An Improved Robust Fractal Image Compression Based on M-Estimator

Penghe Huang 1, Dongyan Li 1,*, and Huimin Zhao 2

1 Software Technology Institute, Dalian Jiaotong University, Dalian 116028, China; hph@djtu.edu.cn
2 School of Electronic Information and Automation, Civil Aviation University of China, Tianjin 300300, China; hmyzhao@cauc.edu.cn
* Correspondence: ldy@djtu.edu.cn; Tel.: +86-0411-86223615

Abstract: In this paper, a robust fractal image compression method based on M-estimator is presented. The proposed method applies the M-estimator to the parameter estimation in the fractal encoding procedure using Huber and Tukey’s robust statistics. The M-estimation reduces the influence of the outliers and makes the fractal encoding algorithm robust to the noisy image. Meanwhile, the quadtree partitioning approach has been used in the proposed methods to improve the efficiency of the encoding algorithm, and some unnecessary computations are eliminated in the parameter estimation procedures. The experimental results demonstrate that the proposed method is insensitive to the outliers in the noisy corrupted image. The comparative data shows that the proposed method is superior in both the encoding time and the quality of retrieved images over other robust fractal compression algorithms. The proposed algorithm is useful for multimedia and image archiving, low-cost consumption applications and progressive image transmission of live images, and in reducing computing time for fractal image compression.

Keywords: fractal image; image compression; robustness; M-estimator

1. Introduction

The idea of fractal image compression (FIC) was originally introduced by Barnsley et al. in 1987 [1], and the first automatic algorithm for FIC was developed by Jacquin in 1992 [2]. Due to the high compression ratio and simple decompression method, the FIC algorithm attracts many researchers’ interests [3–6]. However, there are two drawbacks of fractal encoding: its complex computation and its retrieved image quality. Many fractal-coding algorithms have been developed to improve on these drawbacks [7–12]. Firstly, based on the basic concepts and methods of fractal image compression algorithms [13–19], a particle swarm optimization (PSO) method by utilizing the visual information of the edge property is proposed, and this method speeds up the encoder 125 times faster with only 0.89 dB decay of image quality in comparison to the full search method [20–24]. Then, several fractal image compression algorithms based on modified gray-level transform with fitting plane, characteristics of fractal and partitioned iterated function system, and wavelet transform with diamond search are proposed, which have better results in terms of compression speed and quality of results. Thereafter, high-efficiency video coding provided significant performance improvement in the compression ratio [25–29]. At the same time, various optimization algorithms [30–33], robustness [34–37], and feature extraction research [38–41] provide new ideas for the field of image compression.

However, little attention has been given to the robustness of such fractal-based methods until Ghazel et al. [42] proposed a fractal-based method to enhance and restore a noisy image. The method is based on the Additive White Gaussian Noise (AWGN), whose amplitude distribution follows a Gaussian distribution, while the power spectral density is uniformly distributed. Indeed, one of the original motivations for this study was the observation that a noisy image is somewhat denoised when it is fractally coded. Lu et al. [43]...
presented an enhanced fractal predictive denoising algorithm for denoising the AWGN images by using a quadratic gray-level function. A new similarity measure for fractal encoding has been introduced by Jeng et al. [44], the so-called, Huber Fractal Image Compression (HFIC) schema. It brings the linear Huber regression technique from the robust statistics into the encoding procedure of the fractal image compression. Lin [45] proposed a similar technique of linear robust regression with least absolute derivation (LAD), least trimmed squares (LTS), and Wilcoxon to FIC.

The linear regression techniques are the attempts toward the design of robust fractal image compression [46–48]. However, there is a primary drawback to the current schema: the complexity of computations [49–52]. It needs almost ten thousand seconds for a full search HFIC algorithm to encode a standard gray-level Lena image. In our analysis, there are many calculations performed that are not required. For example, the fractal graphics compression technology based on a genetic algorithm [53,54] considers and utilizes the self-transformation characteristics of the image, but its disadvantage is that it requires an extensive search and is time-consuming. The algorithm proposed in this paper can reduce the compression time by effectively eliminating many unnecessary calculations. This approach is shown to significantly improve the time of compression procedures without affecting the quality of the reconstructed image. The proposed algorithm is a robust M-estimator-based fractal image compression algorithm (RMFIC). It is a full-search fractal encoding algorithm using the quadtree partitioning method and it uses the principle of linear robust statistics. The goal of the work is to improve the efficiency and the retrieved image quality of existing robust fractal encoding methods, such as HFIC and LAD-FIC. We chose two M-estimators for simulation; Huber’s M-estimator [55] and Tukey’s bisquare M-estimator [56].

