Hybrid Partitioning Technique For Fractal Image Compression

Bashar Talib Hameed
Diyala University, College of Science,
Computer Science Department

Abstract:

In fractal compression the image to be encoded is partitioned into blocks called ranges. Each range is coded by reference to some other part of the image called domain and by some affine transformation parameters. The number of ranges plays an important rule in encoding parameters. In order to obtain high compression ratio, only a small number of blocks are allowed in encoding process. In this paper we suggested a new technique called Composed Partitioning Technique (CPT) to reduce the number of partitioning blocks with keeping as much as possible the quality of the reconstructed image. CPT is used the features of both quadtree and H-V techniques, it uses the uniformity criterion to decide the range block is to be partitioned or not, then it uses a criterion that depends on both the pixel and the edges information of the image to decide the type and the direction of the partitioning and this will give a more restriction to the partitioning process.

Keywords: fractal, partition, affine transforms quadtree and H-V partitioning

1. Introduction

In fractal image compression the image to be coded is partitioned into blocks called ranges (R_i). As a common principle, more partitioning techniques are intended to cut the complicated and detailed regions into smaller pieces and to group the simple and predictable ones together [1]. Partitioning techniques can be classified into two major types:

❖ Fixed Partitioning Technique

Fixed Partitioning Technique (FPT) is simple and it requires less computational time than the other techniques. The image to be compressed is partitioned into blocks of equal size (fixed-size square-shaped). So in this method block size is an affect parameter that affects the partitioning process. The resulting partitioned image will contain blocks of the same size; each of them represents a range block. Figure (1) illustrates FPT:

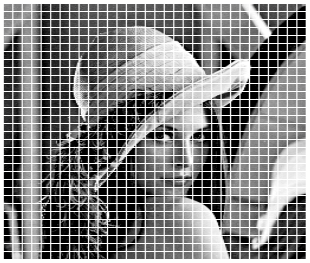

Figure (1): Example of fixed partitioning when: Block size = 8.

❖ Hierarchical Partitioning techniques:

Hierarchical Partitioning Techniques (HPT) are adaptive partitioning schemes that take the difficulties and the natural connections between areas into account, so the image to be partitioned using HPT will be partitioned into regions (i.e. Ranges) of different size depending on the image region content, smaller ranges for the regions having more details and larger ranges for regions having simple details or smooth [5],[6].

There are many techniques work under this type, like: Quadtree and H-Partitioning (see figs (2 and 3)).

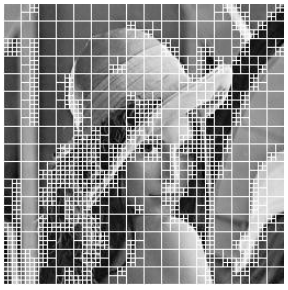

Figure (2): Quadtree partitioning, when MaxBlSz=16, MinBlSz=4,I_f=0.3, R_a=0.1,and No.of blocks=2458

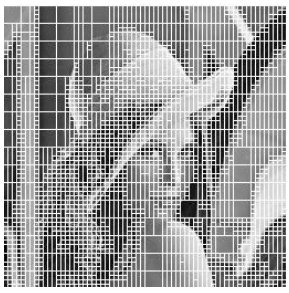

Figure (3): H-V partitioning, when MaxBlSz=16, MinBlSz=4, I_f =0.3, R_a=0.1, andNo.of bocks=2256

2. Proposal Technique

In Quadtree Partitioning Technique (QPT), the image to be coded is partitioned into square-shape blocks of different sizes to discover the uniformity criteria among blocks. After making the decision of partitioning, the range will be divided directly into four equal sub-squares. The result partitioned image is called range blocks.

The range blocks are created as follows:

1) Assume the whole input image is defined as only one range block R_0,
2) Then compute the global mean ($\overline{M}$), the global squared mean ($\overline{M^2}$) and the standard deviation (σ) of the input image (i.e., for $_{R0}$) using the following equations:

$$\overline{M} = \frac{1}{W \times H} \sum_{y=1}^{W} \sum_{x=1}^{H} R_0(x, y) \quad \text{.....(1)}$$

$$\overline{M^2} = \frac{1}{W \times H} \sum_{y=1}^{W} \sum_{x=1}^{H} (R_0(x, y))^2 \quad \text{.....(2)}$$

$$\sigma = \sqrt{\overline{M^2} - (\overline{M})^2} \quad \text{.....(3)}$$

$(\overline{M})$ And σ are measures of distribution of the pixel values in the image.

The extended standard deviation (σ_E) is used as a tolerance criterion. It is computed as multiples of standard deviation.

3) Select some control parameters of partitioning process to decide the image to be partitioned or not these parameters are:

I) Maximum Block Size: represents the maximum size of the range block which corresponds to the minimum depth of the tree partitioning.

II) Minimum Block Size: represents the minimum size of the range block which corresponds to the maximum depth of the tree partitioning.

