

# Fractal Image Compression Applied to Remote Sensing

B. Sankaragomathi, L. Ganesan, and S. Arumugam

**Abstract**—Remote sensing images contain huge amount of geographical information and reflect the complexity of geographical features and spatial structures. Fractal geometry provides a means of describing and analyzing the complexity of different geographical features in remotely sensed images. It also provides a more powerful tool to compress the remote sensing data than traditional methods. In this study, based on fractal concepts, compression and decompression algorithms were developed and applied to Landsat TM images of eight study areas with different land cover types; the fidelity and efficiency of the algorithms and their relationship with the spatial complexity of the images were evaluated. Three research hypotheses were tested and the fractal compression was compared with commonly used compression methods. The effects of spatial complexity and pixel resolution on the compression rate were also examined.

**Keywords**—Fractal image compression, Iterated Function Systems (IFS), Partitioned Iterated Function Systems, Remote sensing.

## I. INTRODUCTION

THE rapid development of remote sensing technology has provided the capability to generate data at a far greater rate than data can be analyzed and used. In fact, the development of sensor, data receiving, and data storage capabilities has proceeded faster than the development of practical applications. Often, those who could benefit most from the technology do not have the information or the context to assess this information. For this reason, remote sensing has been given a more prominent treatment than other data sources commonly used in geographic information systems (Star et al. 1997).

Based on the three key concepts described in the three theorems, the fractal image compression and decompression algorithms are developed in C code and implemented in this study. The image compression is named fractal because the method used to encode images shares many features in common with fractal generating algorithms (Fisher 1992).

In this study, both fractal image compressing and decompressing algorithms will be applied on images reflecting different landscapes that have different spatial complexities.

B. Sankaragomathi is with the National Engineering College, Electronics and Instrumentation Engg Department, Kovilpatti Pin-628 503, Tamil Nadu (corresponding author to provide phone: +91-4632-222502, e-mail: sankaragomathi.paramasivan@gmail.com).

Dr. L. Ganesan is with the Professor and Head of CSE Department, A.C. College of Engg&Tech, Karaikudi.

Dr. S. Arumugam is with the Senior Professor (Research), AKCE, Srivilliputhur, Tamil Nadu, India.

Some images may be smooth, e.g., water bodies; some may be rough, as hybrid vegetation; and some may exhibit more structural information, as urban areas. The results, including computational time and compression rates, may be affected by geographic complexities. It is an important part of the study to analyze the effects of different landscapes on the compression rate and image fidelity. The relationship between the landscapes and the image compression rates will be more clearly revealed, moreover, the effect of changing pixel resolution on image compression and how the scale affects change with image complexity can be understood.

## II. FRACTAL IMAGE COMPRESSION

The task of compressing an image includes three important parts:

- 1) Partition the image and find transformations for each partitioned part;
- 2) Encoding (compressing) the image; and
- 3) Decoding (decompressing) the image.

### A. Partitioning Images

For the issue of fractal image compression, we choose the quad tree method to partition images because of its orderliness, simplicity, and efficiency.

#### 1. Quad tree Partition

Quad tree is an image structure, which appeared at the end of the 1970s, and was developed and applied in the 1980s and the 1990s. The root of the tree is the initial image. First, the image is divided into domains with different sizes using the quadtree method and the domain pool is built, including sizes and positions of the quadtree partitions are prepared. The squares at the nodes are compared with domains in the domain pool (or domain library)  $D$ , which are twice the range size. Nodes are compared with domains in the domain pool (or domain library)  $D$ , which are twice the range size. The pixels in the domain are averaged in groups of four so that the domain is reduced to the size of the range, and the affine transformation of the pixel values is found that minimizes the rms difference between the transformed domain pixel values and the range pixel values. All the potential domains are compared with a range. If the resulting optimal rms value is above a pre-selected threshold and if the depth of the quadtree is less than a preselected maximum depth, then the range square is subdivided into four quadrants, and the process is repeated. If the rms value is below the threshold, the optimal domain and the affine transformation on the pixel values are

stored. Thus one map  $w_i$  is found. The collection of all such maps  $W = \bigcup w_i$  constitutes the encoding.

