RECEIVER-SIDE FINGERPRINTING METHOD FOR COLOR IMAGES
BASED ON A SERIES OF QUATERNION ROTATIONS

Abstract: The proposed method is a new Joint Fingerprinting and Decryption (JFD) method that uses a cipher based on quaternion rotation to encrypt color images that are then sent to all users via multicast transmission. Individual decryption keys depend on the users’ fingerprints, so that a unique fingerprint is introduced into the image during decryption for each decryption key. A simulation-based research was conducted to examine the method’s robustness against collusion attacks.

1. INTRODUCTION

This paper addresses the problem of unauthorized redistribution of multimedia content by malicious users (pirates). There are two ways to protect distributed multimedia: encryption and digital fingerprinting [1]. The goal of encryption is to ensure that only authorized users with proper decryption keys are able to use distributed multimedia. However, after decryption the data loses its protection and may be illegally redistributed by malicious users. In order to maintain security after decryption it is necessary to use a digital fingerprint. Digital fingerprinting is a data hiding technique in which data is protected by unique sequences, called fingerprints. Each fingerprint identifies an individual user and is embedded in the image in such a way that it is imperceptible to the human eye. If a pirate redistributes his or her copy, the analysis of the embedded fingerprint should allow to identify the pirate.

Joint Fingerprinting and Decryption (JFD) [2,3,4,5] methods combine encryption and fingerprinting through embedding fingerprints during decryption process. The distribution side encrypts multimedia by using the encryption key and then sends the encrypted data via multicast transmission. Each user has a unique decryption key which is different from the encryption key and introduces some minor changes into decrypted images. These changes are imperceptible to the human eye and are unique for each user, hence they are the user’s fingerprint. The only operation performed on the distribution side is encryption. Moreover, data is encrypted only once, regardless of the number of users. At the same time the only operation performed on the receiving side is decryption, which simultaneously embeds a fingerprint into the image. Most importantly, only one copy of the data is sent over the network, which leads to a minimum demand for bandwidth. The above properties make JFD methods highly scalable. Note that in JFD methods the priority is the pirate tracing, while the encryption part is only to take advantage of multicast transmission rather than to provide a high security level.

This paper presents a new JFD method for color images. In the proposed method three color channels, based on RGB color model, form a 3D space and each component of the image is a point in this 3D space. The distribution side encrypts perceptually most essential components of the image with the encryption key and then sends the encrypted data via multicast transmission to all users. An encryption involves a series of rotations in 3D color space. All calculations are performed using quaternions which are 4-dimensional complex numbers, and they are very convenient for 3D rotations. Each user has a unique decryption key, which is different from the encryption key. The differences between the common encryption key and the individual decryption keys enable joint fingerprinting and decryption.

While image processing done by a single pirate can be a way of removing embedded fingerprint, a collusion attack is far greater threat to fingerprinting. A collusion attack [1,6,7] is performed when a group of pirates analyze their fingerprinted copies of the same data and together generate a pirate copy with a highly damaged fingerprint. The greatest attention should be paid to an linear collusion attack by averaging copies because in this attack the energy of each pirate’s fingerprint is reduced by the same factor, thus the risk of being identified is evenly shared among the pirates, and the visual quality of the multimedia content is not degraded. Moreover, most other fingerprinting methods have used this type of collusion attack in their studies, so it has become a commonly accepted tool for comparison between different fingerprinting methods. Thus, the robustness against linear collusion attack by averaging is the main focus in this paper. Due to the limited volume of the article, nonlinear attacks will not be considered.

The structure of the paper is as follows. Related work significant to the development of the proposed method is listed in Section 2. Section 3 is a brief introduction to the quaternion algebra. Section 4 describes the proposed JFD method. Research results on robustness against collusion attacks are presented in Section 5. The conclusions are discussed in Section 6.
2. RELATED WORK

The concept of fingerprinting combined with decryption process was first introduced in the Chameleon [3] by Anderson and Manivatis. The Chameleon is a stream cipher for audio data in which fingerprints are embedded in the least significant bits of protected data. This method has several major drawbacks: it is vulnerable to noise and compression, it operates on the uncompressed data, and it is not robust against large collusions of pirates.