Acceptance Ratio: represents the ratio of the number of pixels whose gray values differ from the block mean by a distance farther than the expected σ_E value. As the 'Ratio' value selected is small it can be get a higher quality of the reconstructed image, and vice versa

III) Inclusion Factor (I_f): represents the multiple factor, when it is multiplied by σ, it will define the value of σ_E. The selection of small I_f value will lead to high quality and low compression ratio of reconstructed image, and vice versa.

4) A linked list has been designed to store all information about QP process. This linked list is defined as an array of records. Each record consists of the following fields:

I) Position: represents the upper left coordinates of each image block (i.e. x and y).

II) Size: represent the size of each image block, which is equal either to width or height of the image block, since in quadtree the blocks are of square shape (i.e., width = height), thus there is no problem to choose any one of them.

III) Next: It is a pointer to the next range block.

5) At the end, QP process usually starts with partitioning the image into blocks whose size is equal to the maximum block size. If any sub-block satisfies the uniformity condition then the block is accepted and stored, otherwise, the block should be partitioned into four sub-blocks. Each block should be examined by measuring its uniformity. If it does not satisfy the uniformity criteria, partitioning should be proceeded. The partitioning process is repeated until the Uniformity condition is satisfied (see algorithm 1). After that, the constructed Quadtree will consist of partitions whose size value will

be between minimum and maximum block size. All these steps can be summarized in algorithm (2).

Two parameters are used to control uniformity function, which are the Inclusion Factor (I_f) that is used to adjust the extended standard deviation ($\sigma_E = I_f.\sigma$) (Extended standard deviation is used as a metric for the degree of minimum allowed dispersion in the spatial distribution of the gray values). The other parameter is the Ratio. This factor is used as a tolerance for relative number of pixels within the block, which may differ from the mean value of the block by distance of more than σ_E.

Algorithm (1): Uniformity function.

Step1: If (B <= Min) then return (True)
Step2: Else if (B > Max) then return (False)
Step3: Else
a) Compute the local Mean (M) of each image block defined by Q[Current Item].
b) Set N_p=0 {No. of undesired pixels}
c) For each pixel G(x,y) within the image block do {
If abs(G(x,y)-M)> σ_E then N_p=N_p+1
d) If ((N_p/block size)> Ratio) then return (False) else return (True)

Where: B is the blocks QuadTreeLinkList record,
Max is the Maximum Block Size,
Min is the Minimum Block Size and
Thr is the threshold difference value

Algorithm (2) Quadtree partitioning process

Step1: Set $R_0=I^2$.
Step2: Compute the global mean ($\overline{M}$) and squared mean ($\overline{M^2}$) of the input image (i.e., of R_0).
Step3: Compute the global standard deviation (σ) of the input image.
Step4: Compute the extended standard deviation (σ_E) of the input image.
Step5: Choose the partitioning control parameters.
Step6: Set NoItem=1and Current Item=0.
Step7: Initialize the first record {Q[0] } of QuadTreeLinkList with the primitive values:
Q[0].X=0, Q[0].Y=0,
Q[0]. Siz =W, Q[0]. Nxt =Nil.
where W is the image width.
Step8: While content of Q[Current Item]<> Nil do
i) If the Uniformity (Q[Current Item]) = False then Partition the image block Q[Current Item] into four quadrants, by following the steps:
1. Set
X_p= Q[Current Item].X,
Y_p= Q[Current Item].Y,
S= Q[Current Item] .Siz/2,
N= Q[Current Item].Nxt.

2. Create the following new link list record
Q[Current Item].X =X_p, Q[Current Item].Y =Y_p,
Q[Current Item].Siz =S,
Q[Current Item].Nxt =NoItem.
Q[NoItem].X =X_p+S, Q[NoItem].Y =Y_p,
Q[NoItem].Siz =S,
Q[NoItem].Nxt =NoItem+1.
Q[NoItem+1].X =X_p,
Q[NoItem+1].Y =Y_p+S,
Q[NoItem+1].Siz =S,

Q[NoItem+1].Nxt =NoItem+2.
Q[NoItem+2].X =X_p+S,
Q[NoItem+2].Y =Y_p+S,
Q[NoItem+2].Siz =S, Q[NoItem+2].Nxt =N.

ii) Else Set Current Item = Q[Current Item].Nxt

In fact the superiority of HV Partitioning Technique upon the quad tree scheme is motivated by the following reasons [2, 3, and 6]:

I) HV rectangular partitions have been shown to lead to better decoding results than the quad tree technique, both in terms of rate– distortion performance and visual quality.

II) HV partitioning technique produces the fine–grained partition, which is computationally faster than utilizing the quad tree scheme.

Though the HV has superiority in some aspects, but the quad tree technique may be considered more convenient in others, (i.e., in H-V partitioning, extra bits should be used to retain the dimension of each partition block in the range pool. Extra bits will as a result, decrease the resulting compression factor. In the quad tree technique only two parameters have to be stored which represent the initial x and y location of each block in the range pool).

To create the ranges, the following steps are followed:

I)Assume the entire image is defined as one potential range R_0.

II)Then the global mean $(\overline{M})$, the global squared mean $(\overline{M^2})$ and the standard deviation (σ) of the input image (i.e., for R_0) are computed using equations (1), (2) and (3) respectively, then the extended deviation (σ_E) of the input image is determined as a difference tolerance criteria (i.e., condition).

