*I.J. Image, Graphics and Signal Processing,* **2022, 5, 10-27**
Published Online on October 8, 2022 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijigsp.2022.05.02

# Transformation of Classical to Quantum Image, Representation, Processing and Noise Mitigation

**Shyam R. Sihare**
Dr. APJ Abdul Kalam Government College, Department of Computer Science and Application, Silvassa, India
E-mail: shyams_sihare1979@rediffmail.com

Received: 31 March 2022; Revised: 19 April 2022; Accepted: 27 May 2022; Published: 08 October 2022

**Abstract:** Quantum and classical computers have drastically different image representations. In a classical computer, bits are used. However, in a quantum computer, qubits are used. In this paper, the quantum image representation is the similar as the classical image representation. To represent quantum images, qubits and their associated properties have been used. Quantum imaging has previously been done via superposition. As a result, quantum imaging implemented using the superposition feature. Unitary matrices are then used to represent quantum circuits. For the quantum representation, we've gone with a modest image. To create quantum circuits, IBM's Qiskit software and Anaconda Python was used. On an IBM real time computer and an Aer simulator, a quantum circuit with 10,000 shots runs. Noise has been reduced more in the IBM real time computer than in the IBM Aer simulator. As a result, the Aer simulator's noise and qubit errors are higher than the IBM real time computer's. Quantum circuit design and image processing are both done with Qiskit programming, which is an appendix at the end of the paper. As the number of shots raise, the noise level decreases even further. Noise and qubit errors increase when the image operates at a low number of shots. Quantum image processing, noise reduction, and error correction done by circuit computation shots increase. Quantum image processing, representation, noise reduction, and error correction all make use of the quantum superposition concept.

**Index Terms:** Aer simulator, real time quantum computer, quantum image, quantum image pixels, superposition, quantum mechanics.

## 1. Introduction

Image plays a significant role in every walk of life. From childhood to the end of life, images play an important role in understanding, dissipation, summarization, and dissemination. In childhood, most of the textbooks incorporated pictures (images) to speed up the grasping power. One Chinese proverb says, "One picture is worth ten thousand words." From this proverb, we know the importance of image in daily life. Digital images are processed for a variety of reasons, such as handwritten text analysis, image recognition, medical diagnosis, and many more. In the early 1960s, we switched from classical images to digital ones. Before 1960, we were stuck with classical image patterns to learn more about the world. After 1960, world perception was changed completely by various digital means, such as mobiles, palmtops, laptops, and desktops.

The classical image is used for different applications, such as image sharpening and restoration, the medical field, remote sensing, transmission and encoding, machine or robot vision, color processing, pattern recognition, video processing, and many more. The face detection mechanism is widely used to know how to detect a person's identity. Digital forensic science has been widely used as a face detection mechanism. Physical appearance has also been used, such as fingerprint and retina scanning for safety purposes. It is well known that classical images transform from 2-dimesional to 3-dimesional, and their application areas are also wide nowadays. There is different classical image processing (CIMP) algorithms (e.g., error diffusion, Floyd-Steinberg dithering, ordered dithering, and Riemersma dithering algorithms) that are commercially used. Over time, system processing capability speeds up, hence CIMP applications are visible in every field (engineering, medical, robotics, AI, ML, ANN, and many more).

It is well known that working digitally means working with bits. Bits are in discrete patterns, so security is imposed on them by image encryption to prevent eavesdroppers. However, cyber-attacks, cyber threats, and data breaches have been growing very rapidly for a few years. Looking at digital image applications, they need to find another alternative that offers absolute security during processing and transmission. Complex images cannot be processed in the time frame required by a digital computer. In order to execute each aspect concurrently and derive a definite conclusion at the output end, different components of image processing must be considered. Because of quantum mechanics features such as superposition, quantum computers can process information much faster than classical computers. As a result, it is appropriate for processing complex images within the timeframe specified.

Quantum computing and communication (QCC) guarantee absolute security. Because QCC functionality is enabled by quantum mechanics features such as superposition, entanglement, no-cloning, particle uncertainty, and Bell’s law. Among these features, superposition and entanglement are prominent features that speed up computing and security, respectively. Quantum cryptography, communication, AI, ML, and pharmaceuticals are a few sectors where more signs of progress have been experienced within a few years. Due to its widespread application, quantum image processing (QIMP) is not exceptional.

