Advances in Electrical and Computer Engineering **Volume 21, Number 4, 2021**

Quantum Steganography Using Two Hidden Thresholds

Alexandru-Gabriel TUDORACHE, Vasile MANTA, Simona CARAIMAN
Department of Computer Science and Engineering,
Gh. Asachi Technical University of Iasi, D. Mangeron 27A, 700050, Iasi, Romania
alexandru-gabriel.tudorache@academic.tuiasi.ro

***Abstract*—This paper describes a novel steganography algorithm that combines quantum computing with the least significant bit technique (LSB). Using the quantum properties, along with the Python programming language and the Qiskit framework for the circuit simulation, a sub-image can be hidden inside a usual image. For a gray image, this article presents how the LSB of each of the first 16 pixels on the edges can be used to hide two threshold values, that are then used to filter out the image and reveal the secret message. The speedup, compared to the classical version, is possible due to the quantum representation of an image (NEQR is used in this paper) and the efficiency of the novel multi-bit quantum comparators.**

***Index Terms*—data security, image communication, image processing, image representation, quantum computing.**

I. INTRODUCTION

The quantum representation and processing of information is a great subject in the information technology area. This has been especially highlighted by the latest innovations, such as D-Wave, IBM Q System One (a 20-qubit quantum computer) and Google's quantum computer (a 54-qubit Sycamore processor) – it has recently claimed "quantum supremacy", by performing a calculation in 200 seconds instead of 10,000 years, the time required on a classical computer. Today we can not only simulate various circuits on classical computers, but also run and measure the results on real platforms, available through cloud services.

The whole quantum universe is based on the core concept that the fundamental unit, named "qubit" (quantum bit), can not only be 0 or 1 like the classical bit, but it can simultaneously be in different states – this property is called superposition. For each of these states, the qubit has a certain probability associated with it. In the quantum world, these states are $|0\rangle$ and $|1\rangle$, and they are also known as "ket" or "vector". Any qubit $|\psi\rangle$ can be written as a superposition of these states, as seen in formula (1) below:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad (1)$$

with α and β complex numbers, which verify the property: $|\alpha|^2 + |\beta|^2 = 1$. The probability for the qubit to be in the $|0\rangle$ state is $|\alpha|^2$ and the probability for it to be in the $|1\rangle$ state is $|\beta|^2$.

Every quantum algorithm used to process data requires a quantum circuit that implements certain operations. In the quantum universe, these operations can be translated using quantum gates; all of the quantum gates are described using unitary transforms, so therefore they are reversible – this implies that the number of inputs be equal to the number of outputs. The gates required to implement the circuits presented in this paper are briefly explained below; the single qubit gates are as follows: the NOT gate ("X"), that acts in a similar way to the classical not gate, (it is a linear gate, the state $a|0\rangle + b|1\rangle$ is changed to $a|1\rangle + b|0\rangle$, and the Hadamard gate ("H"), which allows to set the basic states into superposition, with equal probabilities, see formulas (2) and (3):

$$H|0\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}1\\ 0\end{bmatrix} = \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1\\ 1\end{bmatrix}\right) = = \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1\\ 0\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}\right) = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right), \quad (2)$$

$$H|1\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}0\\ 1\end{bmatrix} = \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1\\ -1\end{bmatrix}\right) = = \frac{1}{\sqrt{2}}\left(\begin{bmatrix}1\\ 0\end{bmatrix} - \begin{bmatrix}0\\ 1\end{bmatrix}\right) = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right). \quad (3)$$

The gates that act on multiple qubits, used in this paper, are:

i. the CNOT gate (controlled-NOT), which can be seen as the equivalent of the classical XOR gate. Its input qubits are called the control qubit and the target qubit; the $|0\rangle$ state for the control qubit means that the target qubit remains unchanged; the $|1\rangle$ state for the control qubit reverses the state of the target qubit. If we name the two qubits $|c\rangle$ (the control qubit) and $|t\rangle$ (the target qubit), then the action of the gate can be summarized as $|ct\rangle \rightarrow |c(c \oplus t)\rangle$.

ii. the CCNOT (Toffoli) gate, that is an extension of the CNOT gate; it has 2 control qubits ($|c_1\rangle$ and $|c_2\rangle$) and 1 target qubit ($|t\rangle$). The target qubit will be set by calculating the XOR value between t and the logic product of the control qubits, see formula (4):

$$|t\rangle \rightarrow |t \oplus (c_1 \wedge c_2)\rangle \quad (4)$$

One of the most important aspects for having better performance regarding a quantum algorithm, compared to its classical counterpart, comes down to being able to utilize the quantum properties such as superposition and quantum entanglement. When referring to quantum steganography, the way the image is represented is crucial to bringing that

Digital Object Identifier 10.4316/AECE.2021.04009

1582-7445 © 2021 AECE

performance edge. Several proposals have been made over the years, and a paper that contains a survey on the possibilities for image representations was published by F. Yan et al. in 2016 [1]. Some of the most notable techniques used to represent images in their quantum form are:

i. FRQI (flexible representation for quantum images), proposed by P.Q. Le et al. in paper [2], contains information about color and position, integrating them into a quantum state; the image is defined as the sum of tensor products of color and position vectors; the color vector is defined as in formula (5):

$$\left|c_i\right\rangle = \cos\theta_i\left|0\right\rangle + \sin\theta_i\left|1\right\rangle, \tag{5}$$

where, θ is the vector of angles which encodes the color component;

ii. NEQR (novel enhanced quantum representation), proposed by Y. Zhang et al. in paper [3], contains information about the grayscale value and the position of each qubit; unlike FRQI, where the color information is referred to using the probability amplitude, in NEQR it is stored in the multi-qubit computational basis of the superposition state;

iii. NCQI (novel quantum representation of color digital images), proposed by J. Sang et al. in paper [4], can be seen as an extension of NEQR, since it encapsulates the same idea, but for RGB images; the color component for each pixel is defined by the superposition of the 3 groups of qubits, one for the representation of each channel, as shown in formula (6):

$$\left|c(y,x)\right\rangle = \left|R_{q-1}...R_0G_{q-1}...G_0B_{q-1}...B_0\right\rangle, \tag{6}$$

where, q is the number of qubits for each color component;

iv. QRCI (new quantum representation model of color digital images), proposed by L. Wang in paper [5] and inspired by NCQI, uses the bit-plane for each of the 3 color components; an image can be defined with formula (7):

$$I = \frac{1}{\sqrt{2^{2n+3}}}\sum_{L=0}^{2^3-1}\sum_{Y=0}^{2^n-1}\sum_{X=0}^{2^n-1}\left|R_{LYX}G_{LYX}B_{LYX}\right\rangle\left|LYX\right\rangle, \tag{7}$$

where, $\left|LYX\right\rangle = \left|L\right\rangle\left|Y\right\rangle\left|X\right\rangle$ and $\left|L\right\rangle$ represents the bit-plane information (in the formula above the number of bit-planes is 8, therefore 3 qubits are required) and $\left|YX\right\rangle$ the position information.