The rest of the paper is organized as follows. Section 2 reviews the principle of original fractal image compression. Section 3 reviews the concepts of M-estimator and further describes Huber’s case and Tukey’s bisquare case. Section 4 describes the proposed RMFIC in detail. The experimental results are demonstrated in Section 5. Last, Section 6 presents the conclusion.

2. Baseline Fractal Image Compression

The fractal coding process can be described as follows: the original image is partitioned into non-overlapping blocks called range blocks (R) which are usually 8 × 8 pixels each. There are also domain blocks (D) which are twice the size of range blocks and overlap such that a new domain block starts every pixel. For each range block (r), a domain block (D) that best approximates this range block under the predetermined contractive transformations (ω), is searched for. In detail, these transformations contract the domain blocks spatially to 8 × 8 pixels and then transform the gray levels by a combination of rescaling, offset adjustment, reflection, and rotation. The general fractal affine transformation can be expressed as

\[ \omega_i(D_t) = \varphi_i(\gamma_i(D_t)) \]  

where \( \gamma_i \) is the down-sampling operation. We denote \( d = \gamma_i(D_t) \), and \( \varphi_i \) can be expressed as

\[ \varphi \left[ \begin{array}{c} x \\ y \\ d(x,y) \end{array} \right] = \left[ \begin{array}{ccc} a & b & 0 \\ c & d & 0 \\ 0 & 0 & s \end{array} \right] + \left[ \begin{array}{c} t_x \\ t_y \\ o \end{array} \right] \]

where \((t_x, t_y)\) is the location of domain block \( D_t \) in original image, \( s \) is the contract scaling, and \( o \) is the brightness offset. The \( 2 \times 2 \) sub-matrix

\[ \left[ \begin{array}{cc} a & b \\ c & d \end{array} \right] \]
represents the isometrical transformations in Equation (3).

\[
\tau_0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \tau_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad \tau_2 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \tau_3 = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix},
\]

\[
\tau_4 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad \tau_5 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad \tau_6 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}, \quad \tau_7 = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix}
\]

(3)

Hence, the general affined transformation can also be written as

\[
\omega_i(D_t) = s_i \cdot \tau_i(\gamma_i(D_t)) + o_i
\]

(4)

Equation (2) calculates the reconstructed (approximated) range block as the transformation of \( d \), denoted as \( \tilde{d} \).

\[
\tilde{d} = s \cdot \tau(d) + o
\]

(5)

The parameters are selected so that the transformed domain block \( D_t \) is a good approximation of the range block \( r_j \) by minimizing the matching error

\[
E = \min_{d \in \Omega} \min_k \| s_k \cdot d_k + o_k I - r \|^2
\]

(6)

where, \( d_k = \tau(d) \), \( k \) is the index of the isometrical transformation. The parameter \( s \) and \( o \) can be obtained by the least-squares method as

\[
s = \frac{< r - \tau I, d - \tilde{d} I >}{\| d - \tilde{d} I \|^2}
\]

(7)

\[
o = \tau - s \tilde{d}
\]

(8)

The fractal code consists of \( t_x^*, t_y^*, k^*, s^* \) and \( o^* \). In practice, \( t_x^*, t_y^*, k^* \), \( s^* \) and \( o^* \) can be quantized by 8, 8, 3, 5, and 7 bits, respectively.

At decoder, the transformation for all range blocks can be iteratively applied to any initial image which then converges to the fractal approximation of the image.

3. Robust M-Estimator

The method of least squares is a prototypical M-estimator [57] since the estimator is defined as a minimum of the sum of squares of the residuals. However, the ordinary least square method is very sensitive to outlying observations. There are several approaches to deal with this problem. One approach is robust regression, which is to employ a fitting criterion that is not as vulnerable as least squares to unusual data. Statisticians have developed many robust methods based on different techniques. The most common general method of robust regression is M-estimation, introduced by Huber. Consider the general regression model

\[
y_i = x_i^T \beta + \epsilon_i, \quad i = 1, 2, \ldots, n
\]

(9)

for the \( i \)-th of \( n \) observations, where \( x_i^T = [1 \ x_1 \ x_2 \ \cdots \ x_{ik}] \) and \( \beta^T = [\alpha \ \beta_1 \ \beta_2 \ \cdots \ \beta_k]^T \). Here, \( \epsilon_i \) is an unobserved random variable, which is added to the linear relationship between the dependent variable \( y_i \) and the vector of regressors \( x_i \).