The following are the research objectives for this study:

i) The conversion of a classical image representation to a quantum representation. The processing, transmission, operation, and manipulation of qubits and bits differs significantly from classical systems.
ii) Complex images have a higher time complexity due to inefficiencies in classical processing. As a result, complex image processing on classical computers is inefficient. Quantum computation efficiency is orders of magnitude greater than classical computing due to the superposition characteristic. Complex and structured images can thus be processed and analyzed within the timeframe necessary. Superposition not only increases picture security but also speeds up image processing. Thus, the image will be difficult to decipher for an eavesdropper.
iii) Quantum circuit design and practical realization were carried out using IBM qiskit tools and the Anaconda Python framework. The performance of quantum circuits was measured using an Aer simulator and an IBM real time quantum computer. Quantum noise and error are determined by repeatedly running a circuit with it. For more information, see the appendix at the end of the paper [1, 2].

Statements of the problem,

i) What's the main distinction between classical and quantum image representation?
ii) What is the complexity of quantum image processing in terms of computation?
iii) Make a comparison between classical and quantum systems in terms of image security, processing, and hardware and software requirements.
iv) Is it possible for quantum images to be mitigated by noise? If so, how much noise from the quantum image can be reduced by increasing the number of shots?

The research paper's limitations

i) The image generates more errors than a classical system due to quantum mechanics aspects like superposition.
ii) Quantum hardware is unlikely to be physically realized within five years. At this moment, usable software is likewise a long way off. As a result, quantum image processing is unlikely to be practicable at this time.
iii) Quantum image representation, processing, noise mitigation, and error reduction are the main areas of research for quantum computers.

Problems in research are solved.

i) A quantum computer works with qubits, whereas a classical computer works with bits. Before and during computing, qubits become unreliable. As a result of the superposition property, classically intractable problems could be solved in polynomial time by quantum systems. Some classical unsolved problems that a quantum computer could solve in polynomial runtime include finding the shortest vector of a lattice, clustered planar drawing, graph isomorphism, and k-leaf.

This research paper's accomplishments are as follows:

i) An IBM simulator and a real time quantum computer are used to bring a mathematical formulation into practice.
ii) Using a simulator and a real time quantum computer, determine the image noise.
iii) Practically, the amplitude probability of vectors (basis states) is experienced.

Section 2 of this paper discusses literature reviews. The experimental setup required for quantum image representation is explained in Section 3. Section 4 then delves into the mathematical formulation of quantum images. Noise mitigation and error reduction are achieved using IBM software and the Anaconda Python platform. In addition, results and suggestions based on experimentally gathered data were given in the same area. Finally, in the conclusion section, all of the discussion is extracted.

## 2. Literature Review

There are only a few quantum algorithms that demonstrate QCC computation speed when compared to a classical computer. On looking at its speed, people around the world show an interest in developing the same mechanism for various applications. Optimal route search, image searching, biometric detection, forensic science, are a few examples where quantum computation speed can be fully utilized for security. Within a few years, they experienced some development in the QIMP also [3].

The Flexible Representation of Quantum Images (FRQI) [4] paper demonstrated the encoding of classical image data into quantum basis states. He used quantum mechanics' superposition feature for efficient computation. There is no straightforward algorithm for QIMP like there is for CIMP. Different quantum gates, namely CNOT, X (NOT), H (Hadamard), Rotation ($\theta$) are used for QIMP operations. In a quantum circuit, the classical image data is encoded into quantum basis states. FRQI represents classical image pixel representation in quantum basis state representation by quantum circuits. He showed greyscale as well as color classical image representation in an efficient way compared to classical optimal algorithms.

A multi-dimensional color image processing mechanism has been proposed by NAQSS (Normal Arbitrary Quantum Superposition State). Multi-dimensional images need high storage and efficient computational capability. It was a very difficult task to represent such images as QIMPs. He focused on image segmentation and content-based image search efficiently through quantum computing [5].