Later, Kundur and Karthnik’s method was introduced by the name of Joint Fingerprinting and Decryption [2]. In this method, an encryption is based on scrambling signs of discrete cosine transform (DCT) coefficients of the image. A decryption unscrambles only part of DCT coefficients and the remaining ones remain encrypted and their combination is the user’s fingerprint. Unfortunately, embedded fingerprints are visible to the human eye. It is possible to reduce the visibility of the fingerprints, but at the expense of detection efficiency. In addition, the method has low robustness against the collusion attacks. The name of this method is also used to describe the concept of fingerprinting method that uses one encryption key and multiple decryption keys to embed fingerprints into the data during decryption.

Another important JFD method is the Fingercasting designed by Adelsbach et al. [4] and later improved by Katzenbeisser et al. [8]. This is a generalization of the Chameleon cipher which embeds spread-spectrum watermarks into audio-visual content. The main difference is that the lookup-table entries are uniformly distributed random elements and that the XOR operation is replaced by a modular addition. The robustness of the Fingercasting method is based on robustness of Cox’s spread spectrum watermarking scheme [9].

Another related work is Parnes and Parvianen’s method [10]. In this method, the transmitting side generates two differently fingerprinted copies of multimedia data that is transmitted using two multicast streams to all receivers. Each packet of these two streams is encrypted by a different key. Each user is equipped with a unique set of decryption keys but each key is able to decrypt only one of two packets in a pair. Then, a unique combination of decrypted packets is the user’s fingerprint. Flaws of this method are low robustness against collusion attacks, very large number of decryption keys and the need to use two multicast transmissions.

Next JFD method significant for the development of the proposed method is the Hillcast [11,12], which is a simple block cipher based on matrix multiplication, i.e. the generalized Hill cipher. An image is encrypted with a group key and sent to all the users via multicast transmission. A decryption process with the unique decryption key embeds a fingerprint in the image. The Improved Hillcast [13,14] involves fingerprint embedding in the frequency domain by using the DCT coefficients instead of embedding in the spatial domain for greater robustness against signal processing. Research on the Hillcast method has led to the development of the robust matrix-based JFD method [5] that is characterized by high robustness against noise, compression, collusion attacks and collusion attacks combined with compression.

The first quaternion-based fingerprinting was introduced as additional feature in CBC quaternion encryption [15,16,17]. In this method, an encryption of images is based on quaternion rotation around quaternion key in spatial domain, i.e. calculations are done on pixel values. The fingerprinting feature is a side effect caused by patterns of errors that occurred in the decrypted image. This is a receiver-side fingerprinting although this is not a JFD method, hence, this fingerprinting method scales very poorly as the required bandwidth grows linearly with the number of users.

The first quaternion-based joint fingerprinting and decryption method was receiver-side fingerprinting based on quaternion sandwich product for color images [18]. In this method, each component of the color image is represented as a point in 3D space, which is formed by 3 color channels. Encryption and fingerprinting can be interpreted as rotation, scaling and translation of these points. All calculations are performed using quaternion calculus. The experimental results show that the method is robust against collusion attacks. Receiver-side fingerprinting based on quaternion sandwich product was the main inspiration for this paper.

To the best of our knowledge, the method proposed in this paper is the second JFD method based on quaternion calculus. The main differences between the proposed method and the previous [18] are as follows. There is a series of quaternion rotations instead of one in order to increase the complexity of the encryption algorithm. Quaternions of encryption and decryption keys are normalized to simplify calculations and to simplify the relationship between embedded fingerprints and the host image. Moreover, the cipher-quaternion chaining mode of encryption is not required to effectively obscure the image in encrypted form.

3. QUATERNIONS

Quaternions are an extension of complex numbers [19,20]. They have 4 dimensions: 1 real dimension and 3 imaginary dimensions. A quaternion \( q \) can be represented as follows:

\[
q = q_w + i \cdot q_i + j \cdot q_j + k \cdot q_k ,
\]

(1)

where \( q_w, q_i, q_j, \) and \( q_k \) are scalar numbers (components) and \( i, j, \) and \( k \) are imaginary unit values.