Over the past years, researches have proposed multiple techniques in the field of quantum steganography. One of these is a novel watermarking protocol, using the NEQR representation, together with the Gray code transform and the least significant bit (LSB) steganography (see [6]). It uses the edges of the image, and the two LSBs. A watermarking algorithm that uses not only the LSBs, but also the Arnold scrambling, is proposed in paper [7]. The classical watermark image is first expanded, then scrambled using the Arnold transform (using the module of parallel adder modulo N), and finally embedded in the carrier image using the LSB method. The illustrated circuit from the simulation presents a lower time complexity. A similar concept, that relies on the LSB, combined with the bit-plane scrambling is described in paper [8]. The original image is modified using the bit-plane transformation, and then expanded to match the size of the cover image, using the secret key. These operations are followed by applying the Arnold scrambling, and the embedding procedure; hiding and extracting the message is done using two keys.

Another steganography idea is based on an algorithm of modified exploiting modification direction (EMD, see paper [9]), where two images are used to store a secret message that is hidden inside a color image. The key is used in correlation with the three-color channels (R, G, B) – two of them are selected from the cover image. An improved EMD algorithm is presented in paper [10], where the new described concepts are those of expanding the modification range and the dynamically sharing between subgroups. Using these ideas, as well as embedding the secret message using bit-planes, allows this protocol to achieve a higher embedding rate; the authors of paper [10] also designed quantum circuits that illustrate how to implement this algorithm.

In paper [11], a method that uses an inverted pattern approach is described. Here, as a final step in hiding/embedding the secret, the authors present a pixel adjustment process, that, depending on the quantum key, inverts (or keeps unmodified) every pixel from the quantum secret image. The key is stored in a variable number of LSBs from the cover image; the algorithm also has the property of extracting the hidden message using the inverse of the illustrated solution.

In a different approach, the authors of paper [12] present a detection method using the LSB steganography in combination with the NEQR representation. This is based on the division of the quantum image in pixel blocks, followed by a classification of these blocks in 3 groups. A subsequent processing of these groups is used to indicate whether the image holds a hidden message. Another concept is presented in paper [13], where a novel coherence-based quantum steganalysis protocol is proposed. The geometric coherence and ½-affinity coherence are analyzed to calculate the probability of correct transmission, and two new performance metrics are introduced (steganographic detection rate and false alarm rate). They are calculated in detail for a BB84-based quantum steganography protocol. LSBq for multi-wavelength quantum images have also been used to implement a quantum steganography algorithm, as shown in paper [14]. The hidden message is embedded using methods such as the modulo method and the Hilbert scrambling, and the proposed protocol is robust against attacks on the stego-image.

Various concepts for quantum key distribution protocols, quantum communication and secret sharing, that could be used to further develop the field of quantum steganography, are presented in detail in papers [15–19]. An interesting idea might also be to combine one of the quantum image representations with a quantum E-payment protocol, such as the one presented in paper [20]. Information on the historical evolution of steganography can be found in [21], while [22–33] present an overview of steganography techniques, as well as some of its current applications, representing a source of inspiration in the quantum universe.

In our paper, the chosen image representation technique is NEQR, described in detail in Section III. Section II presents the Qiskit framework, while Section IV illustrates the proposed algorithm and Section V describes the quantum circuits and the simulation results.

II. THE QUANTUM FRAMEWORK

The programming language that was used to implement, simulate and measure the results for the representation of images is Python, using the Qiskit framework, developed by IBM. A quick overview of this framework and its possibilities will be shown in this section. According to the official IBM website, Qiskit is "an open-source quantum computing software development framework for leveraging today's quantum processors in research, education and business". It offers support for both the execution and simulation of code written for quantum applications and algorithms.

It is made up of 4 major components. Qiskit Terra contains a set of tools for writing quantum programs at the circuit level; it uses different optimizations for the available physical quantum processor and manages the execution of the experiments. Qiskit Aer is the high-performance simulator component – it contains optimized C++ simulator backends for the circuits compiled using Qiskit Terra, along with tools needed to analyze various noise models. Qiskit Aqua contains a set of quantum algorithms that can be used in different applications. Qiskit Ignis is a framework focused on the study of noise in quantum systems.

The results presented in the following sections are obtained using the Aer simulator component, running on the local machine, with a number of program executions ("shots") of 1024.

III. QUANTUM IMAGE REPRESENTATION

Multiple ways of representing an image using the quantum properties have been proposed over the years, and the one chosen in this paper is the novel enhanced quantum representation of digital images – NEQR, as proposed in paper [3]. This is justified by the more precise control of the color information, as well as the fact that the recovery of the image can be done using a finite number of quantum measurements (for each pixel). The NEQR uses a combination of entangled qubits, used to store the color and position information. Assuming a grayscale range of 0 to 2^q-1 (q bits required to represent the image), the gray value for each pixel (Y,X) can be written as $C_{YX}^0 C_{YX}^1 ... C_{YX}^{q-2} C_{YX}^{q-1}$, where C_{YX}^i represents the gray value for the i-th bit.

A 2^nx2^n image can be expressed using the following general formula, as presented in [3], see formula (8) below:

$$\left|I\right\rangle = \frac{1}{2^n}\sum_{Y=0}^{2^n-1}\sum_{X=0}^{2^n-1}\otimes_{i=0}^{q-1}\left|C_{YX}^i\right\rangle\left|YX\right\rangle. \tag{8}$$

The first step in a quantum algorithm involving the processing of an image requires us to design the quantum circuit that represents the image in the NEQR form. Let's take the case of a grayscale image, where the gray level ranges from 0 to 255, and 8 classical bits are necessary for every value. We will need 8 qubits for the gray level, along with a number of qubits required for the binary representation of the total number of pixels – the position qubits. We will also require auxiliary qubits, that entangle with the position qubits; if certain conditions are validated (a set of conditions for every pixel position), then the connection between the last auxiliary qubit and the corresponding gray qubits sets their state. Every position qubit is first set up in superposition using the Hadamard gate, and then, for every position, if the corresponding bit for that position is 0, the inverse (NOT/X) gate is used (one combination will be generated during each phase of the circuit). Once all the position qubits are in the $\left|1\right\rangle$ state, we can use CNOT and CCNOT gates to entangle them with the auxiliary ones. An example is presented below, for the sample image in Figure 1, with the help of a Toffoli gate (see Figure 2). At the end of each cycle, the X gate is used again for each position qubit, if necessary, to restore the initial superposition state (after the Hadamard gate).

