DOI 10.29042/2018-2669-2678

Helix Vol. 8(1): 2669- 2678

Fast Normalized Covariance based Similarity Measure for Fractal Video Compression with Quadtree Partitioning

Yuvaraj.K*[1], B. Deepa[2], S. Geetha Reddy[3]

[1, 2, 3]Department of ECE, SVEW Tirupathi

[*1]***Email:*** *yuvaraj.kunati@gmail.com*

Received: 06th October 2017 Accepted: 14th November 2017, Published: 31st December 2017

Abstract

Fast normalized covariance based similarity measure for fractal video compression with quadtree partitioning is proposed in this paper. To increase the speed of fractal encoding, a simplified expression of covariance between range and overlapped domain blocks within a search window is implemented in frequency domain. All the covariance coefficients are normalized by using standard deviation of overlapped domain blocks and these are efficiently calculated in one computation by using two different approaches, namely, FFT based and sum table based. Results of these two approaches are compared and they are almost equal to each other in all aspects, except the memory requirement. Based on proposed simplified similarity measure, gray level transformation parameters are computationally modified and isometry transformations are performed using rotation/reflection properties of IFFT. Quadtree decompositions are used for the partitions of larger size of range block, that is, 16 × 16, which is based on target level of motion, compensated prediction error. Experimental result shows that proposed method can increase the encoding speed and compression ratio by 66.49% and 9.58%, respectively, as compared to NHEXS method with increase in PSNR by 0.41 dB. Compared to H.264, proposed method can save 20% of compression time with marginal variation in PSNR and compression ratio.

1. Introduction

The emerging multimedia applications such as video conferencing, video over mobile phones, video email, and wireless communications require an effective video coding standard to achieve a low bit rate with good quality. Performance of video coding standard depends on parameters such as quality of reconstructed video, compression ratio, and encoding time. Fractal based video compression [1] is an alternative to accomplish high compression ratio with good quality reconstructed output video than the existing video standards (MPEG, H.263, H.264). Recently, various researchers proposed different algorithms to improve the fractal encoding speed.

Jacquin [2] proposed an innovative technique which is based on the fractal theory of iterated function system for image compression. It reduces an affine redundancy of an image by using its self-similarity properties. Video sequences contain temporal redundancies between consecutive frames that can be easily removed by using fractal based technique. Fractal coding has received attention of researcher due to its advantages of independent resolution, high decoding speed, and high compression ratio [3, 4]. But high encoding time is main drawback of fractal based method; due to this it is not useful for real time applications. The motivation of this method is that it gives high compression ratio with good quality output which is useful for storage and transmission of bulky videos. To increase the encoding speed with keeping motivational parameters, a fast fractal video coder system is proposed.

Cube-based [5, 6] and frame based [7] fractal compression methods are used frequently for video compression. In cube-based compression, the video is divided into groups of frames, each of which in turn is partitioned into three-dimensional (3D) domain and range blocks; however, it has high computing complexity and low compression ratio. In frame based compression, each frame is encoded using the previous frame as a domain pool which introduces and spreads the error between the frames and it can be used to obtain a high compression ratio. Wang proposed a fixed block size hybrid compression algorithm [8] and an adaptive partition instead of fixed-size partition [9], which merges the advantages of cube-based and frame based fractal compression method. Another hybrid coder scheme which combines neighbourhood vector quantization with fractal coding to compress the video as a 3D volume was proposed by Yao and Wilson [10]. Fractal approach for 3D search less [11], prediction of error frame for low bit rate video [12], and wavelet transform based video coding approach [13, 14] are also considered for compression of videos.

Circular prediction mapping (CPM) and noncontractive interframe mapping (NCIM) are proposed by Kim et al. [15], to combine the fractal sequence coder with well-known motion estimation/motion compensation algorithm that exploits the high temporal correlations between the frames. Fractal video coding using a new cross hexagon search (NHEXS) algorithm is proposed [16] for higher motion estimation speed for searching stationary and quasi-stationary blocks. The regions can be defined according to [17, 18] a previously computed segmentation map and are encoded independently using NHEXS based searching technique. A new object-based method [19] is introduced in the transform domain using shape-adaptive DCT for stereo video compression. Zhu et al. proposed

  ISSN 2319 – 5592 (Online)

an automatic region-based video coder [20] with asymmetrical hexagon searching algorithm and deblocking loop filter to improve decompression video quality. High efficiency fractal multiview codec is presented in [21] to encode anchor viewpoint video using intraprediction modes and fractal coder with motion compensation technique. Three-step search algorithm is modified in [22] using two cross search and two cross hexagon search patterns to implement fractal video coder.

Block based motion estimation and motion compensation algorithms exploit the high temporal correlations between the adjacent frames. In frame based fractal video coding, range and domain blocks need to be matched with proper selection of geometrical transformation, scaling, and luminance factors. Normalized covariance, that is, Zero Mean Normalized Cross Correlation (ZNCC) is a method for determining the structural similarity between two blocks from the image [23]. The best matched domain block having high normalized cross correlation [24, 25] value may have large average gray level difference. This difference is reduced to zero or very small value by selecting a proper fractal encoding parameters. But the direct computation of ZNCC for every range block is computationally very expensive. Sum table based method significantly minimizes the computation complexity of ZNCC. It is a precalculated running sum discrete structure of the entire image and acts as a look-up table for the calculation of definite sum according to the size of block.

In this paper, a fast fractal based video coder is proposed using the normalized covariance algorithm as a similarity measure. It uses three levels of quadtree partition for motion estimation, which provides good balance degree of variation to picture content and helps to improve the compression ratio. The complexity of covariance between range and all domain blocks is simplified and implemented in one computation using FFT algorithm. Computational complexity of scaling factor and brightness factor are also minimized with new simple expression based on normalized covariance concept instead of traditional mean square error (MSE). The speed of fractal encoding process is further increased by incorporating a few steps such as FFT based or sum table based method; either one is used to perform the normalization of covariance component, eight isometry transformations' operation using 2D IFFT properties, and the early search termination technique. Performance of video compression using FFT based and sum table based methods is separately verified and they are nearly equal to each other. These techniques can be used to improve the subjective quality of video and coding efficiency.