The general M-estimator is defined to minimize the objective function

\[
E = \sum_{i=1}^{n} \rho(\epsilon_i) = \sum_{i=1}^{n} \rho(y_i - x_i^T \beta)
\]

(10)

where the function \( \rho(\cdot) \) is a symmetric positive definite function with a unique minimum at zero and is chosen to be increasing slower than quadratically.
Let \( \psi = \rho' \) be the derivation of \( \rho \). Differentiating the objective function with respect to the coefficients \( \beta \), and setting the partial derivatives to 0, produces \( k + 1 \) a system of estimating equations for the coefficients

\[
\sum_{i=1}^{n} \psi(y_i - x_i^T \beta) x_i^T = 0
\]  

(11)

Define the weight function \( w(x) = \psi(x) / x \), and let \( w_i = w(\epsilon_i) \). Then the estimating equations can be written as

\[
\sum_{i=1}^{n} w_i(y_i - x_i^T \beta) x_i^T = 0
\]  

(12)

This equation is a weighted least-squares problem, aiming at minimizing \( \sum_{i=1}^{n} w_i \epsilon_i \). An IRLS (iteratively reweighted least-squares) algorithm can be used.

At the iteration \( t \), the new weighted-least-squares estimates are computed by

\[
\beta^{(t)} = [X^T W^{(t-1)} X]^{-1} X^T W^{(t-1)} Y
\]  

(13)

where

\[
X = \begin{bmatrix}
x_1^T \\
x_2^T \\
\vdots \\
x_n^T
\end{bmatrix} = \begin{bmatrix}
1 & x_{11} & \cdots & x_{1k} \\
1 & x_{21} & \cdots & x_{2k} \\
\vdots & \vdots & \ddots & \vdots \\
1 & x_{n1} & \cdots & x_{nk}
\end{bmatrix}
\]

and \( W^{(t-1)} = \text{diag}\{w_i^{(t-1)}\} \) is the current weight matrix.

3.1. Huber’s M-Estimator

Huber first introduced a robust M-estimator [58], named Huber’s M-estimator. He chose the \( \rho \)-function to be formed by Equation (14)

\[
\rho_H(\epsilon) = \begin{cases} 
\epsilon^2 / 2, & \text{for } |\epsilon| \leq k \\
ke - \epsilon^2 / 2, & \text{for } |\epsilon| > k
\end{cases}
\]  

(14)

Here \( k \) is the tuning constant. In particular, \( k = 1.345\sigma \) for the Huber case [34], where \( \sigma \) is the standard deviation of the errors. A common approach [58] is to take

\[
\hat{\sigma} = \text{median}\{\text{abs}(e - \text{median}(e))\} / 0.6745
\]  

(15)

Huber’s M-estimator is a mixed \( l_2 \) and \( l_1 \) minimization problem, but LS is only a \( l_2 \) minimization problem (since in LS case, \( \rho(\epsilon) \) is always equal to \( \epsilon^2 / 2 \)). When there are outliers in the measurements, generally the corresponding residuals are relatively large. Since \( \rho_H(\epsilon_i) \) of Huber’s M-estimator with \( n \) beyond 7 is smaller than that of LS, Huber’s M-estimator is less sensitive to relatively large residuals. Hence it can suppress the influences of outliers.

From Equation (14), we obtain

\[
\psi(\epsilon) = \rho_H'(\epsilon) = \begin{cases} 
\epsilon, & \text{for } |\epsilon| \leq k \\
ke / |\epsilon|, & \text{for } |\epsilon| > k
\end{cases}
\]  

(16)

Hence, in the Huber case, the weight function, which we denote as \( w_H(\epsilon) \), is defined by Equation (17)

\[
w_H(\epsilon) = \psi(\epsilon) / \epsilon = \begin{cases} 
1, & \text{for } |\epsilon| \leq k \\
\epsilon / k, & \text{for } |\epsilon| > k
\end{cases}
\]  

(17)
Considering the general regression model (1), Let $\epsilon = e/\sigma$ and $w_i = w(\epsilon_i) = w(e/\sigma)$, where $\sigma$ is defined by Equation (15). The main steps of IRLS for the estimation of the parameter using the Huber M-estimator are described as Algorithm 1.

