Compression of an AVI Video File Using Fractal System

Nevart A. Minas , Faten H. MohammedSediq
Computer Systems, Technical Institute,
Kirkuk, Iraq

Abstract

Compression of video files has become a necessity and very important because of transmission and storage of uncompressed video would be extremely costly and impractical.In this paper, the theory of fractal image compression is used to compress an Audio/ Video Interleaved file (AVI) Video file. The problem that the typical fractal compression approaches always discussed, is the long encoding time. A fast fractal image compression (FFIC) approach is used based on centralized moment descriptors to encode the video file images with low encoding time and high quality.

The first stage is opening the AVIvideo file and split its signal into separated images (frames) of type bitmap (BMP). In the second stage the digital image or frame of digital video is transformed from Red-Green-Blue(RGB) color space to the YC_bC_r color space image which is based on luminance (L) and chrominance (C) values.In third stage, each of the YC_bC_r components (y, C_b, and C_r) is compressed alone using FFIC and the fractal codes are savedgenerating a highly scalable layered bit stream that can be decoded at different qualities in terms of spatial resolution. The decoded images can be displayed at arbitrary resolution, with a high compression ratio. Furthermore, the algorithm is designed to require only a minimal coding delay. In the decoding stage all used compression techniques are executed in inverse sequence to obtain the decompressed image of video file and then reconstruct the AVI video file.

Keywords: *Video Compression, Video Coding, Fractal Image Compression, IFS, Centralized Moment.*

1. Introduction

Digital video has become very important form of information technology and is now used in many different areas, such as board casting, teleconferencing, mobile telephone, surveillance, and entertainment. People now expect to be able to access video through a wide range of different devices and over various networks.

Compression refers to the process of reducing the number of bits required to represent the image and video comes in two forms lossless and lossy. The lossless compression is a process to reduce image or video data for storage and transmission while retaining the quality of original image (i.e. the decoded image quality is required to be identical to image quality prior encoding. In lossy compression, on the other hand, some information present in the original image or video is discarded so that the original raw representation of image or video can only be approximately reconstructed from the compressed representation with high compression ratio. For more compression can be achieved with approximate quality to source image lossy compression is almost usually used than lossless compression to compress digital video. In general, in video compression the video sequences contain significant amount of statistical and subjective redundancy within and between frames. The ultimate goal of video source coding is the bit rate reduction for storage and transmission by exploring both spatial and temporal redundancy and to encode “minimum set” of information using entropy coding techniques .This is usually results in compression of coded video data compared to original source data. These statistical and subjective redundancy that can be exploited for compression are called spatial and temporal redundancy.[1]

The main idea of video compression is to exploit redundancies that are present in the video. There are two compression standards present that have been developed for low bit-rate applications. These are the H.263 [2] and MPEG-4 [3] video compression standards. Both standards utilizes wavelet transform based techniques. Wavelet transform techniques combine both transform and sub band coding. The other compression technique that has the potential for high compression is the fractal coding technique. Fractal compression differs from the standard transform coder methods. They were created as a result of the study of iterated function systems (IFS). Fractal methods store images as contraction maps of which the image is the fixed point. The decoding procedure in which the image is recovered by iterating the maps to its fixed point is simple. However, the recovered image suffers from the tiling effect, especially at low bit-rates[4].

Fractal Image Compression (FIC) has been an area of intensive research. Despite the diverse advantages offered by fractal compression, such as high decompression speed, high bit rate and resolution independence, the greatest disadvantage is the high computational cost of the coding phase, which means fractal coding cannot compete with other techniques (wavelets, etc.) in [5, 6]. During more than two decades of the development in FIC, a lot of research has been done to improve the performance. Generally, the attempts to speed up the fractal encoding consist of modifying the following aspects: the composition of the domain pool, the type of search used in block matching, or the representation/quantization of the transform parameters. Apart from these attempts focusing on the coding speed, some hybrid coders, such as wavelet-based and DCT-based fractal encoders, have been developed in [7, 8]. Various FIC side applications now are also explored in many other fields such as image database indexing and even face recognition [9].