Let's analyze the following 2 x 2 image (Figure 1):

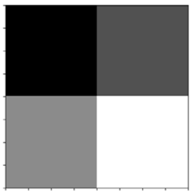

Figure 1. Sample 2x2 image

The gray levels for the image in Figure 1 are:

$$\begin{bmatrix} 38 & 96 \\ 136 & 217 \end{bmatrix}.$$

Using the general formula for the NEQR, this image could be represented as in formulas (9) and (10):

$$\begin{aligned}\left|I\right\rangle = \frac{1}{2}(&\left|38\right\rangle\otimes\left|00\right\rangle + \left|96\right\rangle\otimes\left|01\right\rangle + \\ &+\left|136\right\rangle\otimes\left|10\right\rangle + \left|217\right\rangle\otimes\left|11\right\rangle)\end{aligned} \tag{9}$$

$$\begin{aligned}\left|I\right\rangle = \frac{1}{2}(&\left|00100110\right\rangle\otimes\left|00\right\rangle + \left|01100000\right\rangle\otimes\left|01\right\rangle + \\ &+\left|10001000\right\rangle\otimes\left|10\right\rangle + \left|11011001\right\rangle\otimes\left|11\right\rangle)\end{aligned} \tag{10}$$

For a 2 x 2 image, 2 qubits are required to represent the position (the *qp* register). The gray intensity is saved in the $q_7q_6q_5q_4q_3q_2q_1q_0$ vector, one ancilla qubit is used, and the classical registers used for measurements are cr_q and cr_a.

The NEQR part of the circuit for the first pixel, with gray value 38, is presented in Figure 2.

By measuring the probabilities using the local simulator, we can verify that the gray value qubits are set only when the auxiliary qubit is in the $\left|1\right\rangle$ state, as shown in Figure 3. The full NEQR circuit would contain the representation of all 4 pixels, in a similar manner as the one presented; this is the reason why the position qubits are set in superposition, so that at a single shot, the system would collapse and only one pixel value would be indicated, with its coordinates (position qubits). By repeating the experiment, there would theoretically be a 25% chance of obtaining each of the 4 pixel values. If we simulate only one part of the NEQR circuit, for the first pixel, we expect the system to collapse to the desired state (ancilla qubit to 1, and the gray qubits to 00100110 – the binary representation of 38) in only 25% of cases, the other 75% keeping the state of the gray qubits unmodified, as the position qubits indicate the other coordinates.

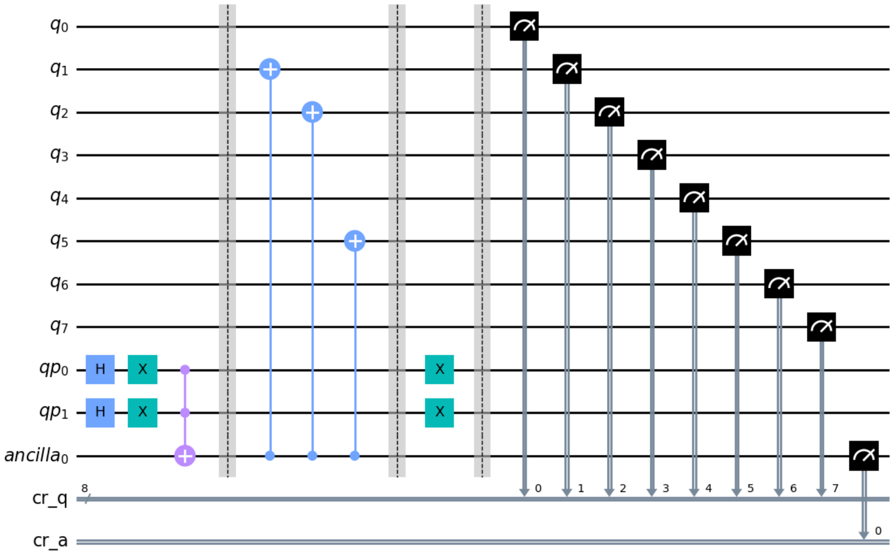

Figure 2. Quantum circuit for the first pixel. After the first set of NOT gates, the state of each position qubit we are looking for is $\left|1\right\rangle$. The ancilla qubit will be then set to state $\left|1\right\rangle$, which in turn sets the 7 qubits (q7-q0) to the equivalent binary representation of value 38

The obtained values, 24.1% and 75.9%, are relatively close to the ideal probabilities.

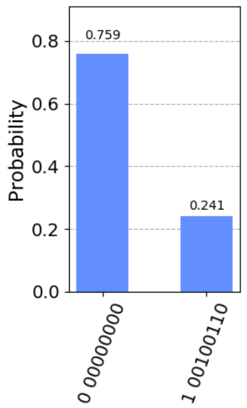

Figure 3. Measured results, after simulating the circuit in Figure 2

IV. ALGORITHM DESIGN

The current paper proposes to hide information inside an image, using the LSBs from the representation of various pixels; different patterns have been proposed over the years – some use the edges, or perhaps a certain algorithm, like skipping the odd or even bytes.

The proposed algorithm uses the following conventions: the lower threshold, hidden in a grayscale image, is made up of the LSB of each of the first 8 pixels, starting with the edge in the left upper corner and going clockwise. The upper threshold is made up of the LSBs from the following 8 pixels. If the image is not large enough (doesn't have enough pixels) to hide the thresholds on the edges, then the next layer can be used; this also means that the image has already been processed in a classical or quantum manner to set the last bits of the first 16 pixels and the bits for the actual message.

An *n* x *m* gray image can be written using a binary matrix as follows in formula (11):

$$I = \begin{bmatrix} I_{0,0} & \cdots & I_{0,m-1} \\ \vdots & \ddots & \vdots \\ I_{n-1,0} & \cdots & I_{n-1,m-1} \end{bmatrix}. \tag{11}$$

If the LSB value of a pixel, located at the *x* row and *y* column ($I_{x,y}$), is expressed as $LSB_{x,y}$, then depending on the size of the image, the threshold, with the MSB (most significant bit) saved in the first pixel, can be written as shown in formulas (12), (13), (14), (15), (16) and (17) below:

i. for *m* equal or greater than 16 (the first row contains enough pixels, so that their LSBs cover both of the thresholds):

$$\begin{aligned} Th_{lower} = {} & LSB_{0,0}LSB_{0,1}LSB_{0,2}LSB_{0,3} \\ & LSB_{0,4}LSB_{0,5}LSB_{0,6}LSB_{0,7}, \end{aligned} \tag{12}$$

where, $LSB_{0,0}$ represents the MSB of the threshold and

$$\begin{aligned} Th_{upper} = {} & LSB_{0,8}LSB_{0,9}LSB_{0,10}LSB_{0,11} \\ & LSB_{0,12}LSB_{0,13}LSB_{0,14}LSB_{0,15}; \end{aligned} \tag{13}$$

ii. for *m* less than 16 and *n* equal to or greater than 16, for example *m=4* (the first row and the last column are encoded with the values for the thresholds):

