A Proposed Encoding and Hiding Text in an Image by using Fractal Image Compression

Dr. Fadhil Salman Abed
Lecturer in University
Technical Institute of Kalar
Iraq-Diyala-Jalawla
E-Mail: Fad_Sal_Abed@Yahoo.com

Keywords
Image processing , Hiding, Fractal Image Compression, Quadtree, Steganography

Abstract

The problem of protecting data and information originates from the dependency upon encryption and steganography separately. Therefore, it is easy to break them and detecting the contents of the secret massages available in data, pictures, texts or audios.

To add multiple layers of security our research proposed an algorithm that can hide a text in an image in a such away that prevents as much as possible any suspicion having a hidden text, after RSA cipher.

In this research a new hiding method is proposed based on encoding the information by using RSA cryptosystem and hiding the cipher information(ciphertext) by using proposed fractal image compression(FIC). FIC based on IFS exploits the spatial redundancy with it is different parts and they are blocked based algorithm with variable block size that intend to approximate blocks of a determined size with contractive transformation. This blocks based allow hiding capacity to be large.

1. 0 Introduction

With the increases in the number of digital networks and recording devices, digital images appear to be a material for steganography and information hiding techniques have been developed to be a strong basis for steganography area.

The aim of this paper is to describe a method for integrating together cryptography and Steganography for secure communication using an image file. The proposed system first compresses the secret message (i.e. word document) and then implements cryptographic algorithms to the compressed message. The resulted file is used as the secret message to be hidden in the digital an image file**[1]**.

This research project produces new information hiding scheme and it consists of three parts, the first part is fractal Image Compression(FIC) which needs several stages of operations to be done on the colour image to find out the best approximate the affine transform is used to determine the necessary scale coefficient, offset coefficient and the isometric mapping needed to achieve best match between the range and domain blocks then the process of embedding the information have to be done.

The second part is implements cryptographic algorithms by using public-Key type RSA cryptosystem to the compressed message**[2].**

The third part is embedding cipher information(Plaintext) in an image. A new scheme has been produced for hiding by using the scaling and offset values of the blocks as a cover for hiding by getting secret text (character by character) and converting each character into its ASCII representation, then store them in to data array and adding the characters to the fraction part of scaling value without affecting the quality of the image. It can achieve a very high hiding rate of information.

In order to obtain the best approximate for each non-overlapped block of range, all overlapped blocks of the domain should be tested, taking into consideration that there are eight isometric possible states for each cover block and all of these possible states should be tested when trying to find the closest approximate. In another words good approximate can be obtained when many domain blocks are involved in the approximation process taking into consideration that the process of searching the pool of domain time consuming process.

2. Combination of cryptography and Steganography for secure communication

Combination of cryptography type RSA cryptosystem and Steganography by using FIC for secure communication is an application, which combines both Cryptography methods (i.e. Encryption, Decryption) and Steganography techniques to make the communication more secure. The outcome of this paper is to create a cross-platform tool that can effectively hide a message (i.e. Word document) inside a digital an image file. It is concerned with embedding information in a secure and robust manner. In this paragraph, we illustrate the cryptography and FIC (fractal Image Compression) for compress and hiding the information**[1],[3].**

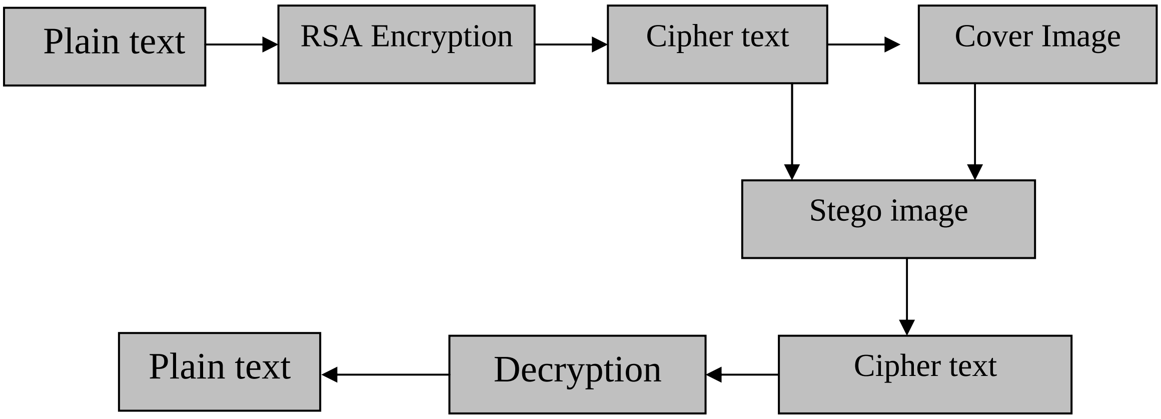


Figure (1): Combination of cryptography and Steganography

2.1 Cryptography Stage

There are several types of asymmetric algorithms used in the computing world today. They may have different internal mechanisms and methods, but the one thing they do have in common is that they are all asymmetric. This means that a key is used to encrypt a message different from the key that is used to decrypt a message. RSA is a public key algorithm that is the most understood, easiest to implement, and the most popular when it comes to asymmetric algorithm.

2.1.1 RSA Algorithm

RSA involves a public and private key. The public key can be known to everyone and it is used for encrypting messages.

The secret message in this work ends by (#) to determine the end of the message in an extracting stage. Each letter in the message is converted to a decimal number by ASCII code, and encrypted by using the public key.

The keys for the RSA algorithm are generated in the following way**[4],[5],[6]:**

- Choose two distinct large random prime numbers P and q.
- Compute $n = pq$, n is used as the modulus for both the public and private keys.
- Compute the: $\varphi(n) = (p-1)(q-1)$.
- Choose an integer e such that $1\langle e\langle \varphi(n)$, and e and $\varphi(n)$ share no factors other than 1 (i.e. e and $\varphi(n)$ are prime), e is released as the public key exponent.
- Compute d to satisfy the congruence relation $de \equiv 1(\operatorname{mod} \varphi(n))$, d is kept as the private key. The public key consists of the modulus n and the public (or encryption) exponent e. The private key consists of the modulus n and the private (or decryption) exponent d which must be kept secret.