The rest of the paper is organized as follows. The basic fractal block coding for the image is described in Section 2. Normalized covariance based motion estimation and quadtree partition are explained in Section 3. Fast fractal video coding using FFT is presented in Section 4. The experimental results and comparative study of the proposed algorithm with existing algorithms are presented in Section 5. The conclusion is outlined in Section 6.

2. Fractal Image Coding Theory

Fractal image coding is based on the theory of the partitioned iterated function system (PIFS) [2]. It consists of a set of contractive transformations; when this transformation is applied iteratively to an arbitrary image it will converge to an approximation of the original image. Images are stored as a collection of transformations, which will result in image compression.

The original image of size **M × M** is initially partitioned into non overlapping range blocks (R_i) of each size **N × N (i=1, 2,$\frac{M^2}{N^2}$)**. Similarly, the same image is partitioned into overlapping domain blocks (D_j) of each size 2**N × 2N** as a domain pool with one pixel shift in horizontal and vertical direction ($j = 1,2,\ldots\ldots(M-2N+1)^2$). For each range block, locate the best matching domain block from the domain pool and then apply contractive mapping which minimizes the MSE between range and contractive domain block. A range-domain mapping consists of three operations [3] sequentially on each domain block of size 2**N × 2N**: (1) spatial contraction of the domain block (D_j) by down sampling or averaging the four neighbouring pixels of disjoint group forming a block (D_{cj}) of size **N × N**; (2) taking 8 geometrical transformations of each block which includes 4 rotations with 90 degrees and 4 mirror reflections; (3) for each geometrical transformed block perform contractive affine transformation on the greyscale values and select the parameters which give lowest MSE. The error between range ($h = R_i$) and one of the domain ($g = D_{cj}$) blocks is measured by equation (1) and scaling factor "s" and brightness factor "o" of an affine transformation are calculated by (2) and (3), respectively.

$$MSE = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(s.g(i,j)+0-h(i,j)\right)^2 \text{--------(1)}$$

$$s = \frac{N^2\left(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}h(i,j).g(i,j)\right)-\left(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}h(i,j)\right).\left(\sum_{j=0}^{N-1}\sum_{j=0}^{N-1}g(i,j)\right)}{N^2\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}g^2(i,j)-\left(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}g(i,j)\right)^2} \text{--------(2)}$$

$$0 = \frac{1}{N^2}\left[\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}h(i,j)-s\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}g(i,j)\right] \text{--------------(3)}$$

Where N^2 is the number of pixels in the block and $h(i,j)$,$g(i,j)$ are the pixel values of range block and contractive domain block at coordinates (i,j). For each range block, the parameters which need to be stored as a fractal encoded data are the coordinates of domain block

along with *s* and *o* and geometric transformation index. Gray level transformation parameters "*s*" and "*o*" should be in the range of −1.2 to 1.2 and −255 to 255, respectively [3], to make sure that the transformation is contractive. At the decoder, these fractal parameters are iteratively applied to an arbitrary initial image according to the encoding block size, which will finally converge to a reconstruction of the original image after certain number of iterations.

3. Normalized Covariance Based Motion Estimation

The ZNCC is a recognized similarity measure criterion and is considered as one of the accurate motion estimators in video compression. In fractal video coding, the best matched domain block is decided after applying the proper affine transformation which gives least mean square error. The ZNCC method is more robust under uniform illumination changes and less sensitive to noise; hence it can help to increase the coding efficiency and improve subjective visual quality of output video. For identical regions, it may give a high NCC value for the best match, but with a large average gray level difference. This difference is minimized by selecting the accurate luminance and geometrical transformation parameters. These fractal parameters are interpreted as a kind of motion compensation technique due to unavailability of error frame. In discrete domain, the range block (h_t) of current frame is shifted pixel by pixel across the search window (g_{t-1}) of the reference frame. The estimation of motion relies on the detection of maximum ZNCC function between h_t and g_{t-1}. The ZNCC is defined as

$$c(x,y)=\frac{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(h_t(i,j)-h_{t_0}\right).\left(g_{t-1}(x+i,y+i)-g_{t-1,o(x,y)}\right)}{\sqrt{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(g_{t-1}(x+i,y+i)-g_{t-1,o(x,y)}\right)^2}\sqrt{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(h_t(i,j)-h_{t_0}\right)^2}} \quad 0\le x,y\le 2V \quad \text{--------}$$

In (4)$g_{t-1,o(x,y)}$ denotes the mean value of $g_{t-1,}$within the area of the range block $h_t$shifted to (x, y) position and similarly h_{t0}denote the mean value of the range blockh_t. $C(x,y)$represents the ZNCC surface matrix between the current macro block and reference search window. If the search range is ±V pixels in both directions from the same location of h_t then the size of the $C(x,y)$ will be$(2V+1,2V+1)$. Fractal encoding is itself complex method and use of ZNCC function to estimate motion vector can make the combined technique computationally expensive and time consuming. To overcome these complexity problems, an efficient method of ZNCC calculation has been proposed which is based on new optimal *s* and *o* transformation parameters.

3.1. Quadtree Based Partition

Quadtree decomposition method initially partitions the current frame into a set of larger size (16 × 16 pixels) range blocks h_t at level-1 ($L=1$). Motion vector of the matched block is decided after verifying the highest three peak locations of ZNCC surface matrix. This verification is required because after quantizing gray level transformation parameters, the error between the blocks may vary. If the lowest quantizing error of larger size block (h_t) is above the prespecified threshold, then the block h_t is partitioned in four quadrants $\{h_{t,i}\}\, i=,\dots,4$. The partitioning scheme [26] can be recursively continued until the error becomes smaller than the threshold or 4 × 4 pixels block size ($L=3$) is reached.

