Paper

Quaternion Feistel Cipher with an Infinite Key Space Based on Quaternion Julia Sets

Mariusz Dzwonkowski[1,2] and Roman Rykaczewski[1]

[1] *Faculty of Electronics, Telecommunications and Informatics, Department of Teleinformation Networks, Gdańsk University of Technology, Gdańsk, Poland*

[2] *Department of Radiological Informatics and Statistics, Medical University of Gdańsk, Gdańsk, Poland*

Abstract—In this paper Quaternion Feistel Cipher (QFC) with an infinite key space based on quaternion Julia sets is proposed. The basic structure of the algorithm is based on the scheme proposed in 2012 by Sastry and Kumar. The proposed algorithm uses special properties of quaternions to perform rotations of data sequences in 3D space for each of the cipher rounds. It also uses Julia sets to form an infinite key space. The plaintext is divided into two square matrices of equal size and written using Lipschitz quaternions. A modular arithmetic was implemented for operations with quaternions. A computer-based analysis has been carried out and obtained results are shown at the end of this paper.

Keywords—cryptography, lossless scheme, multimedia encryption, security.

1. Introduction

Quaternion encryption as presented in [1]–[3] uses the unique properties of quaternions in order to rotate vectors of data in a three-dimensional space. The concept, however, lacks general interest and is not popular, thus it is difficult to find a paper that would introduce any significant contribution to the field. However, an authors' papers [4]–[8] show different, possible implementations of quaternion encryption and they discuss the security aspect of each proposed algorithm.

The cipher proposed by the authors in [9] is a modification of the Feistel Cipher with the implementation of modular arithmetic. In this paper an another modification of the Feistel Cipher is proposed. In this modification, modular quaternion rotations are used to encrypt subsequent rounds without the need to use matrix multiplication as introduced in [9]. The proposed quaternion model features very fast computation advantages over its matrix-based counterpart in [9], thus this makes it a perfect match for encrypting multimedia. The specific properties of computations in the field of quaternions are covered more extensively in [10], [11]. Additionally, when encrypting a color image in RGB representation, it is possible to increase encryption efficiency even further because in that case a single quaternion can successfully store information about all three RGB channels.

It is important to note that the algorithm proposed here is part of an ongoing project, thus further studies on the method are necessary and are highlighted in the paper. The algorithm proposed here was originally designed to encrypt multimedia data, thus the security aspect is not the focus of this paper.

The paper is organized as follows. In Section 2 a brief introduction to quaternion calculus and quaternion Julia sets is provided. Section 3 describes the quaternion rotation concept as well as the application of quaternion Julia sets. Section 4 concerns the proposed encryption scheme with an illustration of the Quaternion Feistel Cipher. In Section 5 the simulation results are shown, the avalanche effects are discussed and the computation speed of the proposed algorithm with the AES algorithm is compared. Finally, in Section 6, the conclusions are drown.

2. Quaternion Calculus

Quaternions are hyper-complex numbers of rank 4 and have two parts – a scalar part and a vector part, which is an ordinary vector in a three-dimensional space $\mathbb{R}^3$. A quaternion q is defined by formula [10], [12]:

$$q = w + xi + yj + zk, \tag{1}$$

where w, x, y, z are real coefficients of quaternion q, and i, j, k are imaginary units with the following properties [10], [12]: $i^2 = j^2 = k^2 = ijk = -1$, $ij = -ji = k$, $jk = -kj = i$, $ki = -ik = j$. A quaternion could also be considered as a vector represented by a column matrix (all vectors in this paper are represented by column matrices) or as a composition of scalar part w and vector part $\vec{v}$

$$q = [w\ x\ y\ z]^T \quad or \quad q = (w,\ \vec{v}) = \left(w,\ [x\ y\ z]^T\right). \tag{2}$$

The sum of two quaternions q_1, q_2 is defined by adding the corresponding coefficients of those quaternions, i.e., in the same manner as for complex numbers [10], [13]:

$$q_1 + q_2 = (w_1 + w_2) + (x_1 + x_2)i + (y_1 + y_2)j + (z_1 + z_2)k. \tag{3}$$

The product of two quaternions is more complex due to the anti-commutativity of the imaginary units of quaternions. The product of the two quaternions q_1, q_2 consists of scalar

and vector products ("$\circ$" denotes the scalar product and "$\times$" denotes the vector product) [10], [13]:

$$q_1 \cdot q_2 = (w_1 w_2 - \vec{v}_1 \circ \vec{v}_2,\ w_1 \vec{v}_2 + w_2 \vec{v}_1 + \vec{v}_1 \times \vec{v}_2). \quad (4)$$

In this paper "$\cdot$" denotes the quaternion multiplication. Furthermore, it is important to define the other properties of quaternions: a conjugate q^*, a norm $\|q\|$ and an inverse q^{-1} of a quaternion q:

$$q^* = w - xi - yj - zk, \quad \|q\| = \sqrt{w^2 + x^2 + y^2 + z^2}, \quad (5)$$

$$q^{-1} = \frac{q^*}{\|q\|^2} = \frac{w - xi - yj - zk}{w^2 + x^2 + y^2 + z^2}. \quad (6)$$

It is important to notice that in the case of a unit quaternion, for which the norm is equal to 1, there is the following relation: $q^{-1} = q^*$.