All parts of the private key must be kept secret in this form. The decoder can recover m from c by using her private key exponent d by the following computation:

$m \equiv c^d \pmod{n}$. Given m, the decoder can recover the original message m.

This shows that we get the original message back:

$$c^d \equiv m \pmod{n}.$$

The public key is the product of two randomly selected large prime numbers, and the secret key is the two primes themselves. The algorithm encrypts data using the product, and decrypts it with the two primes, and vice versa. A mathematical description of the encryption and decryption expressions is shown below:

Encryption: $C = m^e (\bmod n)$, **Decryption:** $m = c^d (\bmod n)$

Where:

m : The plain-text message, c : the encrypted message expressed as an integer number.
n : the product of two randomly selected, large primes p and q .

e : a large, random integer relatively prime to $(p-1)*(q-1)$.

d : the multiplicative inverse of e , that is:

$$(e * d) = 1(\bmod(p-1)*(q-1)) .$$

The public key is the pair of numbers (n, e) . The private key is the pair of numbers (n, d) .

2.2 Fractal Image Compression Stage

In fractal image compression the image to be coded is partitioned into blocks called ranges. Each range is approximated by another part of the image called domain. Finding a partitioning that minimizes the approximation error while not exceeding a given bit rate is a hard problem in FIC.

The goal of utilizing image partitioning in FIC is to approximate the pixel intensities of the range block with those of the domain block. Good approximation can be obtained when many domain blocks are involved in the matching (approximation) process. Taking into consideration that the process of searching the pool of domain blocks is time consuming (greedy) process, **[7].**

2.2.1 Range Pool Generation

This module consists of two-sub modules, image loading and range partitioning.

2.2.2 Image Loading

At first the image to be coded must be loaded as a two dimensional array. After loading the array, the image data stored in this two-dimensional array will be converted to a gray level image. This array is called the "Range".

2.2.3 Range Partitioning (Range Pool)

Before the image to be coded using PIFS(Partition Iterated Function System), it must be partitional into non– overlapping block of different sizes, by using one of the adaptive partitioning methods. The resulting blocks are called "Range Blocks" (non-overlapped blocks).

In the quadtree fractal compression method, the partitioning depends on the maximum and minimum allowable block size. As a special case, quadtree partitioning becomes fixed partitioning; when the adopted maximum and minimum allowable block sizes are same.

2.2.4 Domain Pool Generation

This module is responsible for generating another two-dimensional array ($H_d \times W_d$), but with quarter the size of the range (i.e., $H_d = H/2$ and $W_d = W/2$). This array is called the “Domain”.

2.2.5 The Search

For each range block R the optimal affine approximation (mapping) can be represented as**[8]:**

$$R_i \approx sD_i + o$$

Where R_i is the range pixel value,

D_i is the corresponding domain pixel value,

s is the scaling coefficient, o is the offset coefficient.

For mapping each range block, the search process implies that for all domain blocks (D_i), listed in the domain pool, should be matched with the considered block, by computing the optimal approximation ($R_i \approx sD_i +$

o). The eight symmetric versions of the domain block should be taken into consideration. Using the following isometric mappings produces the symmetric blocks:

1.Identity: Sym (x, y)=R(x, y)

2.Rotation (+90): Sym(x, y)=R(y, BS-x)

3.Rotation (+180): Sym(x, y)=R(BS -x, BS -y)

4.Rotation (+270): Sym(x, y)=R(BS -y, x)

5. Reflection about mid-vertical axis:Sym(x, y)=R(y, x)

6. Reflection and rotation (-90): Sym(x, y)=R(x, BS -y)

7. Reflection and rotation (-180): Sym(x, y)=R(BS-y, BS -x)

8. Reflection and rotation (-270): Sym(x, y)=R(BS-x, y)

where BS is the size of the domain block, Sym is the symmetric of block .

The optimal approximation could be done as follows:

1. Compute the scale (s) and offset (o) coefficients, using the following equations:

$$s=\frac{n^2\left(\sum_{i=1}^{n} d_i r_i\right)-\left(\sum_{i=1}^{n} d_i\right)\left(\sum_{i=1}^{n} r_i\right)}{n^2\left(\sum_{i=1}^{n} d_i^2\right)-\left(\sum_{i=1}^{n} d_i\right)^2}\ ,\quad o=\frac{1}{n^2}\left(\sum_{i=1}^{n} r_i\ -s\sum_{i=1}^{n} d_i\right)$$

2. Check the coefficients:
   a. for scaling coefficient:
      if $s<s_{min}$ then $s= s_{min}$ else if $s>s_{max}$ then $s=s_{max}$
   b. for offset coefficient:
      if $o<o_{min}$ then $o= o_{min}$ else if $o>o_{max}$ then $o=o_{max}$

3. Quantization

$$s_q=\mathrm{round}\left(\frac{2^{b_s}-1}{s_{max}-s_{min}}(s-s_{min})\right),\quad o_q=\mathrm{round}\left(\frac{2^{b_o}-1}{o_{max}-o_{min}}(o-o_{min})\right)$$

where:

b_s: number of bits assigned to scaling coefficients, b_o number of bits assigned to offset coefficients,

s_{max}: maximum value of scaling coefficients, s_{min}: minimum value of scaling coefficients,

o_{max}: maximum value of offset coefficients, o_{min} : minimum value of offset coefficients .

4. Compute the approximation error (d_{RMS}) using equation :

$$E^2(R,D)=\frac{1}{n^2}\left[\sum_{i=1}^{n} r_i^2+s\left(s\sum_{i=1}^{n} d_i^2-2\sum_{i=1}^{n} d_i r_i+2o\sum_{i=1}^{n} d_i\right)+o\left(on^2-2\sum_{i=1}^{n} r_i\right)\right]$$

5. Compare the computed error with the minimum registered error (d_{min}) i.e., if $d_{RMS}>d_{min}$ then jump to step (7).