The sum of two quaternions is defined as follows:

\[
p + q = (p_w + q_w) + i \cdot (p_i + q_i) + j \cdot (p_j + q_j) + k \cdot (p_k + q_k).
\]

(2)

The product of two quaternions is defined as:

\[
p \cdot q = (p_w q_w - p_i q_i - p_j q_j - p_k q_k) +
+ i \cdot (p_i q_w + p_w q_i + p_j q_k - p_k q_j) +
+ j \cdot (p_j q_w + p_w q_j + p_k q_i - p_i q_k) +
+ k \cdot (p_k q_w + p_w q_k + p_i q_j - p_j q_i)
\]

(3)
where the rules of multiplication of imaginary units are as follows: \( i \cdot i = j \cdot j = k \cdot k = -1 \), \( i \cdot j = k \), \( j \cdot i = -k \), \( j \cdot k = i \), \( k \cdot j = -i \), \( k \cdot i = j \), and \( i \cdot k = -j \). As follows from (3) the product of quaternions is non-commutative.

The conjugate of quaternion is a quaternion with the changed signs of the imaginary components:

\[
q^* = q_w - i \cdot q_x - j \cdot q_y - k \cdot q_z. \tag{4}
\]

The norm of a quaternion is:

\[
|q| = \sqrt{q \cdot q^*} = \sqrt{q_w^2 + q_x^2 + q_y^2 + q_z^2}. \tag{5}
\]

To normalise a quaternion we divide each quaternion component \( q_w, q_x, q_y \) and \( q_z \) by the norm. The norm of a normalised quaternion is equal to 1.

The inverse of a quaternion is:

\[
q^{-1} = q^* / |q|^2 = q_w - i \cdot q_x - j \cdot q_y - k \cdot q_z / q_w^2 + q_x^2 + q_y^2 + q_z^2. \tag{6}
\]

Quaternions can represent many 3D transformations, such as reflections, scaling, translations and more. However, the main profit from the use of quaternions appears in 3D rotations. Rotations in 3D space using 3 scalar values are nonlinear, they have singularities and they are difficult to combine. To solve these problems we can model 3D rotations in a 4D space using quaternions [19,20].

For a rotation of a point in 3D space we can use the sandwich product:

\[
P_{out} = q \cdot P_{in} \cdot q^*. \tag{7}
\]

where \( q \) is a quaternion representing rotation (\( q \) is normalised), \( P_{in} \) is a quaternion representing point in 3D space before rotation (real part of \( P_{in} \) is equal to 0), and \( P_{out} \) is a quaternion representing point in 3D space after being rotated (real part of \( P_{out} \) is equal to 0).

4. PROPOSED METHOD

In the proposed method, there is one encryption key for all users and one decryption key per user. The image is encrypted once and sent to all the users via multicast transmission. Each user decrypts the received data and at the same time embeds his or her fingerprint into the image. The embedded fingerprint is based on the difference between the encryption key and the user’s decryption key.

The encryption key is constructed as follows:

\[
E = \begin{bmatrix}
  e_{1,1} & e_{1,2} & \cdots & e_{1,R_2} \\
  e_{2,1} & e_{2,2} & \cdots & e_{2,R_2} \\
  \vdots & \vdots & & \vdots \\
  e_{R_1,1} & e_{R_1,2} & \cdots & e_{R_1,R_2}
\end{bmatrix}, \tag{8}
\]

where \( E \) is the matrix of the encryption key, \( e_{r_1,r_2} \) is a normalized quaternion with random scalar components, and \( r_1 = 1,2,\ldots,R_1, r_2 = 1,2,\ldots,R_2 \).