$$\begin{aligned} Th_{lower} = {} & LSB_{0,0}LSB_{0,1}LSB_{0,2}LSB_{0,3} \\ & LSB_{1,3}LSB_{2,3}LSB_{3,3}LSB_{4,3}, \end{aligned} \tag{14}$$

$$\begin{aligned} Th_{upper} = {} & LSB_{5,3}LSB_{6,3}LSB_{7,3}LSB_{8,3} \\ & LSB_{9,3}LSB_{10,3}LSB_{11,3}LSB_{12,3}; \end{aligned} \tag{15}$$

iii. for both *m* and *n* less than 16, for example *m=7* and *n=5*:

$$\begin{aligned} Th_{lower} = {} & LSB_{0,0}LSB_{0,1}LSB_{0,2}LSB_{0,3} \\ & LSB_{0,4}LSB_{0,5}LSB_{0,6}LSB_{1,6}, \end{aligned} \tag{16}$$

$$\begin{aligned} Th_{upper} = {} & LSB_{2,6}LSB_{3,6}LSB_{4,6}LSB_{4,5} \\ & LSB_{4,4}LSB_{4,3}LSB_{4,2}LSB_{4,1}. \end{aligned} \tag{17}$$

If the image is too small and there are not enough pixels on the edges *(2n+2m-4<16)*, then the thresholds are embedded in the next edge layer (clockwise, starting with pixel $I_{1,1}$).

After the threshold values have been calculated (they can be extracted – which can be done in a classic manner, in parallel, without any performance loss or in quantum, by entangling the LSB qubits of the first 16 pixels to other qubits, especially chosen for the threshold values; they can also be used directly in quantum, since with NEQR we already have access to them), each group of 8 qubits in the quantum image (the representation of a grayscale value) should be compared to these values. One such design is the 8-bit half comparator proposed by Xia et al. in paper [34] and will be treated as a black box in the 2 figures below; except for the two 8 qubit values, one more auxiliary qubit is needed (*comp*), that will be set to the $\left|1\right\rangle$ state if the second value is greater than the first and $\left|0\right\rangle$ otherwise. Figure 4 presents these comparators:

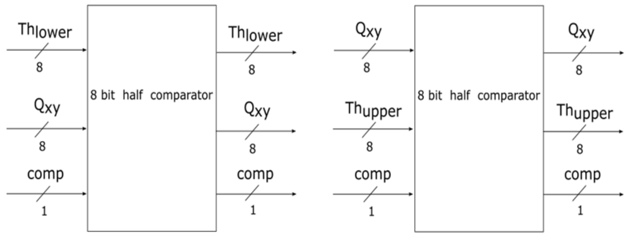

Figure 4. Design of the two 8-bit comparators required for each pixel

In other words, in Figure 4 and the rest of this paper, the representation convention in the design above is that the c*omp* qubit will be set to $\left|1\right\rangle$ if the value in the image is within the boundaries of the thresholds.

The general diagram for each group of 8 qubits requires an auxiliary qubit, *pass*, that will be set to $\left|1\right\rangle$ if both output (*comp*) values of the comparators (*comp*$_1$ and *comp*$_2$) in the figure below) will be in the $\left|1\right\rangle$ state – this is done by using a Toffoli gate. After this operation, the value must be inverted and connected to a reset block, that will either keep the state of the image qubits, or set them to the $\left|0\right\rangle$ state (the output of this block is called Q'_{xy} – this is the decrypted image). This diagram is represented in Figure 5.

There are two ways to go forward depending on the approach regarding the reset operation of a qubit:

1. The first method assumes that a hypothetical reset gate (or a general gate that could set a qubit to a desired state, $\left|0\right\rangle$ or $\left|1\right\rangle$) will be implemented in the future, then basically the reset control block connects the *NOT*(*pass)* state to the ENABLE signal of a simple pass-or-reset block for each qubit of the Q_{xy} vector. The feasibility of using NEQR as a way to represent an image, along with certain conditions for traceability concerning the reset gate, are presented by Mario Mastriani in [35] – this justifies the second method.

2. If we assume that a reset gate cannot be implemented, an alternative solution would be to stop the quantum processing right after the NOT gate in Figure 5, by measuring (and thus collapsing) the image qubits. The logic for the reset control block can be continued in a classical manner, with a AND gate between the measured (*NOT*(*pass)*) value and the image bits (for a group of image pixels this can be done in parallel on multiple threads, since keeping or resetting the values for the image pixels are independent processes).

If we want to hide a grayscale sub-image using the proposed protocol, the following steps need to be taken: first, we need to classically modify the edge pixels in the cover image, in order for them to indicate the thresholds, such that the pixels from the secret image have values in the desired interval. Then, the sub-image is hidden inside the cover object, either by replacing the target pixels belonging to the secret, or by using a different value modification technique for the pixels' value, to better embed the hidden sub-image; any method can be used, as long as the secret does not stand out. The decryption implies the classical recovery of the thresholds, the design of the quantum image using the NEQR representation, followed by the addition of the comparator blocks (see Figure 5). Two circuits, one for each threshold (together with all the pixels and the comparators) would be required for the entire process. Also, in order to obtain the best results, it would help to select or modify a sub-image such that its pixels' gray range is as small as possible.

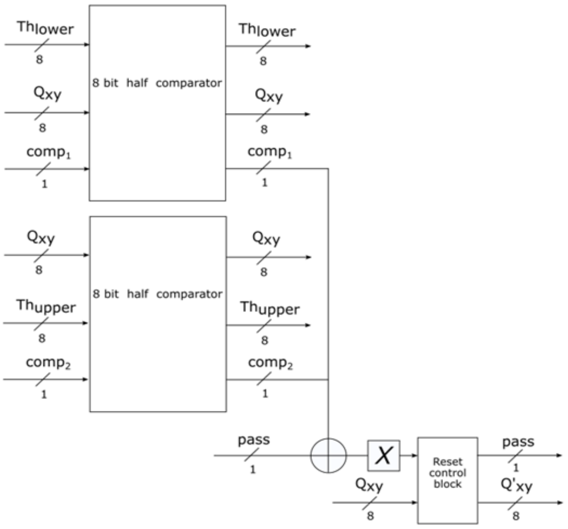

Figure 5. General diagram of the threshold comparing process

Hiding an RGB sub-image can be done in two manners: if we want to hide it inside an RGB image, then using a different quantum representation technique (for color images) is recommended, such as NCQI or QRCI. If we want to hide an RGB image inside a grayscale image, then we would require a different, more complex design; for example, we can convert the RGB image to a grayscale image, hide it as explained above, and then keep some information (besides the thresholds) regarding the RGB channels for some pixels of the secret on the edges – this may not be optimal, since we would have to use a lot of edge pixels, and could only work if the hidden RGB sub-image is relatively small compared to the cover object. The current paper focuses on cover objects and hidden messages represented using grayscale images.

V. IMPLEMENTATION AND RESULTS

This section of the paper offers details regarding the configuration used within the Qiskit framework, together with the results obtained for the described algorithm.