### B. Determining Parameters

Once the image is partitioned, we need to determine the transformation coefficients for each partition. Given two squares containing  $n$  pixels intensities,  $a_1, \dots, a_n$  (from  $D_i$ ) and  $b_1, \dots, b_n$  (from  $R_i$ ). We can seek  $S$  and  $O$  to minimize the quantity

$$R = \sum_{i=1}^n (s \cdot a_i + o - b_i)^2 \quad (1)$$

$$S = [n^2 (\sum_{i=1}^n a_i b_i) - (\sum_{i=1}^n a_i)(\sum_{i=1}^n b_i)] / [n^2 \sum_{i=1}^n a_i^2 - (\sum_{i=1}^n a_i)^2] \quad (2)$$

$$O = [\sum_{i=1}^n b_i - s \sum_{i=1}^n a_i] / n^2 \quad (3)$$

### C. Encoding (Compressing) Algorithm

The steps of the encoding (compressing) procedure are as follows:

- 1) Determine the parameters for compressing:
  - i) Image name, image size, minimum partition exponent, maximum partition exponent both of which determine the size of domains and ranges), tolerance for fidelity, e.g., an image of size 256 x 256, 4 (corresponding to 16 x 16 blocks), 6 (corresponding to 4x4 blocks), 0 (corresponding to tolerance as zero).
- 2) Read the image to be compressed.
- 3) Process 'domains'
  - a. Scale the image by calculating the average values of each four-pixel group, then save the calculated values into an array 'domain'.
  - b. Divide the image (in 'domain') into overlapping domains (16x16 or 8x8).
  - c. Divide each domain block into four quadrants and calculate the variance of each quadrant.
  - d. Classify the domains into 24 classes according to the order of the each variances of the quadrants of the domain blocks. Record the position, the size and the class of the domain blocks in the corresponding class chain.
  - e. After processing the 16x16 domains, the procedure is repeated until you reach the smallest domains (4 x 4) as specified by the maximum partition exponent.
- 4) Record the parameters into the output file including image size, maximum Partition, minimum partition exponents, and maximum and minimum values of the image.
- 5) Process 'ranges'
  - a. Partition the original image into 'ranges' according to the 2DRE quadtree method. If the minimum partition is not reached: go ahead to continue the partition until The minimum partition is reached (e.g. 16x16 range).
  - b. Classify the range according to the same rule as did for domains, i.e. 24 classes based on the order of the variances of the quadrants of the block.

- c. Search the domain class chains to find the domain which matches the current range best by calculating the RMSE between the domain and range as  $Y - AX + B$  ( $Y$  corresponds to the values in the range,  $X$  the values in the domain). Record the position and the Rotating factor (if the domain has a different class than the ranges) of the domain with the smallest RMSE,  $A$  and  $B$ .
- d. Check a) if the smallest RMSE is less or equal to the tolerance and
  - b) if the maximum partition is reached.
    - i. If both are 'Yes', go to (5);
    - ii. If a) is 'Yes' and b) is 'No', go to (5);
    - iii. If a) is 'No' and b) is 'Yes', go to (5);
    - iv. If both are 'No', put a bit '1' as a flag into the output file, then
    - v. Divide the current range into 4 sub ranges according to the quadtree method, then go to (2).
  - e. Write  $A$ ,  $B$ , the rotating factor, the position of the domain with smallest RMSE, into the output compressed file.
- 6) Calculate the compression rate: the number of bytes of the Original image divided by the number of bytes in the output compressed file.

An encoding of an image consists of the following data:

- The final quadtree partition of the image
- The scaling and offset values  $S_i$  and  $O_i$  for each range
- For each range, a domain that is mapped to it
- The symmetry operation (orientation) used to map the domain pixels onto the range pixels.

### D. Decoding (Decompressing) Algorithm

Decoding an image consists of iterating  $W$  from any initial image. The quadtree partition is used to determine all the ranges in the image. For each range  $R_i$ , the domain  $D_i$ , which maps to it is shrunk by two in each dimension by averaging non-overlapping groups of 2 x 2 pixels. The shrunken domain pixel values are then multiplied by  $s_i$ , added to  $o_i$ , and placed in the location in the range determined by the orientation information. This constitutes a decoding iteration. The step is iterated until the fixed point is approximated, that is, until further iteration does not change the image or until the change is below some small threshold value.

The steps of the decoding (decompressing or restoring) procedure are as follows:

- 1) Determine the parameters of the restoring procedure:
  - a) Hypothetical image (the "initial image") name,
  - b) Compressed file name and
  - c) Number of iteration.
- 2) Read the input file (the compressed file).
- 3) Read the hypothetical image into an array 'image', which will be used as the domains. The hypothetical image can have any values (e.g. all 0's, or all 1's, etc.).
- 4) Get the parameters of the compressed image from the input file:

- a) Image size,
  - b) Maximum partition,
  - c) Minimum partition exponents, and
  - d) Maximum and minimum values of the original image.
- 5) Prepare a blank image 'image1' for the restored image.
- 6) Read the transformations from the input file:
- a) Set 'level'=0.
  - b) Checking level is less/more than minimum partition.
    - i) If 'Yes', divide the blank image block into 4 sub-blocks, with level +1, put sub-blocks into a queue, then go to (3);
    - ii) If 'No', i.e. level is equal to minimum partition, go to (4).
  - c) Check the queue. If it is not empty:
    - i) Obtain the first one and remove it from the queue,
    - ii) Then go to (2); if it is empty, go to (7).
  - d) Checking if 'level' is less than maximum partition and the flag bit '1' in the input file.
    - i) If both are 'Yes', divide the current block into 4 sub-blocks
    - ii) With level +1, put them into the queue, then go to (3);
    - iii) If a) or b) or both are 'No', i.e. either the maximum partition is reached or good transformation is encountered, then go to (4).
    - iv) Get the transformation parameters from the input file: A, B, rotating factor and position of the domain, then put them and the corresponding positions of the restored image into a transformation chain.
    - v) go to (3)
    - vi) Finish the procedure starting from step (6).
- 7) Do the transformations:
- a) Set a counter=0;
  - b) Check if counter is greater than the predetermined iteration number.
    - If 'Yes', go to (8);
    - If 'No', go to (3).
  - c) Check the transformation chain.
    - if it is empty, counter+1, then go to (b);
    - if it is not empty, go to (d).
  - d) Get the parameters in the transformation chain, calculating the pixel values for the restored image by scaling the hypothetical image and using  $Y=AX+B$ , if necessary, rotating it (A, B and rotating factor are all in the transformation chain, Y represents the restored value, and X the 'initial value' in hypothetical image), then put the pixel values into 'image1'.
  - e) go to (c).
  - f) Copy 'image1' to 'image'. The later will be the 'hypothetical image' or the domain of the next iteration.
  - g) Counter+1, then go to (2).
  - h) Finish the iteration.
- 8) Post-process the restored image by using maximum and minimum values of the original image.
- 9) write 'image' into output file.

### III. EXPERIMENTAL RESULTS

#### A. Analysis and Hypothesis Testing

To evaluate the fractal image compression and decompression methods on the remote sensing images, the study first computes the performance measure, then evaluates the three research hypothesis listed.

#### 1. Performance Measures

These measures will be computed and evaluated. They are:

##### i) Image fidelity

It is the most important criterion for evaluating the performance of the method. The classification method of the image will be used to test if useful information in the image is kept or lost. Quantitative comparison testing percentage of matching with the classification result of the original image will also be made to determine if the fidelity is preserved.

##### ii) Compression rate

Compression rate is the main standard for evaluating the compression algorithm since the goal of the procedure is to find an accurate and efficient method to solve the storage problem. The analysis also includes a comparison of the compression rates for different land covers, such as vegetation, water body, water/land boundary, and urban area to test how different landscapes affect the rate.

##### iii) Time complexity of the encoding/decoding procedure

The computational efficiency is essential for image compression and decompression. If the encoding/decoding procedure is too time consuming, the value of the algorithm, or the improvement of the compression rate may not be meaningful because it may not be practical or operational.

Three research hypotheses were also tested.

- i. For testing the research hypotheses 1, the results of fractal compression method are compared with the other two commonly used data compression methods.
- ii. For the hypothesis 2, the effects of spatial complexity to the compression result is also tested; besides the implementation of the algorithms.
- iii. Finally, for hypotheses 3, the effects of pixel resolution on the compression results are analyzed.

The results include the comparisons of the original and reconstructed images; fidelity evaluation; analysis of compression rates on different land covers and different resolutions; and discussion on the efficiency of the algorithms, as well as a comprehensive evaluation of the research method in detail. These descriptive statistics show that the original and the restored images are very similar. Tables I display the compression rates of each image, with compression and decompression times. The results show that different bands have different compression rates while different land cover types also have different rates. Study area 4 has the highest compression rate because area 4 possesses the characteristics of both smoothness and homogeneity. Other land cover

types also have relatively high compression rates except study area 5. Study area 5, has the longest average compression time because of its complex geographical features and textures, while study area 4, has the shortest compression time due to both its smoothness and homogeneity.