III) Standard deviation (σ_E) of the input image is determined as difference tolerance criteria (i.e., condition).

IV) Same partitioning control parameters (i.e., maximum, minimum block size, acceptance ratio and inclusion factor) are adopted.

V) HV partitioning information is stored by designing the HVLinkList which is looks like the QuadTreeLinkList used in the quad tree technique, it is consist of the same fields except the size field, since HV partitioning based on rectangles (i.e., width not equal to height), thus needed to assign the size in both directions (i.e., x and y) of each range block. The designed HVLinkList consists of the following fields:

1. **Position:** Represented by the x and y coordinates, which represent the position of top-left corner of the range block.
2. **X_Size:** Represent the width of range block.
3. **Y_Size:** Represent the height of range block.
4. **Next:** Represent a pointer to next block.

VI) Finally the Uniformity criterion is used as a measure to decide whether the range block will be partitioned into halves sub blocks, or not, and in the case of partitioning the direction of partitioning is also assigned by the uniformity.

All the above steps can be summarized by the algorithms (3) and (4) which they are explain the partitioning process and the partitioning decision (i.e., uniformity).

Algorithm (3): HV Partitioning Process.

Step1: Set $R_0=I^2$.
Step2: Compute the global mean ($\overline{M}$) and squared mean ($\overline{M^2}$) of the R_0.
Step3: Compute the global standard deviation (σ) of the R_0.
Step4: Compute the global extended standard deviation (σ_E) of the R_0.
Step5: Assign the values of the partitioning control parameters.
Step6: Set NoItem =1and CurrentItem=0.
Step7: initialize HVLinkList array {p[0]} with the primitive values:
P[0].X=0, P[0].Y=0 P[0].XSize= W,
P[0].Ysize= H, P[0].Next= Nil.
Where W is the image width and H is the image height.

Step8*:* While the P[CurrentItem]<>Nil do{

- Determine the index (Q) where (Q= Uniformity (P[CurrentItem]))
- If (Q>0) then {
  I= P[CurrentItem].Next,
  P[CurrentItem].Next= NoItem,
  P[CurrentItem].Next= I.
  - If (Q=2) (Horizontal partition) then {
    W= P[CurrentItem].XSize/2,
    P[CurrentItem].XSize= W,
    P[NoItem].X= p[CurrentItem].X+W,
    P[NoItem].XSize=W.
  - Else if (Q=1) (Vertical partition) then {
    H= P[CurrentItem].YSize/2,
    P[CurrentItem].YSize= H,
    P[NoItem].Y= P[CurrentItem].Y+H,
    P[NoItem].YSize=H.
- Increment NoItem by one.

else {
CurrentItem = p[CurrentItem].Next

Algorithm (4): Uniformity Function

Step1: If (B. XSize >Max) then Return (Horizontal partition) (i.e., Q=2).
Step2: Else if (B. YSize >Max) then Return (Vertical partition) (i.e., Q=1).

Step3: Else if ((B. XSize > Min) or (B. YSize > Min)) then

a) Compute the local Mean (M) of image block defined by Q[Current Item].
b) Set $N_p=0$ "No. of undesired pixels"
c) For each pixel G(x,y) within the image block do
If abs(G(x,y)-M)> σ_E then $N_p=N_p+1$

determine the location of G(x,y)
if G(x,y) on the Left half then SL++
else if G(x,y) on the Right half then SR++
else if G(x,y) on the Top half then ST++
else if G(x,y) on the Bottom half then SB++

d) If ((N_p/block size)<= Ratio) return (Don't partition) (i.e., Q=0)
else
Determine the direction of partitioning as following
if SR>SL then Dx=SR else Dx=SL
if ST>SB then Dy=ST else Dy=SB
if Dx >= Dy then return (Horizontal partition) (i.e., Q=2)
else if Dy > Dx)) then return (Vertical partition) (i.e., Q=1)
else return (Don't partition) (i.e., Q=0)

Where: B: Is the blocks HVLinkList record, Max: Is the Maximum block size,
Min: Is the Minimum block size,
SL, SR, ST and SB: Are the summations of the undesired pixel on the left half, right half, top half and bottom half respectively.

Our suggested technique called Composed Partitioning Technique (CPT) based on new criteria for partitioning decision. CPT will also depend on the image region details (pixel value) to make the decision about partitioning (i.e the region will be divided or not).

CPT after make the decision of partitioning, additional computations must be done to determine the type of the partitioning (H-V or Quadtree), and if the type is H-V then the range will be divided horizontally or vertically into two equal sub-rectangles else the range will be divided into four equal sub-squares.

