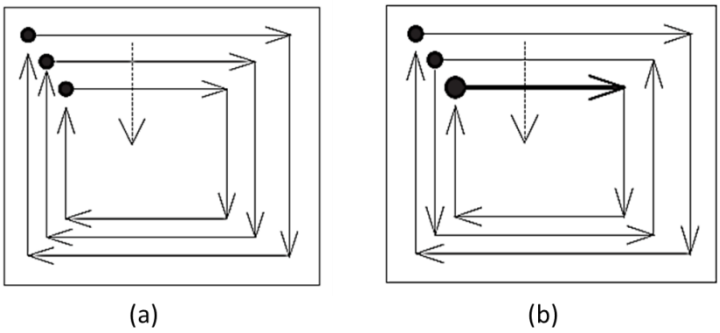

Figure 25: (a) spiralling inward scanning order, (b) alternating spiralling inward scanning order

However, these algorithms are not effective since decreasing the number of checking rows does not necessarily lead to enhancing the real time needed, because a lot of add/subtract operation is required per MB to compute the gradient magnitude in order to decide the matching order, which may render it unsuitable for real-time video coding systems. Therefore, three low complexity scanning orders were proposed by Grecos et al. (2004) which show improvements of ¼ operation count ratio and show an increase in the speed-up ratio of 45 times on average as compared with an adaptive matching scan algorithm based on gradient magnitude. Unlike the *adaptive matching scan algorithm,* two of Grecos et al.'s algorithms – *spiralling inward scanning order* and *alternating spiralling inward scanning order* – used fixed order of SAD computation between current and reference MBs to eliminate unsuitable predictors in the reference frame. These algorithms are based on the idea that the sides of the MB could represent the most information. Therefore, the representative pixels are examined earlier than other pixels without pre-processing, by computing the SAD value between pixels located on the sides of the squares of decreasing size inside the current and reference macroblocks, as shown in Figure 25, in order to reject impossible candidate predictors faster than the conventional top-to-bottom scan. The fixed direction scanning of the *spiralling inward scanning order* starts from top-horizontal and ends in left-vertical (Figure 25); it may increase computations since the complexity of candidate MB could be in any vertical or horizontal sides. If a candidate MB should be rejected on the basis of left-vertical SAD information then it has to wait until three sides of SAD computations are completed. For this reason, the *alternating spiralling inward scanning order* was

designed to reject the candidate MB on the basis of horizontal and vertical SAD information, as shown in Figure 29 (b).

The last algorithm of Grecos et al.'s, which is *horizontal/vertical scanning order*, utilises very limited pre-processing to avoid increasing the real time needed for computation and hence losing the benefit of computational reduction that happened with the *adaptive matching scan algorithm*. It determines the scanning order by examining only the SAD between the boundary rows and columns of the current and candidate MBs. The scanning direction will be the direction of the maximal SAD.

Successive Elimination Algorithm (SEA)

The SEA [85] eliminates impossible candidate MB by checking if the absolute difference between the summation of current MB pixels and the summation of candidate MB pixels is larger than the updated minimum SAD; if it is, then this candidate MB should be rejected. Thus, a large part of unnecessary computation for impossible candidate MBs can be avoided. This algorithm is based on the triangular mathematical inequality given by:

$$\left|\sum_{i} x_i\right| \leq \sum_{i} |x_i| \tag{19}$$

where x_i are arbitrary real numbers. Appling this inequality to the SAD achieves:

$$\begin{aligned}\left|\sum_{i=1}^{N}\sum_{j=1}^{N} C(i,j) - \sum_{i=1}^{N}\sum_{j=1}^{N} R(i+x, j+y)\right| &= \left|\sum_{i=1}^{N}\sum_{j=1}^{N} C(i,j) - R(i+x,j+y)\right| \\ &\leq \sum_{i=1}^{N}\sum_{j=1}^{N} |C(i,j) - R(i+x,j+y)|\end{aligned} \tag{20}$$

where $C(i,j)$ is the pixel value of current MB at the position (i,j) and $\mathrm{R}(i+x,j+y)$ is the pixel value of reference frame with the vector (x,y), which are within the search range $[-p,p]$. In other words, the previous inequality can be written as:

$$|SC - SR(x,y)| \leq SAD(x,y) \tag{21}$$

where SC is the summation of current MB and $SR(x,y)$is the summation of candidate MB at the vector (x,y). If $SAD_{min}(x_0,y_0)$ is the current updated minimum SAD at the search location (x_0,y_0), then to achieve better match MB at the location (x,y) the SAD should be less than SAD_{min} , that is $SAD(x,y) \leq SAD_{min}(x_0,y_0)$. This will

substitute in (22) to get: $|SC - SR(x,y)| \leq SAD_{min}(x_0, y_0)$. This means that a MB at location (x, y) can be immediately skipped from the search if:

$$|SC - SR(x,y)| \geq SAD_{min}(x_0, y_0) \quad (22)$$

While, if the difference $|\mathrm{SC} - \mathrm{SR}(\mathrm{x},\mathrm{y})|$ is smaller than $SAD_{min}(x_0, y_0)$, then the candidate MB is elected to calculate SAD between these two MBs and the new SAD becomes SAD_{min}. Since the candidate MBs are overlapping then the two horizontal neighbouring candidate MBs $SR(x, y)$ and $SR(x + 1, y)$ are also overlapping and they share N−1 columns. Therefore, subtracting the sum of the first column of MB $SR(x, y)$ and adding the sum of the last column in MB $SR(x + 1, y)$ will improve the block matching computation. A similar procedure can be used for vertical neighbouring candidate MBs.

Note that, similar to PDE, if the global minimum in a given search range is detected at the initial search, then SEA will be faster [54], [60]. Various algorithms have been introduced to enhance SEA [120], [73], [62], [91].

5.4.2 Enhanced Mean Predictive Block Matching Algorithm (EMPBM) Using Edge Detection

Enhanced Mean Predictive Block Matching Algorithm proposed to decrease the computations of the previous fast block matching algorithm Mean Predictive Block Matching algorithm [17-19]. In order to find the matching macroblock for the current macroblock from the previous frame, this technique classifies the current macroblock into shade and edge. The shade macroblock has a high probability to move in the same direction as its neighbouring macroblocks [67]. This will lead to search only the motion vectors of the neighbouring macroblocks and ignore other motion vectors that were utilised in the first search step of the Mean Predictive Block Matching algorithm. For edge macroblock, the proposed technique will use the same approach that was used in the Mean Predictive Block Matching algorithm [41-43].

Edge information can be described as a straight line across the macroblock with a sharp change of intensity in the spatial domain [20]. A fixed small size 4×4 macroblock is utilised to achieve good subjective quality. Therefore, this technique can be useful for small MBs in variable block-size motion estimation. In order to

avoid more computations in the existing edge detection methods, the absolute value approach has been used. The idea is to use the absolute value between the summation values of the vertical halves of the macroblock and the absolute value of the difference between the summation values of the horizontal halves, as shown in Figure 26.