Here, 3 levels of quadtree partitioning are employed with block sizes from 16 × 16 to 4 × 4 pixels. All subsequent partitions of 16 × 16 block size are represented by one code word as shown in Figure 1. The length of code word can be either 1 bit or 5 bits; it depends on only level-1 and level-2 nodes. If the level-1 block is not partitioned then the code word is 1 bit; otherwise it is assigned 5 bits. Similarly, if the corresponding block is partitioned, then it is represented by bit 1; otherwise it is represented by bit 0 as shown in Figure 1.

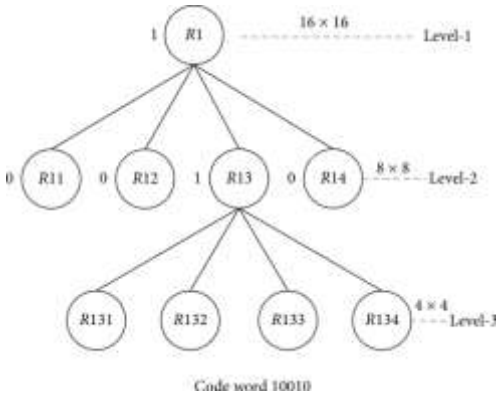


Figure 1: Quadtree structure with code word

4. Fast Fractal Video Coding Algorithm

The most important factor that affects the speed of fractal encoding is the searching of matched domain block. In fractal video compression, searching process is limited to the size of search window but it is also expensive in comparison with standard video coder (H.264/MPEG4). We proposed a FFT based ZNCC method with new matching criteria which reduced the block searching time. Current frame (f_t) is initially partitioned into non overlapped range blocks "$h=h_t$" of size **N × N** pixels. The search window "$g=g_{t-1}$" of size **(N+16) × (N+16)** is defined on the previously decoded frame (f_{t-1}), that is, reference frame of size $M_1 \times M_2$. The flow diagram of the proposed fast fractal video coder is shown in Figure 2(a). The ZNCC matching process may find the wrong matched domain block if the range block is homogeneous. If the variance of any range block is below the smallest predefined threshold, then only mean value of that homogeneous block needs to be encoded; otherwise block belongs to non homogeneous group. The root mean square (RMS) measure is used to find the prediction error (E_m) between

range (h) and matched domain ($g_{x,y}$) block. All the fractal parameters along with corresponding error (E_m) are the output of fast fractal searching algorithm as shown in Figure 2(b). The given block "h" is partitioned into four quadrants when E_m is above the specified threshold (t_{hL}); otherwise encode and save the fractal parameters. In this paper depth of quadtree is three levels; that is, $L_{max} = 3$, so two error thresholds are specified, that is t_{h1}, andt_{h1}.

Due to the high temporal correlation in video sequences, range-domain mapping becomes more effective if the sizes of range and domain block are the same [15]. The quality of reference frame plays an important role while mapping two same size blocks. If interframe motion vector prediction is based on good quality reference frame (previous frame), then the prediction error will be low. But for subsequent frames this error monotonically increases due to cumulative process. So to get a better quality reference frame at the beginning, intraframe is compressed using DCT transformation and quantization technique [27, 28].

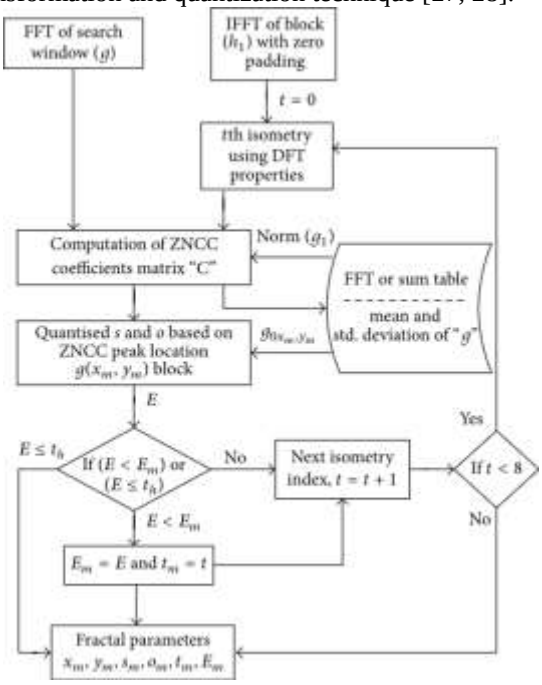


Figure 2: Flow diagrams of (a) fast fractal video coder and (b) efficient searching algorithm.

4.1. Efficient Searching with Simplified Similarity Measure

The high computational complexity is the main drawback of ZNCC similarity measure method in spatial domain. Therefore, (4) is simplified to minimize the complexity. Numerator component of (4) is represented as $n(x,y)$ and is rewritten as follows:

$$n(x,y) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g(x+i,y+i).h_1(i,j) - g_{0x,y}\sum_{i=0}^{N-1}\sum_{j-0}^{N-1} h_1(i,j) \text{----------}$$

where $h_1(i,j) = (h(i,j) - h_0)$; it has zero mean and because the sum of $h_1(i,j)$ is zero, the term $g_{0x,y}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} h_1(i,j)$ will also be zero and (5) can be written as

$$n(x,y) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g(x+i,y+i).h_1(i,j) =< g_{x,y}, h_1 > \text{----------} (6)$$