The decision of determining the partitioning type (H-V or Quadtree) and the direction of the partitioning (in case of H-V) will be depended on new more powerful computations, these computations will take in its account additional features in the image which is the strength of its edges that generated from the original image using Sobel filter[7].

$$S_h = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad S_v = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

The edge information of the image at the pixel point (x, y) is determined by equation (4):

$$G(x,y) = \sqrt{(S_h)^2 + (S_v)^2} \quad \text{.....(4)}$$

2.1 Partitioning Criteria

In this section we will discuss the Partitioning Criteria are used in the CP technique to decide if the range is to be partitioned or not and the criteria of determining the type and the direction of the partitioning.
We are use three type of criteria, they are:

- **Homogenous Criteria**

Image partitioning is implemented by "Uniformity" criteria. This criterion is used as measurement to decide whether the image block will be partitioned or not. However the uniformity criteria implies the following condition [4]

$$uniformity\ criteria = \begin{cases} True & if\ n > R_a BS \\ False & otherwise \end{cases} \quad \text{.....(5)}$$

Where

n: is the number of block pixels, that yields a difference from the mean of the block higher than a threshold value (σ_E),

R_a : is a proposed parameter called acceptance ratio ranging (0.07-0.2)

BS: is the range block size.

While the threshold value (σ_E) has been considered to be:

$\sigma_E = If\sigma$ ….. (6)

Where

σ_E: is the standard deviation of the whole soble image, and

If: is a proposed parameter called inclusion factor ranging (0.7-2).

- **Partitioning Type and Direction Criteria**

Addition computations must be done to decide the type and the direction of the partitioning, when the image regions must be partitioned, these computations are:

1. Determine the mean values of the four quarters of tested range block. Let M_{LT}, M_{RT}, M_{LB}, M_{RT} are the mean values of the left-top, right-top, left-bottom and right-bottom quarters, respectively.
2. Determine the mean values of the left half (M_L), right half (M_R), top half (M_T) and bottom half (M_B) of the range block as following:

$$M_L=½ (M_{LT}+M_{LB})$$
$$M_R=½(M_{RT}+M_{RB}) \quad \text{….. (7)}$$
$$M_T= ½(M_{RT}+M_{LT})$$
$$M_B= ½ (M_{RB}+M_{LB})$$

3. Using the results of eq. (7) to determine the horizontal and vertical differences:

$$d_H =| M_L-M_R| \qquad d_V =| M_T-M_B| \quad ...(8)$$

4. Then the greater difference is checked, if $d_H>d_V$ and $d_H>T$ (i.e threshold value) then a horizontal partitioning decision will be taken. While, if $d_V>T$ then a vertical partitioning decision is taken, if the $d_H=d_V$ or if they are equal to or less than T then the Quadtree partitioning decision is taken (that means this range block has significant differences between all its quarters), otherwise there is no partition (i.e. the range block need not to be partitioned).

This can be done by determining the index (*Dir*) using the following equation:

$$Dir = \begin{cases} 3 \text{ if(} (d_H > d_V \text{ and } d_H > T \text{ and } X\text{-}Size > MinBlSiz) \text{ or} \\ \quad (X\text{-}Size > MaxBlSiz)). \\ 2 \text{ if } ((d_V > d_H \text{ and } d_V > T \text{ and } Y\text{-}Size > MinBlSiz) \text{ or} \\ \quad (Y\text{-}Size > MaxBlSiz)). \\ 1 \text{ if}((d_V = d_H \text{ or } (d_V <= T \text{ and } d_H <= T)) \text{ and} \\ \quad (X\text{-}Size > MinBlSiz \text{and } Y\text{-}Size > MinBlSiz) \\ 0 \text{ otherwise} \end{cases} \quad \text{.....(9)}$$

Where

The value (3): Means partition horizontally,
(2): Partition vertically,
(1): Partition quadtree and
(0): No partition.

T= R_a Pmean, and

Pmean: Is the mean value of the range pixels.

• Edge Criteria

1. Determine the mean values of the four sub-squares of range block for Sobel image. Let MS_{LT}, MS_{RT}, MS_{LB}, MS_{RT} are the edges mean values of the left-top, right-top, left-bottom and right-bottom quarters, respectively.
2. Determine the edges mean values (see figure (4)) of the left half (MS_L), right half (MS_R), top half (MS_T) and bottom half (MS_B) by using equation (10)

$MS_L = ½(MS_{LT}+MS_{LB})$
$MS_R = ½(MS_{RT}+MS_{RB})$ ….. (10)
$MS_T = ½(MS_{RT}+MS_{LT})$
$MS_B = ½(MS_{RB}+MS_{LB})$


Fig. (4): Extracted and detected the edges of Lenna Image

3. Using equation (10) to determine the horizontal and vertical edges differences as follow:
$d_{HS} = | MS_L - MS_R |$,and
$d_{VS} = | MS_T - MS_B |$ …….(11)

4. Now, we will make use of both the features of pixel information and the edge information in the image to decide whether the range block will be partitioned into two equal rectangular (horizontally or vertically) using the H-V technique or it will be partitioned into four sub-squares using quad-tree technique. This can be done by checking the larger differences in both of the image range block based on the pixel information and the edges information, if $d_H > d_V$ and $d_H > T$ and if $d_{HS} > d_{VS}$ and $d_{HS} > T$ then a horizontal partitioning decision will be taken. While, if $d_V > d_H$ and $d_V > T$ and if $d_{VS} > d_{HS}$ and $d_V > T$ then a vertical partitioning decision is taken, if the $d_H = d_V$ and $d_{HS} = d_{VS}$ or if they are equal to or less than a threshold value then the Quadtree partitioning decision is taken, otherwise there is no partition (i.e. the range block need not to be partitioned).