The fingerprint for \( u \)-th user is constructed as follows:

\[
F^{(u)} = \begin{bmatrix}
  f_{1,1}^{(u)} & f_{1,2}^{(u)} & \cdots & f_{1,R_2}^{(u)} \\
  f_{2,1}^{(u)} & f_{2,2}^{(u)} & \cdots & f_{2,R_2}^{(u)} \\
  \vdots & \vdots & & \vdots \\
  f_{R_1,1}^{(u)} & f_{R_1,2}^{(u)} & \cdots & f_{R_1,R_2}^{(u)}
\end{bmatrix}, \tag{9}
\]

where \( F^{(u)} \) is the matrix of the fingerprint for the \( u \)-th user, \( f_{r_1,r_2}^{(u)} \) is a quaternion with random scalar components of values \([-1; +1]\), e.g. in our implementation fingerprints were taken from bipolar Gold codes [21], \( r_1 = 1,2,\ldots,R_1, r_2 = 1,2,\ldots,R_2, \) \( u = 1,2,\ldots,U \) and \( U \) is the number of all the users.

The decryption key for \( u \)-th user is constructed as follows:

\[
D^{(u)} = \begin{bmatrix}
  d_{1,1}^{(u)} & d_{1,2}^{(u)} & \cdots & d_{1,R_2}^{(u)} \\
  d_{2,1}^{(u)} & d_{2,2}^{(u)} & \cdots & d_{2,R_2}^{(u)} \\
  \vdots & \vdots & & \vdots \\
  d_{R_1,1}^{(u)} & d_{R_1,2}^{(u)} & \cdots & d_{R_1,R_2}^{(u)}
\end{bmatrix}, \tag{10}
\]

\[
d_{r_1,r_2}^{(u)} = \frac{e_{r_1,r_2} \cdot (1 + \alpha \cdot f_{r_1,r_2}^{(u)})}{|e_{r_1,r_2} \cdot (1 + \alpha \cdot f_{r_1,r_2}^{(u)})|}, \tag{11}
\]

where \( D^{(u)} \) is the matrix of the decryption key for the \( u \)-th user, \( d_{r_1,r_2}^{(u)} \) is a normalized quaternion of the decryption key, \( e_{r_1,r_2} \) is a quaternion of the encryption key, \( f_{r_1,r_2}^{(u)} \) is a quaternion of the fingerprint, \( \alpha \) is the embedding strength factor which determines the balance between the imperceptibility and the robustness of embedded fingerprints, \( r_1 = 1,2,\ldots,R_1, r_2 = 1,2,\ldots,R_2, \) \( u = 1,2,\ldots,U \) and \( U \) is the number of all the users.

In order to encrypt the data, the image has to be processed in a following manner. Firstly, a color image in RGB color representation is split into three sub-images, one per each color channel. Secondly, each sub-image is divided into 8 by 8 pixels blocks. Thirdly, the discrete cosine transform (DCT) is performed for each block independently. This operation converts 64 pixel values of each block into 64 DCT coefficients.

The fourth step is to select the most perceptually essential coefficients for an encryption. Encrypting all coefficients will effectively conceal the image. However, it is highly recommended to encrypt only the perceptually essential components of the image because this lowers computation and delay and is enough to prevent image estimation with good quality. This concept is called partial encryption [22]. The perceptually essential coefficients are clustered in the upper left corner of each block. However, the DC coefficient (in 1st row and 1st column of each block)
should not be selected because it could significantly degrade the image quality after fingerprint embedding. In order to spread a fingerprint over the entire image, the coefficients should be selected from as many blocks as possible. As a result, three matrices \( X^{(\text{red})} \), \( X^{(\text{green})} \), and \( X^{(\text{blue})} \) are formed from coefficients of each color channel:

\[
X^{(\text{red})} = \begin{bmatrix}
X_{1,1}^{(\text{red})} & X_{1,2}^{(\text{red})} & \cdots & X_{N_2}^{(\text{red})} \\
X_{2,1}^{(\text{red})} & X_{2,2}^{(\text{red})} & \cdots & X_{2,N_2}^{(\text{red})} \\
\vdots & \vdots & \ddots & \vdots \\
X_{N_1,1}^{(\text{red})} & X_{N_1,2}^{(\text{red})} & \cdots & X_{N_1,N_2}^{(\text{red})}
\end{bmatrix},
\]