Equation (6) is independent of the mean ($g_{0x,y}$) value of domain block. All the domain blocks are overlapped and $g_{0x,y}$ value of each block is different. $n(x,y)$ is the cross correlation of h_1 and $g_{x,y}$ block, where and are the location of domain block $g_{x,y}$. It is equivalent to the inner product operation of two bocks represented by $< .,. >$ operator. This cross correlation is equal to the complex conjugate multiplications of two frequency domain components. The complex conjugate part can be avoided by taking the inverse FFT (IFFT) of one of the input instead of FFT because $conj\ (FFT(h_1) = (N+16)^2 \times IFFT(h_1)$, where $(N+16)^2$ is a constant term. In frequency domain, size of both the input blocks must be equal. The size of block h_1 is increased to the size of g by padding zeros on the left and down the side of the block. The calculation of (6) for all the blocks can be computed in one computation using FFT as given below.

$$n = IFFT[FFT(g).IFFT(h_1)] = IFFT[G.H_1] \text{----------} (7)$$

Similarly, the denominator component of (4) is the product of standard deviation of $g_{x,y}$ and h blocks. Due to cancellation of ($\frac{1}{N^2}$) factor of standard deviation with numerator term, it turns into L_2 norm as (8) and (9). The norm of $g_{1x,y}$ (8) is also expensive because it is repeated for each overlapped domain block, whereas the norm of h_1 (9) is unique for all domain blocks.

$$\|g_{1x,y}\| = \left[\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g(x+i,y+j) - g_{0x,y})^2\right]^{1/2} \quad where\ \ g_{1x,y} = g_{x,y} - g_{0x,y} \text{-----} (8)$$

$$\|h_1\| = \left[\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} (h(i,j) - h_0)^2\right]^{1/2} \quad where\ \ h_1 = h - h_0 \text{------} (9)$$

The norm of expression is simplified and is written as (10) in the following:

$$\|g_{1x,y}\| = \left[\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g^2(x+i,y+j) - 2g_{0x,y}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g(x+i,y+j) + \sum_{i=0}^{N-1} g_{0x,y}^2\right] \text{--------} (10)$$

$$\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g_{0x,y}^2 = N^2\left(\frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g(x+i,y+j)\right)^2 \text{------} (11)$$

Equation (10) can be simplified using (11) to

$$\|g_{1x,y}\| = \left[\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g^2(x+i,y+j) - \frac{1}{N^2}\left(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g(x+i,y+j)\right)^2\right]^{1/2} \text{------} (12)$$

This can be represented in terms of mean value of domain block as

$$\|g_{1x,y}\| = \left[\left(\sum_{i=0}^{N-1}\sum_{j=0}^{N-1} g^2(x+i,y+j)\right) - N^2 g_{0x,y}^2\right]^{1/2} \text{----------} (13)$$

All the overlapped domain blocks which are shifted by one pixel also require similar computations and mostly these will be repeated from the previous blocks. Two different methods are proposed to prevent unnecessary computations of the mean and norm of all overlapped blocks, that is, FFT based and sum table based. The ZNCC (4) can be rewritten using (7), (8), (9), and (13) to (14).

$$c(x,y)=\frac{n(x,y)}{\left\|g_{1x,y}\right\|.\left\|h_1\right\|}--------(14)$$

The flow diagram of fast efficient searching algorithm with fast computation of ZNCC (14) is shown in Figure 2(b). In video sequences, more number of the blocks can be observed as stationary blocks with zero/nonzero motions. Such blocks can be identified when the highest value on ZNCC surface is nearly equal to one ($C(x_m,y_m) \approx 1$) and corresponding quantized value of *s* and *o* is one and zero, respectively. After finding the motion vector (x_m,y_m) of the stationary domain block, searching process can be terminated if the RMS error (E) is below the threshold (t_h) with isometry index$t=0$. Due to early termination of searching process coding efficiency is increased with unchanged quality. The norm of h_1 can be neglected from (14) because it is constant and divisional term for all domain blocks and it does not change the final ZNCC result.

4.1.1. FFT Based Method for Mean and Norm Calculation of Overlapped Blocks

The sum of squared pixels and the sum of pixels according to the size of range block are similar to the convolution with unit (*I*) matrix of same size. FFT and IFFT combinations are used for fast computation. The flow diagram of the mean and norm calculation of all the domain blocks in one computation as matrices using FFT is shown in Figure 3. FFT of input g is readily available from (7) and the IFFT of unit (*I*) matrix with zero padding is constant only one time computation. So, only FFT of squared pixels and last two IFFT are required along with others blocks. The mean (g_0) and norm (g_1) of all the blocks are required during gray level transformation and normalized covariance calculations.

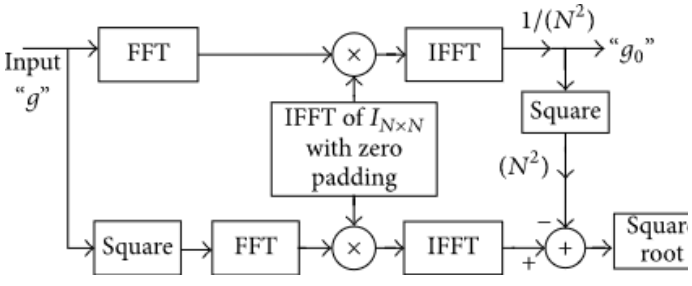


Figure 3: FFT based mean and norm calculation of all domain blocks.