After the quantum image representation, described in detail in section III, the next obstacle was to create a subcircuit capable of implementing a custom Toffoli gate, required by the 8-bit half comparator (see Figure 6, from paper [34]).

The custom Toffoli gate, named "ccx2", implements a subcircuit that performs the following operations: when applied on three qubits, it leaves the first unchanged, it inverts the second, applies a Toffoli gate with the first two as control qubits and the third as the target qubit, and then reverts the second to its original state using another "X" gate. The following figures illustrate a test circuit for this gate: two Hadamard gates for the control qubits and the circuit itself – Figure 7 shows the decomposed circuit, and Figure 8 its symbolic representation.

This circuit can be quickly verified by checking that qubit q_3 is set to $|1\rangle$ only when q_1 is set to $|1\rangle$ and q_2 to $|0\rangle$, which can be seen in Figure 9. The next step is to create a circuit that contains the following qubits: the first group of 8 qubits (plus one ancilla qubit and qubits for position, their number depending on the image size) will be used to represent the current pixel of an image, and the following groups will represent the first 8 pixels for the lower threshold or the following 8 for the upper threshold.

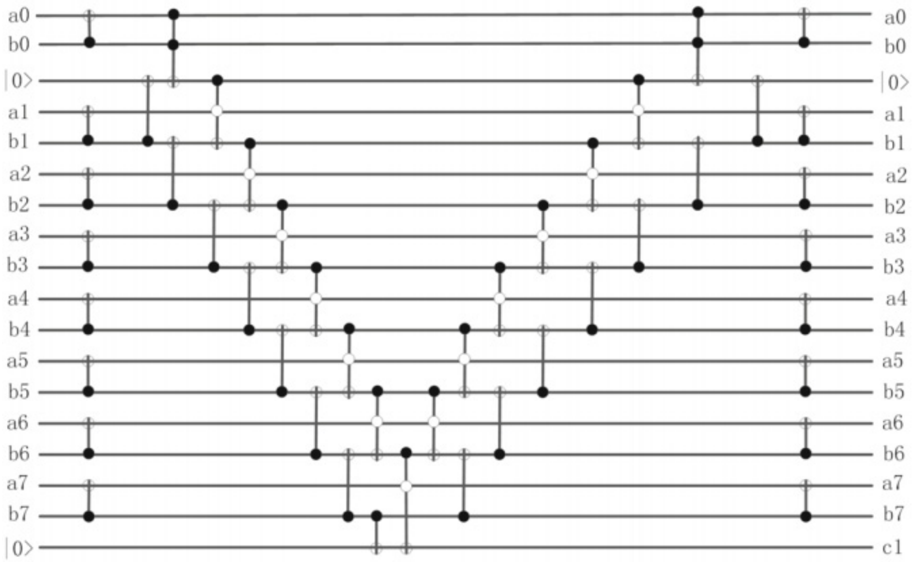

Figure 6. The 8-bit half comparator [34]

The current pixel, along with the threshold pixels are represented using NEQR; in order to simplify the visual representation of the circuit we can use a "multi-cX" gate ("mcx" in Qiskit), that correlates only one ancilla qubit to the position qubits (there are multiple techniques in which the mcx gate can be implemented, but it can generally be seen as an extension of the CNOT gate with a variable number of control qubits).

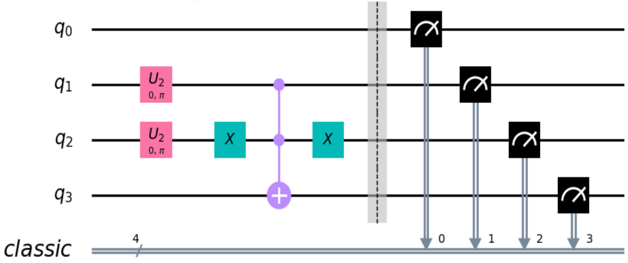

Figure 7. The custom Toffoli gate (expanded view)

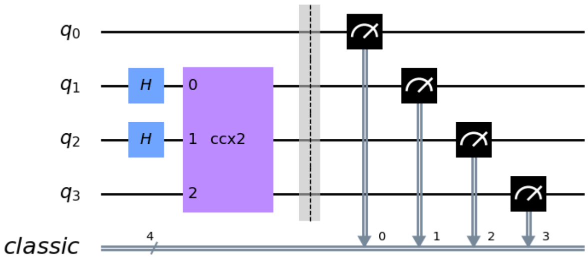

Figure 8. The custom Toffoli gate (compressed view)

Once all these qubits are in the desired states, the comparator is added to the circuit (it uses 18 qubits – 2 groups of 8 for the current pixel and the threshold, along with 2 auxiliary qubits, see Figure 6). In total, this circuit requires a group of 8 gray level qubits, 6 position qubits and 1 auxiliary qubit for the NEQR representation, multiplied by 9, to which we add 2 auxiliary qubits for the comparator, that is 137 qubits.

By zooming in on the first part of this circuit, we can see the representation of the current pixel (in this particular case it is the first pixel of the image), together with the representation of the first two pixels, the last qubit of each group being required for the lower threshold (Figure 10).

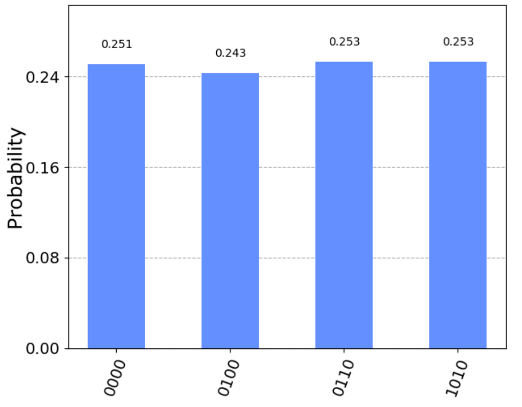

Figure 9. Measured results for the custom Toffoli gate

Obtaining the desired results can be done by simulating a circuit that keeps the same core elements as the one above, but only contains the minimum number of qubits required for measurements. For example, for an 8x8 image, where 6 qubits for position are required, a circuit would require the following 30 qubits:

i. 9 qubits are required for the lower threshold – 8 qubits for the gray and an auxiliary one that can set the gray level qubits to mirror the qubit states of the last qubit for each pixel representation (of the first 8) from the full circuit;

ii. 19 qubits are required for the current pixel – the NEQR representation in this case implies the use of 8 qubits for the gray levels, 6 for the position and 5 ancilla qubits;

iii. 2 ancilla qubits (*aux1* and *res1*) are required by the comparator.

The described configuration can be seen in the circuit from Figure 11, for the value of the lower threshold of 128 and the value of the current pixel of 200.