6. Register

$s_{opt}=s_q$, $o_{opt}=o_q$, $d_{min}= d_{RMS}$,S_m=Symmetry index,

x_d = x (the coordinate of the current tested domain block),

y_d = y (the coordinate of the current tested domain block).

7. Repeat the steps (1) to (5) for all symmetry versions of the tested domain blocks.

8. Repeat the steps (1) to (7) for all domain blocks listed in the domain pool.

9. Output the set of parameters (s_{opt}, o_{opt}, x_d , y_d and S_m) as a set of fractal coding parameters for the tested range block

3.0 The Proposed Hiding and Encoding System

Figure (2) describes the proposed system starting with Encoding Text by Using RSA Cryptography and loading cover image, then performing the quad-tree partition and then hiding module by starting search process for similarity between the image blocks (range blocks and domain blocks) and embedding the secret message in the scaling and offset values of the blocks. The output of this stage is a data file of stego-image which is sent from a sender to a recipient. When the recipient receives the data file of the stego-image, the process of extraction could be applied to obtain the secret message with an approximate or the reconstructed image.

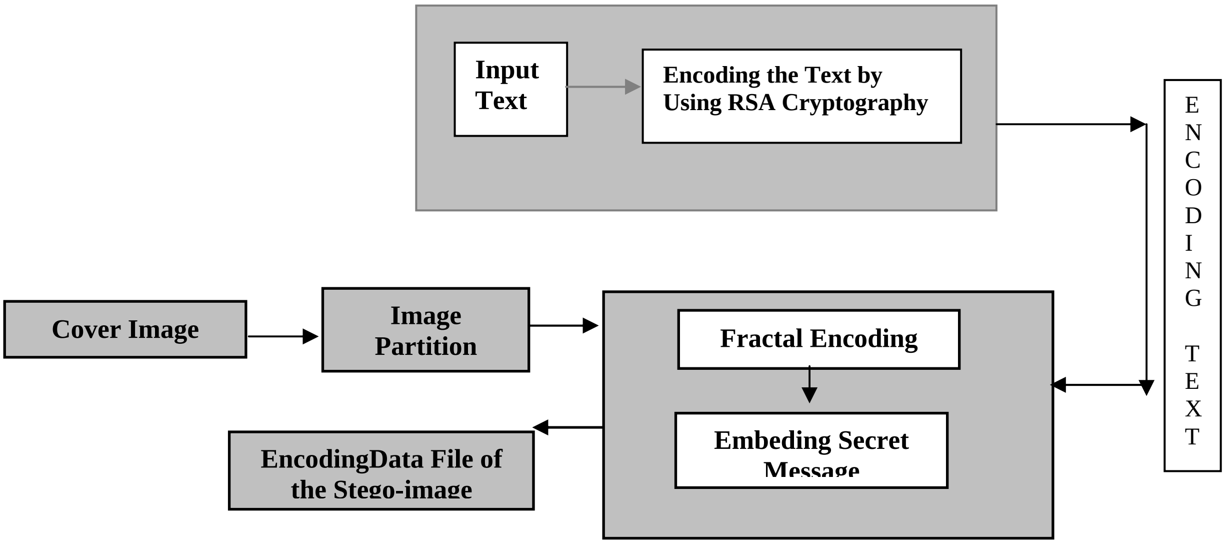


Figure (2): Embedding Information Unit

3.1 Secret Message Embedding

For embedding the secret massage, a new method has proposed and in this new method a huge number of characters can be embedded, beside the characters or the secret message there may not necessary be a text, it may consist of equation or numbers with another language**[9].** After the IFS mapping is coming to the end of the last block and the parameter values have been set, the process of embedding starts by reading the inputted secret message and converting it to its binary representation then store them in a new array separately.

Next, the secret message (SecData) characters are taken one by one and they are converted to its ASCII representation. The length of the secret message is limited to the number of blocks. As the number of blocks increase in tern more characters can be embed, in other words this embedding method depends upon the number of blocks.

Length of secret message (SecData) = (No. of blocks ×2) -1

Each character is embedded in the scaling value of the blocks by taking the integer part of the scaling using this equation.

The Proposed Hiding and Encoding System include the following steps:-

1. Load a colour image (bitmap format 24 bits), and part the colours into the red, green, and blue.
2. Convert the image formula from RGB to YCbCr**[10]**

   $Y = (77/256)\ R + (150/256)\ G + (29/256)\ B$

Cb = -(44/256) R – (87/256) G + (131/256) B + 128
Cr = (131/256) R – (110/256) G – (21/256) B +128

3. Partitioning the cover image by using Quad-tree Partition the algorithm illustrated In **figure (8) in appendix**.

4. V= IFS(I). Scl – Fix (IFS(I).Scl)

Then taking the two digit of the fraction part (2 digit after the decimal point) of the value and neglecting the other digits

5. *IntFraction = fix(V×100)*
6. Fraction =IntFraction ×0.01

Now the secret message will be added to the fraction value in order to occupy the **3rd, 4th** and **5th** places after the decimal point without affecting the **1st** and **2nd** places of the original fraction.

7. Encoding(Cryptography) the input message by using RSA cryptosystem, the output is called Enc***SecData(I).***

8. Calculate Iterated Function System(IFS)
IFS(I). Scl = Fix(IFS(I).Scl) + Fraction + Enc***SecData(I)×0.00001***

The output is a set of scale and offset values that contain the secret massage (SecData) is the last stage in hiding unit.