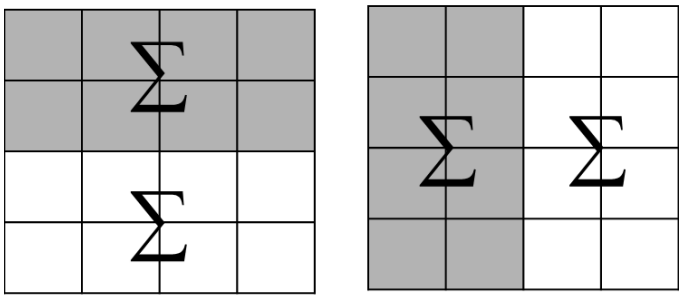

Figure 26: Vertical halves and horizontal halves for 4×4 MBs

When the sum of theses difference is less than a threshold value, the macroblock is classified as shade; otherwise, the macroblock will be classified as edge, as follows:
Let $B = \{b_{ij};\ 1 \le i,j \le 4\}$ represent a 4×4 frame macroblock. In this case, $b_{ij}$is a grey level pixel value corresponding to position (i, j) of row i and column j in the image block B. The discrete gradients of the macroblock B in the x and in the y directions are determined as follows:

$$G_x = \left| \sum_{i=1}^{2} \sum_{j=1}^{4} b_{ij} - \sum_{i=3}^{4} \sum_{j=1}^{4} b_{ij} \right|$$
$$G_y = \left| \sum_{i=1}^{4} \sum_{j=1}^{2} b_{ij} - \sum_{i=1}^{4} \sum_{j=3}^{4} b_{ij} \right| \qquad (23)$$

The gradient magnitude is defined by:

$$G = G_x + G_y \qquad (24)$$

If the gradient magnitude G in Equation (26) of the macroblock B is smaller than threshold T, then it is considered that the macroblock contains no significant gradient and it is classified as a shade macroblock; otherwise, it will be classified as an edge macroblock.

The shade macroblock has a high probability to move in the same direction of its neighbouring macroblock. This fact has been used in MPBM to decrease the search points [21].

6. DISCUSSION

Various block-based matching algorithms demonstrate varying behaviours in terms of quality measures, processing times, levels of distortion and the number of points evaluated during search procedures.

The number of search locations shown within FS operations are extremely resource intensive and a considerable number of computations are incurred during motion vector assignment as shown in Table 1.

The values also grow exponentially as the search parameter size increases and therefore computational complexity. FS technique renders a lossless form of video compression, whereas the subsequently developed fast block matching solutions are of the lossy category. Therefore, implementation of the FS technique within a video coding environment is largely dependent on the hardware constraints under which an individual operates. Fast block matching algorithms have been proposed to alleviate the demand on computational resources through the imperceptible loss of redundant visual content.

Maximum Displacement Region	Number of Search Locations
±1	9
±3	49
±7	225
±14	841
±28	3249

Table 1. The number of search location for FS.

TSS is considered as one of the pioneering solutions for fast, block-based motion estimation and remains a viable option for the compression of video sequences to this present day [3].

The number of search locations are fixed within each step with nine coordinates for the initial event and eight for subsequent events (as the point exhibiting the least distortion becomes the origin of successive search events). In order to provide an indication of the computational complexity associated with the TSS, Table 2 demonstrates the number of calculations required by this algorithm during certain displacement region scenarios.

Maximum Displacement Region	Number of Search Locations
±1	9
±3	17
±7	25
±14	33
±28	41

Table 2. The number of calculations required by TSS

The TSS and 2-D logarithmic search algorithms were both developed in order to alleviate the computational intensity associated with FS procedures and share a similar functional characteristic, in that their subsequent stages of execution employ a search pattern size of reduced dimensions (based on half that of the current step size). The most obvious difference between these two techniques is that the 2-D logarithmic search utilises a pattern with five checking points during the initial search steps in comparison to nine locations in the instance of the TSS. At stage in which the motion vector is specified, the 2-D logarithmic search uses a search pattern identical to that of a TSS window with step size equal to one.

In comparison to a TSS pattern, a reduction in the allocation of search points within the 2-D logarithmic search mean that distortion calculations for its initial search operations are minimised. Thus, the can 2-D logarithmic search be considered less resource intensive than the TSS for video sequences with small ranges of motion. The negative aspect of adopting a search pattern of this nature, is that the scenarios in which a significant step size is specified, the 2-D logarithmic search is forced to conduct several steps until the search area is narrowed towards the vicinity of the optimum motion vector.

The 2-D logarithmic search was released during the formative years of fast block matching algorithm development, as such the simulations conducted by Jain and Jain [68] provide no comparison with other homogeneous techniques. During this time, motion estimation technologies were in their infancy and their suitability for application within video coding yet to be extensively affirmed. Experimental investigations were still conducted however, with the peak-to-signal noise ratio of an uncompressed video sequence used as the baseline for comparison.

Within the 2-D logarithmic search publication, Jain and Jain [68] specify that the BDM used to assess the suitability of motion vectors for this technique is the Mean Absolute Error. Simulations conducted within this paper incorporate two test video sequences; namely “Cronkite” and “Chemical Plant”. Testing of the 2-D logarithmic search is implemented across sixteen frames of each video sequence, each with a resolution size of 256 x 256 pixels. Macroblocks used for the focal point of motion estimation between frames of the specified video sequences are comprised of 16 x 16 pixel dimensions.

Their PSNR figures present a strong argument for the validity of the 2-D logarithmic search as a solution for motion estimation processing. The experimental results indicate that the 2-D logarithmic search is consistently able to achieve higher levels of fidelity compared to instances where motion compensated frames are not applied. The figures also demonstrate that use of the 2-D logarithmic search for predicting motion vectors provides a reduced level of deviation on average in relation to PSNR values. This is particularly evident between frames six and seven of the “Cronkite” sequence, where a deficiency in motion compensation results in an interframe variance of approximately four dBs. The consistency of PSNR figures shown within the simulation results suggest that this technique unvaryingly navigates towards the domain of global minimum distortion.

Chronologically, Orthogonal Search [68] succeeded the initial endeavours of block-based algorithm development and adopted a similar search pattern configuration to that of the2-D logarithmic search. This pattern design was adapted so that the horizontal and vertical application of possible motion vector locations is conducted during subsequent stages of execution. This compounds the reduction of search point locations in comparison to that of the TSS as a maximum of three calculations are implemented at each step. Orthogonal Search simulations indicate that its computational complexity is approximately half that of the TSS, however steps of search double in frequency and therefore prolonged processing times are observed.

Proposal of the NTSS in the mid-nineties prompted the speculation of an additional assumption in relation to the behaviours exhibited by distortion distributions. This technique relied heavily on the suggestion that the motion vectors assigned from reference frames are in a centralised locality accordant to that of the candidate

macroblock coordinates. This reasoning is exploited by the NTSS [108] through the inclusion of an additional 3 x 3 coordinate search pattern during distortion level assessments in order to identify static macroblocks, at which point execution of the algorithm may be terminated. Implementation of two search patterns at the first step will incur seventeen BDM calculations and is therefore highly dependent of the validity of said assumption to ensure computational demands are not overly exuberant for motion estimations procedures.

Simulations were conducted by the developers of the NTSS with the TSS placed under identical testing conditions. The findings were particularly significant as the NTSS was shown to be consistently closer to point of global minimum distortion, but more significantly it is the most likely to identify the optimum motion vector match. The study does not however, indicate the required number of search locations for application of the NTSS within the tested video sequences. The NTSS is liable to incur an increased number of BDM calculations per step compared to that of the TSS, however this is mitigated by the innovative half-way stop technique which can also reduce processing times for static macroblocks.

The success demonstrated by the NTSS in adopting a centre-biased search functionality stimulated the emergence of a number of similar, block-based matching techniques in the years to follow. This is the framework in which algorithms described previously, such as the 4SS [82] and Block-Based Gradient Decent Search Algorithm (BBGDS) [127], are configured to exploit in order to increase the efficiency of motion estimation procedures whilst maintaining acceptable levels of fidelity.

Transformations made by the 4SS and BBGDS algorithms during the implementation of block matching, motion estimation were variable in regards to both computational complexity and also levels of observable distortion. Distortion levels incurred by the BBGDS in the tested video sequences were comparable with that of the NTSS, whilst computations were reduced by a factor of six. The ramifications of such, are that the computational resources required to achieve motion vector assignment are considerably less and the time required in order process such is also attenuated.