**Algorithm 1 IRLS using Huber M-estimator**

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>1: Select initial estimates $\hat{\beta}^{(0)}$, $\hat{\epsilon}^{(0)}$, and weight $w_i^{(0)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2: while $\hat{\beta}$ didn’t converge to a convergence value do</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3: Compute residuals by $\hat{e}^{(t-1)}_i = y_i - x_i^T \hat{\beta}^{(t-1)}$ and weights by Equation (17)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4: Compute the new weighted least squares estimates $\hat{\beta}^{(t)}$ by Equation (13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5: Compute the objective value $E$ based on the objective function by Equation (10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6: end while</td>
</tr>
</tbody>
</table>

3.2. Tukey’s Bisquare M-Estimator Figures, Tables and Schemes

The Tukey’s bisquare function can suppress the outlier even further. In Tukey’s bisquare case, $\rho$-function is defined by Equation (18) as follows:

$$
\rho_{TB}(e) = \begin{cases} 
\frac{2k^3}{t^3} (1 - [1 - (e/k)^2]^3), & \text{for } |x| \leq k \\
\frac{k^3}{e^3}, & \text{for } |x| > k
\end{cases}
$$

(18)

where, the tuning constant $t = 4.6851\sigma$ for Tukey’s Bisquare case in particular [34], where $\sigma$ is the standard deviation of the errors, and can be obtained by Equation (15).

The objective function is defined in Equation (10). Note that the Huber objective function increase without bound as the residual $e$ deviates from 0. In contrast, Tukey’s bisquare objective function levels eventually level off (for $|e| > k$). The weights for the Huber estimator decline when $|e| > k$, and the weights for Tukey’s bisquare are 0 for $|e| > k$.

Here $W_{TB}$ denotes the weight function of Tukey’s bisquare case.

$$
w_{TB}(e) = \begin{cases} 
[e[1 - (e/k)^2]^2, & \text{for } |e| \leq k \\
0, & \text{for } |e| > k
\end{cases}
$$

(19)

Considering the general regression model (1), Let $\epsilon = e/\sigma$ and $w_i = w(\epsilon_i) = w(e/\sigma)$. The main steps of IRLS for the estimation of the parameter using Tukey’s bisquare M-estimator are described as Algorithm 2.

**Algorithm 2 The IRLS algorithm using Tukey’s bisquare M-estimator**

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>1: Select initial estimates $\hat{\beta}^{(0)}$, $\hat{\epsilon}^{(0)}$, and weight $w_i^{(0)}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>2: while $\hat{\beta}$ didn’t converge to a convergence value do</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3: Compute residuals by $\hat{e}^{(t-1)}_i = y_i - x_i^T \hat{\beta}^{(t-1)}$ and weights by Equation (19)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4: Compute the new weighted least squares estimates $\hat{\beta}^{(t)}$ by Equation (13)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5: Compute the objective value $E$ based on the objective function by Equation (10)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6: end while</td>
</tr>
</tbody>
</table>

4. Robust M-Estimator for FIC

To apply the general model (9) to the fractal coding process, a 2-D image block with $L \times L$ size can be rearranged as a 1-D vector, denoted as $d_1, d_2, \ldots, d_{L^2}$. The model can be represented by

$$
r_i = o + s \cdot d_i + \epsilon_i, i = 1, 2, \ldots, L^2
$$

(20)
Compared with the general regression model, $o$ and $s$ are taken as the parameters to be estimated, denoted $\hat{\beta} = [o \ s]^T$ as the parameter vector and the total number of regression data is equal to $L^2$. Let $\hat{\beta} = [\hat{o} \hat{s}]^T$ to be the estimated parameter vector. The residuals can be defined as follows.

$$e_i(\hat{\beta}) = r_i - \hat{o} - \hat{s} \cdot d_i, i = 1, 2, \ldots, L^2$$ (21)

The M estimate $\hat{\beta}$ minimizes the objective function of Equation (10), where $n = L^2$, $\varepsilon = e/\sigma$ and $\sigma$ is the tuning constant. The optimized scaling parameter $\hat{s}$ and offset parameter $\hat{o}$ can be obtained by the IRLS algorithm. We defined weights for residuals as follows:

$$w_i = w(e_i) = w(e_i/\sigma), i = 1, 2, \ldots, L^2,$$

where, the $w(\cdot)$ function is defined by Equations (17) and (19), respectively, standing for Huber and Tukey’s statistics.