Li et al. used a three-way set for quantum basis state processing [6]. The QSMC (Quantum States for M Colors of an Image) is used for color representation, whereas the QSNC (Quantum States for N Coordinates of an Image) is used for coordinates. S. Yuan et al. explained pixel information stored in a quantum state [7]. The pixel information of an infrared image is converted into radiant energy and then into a quantum state by the converter. The QUALPI uses polar coordinates for the representation of pixel information in quantum state forms [8]. It can be used, namely for affine transformation operations. The NEQR is used for the representation of images in color format. The FRQI is used for greyscale image processing, whereas the NEQR is used for color image processing. The FRQI has adapted the NEQR methods and extended them for color image processing [9].

The laws of classical pixel representation are governed by quantum image representation and compilation. Quantum greyscale code frequently employs the binary pattern. The retrieval of stored image states was normalized using Hadamard states. Only greyscale photos are supported by FRQI. It is not compatible with quantum color images. Color images that are three-dimensional or multi-dimensional require more storage capacity on a normal computer. As a result, image segmentation maintenance is a difficult task. Furthermore, in low-processing devices, multicolor and multidimensional image efficiency and accuracy are limited. On a classical computer, looking for a sub-image takes a long time. The NAQSS algorithm proposes a solution to all of these issues. For color, position, and segmentation information, the NAQSS method uses less qubits. The efficiency and accuracy are higher than other quantum and classical images representation techniques due to the smaller amount of qubits. QSMC is generally used for basis states, which means it deals with color information, whereas QSNC is primarily used for image storage, which means it deals with image coordinates [10]. Except for pixel measurement, this quantum technique is particularly beneficial for image segmentation and compression. It is turned into quantum image basic states and employed in infrared for image pixel representation. A converter transforms infrared into qubit states. The retrieval of color information for further processing and storage is far more difficult than the depiction of color images [11].

The CQIR (Caraiman's Quantum Image Representation) technique is used to store pixel coordinates in a quantum register. Another qubit register has been used to store the corresponding color details. The greyscale color coding employs a superposition state. It comes in handy when working with layered images. For greyscale images, this approach is simpler to implement. It is more secure when transmission from one location to another because of the superposition state [12].

Greyscale color images are processed and retrieved using the MCQI (Multi-Channel Quantum Images) technique. When an image becomes more complicated, obtaining pixel coordinate information becomes a time-consuming process. For a three-dimensional image, this technique uses RGB (Red, Green, and Blue) color. Color information is encoded via controlled rotation and superposition gates. A controlled phase gate creates the image color. MCQI is less useful in watermarking and movie making [13].

The INEQR (Improved NEQR) is a superset of the NEQR algorithm that was previously proposed. Horizontal and vertical scaling present disparity in NEQR. This algorithm eliminated this stumbling block. In comparison to the previously proposed NEQR algorithm [14], it improves the image information retrieval approach. Greyscale images are processed using the NEQR and INEQR algorithms, whilst multicolor quantum images are processed using the GNEQR (A Generalized Model of NEQR) method [15].

NCQI (A Novel Quantum Representation of Color Digital Images) is utilized for image processing in superposition. This approach retrieves all pixels and color information at the same time. For quantum greyscale and color images, color information sequences are red, green, and blue. The majority of the ideas are based on the NEQR algorithm [16].

For greyscale images, the BRQI (A Bitplane Representation of Quantum Images) is utilized in the 8-bitplane. These bitplanes are superimposed on the quantum image obtained by GNEQR. Bitplanes, on the other hand, make it 16 times faster than GNEQR. For quantum color images, the bitplanes can be expanded to 24 bitplanes. In comparison to GNEQR and NCQI, BRQI bitplanes are less costly to implement [17].

QRMMI (A Quantum Representation Model for Multiple Pictures) is a quantum representation model for multiple greyscale quantum images that is particularly useful for storage. Quantum images relating to color and pixel position are created from greyscale digital images [18]. For the basis states of qubits, QRMW (Quantum Representation of Multi-Wavelength Images) is employed at various wavelengths. For storing position, wavelength, and color detail, three registers were used. Because superposition is frequently employed for position, wavelength, and greyscale information, image processing can be done much faster than before [19].