(12)

\[
X^{(\text{green})} = \begin{bmatrix}
X_{1,1}^{(\text{green})} & X_{1,2}^{(\text{green})} & \cdots & X_{N_2}^{(\text{green})} \\
X_{2,1}^{(\text{green})} & X_{2,2}^{(\text{green})} & \cdots & X_{2,N_2}^{(\text{green})} \\
\vdots & \vdots & \ddots & \vdots \\
X_{N_1,1}^{(\text{green})} & X_{N_1,2}^{(\text{green})} & \cdots & X_{N_1,N_2}^{(\text{green})}
\end{bmatrix},
\]

(13)

\[
X^{(\text{blue})} = \begin{bmatrix}
X_{1,1}^{(\text{blue})} & X_{1,2}^{(\text{blue})} & \cdots & X_{N_2}^{(\text{blue})} \\
X_{2,1}^{(\text{blue})} & X_{2,2}^{(\text{blue})} & \cdots & X_{2,N_2}^{(\text{blue})} \\
\vdots & \vdots & \ddots & \vdots \\
X_{N_1,1}^{(\text{blue})} & X_{N_1,2}^{(\text{blue})} & \cdots & X_{N_1,N_2}^{(\text{blue})}
\end{bmatrix},
\]

(14)

where \( x_{n_1,n_2}^{(\text{red})} \), \( x_{n_1,n_2}^{(\text{green})} \), and \( x_{n_1,n_2}^{(\text{blue})} \) are selected DCT coefficients from red, green, and blue channel respectively, \( n_1 = 1,2,\ldots,N_1 \) and \( n_2 = 1,2,\ldots,N_2 \) is the number of selected coefficients from each color channel.

The fifth step is to form a matrix of quaternions which will be encrypted:

\[
X = \begin{bmatrix}
x_{1,1} & x_{1,2} & \cdots & x_{N_1,N_2} \\
x_{2,1} & x_{2,2} & \cdots & x_{2,N_1,N_2} \\
\vdots & \vdots & \ddots & \vdots \\
x_{N_1,1} & x_{N_1,2} & \cdots & x_{N_1,N_1,N_2}
\end{bmatrix},
\]

(15)

\[
x_{n_1,n_2} = 0 + x_{n_1,n_2}^{(\text{red})}i + x_{n_1,n_2}^{(\text{green})}j + x_{n_1,n_2}^{(\text{blue})}k,
\]

(16)

where \( X \) is the matrix of the image, \( x_{n_1,n_2} \) is a quaternion representing a point in 3D space of color channels, \( n_1 = 1,2,\ldots,N_1 \) and \( n_2 = 1,2,\ldots,N_2 \) is the number of selected coefficients from each color channel.

The matrix of the image \( X \) is encrypted by using the encryption key \( E \) according to the formula:

\[
y_{n_1,n_2} = (\prod_{z=1}^{N_2} e_{f_1(z-\tilde{z}+1),f_2(z-\tilde{z}+1)}) \cdot (x_{n_1,n_2} + \beta \cdot \text{im}(e_{h_1,h_2})) \cdot (\prod_{z=1}^{N_2} e_{f_1(z-\tilde{z}+1),f_2(z-\tilde{z}+1)})^\ast,
\]

(17)

\[
h_1 = [(n_1 - 1) \mod R_1] + 1,
\]

(18)

\[
h_2 = [(n_2 - 1) \mod R_2] + 1,
\]

(19)

\[
g_1(z) = [(n_1 - 1 + z) \mod R_1] + 1,
\]

(20)

\[
g_2(z) = [(n_2 - 1 + z) \mod R_2] + 1,
\]

(21)

\[
Y = \begin{bmatrix}
y_{1,1} & y_{1,2} & \cdots & y_{N_1,N_2} \\
y_{2,1} & y_{2,2} & \cdots & y_{2,N_2} \\
\vdots & \vdots & \ddots & \vdots \\
y_{N_1,1} & y_{N_1,2} & \cdots & y_{N_1,N_2}
\end{bmatrix},
\]