Thus, the estimated parameter $\hat{\beta}$ can be calculated via Equation (13) as

$$\hat{\beta} = \begin{bmatrix} \hat{o} \\ \hat{s} \end{bmatrix} = (X^T WX)^{-1} X^T WY$$ (23a)

$$X^T WX = \begin{bmatrix} 1 & d_1 \\ \vdots & \vdots \\ 1 & d_{l^2} \end{bmatrix}^T \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & w_{l^2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} 1 & d_1 \\ \vdots & \vdots \\ 1 & d_{l^2} \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{l^2} w_{i} l^2 \\ \sum_{i=0}^{l^2} w_{i} d_{i} l^2 \\ \sum_{i=0}^{l^2} w_{i} d_{i} l^2 \\ \sum_{i=0}^{l^2} w_{i} d_{i} l^2 \end{bmatrix}$$ (23b)

And

$$X^T WY = \begin{bmatrix} 1 & d_1 \\ \vdots & \vdots \\ 1 & d_{l^2} \end{bmatrix}^T \begin{bmatrix} w_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & w_{l^2} & \cdots & 1 \end{bmatrix} \begin{bmatrix} r_1 \\ \vdots \\ r_{l^2} \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{l^2} w_{i} r_{i} l^2 \\ \sum_{i=0}^{l^2} w_{i} d_{i} r_{i} l^2 \\ \sum_{i=0}^{l^2} w_{i} d_{i} r_{i} l^2 \end{bmatrix}$$ (23c)

Thus, we can obtain the estimated parameter in the IRLS iterated process directly by

$$\hat{\beta}^{(t)} = \begin{bmatrix} \hat{o}^{(t)} \\ \hat{s}^{(t)} \end{bmatrix} = \begin{bmatrix} l^2 \\ \sum_{i=0}^{l^2} w_{i}^{(t-1)} \\ \sum_{i=0}^{l^2} w_{i}^{(t-1)} d_{i} l^2 \\ \sum_{i=0}^{l^2} w_{i}^{(t-1)} d_{i} r_{i} l^2 \end{bmatrix}^{-1} \begin{bmatrix} l^2 \\ \sum_{i=0}^{l^2} w_{i}^{(t-1)} r_{i} l^2 \\ \sum_{i=0}^{l^2} w_{i}^{(t-1)} d_{i} r_{i} l^2 \end{bmatrix}$$ (24)

In the HFIC algorithm proposed by Jeng et al. [19], the Huber M-estimator is employed in the scaling parameter and offset parameter estimation. The principle of HFIC is as follows: for each isometrical transformation, find the optimized scaling and offset parameters by minimizing the objective function of Equation (10). After all eight isometrical transformations are applied, record the optimized parameters with the minimized objective.

Note that HFIC is very time-consuming since there will be eight times of iterated IRLS algorithm for one pair of range block $r$ and domain block $d$. The experimental results also confirm that it needs ten thousand seconds to finish a full-search Huber fractal coding. Here what we improved is eliminating the calculation during the obtaining of the optimized parameters to speed up the denoising FIC algorithm.

Although decreasing the domain’s searching scope can speed up the compression process, it will have a negative effect on the quality of the reconstruction image. To show the performance of a robust estimator, in this study, we focus on the full-search algorithm. Namely, for a given range block $r$, all domain blocks will be applied to the fractal affine transformation to find the optimized one with the minimized matching error.

Quadtree partitioning is a method used to represent an image as a hierarchical quadtree data structure in which the root of the tree is the initial image and each node contains four subnodes [4]. A node represents a square image portion and its four subnodes correspond to the four quadrants of the square image portion. Quadtree-based FIC is the
FIC algorithm used in quadtree partitioning. Since quadtree-based FIC is widely used in fractal compression algorithms, and it is already proven to be superior in performance over the fixed partitioning in the FIC algorithms, we chose quadtree-based FIC as the main frame of the proposed robust algorithm.

As in the analysis above, the most time-consuming parts of the original HFIC are the eight times iterated calculations on the parameter estimation. In the proposed algorithm, we decrease it to only once with almost no sacrifice on the quality. For each range block \( r \) and down-sampled domain block \( d \), in the preprocess stage, the four quadrants of the given block are ordered based on the brightness average value \( r' \). Then it is rotated to the one with the brightest quadrant in the upper left corner. After this preprocess, the matching error is compared between the range block \( r' \) and domain block \( d' \). Here \( r' \) and its quadrants are all with the brightest quadrant in the upper left, as is domain block \( d' \). Hence, in the proposed algorithm, the IRLS algorithm is applied during the matching between \( r' \) and \( d' \), instead of \( r \) and \( d \). Hence, the eight times of parameters estimation for each \( r \) and \( d \) is changed to be only once, which is between \( r' \) and \( d' \).