Here index (*Dir*) in equation (9) can be re determined using the following equation:

$$Dir = \begin{cases} 3 \ \text{if}(((d_H > d_V \text{ and } d_H > T) \text{ and } (d_{HS} > d_{VS} \text{ and } d_{HS} > T)) \text{ and} \\ \quad (X\text{-Size} > \text{MinBlSiz})) \text{ or} \\ \quad (X\text{-Size} > \text{MaxBlSiz})). \\ 2 \ \text{if } ((((d_V > d_H \text{ and } d_V > T) \text{ and } (d_{VS} > d_{HS} \text{ and } d_{VS} > T)) \text{ and} \\ \quad (Y\text{-Size} > \text{MiniBlSiz})) \text{ or} \\ \quad (Y\text{-Size} > \text{MaxBlSiz})). \\ 1 \ \text{if}(((d_H = d_V \text{ and } d_{HS} = d_{VS}) \text{ or} \\ \quad (d_H <= T \text{ and } d_V <= T \text{ and } d_{HS} <= T \text{ and } d_{VS} <= T)) \text{ and} \\ \quad (X\text{-Size} = \text{MinBlSiz and } Y\text{-Size} = \text{MinBlSiz})) \\ 0 \ \text{otherwise} \end{cases} \quad \ldots.(12)$$

Where:
The value (3) means: The range block is to be partitioned, will be partitioned into two range blocks horizontally,
The value (2) means: The range block is to be partitioned, will be partitioned into two range blocks vertically,
The value (1) means: The range block is to be partitioned, will be partitioned into four equal range blocks (Quadtree) and
The value (0) means: The range block is to be partitioned, will be not partitioned.

To create the ranges blocks in our suggested technique the following steps are followed:

1. Assume that the entire image is defined as one potential range (and call it R_0).
2. Then the global mean $(\overline{M})$, squared mean $(\overline{M^2})$ and standard deviation (σ) of the input image (i.e., for R_0), then the extended standard deviation (σ_E) of the input image is determined as a difference tolerance criteria (i.e., threshold).
3. The same partitioning control parameters (i.e., maximum and minimum block size, inclusion factor, and acceptance ratio) that used earlier (in the quad tree and H-V partitioning) are adopted.
4. The Composed partitioning information is stored by designing the ComLinkList which is looks like the HVLinkList used in the H-V method, it is consist of the same fields, since the composed partitioning technique are mixed between squares and rectangles, thus we need to assign the size in both directions (i.e., x & y) of each range block. The designed ComLinkList consists of the following fields:

- Position: Represented by the x and y coordinates, which represent the position of top-left corner of the range block.
- X_Size: Represent the width of the range block.
- Y_Size: Represent the height of the range block.
- Next: Represent a pointer to the next range block.

5. Finally the suggested Uniformity criterion is used as a measure to decide whether the image block will be partitioned into sub blocks or not and in the case of partitioning the type and the direction of partitioning is also assigned by the uniformity.

Algorithms (5) and (6) illustrate the composed partitioning in more details. These algorithms are simple to describe and easily to perform.

Algorithm (5): Composed partitioning process.

Step1: Set $R_0=I^2$.

Step2: Compute the global mean $(\overline{M})$ and squared mean $(\overline{M^2})$ of the R_0.

Step3: Compute the global standard deviation (σ) of the R_0.

Step4: Compute the global extended standard deviation (σ_E) of the R_0.

Step5: Assign the values of the partitioning control parameters.

Step6: Set NoItem =1 and CurrentItem=0.

Step7: Initialize ComLinkList array {p[0]} with the primitive values:

P[0].X=0, P[0].Y=0, P[0].XSize= W,
P[0].Ysize= H, P[0].Next= Nil.
Where W is the image width and H is the image height.

Step8: while the P[CurrentItem]<>Nil do{

- Determine the index (*Dir*) where (*Dir* = Uniformity(P[CurrentItem])
- If (*Dir* >0) then {
  I= P[CurrentItem].Next.
- If (*Dir* =3)(Horizontal partition) then {
  W= P[CurrentItem].XSize/2,
  P[CurrentItem].XSize= W,
  P[NoItem].X= p[CurrentItem].X+W,
  P[NoItem].XSize=W,
  P[NoItem].Next= I, Increment NoItem by one. }
- else if (*Dir* =2) (Vertical partition) then {
  H= P[CurrentItem].YSize/2, P[CurrentItem].YSize= H,
  P[NoItem].Y= P[CurrentItem].Y+H, [NoItem].YSize=H.
  P[NoItem].Next= I, Increment NoItem by one. }
- else if (*Dir* =1) (Quadtree Partitioning) then {
  W= P[CurrentItem].XSize/2, H=P[CurrentItem].YSize/2,
  X_p= P[CurrentItem].X, Y_p= P[CurrentItem].Y,
  P[CurrentItem]. XSize =W, P[CurrentItem]. YSize =H
  P[CurrentItem].Nxt =NoItem.
  P[NoItem].X =X_p+W, P[NoItem].Y =Y_p,
  P[NoItem]. XSize =W, P[NoItem]. YSize =H
  P[NoItem].Nxt =NoItem+1.
  P[NoItem+1].X =X_p, P[NoItem+1].Y =Y_p+H,
  P[NoItem+1]. XSize =W, P[NoItem+1]. YSize =H,
  P[NoItem+1].Nxt =NoItem+2,
  P[NoItem+2].X =X_p+W, P[NoItem+2].Y =Y_p+H,
  P[NoItem+2]. XSize =W, P[NoItem+2]. YSize =H,
  P[NoItem+2].Nxt =I, Increment NoItem by three.
  }
  else { CurrentItem = p[CurrentItem].Next }
}