For entangled basis states, the QMCR (A Digital RGB Multi-Channel Representation for Quantum Colored Images) technique is utilized. Pixel location and color are encoded using entangled qubits. It works well with quantum color images. As a result, it has a wide range of applications, including communication, processing, watermarking, steganography, and data concealment. It's a follow-up to the NEQR work. The IFRQI algorithm uses less storage than other quantum algorithms. It employs entangled qubit sequences to encode position and color code instead of superstition [20].

More than pixel position and color ordered images are indexed image processing and searching capabilities. To make calculation more versatile, color and coordinates are placed in a matrix [21, 22].

The OQIM (Order-encoded Quantum Image Model) [23], QBIR (A Quantum Block Image Representation) [24], IFRQI [25], FRQCI (Improved Flexible Representation of Quantum Images) [26], SQR (Simple Quantum Representation of Infrared Images) [11], and QRCI (A New Quantum Representation Model of Color Digital Images) [27] research papers addressed quantum image processing on grey-scale or (and) color pixels. Grover's algorithm was designed to find a specific entity among a collection of unstructured entities [28, 29, 30, 31]. This mechanism has the potential to be used to locate quantum image pixels. Bloch spheres are used in quantum image processing. This paper will discuss classical computer image representation in pixels in Section 1. Later, quantum gates are used to represent the same classical image in qubit format (section 2). Finally, in the methodology and results section, we will discuss quantum image noise. The IBM qiskit software and Anaconda Python platform is used for the measurement of quantum image qubits and their noisy ratio.

## 3. Method

### *3.1. Quantum Counterpart of Classical Image Representation*

Pixels in 2- or 3-dimensional coordinates are used to represent classical images. Each pixel has its own address in memory for processing and storage. When dealing with complex images, image processing becomes a difficult task. Each pixel has a pixel value, a color combination value, and an orientation. Classical computer processing speeds are incapable of handling complex images such as face recognition in a crowd, image extraction, and processing. On a classical computer, such work takes time. It is necessary to find an alternative. When it comes to features like superposition, a quantum computer is a good alternative. It provided complete security when transmitting images from one location to another via guided or (and) unguided media. A quantum computer is working on features of quantum mechanics. As a result, combining classical and quantum computers is a difficult task. Combining classical physics (a classical computer has features similar to classical physics) and quantum physics (a quantum computer has features similar to quantum mechanics) is difficult.

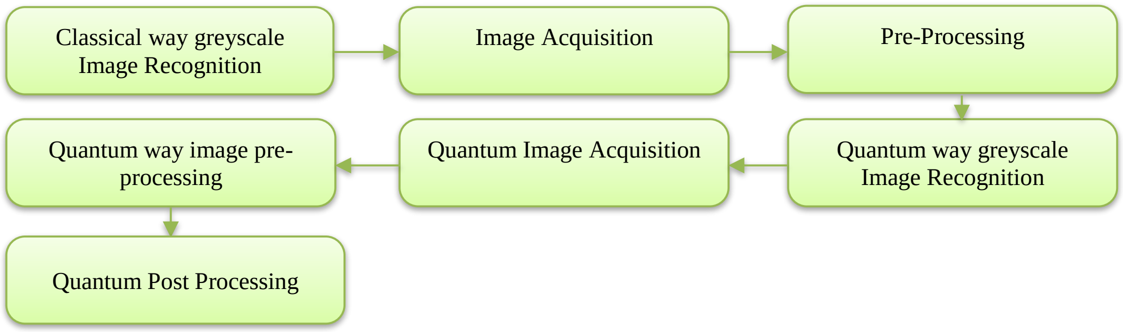

Classical way greyscale Image Recognition
Image Acquisition
Pre-Processing
Quantum way greyscale Image Recognition
Quantum Image Acquisition
Quantum way image pre-processing
Quantum Post Processing

Fig.1. Classical grey color image transformation into quantum image

Fig. 1 indicates the classical image transformation in a quantum image. First, a greyscale classical image is identified with pixel information. After that, image acquisition has been done, like color coding, pixel information, pixel address, and image frames. Outlier detection, removing unwanted noisy data, and detecting and correcting errors are all achieved through image preprocessing. That is, in image preprocessing, image segmentation is performed to speed up the processing of an image. The Quantum Way greyscale image recognition block identifies the classical greyscale image. The color code, pixel information, pixel address, and image frame are transformed in a quantum way by the quantum image acquisition block. Quantum image pre-and post-processing extracts classical pixel information and its relevant information and converts it into quantum mechanics qubits for further quantum image processing.