(22)

where \( Y \) is the matrix of the encrypted image, \( y_{n_1,n_2} \) is an encrypted quaternion, \( x_{n_1,n_2} \) is a quaternion formed from selected DCT coefficients, \( e_{h_1,h_2} \) is a quaternion from the encryption key, \( \text{im}(\cdot) \) is a function which returns the imaginary part of the quaternion, \( \beta \) is an encryption parameter which increases the influence of the rotation by extending a distance between rotating points from the origin of 3D space, \( Z \) is a length of the series of rotations, \( z = 1,2,\ldots,Z \), \( n_1 = 1,2,\ldots,N_1 \), \( n_2 = 1,2,\ldots,N_2 \).

Afterwards, the encrypted DCT coefficients have to be extracted from quaternions \( y_{n_1,n_2} \) and put back in the same places in DCT blocks from which they were previously selected. All blocks are quantized for a fixed level of quality, set in a zigzag order, converted to a binary form and RLE/DPCM/Huffman-coded according to the JPEG standard. The encrypted image is sent to all users via one multicast transmission.

In order to decrypt the data, a receiver has to reverse the above-mentioned coding operations, and then extract the encrypted coefficients from the DCT blocks and recreate the \( Y \) matrix. The joint fingerprinting and decryption for the \( u \)-th user is performed according to the formula:

\[
x_{n_1,n_2}^{(u)} = (\prod_{z=1}^{N_2} d_{f_1(z-\tilde{z}+1),f_2(z-\tilde{z}+1)}) \cdot y_{n_1,n_2} \cdot (\prod_{z=1}^{N_2} d_{f_1(Z-z+1),f_2(Z-z+1)}) - \beta \cdot \text{im}(d_{h_1,h_2}^{(u)}),
\]

(23)

\[
h_1 = [(n_1 - 1) \mod R_1] + 1,
\]

(24)

\[
h_2 = [(n_2 - 1) \mod R_2] + 1,
\]

(25)

\[
g_1(z) = [(n_1 - 1 + z) \mod R_1] + 1,
\]

(26)

\[
g_2(z) = [(n_2 - 1 + z) \mod R_2] + 1,
\]

(27)

\[
X^{(u)} = \begin{bmatrix}
x_{1,1}^{(u)} & x_{1,2}^{(u)} & \cdots & x_{N_1,N_2}^{(u)} \\
x_{2,1}^{(u)} & x_{2,2}^{(u)} & \cdots & x_{2,N_1,N_2}^{(u)} \\
\vdots & \vdots & \ddots & \vdots \\
x_{N_1,1}^{(u)} & x_{N_1,2}^{(u)} & \cdots & x_{N_1,N_1,N_2}^{(u)}
\end{bmatrix},
\]

(28)

where \( X^{(u)} \) is the matrix of the decrypted and fingerprinted image for the \( u \)-th user, \( x_{n_1,n_2}^{(u)} \) is a
quaternion with the embedded fingerprint of the \( u \)-th user, \( d_{(u)}^{(u)} \), is a quaternion from the decryption key of the \( u \)-th user, \( u = 1, 2, \ldots, U \), and \( U \) is the number of all the users. The difference between \( E \) and \( D^{(u)} \), shown in the formula (11), embeds fingerprint into the decrypted image during the decryption process shown in the formula (23).

Afterwards, the decrypted and fingerprinted DCT coefficients have to be extracted from quaternions \( x_{(u)}^{(u)} \) and put back in the same places in DCT blocks from which they were previously selected. In the end, the receiver has to perform the inverse discrete cosine transform (IDCT) for each block independently and recreate an image by combining all of the blocks and all the color channels. As the result, the fingerprint spreads through all of the pixels in each block.

As it can be seen, in formulas from (17) to (22) and from (23) to (28), consecutive quaternions of the encryption key \( E \) and the decryption key \( D^{(u)} \) are selected along the diagonal. However, this is not the only approach. E.g., these quaternions may be selected in a pseudorandom way, but then the pseudorandom number generator and its seed, chosen on the transmitting side, must be known on the receiving side.