Algorithm (6): Uniformity function

Step1: If (B.XSize >Max) then Return (Horizontal partition) (i.e., *Dir* =3).

Step2: Else if (B.YSize >Max) then Return (Vertical partition) (i.e., *Dir* =2).

Step3: Else if ((B.XSize > Min) or (B.YSize > Min)) then {

a) Compute the mean for the quarters (M_{TL}, M_{TR}, M_{BL}, and M_{BR}) for the original image and the mean for the quarters (MS_{TL}, MS_{TR}, MS_{BL}, and MS_{BR}) for its edges image (sobel image).

b) Compute the mean for the four halves (M_L , M_R , M_T and M_B) of the original image and the mean for the four halves (MS_L, MS_R, MS_T and MS_B) of its edges image (sobel image) :

$M_L=½(M_{TL}+M_{BL})$, $M_R=½(M_{TR}+M_{BL})$,
$M_T=½(M_{TL}+M_{TR})$, $M_B=½(M_{BL}+M_{BR})$.

And

$MS_L=½(MS_{TL}+MS_{BL})$, $MS_R=½(MS_{TR}+MS_{BL})$,
$MS_T=½(MS_{TL}+MS_{TR})$, $MS_B=½(MS_{BL}+MS_{BR})$.

c) Compute the Horizontal difference for the original image (d_H) and its edges (d_{HS})

$d_H= |M_L-M_R|$, $d_{Hs}= |MS_L-MS_R|$

d) Compute the Vertical difference for the original image (d_V) and its edges (d_{VS})

$d_V= |M_T-M_B|$, $d_{VS}= |MS_T-MS_B|$

e) if ((($d_H > d_V$) and (d_H > Thr)) and (($d_{HS} > d_{VS}$) and (d_{HS} > Thr)))
then return (Horizontal partition) (i.e., *Dir* =3)

f) else if ((($d_V > d_H$) and (d_V> Thr)) and (($d_{VS} > d_{HS}$) and (d_{VS}>Thr)))
then return (Vertical partition) (i.e., *Dir* =2)

g) else if ((($d_H = d_V$ and $d_{HS} = d_{VS}$) or
(d_H <= Thr and d_V<= Thr and d_{HS} <= Thr and d_{VS}<= Thr))
and
(B. XSize > Min and B. YSize > Min))
then return (Quadtree partition) (i.e., *Dir* =1)

h) else return(Don't partition) (i.e., *Dir* =0)
}

Step 4: else return (Don't partition) (i.e., *Dir* =0).

Where B: is the blocks ComLinkList record,
Max: is theMaximum block size,
Min: is the Minimum block size, and
Thr: is the threshold difference value

Experimental Results

As we saw in CPT, varying the values of acceptance ratio or inclusion factor when other parameters are kept fixed, different values for block numbers, compression ratio and PSNR can be achieved. But we have found that the same values of control parameters lead to number of blocks in Quadtree > number of blocks in H-V > number of blocks in Composed. And about the same reconstructed image quality can be achieved in H-V Partitioning with less number of blocks than in Quadtree Partitioning and with less number of blocks in Composed Partitioning than in H-V Partitioning Technique. This means that we can produce a high reconstructed image quality with more compression ratio using Composed Partitioning Technique than using other techniques. Table (1) and Figures (5), (6), (7, and 8) demonstrate the ability of composed partitioning technique over the Quadtree and H-V partitioning techniques