2.1. Quaternion Julia Sets

Julia sets are produced by a procedure of repeated iterations [14], [15]. The polynomial used in the process of iteration is quadratic, cubic, quartic or any higher order degree [15]. A Julia set consists of all points $p \in \mathbb{C}$ for which a recursive sequence:

$$z_0 = p, \quad (7)$$

$$z_{n+1} = z_n^2 + c, \quad (8)$$

does not approach infinity. The parameter c in Eq. (8) is a complex number, i.e., a parameter determining the shape of the produced set. For a quaternion Julia set, the starting point p is determined in a three-dimensional space (1, i, j) for a constant dimension k. Parameter c is then considered a quaternion. An exemplary quaternion Julia set is shown in Fig. 1.

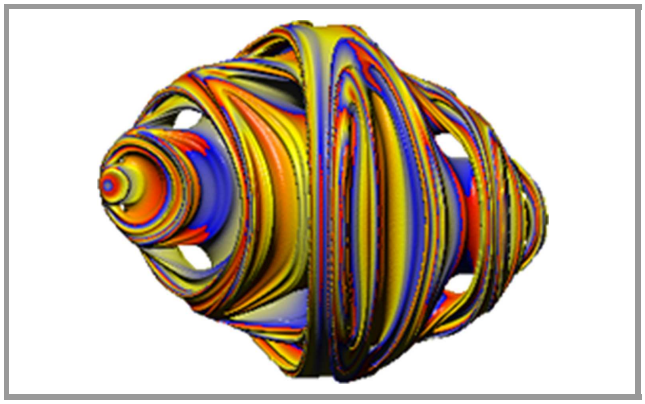


Fig. 1. Exemplary quaternion Julia set, number of iterations = 12, $c = 0.0882 + 0.1251i - 0.7555j + 0.1552k$, control number = 16, without an intersection plane.

In order to generate a quaternion Julia set, first must be set: all of the necessary initialization parameters, such as the number of iterations, to perform rule given by Eq. (8), the coefficients of quaternion c, a control number determining convergence of the starting points and the intersection plane.

3. Quaternion Rotation

The quaternion rotation can be performed by possessing a quaternion around which we will be rotating another quaternion. If the rotated quaternion as a data vector in three-dimensional space is considered, then the idea of quaternion encryption could be implemented.
Let us consider two quaternions $q = [w\ x\ y\ z]^T$ and $P = [0\ a\ b\ c]^T$, where a vector $[a\ b\ c]^T$, which represents a vector part of the quaternion P with a zero scalar part, will store information about a piece of data to be rotated around quaternion q. The obtained quaternion P_{rot} will be a spatial mapping of the rotated data vector $[a\ b\ c]^T$. The quaternion rotation is written as

$$P_{rot} = q \cdot P \cdot q^{-1}. \quad (9)$$

If we possess a tool, which can handle quaternion calculations, it is possible to implement encryption according to the Eq. (9).

3.1. Encryption Concept

The encryption method that was implemented in presented algorithm is entirely based on the quaternion rotation (9). It is possible to optimize the rotation process by extending the vector part of quaternion P in order to obtain a new quaternion B, as is shown in Formula (10):

$$P = \left(0, \begin{bmatrix} a \\ b \\ c \end{bmatrix}\right) \rightarrow B = \left(0, \begin{bmatrix} [a_1 & a_2 & a_3] \\ [b_1 & b_2 & b_3] \\ [c_1 & c_2 & c_3] \end{bmatrix}\right). \quad (10)$$

The encryption and decryption process for the quaternion method with the new extended quaternion B (meant to store data information) is shown in Eqs. (11) and (12), respectively.

$$B_{rot} = q \cdot B \cdot q^{-1}, \quad (11)$$

$$B = q^{-1} \cdot B_{rot} \cdot q, \quad (12)$$

where B_{rot} is the rotated (encrypted) quaternion B and q is the quaternion-key (encryption key).