Algorithm 3 describes the main principle of the proposed decode algorithm.

<table>
<thead>
<tr>
<th>Algorithm 3 The Robust M-estimator for FIC algorithm (RMFIC)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input</strong> the original image, the tolerance value ( T ),</td>
</tr>
<tr>
<td><strong>Output</strong> the fractal codes</td>
</tr>
<tr>
<td>1: Pre-decimate the image into domain image and partition the original image into blocks with size of ( L \times L ).</td>
</tr>
<tr>
<td>2: Preprocess the blocks. Do flip or rotation to make the blocks with its brightest quadrant in its upper left corner. Record the index ( k ) of the transformation for each block and its quadrants</td>
</tr>
<tr>
<td>3: while there is a block didn’t encoded do</td>
</tr>
<tr>
<td>4: Fetch a range block ( r_j )</td>
</tr>
<tr>
<td>5: Obtain the optimal scaling parameter ( s ) and offset parameter ( o ) by IRLS algorithm and record its corresponding fractal codes</td>
</tr>
<tr>
<td>6: Calculate the matching error ( E_i ) by Equation (10).</td>
</tr>
<tr>
<td>7: if ( E_i &lt; T ) or the block size of the ( d_i ) is already small enough then</td>
</tr>
<tr>
<td>8: go to 13</td>
</tr>
<tr>
<td>9: else</td>
</tr>
<tr>
<td>10: Partitioning the current block into 4 sub-blocks (quadrants) by size of ( L_i/2 \times L_i/2 )</td>
</tr>
<tr>
<td>11: go to 5</td>
</tr>
<tr>
<td>12: end if</td>
</tr>
<tr>
<td>13: Record the fractal codes with the minimum matching error.</td>
</tr>
<tr>
<td>14: end while</td>
</tr>
</tbody>
</table>

Two M-estimators are chosen, viz, Huber and Tukey’s bisquare statistics. The respective IRLS algorithms are described in Sections 3.1 and 3.2.

5. Experimental Results

In order to evaluate the performance of this proposed algorithm (RMFIC), a computer simulation is performed. The standard gray-level image of Lena and Boats with 512 × 512 pixels size was used. The coding block size in the RMFIC is set to be 32 × 32, 16 × 16, or 8 × 8 using the quadtree partitioning method, and the tolerance value \( T = 8 \).

For the proposed algorithm based on Huber’s M-estimator (RMFIC-H), we adopt the objective function of Equation (10) as the matching error function. The initial values of the estimated parameters are obtained by Least Squares. The tuning constant parameter \( k = 1.345\sigma \), and in the study case, \( \sigma \) is defined to be fixed value 5. For the proposed algorithm based on Tukey’s bisquare M-estimator (RMFIC-TB), we adopt MSE (mean-square-error) in Equation (25), as the matching error function. The initial values of the estimated parameters are obtained by Least Squares regression. The tuning constant parameter \( t = 4.6851\sigma \), and \( \sigma \) is defined by Equation (15). We use a blank image as the initial image for decoding and the iteration number is 20, which is sufficient to reconstruct the image. The simulation
is done under the Microsoft Visual C++ 2008 on Windows XP with SP3, Intel Core2 Duo 2.53 GHz CPU, and 2G RAM platform.

$$\text{MSE} = \frac{1}{N} \sum_{i=1}^{N} (f_i - \hat{f}_i)^2$$  \hspace{1cm} (25)

where $f_i$ is the grey value of the pixel $i$ in the original image and $\hat{f}_i$ is the gray level of the pixel $i$ in the retrieved image. $N$ denotes the total number of pixels.

