Research Article GU J Sci 38(1): x-x (2025) DOI: 10.35378/gujs.1425702

Gazi University

Journal of Science

http://dergipark.gov.tr/gujs

Fractal Image Decompression Via Non-Affine Contractions

Nisa ASLAN[1], Ismail ASLAN[2,*]

[1]*Eskişehir Technical University, Department of Mathematics, TR-26470, Eskişehir, Türkiye*
[2]*Hacettepe University, Department of Mathematics, TR-06800, Ankara, Türkiye*

Highlights

- This paper focuses on fractal image compression method.
- Unlike classical fractal image compression, images were decompressed by non-affine transformations.
- Similar approach is applied for audio decompression.

Article Info

Received: 25 Jan 2024
Accepted: 08 Nov 2024

Keywords

Fractal image compression,
Iterated function system,
Audio compression
Non-affine contraction

Abstract

In this study, considering the well-known fractal image compression, we introduce the image decompression method through non-affine contraction mappings. To achieve this, we convert affine contraction mappings into non-affine contraction mappings using Lipschitz continuous functions, subject to certain assumptions. Our expectation is to obtain decompressed images of superior quality compared to the classical fractal image compression method. We also apply our method for audio decompression. At the end, we illustrate the proposed method with some examples.

1. INTRODUCTION

The concept of fractals was first introduced by B. Mandelbrot in 1975 as a way to describe the irregular shapes in the nature. However, the theory of iterated function system (IFS), which is the most well-known method to generate fractals, was developed by Hutchinson [1]. Thanks to this method, many classical fractals such as Cantor set, Sierpinski triangle, Koch curve, Sierpinski carpet, Box fractal, etc. which were also discovered earlier, can be obtained as an attractor of their related IFSs. On the other hand, as an application of IFS, Barnsley and Sloan presented a method to compress images, which is called fractal image compression [2, 3]. Since Hutchinson theory is based on Banach fixed point theory, fractal image compression relies on the concept of contractive mappings and the uniqueness of the fixed point.

In recent years, analyzing digital images using algorithms and mathematical operations on computer vision, generally called image processing, has become quite popular [4-8]. There are various types of techniques in image processing, such as image segmentation, edge detection, image compression, etc. The fractal image compression method discussed in this study is one of the image compression techniques. The purpose of this method is to approximate the original images by storing less data for a given image, which involves contraction mappings of the associated IFSs rather than the data of the original image.

If we were to summarize, fractal image compression is a technique used to represent and compress digital images by exploiting the self-similar properties of fractal geometry. Instead of storing pixel values directly, the method involves encoding the image as a set of affine contraction mappings within an IFS. These mappings transform blocks of pixels into approximate copies of other parts of the image. During the compression process, the algorithm identifies and stores the contraction mappings that best represent the image. In the decompression part, the original image is reconstructed by iteratively applying the stored

*Corresponding author, e-mail: ismail-aslan@hacettepe.edu.tr mappings. The goal is to achieve compression with reduced data storage while maintaining image quality [2, 3].

1.1. Literature Review

Fractal image compression methods given in [2, 3] have been improved over time. These studies can be exemplified by fractal block coding in [9]. Then Fischer advanced new techniques for image compression based on fractals in [10]. Using quadtree and adaptive quadtree partitioning methods, efficiency of the compression methods has been enhanced, notably by reducing encoding time [8, 11, 12]. Fractal image compression remains a popular research topic, with ongoing studies focused on improving compression performance, optimizing methods, and further reducing encoding time [5, 13].

In the present paper, by using non-affine contraction mappings, we aim to decompress the images, which are encoded in the classical way. Even though, our decompression can require a little more time, we expect to obtain better or at least similar-quality decompressed images. To achieve this goal, we utilize from non-linear transformations compatible with contractions under some certain assumptions and related approximation theorem in [14].

2. PRELIMINARIES

In this part, we give the basics of the fractal image compression. Thus, we first recall the theory of IFS, given in [1].

Definition 2.1. Let (X,d) be a complete metric space and $f_i : X \to X$ $(i \in \{1,2,...,n\})$ be the contraction mappings, that is, there exists a contractivity factor $r_i < 1$ such that for all $x,y \in X$

$$d\big(f_i(x), f_i(y)\big) \leq r_i d(x,y)$$

where $0 \leq r_i < 1$ for all $i \in \{1,2,...,n\}$. Then $\{X; f_1, f_2, ..., f_n\}$ is called the iterated function system (IFS).

Theorem 2.2. Let (X,d) be a complete metric space and $\{X; f_1, f_2, ..., f_n\}$ be an iterated function system. Then for every $S \in \mathcal{H}(X)$, the function $F : \mathcal{H}(X) \to \mathcal{H}(X)$ defined by

$$F(S) := \bigcup_{i=1}^{n} f_i(S)$$

is a contraction mapping on $\mathcal{H}(X)$, which is set of all compact nonempty subsets of X and complete metric space with respect to Hausdorff metric h. Note that

$$r = \max_{i \in \{1,2,\dots,n\}} \{r_i : r_i \text{ is the contractivity factor of } f_i\}$$

is called the contraction constant of F.