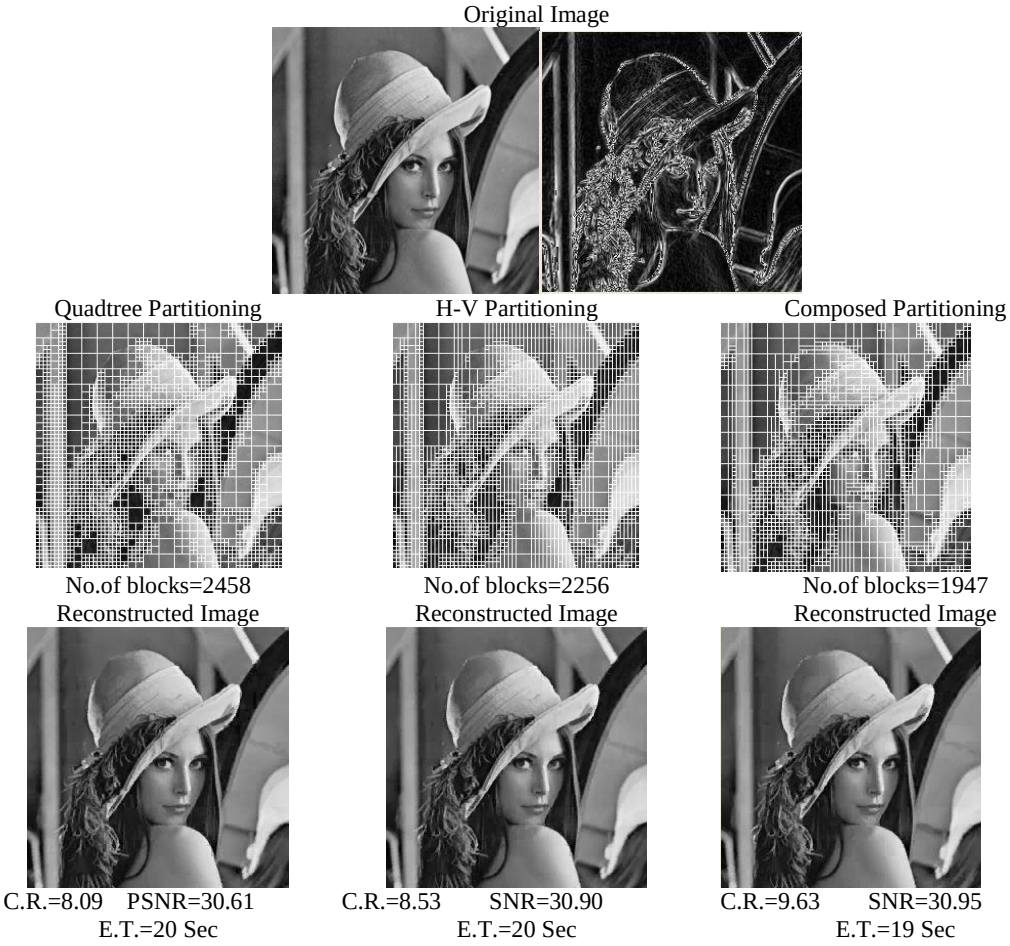

Fig. (5): Comparison between FIC based on Quadtree, H-V and Composed Partitioning Techniques applied on 256x256 gray scale Lenna image.

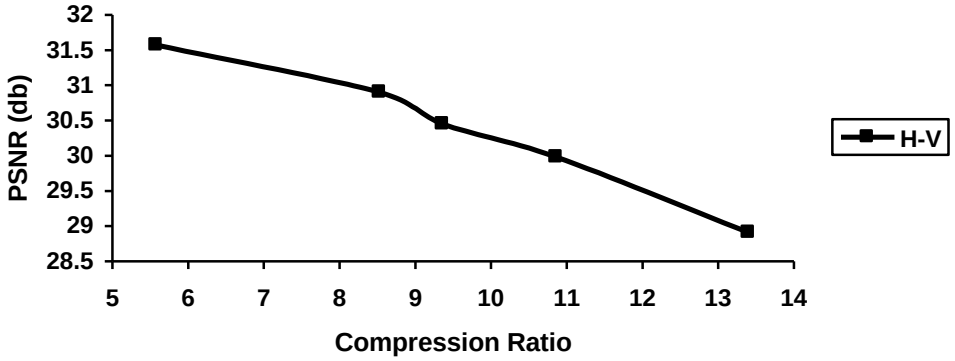

Figure (6): Compression ratio versus PSNR (when SizMin=4, SizMax=16 and R_a=0.1).

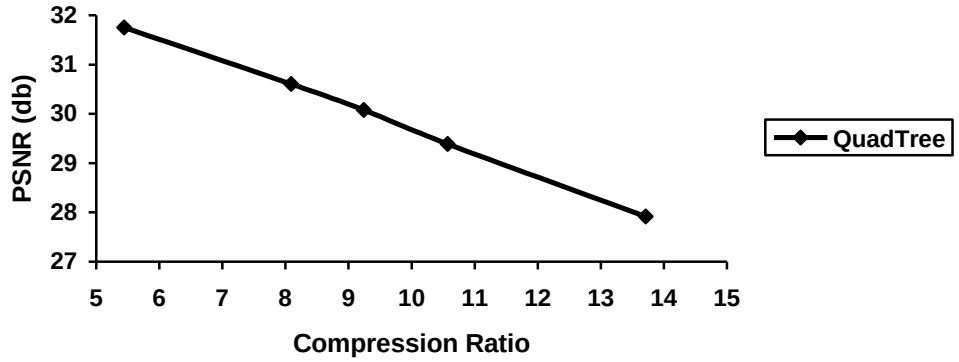

Figure (7): Compression ratio versus PSNR (when SizMin=4, SizMax=16 and R_a=0.1)