The ARPS algorithm is another pioneering block matching solution. The ARPS incorporates, what were at this period in time, two novel features; ZMP for stationary or quasi-stationary macroblocks and a predicted motion vector location within the primary search event. ZMP was shown to be effective for eliminating the superfluous computations that are incurred by previous algorithms for macroblocks with zero or minimal amounts of motion in the temporal domain of digital video sequences. The reasoning behind implementation of an additional, predicted motion vector location is that this ensures the path of search originates in an auspicious location and removes the likelihood of becoming trapped into a local minimum as is apparent in earlier techniques.

Scenarios in which the ARPS is applied without the implementation of ZMP, the number of motion vector assessments is reduced by a factor of two in comparison to the DS. Instances in which ZMP is utilised are shown to make further decreases in necessary computations, however this is less remarkable for high resolution video sequences due to the increases in paths of movement which obviously result in greater dispersion of motion vector locations between current and reference frames.

Table 3 shows the properties of the six well-studied block matching algorithms. The quality measures used to evaluate the performance of the block matching algorithm are shown in Table 4.

Algorithm	**Properties**
Full Search (FS) [110], [121], [60]	Simple full exhausted search algorithm used for benchmarking with the other techniques. Correlation window moves to each candidate position within the search area.
Diamond Search (DS) [113]	Utilises two search patterns
New Three Steps Search (NTSS) [108]	Provides improvement over the quality results of TSS. Widely accepted fast block matching algorithms. Used in earlier standards like MPEG 1 and H.261 [96].
Four Step Search (4SS) [82]	Halfway-stop technique with searching steps of 2 to 4. Reduces the computational requirement of full search.
Simple and Efficient TSS (**SESTSS) [70]**	Requires around half of the computation for TSS. keeping the same regularity and good performance of TSS.
Adaptive Rood Pattern Search (ARPS) [98]	Faster speed of 2-3 times than that DS. Maintains similar performance to the DS [138].

Table 3. Comparison of Six of the Standard Lossy Block Matching Algorithms.

Quality Measures	Calculation
Mean Square of the Error	$$MSE = \frac{1}{M \times N}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(f(i,j) - \hat{f}(i,j)\right)^2$$ M and N are the horizontal and vertical dimensions of the frame

	$f(i,j)$ = pixels values at location (i,j) $\hat{f}(i,j)$ = predicted values at location (i,j)
Processing time	$t_s =$ search time per macro block
Peak Signal to Noise Ratio	$PSNR = 10\ log_{10}\left(\frac{(f_{max})^2}{MSE}\right)$
Mean Absolute Differences	$MAD = \frac{1}{M \times N}\sum_{i=1}^{M}\sum_{j=1}^{N}\left[\left(f(i,j) - \hat{f}(i,j)\right)\right]$
Sum of Absolute difference	$SAD = \sum_{i=1}^{M}\sum_{j=1}^{N}\left[\left(f(i,j) - \hat{f}(i,j)\right)\right]$

Table 4: Quality measures to evaluate the performance of the block matching algorithms.

To evaluate the performance of those standard block marching algorithms, Table 5 shows the average number of search points for a Microblock of size 16x16 which indicates that significant improvements are shown in comparison to the FS algorithm. Table 6 shows that the performance of fast black matching algorithms in terms of PSNR is comparable to the FS algorithms.

Sequence	Format	FS	DS	NTSS	4SS	SESTSS	ARPS
Claire	QCIF	184.6	11.63	15.09	14.77	16.13	5.191
Akiyo	QCIF	184.6	11.46	14.76	14.67	16.2	4.958
Carphone	QCIF	184.6	13.76	17.71	16.12	15.73	7.74
News	CIF	204.3	13.1	17.07	16.38	16.92	6.058
Stefan	CIF	204.3	17.69	22.56	19.05	16.11	9.641
Coastguard	CIF	204.3	19.08	27.26	19.91	16.52	9.474

Table 5: Average number of search points per MB of size 16 ×16

Sequence	Format	FS	DS	NTSS	4SS	SESTSS	ARPS
Calire	QCIF	38.94	38.94	38.94	38.92	38.89	38.94
Akiyo	QCIF	39.61	39.61	39.61	39.61	39.61	39.61
Carphone	QCIF	30.82	30.69	30.7	30.4	30.1	30.58
News	CIF	33.77	33.45	33.63	33.42	33.19	33.39
Stefan	CIF	22.16	21.49	21.81	21.51	21.04	21.82
Coastguard	CIF	26.19	25.98	26.05	26.02	25.6	26.05

Table 6: The simulation results of mean PSNR for 50 frames

7. CONCLUSION

Our investigative study into fast block matching algorithms has provided insight into the competitiveness of such solutions with regard to their computational complexity, processing times and levels of observed distortion. We have looked at various block matching algorithms and emphasise on their properties, this is different to other survey researches which looked at video compressions and briefly discussed block matching algorithms as illustrated in Table 6. In this survey, we have found that TSS was able to alleviate the resource intensity of FS operations, whilst providing satisfactory fidelity levels within encoded video sequences. Thus, the solution was widely adopted for motion estimation procedures [3], until the mid-nineties when centre-biased search pattern algorithms were introduced with the proposal of the NTSS. The NTSS demonstrated similar distortion levels to that as the TSS, however the additional (centralised) search window and half-way stop functionalities achieved transformations in regards to computational complexity and therefore times for processing motion vectors. Several homogeneous techniques such as 4SS emerged, providing further gains than that of the NTSS, albeit for contrasting operational benefits.

A shift in search pattern design was demonstrated by the development of the DS algorithm, which also intended to exploit the assumption that the majority of true motion vectors reside in a centralised locality. The DS achieved this to a degree by providing desirable levels of fidelity, whilst minimising the number distortion calculations in comparison to that of the NTSS and 4SS. The reductions made in terms of computational complexity by the BBGDS are evidently more remarkable and suggest the hexagonal conformation of motion vector points are less effective than standard 3 x 3, square-based windows of search.

Ref.	Topics	Summary
[140]	Survey in perceptual video compression	The paper has defined three important stages in developing perceptual video compression algorithms which are perceptual model definition, implementation of coding, and performance evaluation
[141]	Various techniques of video compression was survived	Indicated that four-step search algorithm for fast block motion estimation is widely used video compression techniques.
[142]	A Review on Motion Estimation in Video Compression	Concluded that there are still lots of improvement of video compression technique still to be searched
[143]	Classification of motion estimation algorithms used for video compression were discussed	The paper focused on block matching algorithms indicating that they are widely used in MPEG1 / H.261 to MPEG4 / H.263 and H.264/AVC
[144]	Looked at motion estimation techniques	The paper looked at MPEG and how it make use of the temporal redundancy inherent in sequences of full-motion video to achieve compression