TABLE I  
THE RELATIONSHIP BETWEEN FRACTAL COMPRESSION RATE AND COMPLEXITY FOR STUDY AREA 1-4

AREA	BAND	COMPRESSION RATE	F-DIMENSION (TRIANGULAR)	C.V.	ENCODE TIME(m's")	DECODE TIME(m's")
1	1	32.9327	2.5882	1.197	immediate	0'27.12"
	2	105.7459	2.5509	0.795	immediate	0'27.60"
	3	21.0693	2.508	1.540	immediate	0'27.47"
	4	8.3445	2.5832	1.828	0'29.61"	0'28.84"
	5	5.0910	2.4903	5.570	0'55.88"	0'28.80"
	6	306.2430	2.2295	0.710	immediate	0'26.40"
	7	8.9911	2.4858	2.759	0'31.50"	0'28.08"
2	1	19.3650	2.5472	1.307	immediate	0'28.00"
	2	62.2374	2.529	0.959	immediate	0'26.56"
	3	16.0874	2.4886	1.551	immediate	0'27.54"
	4	8.1391	2.4413	3.218	0'36.36"	0'28.21"
	5	5.1222	2.452	5.576	0'58.87"	0'29.97"
	6	373.9572	2.4469	0.530	immediate	0'26.10"
	7	8.0417	2.4623	2.727	0'36.38"	0'28.64"
3	1	33.0031	2.5928	1.122	immediate	0'27.28"
	2	100.2846	2.5917	0.902	immediate	0'26.86"
	3	15.3606	2.5589	1.565	0'23.4"	0'27.88"
	4	5.4228	2.5459	4.081	0'53.34"	0'29.05"
	5	4.8432	2.5237	6.917	1'0.27"	0'29.24"
	6	364.5953	2.5159	0.758	immediate	0'26.18"
	7	6.8913	2.5342	2.694	0'44.85"	0'28.12"
4	1	403.9199	2.55	0.415	immediate	0'26.22"
	2	411.5290	2.5402	0.250	immediate	0'26.04"
	3	407.0559	2.5427	0.317	immediate	0'26.64"
	4	397.1879	2.5954	0.555	immediate	0'27.60"
	5	500.2748	2.5775	0.277	immediate	0'26.64"
	6	390.0952	2.4068	0.233	immediate	0'26.64"
	7	380.4702	2.5842	0.276	immediate	0'27.09"

TABLE II  
AVERAGE COMPRESSION RATES OF IMAGES OF STUDY AREAS

STUDY AREA	COMPRESSION RATE
1	30.362401
2	19.832137
3	27.634278
4	433.910347
5	5.610168
6	50.583721
7	28.852589
8	39.366779

TABLE III  
AVERAGE COMPRESSION RATES OF SEPARATE BANDS

OVERALL BAND	COMPRESSION RATE
1	33.978055
2	88.545123
3	26.434775
4	7.367679
5	5.842096
6	319.328277
7	11.182619

## 2. Hypothesis I: Comparison with Different Compression Methods

To test the first research hypothesis of this study, i.e. fractal image compression methods are more accurate and efficient than other commonly used compression schemes, two non-fractal compression methods, JPEG and Wavelet, were also tested on the same data set as the fractal compression techniques.

## 3. Hypothesis II: Effects of Spatial Complexity

The second hypothesis is that the spatial complexity will affect the compression rate. The more complex the image, the lower the compression rate; the less complex the image, the higher the compression rate. In other words, if smoother or more homogeneous images have lower spatial complexity, then higher compression rates can be expected, while coarser or more heterogeneous images have higher complexity and lower compression rates. These principles were tested in this study. Table I lists the relationship between compression rate and complexity.

The measurements of complexity in images include fractal dimension, and coefficient of variation (c.v.), which is the standard deviation divided by the mean. Fractal dimension is a measure of spatial complexity; whereas c.v. is a measure of non-spatial complexity. It can be found that the smoothest area (area 4), has less complexity for both the spatial and the

non-spatial measure, and the compression rates for the images of all bands are the lowest in the images.

## 4. Hypothesis III: Effects of Resolution

The third hypothesis is: Changing scale or resolution of images will result in a change in the fractal compression rate. More specifically, it is expected that increase in pixel size (i.e. decrease in spatial resolution) will lead to a higher compression rate. The increase in compression rates for the urban area is more than the increase of compression rate for the water area. The former is about 3 to 4 times higher than the original while the latter is smaller. The reason is because the water area is already very smooth, the effect of resampling or smoothing for the water will be smaller than for the urban area. This is consistent with the fact that compression rate of band 6 image of water area is not higher than the other bands as the complexity of band 6 of water area is already very low. The effect of resolution is consistent with the second hypothesis: changing resolution results in a change of complexity; and changing complexity will result in a change of compression rate.