A pixel is the smallest addressable unit of a computer display unit. Image processing has been done by individual pixels as per requirements. Let us consider $P_{(x,y)}$ is a pixel from pixels set $S=\sum_{x=0}^{2^n-1}\sum_{y=0}^{2^m-1}P_{(x,y)}$ with $((x,y,n,m)\geq 0)$ where $x$ and $y$ are coordinates and let the pixel address be $A=\sum_{x=0}^{2^n-1}\sum_{y=0}^{2^n-1}P_{(x,y)}$ whatever it wants to be. The pixel $P_{(x,y)}(i),\ 0\leq i\leq n$ represents the current position. And $P_{(x,y)}(i-1)$, $P_{(x,y)}(i)$, and $P_{(x,y)}(i+1)$ are sequences of pixels. Fig. 2 indicates the classical way of representing an image with binary values. Every image pixel and color value are stored in memory with their respective addresses. There is $4\times 4$ a matrix with a corresponding bit sequence such as 0110, 1001, 1111, and 10001. A square matrix should be essential to transform a classical image into a quantum image. Later, the bit sequences 0110, 1001, 1111, and 1001 are converted into qubits basis states such as $|0110\rangle$, $|1001\rangle$, $|1111\rangle$, and $|1001\rangle$.

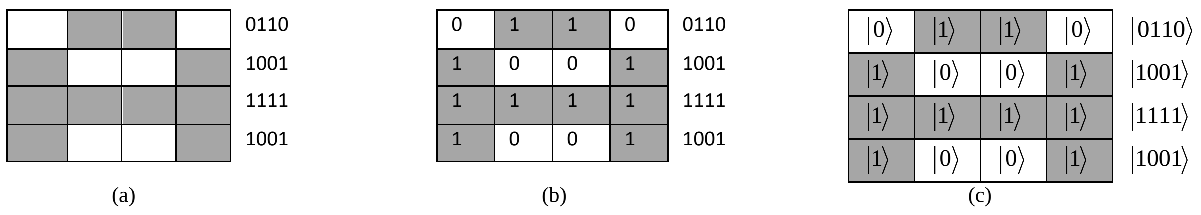

0110
1001
1111
1001
0 1 1 0
1 0 0 1
1 1 1 1
1 0 0 1
$|0\rangle$ $|1\rangle$ $|1\rangle$ $|0\rangle$
$|1\rangle$ $|0\rangle$ $|0\rangle$ $|1\rangle$
$|1\rangle$ $|1\rangle$ $|1\rangle$ $|1\rangle$
$|1\rangle$ $|0\rangle$ $|0\rangle$ $|1\rangle$
$|0110\rangle$
$|1001\rangle$
$|1111\rangle$
$|1001\rangle$
(a)
(b)
(c)

Fig.2. (a) Classical greyscale image representation (b) pixel representation by bit and corresponding binary address (c) quantum way image representation of corresponding classical greyscale image

Fig. 3 shows the quantum circuit of Fig. 2(c). The $q_i$, $0\leq i\leq 7$ are qubit input lines and $c$ are classical bit output lines. The first four qubit input lines accept basis states such as $|x_3x_2x_1x_0\rangle$ where $x_j=\{0,1\},?\ \leq j\leq 3$. The Hadamard gate $(H)$ applies to the first four qubits of Hilbert space. The $q_{0\leq j\leq 7}=|0\rangle$ basis state and the $H$ gate apply to the superposition of the basis state $q_{0\leq j\leq 3}=|0\rangle$. The $q_{4\leq j\leq 7}=(H\otimes I)\otimes(X\otimes I)|x_7x_6x_5x_4\rangle$ where $X$ is Pauli-X gate, $I$ is identity gate (matrix), and $H$ is Hadamard gate. The C-NOT gate is then used for basis states $|0110\rangle$, $|1001\rangle$, $|1111\rangle$, and $|1001\rangle$, and it is applied to the qubit input lines when the qubit is $|1\rangle$ basis states. The output of the basis state are $q_0=|0110\rangle$, $q_0=|1001\rangle$, $q_0=|1111\rangle$, and $q_0=|1001\rangle$ before measurement.