Furthermore, F has a unique fixed point A such that $F(A) = A$. This fixed point is also called the attractor of the IFS, and for each $S \in \mathcal{H}(X)$, $F^{(k)}(S) := (F \circ F \circ \dots \circ F)(S)$ converges to A as $k \to \infty$.

Now, we can give the details of the fractal image compression.

Through the process of fractal image compression, the technique is mainly based on the Collage Theorem (see [2, 10]), which states that for a given set S, if there exist contraction mappings w_i, whose attractor A is close enough to S, that is,

$$h\left(S, \bigcup_{i=1}^{N} f_i(S)\right) < \varepsilon,$$

then *S* is close enough to *A*, that is,

$$h(S,A) < \frac{\varepsilon}{1-r}$$

where *h* is the Hausdorff metric and *r* is the contraction constant of the system. Therefore, instead of the original image, if we find the contraction mappings of an image, which is sufficiently close to the original one, then we obtain a very similar image to the original one.

Now we can mention about the encoding process of a digital image. A gray-scale image can be represented by functions as follows:

Let an $n \times n$ pixel resolution gray-scale image be given. Then taking $I = [0,1]$, define $f : I^2 \to I$, such that

$$f(x,y) \coloneqq \sum_{i=1}^{n}\sum_{j=1}^{n} p_{ij} 1_{ij,n}(x,y)$$

where

$$1_{ij,n}(x,y) \coloneqq \begin{cases} 1, & (x,y) \in (\frac{i-1}{n},\frac{i}{n}] \times (\frac{j-1}{n},\frac{j}{n}] \\ 0, & (x,y) \in I^2 \backslash (\frac{i-1}{n},\frac{i}{n}] \times (\frac{j-1}{n},\frac{j}{n}] \end{cases} \quad (i,j = 1,2,\dots,n) \qquad (2.1)$$

and p_{ij} corresponds to (i,j)'th normalized gray-scale levels of the given image. To measure the distances between two graphs of images *f* and *g*, the following metric

$$d_{L^2}\big(\mathrm{graph}(f),\mathrm{graph}(g)\big) = \left(\int_{I^2} |f(x,y) - g(x,y)|^2 dA\right)^{\frac{1}{2}}$$

$$= \frac{1}{n}\left(\sum_{i,j=1}^{n} \left|p_{ij} - q_{ij}\right|^2\right)^{\frac{1}{2}}$$

can be used, where p_{ij} and q_{ij} correspond to (i,j)'th normalized pixel value of *f* and *g* respectively. Here, the space of all graphs of images together with the above metric is a complete metric space. Using the definition in (2.1), the graph of "Lena" image is plotted in Figure 1 for $n = 64$.

Every image (such as face, tree etc.) can not be obtained as an attractor of iterated function systems. Although such images are not self-similar, they have different type of self-similarity. For example, a piece of an image can be similar to different portion of its another piece (see Figure 2). There can also be similarities under the symmetries of the square, which are obtained by rotation and reflection. Now for a given $n\times n$ pixel resolution gray-scale digital image, divide I^2 into m^2 (*m* is a divisor of *n*) non-overlapping blocks with equal area (inclusion or exclusion of borders do not change the area),

$$\left(\frac{k-1}{m},\frac{k}{m}\right] \times \left(\frac{l-1}{m},\frac{l}{m}\right] \subseteq R_{k,l} \subseteq \left[\frac{k-1}{m},\frac{k}{m}\right] \times \left[\frac{l-1}{m},\frac{l}{m}\right]$$

such that

$$\bigcup_{k,l=1}^{m} R_{k,l} = I^2$$

and, only $p \times p$ ($p = n/m$) pixel resolution parts of the image lie on each $R_{k,l}$. Moreover, divide I^2 into all possible (overlapping) blocks with equal area $D_{i,j}$ such that $q \times q$ pixel parts of the image lie on $D_{i,j}$ (usually it is assumed that $q = 2p$), where $q > p$.