3.2. Infinite Key Space

In order to generate an infinite key space it is necessary to first calculate a rotation matrix [1]–[3]. By using the Formula (9) and applying the Formulas (4)–(6) it is possible to introduce a rotation matrix [3], [12], [13], [16] and write:

$$\mathbf{P}_{\text{rot}} = \mathbf{\Gamma}(q)\mathbf{P}, \quad (13)$$

$$\mathbf{P} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}, \quad \mathbf{\Gamma}(q) =$$

$$\begin{bmatrix} w^2 + x^2 - y^2 - z^2 & 2xy - 2wz & 2xz + 2wy \\ 2wz + 2xy & w^2 - x^2 + y^2 - z^2 & 2yz - 2wx \\ 2xz - 2wy & 2yz + 2wx & w^2 - x^2 - y^2 + z^2 \end{bmatrix},$$

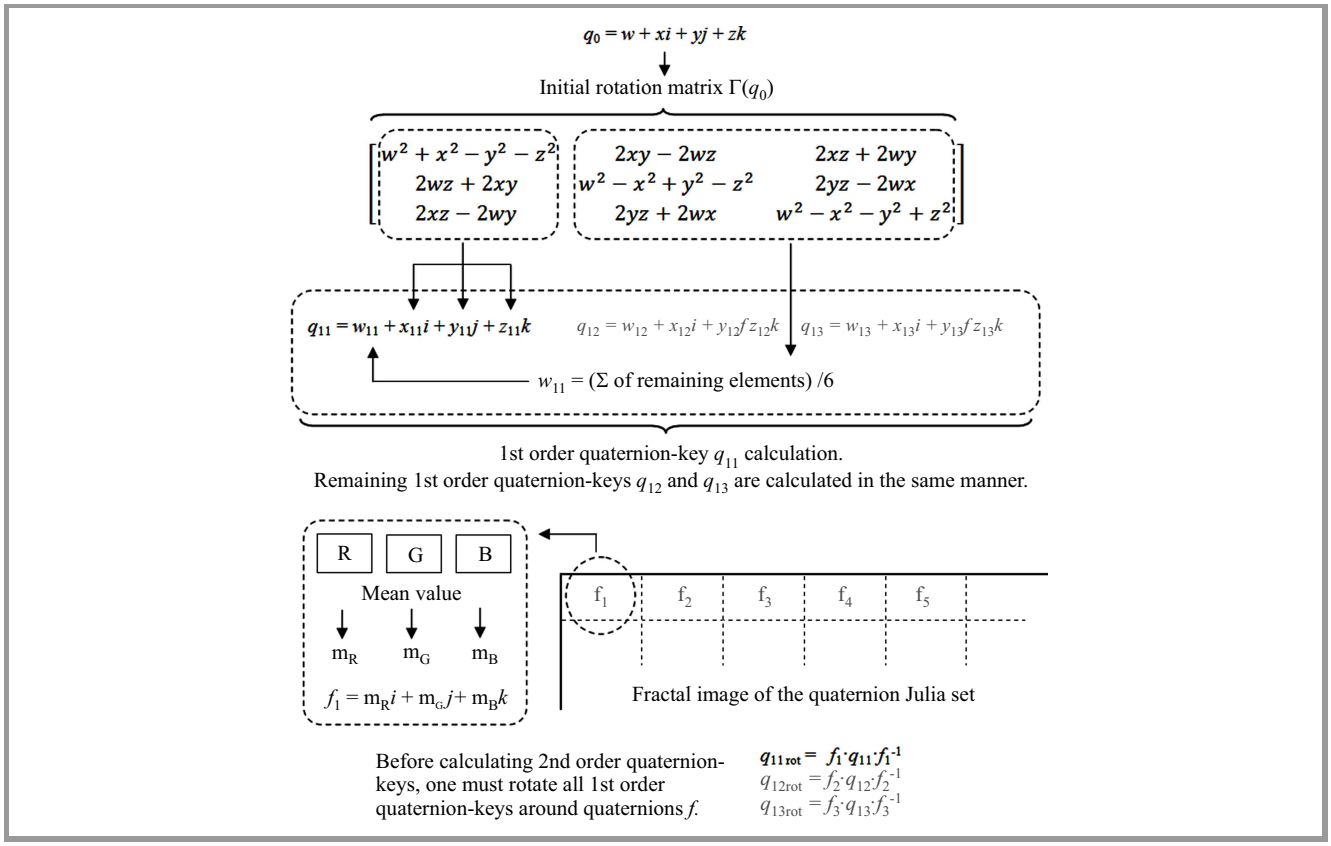


Fig. 2. Process of calculating 1st order quaternion-keys from an initial rotation matrix.

where $\mathbf{\Gamma}(q)$ is the rotation matrix calculated from the vector part of quaternion P_{rot}, which is defined by Formula (9). The rotation matrix is directly linked to quaternion q from which it was calculated.

The authors propose a key-generation algorithm based on concept [1], but additionally a quaternion Julia sets is introduced. The idea of the process is to treat elements of each column of the rotation matrix as coefficients (x, y, z) of subsequent quaternions from which other rotation matrices can be generated. In order to obtain coefficient w of subsequent quaternions, the mean value from six elements of the rotation matrix must be calculated, which were not used to determine coefficients (x, y, z), see Fig. 2.