2. Digital Video Coding

A digital video sequence is a collection of pictures, also called frames, spaced at fixed time intervals. In a color video sequence, each frame consists of three components, which can be either red-green-blue (RGB primaries) or luminance and two chrominance (YC_bC_r format) components. The luminance (Y) component is a monochrome image containing the structural information of the frame. The two chrominance (C_b and C_r) components contain color hue and saturation information of the frame. RGB format is used in displaying. RGB color space can be converted into YC_bC_r format as in equation (1).

$$\begin{pmatrix} Y \\ C_b \\ C_r \end{pmatrix} = (A) \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 16 \\ 128 \\ 128 \end{pmatrix} \quad (1)$$

$$\text{where }, A = \begin{pmatrix} 0.257 & 0.504 & 0.098 \\ -0.148 & -0.291 & 0.347 \\ 0.439 & -0.368 & -0.071 \end{pmatrix}$$

Three types of chrominance sampling formats relative to the luminance are used in video coding. These are labeled as 4:4:4, 4:2:2 and 4:2:0. In 4:4:4 formats; the same sampling grid for all three components is used. In 4:2:2 formats; the chrominance is sampled 2:1 horizontally but not vertically. The 4:2:0 formats have the chrominance sampled 2:1 both horizontally and vertically. Each component of a frame is a two-dimensional (2-D) signal, which can be represented by a matrix. The elements of the frame matrix are called as pixels. Therefore, a video sequence can be considered as a three-dimensional (3-D) signal for each spectral (color) component.

Digital video coding or compression is concerned with reducing the number of data storing units (bps) used to represent given information content in a video sequence. In addition to inter pixel spatial redundancy within a frame, video sequences contain high temporal (inter frame) redundancy, which is usually exploited in video coding algorithms by coding some frames using motion compensated prediction with reference to previously coded frames [10].

3. AVI File Format

The Audio Video Interleave (AVI) file is a file that is essentially a multimedia container format that was first pioneered by Microsoft to show video content for the Windows Operating System. This file allows both video and audio to play together, as well as making possible the concept of multiple streaming of video with sound. In general, AVI files tend to use file format extensions that were created in February of 1996 by Matrox Open DML group. These extensions, also commonly known by the name AVI 2.0, are completely supported by Microsoft.

The AVI file format is based on the RIFF (Resource Interchange File Format) which works by dividing all the raw data in one file into what are known as 'chunks' of data, where each chunk has its own 4 character tag for identification purposes. In any RIFF formatted file, an AVI file is just one chunk of the total data. This AVI file (one chunk) is further sub-divided into two obligatory chunks and one elective chunk of data.

Here's how the AVI file (chunk) works: The 1st sub-chunk holds integral information such as the metadata regarding the video and its characteristics: frame rate, height and width. This chunk is known as the file header, who identification is the four-character string hdrl. The 2nd sub-chunk, identification stream movi, is the one that consists of the actual data that comprises of the movie. The third sub-chunk, which is optional, is used to index all the chunks within the AVI file [11].

4. Fractal Block Coder

Fractal compression techniques are based on IFS, in which the image is described by sets of equations that provide contractive mappings. The first effective fractal coder was introduced by Jacquin [4], who noticed that a part of an image is similar to another part of the image. This coder makes use of this local self-similarity for compression, by forming images from properly transformed copies of parts of itself. This system requires the image to be divided into smaller, non-overlapping blocks called range blocks. Thereafter, larger domain blocks are constructed from the same image. The idea of this coder is to find an affine transformation of a domain block that closely matches each range block.

Two issues need to be considered in the design of fractal based compression systems. The first is the size and shape of range and domain blocks. Since range blocks are the attractors, the types of affine transformations become limited, since it depends on the size and shape of the range block. The simplest partitioning scheme is to divide the image into non-overlapping square range blocks of a fixed size. The domain blocks are normally twice the size of the range blocks and can overlap. Results reveal that small block sizes yields small compression ratios and a large block size, a large compression ratio. However, the clarity of the recovered image suffers with large block sizes. The downfall of the fixed size partition is that the image is partitioned without considering the contents of the image. There are regions of the image that will be covered well using small range block sizes and similarly, there are regions that could be covered well with larger range blocks. This will increase the compression ratio and maintain the clarity. This observation leads to the use of the variable block size partitioning technique.