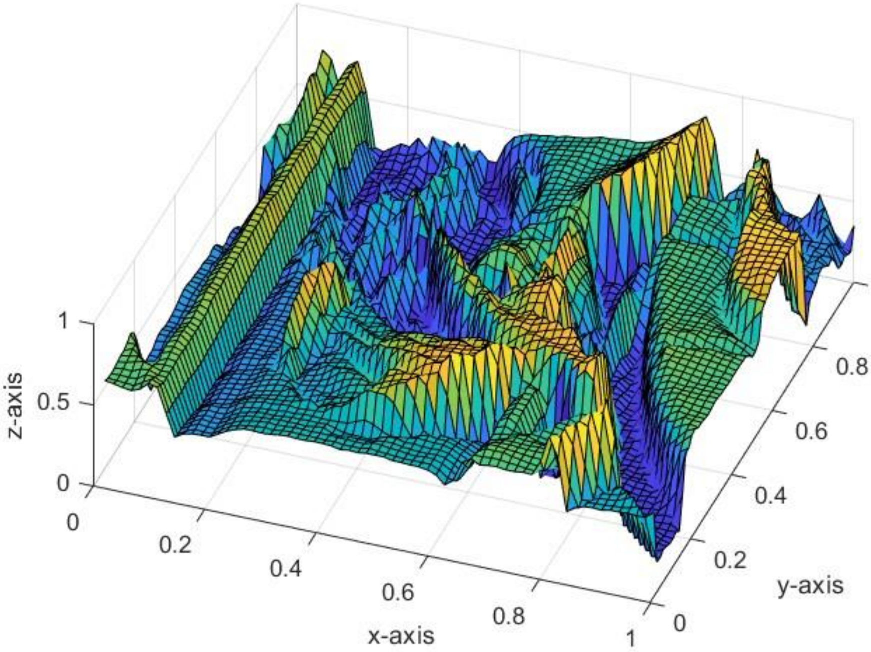


Figure 1. Graph of 64 × 64 pixel Lena

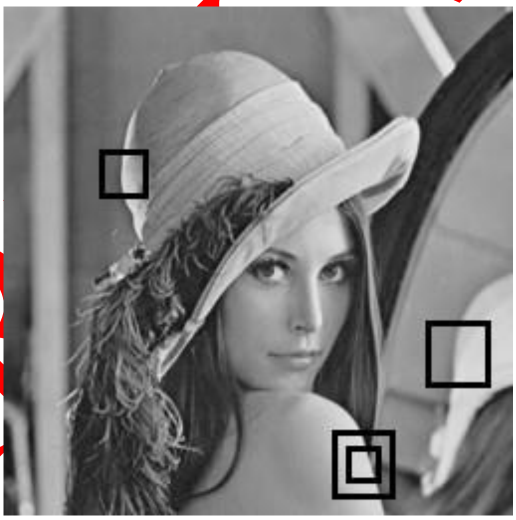


Figure 2. Some self-similar parts of Lena

Here $R_{k,l}$ and $D_{i,j}$ are called by range blocks and domain blocks respectively. For each $R_{k,l}$, we need to find $D_{i,j}$ and the contraction mappings

$$w_{i_{k,l},j_{k,l}}: \left(D_{i,j} \times I\right) \cap \text{graph}(f) \to R_{k,l} \times I$$

in the following form

$$w_{i_{k,l},j_{k,l}}(x,y,z) = \begin{pmatrix} a_{i_{k,l},j_{k,l}} & b_{i_{k,l},j_{k,l}} & 0 \\ c_{i_{k,l},j_{k,l}} & d_{i_{k,l},j_{k,l}} & 0 \\ 0 & 0 & e_{i_{k,l},j_{k,l}} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} g_{i_{k,l},j_{k,l}} \\ h_{i_{k,l},j_{k,l}} \\ o_{i_{k,l},j_{k,l}} \end{pmatrix} \qquad (2.2)$$

$$w_{i_{k,l},j_{k,l}}(x,y,z) = \left(a_{i_{k,l},j_{k,l}}x + b_{i_{k,l},j_{k,l}}y + g_{i_{k,l},j_{k,l}}, c_{i_{k,l},j_{k,l}}x + d_{i_{k,l},j_{k,l}}y + h_{i_{k,l},j_{k,l}}, e_{i_{k,l},j_{k,l}}z + o_{i_{k,l},j_{k,l}}\right)$$

such that the distance between

$$w_{i_{k,l},j_{k,l}}\left(\left(D_{i,j} \times I\right) \cap \text{graph}(f)\right)$$

and

$$\left(R_{k,l} \times I\right) \cap \text{graph}(f)$$

is minimum. Here the mapping $w_{i_{k,l},j_{k,l}}$ has both spatial part (first two components) and image part (third component), which are independent of each other. We first find the coefficients of the third component of $w_{i_{k,l},j_{k,l}}$ (this actually means we seek $D_{i,j}$, whose pixel values above is most similar to the pixel values on $R_{k,l}$). However, since the parts of the image, which lie on $D_{i,j}$ and $R_{k,l}$ have different number of pixels, we divide $D_{i,j}$ into p^2 equal size of non-overlapping sub-blocks having 2 × 2 pixels part of the image, and take one from each as a representative part. Subsequently, for a given $R_{k,l}$, using the least squares method, we can find the coefficients $e_{i_{k,l},j_{k,l}}$ and $o_{i_{k,l},j_{k,l}}$ of $w_{i_{k,l},j_{k,l}}$ for all $i,j = 1,...,n$. Then we compare the distance

$$d_{L^2}\left(w_{i_{k,l},j_{k,l}}\left(\left(D_{i,j} \times I\right) \cap \text{graph}(f)\right), \left(R_{k,l} \times I\right) \cap \text{graph}(f)\right)$$

for all i,j, and store the coefficients $\tilde{e}_{i_{k,l},j_{k,l}}$ and $\tilde{o}_{i_{k,l},j_{k,l}}$ of the map

$$\widetilde{w}_{i_{k,l},j_{k,l}}: \left(\widetilde{D}_{i_{k,l},j_{k,l}} \times I\right) \cap \text{graph}(f) \to \left(R_{k,l} \times I\right) \cap \text{graph}(f),$$