Let us assume that the first quaternion q from which a rotation matrix was generated (13) is called an initial quaternion q_0. After grouping the elements of initial rotation matrix $\mathbf{\Gamma}(q_0)$ three quaternion-keys of 1st order (q_{11}, q_{12}, q_{13}) can be obrained. From these quaternions three rotation matrices can be generated, from which there is possibility to generate nine quaternion-keys of 2nd order. If we assume n as a rotation order, then the number of obtained quaternion-keys for the appropriate order n is equal to 3^n. The process is iterative, which means that in order to obtain higher order quaternion-keys we first need to define the rotation matrices of the lower orders.

However, before generating rotation matrices from quaternion-keys, we must first rotate every quaternion-key around quaternion f_i. Quaternions f_i are calculated from a random quaternion Julia set (Fig. 2).

The authors used the Quat generator [17] for visualization of quaternion Julia sets in 3D space as color images. The fractal image is divided into smaller fragments. The number of fragments is based on the size of the key space, e.g., for order = 2 nine quaternion-keys of 2nd order would be produced, thus we will need to divide the fractal image into 12 fragments (9 for quaternion-keys of 2nd order and 3 for quaternion-keys of 1st order). From each fragment a different quaternion f is calculated. Its coefficient w is always equal 0 and coefficients (x, y, z) represent a mean value calculated from the pixels values in the R, G and B channels of the fragments (Fig. 2).

The process of calculating 1st order quaternion-keys is shown in Fig. 2. The key generation process can be summarized by the following steps:

1) define q_0,
2) calculate $\mathbf{\Gamma}(q_0)$,
3) calculate $q_{11}\ q_{12}\ q_{13}$,
4) calculate $f_1 \cdot q_{11} \cdot f_1^{-1}\ \ f_2 \cdot q_{12} \cdot f_2^{-1}\ \ f_3 \cdot q_{13} \cdot f_3^{-1}$,
5) calculate $\mathbf{\Gamma}(q_{11\,rot})\ \mathbf{\Gamma}(q_{12\,rot})\ \mathbf{\Gamma}(q_{13\,rot})$,
6) calculate $q_{21}\ q_{22}\ q_{23}\ q_{24}\ q_{25}\ q_{26}\ q_{27}\ q_{28}\ q_{29}$,
7) ...

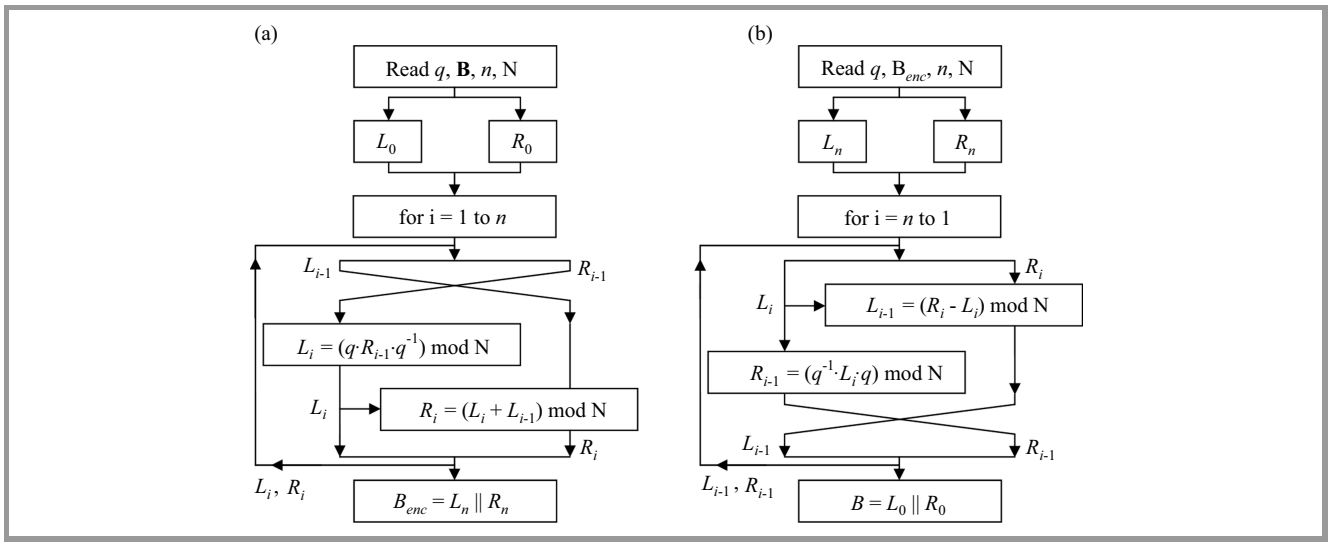


Fig. 3. Process of encryption (a) and decryption (b) for the proposed Quaternion Feistel Cipher.