4.1.2. Sum Table Based Method for Mean and Norm Calculation of Overlapped Blocks

The sum table based method also can be used to reduce the number of computations required to compute the mean and norm of all domain blocks. The pre computation of two sum tables $st(k,l)$ and $st^2(k,l)$ over the previous frame $f_{t-1}(k,l)$ and squared ${f_{t-1}}^2(k,l)$ pixel frame, respectively, acts as look-up tables. These tables are recursively constructed for each frame before the beginning of encoding process, defined by

$$st(k.l)=f_{t-1}(k,l)+st(k-1,l)+st(k,l-1)-st(k-1,l-1)$$

$$st^2(k,l)=f_{t-1}^2(k,l)+st^2(k-1,l)+st^2(k,l-1)-st^2(k-1,l-1)--------(15)$$

Where k and l are the pixel coordinates of frame f_{t-1} with $k=0,1,\ldots.,M_1-1$ and l= $0,1,\ldots.,M_2-1$. So initial condition $st(k,l)=st^2(k,l)=0$ when either k or l $=-1$. These two sum tables are partitioned according to the size of search window to s_1 and s_1^2 subtables of size $(N+17\times N+17)$. It consists of one additional row and column at the initial position in comparison with search window size. The sum expressions in (12) over $g(x+i,y+j)$ and $g^2(x+i,y+j)$ can be calculated efficiently using sub tables.

$$\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}g(x+i,y+j)=s_1(x+N-1,y+N-1)+s_1(x-1,y-1)-s_1(x-1,y+N-1)-s_1(x+N-1,y-1)---(16)$$

$$\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}g^2(x+i,y+j)=s_1^2(x+N-1,y+N-1)+s_1^2(x-1,y-1)-s_1^2(x-1,y+N-1)-s_1^2(x+N-1,y-1)------(17)$$

Using (16) and (17) the norm (12) and the mean (18) of all domain blocks can be calculated as matrices of size 17 × 17, $x \in \{0,1\ldots.,16\}$ and $y \in \{0,1\ldots.,16\}$.

$$g_{0x,y}=\frac{1}{N^2}(s_1(x+N-1,y+N-1)+s_1(x-1,y-1)-s_1(x-1,y+N-1)-s_1(x+N-1,y-1))----(18)$$

According to the three levels' quadtree partition, it may require total six memories to store these tables of each size$M_1\times M_2$. But instead of six only two memory spaces are defined according to the size of level three blocks and based on that remaining two sub tables are calculated.

4.2. New Gray Level and Isometry Transformation

The gray level transformation parameters such as scaling factor "s" and brightness factor "*o*" for each range block h is based on certain domain block $g_{x,y}$ which strongly matches (19) when MSE is zero.

$$h=s.g_{x,y}+0---------(19)$$

Using (8) and (9), (19) can be represented as

$$h_1+h_0=s(g_{1x,y}+g_{0x,y})+0----------(20)$$

So $h_1=s.g_{1x,y}$ and $h_0=s.g_{1x,y}+o$.

When the highest peak of ZNCC is close to one that is, $C(x,y)\approx 1$ as per (14), then it denote the coordinates of matched domain block. The gray level parameters (s and o) are estimated by substituting

$g_{x,y} = {}^{h_1}/_{S}$ in (14), which are computationally more efficient form as given below

$$s = \frac{\|h_1\|^2}{n(x,y)} \text{----------} (21)$$

$$0 = h_0 - s.g_{0x,y} \text{----------} (22)$$

As per (21) both numerator and denominator terms are available in (14) except square operation and in (22) h_0 and $g_{0x,y}$ are also available while finding $\|h_1\|$ and $\|g_{1x,y}\|$. So the computational complexity of these both the equations are minimal.

The searching algorithm has the difficulty of applying 8 different isometry transformations to the individual domain block since it operates on the entire search window using 2D FFT. Hence the isometry transformations are performed on the range block instead of domain block. The initial IFFT of zero padded range block acts as a IFFT of zero degree transformation block. The remaining seven isometry transformations along with their IFFTs are calculated based on the previous IFFT of range block, by applying rotation and reflection properties of 2D IFFT [26]. This helps to avoid the repeated IFFT calculations; due to this it increases the searching speed of the algorithm. In fractal based fast motion estimation using simplified similarity measure with quadtree partition, operating in frequency domain is one of the features of this paper.

5. Experimental Results

The performance of the proposed fast normalized covariance based fractal video coder with simplified similarity measure is evaluated. The denominator of (14) is implemented by using two different methods; one FFT based method is represented as proposed-FFT and other sum table based method is represented as proposed-ST. The popular video sequences (352 × 288 pixels of each sequence) [29], foreman, carphone, Tennis, news, and coastguard, are used to evaluate the performance of proposed methods. Range blocks are formed according to three-level quadtree partition criterion with block size 16 × 16 pixels at level 1, 8 × 8 pixels at level 2, and 4 × 4 pixels at level 3. The smallest predefined threshold to detect a homogeneous block at each level is (0.25×N) and the remaining two thresholds are t_{h1}= 6.4±0.5 and t_{h2}= 8.4±0.6 to obtain good quality output video. A search area on the reference frame is ±8 pixels in both vertical and horizontal directions from the same position as of range block on the target frame. Along with proposed methods, CPM/NCIM and NHEXS algorithms are also implemented. In CPM/NCIM method first 3 frames are set for CPM and the remaining frames are using NCIM. The video sequences are also compressed using H.264, JM 18.6 reference software [30] to compare the performances. The parameters of H.264 coder are defined as high profile, quantization parameter OPP: between 28 and 36 selected to ensure good quality, search range: 16, macroblock partitioned: 4 × 4, 8 × 8, and 16 × 16, group of pictures (GOP): 12 or 15, and entropy based coding method: universal variable length coding (UVLC). All the methods including proposed methods are implemented in MATLAB 7.14 and simulated on a PC (Intel Core i5-2400 CPU, 3.10 GHz, 3.16 GB RAM).Fractal parameters of each range block are quantized separately: gray scale factors s and o are quantized by assigning 5 bits and 7 bits, respectively; coordinates of matched domain block (x,y) are encoded with 4-bit length code words and 3 bits for the indexing of isometry transformations. In comparison with all the presented methods, only sum table based method requires two additional memories of each size of $(M_1 \times M_2)^2$ bytes.

Table 1: Comparison of average video coding results using different methods.