which gives the minimum distance for each k,l (we also consider the symmetry group of the domain block $D_{i,j}$, which is obtained by reflections and rotations of itself). In addition, instead of finding the first two component $\tilde{a}_{i_{k,l},j_{k,l}}$, $\tilde{b}_{i_{k,l},j_{k,l}}$, $\tilde{c}_{i_{k,l},j_{k,l}}$, $\tilde{d}_{i_{k,l},j_{k,l}}$, $\tilde{g}_{i_{k,l},j_{k,l}}$ and $\tilde{h}_{i_{k,l},j_{k,l}}$ for $w_{i_{k,l},j_{k,l}}$, it is enough to store the pairs $(i_{k,l}, j_{k,l})$, which tells us $\widetilde{D}_{i_{k,l},j_{k,l}}$ maps to $R_{k,l}$. It is worth mentioning that, since

$$\widetilde{w}_{i_{k,l},j_{k,l}}\left(\left(\widetilde{D}_{i_{k,l},j_{k,l}} \times I\right) \cap \text{graph}(f)\right)$$

is not exactly equal to

$$\left(R_{k,l} \times I\right) \cap \text{graph}(f),$$

we have to allow some error in this part. Once this process is applied for all $R_{k,l}$, we have the contraction mappings of all pieces of the image, which is called by encoding process. This process is time consuming, however, decoding process is faster and easier.

Assume that f_0 is any $n \times n$ pixel resolution gray-scale image. The contraction mapping of W of the image f is defined as follows:

$$W\left(\text{graph}(f_0)\right) = \bigcup_{k,l=1}^{m} \widetilde{w}_{i_{k,l},j_{k,l}}\left(\left(\widetilde{D}_{i_{k,l},j_{k,l}} \times I\right) \cap \text{graph}(f_0)\right),$$

where the domain blocks $D_{i,j}$ correspond to the image f_0. Then by iterating W, we can obtain the image f with some error, since

$$W^{(n)}\left(\text{graph}(f_0)\right) \cong \text{graph}(f).$$

Remark 2.1. In the theory above, one question may arise that domains $D_{i,j}$ and ranges $R_{k,l}$ may not be compact, which may result to the failure of Theorem 2.2. However, our aim is not exactly to obtain the graphs of the given images, but to obtain the pixel values. Therefore, instead of the $D_{i,j}$ and $R_{k,l}$ defined above, taking compact subsets of them can complete the theory literally.

3. DECOMPRESSION BY NON-AFFINE CONTRACTION MAPPINGS

In this section, we aim to decompress images compressed using FIC method, through non-affine contraction mappings. Our decompression method is mainly based on the following theorem.

Theorem 3.1. ([14]). Let $\{X;f_1,f_2,...,f_m\}$ be an IFS and $H_k: X \to X$ is a sequence of Lipschitz continuous functions with Lipschitz constant M such that $\lim_{k\to\infty} H_k(u) = u$ for all $u \in X$. Then for any $S \in \mathcal{H}(X)$

$$\lim_{k\to\infty}\lim_{n\to\infty} F_k^{(n)}(S) = \lim_{n\to\infty} F^{(n)}(S) = A$$

holds, where A is the attractor of $\{X; f_1, f_2, \dots, f_m\}$, $F(S) = \bigcup_{i=1}^{m} f_i\,(S)$ and $F_k(S) = \bigcup_{i=1}^{m}(H_k \circ f_i)\,(S)$. Here we assume M is sufficiently small such that $M*r<1$, where r is the contraction constant of $\{X;f_1,f_2,...,f_m\}$.

Remark 3.1. Note that the Lipschitz constant $M \geq 1$, since

$$|H_k(u) - H_k(v)| \leq M|u - v|$$

for all $u,v \in X$ and $k \in N$ (where the Lipschitz constant is uniform w.r.t k) and

$$\lim_{k\to\infty} |H_k(u) - H_k(v)| = |u - v| \leq M|u - v|.$$

We should also state that, sometimes the system can be non-contractive but eventually contractive (see page 52, [10]), that means w is not contractive but $w^{(m)}$ is contractive for larger values of m, in this case the theory still holds. Experiments show that, although contractivity factor $r \in (1,1.2]$, the system can be eventually contractive [10].

This theorem allows us to approximate self-similar fractals with non-affine contraction mappings. Some examples of H_k functions satisfying the conditions of Theorem 3.1 can be found in [14].

Now, in this paper, using non-affine contraction mappings we aim to decompress images with better qualities with respect to classical FIC, which is given in detail above. The idea in this part is that, since we allow some error in the classical fractal image compression, choosing a suitable H_k, we can make it possible to overcome this error to some extent and may obtain better results. As it is common in the literature, in order to measure the quality of the decompressed $n \times n$ pixel resolution gray-scale image I_1 with respect to original image I_2, we will use PSNR (peak signal to noise ratio) metric

$$\text{PSNR} = 20\log_{10}\left(\frac{1}{\sqrt{\text{MSE}}}\right),$$

where MSE (mean squared error) is given by

$$\frac{\sum_{M,N=1}^{n}[I_1(M,N) - I_2(M,N)]^2}{n^2}$$

and $I_i(M,N)$ corresponds to (M,N)'th pixel of the image I_i for $i = 1,2$. According to the above definition, higher PSNR values means better images.