4. Proposed Scheme

The proposed algorithm is designed to encrypt images (both color and gray-tone) but it can also be used to encrypt textual data. For the purpose of this paper an implementation for RGB color images is presented.

Let us now consider a plaintext B, which will be treated as RGB color image data. The plaintext can be written as three matrices **B**: $\mathbf{B_R}$, $\mathbf{B_G}$, $\mathbf{B_B}$. Each matrix **B** is of equal size with the image. Each element of all three matrices **B** is a value in the range of 0–255. For the purpose of the algorithm, all three matrices **B** should be rewritten as matrices with m rows and $2m$ columns, where m is calculated according to the rule:

$$m = \left\lceil \sqrt{\frac{width_B \cdot height_B}{2}} \right\rceil . \tag{14}$$

If the number of elements in such matrices exceeds the original amount of the images' pixels, then the additional elements are filled with random numbers in the range of 0–255. The three obtained matrices **B** ($m \times 2m$) are split into three square matrices: $\mathbf{L_{0R}}$, $\mathbf{L_{0G}}$, $\mathbf{L_{0B}}$ and three square matrices: $\mathbf{R_{0R}}$, $\mathbf{R_{0G}}$, $\mathbf{R_{0B}}$, each of size $m \times m$. All six square matrices are then written as components of two quaternions, L_0 and R_0 using the following rule:

$$L_0 = w_0 + x_0 i + y_0 j + z_0 k, \quad \text{where} \tag{15}$$
$$w_0 = 0, \quad x_0 = [\mathbf{L_{0R}}], \quad y_0 = [\mathbf{L_{0G}}], \quad z_0 = [\mathbf{L_{0B}}].$$

$$R_0 = w_0 + x_0 i + y_0 j + z_0 k, \quad \text{where} \tag{16}$$
$$w_0 = 0, \quad x_0 = [\mathbf{R_{0R}}], \quad y_0 = [\mathbf{R_{0G}}], \quad z_0 = [\mathbf{R_{0B}}].$$

The basic equations governing the encryption (17) and decryption (18) in proposed scheme are very similar in concept to the one presented in [9]. In this work, matrix multiplication is substituted by quaternion multiplication:

$$L_i = (q \cdot R_{i-1} \cdot q^{-1}) \mod N, \tag{17}$$
$$R_i = (L_{i-1} + L_i) \mod N \text{ for } i = 1 \text{ to } n,$$

$$R_{i-1} = (q^{-1} \cdot L_i \cdot q) \mod N, \tag{18}$$
$$L_{i-1} = (R_i - L_i) \mod N \text{ for } i = n \text{ to } 1.$$

The flowcharts depicting both the encryption and decryption processes of the proposed cipher are presented in Fig. 3.

According to Fig. 3, it should be noted that the symbol $\|$ is used for placing the vector part of a quaternion L adjacent to the vector part of quaternion R. The value n indicates the number of rounds in the cipher. Each round is encrypted with a different quaternion-key q_i, $i = 1, 2, \ldots, n$. The unique round keys are provided by the key generation algorithm (see Section 3.2).

4.1. Modular Arithmetic

Modular arithmetic operations were implemented in order to remain in the same field of values for data and cipher text. For that reason it was necessary to use Lipschitz integers (quaternions with integer components).

In order to calculate a modular inversion of any integer from the range of 0–255, and also a modular inversion of a Lipschitz integer, which is needed according to rotation rule (9), it was necessary to choose a special modulus value (a prime number) that together with all the integers in 0–255 would yield their Greatest Common Divisor (GCD) equal to 1. That is why for the RGB color images we cannot go for the most obvious choice and select a modulus of value 256, as such a number is not prime ($256 = 2^8$) and will not make it possible to calculate a modular inversion for most cases. Instead, a modulus equal to

prime integer N = 257 is used. Therefore values of the range 0–256 for all quaternions L_i and R_i can be obtained. It is important to note that despite the fact that in the encrypted image we obtained modified values from the range (0–256), the algorithm is so constructed that we will still be able to obtain the exact same image without any errors after the decryption process. The values obtained in the decryption process will be set into the appropriate range of 0–255.

The modular arithmetic with modulus 257 was implemented not only for the encryption/decryption process but also for the key generation algorithm (Section 3.2).

5. Simulation Results

The proposed scheme was scrutinized by computer-based simulation. The results of the encryption and decryption processes are shown in Fig. 4b and 4c, respectively.