Table 7: Selected examples of survey on video compression algorithms

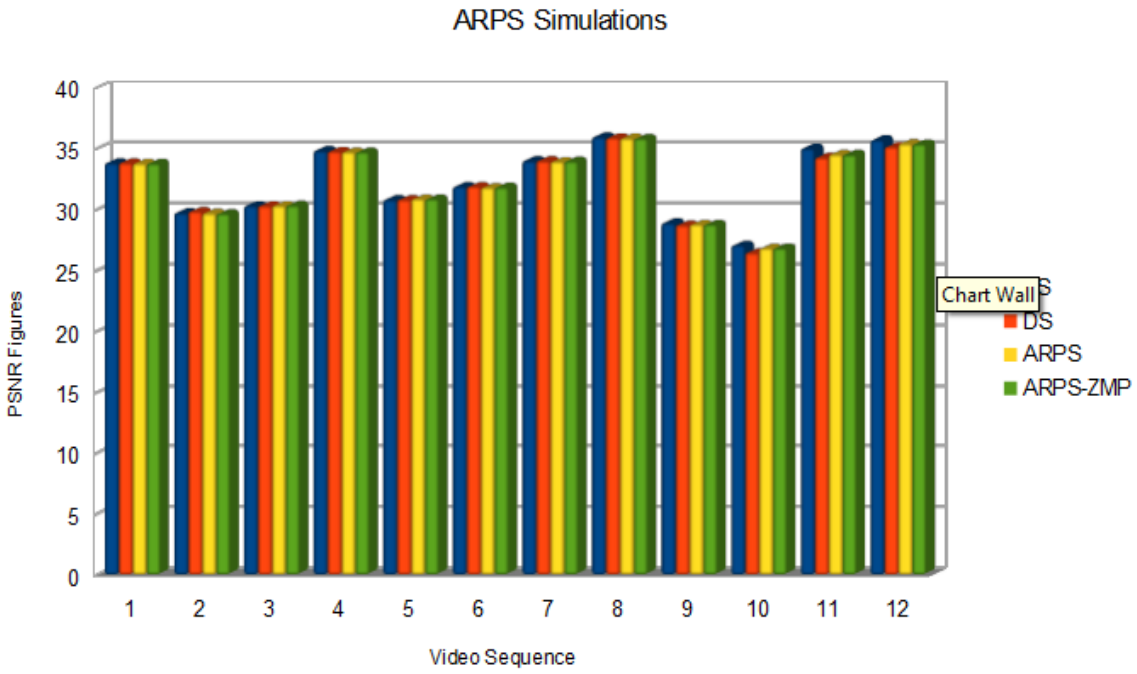

Figure 27. Comparison between DS, ARPS and ARPS-ZMP [98]

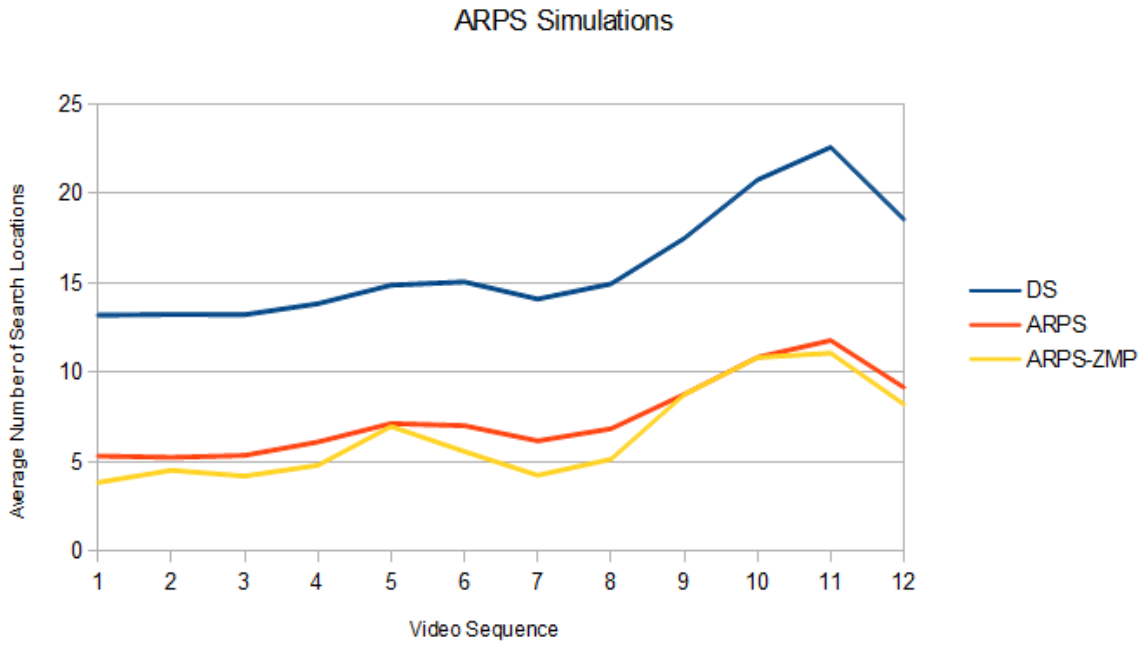

Figure 28. Average number of search location [98]

ARPS experimental data provides empirical evidence showing that its observed distortion levels are comparable with that of the DS (demonstrated by Figure 27), but also demonstrates that approximately half the calculations are required in order to

achieve such, as shown in Figure 28. Thus, from our investigation into a variety of fast block matching algorithms it can be determined that ARPS can provide competitive solution due to its computational simplicity and the desirable levels of fidelity in which are achieved.

Ref.	Technique	Details
Jha et al. [145]	Wavelet Based	Hybrid video compression algorithm. Adaptive motion compensation scheme is used. Spatial orientation tree modified zero tree algorithms are also used
Fabrizio et. al. [146]	Particle Swarm Optimization	PSO approach used to achieve high accuracy in block matching.
Cai et al. [147]	Block matching using DCT & DWT	Develop block matching algorithm with DCT & DWT
Aziz et al [149]	Wavelet domain	Develop motion estimation and compensation in the wavelet domain.
Pandian et al. [150]	PCA	PCA applied to the frames. It algorithm kept the bandwidth of frequency and improve the Edges of frames

Table 8: Summary of video compression techniques

In summary, modern world video compression technology is developed to be in one of the bloomed field of research, there are enormous techniques available for a wide range of applications as illustrated in Table 8.

References

[1] TIAN, L., LI, S., AHN, J., CHU, D., HAN, R., LV, Q. & MISHRA, S. (2013). "Understanding User Behavior at Scale in a Mobile Video Chat Application". In Proc. UbiComp '13 ACM International Joint Conference on Pervasive and Ubiquitous Computing; pp.647-656.

[2] EMARKETER (2013). "*Mobile, Video Drive Up Digital Ad Investment in the UK".* [Online]. Available: http://www.emarketer.com/Article/Mobile-Video-Drive-Up-Digital-Ad-Investment-UK/1010097; [Accessed 5.05. 2017].

[3] MUKHTAR, H., AL-DWEIK, A., AL-MUALLA, M. (2016). “Content-Aware and Occupancy-Based Hybrid ARQ for Video Transmission”, IEEE 59TH International Midwest Symposium on Circuits and Systems, Abu Dhabi, United Arab Emirates.

[4] AL-MUALLA, M. E., CANAGARAJAH, C. N. & BULL, D. R. (2002). "Video Coding for Mobile Communications: Efficiency, Complexity and Resilience": *Academic Press*.

[5] ISO/IEC (1993). "Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 2: video". ISO/IEC 11172-2.

[6] ISO/IEC (1996). "Information technology – Generic coding of moving pictures and associated audio – Part 2: video". ISO/IEC 13818-2.

[7] ISO/IEC (1999). "Coding of moving pictures and associated audio for digital storage media at up to about 1.5 Mbit/s". ISO/IEC 11172-3.

[8] ITU-T & ISO/IEC (2003). "Advanced Video Coding for Generic Audiovisual Services". H.264, MPEG, 14496-10.

[9] SULLIVAN, G., TOPIWALA, P. & LUTHRA, A. (2004). "The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions". *SPIE conference on Applications of Digital Image Processing XXVII*.

[10] SULLIVAN, G. J. & WIEGAND, T. (2005). "Video Compression - From Concepts to the H.264/AVC Standard". *Proceedings of the IEEE;* Vol.93(1); pp.18-31.