Videos	Methods	PSNR (dB)	Time (sec.)	CR
Foreman	CPM/NCIM	30.63	42.81	48.41
	NHEXS	34.19	2.46	88.50
	H.264	34.33	1.20	97.00
	Proposed-ST	34.25	0.87	95.11
	Proposed-FFT	34.25	0.88	94.43
Carphone	CPM/NCIM	31.32	35.90	52.42
	NHEXS	34.97	1.95	108.31
	H.264	35.12	1.04	120.80
	Proposed-ST	35.00	0.68	119.24
	Proposed-FFT	35.00	0.69	118.93
Tennis	CPM/NCIM	29.22	66.58	28.28
	NHEXS	31.27	4.91	53.81
	H.264	31.72	1.49	63.66
	Proposed-ST	31.50	1.39	61.69
	Proposed-FFT	31.53	1.42	61.58
News	CPM/NCIM	29.31	47.99	55.70
	NHEXS	37.07	1.54	122.21

	H.264	38.12	0.90	105.84
	Proposed-ST	38.41	0.66	126.31
	Proposed-FFT	38.40	0.66	126.82
Coastguard	CPM/NCIM	28.21	90.36	18.80
	NHEXS	29.85	6.51	45.29
	H.264	30.30	1.64	52.44
	Proposed-ST	30.13	1.58	52.71
	Proposed-FFT	30.12	1.60	53.51

Table 2: Results comparison between proposed methods with other methods.

(a)

Video	Methods	GOP	PSNR (dB)	Compression ratio
Highway	Object-based [18]	15	35.53	81.18
	Region-based [17]		35.38	82.76
	Proposed-ST		37.30	288.48
	Proposed-FFT		37.27	287.29
	H.264		37.05	202.14

(b)

Videos	Methods	GOP	PSNR (dB)	Bit rate (KBPS)
Silent	Region-based [20]	20	32.3	398.33
	Proposed-ST		37.06	330.62
	Proposed-FFT		37.05	331.76
	H.264		37.01	390.48
Mother-daughter	Region-based [20]	20	37.05	314
	Proposed-ST		40.50	161.72
	Proposed-FFT		40.48	161.80
	H.264		40.04	187.87

To evaluate the performance, the results of proposed algorithms are compared with traditional CPM/NCIM and NHEXS algorithms. All these algorithms are also compared with H.264. BY keeping the fixed value of PSNR for each video sequence, encoding time and compression ratio (CR) are calculated using all the methods. Table 1 shows the comparison of average coding results of five video sequences. In each sequence, PSNR and CR of proposed methods are the references for other methods analysis. The average coding time of the proposed method is decreased by 98.17% and 66.49% of the CPM/NCIM and NHEXS methods, respectively. In comparison with proposed method, the average compression ratio of CPM/NCIM and NHEXS is decreased by 55.75% and 9.58%, with 4.12 dB and 0.41 dB reduction in PSNR. The proposed methods present an average of 20% reduction in coding time and 2% decrease in compression ratio with marginal degradation of PSNR by 0.15 dB if compared to the H.264.

Table 2 shows the comparison of performances of the proposed methods with existing fractal video coder methods [17, 18, 20]. In each existing method, the different video sequences with different GOPs are used for analysis. The sequence highway (15 frames, 352 × 288 pixels) is used and the compression ratio is 3.55 times with PSNR 1.72 dB higher than average of [17, 18]. The silent and mother-daughter (20 frames each, 352 × 288 pixels) sequences are used and their average PSNR increased by 4.11 dB and bit rate decreased by 33% in comparison with algorithm in [20]. For low bit rate videos, these proposed methods give much better performance than H.264. In Table 2 the proposed method can save the compression time by 55% with marginal reduction of PSNR in comparison with H.264. The performance of proposed-ST and proposed-FFT based methods are almost equal in terms of PSNR, encoding time, and compression ratio as shown in Tables 1 and 2. For low bit rate videos, the proposed methods give high compression ratio and very less time in comparison with H.264. News, highway, mother-daughter, and silent are the low bit rate videos and others are high bit rate videos.

Statistical measures were used to compute the score distribution range of result parameters. Confidence interval (CI) generates an upper and lower limit for the mean; it gives an indication of how much uncertainty there

is with true mean. The -distributions for each of the test parameters with sample size , sample mean , sample standard deviation sd, and desired significance level are used to define the confidence limits as follows:

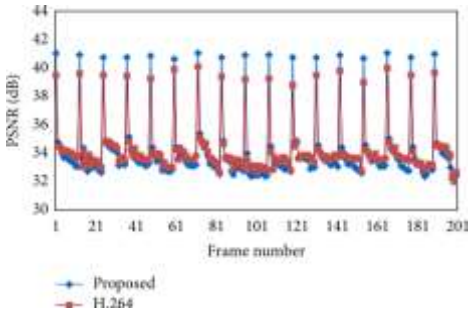
$$CI = \mu \pm t_{(1-\alpha/2,R-1)} \cdot \frac{s_d}{\sqrt{R}} \text{-----(23)}$$

The confidence interval can be computed for different value of and () degrees of freedom. In the result analysis, the 90% confidence interval is calculated for each test parameter with the value of in (23) being 1.667. Table 3 shows a 90% CI width for a mean of PSNR, encoding time, and compression ratio for various video sequences. Due to fixed value of PSNR, the average half-width CI of PSNR is almost equal for both the methods. The 90% interval width of encoding time and compression ratio is narrow; it means proposed method also gives higher accuracy as compared to standard video coder with less encoding time.