3.3 Information Extraction

After the data file (stego-image) has been loaded, the process of reconstructing SecData is applied to extract the array of embedded secret characters, which have been stored in the (IFS) coefficients (s, o) in a reverse way. This stage implies the following steps:

1. Extract the two digit of the fraction part of the coefficient (s and o) with keeping the integer part.
V= IFS(I).scl – Fix (IFS(I).scl)
Vs = v ×1oo

2. Decoding the secret massage by using RSA decoding Algorithm

3. Convert the extracted data to byte

Dec***SecData(I) = CByte((Vs – Fix(Vs)) ×1000***

4. Convert the bytes to string representation

recSecData = recSecData & CStr(Chr(Dec***SecData(I)))***

5. Display the secret message.

3.4 Saving Reconstructed Image

When the completion of all the above decoding steps, the secret message reconstructed as an output beside the reconstructed cover image.

3.5 System Implementation

The goal of this system is to Encoding Text by Using RSA Cryptography and hiding information (text, numbers, symbols or equations) in a cover-image (BMP format) after compressing the image to produce the stego-image as a data file.

System implementation accepts seven inputs in the embedding stage**:**

1. Loading the cover image (BMP.format) as the input file.
2. Input control parameters.
3. Quad-tree partitioning the colour component.
4. Domain generating.

5. Inputting the secret message for embedding.
6. Encoding secret message to be embedding by using RSA cryptosystem.
7. Fractal encoding which include embedding.

System implementation accepts one input in the extracting stage:

1. Input the data file which contains the secret message beside image data array.
2. Extracting the secret message and reconstructing the image in the same time.

3.6 System Requirements

The Microsoft Window XP has been used as an operation system and Visual Basic (VB6) as a programming language.

3.7 System Steps

The proposed system starts operation with the general form interface of the project

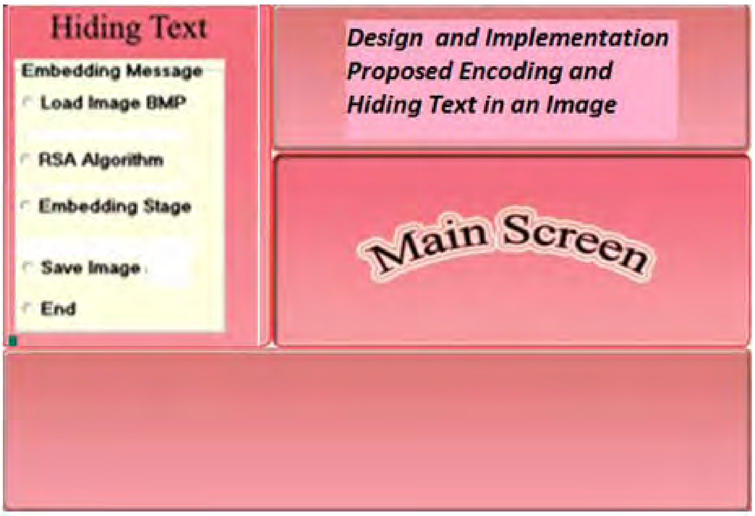


Figure (3): Starting a choice form

- **Partitioning the Cover Image**

After the cover-image has been chosen, control parameter will be entered to perform quad-tree partition for each colour component (R component, G component, B component) separately**, Flowchart (1) in Appendix** illustrates the quadtree partitioning procedure.

- **Generating the Domain Image**

Generating the domain image and domain pool is next to the partition step, the domain size taken is quarter the image size with overlapped blocks.

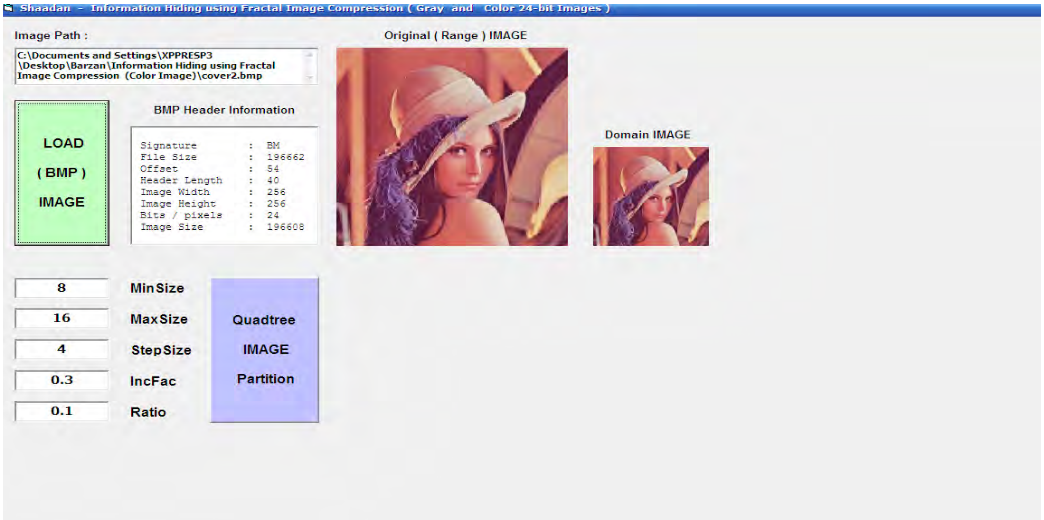

Shaadan - Information Hiding using Fractal Image Compression (Gray and Color 24-bit Images)

Image Path : C:\Documents and Settings\XPPRESP3\Desktop\Barzan\Information Hiding using Fractal Image Compression (Color Image)\cover2.bmp

BMP Header Information

```
Signature      :  BM
File Size      :  196662
Offset         :  54
Header Length  :  40
Image Width    :  256
Image Height   :  256
Bits / pixels  :  24
Image Size     :  196608
```

8 MinSize
16 MaxSize
4 StepSize
0.3 IncFac
0.1 Ratio

Figure (4): Domain Image Generating

- **Input Secret Message**

Before starting fractal encoding, the message must be written in order to be embedded after the scale and offset values will be stored**.**