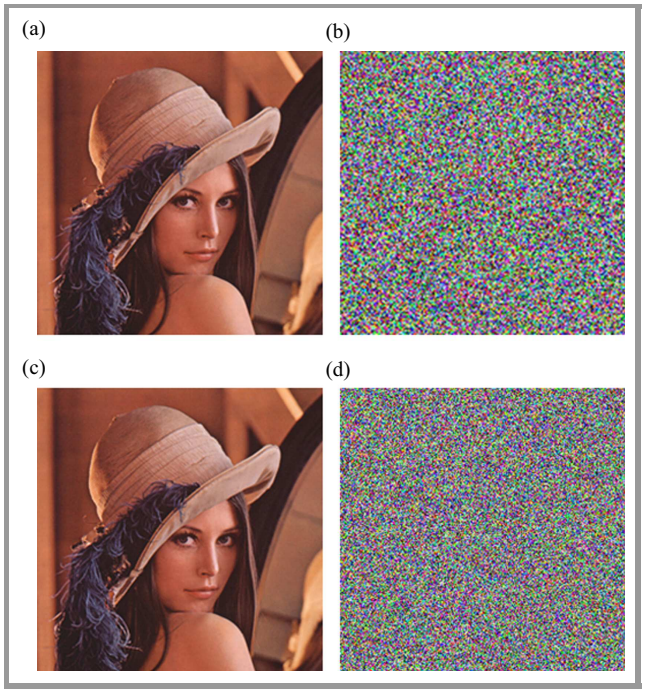


Fig. 4. Encryption and decryption of a color Lena image (https://en.wikipedia.org/wiki/Lenna) performed by the proposed quaternion cipher: (a) original image, (b) encrypted image, (c) decrypted image, (d) decrypted image using one key that is different in its binary form (1 bit) from the original one.

In Fig. 5, two histograms, i.e. of the original and the encrypted color Lena image, are shown. One can notice that the pixel values of the encrypted image are governed by a uniform distribution.

5.1. Randomness Tests

Chi-squared tests of randomness based on the diehard package [18] as well as on the freeware software Cryptool [19] were implemented in order to estimate the security level

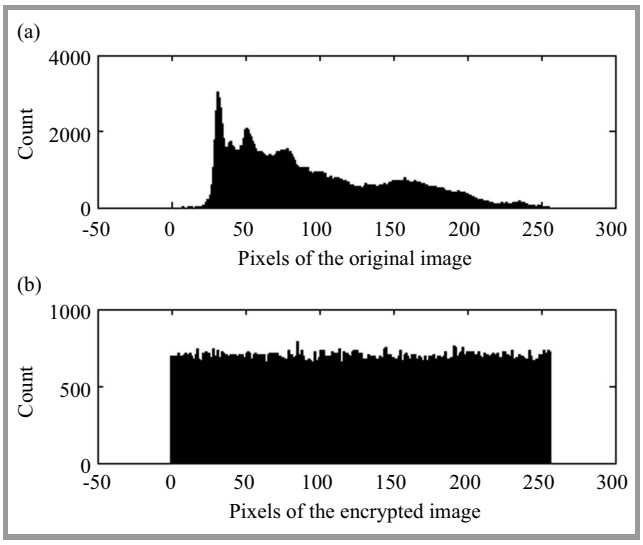


Fig. 5. Histograms for a color Lena image: (a) original image, (b) encrypted image.

of the proposed encryption model. The primary factor, which informs us about the effectiveness of the obtained randomness in encrypted data is a parameter called the p-value. Its values differ from 0 to 1, and the authors always aim for values as close to 0.5 as possible. If the p-value equals 0 or 1, that means that the encrypted data fails a particular randomness test. The obtained results for selected diehard randomness tests are shown in Table 1.

Table 1
p-values for different randomness tests from the diehard package

	Birthday spacings	Binary rank	Parking lot
p-values	0.613	0.585	0.209
	The 3Dsphere	Up-down runs	Craps
p-values	0.293	0.147	0.692
	DNA	Count-the-1's	OQSO
p-values	0.726	0.241	0.363
	OPERM5	Minimum distance	Overlapping sums
p-values	0.588	0.811	0.534

The authors also analyzed the randomness of proposed algorithm by using the Cryptool [19]. It offers 6 statistical tests: the Frequency Test, Poker Test, Runs Test, Serial Test and two additional tests embedded in Cryptool's battery test: Long-Run Test and Mono-Bit Test. Presented algorithm successfully passes all of them.

5.2. Avalanche Effect

In order to study the avalanche effect, let us consider a color Lena image as a plaintext. If 1 bit in one of the calculated

round keys (quaternion-keys of higher order) is changed, a new cipher text (encrypted image) will be obtained. Such a cipher text will yield a nearly 50% difference in its binary form in reference to the original cipher text.

The same situation is achieved when changing 1 bit in the initial encryption key (initialization quaternion). The difference in binary form of the modified cipher text and the original cipher text is also very close to 50%.