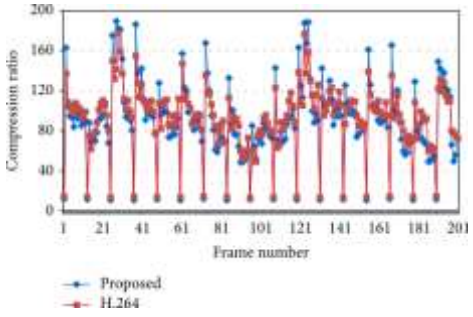
Figure 4 shows a frame-wise performance comparison for 204 frames of foreman video sequence between the proposed method and H.264. Due to cumulative error, the PSNR of decoded frames slightly decreases as the frame number increases from every intraframe in proposed method. This error is minimized by using proper selection of gray level transformation as shown in Figure 4(a). The results show that the compression ratio and PSNR of the proposed method for each frame are marginal changes proportional to the H.264 results. The compression time of the proposed method as shown in Figure 4(c) for each frame is, on average, 0.6 sec. (27%) lesser than H.264. High encoding time drawback has been overcome by using proposed fast fractal video coder method. In addition to this it gives good quality output and high compression ratio approximately equal to standard (JM v18.6) video coder as shown in Table 1. Human visual system (HVS) does not perceive the smallest change in PSNR (≤0.8 dB) between the H.264 and proposed method. Figure 5 shows the 62nd original and decoded frame of "Tennis" sequence using H.264 with 32.26 dB and proposed method with 32.12 dB.

Table 3: Comparison of 90% confidence interval width for mean of result test parameters.

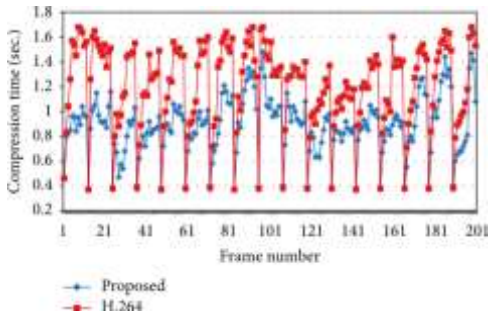
Videos	PSNR (dB)		Time (Sec)		Compression ratio	
	H.264	Proposed	H.264	Proposed	H.264	Proposed
Foreman	33.98–34.68	33.84–34.25	1.13–1.27	0.83–0.91	90.49–103.51	86.42–103.8
Carphone	34.75–35.49	34.61–35.39	0.97–1.11	0.64–0.72	112.71–128.89	108.7–129.78
Tennis	31.24–32.20	30.97–32.09	1.38–1.60	1.29–1.49	58.05–69.35	54.50–68.88
News	37.71–38.53	38.12–38.70	0.83–0.97	0.62–0.70	95.61–116.07	112.17–141.48
Mother-daughter	38.38–39.30	38.74–39.42	0.80–0.96	0.59–0.63	212.86–230.44	213.82–238.1
Highway	35.90–36.66	35.66–36.52	1.17–1.29	0.41–0.47	176.44–196.08	195.18–215.12
Coastguard	29.79–30.81	29.50–30.76	1.51–1.78	1.45–1.71	41.76–63.12	40.45–64.97
Silent	36.24–37.56	36.34–37.78	0.62–0.70	0.48–0.52	175.42–198.58	178.61–203.39



(a)



(b)



(c)

Figure 4: Frame to frame performance comparison using proposed method and H.264

Figure 5: Original and decoded 62nd frame of "Tennis" sequence

6. Conclusion

In this paper, a quadtree partition based fast normalized covariance for fractal video compression is presented. A simplified normalized covariance for similarity measure, eight isometry transformations using IFFT properties, and modified new gray level transformation parameters are proposed and estimated using FFT to improve the encoding speed and output quality. Meanwhile, this method can use FFT based or sum table based approaches to normalize the covariance matrix, which further increases the encoding speed significantly. They are used for the calculation of mean and standard deviation of all overlapped blocks in one computation. The results of using these approaches are almost equal in all perspective. The main drawback of sum table based method is that it required large memory space to store the tables as compared to the FFT based method. Quadtree partition helps to achieve high compression ratios with good quality output. The proposed methods can save the encoding time by 98.17% and 66.49%, compression ratio is increased by 129% and 9.58%, and the output quality increased by 4.12 dB and 0.41 dB in comparison with CPM/NCIM and NHEXS methods, respectively. In comparison to H.264, this method saves 20% of compression time with marginal degradation in frame quality and compression ratio.

References

1. M. S. Lazar and L. T. Bruton, "Fractal block coding of digital video," IEEE Transactions on Circuits and Systems for Video Technology, vol. 4, no. 3, pp. 297–308, 1994. View at Publisher · View at Google Scholar · View at Scopus
2. A. E. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations," IEEE Transactions of Image Processing, vol. 1, no. 1, pp. 18–30, 1992. View at Publisher · View at Google Scholar · View at Scopus
3. Y. Fisher, Fractal Image Compression: Theory and Application, Springer, New York, NY, USA, 1995. View at Publisher · View at Google Scholar · View at MathSciNet
4. Y. Zheng, G. Liu, and X. Niu, "An improved fractal image compression approach by using iterated function system and genetic algorithm," Computers & Mathematics with Applications, vol. 51, no. 11, pp. 1727–1740, 2006. View at Publisher · View at Google Scholar · View at Scopus
5. K. U. Barthel and T. Voye, "Three-dimensional fractal video coding," in Proceedings of the IEEE International Conference on Image Processing, vol. 3, pp. 260–263, IEEE, Washington, DC, USA, 1995. View at Publisher · View at Google Scholar
6. C.-C. Wang and C.-H. Hsieh, "Efficient fractal video coding algorithm using intercube correlation search," Optical Engineering, vol. 39, no. 8, pp. 2058–2064, 2000. View at Publisher · View at Google Scholar · View at Scopus
7. M. Wang and C.-H. Lai, "A hybrid fractal video compression method," Computers and Mathematics with Applications, vol. 50, no. 3-4, pp. 611–621, 2005. View at Publisher · View at Google Scholar · View at MathSciNet · View at Scopus
8. M. Wang, R. Liu, and C.-H. Lai, "Adaptive partition and hybrid method in fractal video compression," Computers & Mathematics with Applications, vol. 51, no. 11, pp. 1715–1726, 2006. View at Publisher · View at Google Scholar · View at MathSciNet · View at Scopus
9. M. Wang and C.-H. Lai, "Grey video compression methods using fractals," International Journal of Computer Mathematics, vol. 84, no. 11, pp. 1567–1590, 2007. View at Publisher · View at Google Scholar · View at MathSciNet · View at Scopus
10. Z. Yao and R. Wilson, "Hybrid 3D fractal coding with neighbourhood vector quantisation," EURASIP Journal on Applied Signal Processing, vol. 16, pp. 2571–2579, 2004. View at Publisher · View at Google Scholar
11. D. V. Lima, W. R. Schwartz, and H. Pedrini, "3D searchless fractal video encoding at low bit rates," Journal of Mathematical Imaging and Vision, vol. 45, no. 3, pp. 239–250, 2013. View at Publisher · View at Google Scholar · View at Scopus