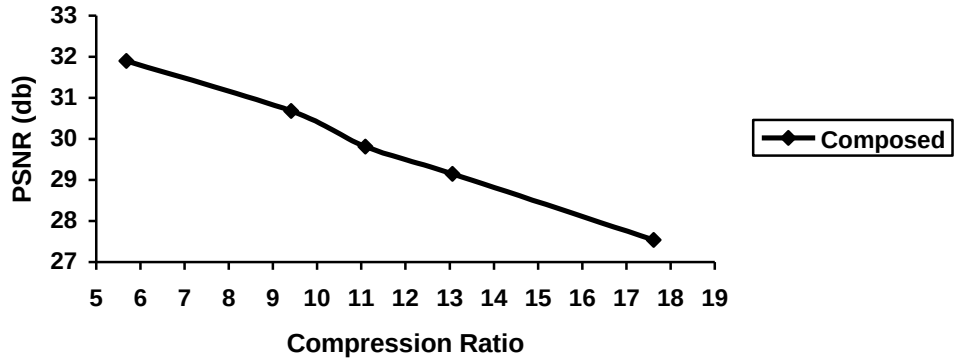

Figure (8): Compression ratio versus PSNR (when SizMin=4, SizMax=16 and R_a=0.1).

Table (1): The encoding results of FIC based on Composed Partitioning implemented on 256 x 256 Lenna gray scale image

NoBitScl=5, NoBitOfs=7, DomStp=4, Maxscl=1.2, Minscl=-1.2, Maxofs=256, Minofs=-256.									
Max. B. S.	Min. B. S.	I_f	R_a	No. Blocks	Comp. Ratio	RMSE	SNR	PSNR	E.T. (Sec.)
8	2	0.2	0.1	7061	2.66	2.72	32.80	39.46	39
8	2	0.5	0.1	3721	5.03	5.27	27.04	33.70	26
8	٢	0.7	0.1	2801	6.68	6.61	25.06	31.73	23
8	2	0.9	0.1	2092	8.92	8.21	23.17	29.85	20
8	2	1.0	0.1	1846	10.12	8.83	22.54	29.21	18
8	2	0.3	0.2	4369	4.29	4.98	28.44	35.10	29
8	2	0.6	0.2	2222	8.43	7.64	23.80	30.47	20
8	2	0.3	0.07	5568	3.37	3.37	30.93	37.58	33
8	2	0.7	0.5	1348	13.88	10.47	21.04	27.73	16
8	2	0.5	0.5	1761	10.61	8.97	22.40	29.08	18
16	2	0.7	0.1	1950	9.60	9.90	21.55	28.22	19
16	2	0.9	0.1	1244	14.99	12.07	19.81	26.50	16
16	٢	0.7	0.05	2699	6.93	7.55	23.91	30.57	22
8	4	0.7	0.1	1734	10.78	8.14	23.24	29.91	18
8	4	0.5	0.1	1996	9.37	7.46	24.00	30.67	20
8	4	0.3	0.1	2409	7.77	6.84	24.70	31.42	21
8	4	0.7	0.05	1949	9.59	7.54	23.91	30.58	20
16	4	0.7	0.1	1059	17.62	10.71	20.86	27.54	14
16	4	0.5	0.1	1431	13.06	8.89	22.48	29.15	16
16	4	0.4	0.1	1686	11.09	8.24	23.14	29.81	17
16	4	0.3	0.1	1990	9.41	7.46	24.01	30.68	18
16	4	0.1	0.1	3316	5.68	6.48	25.24	31.90	24
16	4	0.7	0.07	1257	14.85	9.55	21.86	28.53	15
16	4	0.7	0.05	1392	13.41	8.94	22.44	29.11	16
16	4	0.7	0.03	1629	11.48	8.07	23.33	29.99	17
16	4	0.7	0.01	1753	10.69	7.67	23.77	30.44	18
16	4	0.5	0.05	1747	10.70	7.87	23.54	30.21	19
32	4	0.7	0.01	1671	11.21	8.26	23.11	29.79	17
32	4	0.3	0.01	2547	7.37	6.74	24.89	31.55	21
32	4	0.5	0.05	1629	11.48	9.49	21.90	28.59	17

5. References

[1]: Schuler A., "An Introduction to fractal Image Encoding", http://inls.ucsd.edu/y/fractals/2004.

[2]: Hanna, H., "Block Indexing Technique", M.Sc. thesis College of scence, Al-Mustansiriah University, 2005.

[3]: Ghada, K., "Adaptive Fractal Image Compression", M.Sc. thesis, National computer Center /Higher Education Institute of computer and Informatics, 2001.

[4]: Jamila, H. S., "Fractal Image compression", Ph.D. thesis, College of Science, University of Baghdad, January,2001.

[5]: Razaak,H.H.,"Adaptive Fractal Image Compression", M.Sc. thesis, college of Science, AL-Mustansiriya University, 2000.

[6]: Saupe, D., Hamzaoui, R., Ruhl, M., Grandi, L., Marini, D., "OPTIMAL HIERARCHICAL PARTITIONS FOR FRACTAL IMAGE COMPRESSION", IEEE International Conference on Image Processing (ICIP'98), Chicago, second edition, 2004.

[7]: Scott, E. U., "Computer Vision and Image Processing: Practical Approach using CIVP tools ", Prentice-hall, Inc., USA, third edition, 2006.