We remark that our method gives the possibility of having a higher quality image. In addition, since $\lim_{k\to\infty} H_k(S) = S$ (see Lemma 2.1 in [14]), taking sufficiently large $k \in$ N, we at least get the same quality image, which is obtained by classical FIC. Furthermore, if we assume $H_k(u) = u$ for all $k \in$ N, then our decompression method reduces to the classical case.

We assume that $p = 8$ and $q = 16$ in the above method and compress "Lena"'s 256 × 256 pixels image given in Figure 4f using the FIC method. Now, we obtain two 32 × 32 matrices and one 2 × 32 × 32 matrix, which correspond to coefficients of the contraction mappings and the places of the domain and range blocks. This means that we compress Lena at a 1/16 ratio. In the decoding process we will use non-affine contraction mappings, which are obtained by applying H_k to third component of each contraction mappings $\left(\widetilde{w}_{i_{k,l},j_{k,l}}\right)_3 = \tilde{e}_{i_{k,l},j_{k,l}} z + \tilde{o}_{i_{k,l},j_{k,l}}$ (see (2.2)), i.e.,

$$H_k\left(\left(\widetilde{w}_{i_{k,l},j_{k,l}}\right)_3\right) = H_k\left(\tilde{e}_{i_{k,l},j_{k,l}} z + \tilde{o}_{i_{k,l},j_{k,l}}\right).$$

Here we assume that

$$H_k(u) = \frac{ku^3}{ku^2 - 0.0116}.$$

whose Lipschitz constant is $M \approx 1.1$ (see Remark 3.1). In this application, we start with the well-known "Baboon" image given in Figure 3. Using non-affine mappings for $k = 64$, we iterate the "Baboon" image for 9 times and obtain the following images given in Figure 4. The PSNR value obtained using our method was slightly higher than the classical method. Some comparisons of the PSNR values can also be found in Table 1.

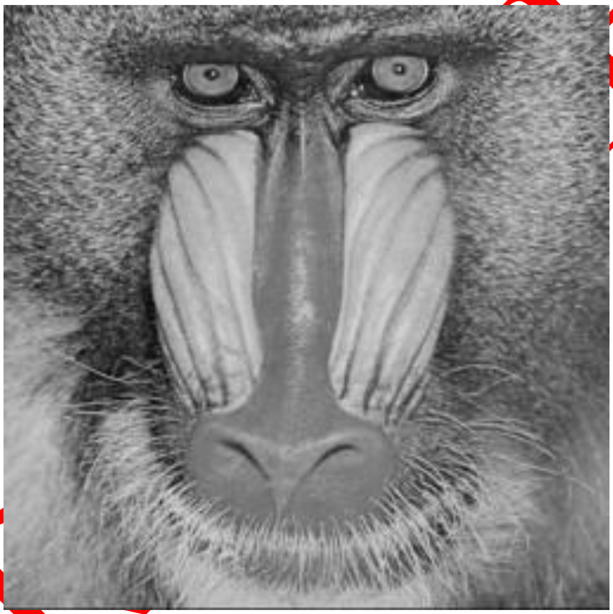
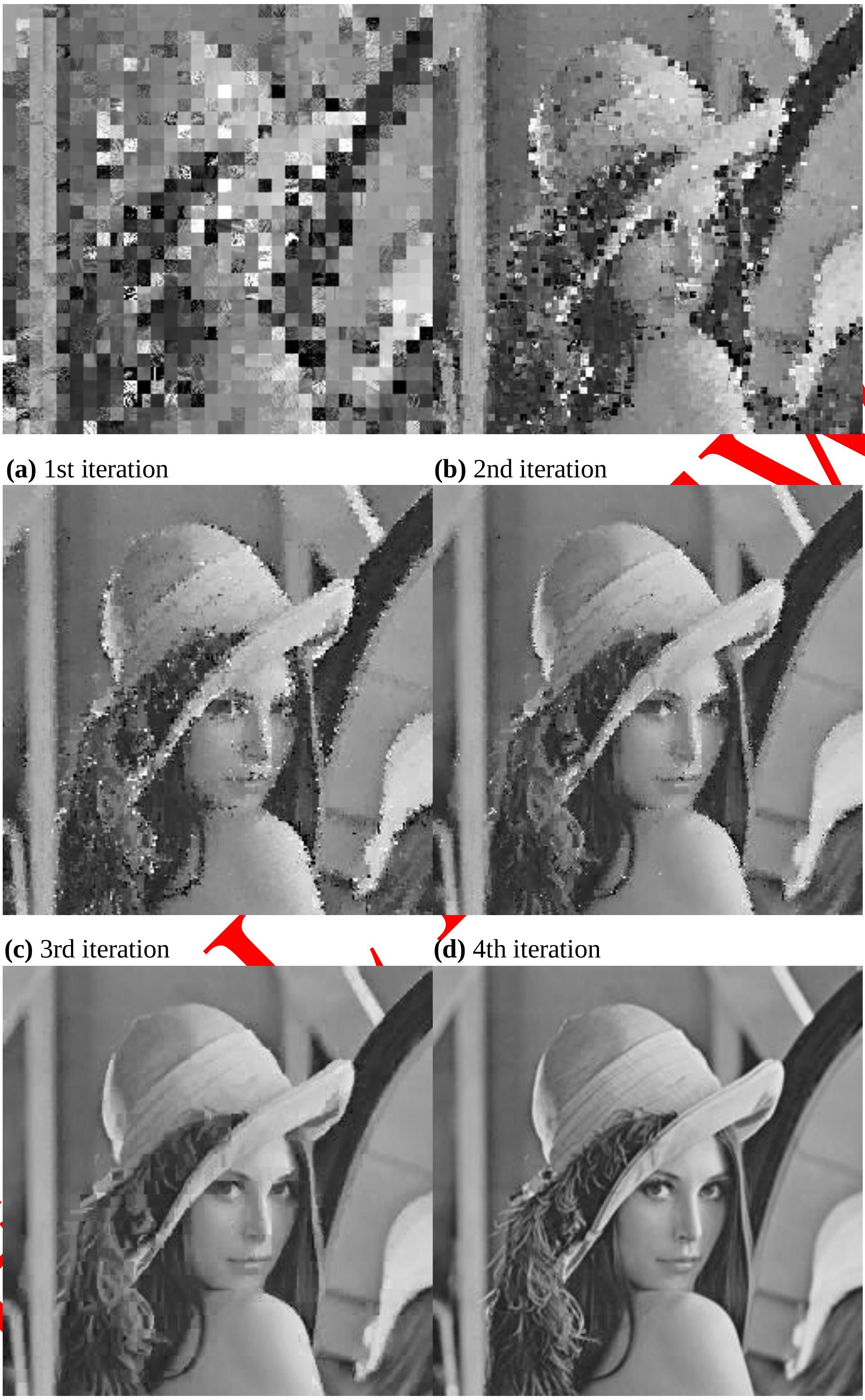