## V. DISCUSSION

### A. Evaluation of Fractal Image Compression

Based on the results shown in the previous sections, it is apparent that the algorithms of fractal compression and decompression developed and implemented in this study are efficient, effective and consequently helpful methods for solving storage problems and providing an approach to reveal the relationship between the complexities of geographical features and the compression rates. Below is a discussion of each criterion. More evidence is provided by the result of the unsupervised classification applied on both original and restored images. Every composite image consisting of all but the thermal infrared band is classified into 10 classes (the only exception is that the image of study area 4: has only 7 classes) and the results of each pair of the original and restored images for the same study area are put together to make comparison. Then the degree of matching can be clearly tested. All of the classes have better than 95% matching (in fidelity matching most of them reach 99% and above while in the overmatching, the largest mismatching value is 3.5%. Most mismatching are below the 1% threshold). Based on the results and the analysis above, it can be said that the restored images maintain the quality of the original images within a very acceptable range of fidelity. The conformity criterion of the evaluation of the compression and decompression algorithms is met.

### B. Efficiency Evaluation

This evaluation includes two categories: compression rate and processing time. First, the compression rate will be evaluated. The maximum compression rates occur in images of study area 4, which is a water area. This phenomenon can be explained due to the smoothness of water area in all the bands. Similarly, the images of band 6 of all the study areas have high compression rates, and the average is 319.33. Accordingly, the images of this band appear smoother than other bands. When compared to traditional methods, the maximum is below 4 (Chen et al. 1987).

The computational time of fractal compression and decompression is also shown in Tables II and III. It can be seen that almost all the compressing procedures are shorter than one minute. Moreover, more than half of the compressing procedures are done immediately. All decompressing times are shorter than 30 seconds and all are similar in time, thus suggesting a uniform method of compression. This is because the decompression procedures take the same numbers of iteration. In summary, in terms of image fidelity, compression rate, and computational time, the fractal algorithms perform very well. The algorithms reach higher compression rates compared to traditional methods based on the results reported from the literature, and the processing time is also shorter.

## VI. CONCLUSION

Remote sensing images contain huge amount of geographical information and reflect the complexity of geographical features, landscape, land cover types, and spatial structures. Fractal geometry provides a means of describing and analyzing the complexity of different geographical features in remotely sensed images. It also provides a more powerful tool to compress the remote sensing data than the traditional methods. In this study, the requirement for more powerful compression tools to expedite storage and analysis for remotely sensed images was analyzed. Fractal compression and decompression methods were developed, described, and discussed from concepts and algorithms, to the final application of remotely sensed imagery. The compression results were compared with the original data in different ways, both visually and quantitatively. Fractal compression results were also analyzed with other approaches including image processing techniques, image classification method, and statistical tools. Findings of this study are as follows:

Fractal techniques can be applied to compress and decompress remote sensing images. Fidelity and efficiency of the restored images by fractal techniques are better than those of the previous image compression methods.

## REFERENCES

- [1] Anil K.Jain., Fundamentals of Digital Image Processing, PHI, New Delhi. 1975.
- [2] M. F. Barnsley, Fractals Everywhere, New York Academic, 1988.
- [3] M. F. Barnsley, and L.P. Hurd, Fractal Image Compression, Peters, Wellesley, 1993.
- [4] M.F. Barnsley. & Alan D. Sivan, "A better way to compress Images", BYTE, 1988.
- [5] W.H. Chen., C.H. Smith., and S.C.Fralick. ., "A Fast Computational Algorithm for the Discrete Cosine Transform", IEEE Transactions on Communication, Vol. 25, No. 9, 1004-1009, 1977.
- [6] R. Hamzaoui, H. Hartenstein, D. Saupe, "Local iterative improvement of fractal image codes", Image and Vision Computing, Vol. 18, No : 6-7 ,pp.565-568, 2000.
- [7] A.E. Jacquin, "Image Coding based on a Fractal theory of iterated contractive image transformations", in Proceedings SPIE Visual Communications and Image Processing '90, Vol.4, pp.2225-2228, 1990a.
- [8] A.E. Jacquin, "Fractal Image Coding: A review", Proceedings of the IEEE, Vol.81, pp.1451-1465, 1993c.

- [9] G. Lu. and T.L.Yew, "Image Compression using quad tree Partitioned Iterated function systems", Electron Letters, Vol. 30, No. 1, pp. 23-24, 1994.
- [10] Rafael C. Gonzalez and Richerd E. Woods., Digital Image Processing, Addison-Wesley Publishing Company, New York, 1992.
- [11] V. Stankovic, R. Hamzaoui, & D. Saupe, "Fast algorithm for rate-based optimal error protection of embedded codes", IEEE Transactions on Communications, Vol. 51, pp. 1788-1795, 2003.
- [12] L. Thomos., and F. Deravi, "Region-based fractal image compression using heuristic search", IEEE Transactions on Image Processing Vol.4, No. 6 pp.832-838, 1995.