[11] OHM, J. & SULLIVAN, G. J. (2013). "High Efficiency Video Coding: The Next Frontier in Video Compression [Standards in a Nutshell]". *IEEE Signal Processing Magazine;* Vol.30(1); pp.152-158.

[12] RICHARDSON, I. (2003). "H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia": *Wiley*.

[13] RICHARDSON, I. E. G. (2010). "The H.264 advanced video compression standard", *2nd* edition; UK: *John Wiley & Sons Inc*.

[14] AKRAM, M. & IZQUIERDO, E. (2010). "A Multi-Pattern Search Algorithm for Block Motion Estimation in Video Coding". *In Proc. IEEE 12th International Asia-Pacific Web Conference (APWEB)*; pp.407-410.

[15] D.S.Huang, Systematic Theory of Neural Networks for Pattern Recognition, Publishing House of Electronic Industry of China, May 1996.

[16] D.S.Huang, “Radial basis probabilistic neural networks: Model and application,” International Journal of Pattern Recognition and Artificial Intelligence, 13(7), pp.1083-1101, 1999.

[17] AHMED, Z., HUSSAIN, A. J. & AL-JUMEILY, D. (2011a). "Enhanced Computation Time for Fast Block Matching Algorithm". *In Proc. IEEE Developments in E-systems Engineering (DeSE)*; pp.289-293.

[18] AHMED, Z., HUSSAIN, A. J. & AL-JUMEILY, D. (2011b). "Mean Predictive Block Matching (MPBM) for fast block-matching motion estimation". *In Proc. IEEE 3rd European Workshop on Visual Information Processing (EUVIP)*; pp.67-72.

[19] AHMED, Z., HUSSAIN, A. J. & AL-JUMEILY, D. (2012). "Edge detection for fast block-matching motion estimation to enhance Mean Predictive Block Matching algorithm". *In Proc. IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*; pp.1-5.

[20] AL-FAYADH, A., HUSSAIN, A. J., LISBOA, P., AND AL-JUMEILY, D. (2009). "Novel hybrid classified vector quantization using discrete cosine transform for image compression ". *Journal of Electronic Imaging;* Vol.18(2).

[21] W.B.Zhao, D.S.Huang, Ji-Yan Du and Li-Ming Wang, “Genetic optimization of radial basis probabilistic neural networks,” International Journal of Pattern Recognition and Artificial Intelligence, vol. 18, no. 8, pp. 1473-1500, 2004.

[22] ANANTHASHAYANA, V. K. & PUSHPA, M. K. (2009). "Joint Adaptive Block Matching Search (JABMS) Algorithm for Motion Estimation". *International* Journal of Recent Trends in Engineering; Vol.2(2); pp.212-216.

[23] D.S.Huang, Ji-Xiang Du, “A constructive hybrid structure optimization methodology for radial basis probabilistic neural networks,” IEEE Transactions on Neural Networks, vol. 19, no.12, pp 2099-2115, 2008.

[24] D.S.Huang, W.B.Zhao, "Determining the centers of radial basis probabilistic neural networks by recursive orthogonal least square algorithms," Applied Mathematics and Computation, vol.162, no.1, pp.461-473, 2005.

[25] D.S.Huang and S.D.Ma,“Linear and nonlinear feedforward neural network classifiers: A comprehensive understanding,”Journal of Intelligent Systems, vol.9, no.1, pp.1-38,1999.

[26] BARJATYA, A. (2004). "Block Matching Algorithms for Motion Estimation". *Final Project Paper, DIP 6620.*

[27] BHASKARAN, V. & KONSTANTINIDES, K. (1997). "Image and Video Compression Standards: Algorithms and Architecture", *2nd* edition: *Kluwer Academic Publishers*.

[28] D.S. Huang, The Study of Data Mining Methods for Gene Expression Profiles, Science Press of China, March 2009.

[29] D.S.Huang, “A constructive approach for finding arbitrary roots of polynomials by neural networks,” IEEE Transactions on Neural Networks, vol.15, no.2, pp.477-491, 2004.

[30] D.S.Huang, and Wen Jiang, “A general CPL-AdS methodology for fixing dynamic parameters in dual environments,” IEEE Trans. on Systems, Man and Cybernetics - Part B, vol.42, no.5, pp.1489-1500, 2012.

[31] BOVIK, A. (2010). "Handbook of Image and Video Processing", *2nd* edition: *Academic Press*.

[32] BOVIK, A. C. (2009). "Chapter 1 - Introduction to Digital Video Processing". *In: The Essential Guide to Video Processing,* 2nd edition; Boston: Academic Press.

[33] BROSS, B., HAN, W. J., OHM, J. R., SULLIVAN, G. J. & WEINGAND, T. (2012). "High Efficiency Video Coding (HEVC), text specification draft 6". *Doc. JCTVC-H1003, Joint Collaborative Team on Video Coding (JCT-VC) of ITU-T VCEG and ISO/IEC MPEG;* Vol.21.

[34] CAI, C., ZENG, H. & MITRA, S. (2009). "Fast motion estimation for H.264". *Signal Processing: Image Communication*.

[35] D.S.Huang, Horace H.S.Ip, Law Ken C K and Zheru Chi, ”Zeroing polynomials using modified constrained neural network approach,” IEEE Trans. On Neural Networks, vol.16, no.3, pp.721-732, 2005.

[36] Xiao-Feng Wang, D.S.Huang and Huan Xu, "An efficient local Chan-Vese model for image segmentation," Pattern Recognition, vol. 43, no.3, pp. 603-618, 2010.

[37] Xiao-Feng Wang, D.S.Huang, "A novel density-based clustering framework by using level set method," IEEE Transactions on Knowledge and Data Engineering, vol. 21, no.11, pp 1515-1531, 2009.

[38] CE, Z., XIAO, L., CHAU, L. & LAI-MAN, P. (2004). "Enhanced hexagonal search for fast block motion estimation". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.14(10); pp.1210-1214.

[39] CHALIDABHONGSE, J. & KUO, C. C. J. (1997). "Fast motion vector estimation using multiresolution-spatio-temporal correlations". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.7(3); pp.477-488.

[40] CHANG-DA, B. & GRAY, R. (1985). "An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization". *IEEE Transactions on Communications;* Vol.33(10); pp.1132-1133.

[41] Wen Jiang, D.S.Huang, Shenghong Li, “Random-walk based solution to triple level stochastic point location problem,” IEEE Trans. on Cybernetics, vol.46, no.6, pp.1438-1451, 2016.

[42] Zhan-Li Sun, D.S.Huang, and Yiu-Ming Cheung, “Extracting nonlinear features for multispectral images by FCMC and KPCA,” Digital Signal Processing, vol.15, no.4, 331-346, 2005.

[43] Zhan-Li Sun, D.S.Huang, Yiu-Ming Cheung, Jiming Liu and Guang-Bin Huang, “Using FCMC, FVS and PCA techniques for feature extraction of multispectral images,” IEEE Geoscience and Remote Sensing Letters, vol.2, no.2, pp.108-112, 2005.

[44] CHANYUL, K. (2010). "*Complexity adaptation in video encoders for power limited platforms".* Dublin City University.

[45] CHAO-FENG, T., YEN-TAI, L. & MENG-JE, L. (2012). "A VLSI architecture for three-step search with variable block size motion vector". *In Proc. IEEE 1st Global Conference on Consumer Electronics (GCCE)*; pp.628-631.

[46] CHEUNG, C.-H. & PO, L.-M. (2002). "A novel cross-diamond search algorithm for fast block motion estimation". *IEEE Trans. Circuits Syst. Video Technol;* Vol.12(12); pp.1168- 1177.