Let us now consider an example shown in Fig. 4. The aim is to encrypt a color Lena image. In the encryption process 9 rounds of the proposed algorithm are used. Each round has its own unique key (quaternion-key of higher order). Assume that the attacker knows 8 unique keys and possesses substantial knowledge about the last key (let us assume that the only difference is 1 bit in the binary form in the value y of the last quaternion-key). Decryption of the image performed by the attacker in such a scenario is shown in Fig. 4d. The presented result proves to be a strong avalanche effect.

5.3. Computation Speed

The authors analyzed the computation speed of proposed algorithm in comparison to the Advanced Encryption Standard (AES). For the purpose of this test the fastest AES version is implemented, i.e. AES-ECB, and its capabilities on the same machine as for our quaternion algorithm are measured. The machine used for the test was: Intel Core i5-3570 CPU @ 3.40 GHz, 16 GB RAM, and simulation environment was Matlab. The results of the comparison for a color Lena image are presented in Table 2. The expected values with confidence intervals calculated from 20 simulations are shown. The expected values presented in Table 2 are an estimation of the expected value μ, determined according to equation [20]:

$$P\left(\overline{X} - t_\alpha \frac{S}{\sqrt{N-1}} < \mu < \overline{X} + t_\alpha \frac{S}{\sqrt{N-1}}\right) = 1-\alpha, \quad (19)$$

where $\overline{X}$ is the expected value of the sample, S is the standard deviation of the sample, N is the size of the sample (20 simulations), t_α is a value obtained from the t-Student table for $N-1$ degrees of freedom and $1-\alpha$ is the confidence coefficient equal to 95%.

The initialization time refers to the time needed to calculate all of the necessary initialization parameters/values/matrices in order to perform the encryption/decryption. For AES, the initialization parameters refer to: the substitution box and its inverse, an arbitrary 16-byte cipher key, key expansion, a polynomial transformation matrix and its inverse. For the proposed algorithm, the initialization parameters refer to: the components of the initialization quaternion, initialization values for the quaternion Julia fractal, and the rotation order for which an appropriate key space is calculated (Fig. 2).

According to the results as presented in Table 2, one can see one of the main advantages of proposed algorithm, i.e. its fast computation speed. Moreover, in a practical scenario, encryption with AES could take even more time, especially considering the fact that a more secure implementation will be required, e.g., AES-CBC, AES-CTR, AES-OFB, or AES-OCB.

Table 2
Comparison of the computation speed of AES and the proposed algorithm for a color Lena image of size 243×243 pixels

AES-ECB	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$
Initialization time [s]	0.6226	4.851E-03
Encryption time [s]	68.05	0.1416
Decryption time [s]	98.55	0.2095
Total time [s]	167.2	–
Proposed algorithm	$\overline{X}$	$\frac{t_\alpha S}{\sqrt{N-1}}$
Initialization time [s]	0.05428	9.101E-03
Encryption time [s]	0.2643	2.018E-03
Decryption time [s]	0.2642	1.270E-03
Total time [s]	0.5829	–

6. Conclusions

According to the presented simulation results, it is relatively easy to show that a very good randomness of bit sequences in an encrypted image can be obtained using the proposed encryption scheme. Moreover, one of the main advantages of QFC is its fast computation speed. Because of the structure of the algorithm, the specific properties of quaternions and enormous key space, the algorithm's resistance to cryptanalytic attacks should significantly exceed multimedia encryption requirements. Of course, many possibilities exist according to which the proposed model could further be improved, e.g., main research focus now is to introduce additional operations to the cipher which will further increase its robustness capabilities.

It is important to note that when using the proposed key generation scheme, the number of possible encryption keys for each round is particularly large because of the possibility of setting the rotation order, the values of the 4 parameters of the initialization quaternion and the values for all quaternions f calculated based on a random quaternion Julia fractal image [1]–[3], [12].

References

[1] T. Nagase, M. Komata, and T. Araki, "Secure signals transmission based on quaternion encryption scheme", in *Proc. 18th Int. Conf. Adv. Inform. Netw. Appl. AINA 2004*, Fukuoka, Japan, 2004, vol. 2, pp. 35–38.

[2] T. Nagase, R. Koide, T. Araki, and Y. Hasegawa, "A new quadripartite public-key cryptosystem", in *Proc. Int. Symp. on Commun. and Inform. Technol. ISCIT 2004*, Sapporo, Japan, 2004, pp. 74–79.

[3] T. Nagase, R. Koide, T. Araki, and Y. Hasegawa, "Dispersion of sequences for generating a robust enciphering system", *Trans. Commun. Inform. Technol. (ECTI-CIT 2005)*, vol. 1, no. 2, pp. 9–14, 2005.

[4] M. Dzwonkowski and R. Rykaczewski, "A new quaternion encryption scheme for image transmission", *ICT Young 2012 Conf.*, Gdańsk, 2012, pp. 21–27.