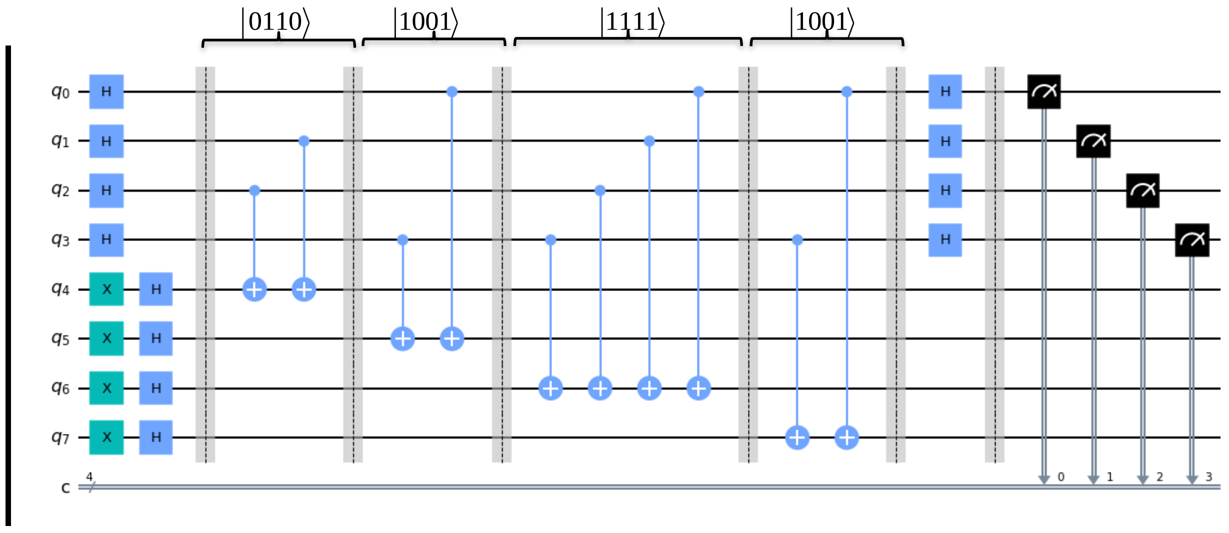


Fig.3. Quantum circuit representation of Fig. 2

The following is the matrix representation of Pauli-X $(X)$, $CNOT$, and Hadamard $(H)$ gates:

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, CNOT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \text{and } H = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \tag{1}$$

A quantum circuit is initialized by $|0\rangle$. The gates $X$ and $H$ are single qubits, whereas $CNOT$ is two-qubit gates. The $X$ and $CNOT$ act as control qubit gates, whereas $H$ act as qubit superposition. The $CNOT$ gate controls the input qubit and alters the target qubit if the control qubit is $|1\rangle$, if the control qubit is $|0\rangle$, then the target qubit is unaltered as indicated in Fig. 3.

$$i.e. |10\rangle \overset{CNOT}{\rightleftarrows} |11\rangle \tag{2}$$

Basis states $q_{i\,(3 \le i \le 7)}$ act as ancilla (garbage basis states) qubits to generate required outputs (i.e. $|0110\rangle$, $|1001\rangle$, $|1111\rangle$, and $|1001\rangle$). After measurement, the basis states transform into binary.

$$q_{i\,(0 \le i \le 3)} = H \otimes CNOT \otimes (H \otimes I) |x_{i+3} x_{i+2} x_{i+1} x_i\rangle \tag{3}$$

Eq. (3) generates $|0110\rangle$, $|1001\rangle$, $|1111\rangle$, and $|1001\rangle$ basis states. After measurement, it converts into binary.