[47] Fischer, W. (2010). Digital and Audio Broadcasting Technology: A Practical Engineering Guide, Signals and Communication Technology, 3rd ed., DOI 10.1007/978-3-642-11612-4_2.

[48] ERTURK, S. (2007). "Multiplication-Free One-Bit Transform for Low-Complexity Block-Based Motion Estimation". *IEEE Signal Processing Letters;* Vol.14(2); pp.109-112.

[49] Jian-Xun Mi, D.S.Huang, Bing Wang, Xingjie Zhu, “The nearest-farthest subspace classification for face recognition,” Neurocomputing, vol.113, pp.241-250, 2013.

[50] Can-Yi Lu, and D.S.Huang, “Optimized projections for sparse representation based classification,” Neurocomputing, vol.113, pp.213-219, 2013.

[51] Yang Zhao, and D.S.Huang, “Completed local binary count for rotation invariant texture classification,” IEEE Trans. on Image Processing, vol.21, no.10, pp. 4492 - 4497, 2012.

[52] Bo Li, Chao Wang and D.S.Huang, “Supervised feature extraction based on orthogonal discriminant projection,” Neurocomputing, vol. 73, nos.1-3, pp 191-196, 2009.

[53] Xiao-Feng Wang, D.S.Huang, Ji-Xiang Du, Huan Xu, Laurent Heutte, “Classification of plant leaf images with complicated background,” Applied Mathematics and Computation, vol. 205, no.2, pp 916-926, 2008.

[54] ESSANNOUNI, F., THAMI, R. O. H., SALAM, A. & ABOUTAJDINE, D. (2006). "An efficient fast full search block matching algorithm using FFT algorithms". *IJCSNS International Journal of Computer Science a 130 nd Network Security;* Vol.6(3); pp.130-133.

[55] EZHILARASAN, M. & THAMBIDURAI, P. (2008). "Simplified Block Matching Algorithm for Fast Motion Estimation in Video Compression ". *Journal of Computer Science;* Vol.4(4); pp.282-289.

[56] GOEL, S. & BAYOUMI, M. A. (2006). "Multi-Path Search Algorithm for Block-Based Motion Estimation". *In Proc. of IEEE International Conference on Image Processing*; pp.2373-2376.

[57] GONZALEZ, R., WOODS, R. & EDDINS, S. (2009). "Digital Image processing using MATLAB", *2nd* edition: *Gatesmark Publishing*.

[58] HORN, B. & SCHUNCK, B. (1981). "Determining Optical Flow". *ARTIFICAL INTELLIGENCE;* Vol.17; pp.185-203.

[59] HUANG, D.Y. (2005). "*A XviD-based Video Codec for Computer Animation".* MSC Thesis, National Central University.

[60] HUANG, Y.W., CHEN, C.-Y., TSAI, C.-H., SHEN, C.-F. & CHEN, L.-G. (2006). "Survey on Block Matching Motion Estimation Algorithms and Architectures with New Results". *The Journal of VLSI Signal Processing;* Vol.42(3); pp.297-320.

[61] HUI-YU, H. & SHIH-HSU, C. (2011). "Block motion estimation based on search pattern and predictor". *In Proc. IEEE Symposium on Computational Intelligence for Multimedia, Signal and Vision Processing (CIMSIVP)*; pp.47-51.

[62] HWAL-SUK, L., JIK-HAN, J. & DONG-JO, P. (2008). "An effective successive elimination algorithm for fast optimal block-matching motion estimation". In Proc. of IEEE 15th International Conference on Image Processing (ICIP); pp.1984-1987.

[63] D.S.Huang, Xing-Ming Zhao, Guang-Bin Huang, and Yiu-Ming Cheung, “Classifying protein sequences using hydropathy blocks,” Pattern Recognition, vol.39, no.12, pp. 2293–2300,2006.

[64] Chun-Hou Zheng, D.S.Huang, and Li Shang, “Feature selection in independent component subspace for microarray data classification,” Neurocomputing, vol.69, nos.16-18, pp.2407-2410, 2006.

[65] Jun Zhang, D.S.Huang, Tat-Ming Lok, and Michael R. Lyu, “A novel adaptive sequential niche technique for multimodal function optimization,” Neurocomputing, vol.69, nos.16-18, pp.2396-2401, 2006.

[66] Fei Han, D.S.Huang, "Improved extreme learning machine for function approximation by encoding a priori information," Neurocomputing, vol.69, nos.16-18, pp.2369-2373, 2006.

[67] JAE-YONG, K. & SUNG-BONG, Y. (1999). "An efficient hybrid search algorithm for fast block matching in video coding". *Proceedings of the IEEE the Region 10 Conference TENCON 99*; Vol.1; pp.112-115.

[68] JAIN, J. & JAIN, A. (1981). "Displacement Measurement and Its Application in Interframe Image Coding". *IEEE Transactions on Communications;* Vol.29(12); pp.1799-1808.

[69] JIAN, F., KWOK-TUNG, L., MEHRPOUR, H. & KARBOWIAK, A. E. (1995). "Adaptive block matching motion estimation algorithm using bit-plane matching". *In Proc. IEEE International Conference on Image Processing*; Vol.3; pp.496-499.

[70] JIANHUA, L. & LIOU, M. L. (1997). "A simple and efficient search algorithm for block-matching motion estimation". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.7(2); pp.429-433.

[71] JIZHENG, X., FENG, W. & WENJUN, Z. (2009). "Intra-Predictive Transforms for Block-Based Image Coding". *Signal Processing, IEEE Transactions on;* Vol.57(8); pp.3030-3040.

[72] JONG-NAM, K., SUNG-CHEAL, B. & BYUNG-HA, A. (2001). "Fast Full Search Motion Estimation Algorithm Using various Matching Scans in Video Coding". *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews;* Vol.31(4); pp.540-548.

[73] JUNG, S. M., SHIN, S. C., BAIK, H. & PARK, M. S. (2002). "Efficient multilevel successive elimination algorithms for block matching motion estimation". *IEE Proceedings -Vision, Image and Signal Processing;* Vol.149(2); pp.73-84.

[74] KIM, C. (2010). "*Complexity Adaptation in Video Encoders for Power Limited Platforms".* PhD Thesis, Dublin City University.

[75] KIM JONG-NAM, BYUN SUNG-CHEAL, KIM YONG-HOON & AHN BYUNG-HA (2002). "Fast full search motion estimation algorithm using early detection of impossible candidate vectors". *IEEE Transactions on]Signal Processing;* Vol.50(9); pp.2355-2365.

[76] KIM JONG-NAM & CHOI TAE-SUN (2000). "A fast full-search motion-estimation algorithm using representative pixels and adaptive matching scan". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.10(7); pp.1040-1048.

[77] Xiao-Feng Wang, D.S.Huang, “A novel multi-layer level set method for image segmentation,” Journal of Universal Computer Science, vol.14, no.14, pp.2428-2452, 2008.

[78] Bo Li, D.S.Huang, Chao Wang and Kun-Hong Liu, “Feature extraction using constrained maximum variance mapping,” Pattern Recognition, vol.41, no.11, pp. 3287-3294, 2008.

[79] Zhong-Qiu Zhao, D.S.Huang, “Palmprint recognition with 2DPCA+PCA based on modular neural networks,” Neurocomputing, vol.71, nos.1-3, pp. 448-454, 2007.

[80] KOGA, T., ILINUMA, K., HIRANO, A., IIJIMA, Y. & Y.ISHIGURO (1981). "Motion compensated interframe coding for video conferencing". National Telecommum Conference; pp.531–535.