[5] M. Dzwonkowski and R. Rykaczewski, "Quaternion encryption method for image and video transmission", *Przegl. Telekom. + Wiad. Telekom.*, vol. 8–9, pp. 1216–1220, 2013.

[6] B. Czaplewski, M. Dzwonkowski, and R. Rykaczewski, "Digital fingerprinting based on quaternion encryption for image transmission", *Przegl. Telekom. + Wiad. Telekom.*, vol. 8–9, pp. 792–798, 2013.

[7] B. Czaplewski, M. Dzwonkowski, and R. Rykaczewski, "Digital fingerprinting for color images based on the quaternion encryption scheme", *Pattern Recogn. Lett.*, vol. 46, pp. 11–19, 2014.

[8] M. Dzwonkowski and R. Rykaczewski, "A quaternion-based modified feistel cipher for multimedia transmission", *Przegl. Telekom. + Wiad. Telekom.*, vol. 8–9, pp. 1177–1181, 2014.

[9] V. U. K. Sastry and K. A. Kumar, "A modified feistel cipher involving modular arithmetic addition and modular arithmetic inverse of a key matrix", *Int. J. Adv. Comp. Sci. Appl. (IJACSA 2012)*, vol. 3, no. 7, pp. 40–43, 2012.

[10] R. Goldman, "Understanding quaternions", *Graphical Models*, vol. 73, no. 2, pp. 21–49, 2011.

[11] R. Goldman, *An Integrated Introduction to Computer Graphics and Geometric Modeling*. New York: CRC Press, 2009.

[12] F. Zhang, "Quaternion and matrices of quaternions", *Linear Algebra and its Applications*, vol. 251, pp. 21–57, 1997.

[13] D. Eberly, "Quaternion Algebra and Calculus", Geometric Tools, LLC, 2010 [Online]. Available: http://www.geometrictools.com/Documentation/Documentation.html

[14] G. M. Julia, "Memoir on iterations of rational functions", *J. de Mathématiques pures et appliquées*, 4th tome (83th volume of the collection), pp. 47–246, 1918.

[15] A. Douady, "Julia Sets and the Mandelbrot Set", in *The Beauty of Fractals: Images of Complex Dynamical Systems*, H.-O. Peitgen and P. H. Richter, Eds. Berlin: Springer, 1986, p. 161.

[16] C. Corrales-Rodrigáñez, "Rotations and units in quaternion algebras", *J. Number Theory*, vol. 132, no. 5, pp. 888–895, 2012.

[17] Quat – A 3D-Fractal-Generator, Version 1.20 [Online]. Available: http://www.physcip.uni-stuttgart.de/phy11733/quat_e.html

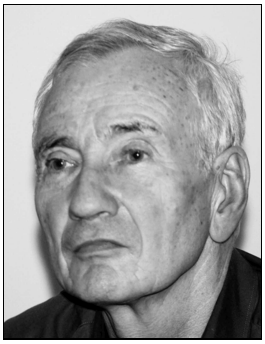
[18] Robert G. Brown's General Tools Page [Online]. Available: http://www.phy.duke.edu/∼rgb/General/dieharder.php

[19] Cryptool Portal [Online]. Available: http://www.cryptool.org/en/

[20] P. Armitage, G. Berry, and J. N. S. Matthews, *Statistical Methods in Medical Research*, 4th ed. (revised). Chichester: Wiley – Blackwell, 2001.

Mariusz Dzwonkowski received his M.Sc. Eng. degree in Telecommunication from Department of Teleinformation Networks, Gdańsk University of Technology. He is now working toward his Ph.D. degree in Telecommunication from Gdańsk University of Technology. He is currently a lecturer at Medical University of Gdańsk, Poland in Department of Radiological Informatics and Statistics. His research interests include steganography, cryptography with emphasis on quaternion encryption, network security and image processing.
E-mail: mar.dzwonkowski@gmail.com
Gdańsk University of Technology
Faculty of Electronics, Telecommunications and Informatics
Department of Teleinformation Networks
Gabriela Narutowicza st 11/12
80-233 Gdańsk, Poland

Medical University of Gdańsk
Department of Radiological Informatics and Statistics
Tuwima st 15
80-210 Gdańsk, Poland



Roman Rykaczewski received his M.Sc. (1968) and Ph.D. (1975) from Gdańsk University of Technology – Faculty of Electronics, Telecommunications and Informatics. From 1968 to the present he is working as an academic teacher at Gdańsk University of Technology, Faculty of Electronics, Telecommunications and Informatics. His current research mainly focuses on cryptography, watermarking and steganography.
E-mail: romryk@eti.pg.gda.pl
Gdańsk University of Technology
Faculty of Electronics, Telecommunications and Informatics
Department of Teleinformation Networks
Gabriela Narutowicza st 11/12
80-233 Gdańsk, Poland