Figure 3. Initial Image (Baboon)

As shown in Table 1, our method performs better than the classical decompression method. Based on Table 1, for each iteration, we select the optimal integer k that yields the best PSNR. To this end, we considered the interval [1,150] (to determine the interval [1,150] for each iteration, we first checked where the PSNR was increasing and decreasing with a step size of 50 unit and decided to focus on this interval) and, after obtaining the contraction constants, determined the k parameter for the given H_k using Matlab. This step can be considered part of the encoding process. The computational time (in seconds) required to find the parameter k, is shown in Figure 5. The encoding time using the classical encoding method for the 256 × 256 pixels Lena image is approximately 7 minutes. If we also consider the time needed to determine the parameter k, the total time is roughly 7-8 minutes. On the other hand, since this is a compression method, all quadtree partitioning techniques or other methods can be used to reduce the computational complexity of the encoding time.



(a) 1st iteration **(b)** 2nd iteration

(c) 3rd iteration **(d)** 4th iteration

(e) 9th iteration (PSNR=29.3483) **(f)** Original Image (Lena)

Figure 4. Non-affine iterations of Baboon for *k* = 64

Table 1. Comparison of PSNR values of the images

PSNR (classical decompression)	PSNR (non-affine decompression)	iteration number	k
15.4755	15.5077	2	101
19.3177	19.3296	3	80
23.3928	23.4170	4	50
26.4164	26.4353	5	50
27.3533	27.3647	6	57
29.0256	29.0317	7	50
29.2631	29.2671	8	90
29.3467	29.3483	9	64

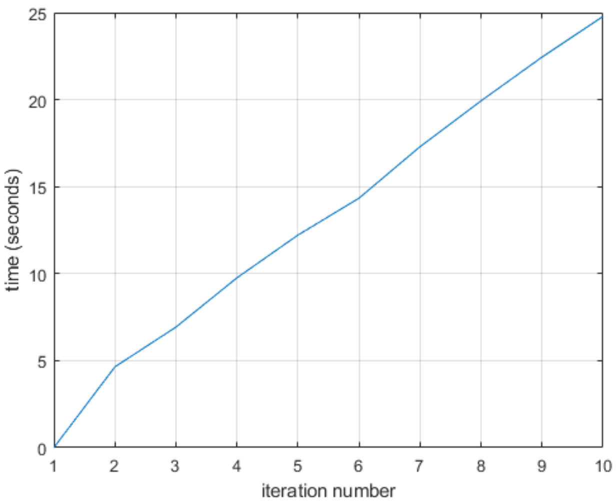


Figure 5. Elapsed time for finding the best parameter k

We show that this method can also be useful for audio compression. It is worth mentioning that a digital audio signal is actually a $1\times n$ matrix, whose components correspond to the amplitude of the audio signal. Similar to the images, audio signals can also be resresented by a function. Since the idea behind FIC is the existence of self-similar parts of the image in itself, we think that the same theory can also be applied for audio compression. As we check on Matlab, many sounds contain self-similar parts in itself. Therefore, we apply our method to audio compression and compare the sound's quality with the classical method using PSNR metric that is modified for digital audios.