The second matter to consider is the type of affine transformations to perform on domain blocks. It is required that these transforms be contractive in order for a fixed point to be reached in the decoding stage. The transforms are normally flip operations such as: horizontal, vertical, and diagonal flip; or rotation operations such as: 90^o, 180^o, and 270^o, and rotation. Also the identity block forms one of the transformations [12].

PIFS image encoder consists of a set of transforms applied on the regions of the image (i.e., range blocks). The transforms are, firstly, used to generate the overlapped

domain regions. Secondly, a set of spatial contractive affine transforms are used to approximate the image range blocks by linearly mapping the most similar domain block. For a range block with pixel values (r0, r1,....,rm-1), and the domain block (d0,d1,....,dm-1), the contractive affine approximation is:

$$r_i^{'} = sd_i + o \quad (2)$$

Where

s (scale) and o (offset) are the affine transform coefficients,

r's are the approximate (constructed) range values.

The affine transform is changed to become:

$$r_i^{'} = s(d_i - \bar{d}) + \bar{r} \quad (3)$$

Where $\bar{r} = \frac{1}{m}\sum_{i=0}^{m-1} r_i$,

$$\bar{d} = \frac{1}{m}\sum_{i=0}^{m-1} d_i \quad (4)$$

Objects are represented as a collection of pixels in an image. Thus, for the purpose of recognition the properties of groups of pixels are needed to be determined. The description is often just a set of numbers (i.e., the object's descriptors). Moments describe the shape's layout (i.e., the arrangement of its pixels), a bit like area. Compactness and irregularity order descriptions.

Moments describe the shape's layout (i.e., the arrangement of its pixels), a bit like area. The calculation of moment invariants for any shape requires knowledge about both the shape boundary and its interior region. The moments used to construct the moment invariants are defined to be continuous but for practical implementation they are computed in the discrete form. Given a function f(x,y), the regular moments are defined as follows[13]:

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y) \quad (5)$$

m_{pq} is a two-dimensional moment of the function f(x,y), the order of the moment is (p+q), where p and q are both integer numbers. The coordinates of the centre of gravity of the image are calculated using the following equations[13]:

$$x = \frac{M_{10}}{M_{00}}, \quad y = \frac{M01}{M_{00}} \quad (6)$$

The central moments can be defined in their discrete representation as:

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x,y) \quad (7)$$

For an image block $f(x, y)$ the central moment of order (p+q), around the block's central point (xc,yc), is defined as:

$$M(p,q) = \sum_x \sum_y (x - x_c)^p (y - y_c)^q f(x,y) \quad (8)$$

When this definition is applied to determine the nth order central moments of the zero mean range and domain image blocks, we get:

$$M_d(n,0) = \sum_{i=0}^{m=1} (x_i - L_c)^n (d_i - \bar{d}) \quad (9)$$

$$M_d(0,n) = \sum_{i=0}^{m=1} (x_i - L_c)^n (d_i - \bar{d}) \quad (10)$$

$$M_r(n,0) = \sum_{i=0}^{m=1} (x_i - L_c)^n (r_i - \bar{r}) \quad (11)$$

$$M_r(0,n) = \sum_{i=0}^{m=1} (y_i - L_c)^n (r_i - \bar{r}) \quad (12)$$

Where, $L_c = \frac{L-1}{2}$, (13)

L is the block width of the image

m is the number of block elements

(x_i, y_i) are the x and y coordinates of ith elements (pixel).

From the pair of nth moments {i.e., $M(n,0) \& M(0,n)$ }the following moments blocks descriptors could defined[13]:

$$R_n = \frac{M^2(0,n) - M^2(n,0)}{M^2(0,n) + M^2(n,0)} \quad (14)$$

Combining equation (3) with equations (9-13), and substitute the result in equation (14) we can easily prove that:

$$R_n^d = R_n^r \quad (15)$$

Where, R_n^d is the descriptor value of the domain block, R_n^r is the range block descriptor value

The above implies that "if any two blocks (from range and domain) satisfy the contractive affine transform, then their moments-based descriptor values should have similar values ($R_n^d = R_n^r$) whatever their isometric state. This does not mean that any two blocks have similar R factors are necessarily similar to each other" .