12. Y. Brijmohan and S. H. Mneney, “Low bit-rate video coding using fractal compression of wavelet subtrees,” in Proceedings of the 7th IEEE AFRICON Conference in Africa: Technology Innovation, pp. 39–44, September 2004. View at Scopus
13. Y. Zhang, L. M. Po, and Y. L. Yu, “Wavelet transform based variable tree size fractal video coding,” in Proceedings of the IEEE International Conference on Image Processing, pp. 294–297, IEEE, Santa Barbara, Calif, USA, 1997. View at Publisher · View at Google Scholar
14. R. Yu, J. Zhou, S. Yu, and D. Chi, “Fractal-based wavelet transform coding for low-bit-rate video,” in Electronic Imaging and Multimedia Systems, vol. 2898 of Proceedings of SPIE, pp. 226–237, Beijing, China, November 1996.
15. C.-S. Kim, R.-C. Kim, and S.-U. Lee, “Fractal coding of video sequence using circular prediction mapping and noncontractive interframe mapping,” IEEE Transactions on Image Processing, vol. 7, no. 4, pp. 601–605, 1998. View at Publisher · View at Google Scholar · View at Scopus
16. K. Belloulata, S. Zhu, and Z. Wang, “A fast fractal video coding algorithm using cross-hexagon search for block motion estimation,” ISRN Signal Processing, vol. 2011, Article ID 386128, 10 pages, 2011. View at Publisher · View at Google Scholar
17. S. Zhu, Y. Hou, Z. Wang, and K. Belloulata, “Fractal video sequences coding with region-based functionality,” Applied Mathematical Modelling. Simulation and Computation for Engineering and Environmental Systems, vol. 36, no. 11, pp. 5633–5641, 2012. View at Publisher · View at Google Scholar · View at MathSciNet · View at Scopus
18. S. Zhu, L. Li, and Z. Wang, “A novel fractal monocular and stereo video codec with object-based functionality,” Eurasip Journal on Advances in Signal Processing, vol. 2012, article 227, 2012. View at Publisher · View at Google Scholar · View at Scopus
19. K. Belloulata, A. Belalia, and S. Zhu, “Object-based stereo video compression using fractals and shape-adaptive DCT,” AEU—International Journal of Electronics and Communications, vol. 68, no. 7, pp. 687–697, 2014. View at Publisher · View at Google Scholar · View at Scopus
20. S. Zhu, L. Li, J. Chen, and K. Belloulata, “An automatic region-based video sequence codec based on fractal compression,” International Journal of Electronics and Communications, vol. 68, no. 8, pp. 795–805, 2014. View at Publisher · View at Google Scholar · View at Scopus
21. S. Zhu, D. Zhao, and L. Zhang, “A novel high efficiency fractal multiview video codec,” Mathematical Problems in Engineering, vol. 2015, Article ID 613714, 12 pages, 2015. View at Publisher · View at Google Scholar · View at Scopus
22. S. D. Kamble, N. V. Thakur, L. G. Malik, and P. R. Bajaj, “Fractal video coding using modified three step search algorithm for block matching motion estimation,” Advances in Intelligent Systems and Computing, vol. 332, pp. 151–162, 2015. View at Publisher · View at Google Scholar
23. A. J. H. Hii, C. E. Hann, J. G. Chase, and E. E. W. Van Houten, “Fast normalized cross correlation for motion tracking using basis functions,” Computer Methods and Programs in Biomedicine, vol. 82, no. 2, pp. 144–156, 2006. View at Publisher · View at Google Scholar · View at Scopus
24. S. B. Dhok, R. B. Deshmukh, and A. G. Keskar, “Efficient fractal image coding using fast fourier transform,” International Journal on Computing, vol. 1, no. 2, 2011. View at Google Scholar
25. R. E. Chaudhari and S. B. Dhok, “Acceleration of fractal video compression using FFT,” in Proceedings of the 15th International Conference on Advanced Computing Technologies (ICACT '13), pp. 1–4, September 2013. View at Publisher · View at Google Scholar · View at Scopus
26. G. J. Sullivan and R. L. Baker, “Efficient quadtree coding of images and video,” IEEE Transactions on Image Processing, vol. 3, no. 3, pp. 327–331, 1994. View at Publisher · View at Google Scholar · View at Scopus
27. A. K. Jain, Fundamentals of Digital Image Processing, PHI Publications, 1989.
28. Y.-M. Zhou, C. Zhang, and Z.-K. Zhang, “An efficient fractal image coding algorithm using unified feature and DCT,” Chaos, Solitons & Fractals, vol. 39, no. 4, pp. 1823–1830, 2009. View at Publisher · View at Google Scholar · View at Scopus
29. CIPR Video Sequences, http://www.cipr.rpi.edu/resource/sequences/.
30. H.264/AVC Software Coordination, http://iphome.hhi.de/suehring/tml/.

  ISSN 2319 – 5592 (Online)