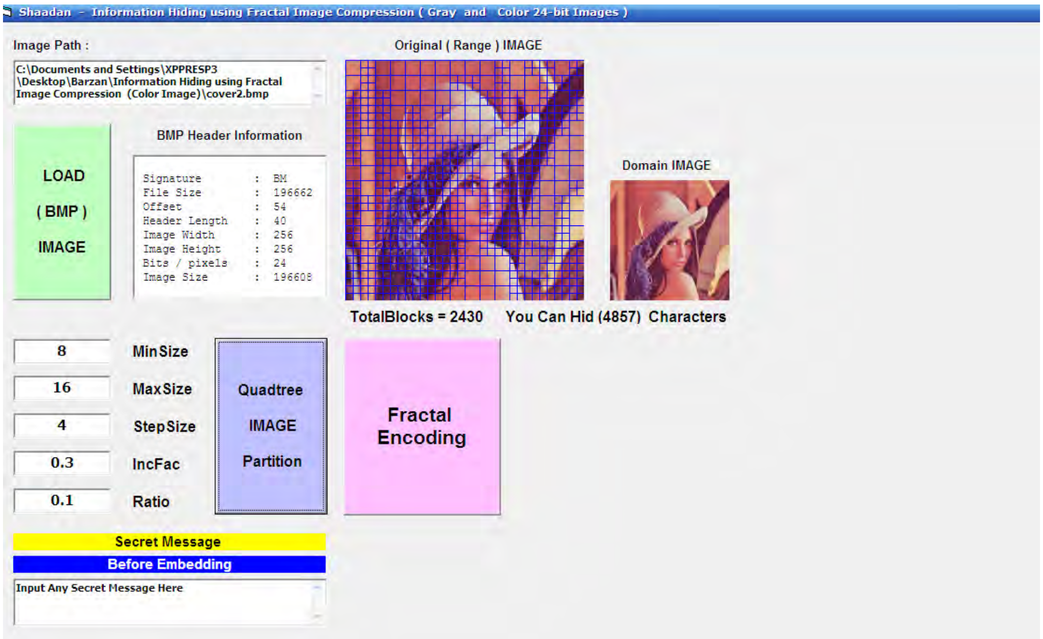

Shaadan - Information Hiding using Fractal Image Compression (Gray and Color 24-bit Images)

Image Path : C:\Documents and Settings\XPPRESP3\Desktop\Barzan\Information Hiding using Fractal Image Compression (Color Image)\cover2.bmp

BMP Header Information

```
Signature      :  BM
File Size      :  196662
Offset         :  54
Header Length  :  40
Image Width    :  256
Image Height   :  256
Bits / pixels  :  24
Image Size     :  196608
```

TotalBlocks = 2430 You Can Hid (4857) Characters

8 MinSize
16 MaxSize
4 StepSize
0.3 IncFac
0.1 Ratio

Secret Message
Before Embedding
Input Any Secret Message Here

Figure (5): Entering the Secret Message for Embedding

- **Fractal Image Compression**

Searching for similarity is performed between the range and the domain blocks and the information is stored in an index, then the image (cover-image) information is stored as a structure array of data containing the secret message.

The received data is a collection of data that represent the image with the secret message. The receiver will extract the embedded information (secret message) and then reconstruct the cover image.

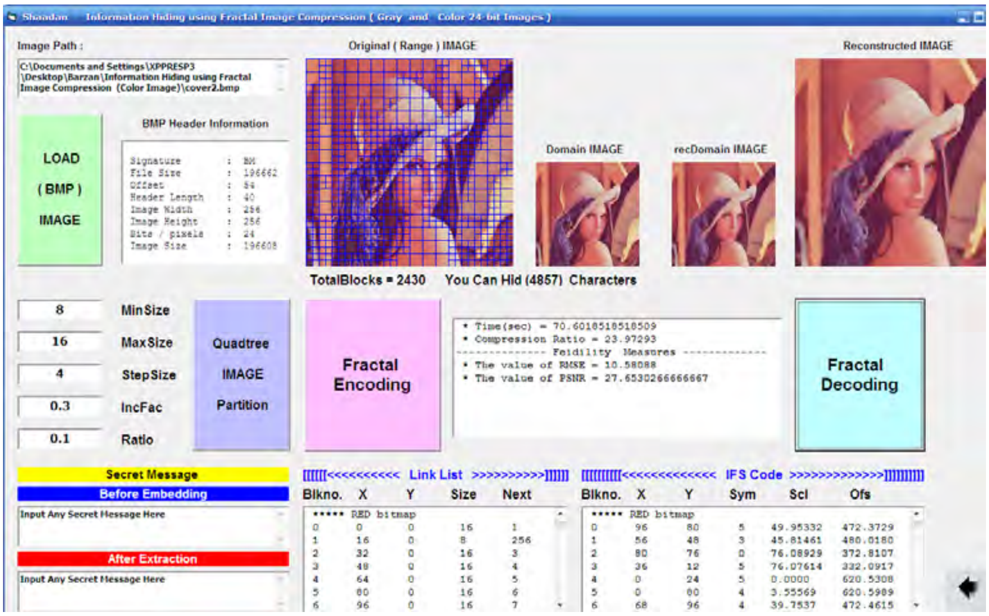


Image Path :

C:\Documents and Settings\XPPRESP3\Desktop\Barzan\Information Hiding using Fractal Image Compression (Color Image)\cover2.bmp

BMP Header Information

```
Signature        :  BM
File Size        :  196662
Offset           :  54
Header Length    :  40
Image Width      :  256
Image Height     :  256
Bits / pixels    :  24
Image Size       :  196608
```

TotalBlocks = 2430   You Can Hid (4857) Characters

8	MinSize
16	MaxSize
4	StepSize
0.3	IncFac
0.1	Ratio

```
• Time(sec) = 70.6018518518509
• Compression Ratio = 23.97293
-------------- Feidility  Measures ------------
• The value of RMSE = 10.58088
• The value of PSNR = 27.6530266666667
```