[81] KOU, W. (1995). "Digital Image Compression: Algorithms and Standards": *Kluwer Academic Publishers*.

[82] LAI-MAN, P. & WING-CHUNG, M. (1996). "A novel four-step search algorithm for fast block motion estimation". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.6; pp. 313–317.

[83] LEONTARIS, A., COSMAN, P. C. & TOURAPIS, A. M. (2009). "Multiple Reference Motion Compensation: A Tutorial Introduction and Survey". *Found. Trends Signal Process;* Vol.2(4); pp.247-364.

[84] LI LIU, COHEN, R., SUN, H., VETRO, A. & ZHUANG, X. (2010). "New Techniques for Next Generation Video Coding". *In Proc. IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*; pp.111 - 116.

[85] LI, W. & SALARI, E. (1995). "Successive elimination algorithm for motion estimation". *IEEE Transactions on Image Processing;* Vol.4(1); pp.105-107.

[86] LIN CHEN-FU & LEOU JIN-JANG (2005). "An adaptive fast full search motion estimation algorithm for H.264". *In Proc. IEEE International Symposium on Circuits and Systems ISCAS*; Vol.2; pp.1493-1496.

[87] LIU, B. & ZACCARIN, A. (1993). "New fast algorithms for the estimation of block motion vectors". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.3(2); pp.148-157.

[88] Ji-Xiang Du, D.S.Huang, Xiao-Feng Wang, Xiao Gu, “Shape recognition based on neural networks trained by differential evolution algorithm,” Neurocomputing, vol.70, nos.4-6, pp. 896-903, 2007.

[89] Ji-Xiang Du, D.S.Huang, Guo-Jun Zhang and Zeng-Fu Wang, “A novel full structure optimization algorithm for radial basis probabilistic neural networks,” Neurocomputing, vol.70, nos.1-3, pp. 592-596, 2006.

[90] Ji-Xiang Du, D.S.Huang, Xiao-Feng Wang, and Xiao Gu, “Computer-aided plant species identification (CAPSI) based on leaf shape matching technique,” Transactions of the Institute of Measurement and Control, vol. 28, no. 3, pp. 275-284, 2006.

[91] MAN-YAU, C. & WAN-CHI, S. (2006). "New results on exhaustive search algorithm for motion estimation using adaptive partial distortion search and successive elimination algorithm". *In Proc. IEEE International Symposium on Circuits and Systems ISCAS*; pp.3977-3981.

[92] MARPE, D., WIEGAND, T. & SULLIVAN, G. J. (2006). "The H.264/MPEG4 advanced video coding standard and its applications". *Communications Magazine, IEEE;* Vol.44(8); pp.134-143.

[93] METKAR, S. & TALBAR, S. (2010). "Fast motion estimation using modified orthogonal search algorithm for video compression". *Signal, Image and Video Processing;* Vol.4; pp.123–128.

[94] MIZUKI, M. M., DESAI, U. Y., MASAKI, I. & CHANDRAKASAN, A. (1996). "A binary block matching architecture with reduced power consumption and silicon area requirement". *In Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP*; Vol.6; pp.3248-3251.

[95] MOERITZ, S. & DIEPOLD, K. (2004). "Understanding MPEG 4: Technology and Business Insights*"*: *Focal Press*.

[96] MOGUS, F. A., XINYING, L. & LEI, W. (2010). "Evaluation of the performance of motion Estimation algorithms in video coding". *In Proc. IEEE 2nd International Conference on Information Science and Engineering (ICISE)*; pp.3693-3696.

[97] NATARAJAN, B., BHASKARAN, V. & KONSTANTINIDES, K. (1997). "Low-complexity block-based motion estimation via one-bit transforms". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.7(4); pp.702-706.

[98] NIE, Y. & MA, K.-K. (2002). "Adaptive rood pattern search for fast block-matching motion estimation ". *IEEE Trans on Image Processing;* Vol.11(12); pp.1442-1448.

[99] NIGHTINGALE, J., QI, W. & GRECOS, C. (2012). "HEVStream: A Framework for Streaming and Evaluation of High Efficiency Video Coding (HEVC) Content in

Loss-prone Networks". *IEEE Transactions on Consumer Electronics;* Vol.58(2); pp.404-412.

[100] Li Shang, D.S.Huang, Ji-Xiang Du, and Chun-Hou Zheng, " Palmprint recognition using FastICA algorithm and radial basis probabilistic neural network," Neurocomputing, vol.69, nos.13-15, pp. 1782-1786, 2006.

[101] Zhan-Li Sun, D.S.Huang, and Chun-Hou Zheng, Li Shang, “Optimal selection of time lags for temporal blind source separation based on genetic algorithm,” Neurocomputing, vol.69, nos.7-9, pp.884–887, 2006.

[102] Chun-Hou Zheng, D.S.Huang, Zhan-Li Sun, Michael R. Lyu, and Tat-Ming Lok, "Nonnegative independent component analysis based on minimizing mutual information technique," Neurocomputing, vol.69, nos.7-9, pp.878–883, 2006.

[103] Li Shang, D.S.Huang, Chun-Hou Zheng, and Zhan-Li Sun, "Noise removal using a novel non-negative sparse coding shrinkage technique," Neurocomputing, vol.69, nos.7-9, pp.874–877, 2006.

[104] PEREIRA, F. C. & EBRAHIMI, T. (2002). "The MPEG-4 Book": *Prentice Hall PTR*.

[105] PRASANTHA, H. S., SHASHIDHARA, H. L. & BALASUBRAMANYA MURTHY, K. N. (2007). "Image Compression Using SVD". *In Proc. IEEE International Conference on Conference on Computational Intelligence and Multimedia Applications*; Vol.3; pp.143-145.

[106] PU, I. M. (2005). "Fundamental Data Compression": *Butterworth-Heinemann*.

[107] PURI, A., HANG, H. M. & SCHILLING, D. (1987). "An efficient block-matching algorithm for motion-compensated coding". *IEEE International Conference on Acoustics, Speech, and Signal Processing;* Vol.12; pp.1063-1066.

[108] REOXIANG, L., BING, Z. & LIOU, M. L. (1994). "A new three-step search algorithm for block motion estimation". *IEEE Transactions on Circuits and Systems for Video Technology;* Vol.4(4); pp.438-442.

[109] RUIZ, G. A. & MICHELL, J. A. (2011). "An efficient VLSI processor chip for variable block size integer motion estimation in H.264/AVC". *Signal Processing: Image Communication;* Vol.26(6); pp.289-303.

[110] SAYOOD, K. (2006). "Introduction to Data Compression", *3rd* edition: *Morgan Kaufmann*.

[111] Xiao-Feng Wang, D.S.Huang, Ji-Xiang Du, Huan Xu, Laurent Heutte, “Classification of plant leaf images with complicated background,” Applied Mathematics and Computation, vol. 205, no.2, pp 916-926, 2008.

[112] Xiao-Feng Wang, D.S.Huang, “A novel multi-layer level set method for image segmentation,” Journal of Universal Computer Science, vol.14, no.14, pp.2428-2452, 2008.

[113] SHAN, Z. & KAI-KUANG, M. (1997). "A new diamond search algorithm for fast block matching motion estimation". *In Proc. IEEE International Conference on Information, Communications and Signal Processing (ICICS)*; Vol.1; pp.292-296.

[114] SONG, B. C. & RA, J. B. (1998). "A hierarchical block matching algorithm using partial distortion measure". *In Proc. SPIE Visual Communications and Image Processing '98*; Vol.3309; pp.88-95.