The main disadvantage of classical affine transform scheme is the greedy search in domain pool which is time

consuming. In this paper some improvements have been made on the searching scheme to be selective instead of exhaustive. The improvements aimed to speed up the affine transform coding drastically without causing degradation in secret image quality. To make the searching process selective two moments based descriptor have been used to index (i.e., classify) the cover and secret blocks listed in domain and range pools, respectively. As mentioned before, the first introduced block index parameter is used to describe the isometric state of the block, while the second parameter is used to classify the blocks into categories.

Decompression is simpler and much faster than the coding process. Using the stored transform data, an iterative process commences, until the final image is reached [13].

The number of blocks partitioned relates directly to the compression ratio and PSNR. The number of blocks is related on the compression ratio and PSNR. As the number of blocks increases, the higher the PSNR and the lower the compression ratio, and vice versa. The peak signal-to-noise ratio (PSNR) per color channel is defined as:

$$\text{per} - \text{channel} - \text{PSNR} = 10 \log \frac{255^2}{\text{MSE}} \text{dB}, \quad (16)$$

$$\text{Where} \quad \text{MSE} = \left(\frac{1}{\text{nxn}}\right) \sum_{i=1}^{n} \sum_{j=1}^{n} \left(x_{ij} - \hat{x}_{ij}\right)^2 x_{ij} \ ,$$

x_{ij} is the value of a specific color channel in the original image,
$\hat{x}_{ij}$ the decompressed one [14].

The first widely used signal coding technique was Pulse Code Modulation (PCM),consisting of the independent digitization and coding of signal samples. Limited compressionis achievable, however, since inter-sample dependence is completely ignored. The dependence between pixels may be taken into account by coding the prediction errorfor each pixel, using a prediction based on the values of previously encountered pixels in each scan line.Linear prediction provides a practical alternative only requiring knowledge of the autocorrelation function of the scan lines. The coding of the resulting linear prediction errors is known as Differential PCM (DPCM) [15]. In this paper, DPCM encoding is applied to the fractal coefficients (scale and average) values for each frame.

The compression performance is computed by the following equation [16]:

Compression-Rate=(Size after compression)/
(Size before compression) (17)

Compression-Ratio=100(1 − Compression Rate) (18)

5. The Proposed System

The system consists of two stages: encoding and decoding. In the decoding stagethefirst step is to open AVI structure and get the sequence of frames / images, each image then will be compressed using the FFIC separately.In FFIC the image is converted from RGB to YC_bC_r color space. The fractal domain blocks for each component (Y, C_b, and C_r), are obtained by down sampling the image by 2. Differentparameters like jump step and block size are used in the matching process between the range and domain blocks. The range and domain blocks can have any length, not like in the classical FIC systems where the block width and height must be divisible by the block length. To solve this problem, a band resizing operation is applied, it is done by inserting some additional lines and/or columns to the images in order to make the width and height of band as multiple of block length. The difference in length is managed at the encoding and decoding stages using image re-sampling method.

By using the centralized moment descriptor the domain blocks are sorted to speed up the search operation to find the best matching block among the domain blocks. The moment descriptor is also used to determine the suitable rotation state for each matching, so the fractal image compression becomes very fast and effective. After encoding each component, the fractal codes are quantized and saved into a file.

The following algorithm shows the encoding steps of the AVI video file (see Fig. 1):

Input: AVI video file
Output: Compressed file.

Step 1: read the AVI file.
Step2: split the AVI file into a series of frames (bitmap images (BMP)).
Step 3: for each frame do the following:
Step 4: read the color image (frame) and convert it to YC_bC_r color space.
Step 5: apply FFIC to compress each component (Y, C_b, and C_r)
Step 6: fractal code is quantized, DPCM is applied for the components (Y, C_b, and C_r)
Step 7: save fractal codes for the components (Y, C_b, and C_r) into the compressed file.
Step 8: goto step 3

End compression.