Input Any Secret Message Here

Input Any Secret Message Here

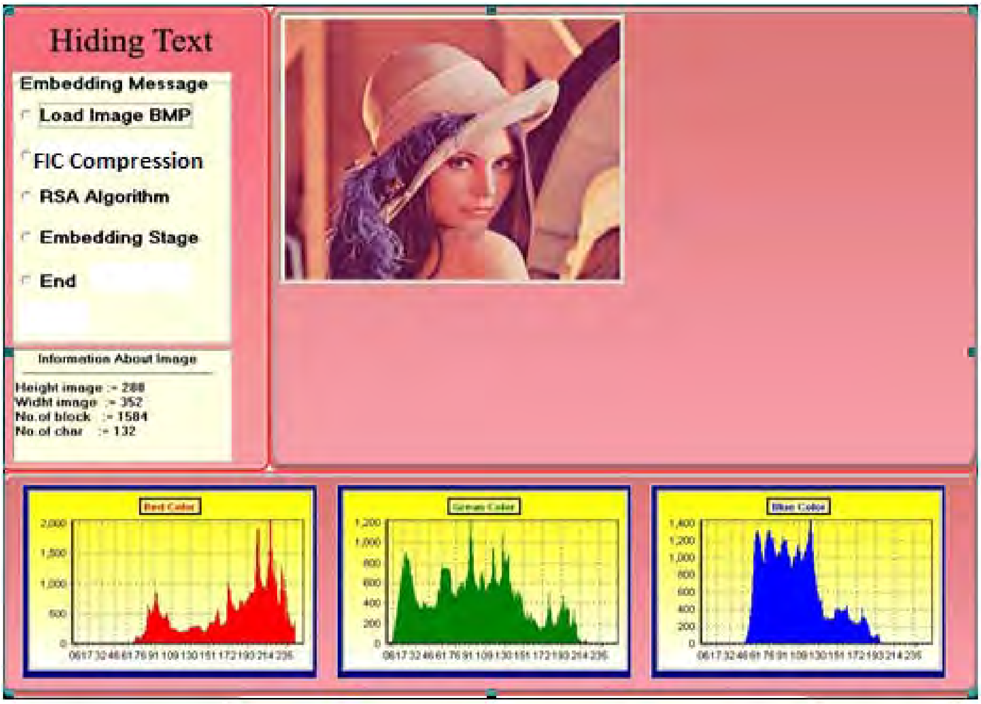
[[[[[<<<<<<<<<< Link List >>>>>>>>>>]]]]]

```
Blkno.   X     Y     Size   Next
***** RED bitmap
0        0     0     16     1
1        16    0     8      256
2        32    0     16     3
3        48    0     16     4
4        64    0     16     5
5        80    0     16     6
6        96    0     16     7
```

[[[[[[[[[<<<<<<<<<<<<< IFS Code >>>>>>>>>>>>>]]]]]]]]]

```
Blkno.   X     Y     Sym   Scl         Ofs
***** RED bitmap
0        96    80    5     49.95332    472.3729
1        56    48    3     45.81461    480.0180
2        80    76    0     76.08929    372.8107
3        36    12    5     76.07614    332.0917
4        0     24    5     0.0000      620.5308
5        0     80    4     3.55569     620.5989
6        68    96    4     39.7537     472.4615
```

Figure (6) Secret message extraction and cover image reconstruction



Information About Image

Height image := 288
Widht image := 352
No.of block := 1584
No.of char := 132

Figure (7): Cover-Image

3.7 The Effect of Block Size

In these sets of tests it is focused on finding out how the quad-tree partitioning is affected when different values of control parameters are adopted, Table (1) shown that the number of blocks produced by quad-tree partition depends on the adopted partitioning parameters (maximum block size, minimum block size, inclusion factor and acceptance ratio). For hiding the secret message, the number of blocks determine the length of the secret message, for obtaining the large number of blocks small values must be chosen which mean more characters will be hidden (according to the number of blocks).

The increase of the number of blocks increase the encoding time and PSNR but decrease the RMSE also the compression ratio.

Table (1:)The compression results of hiding information using FIC method based on quad-tree partitioning applied on Lena image

Public key is given by (n,e)= (6012707, 3674911); Private key is given by (n, d) = (6012707, 422191) MaxSize = 8, MinSize = 4 and StepSize = 4

Ar	**I_f**	No.blocks	No.Char	Time(sec)	C.R.	RMSE	PSNR(db)
0.1	0.5	6489	12975	105.992	8.977	7.689	30.452
0.2	0.5	5550	11097	98.705	10.496	7.988	30.084
0.3	0.7	4251	8499	93.218	13.703	8.517	29.527
0.4	0.7	4008	8013	90.361	14.534	8.602	29.441

3.8 The Effect of Hiding Secret Message

The main objective of the proposed encoding and hiding scheme is to embed a secret message with a huge number of characters as possible with different languages and numbers without degrading the quality of the reconstructed cover image. So to evaluate the effect of the secret message embedding on the cover image, a set of tests is applied. Table (2) show the result of hiding different message process.

Table(2): Hiding effect on Lena image

Public key is given by (n,e)= (6012707, 3674911); Private key is given by (n, d) = (6012707, 422191 MaxSize=8, MinSize=4, StepSize=4, R_a = 0.1, I_f =0.3, PSNR1= Stegoimage fidelity,PSNR2= After extraction fidelity, PSNR3= Without embedding

N0.char	Type	Time(sec)	RMSE	PSNR1(db)	PSNR2(db)	PSNR3(db)
9653	Englishtext	113.537037	7.1480366	31.05610	31.05606	31.06429
10283	Arabictext	113.537037	7.1481766	31.69444	31.05585	31.06429
11390	Mixed	114.69444	7.1404833	31.06484	31.06464	31.06424

4.0 Conclusions

During the implementation and testing phases of the proposed encoding and hiding system, the following remarks have been addressed:

1. Steganography is an effective way to obscure data and hide sensitive information. The effectiveness of Steganography is amplified by combining it with cryptography. By using the properties of the FIC Steganography algorithm for image file and combining it with the RSA cryptography standards, we developed a method, which adds layers of security to the communication. Steganographic methods do not intended to replace cryptography but supplement it.