$$q_{i\,(3 \le i \le 7)} = (H \otimes I) \otimes (X \otimes I) \otimes CNOT |\text{ancilla basis states in superpositon}\rangle \tag{4}$$

Fig 2 (c) indicates that each pixel is represented in a basis state (vector) according to FRQI [4].

$$|I\rangle = \frac{1}{2^n} \sum_{i=0}^{2^{2n}-1} |c_i\rangle \otimes |i\rangle \tag{5}$$

and,

$$|c_i\rangle = \cos\theta_i |0\rangle + \sin\theta_i |1\rangle \tag{6}$$

where $|0\rangle$ and $|1\rangle$ are the two-dimensional basis states, $|i\rangle$ is two-dimensional quantum state, $i = 0, 1, \ldots, 2^{2n} - 1, \theta_i \in [0, \pi/2]$ *and* $\theta = (\theta_0, \theta_1, \ldots, \theta_{2^{2n}-1})$.

To depict a quantum image, a matrix is utilized. As shown in Fig. 2 (a), (b), and (c), an index represents an image address, while quantum pixels are represented by a superposition of basis states. Each quantum pixel address by pixel address is subjected to superposition processing in real time. For greyscale color information, superposition basis states are used. The coordinates of those quantum pixels became active, storing color information. Quantum circuits are created using the active pixel address, as seen in Fig. 3. In order to generate pixel addresses, the circuit utilized Hadamard, CNOT, and Pauli-X gates. The two-dimensional quantum image is the focus of this study. This study's scope does not extend beyond two dimensions.

## 4. Experiments

### *4.1. Quantum Image Noise Mitigation*

Noise is present in a quantum image because qubits are in superposition states $1/\sqrt{2}(|0\rangle \pm |1\rangle)$. Superposition states are uncertain during computation or before measurement. With that, basis states are present in a vector space, and the vector space itself is the Hilbert space. So, vector behavior becomes unpredictable in a vector space before measurement. During quantum computation, every qubit presents noise. Quantum gates also generate some noise during computation. So, at measurement time, the noise proportion around the threshold point. There is a need to mitigate noise before measurement for quantum image filtration. This research article used the IBM Aer simulator and a real quantum computer to know quantum image noise. Make an effort to mitigate quantum image noise.

Fig. 4 (a) shows the quantum image and the surrounding noise. Representation of Fig. 4 as

$$|\psi\rangle = \frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{x=0}^{2^n-1}|f(y,x)\rangle \otimes |yx\rangle \tag{7}$$

$$= \frac{1}{2^n}\sum_{y=0}^{2^n-1}\sum_{x=0}^{2^n-1}\otimes_{i=0}^{q-1}|c_{yx}^{i}\rangle \otimes |yx\rangle \tag{8}$$

where $f(y,x) = c_{yx}^{0}\ldots c_{yx}^{q-2}c_{yx}^{q-1}$, $c_{yx}^{k} \in [0,1]$ and $f(y,x) = [1, 2^q - 1]$.