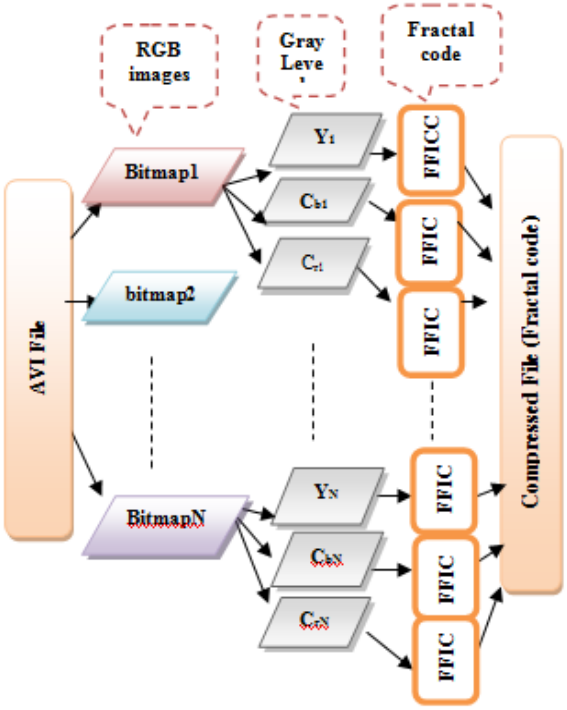


Fig. 1: Encoding flowchart of AVI file (Compression)

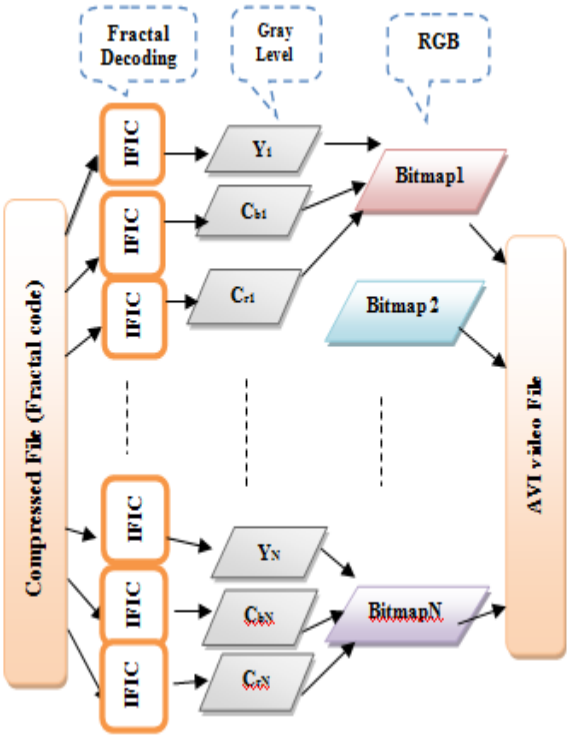


Fig. 2: Decoding steps of the compressed AVI file (Decompression)

The decoding stage in the reverse steps are performed. The fractal codes for each image (with its three components) are loaded from the compressed file and decoded using fractal inverse coding and then converted to the RGB color space. Then the resulted images (frames) are combined together in an AVI video file. The decoding steps are performed as listed in the following algorithm (Fig. 2): The following algorithm shows the decoding steps of the AVI video file:

Input: Compressed video file.
Output: AVI video file

Step 1: read the fractal code for each image (frame) components (i.e. Y, C_b, and C_r) from the compressed file in sequence.
Step 2: for each image component (fractal code) do the following:
Step 3: perform the inverse fractal image compression (IFIC) to each component and retransform them from YC_bC_r to RGB color space images (bitmap images).
Step 4: combine the series of bitmap images into AVI video file decompression program.

End decoding

6. Experimental Results:

The compression algorithm is conducted on the three AVI videos (Bfly, Dance and Cat) which are selected to be different in details and number of frames. The videos runs at (30) frames/second.

The algorithm was tested and performance parameters (encoding time, average PSNR, size before and after compression, compression ratio and rate) were registered. The algorithm used to compress the AVI video many times using different fractal parameters like block size as shown in table (3) and different jump step as shown in table (4). The highlighted rows in these tables represents the best results after decompressing the videos which proved that the three videos have a good quality, good compression ratio and low encoding time when the jump step is 2 and the block length is 5.