[115] Bo Li, D.S.Huang, Chao Wang and Kun-Hong Liu, “Feature extraction using constrained maximum variance mapping,” Pattern Recognition, vol.41, no.11, pp. 3287-3294, 2008.

[116] Kun-Hong Liu, and D.S.Huang, “Cancer classification using rotation forest,” Computers in Biology and Medicine, vol. 38, no. 5, pp.601-610, 2008.

[117] Fei Han, D.S.Huang, “A new constrained learning algorithm for function approximation by encoding a priori information into feedforward neural networks,” Neural Computing and Applications, vol.17, nos.5-6, pp.433-439, 2008.

[118] Fei Han, Qing-Hua Ling, and D.S.Huang, “Modified constrained learning algorithms incorporating additional functional constraints into neural networks,” Information Sciences, vol.178, no.3, pp.907-919, 2008.

[119] Chun-Hou Zheng, D.S.Huang, Kang Li, George W Irwin and Zhan-Li Sun, "MISEP method for Post-Nonlinear Blind Source Separation,” Neural Computation, vol.19, no.9, pp.2557-2578, 2007.

[120] SOO-MOK, J., SUNG-CHUL, S., HYUNKI, B. & MYONG-SOON, P. (2000). "Nobel successive elimination algorithms for the estimation of motion vectors". *In Proc. IEEE International Symposium on Multimedia Software Engineering*; pp.332-335.

[121] SRINIVASAN, R. & RAO, K. R. (1985). "Predictive coding based on efficient motion estimation". *IEEE Transactions on Communications;* Vol.33(8); pp.888–896.

[122] TURAGA, D. & CHEN, T. (2001). "I/P Frame Selection Using Classification Based Mode Decision". *International Conference on Image Processing ICIP*; pp.550-553.

[124] Ji-Xiang Du, D.S.Huang, Xiao-Feng Wang, Xiao Gu, “Shape recognition based on neural networks trained by differential evolution algorithm,” Neurocomputing, vol.70, nos.4-6, pp. 896-903, 2007.

[125] Ji-Xiang Du, D.S.Huang, Guo-Jun Zhang and Zeng-Fu Wang, “A novel full structure optimization algorithm for radial basis probabilistic neural networks,” Neurocomputing, vol.70, nos.1-3, pp. 592-596, 2006.

[126] VANNE, J. (2011). "Design and Implementation of Configurable Motion Estimation Architecture for Video Encoding". PhD Thesis, Tampere University of Technology

[127] Liu, L.K. And Feig, E (1996) A Block-Based Gradient Decent Search Algorithm for Block Motion Estimation in Video Coding - IEEE Trans. Circuits and Systems for Video Technology – Vol. 6, No. 4.

[128] WIEGAND, T., SULLIVAN, G. J., BJONTEGAARD, G. & LUTHRA, A. (2003). "Overview of the H.264/AVC video coding standard". IEEE Transactions on Circuits and Systems for Video Technology; Vol.13(7); pp.560-576.

[129] WIEN, M. (2003). "Variable block-size transforms for H.264/AVC". IEEE Transactions on Circuits and Systems for Video Technology; Vol.13(7); pp.604-613.

[130] Chun-Hou Zheng, D.S.Huang, and Li Shang, “Feature selection in independent component subspace for microarray data classification,” Neurocomputing, vol.69, nos.16-18, pp.2407-2410, 2006.

[131] XIAOQUAN, Y. & NAM, L. (2005). "Rapid block-matching motion estimation using modified diamond search algorithm". In Proc. IEEE International Symposium on Circuits and Systems ISCAS; Vol.6; pp.5489-5492.

[132] XIONG, X., SONG, Y. & AKOGLU, A. (2011). "Architecture design of variable block size motion estimation for full and fast search algorithms in H.264/AVC". Computers & Electrical Engineering; Vol.37(3); pp.285-299.

[133] Jun Zhang, D.S.Huang, Tat-Ming Lok, and Michael R. Lyu, “A novel adaptive sequential niche technique for multimodal function optimization,” Neurocomputing, vol.69, nos.16-18, pp.2396-2401, 2006.

[134] Fei Han, D.S.Huang, "Improved extreme learning machine for function approximation by encoding a priori information," Neurocomputing, vol.69, nos.16-18, pp.2369-2373, 2006.

[135] XUAN, J. & LAP-PUI, C. (2004). "An efficient three-step search algorithm for block motion estimation". *IEEE Transactions on Multimedia;* Vol.6(3); pp.435-438.

[136] YI, X., ZHANG, J., LING, N. & SHANG, W. (2005). "Improved and simplified fast motion estimation for JM (JVT-P021)". *Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6) 16th Meeting: Poznan, Poland*.

[137] YU, L. & PENG WANG, J. (2010). "Review of the current and future technologies for video compression". *Journal of Zhejiang University - Science C;* Vol.11(1); pp.1-13.

[138] ZHAO, H., YU, X.-B., SUN, J.-H., SUN, C. & CONG, H.-Z. (2008). "An Enhanced Adaptive Rood Pattern Search Algorithm for Fast Block-Matching Motion Estimation". *Congress on Image and Signal Processing;* Vol.1; pp.416-420.

[139] ZHU, S. & MA, K.-K. (2000). "A new diamond search algorithm for fast block-matching motion estimation". *IEEE Transactions on Image Processing;* Vol.9(2); pp.287-290.

[140] J. Lee, T. Ebrahim, “J. Lee, T. Ebrahim, “IEEE Journal of Selected Topics in Signal Processing”, IEEE Journal of Selected Topics in Signal Processing, vol. 6, No 6, 2012, pp. 684-697

[141] G. Suganya, K., Mahesh, “A Survey: Various Techniques of Video Compression”, International Journal of Engineering Trends and Technology (IJETT) – Volume 7 Number , 2014, pp. 10-12

[142] S. Bachu and K. Chari, “A Review on Motion Estimation in Video Compression”, International Conference on Signal Processing and Communication Engineering Systems, 2015, pp. 250-256.

[143] D. M. Thomas, “A Study on Block Matching Algorithms and Gradient Based Method for Motion Estimation in Video Compression”, 1st International Conference on Digital Image Processing and Pattern Recognition, 2011, pp. 136-145.

[144] M. Chriqui, P. Sinha, “Survey of motion estimation techniques for video compression”, Conference on Low-Light-Level and Real-Time Imaging Systems, Components and Applications, 2002, pp. 218-226.

[145] D. Jha, F. Kannampuzha, J. Joseph, S. Possa, “Motion Estimation Algorithms for Baseline Profile of H.264 Video Codec”, International Journal of Engineering Trends and Technology (IJETT), Vol. 4, No. 4, pp. 727-731, 2013

[146] J. Fabrizio, S. Dubuisson and D. Bereziat, “Motion compensation based on tangent distance prediction for video compression”, Journal of Signal Processing: Image Communication, Vol. 27, No. 2, pp. 153–171, 2012.

[147] J. Cai, D. Pan, “On fast and accurate block-based motion estimation algorithms using particle swarm optimization”, International Journal of Information Sciences, Vol. 197, pp. 53–64, 2012.

[148] T. Aziz, R. J. Dolly, “Motion Estimation and Motion Compensated Video Compression Using DCT And DWT”, International Journal of Emerging Technology and Advanced Engineering, Vol. 2, No. 12, pp. 667-670, 2012.

[149] D. M. Thomas, S. Varier, “A Novel Based Approach for Finding Motion Estimation in Video Compression”, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 1, No. 8, pp. 514-520, 2012.

[150] I. A. Pandian, J. Bala, B. A. Georg, “A Study on Block Matching Algorithms for Motion Estimation”, International Journal on Computer Science and Engineering, Vol. 3, No. 1, pp. 34-41, 2011.