The example of image encryption and decryption for the proposed method is shown on Figure 1, which contains the original images of Lena and a bus, their encrypted and decrypted versions, as well as the fingerprints extracted from the decrypted images. Note that there are 16 DCT coefficient per block selected for the encryption of the image.

In case of the illegal redistribution of the image, i.e. the piracy, in order to perform the pirate tracing, the distribution side has to extract the fingerprint from the pirate copy of the image, and then run a correlation analysis. The extraction is performed as a non-blind detection according to the following formula:

\[
\mathbf{F}_p = \frac{1}{\alpha} (\mathbf{X}_p - \mathbf{X}),
\]

\[
\mathbf{F}_u^{(u)} = \frac{1}{\alpha} (\mathbf{X}^{(u)} - \mathbf{X}),
\]

where \( \mathbf{F}_p \) is the matrix of fingerprint embedded in the pirate’s image, \( \mathbf{F}_u^{(u)} \) is the matrix of the embedded fingerprint in the \( u \)-th user’s image, \( \mathbf{X} \) is the matrix of the image, \( \mathbf{X}_p \) is the matrix of the pirate image, \( \mathbf{X}^{(u)} \) is the matrix of the fingerprinted image of the \( u \)-th user, \( \alpha \) is the embedding strength factor, \( u = 1, 2, \ldots, U \) and \( U \) is the number of all the users. Note that the \( \mathbf{F}_u^{(u)} \) is different from \( \mathbf{F}^{(u)} \).

Then, the distribution side has to calculate correlation coefficients for the fingerprint extracted from the pirate copy \( \mathbf{F}_p \) and the fingerprints of all users \( \mathbf{F}_u^{(u)} \), \( u = 1, 2, \ldots, U \). If the calculated correlation coefficient for a specific user’s fingerprint is greater than a fixed detection threshold \( t_D \), this specific user is assumed a pirate.

Figure 2 shows the example of correlation analysis for pirate tracing in case of linear collusion attack of 40 pirates. For simplicity, the colluded pirates were users with 40 lowest IDs. It can be seen that correlation coefficients for the fingerprint extracted from the pirate copy and pirates’ fingerprints have much higher values than for innocent users’ fingerprints, which enables the pirate tracing with high credibility.

---

**Fig.1.** Original images of Lena and a bus (a), encrypted images (b) (with 16 DCT coefficient per block selected for the encryption), decrypted and fingerprinted images (c), extracted fingerprints (d).
The following scenario was simulated. A total of 500 users had ordered color images. Ten different color images of size 512 by 512 pixels were used which presented various natural objects, people, animals, etc. The fingerprints that were used were bipolar Gold codes [21] of length 4096 which were encapsulated in sequences of 1024 quaternions. The size of the encryption key was $32 \times 32$, and 1 DCT coefficient per block was selected for the encryption. The parameters $\alpha$ and $\beta$ were set to $\alpha = 0.0055$ and $\beta = 1500$. The average of PSNR of the fingerprinted images was 40.790 dB for the red color channel, 40.780 dB for the green color channel, and 40.788 dB for the blue color channel, with a width of 95%-confidence intervals equal to 0.0046 dB, so the embedded fingerprints remained imperceptible.

A distribution of each of the 10 images was repeated 5 times, each time for a different set of keys. This gave a total of 50 simulated distributions. For each of the 50 simulated distributions, 5 attacks occurred, i.e. collusion attacks by averaging the fingerprinted copies performed by 20, 40, 60, 80 and 100 pirates. Due to the limited volume of the article, nonlinear attacks will not be considered.

The correlation coefficients for the fingerprint from the pirate copy and each user’s fingerprint were compared with a detection threshold $t_D$. If the correlation coefficient value was greater than the threshold $t_D$, a user was considered a pirate. The true positive (TP) means that a pirate is correctly identified and false positive (FP) means that an innocent user is wrongly accused. Figure 4 contains (a) a plot of $P_{TP}$ against $t_D$, (b) a plot of $P_{FP}$ against $t$ and (c) the ROC (Receiver Operating Characteristics) curve, which is a plot of $P_{TP}$ against $P_{FP}$. Different line styles on the plots mean different numbers of pirates involved in a collusion attack.