2. The proposed system is build of muti-layer security ,consequently even if the original copy of the stego-object is available ,the intruder attacked by the second layer of security which is the encryption of the text embedded then he should analyze the cipher text using cipher text only attack . Which difficult to be analyzed.

3. The proposed system first compresses the secret message (i.e. word document) and then implements cryptographic algorithms to the compressed message. The resulted file is used as the secret message to be hidden in the digital image file.
4. The proposed system does not affect the image resolution; we can say it is not noticeable for human eyes. To prove this we show the cover-image and the stage –image to a team of 20 persons to take their opinion if

there is any different between the sego-image and the cover-image and their answer that there is no difference between both images.

References

[1] Sharone Gorla," Combination of Cryptography and Steganography for Secure Communication in Video File ", *Master Thesis in computer scince*, California state university, Sacramento,2009.

[2] Mohammad Ahmad A. Alia," A New Approach to Public-Key Cryptosystem based on Mandelbrot and Julia Fractal sets", University Sains Malaysia, 2008.

[3] Nadia M.G. Al-Saidi and Muhammad Rushdan Md. Said," A New Approach in Cryptographic Systems Using Fractal Image Coding",Journal of Mathematics and Statistics 5 (3):183-189, 2009.

[4] Nada Abdul Aziz Mustafa," Design and Implementation proposed Encoding and Hiding Text in an Image", University of Sulaimani, Ms.c Thesis, 2010.

[5] Mohammad Ahmad A. Alia," A New Approach to Public-Key Cryptosystem based on Mandelbrot and Julia Fractal sets", University Sains Malaysia, 2008.

[6] Mohammad Ahmad Alia and Azman Bin Samsudin," A New Digital Signature Scheme Based on Mandelbrot and Julia Fractal Sets", American Journal of Applied Sciences 4 (11): 848-856, 2007.

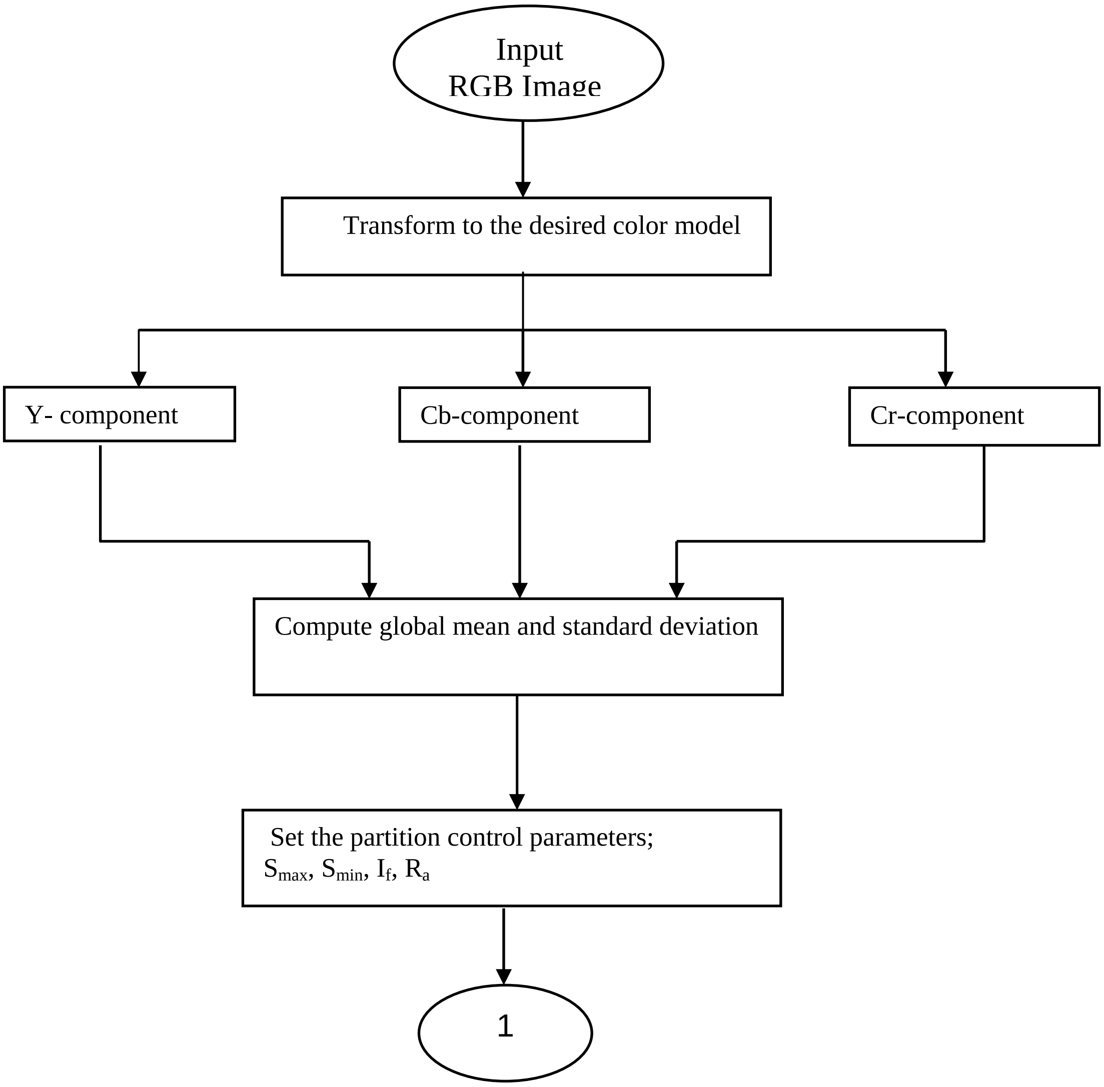
[7] Y. Fisher, "Fractal Image Compression: Theory and Application", Springer-Verlag, New York, NY, USA, 1995.