The results obtained by simulating the circuit can be seen in Figure 12. The measurements from this figure clearly show that when the qubits are in the desired states (200 for the current pixel and 128 for the lower threshold), the first qubit from the measured results (*res1*, measured using the classical register *cl-res*) is in state $|1\rangle$, which means that the comparator identified that the second number (value of current pixel) is higher than the first (lower threshold). The probabilities can be quickly verified by counting the number of qubits set in superposition – in the presented case, these are the 6 qubits used to represent the position, so the theoretical probability of all of them being in state $|0\rangle$, after the Hadamard gate (and all of them in state $|1\rangle$ after the first set of NOT gates) is 1/64=0.015625, relatively close to the 0.012 value from the simulated experiment.

In terms of gate costs, we can see that the circuit presented in Figure 11 requires 13 NOT gates, 6 Hadamard gates, 35 CNOT gates, 7 CCNOT gates and 13 subcircuits, called "ccx2", each of them containing one CCNOT gate and two NOT gates, required for the comparator (so 20 CCNOT gates and 39 NOT gates in total).

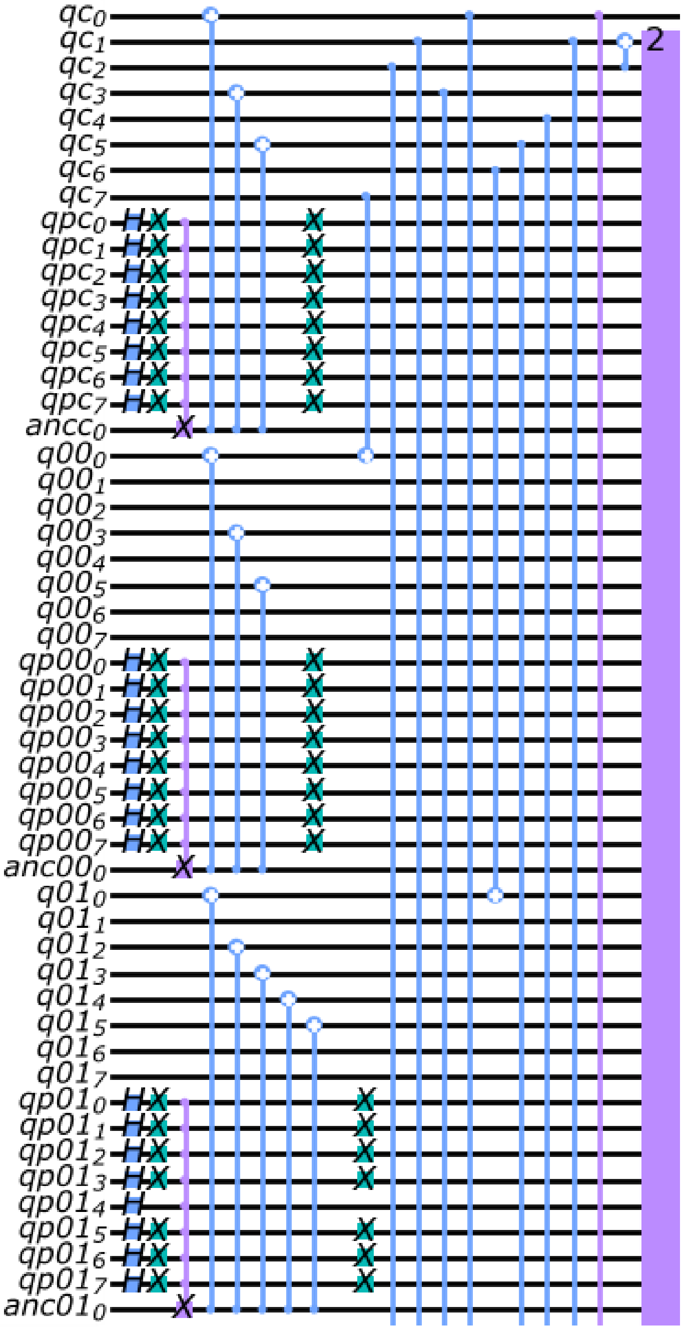

Figure 10. NEQR representation for the current pixel and the threshold, along with the comparator (zoomed in from the full circuit with 137 qubits)

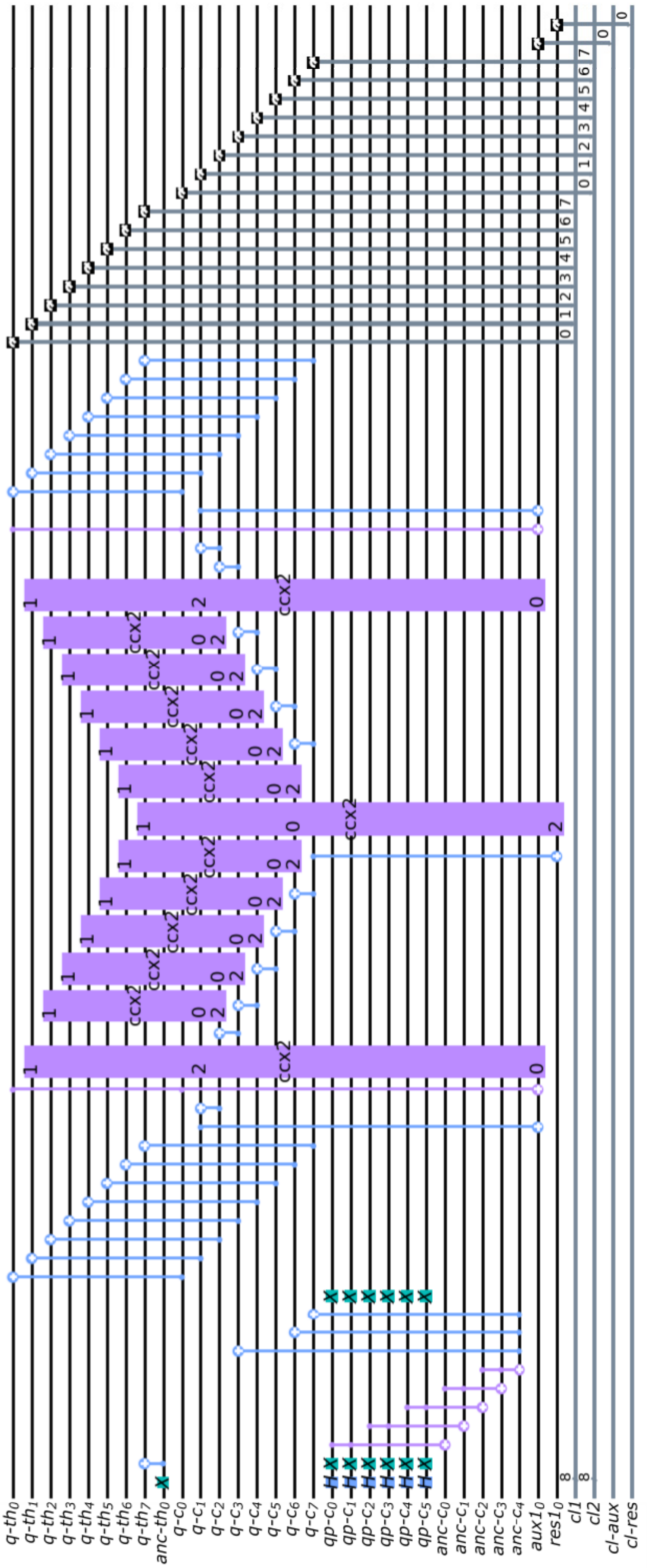

Figure 11. Circuit containing the NEQR representation of the current pixel, the threshold qubits, along with the comparator