The definition of the utilized PSNR metric for audios is taken as follows:

$$\mathrm{PSNR} \coloneqq 20\log_{10}\left(\mathrm{maxval}/\sqrt{\mathrm{MSE}}\right)$$

$$\mathrm{MSE} = \frac{\sum_{i=1}^{n}[\mathrm{signal1(i)} - \mathrm{signal2(i)}]^2}{n}$$

where signal1 and signal2 are the given audio signals,

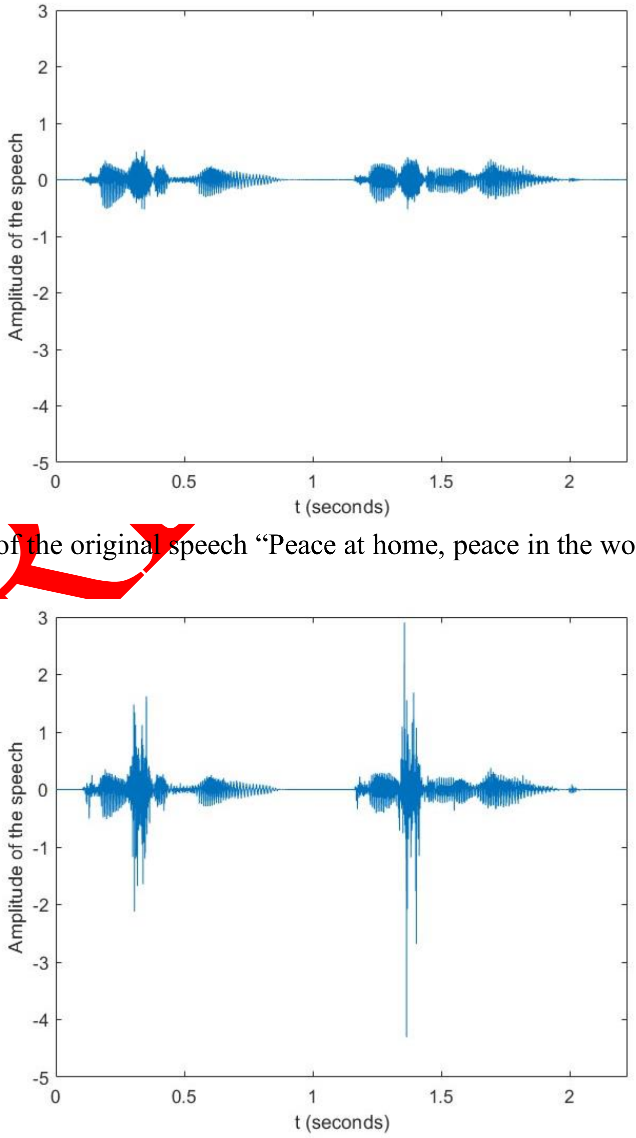
$$\mathrm{maxval} = \max\{\max_{i=1,\ldots,n}|\mathrm{signal1(i)}|, \max_{i=1,\ldots,n}|\mathrm{signal2(i)}|\}$$

and n is the sample number of the digital audio signal.

Now, let us analyze the famous quote of Mustafa Kemal Atatürk, the founder of Türkiye: "Peace at home, peace in the world". We had the computer say "Peace at home, peace in the world" sentence and then we obtain the signal in Figure 6a. When we import the audio file into Matlab, we see that the audio file has n = 106800 sample points, that is, it is stored by 1 × 106800 matrix. In the encoding process, as a domain D_i, we take (overlapping) line blocks, which contain 30 elements of the original audio and as a range R_k, we take (non-overlapping) line blocks containing 10 elements. Similar to the FIC, we also consider the reflection of the audio file in encoding process. When we applied the compression method, we store the audio signal by two 1 × 10680 matrices and one 2 × 10680 matrix, which correspond to the coefficients of the contraction mappings and the places of domain and range blocks. As can be seen from this, the given audio file can be obtained by half time elements of the original one. After obtaining the contraction mappings, we compare the classical method with non-affine iteration method to bring the audio file back. For this aim, we consider a random audio file with 106800 sample points. When we iterate the random audio file 1500 times, taking H_k as

$$H_k(u) = \frac{ku^3}{ku^2+1.7} \tag{3.1}$$

with a Lipschitz constant of $M \approx 1.12$, through non-affine contractions, we successfully recover the original audio file for $k = 6000$. The obtained signal is plotted in Figure 6b.



(a) Signal of the original speech “Peace at home, peace in the world”

(b) Signal of the speech decompressed by non-affine iteration

Figure 6. Original audio and decompressed audio

We also compare the PSNR values, which are obtained by non-affine and affine methods. PSNR of the non-affine decompression method is evaluated by 56.9228, while PSNR of the classical method is 50.2436. Therefore, in this specific case our method gives less noisy audio file than the classical one. We should also note that 1500 times iteration does not take much time, which is computed by a regular computer in almost 20 seconds. In addition, if we iterate the system for 10 times, we still get an understandable audio file in almost 0.2 seconds with more noise. Moreover, we still get better PSNR value with respect to classical method in this case.

4. RESULTS AND CONCLUSION

In this study, we explored a novel approach to decompress images encoded using classical fractal image compression (FIC) methods by employing non-affine contraction mappings. The primary aim was to assess whether these non-affine mappings could yield better decompressed image quality compared to the classical affine-based FIC techniques. We assessed the quality of decompressed images using the Peak Signal-to-Noise Ratio (PSNR), a widely used metric in image processing for quantifying image fidelity. While the improvement in PSNR may appear modest, it is significant that the new method could be useful for obtaining better images with different kind of Lipschitz functions H_k.