The distortion between the original image $f$ and the retrieved image $\hat{f}$ is measured by the peak signal-to-noise ratio (PSNR). The definition of PSNR is

$$\text{PSNR}(f, \hat{f}) = 10 \log_{10} \left( \frac{255}{\text{MSE}(f, \hat{f})} \right)$$  \hspace{1cm} (26)

To show the quality of the reconstruction of the image, we corrupt the Lena image and the Boats image with salt and pepper noise with noise level as high as 10%. Figure 1 illustrates the retrieved images from the corrupted Lena image by full search FIC with normal least square regression (FIC-LS) and proposed RMFIC using Huber’s and Tukey’s robust statistics (RMFIC-H and RMFIC-TB). Figure 1a,b show the original Lena image and corrupted Lena image with a 10% noise level, respectively. Figure 1c–e illustrate the retrieved image from full search FIC-LS, RMFIC-H, and RMFIC-TB methods, respectively, with corresponding qualities of 23.31 db, 28.10 db and 26.46 db by PSNR for Lena corrupted image. Figure 2 is the test case on Boats image which is corrupted by salt and pepper noise with a 10% noise level. Figure 2a,b show the original Boats image, and corrupted Boats image. Figure 2c shows the retrieved image from full search FIC with normal least squares regression with 22.68db by PSNR. Figure 2d,e illustrate the retrieved image from proposed methods (RMFIC) with corresponding qualities of 25.57 db and 24.56 db by PSNR for Boats corrupted image. From the experimental results, we can form the conclusion that the RMFIC algorithm using both the Huber case and Tukey’s bisquare case show the ability of robustness for FIC. Namely, it is much more insensitive to outliers, compared to LS-based full search FIC. Note that there are several tiny blocks not reconstructed correctly from the retrieved image from RMFIC-TB by vision. The existence of these tiny blocks is due to the definition of Tukey’s bisquare M-estimator. From the weight Equation (19), we can see that if $|e| > k$, the weight value is zero. That means if the outlier is “detected”, the bisquare estimator can get rid of this value from the data set. However, in the same situation, Huber’s estimator has set the weight value to 1. Thus, the Tukey’s bisquare M-estimator sacrifices some data. When encoding the small blocks by fractal, there is a case where pixels are taken as outliers and the weight is taken as zero. That is why these tiny incorrect retrieved blocks exist. Further optimization for this case is required in the future.

Table 1 shows the simulation comparative results on Lena’s image with salt and pepper noise of 5%, 10%, 15%, 20%, and 30% noise levels, respectively. Both these algorithms are with full search schema. As we discussed in Section 4, the most time-consuming part of the original HFIC is the eight times of iterated IRLS algorithm for one pair of range block $r$ and domain block $d$. To encode the noisy Lena image in Figure 1b with the full search HFIC method, requires 9704 s. By eliminating the unnecessary calculations on the original HFIC algorithm, the encoding time of the proposed RMFIC-H (using Huber M-estimator) was reduced almost 87.5%, and its relative reconstructed image quality is improved since the quadtree partitioning method is adopted. Even when the noise level is up to 30%, the proposed RMFIC-H method still shows a better quality by PSNR. RMFIC-TB (using Tukey’s bisquare M-estimator) also shows considerable robustness, compared to the original FIC algorithm with normal least squares regression. Compared to the existing robust fractal-based algorithms, such as HFIC and LAD-FIC, RMFIC-TB shows a similar performance on the reconstructed image from the noisy image. However, the encoding time of RMFIC-TB is not optimized enough, and further work is still required in the future.
as zero. That is why these tiny incorrect retrieved blocks exist. Further optimization for this case is required in the future.

**Figure 1.** Retrieved images by Full search FIC-LS, RMFIC-H, and RMFIC-TB for Lena image corrupted by 10% salt and pepper noise. (a) Original image, (b) Corrupt image, (c) FIC-LS 23.31 db/9.70 s, (d) RMFIC-H 28.10 db/973.45 s, (e) RMFIC-TB 26.46 db/5670.20 s.

**Figure 2.** Retrieved images by Full search FIC-LS, RMFIC-H, and RMFIC-TB for Boats image corrupted by 10% salt and pepper noise. (a) Original image, (b) Corrupt image, (c) FIC-LS 22.68 db/9.156 s, (d) RMFIC-H 25.57 db/1100.34 s, (e) RMFIC-TB 24.56 db/5832.00 s.
Table 1. Performance of full search FIC-LS, HFIC, LAD-FIC, RMFIC-H, and RMFIC-TB with salt and pepper noise.