Fig. (4) shows that the encoding time was decreased with the increasing of the block size for all videos, the same happens with the jump step in Fig. (5), but making these parameters higher than the suggested values will affect the quality of the video as shown in Fig. s (6-7). The compression ratio (CR) was increased when the block size and the jump step were decreased (Fig. s 8-9), but making these parameters lower than the suggested values will make the compression operation slower.

Table 3: The effect of jump step on the average testing results for the video images

AVI	Jump step	Total E-time (sec)	Total D-Time (sec)	Total size after compr. (MB)	Average PSNR	Average CR	Average Rate
Bfly	1	168.7	30.2	3.4	25.8	14.2	1.68
	2	160.2	33.22	3.35	26	14.5	1.5
	3	154	32.99	3.25	24.9	14.7	1.4
Dance	1	44.85	32.07	2.57	38.5	18.3	1.31
	2	36.78	32.32	2.31	35.1	20.3	1.7
	3	34.36	31.68	2.2	35.2	21.3	1.12
Cat	1	160.0	24.76	2.25	35.3	14.7	1.62
	2	70.3	24.21	3.1	34.4	15.9	1.5
	3	43.6	24.77	1.97	34.2	16.7	1.43

Table 4: The effect of block size on the average results for the three video images

AVI	Block size	Total-E-time (sec)	Total-D-time (sec)	Average size(MB)	Average PSNR	Average CR	Average Rate
Bfly	4	176.86	30.91	5.18	28.1	9.22	2.62
	5	160.2	33.22	3.35	26	14.99	.5
	6	179.4	31.96	2.29	24.22	20.77	1.15
Dance	4	35.46	30.69	3.63	34.75	13	1.8
	5	37.61	32.23	2.31	33.12	20.38	1.7
	6	32.04	33.61	2.2	31.77	28.99	0.83
Cat	4	57.63	24.91	3.25	36.22	10.17	2.35
	5	64.38	24.21	3.1	34.4	15.93	1.5
	6	73.53	25.42	1.47	33.13	22.52	1.06