Note that the number of NOT gates and CNOT gates, required to set the qubits from qc7 to qc0, depends on the pixel position and gray value. The worst-case scenario for the circuit in Figure 11 occurs when both the threshold and the current pixel have value 255, located in the upper left corner of the image, equivalent to indexes 0 for both the row and the column, which in turn would require the maximum number of NOT gates. In this case, 47 CNOT gates would be needed in total, while the numbers for all the other types of gates above remain the same. A circuit like the one described would have to be applied two times, extended to all pixels in the image (once for the lower threshold and once for the upper one).

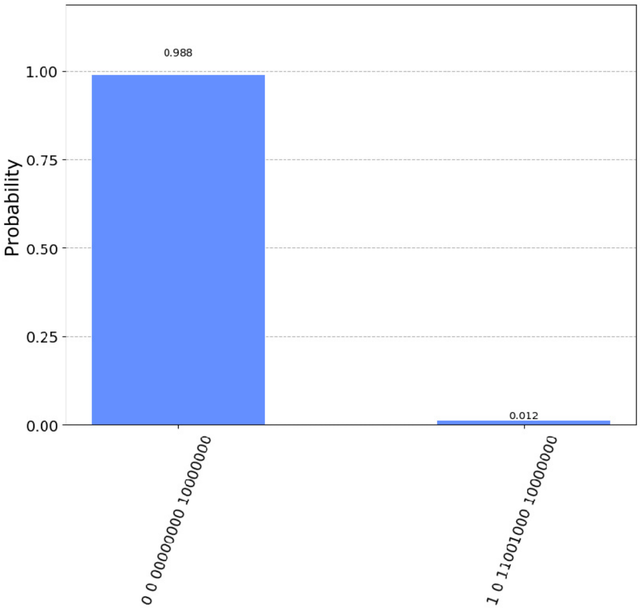

Figure 12. Results obtained for the simulation of the circuit presented in Figure 11

The memory usage for simulating the circuit in Figure 11 was 16,416.2 MB RAM, and the CPU was running at 100% at 4.68 GHz. The simulation took 70.9 seconds. The computer used for this test has an Intel i9-9900K processor at a base speed of 3.60 GHz.

The main advantage of the presented quantum algorithm, in comparison to a classical implementation, is the speedup. In this context, the representation itself (NEQR), if implemented on a quantum computer that is fast enough (shots per second), will ensure the fact that the access to all qubit states (pixel values) could be done much faster. Another key element of the algorithm is the use of the quantum comparators, which are configured as an extension for NEQR in the proposed protocol; they stand out as being highly efficient, using a relatively low number of quantum operations and having a small quantum delay (see [34]). The circuit in Figure 11 represents a part of one of only two greater circuits (one for each threshold), that would contain the encoding of each pixel; for larger images, the quantum solution improves the classical one, which would have to manually go over each pixel at a time and perform two classical comparison operations. If a parallel-processing solution is wanted on the classical devices, then special hardware could be required, and/or multithreaded programming. This quantum protocol presents a higher degree of computing parallelism and usability, since it builds on what is seen as one of the already analyzed quantum image representation techniques, without needing any additional effort for its efficiency.

VI. CONCLUSION

The current paper describes a way of hiding information inside an image, with the help of a quantum framework, by embedding not only a secret message in the image, but also two auxiliary hidden thresholds used for its processing. Among the presented ideas, an implementation of one quantum representation (NEQR) for a grayscale image is shown, using the Qiskit framework (and the Python programming language), where two thresholds are hidden by setting the LSB on the edges (this can be extended to an RGB image). The general diagram of the proposed algorithm involves the extraction of the required threshold values using the design of the 8-bit half comparators proposed by Xia et. al. A short analysis/discussion on the control block for the reset operation is described, which could further contribute to a faster algorithm, should an operation for resetting a qubit be possible in the future. The enhanced performance for the presented scheme is justified by the quantum representation of the image, in combination with the mentioned comparators, allowing for an efficient quantum steganography approach.

REFERENCES

[1] F. Yan, A. M. Iliyasu, S. E. Venegas-Andraca, “A survey of quantum image representations,” Quantum Inf Process, vol. 15, no. 1, pp. 1–35, Jan. 2016. doi:10.1007/s11128-015-1195-6

[2] P. Q. Le, F. Dong, K. Hirota, “A flexible representation of quantum images for polynomial preparation, image compression, and processing operations,” Quantum Inf Process, vol. 10, no. 1, pp. 63–84, Feb. 2011. doi:10.1007/s11128-010-0177-y

[3] Y. Zhang, K. Lu, Y. Gao, M. Wang, “NEQR: a novel enhanced quantum representation of digital images,” Quantum Inf Process, vol. 12, no. 8, pp. 2833–2860, Aug. 2013. doi:10.1007/s11128-013-0567-z

[4] J. Sang, S. Wang, Q. Li, “A novel quantum representation of color digital images,” Quantum Inf Process, vol. 16, no. 2, p. 42, Feb. 2017. doi:10.1007/s11128-016-1463-0

[5] L. Wang, Q. Ran, J. Ma, S. Yu, L. Tan, “QRCI: A new quantum representation model of color digital images,” Optics Communications, vol. 438, pp. 147–158, May 2019. doi:10.1016/j.optcom.2019.01.015

[6] W. Hu, R.-G. Zhou, J. Luo, B. Liu, “LSBs-based quantum color images watermarking algorithm in edge region,” Quantum Inf Process, vol. 18, no. 1, p. 16, Jan. 2019. doi:10.1007/s11128-018-2138-9

[7] R.-G. Zhou, W. Hu, P. Fan, “Quantum watermarking scheme through Arnold scrambling and LSB steganography,” Quantum Inf Process, vol. 16, no. 9, p. 212, Sep. 2017. doi:10.1007/s11128-017-1640-9

[8] R.-G. Zhou, J. Luo, X. Liu, C. Zhu, L. Wei, X. Zhang, “A Novel Quantum Image Steganography Scheme Based on LSB,” Int J Theor Phys, vol. 57, no. 6, pp. 1848–1863, Jun. 2018. doi:10.1007/s10773-018-3710-x

[9] W.-W. Hu, R.-G. Zhou, X.-A. Liu, J. Luo, G.-F. Luo, “Quantum image steganography algorithm based on modified exploiting modification direction embedding,” Quantum Inf Process, vol. 19, no. 5, p. 137, May 2020. doi:10.1007/s11128-020-02641-5