<table>
<thead>
<tr>
<th>Noise Level (%)</th>
<th>FIC-LS</th>
<th>HFIC</th>
<th>LAD-FIC</th>
<th>RMFIC-H</th>
<th>RMFIC-TB</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSNR(db)</td>
<td>Time(s)</td>
<td>PSNR(db)</td>
<td>Time(s)</td>
<td>PSNR(db)</td>
</tr>
<tr>
<td>5%</td>
<td>25.19</td>
<td>9.625</td>
<td>27.92</td>
<td>7013.5</td>
<td>27.20</td>
</tr>
<tr>
<td>10%</td>
<td>23.31</td>
<td>9.704</td>
<td>27.1</td>
<td>7288.1</td>
<td>26.47</td>
</tr>
<tr>
<td>15%</td>
<td>22.09</td>
<td>9.563</td>
<td>25.82</td>
<td>8024.0</td>
<td>25.97</td>
</tr>
<tr>
<td>20%</td>
<td>21.09</td>
<td>9.969</td>
<td>24.20</td>
<td>8311.7</td>
<td>24.23</td>
</tr>
<tr>
<td>30%</td>
<td>19.39</td>
<td>9.735</td>
<td>21.55</td>
<td>8698.9</td>
<td>21.40</td>
</tr>
</tbody>
</table>

For the bell-shaped noises, such as zero-mean Gaussian noise and Laplace noise, the original HFIC algorithm does not show significant robustness under them. We take a simple examination for zero-mean Gaussian noise and the Lena image of 10% noise level to test, by the baseline FIC (LS). And the reconstructed image can obtain PSNR of 26.48, by RMFIC-H algorithm, the PSNR is 26.40 and by RMFIC-TB, this value is 26.43.

6. Conclusions

In this paper, an improved robust fractal encoding method based on M-estimator has been proposed (RMFIC). This proposed algorithm applies the maximum likelihood estimation technique to the quadtree-based fractal image compression procedure. Improvement is made by reducing the unnecessary computations for the parameter estimation procedure. Both Huber’s and Tukey’s bisquare M-estimators are used in this work. The proposed methods show good robustness against salt and pepper noise. Comparative experimental results are given for the proposed method and the existing robust fractal encoding method. The encoding time of RMFIC-H is reduced by almost 87.5%, compared with the original full search HFIC, and its retrieved image quality is better than other robust algorithms such as HFIC and LAD-FIC. The RMFIC-TB also shows considerable robustness.

However, both M-estimators do not make a significant improvement in image quality for Gaussian noise, so further research is still required in the future. In addition, the proposed method only considered the normal raw inputting image, and then added the noise. However, the problem of using the raw corrupted blurred image as the original image has gradually attracted the attention of scholars and has become an important research topic of image compression recently. This will be another topic that the proposed method will further study.

Author Contributions: Conceptualization, P.H. and D.L.; methodology, P.H.; software, H.Z.; validation, H.Z. and P.H.; investigation, P.H.; resources, D.L.; data curation, D.L.; writing—original draft preparation, P.H.; writing—review and editing, D.L.; visualization, D.L.; supervision, H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by NSFC, grant number 51605068.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References


18. Wu, J.; Wang, Z. A hybrid model for water quality prediction based on an artificial neural network, wavelet transform, and long short-term memory. *Water* 2022, 14, 610. [CrossRef]


27. He, Z.; Shao, H.; Lin, J.; Cheng, J.; Yang, Y. Transfer fault diagnosis of bearing installed in different machines using enhanced deep auto-encoder. *Measurement* 2020, 152, 107393. [CrossRef]


33. Cui, H.; Guan, Y.; Chen, H. Rolling element fault diagnosis based on VMD and sensitivity MCKD. *IEEE Access* 2021, 9, 120297–120308. [CrossRef]
34. Zhang, K.; Huang, Q.J.; Zhang, Y.M. Enhancing comprehensive learning particle swarm optimization with local optima topology. *Inf. Sci.* 2019, 471, 1–18. [CrossRef]
38. Liu, Q.; Jin, T.; Zhu, M.; Tian, C.; Li, F.; Jiang, D. Uncertain currency option pricing based on the fractional differential equation in the Caputo sense. *Fractal Fractional* 2022, 6, 407. [CrossRef]
39. Li, G.; Li, Y.; Chen, H.; Deng, W. Fractional-order controller for course-keeping of underactuated surface vessels based on frequency domain specification and improved particle swarm optimization algorithm. *Appl. Sci.* 2022, 12, 3139. [CrossRef]
47. Saad, A.; Abdullah, M.; Alduais, N. Impact of spatial dynamic search with matching threshold strategy on fractal image compression algorithm performance: Study. *IEEE Access* 2020, 8, 52687–52699. [CrossRef]