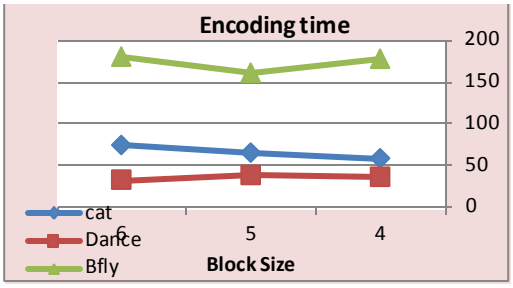


Fig. 4: The effect of block size on the Encoding time

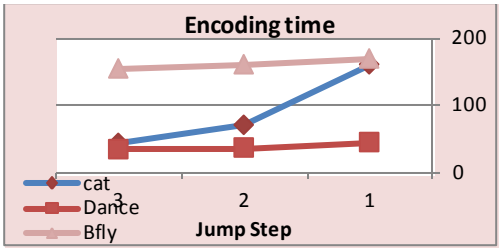


Fig. 5: The effect of Jump step on the Encoding time

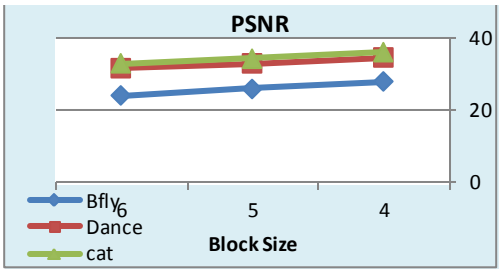


Fig. 6: The effect of block size on the PSNR

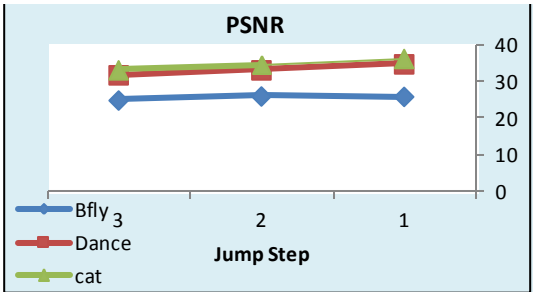


Fig. 7 : The effect of Jump step on the PSNR

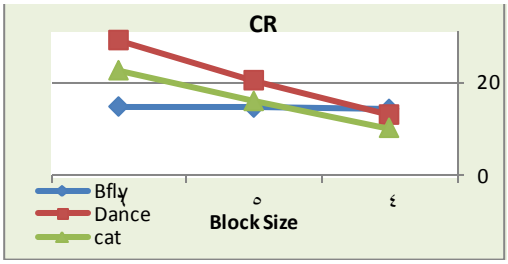


Fig. 8: The effect of block size on the CR

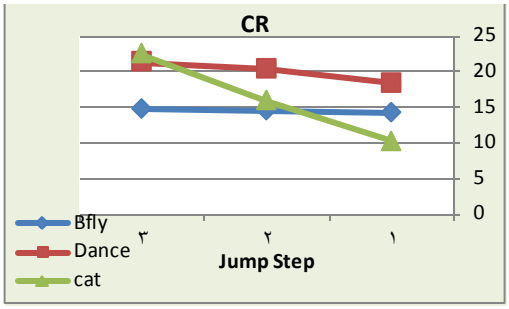


Fig. 9: The effect of Jump step on the CR

The Total results for the whole AVI video frames for all videos are computed and registered in (table 5), which are the encoding time, the average PSNR computed by the equation (16) , the best block size selected from table (3), jump step selected from table (4), the file size to determine the compression ratio that computed from the equations (17 and 18), and sizes of these videos before and after compression.

Table 5: The total tested results for all of the three AVI videos using best block size and jump step

AVI	No-of-Frames	Total E-time (sec)	Av. PSNR	Size before Comp. (MB)	Size after Comp. (MB)	Total CR	Rate
Butt	150	160.2	26	47.4	3.35	92.932	0.071
Man	162	36.78	35.1	476	2.31	95.147	0.049
Cat	150	70.38	34.4	33.1	3.1	93.716	0.063

A comparison was done between the proposed research and other researches that used the video compression using fractal image compression in table (60). The comparison shows that the proposed research have the best compression ratio with good quality and rate. Although the encoding time did not mentioned in some of these researches, it is an important parameter used to evaluate any fractal compression method because of the low compression speed it suffers from. The proposed research proved low encoding time with low rate in all tested videos comparing them with other researches.

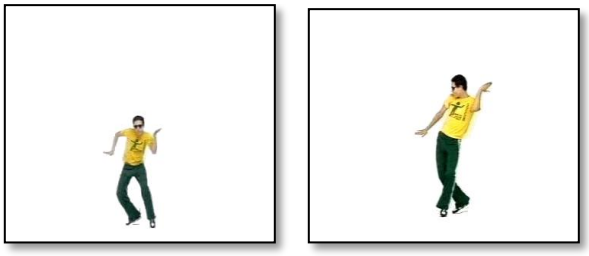
Table 6: Comparison table with others related researches

Research	No-of-Frames	Cr	Av. PSNR	Time (sec)	Rate (bit/ pixel)
[2]	--	74.39	33	--	0.056
[8]	150	--	33.51	--	--
[9]	--	50	40	--	--
[17]	25	--	37	>309.7	8.46
Proposed (Bfly)	150	92.93	26	160.2	0.071
Proposed (Dance)	162	95.147	35.1	36.78	0.049
Proposed (Cat)	150	93.72	34	70.38	0.063

Considering the samples of the reconstructed videos frames in Fig. (3), it is obvious that the reconstructed AVI videos have a very good quality and FFIC is a perfect compression method to be use.

(a) Frames 40 and 140 of the reconstructed "Cat"

(b) Frames 20 and 75 of the reconstructed "Bfly"



(c) Frames 20 and 130 of the reconstructed "Dance".

Fig. 3: Selected frames for each of the three reconstructed videos

7. CONCLUSIONS

Fractal image compression always suffer from long encoding time, but in our research, this problem is managed using FFIC which is a fast method that speeded up the compression by using the moment descriptors. It is an effective way to compress AVI images when comparing it with the traditional and other fractal video compression methods.