[8] Fadhil Salman Abed,"Adaptive Fractal Image Compression", Ph.d Thesis, Al-Rasheed College of Engineering and Science, University of Technology, 2004.

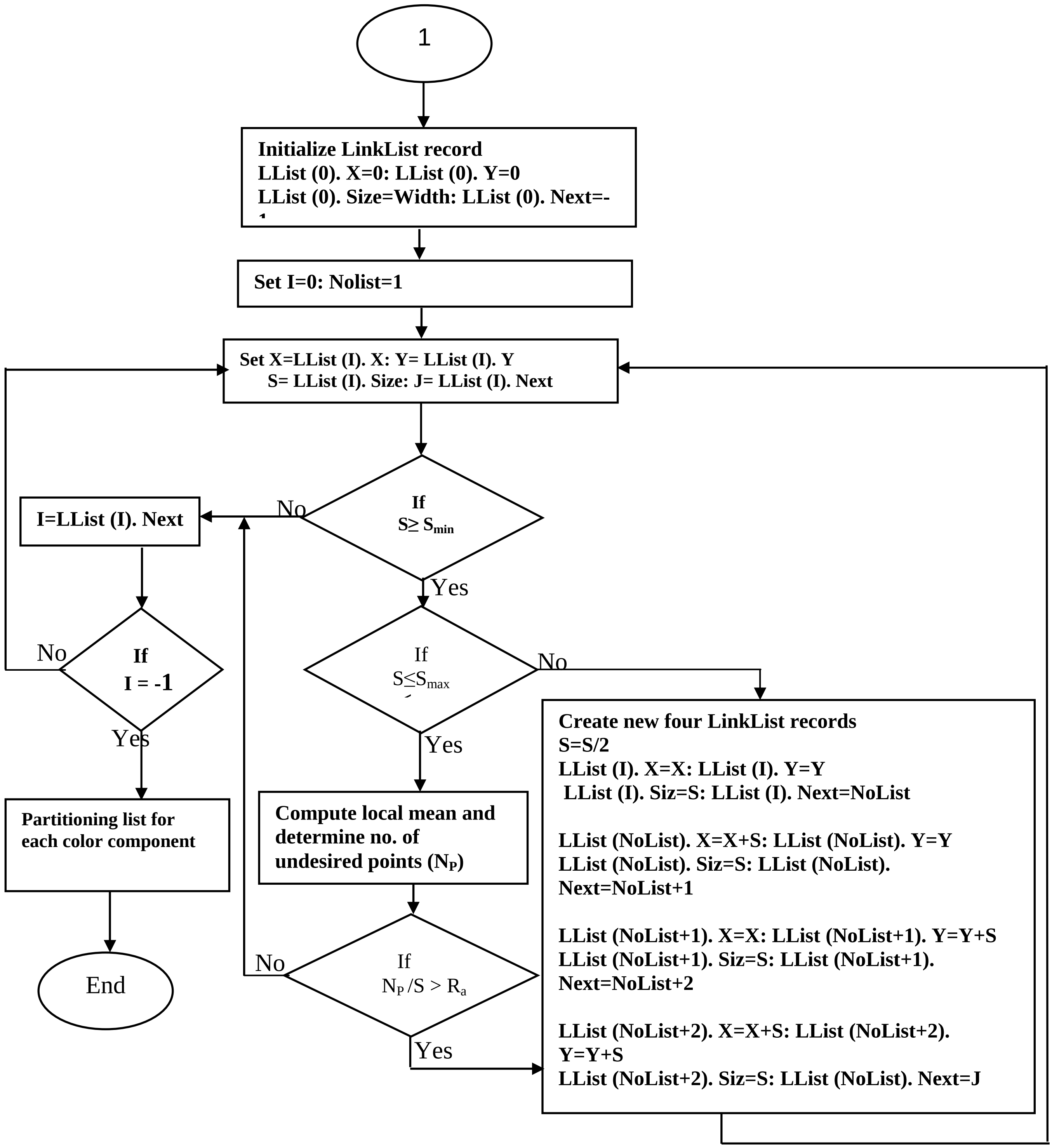
[9] Manoj Kumar Meena, Shiv Kumar, Neetesh Gupta," Image Steganography tool using Adaptive Encoding Approach to maximize Image hiding capacity", International Journal of Soft Computing and Engineering (IJSCE) , Volume-1, Issue-2, May 2011.

[10] Saad Al-Ani," New Approach for Hiding Data in Colored Images", *International Journal of Computer Applications (0975 – 8887),Volume 4 – No.7, July 201 0.*

Appendix


Input
RGB Image
Transform to the desired color model
Y- component
Cb-component
Cr-component
Compute global mean and standard deviation
Set the partition control parameters;
S_{max}, S_{min}, I_f, R_a
1

Flowchart (1): illustrates the quadtree partitioning procedure


1
Initialize LinkList record
LList (0). X=0: LList (0). Y=0
LList (0). Size=Width: LList (0). Next=-1
Set I=0: Nolist=1
Set X=LList (I). X: Y= LList (I). Y
S= LList (I). Size: J= LList (I). Next
If
S≥ S_min
No
Yes
I=LList (I). Next
If
I = -1
No
Yes
If
S≤S_max
No
Yes
Partitioning list for each color component
End
Compute local mean and determine no. of undesired points (N_P)
If
N_P /S > R_a
No
Yes
Create new four LinkList records
S=S/2
LList (I). X=X: LList (I). Y=Y
LList (I). Siz=S: LList (I). Next=NoList
LList (NoList). X=X+S: LList (NoList). Y=Y
LList (NoList). Siz=S: LList (NoList). Next=NoList+1
LList (NoList+1). X=X: LList (NoList+1). Y=Y+S
LList (NoList+1). Siz=S: LList (NoList+1). Next=NoList+2
LList (NoList+2). X=X+S: LList (NoList+2). Y=Y+S
LList (NoList+2). Siz=S: LList (NoList). Next=J

Continuation of Flowchart (1)