We also remark that a smaller contractivity factor leads to faster convergence. Since M ≥ 1 holds in all cases, it is advantageous to select the H_k functions with a smaller Lipschitz constant, ideally close to or equal to 1. However, it is equally important to ensure that the choice of H_k disrupts linearity, particularly within the interval [0,1] as pixel values of images are normalized within this range. This approach enables the system to better tolerate the approximation error introduced by the Collage Theorem.

Now, we compare the computational time of the proposed method with classical decompression method. Here, we utilized from the contraction mapping of the 256 × 256 pixel Lena image. We used the same H_k as the one mentioned above for Lena image for k=100. From 1 to 100 iterations, as shown in Figure 7, although our method takes more time, the difference in computational times is sufficiently slight.

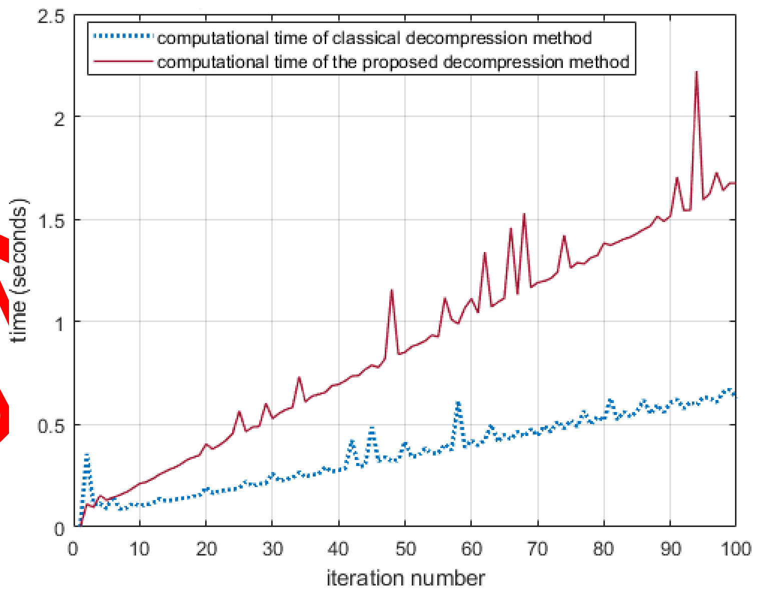


Figure 7. A comparative analysis of the computational time for decompression methods

The results demonstrate that even though the computational complexity of the decompression process increases slightly, we can get better-quality images.

CONFLICTS OF INTEREST

No conflict of interest was declared by the authors.

REFERENCES

[1] Hutchinson, J.E., “Fractals and self similarity”, Indiana University Mathematics Journal, 30(5): 713-747, (1981).

[2] Barnsley, M.F., “Fractals Everywhere”, Dover Publications, Inc: Mineola, New York, USA, (2012).

[3] Barnsley, M.F., Sloan, A.D., “A better way to compress images”, Byte 13(1): 215-223, (1988).

[4] Aslan, İ., Gökçer Ellidokuz, T.Y., “Approximation by N-dimensional max-product and max-min kind discrete operators with applications”, Filomat, 38(5): 1825-1845, (2024).

[5] Çelik, D., Deniz, A., Özdemir, Y., “Graph-Directed Fractal Image Compression”, Eskişehir Technical University Journal of Science and Technology B- Theoretical Sciences, 10(1): 1-10, (2022).

[6] Gökçer, T.Y., Aslan, İ., “Approximation by Kantorovich-type max-min operators and its applications”, Applied Mathematics and Computation 423: 127011, (2022).

[7] Menassel, R., “Optimization of fractal image compression”, Fractal Analysis-Selected Examples. IntechOpen, (2020).

[8] Nandi, U., “Fractal image compression using a fast affine transform and hierarchical classification scheme”, The Visual Computer, 38(11): 3867-3880, (2022).

[9] Jacquin, A.E., “Image coding based on a fractal theory of iterated contractive image transformations”, IEEE Transactions on image processing, 1(1): 18-30, (1992).

[10] Fischer, Y., “Fractal Image Compression Theory and Application”, Springer, New York, (1994).

[11] Nandi, U., Fractal image compression with adaptive quardtree partitioning and non-linear affine Map, Multimedia Tools and Applications, 79(35): 26345-26368, (2020).

[12] Nandi, U., Mandal, J. K., Efficiency and capability of fractal image compression with adaptive quadtree partitioning, The International Journal of Multimedia & Its Applications, 5(4): 53, (2013).

[13] Xu, C., Xie, D., Guo, H., He, J., Chen, M., “Optimization Method for Fractal Image Compression Based on Self-similarity Evaluation and Gradient Bisection Algorithm”, In International Conference on Intelligent Computing (pp. 218-233). Singapore: Springer Nature, Singapore, (2024).

[14] Aslan, N., Aslan, İ., “Approximation to the classical fractals by using non-affine contraction mappings”, Portugaliae Mathematica, 79: 45–60, (2022).