The investigation and performance analysis shows the advantages of the proposed compression method using the FFIC on the three AVI videos, and the resulted values in all above tables and graphs proves that reconstructed videos have high PSNR, also the CR in all tested videos was about (92-94)% of the actual size of the original video for tested videos. The selected block size was 5 with jump step 2, and these parameters are very important to get good quality, CR and low encoding time . To have more compression and faster encoding time, these parameters can be increased, but this will affect the quality of the video.

8. REFERENCES

[1] Nabeel H., Tawfiq A. and Wafaa H., "Video Clip Image Compression Using DCT Technique" , Information technology, University of Babylon, Iraq, 2011.

[2] M. S. Lazar, L. T. Bruton, "Fractal Block Coding of Digital Video", IEEE Trans. Circuits and Systems for Video Technology, vol.4, no.3, pp. 297-308, June 1994.

[3] A. Jacquin, "Image coding based on a fractal theory of iterated contractive image transformations", IEEE Trans. Image Processing, vol. 1, no. 1, pp.18-30, Jan. 1992.

[4] Y. Brijmohan and S. H. Mnene, "Video Compression for Very Low Bit-Rate Communications Using Fractal and Wavelet Techniques ", e-book browse, No 19, 2010.

[5] Y. Fisher, "Fractal Image Compression: Theoryand Application", Springer-Verlag, New York, Inc., 1995.

[6] B. Wohlberg and G. d. Jager,"A Review of the Fractal Image Coding Literature", IEEE Transaction on Image Processing, Vol. No. 12, pp. 1999.

[7] G. Melnikov and A. K. Katsaggelos, "A Jointly Optimal Fractal/DCT Compression Scheme", IEEE Trans. on Multimedia, vol. 4, No.4, pp.413-422, 2002.

[8] G. M. Davis, "A Wavelet-Based Analysis of Fractal Image Compression", IEEE Transaction Image Processing, USA, Vol. No. 2, pp. 1998.

[9] H. Wang, Q. Wu, X. He and T. Hintz, "Preliminary Research On Fractal Video Compression On Spiral Architecture", IEEE, Proc. IPCV , pp.557-562, 2006.

[10] Abhayaratne G. , "Lossless and Nearly Lossless Digital Video Coding", for the degree of Doctor of Philosophy of the University of Bath, 2002.

[11] "File Extension Database", VLC media player for Windows, http://file.ms/extension/avi/.

[12] James D. Murray and William V. Ryper, "Encyclopedia of Graphics File Formats", Second Edition, O'Reilly & Associates, Inc, ISBN: 1-56592-161-5, 1996.

[13] Loay E. George, Suad K. Ahmad, "Hiding Image in Image Using Iterated Function System (IFS)", European Conference of Computer Science (ECCS '10), Image, Sound and Signal Processing, ISBN: 978-960-474-250-9, 2010.

[14]"Encyclopedia of File Format", http://netghost.narod.ru/gff/graphics/book/ch10_03.htm.

[15] Wohlberg B., "Fractal Image Compression and the Self-Affinity Assumption: A Stochastic Signal Modeling Perspective", Ph.D thesis, University of Cape Town, 2003.

[16] Mustafa M. , "Using Shift Number Coding with Wavelet Transform for Image Compression", UK Journal of Information and Computing Science, England, Vol. 4, No. 3, pp. 311-320, ISSN , 746-7659, 2009.

[17] Mukhopadhyay, J. and Ghosh, S.K., "Low bit rate video compression using relative fractal coding" , Proc. of EUSIPCO, France,Vol.II, 125-128, 2002.

Faten H. MohammedSediq Assistant professor in the Faculty of Technical Institutes/ Technical Institute / Kirkuk-Iraq, Graduated in Mathematical Department /College of Science-University of Mosul with the degree of B.Sc., In 1987 graduated from the Computer Science in the Computer National Center-Iraq with the degree of Msc., Ph.D. in computer sciences, Technology University-Iraq in 2006; Research Interests: Detection and Addition in 3D Object Animation, Image Processing.

Nevart A. Minas Assistant teacher in the Faculty of Technical Institutes/ Technical Institute of Kirkuk - Iraq, B.Sc. in Computer Science in Mosul University-Iraq in 1989, M.Sc in Image Processing , collage of Science, University of Sulaimania -Iraq in 2009; Research Interests: image and video compression techniques (transform coding, fractals, wavelet…), and Simulation.