The method’s effectiveness depends on the chosen acceptable $P_{FP}$. E.g., let us assume that the acceptable $P_{FP} = 0.01$, then $P_{TP} = 1$ in case of a collusion attack of 20 and 40 pirates, $P_{TP} = 0.975$ for 60 pirates, $P_{TP} = 0.882$ for 80 pirates, and 0.687 for 100 pirates. Therefore, every pirate can be effectively identified with a very high success rate, even if the number of pirates in the collusion is up to 60. Moreover, we achieved a faultless pirate identification in all of the simulations, i.e. $P_{TP} = 1$ and $P_{FP} = 0$, in case of a collusion of up to 40 pirates.

This is a very good result in comparison to other known methods. For example, the Chameleon cipher [3] is robust against up to 3 pirates in collusion. The Fingercasting scheme [4] is based on Cox’s watermarking scheme [9] which was tested for a collusion of only 5 pirates. Parviainen and Parnes’s fingerprinting method [10] is robust against up to 10 pirates in collusion. The Hillcast method [11,12] can identify up to 10 pirates with a high success rate. A fingerprinting feature in CBC quaternion encryption [17] can identify up to 30 pirates with a high success rate, although this is not a JFD method and it scales very poorly. Very similar results were obtained for the matrix-based JFD method [5] and the receiver-side fingerprinting method based on quaternion sandwich product [18]. The matrix-based JFD method is robust against collusion attacks done by up to 60 pirates [5] for $P_{TP} > 0.9$ and $P_{FP} = 0.01$, and up to 20 pirates for $P_{TP} = 1$ and $P_{FP} = 0$, although, that method doesn’t embed fingerprints in all color channels unlike the proposed

![Fig. 2. Correlation analysis for pirate tracing in case of linear collusion attack of 40 pirates.](image)

**5. RESEARCH RESULTS**

![Fig. 3. The method’s robustness against linear collusion attacks in case of 20, 40, 60, 80, 100 pirates.](image)
method. The receiver-side fingerprinting method based on quaternion sandwich product is robust against collusion attacks done by up to 80 pirates [18] for $P_{FP} > 0.9$ and $P_{FP} = 0.01$, and up to 40 pirates for $P_{FP} = 1$ and $P_{FP} = 0$. In comparison to these techniques and based on the experimental results we can claim that the proposed method has very high robustness against collusion attacks.

6. CONCLUSION

This paper presents a new joint fingerprinting and decryption method based on a series of quaternion rotations for color images. The proposed method uses a quaternion representation of image components in a 3D color space, and the concept of partial encryption, i.e. only the perceptually most significant DCT coefficients are selected for the encryption. Both encryption and fingerprinting are performed in the discrete cosine transform. After decryption, fingerprints are spread over the entire image, so they cannot be cut of the image.

In the paper, the RGB color model was used to determine the 3D color space. Obviously, it is possible to use other color models, e.g. YCbCr, although, the RGB color space is more convenient in that it allows to use the same embedding strength factor $a$ in each dimension. In case of YCbCr model, it would be necessary to modify the proposed method to establish different $a$ for luminance and chrominance. Futhermore, in case of YCbCr model, fingerprints would be more susceptible to JPEG/MPEG compression, because such compression has a strong impact on the chrominance components, which would cause significant distortions in 2/3 of the embedded fingerprints.

The research on the proposed method’s robustness against the linear collusion attacks was conducted and the results were presented in the paper. The method is robust against a collusion of up to 60 with a very high success rate ($P_{FP} > 0.9$ and $P_{FP} = 0.01$). In addition, a faultless pirate identification has been achieved ($P_{FP} = 1$ and $P_{FP} = 0$) in case of a collusion of up to 40 pirates. Although the paper only concerns fingerprinting for still images, the method can also be easily extended to the fingerprinting of video data.

Since the results are very promising, the development of the presented JFD method based on a series of quaternion rotations will continue. Future research will focus on robustness against nonlinear collusion attacks, as well as collusion attacks combined with compression.

REFERENCES