[10] Z. Qu, H. Sun, M. Zheng, “An efficient quantum image steganography protocol based on improved EMD algorithm,” Quantum Inf Process, vol. 20, no. 2, p. 53, Feb. 2021. doi:10.1007/s11128-021-02991-8

[11] G. Luo, R.-G. Zhou, W. Hu, “Efficient quantum steganography scheme using inverted pattern approach,” Quantum Inf Process, vol. 18, no. 7, p. 222, Jul. 2019. doi:10.1007/s11128-019-2341-3

[12] J. Luo, R.-G. Zhou, W.-W. Hu, G.-F. Luo, G. Liu, “Detection of steganography in quantum grayscale images,” Quantum Inf Process, vol. 19, no. 5, p. 149, May 2020. doi:10.1007/s11128-020-02649-x

[13] Z. Qu, Y. Huang, M. Zheng, "A novel coherence-based quantum steganalysis protocol," Quantum Inf Process, vol. 19, no. 10, p. 362, Oct. 2020. doi:10.1007/s11128-020-02868-2

[14] E. Şahin, İ. Yilmaz, "A novel quantum steganography algorithm based on LSBq for multi-wavelength quantum images," Quantum Inf Process, vol. 17, no. 11, p. 319, Nov. 2018. doi:10.1007/s11128-018-2092-6

[15] Y. Tian, J. Li, X.-B. Chen, C.-Q. Ye, H.-J. Li, "An efficient semi-quantum secret sharing protocol of specific bits," Quantum Inf Process, vol. 20, no. 6, p. 217, Jun. 2021. doi:10.1007/s11128-021-03157-2

[16] C. Navas-Merlo, J. C. Garcia-Escartin, "Detector blinding attacks on counterfactual quantum key distribution," Quantum Inf Process, vol. 20, no. 6, p. 196, Jun. 2021. doi:10.1007/s11128-021-03134-9

[17] J. Yang, Z. Li, J. Wu, H. Zhu, "One-round semi-quantum-honest key agreement scheme in MSTSA structure without entanglement," Quantum Inf Process, vol. 20, no. 5, p. 188, May 2021. doi:10.1007/s11128-021-03123-y

[18] R.-G. Zhou, X. Zhang, F. Li, "Three-party semi-quantum protocol for deterministic secure quantum dialogue based on GHZ states," Quantum Inf Process, vol. 20, no. 4, p. 153, Apr. 2021. doi:10.1007/s11128-021-03104-1

[19] C.-Y. Zhang, Z.-J. Zheng, "Entanglement-based quantum key distribution with untrusted third party," Quantum Inf Process, vol. 20, no. 4, p. 146, Apr. 2021. doi:10.1007/s11128-021-03080-6

[20] X. Gou, R. Shi, W. Gao, M. Wu, "A novel quantum E-payment protocol based on blockchain," Quantum Inf Process, vol. 20, no. 5, p. 192, May 2021. doi:10.1007/s11128-021-03126-9

[21] D. Kahn, "The history of steganography," in Information Hiding, vol. 1174, R. Anderson, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 1–5. doi:10.1007/3-540-61996-8_27

[22] P. R and I. R.J, "An overview of digital image steganography," IJCSES, vol. 4, no. 1, pp. 23–31, Feb. 2013. doi:10.5121/ijcses.2013.4102

[23] B. A. Usha, H. S. Anupama, K. N. Sangeetha, I. Gonnagar, "Image steganography using hybrid soft computing techniques–A survey," in 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), Tirunelveli, India, Feb. 2021, pp. 1081–1085. doi:10.1109/ICICV50876.2021.9388393

[24] M. Dahiya, R. Kumar, "A Literature Survey on various Image Encryption & Steganography Techniques," in 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), Jalandhar, India, Dec. 2018, pp. 310–314. doi:10.1109/ICSCCC.2018.8703368

[25] T. Jitha Raj, E. T. Sivadasan, "A survey paper on various reversible data hiding techniques in encrypted images," in 2015 IEEE International Advance Computing Conference (IACC), Banglore, India, Jun. 2015, pp. 1139–1143. doi:10.1109/IADCC.2015.7154881

[26] E. Zielińska, W. Mazurczyk, K. Szczypiorski, "Trends in steganography," Commun. ACM, vol. 57, no. 3, pp. 86–95, Mar. 2014. doi:10.1145/2566590.2566610

[27] R. J. Anderson, F. A. P. Petitcolas, "On the limits of steganography," IEEE J. Select. Areas Commun., vol. 16, no. 4, pp. 474–481, May 1998. doi:10.1109/49.668971

[28] P. Sallee, "Model-based steganography," in Digital Watermarking, vol. 2939, T. Kalker, I. Cox, and Y. M. Ro, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 154–167. doi:10.1007/978-3-540-24624-4_12

[29] S. Othman, A. Mohamed, A. Abouali, Z. Nossair, "Lossy compression using adaptive polynomial image encoding," Adv. Electr. Comp. Eng., vol. 21, no. 1, pp. 91–98, 2021. doi:10.4316/AECE.2021.01010

[30] M. Sharifzadeh, M. Aloraini, D. Schonfeld, "Adaptive batch size image merging steganography and quantized gaussian image steganography," IEEE Trans.Inform.Forensic Secur., vol. 15, pp. 867–879, 2020. doi:10.1109/TIFS.2019.2929441

[31] A. Z. Aos, A. W. Naji, S. A. Hameed, F. Othman, B. B. Zaidan, "Approved undetectable-antivirus steganography for multimedia information in PE-File," in 2009 International Association of Computer Science and Information Technology - Spring Conference, Singapore, Apr. 2009, pp. 437–441. doi:10.1109/IACSIT-SC.2009.103

[32] A. Sengupta, M. Rathor, "Crypto-based dual-phase hardware steganography for securing IP cores," IEEE Lett. of the Comput. Soc., vol. 2, no. 4, pp. 32–35, Dec. 2019. doi:10.1109/LOCS.2019.2942289

[33] K. Tutuncu, B. Demirci, "Adaptive LSB steganography based on chaos theory and random distortion," Adv. Electr. Comp. Eng., vol. 18, no. 3, pp. 15–22, 2018. doi:10.4316/AECE.2018.03003

[34] H. Xia, H. Li, H. Zhang, Y. Liang, J. Xin, "Novel multi-bit quantum comparators and their application in image binarization," Quantum Inf Process, vol. 18, no. 7, p. 229, Jul. 2019. doi:10.1007/s11128-019-2334-2

[35] M. Mastriani, "Quantum image processing: the pros and cons of the techniques for the internal representation of the image. A reply to: A comment on 'Quantum image processing?,'" Quantum Inf Process, vol. 19, no. 5, p. 156, May 2020. doi:10.1007/s11128-020-02653-1