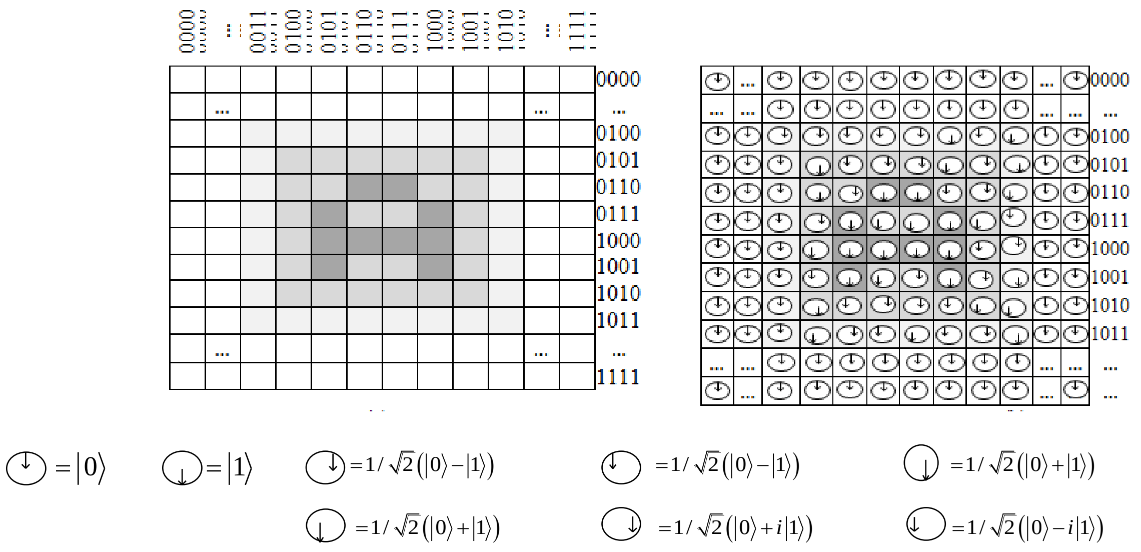

0000
...
0011
0100
0101
0110
0111
1000
1001
1010
...
1111
0000
...
0100
0101
0110
0111
1000
1001
1010
1011
...
1111
$=|0\rangle$
$=|1\rangle$
$=1/\sqrt{2}(|0\rangle-|1\rangle)$
$=1/\sqrt{2}(|0\rangle-|1\rangle)$
$=1/\sqrt{2}(|0\rangle+|1\rangle)$
$=1/\sqrt{2}(|0\rangle+|1\rangle)$
$=1/\sqrt{2}(|0\rangle+i|1\rangle)$
$=1/\sqrt{2}(|0\rangle-i|1\rangle)$

Fig.4. Quantum image representation

The noise basis states are in superposition states in a quantum image before measurement. After measurement, superposition states collapse into discrete binary states. So, discrete binary states are probabilistic after measurement, which has created noise in an image. Namely superposition basis state is $\left[1/\sqrt{2}(|0\rangle \pm |1\rangle), 1/\sqrt{2}(|0\rangle \pm i|1\rangle)\right]$ that collapses into either $|0\rangle$ or $|1\rangle$ after measurement as output. Due to superposition, images generate noise that needs to be mitigated. At a particular pixel, there is no information regarding the image, and it generates $|1\rangle$ instead of $|0\rangle$. Grover's algorithm needs to apply for phase inversion from $|1\rangle$ to $|0\rangle$. For example, a method to reduce noise from quantum images of the rest of the pixels

Fig. 4 (b) represents the Bloch sphere vector orientation. A vector could be in any position during and after computation.

$$|\psi\rangle = \frac{1}{2^{2n}}\sum_{i=0}^{2^{2n}-1}\alpha_i \otimes |x_{i+3}x_{i+2}x_{i+1}x_i\rangle \tag{9}$$

where $\alpha_i$ is amplitude probability, $|x_{i+3}x_{i+2}x_{i+1}x_i\rangle$ is 2n-dimensional quantum state, $0 \le i \le 15$ and $\sum_{i=0}^{2^{2n}-1}|\alpha_i|^2 = 1$. The $\otimes$ is tensor product notation.

Let $\sum_{i=0}^{2^{2n}-1}|\psi_i\rangle = |x_{i+3}x_{i+2}x_{i+1}x_i\rangle$ and by Fig. 4 (a) and (b), we can rewrite Eq. (9) as,

$$|\psi\rangle = \frac{1}{2^{2n}}\sum_{i=0}^{2^{2n}-1}\alpha_i \otimes |\psi_i\rangle, n = 3 \tag{10}$$

The $|\psi_i\rangle$ is linear equation for 2n-dimesional quantum image address with respect to amplitude probability $\alpha_i$. Quantum image address is in superposition states. After measurement, only one amplitude high and rest of amplitude values collapsed toward of 0. So, $|\psi_i\rangle$ rewrite as,

$$|\psi_i\rangle = \frac{1}{2^{2p_o}} \sum_{p_o=0}^{2^{2p_o}-1} \alpha_{p_o} \otimes \left| x_{p_o+3} x_{p_o+2} x_{p_o+1} x_{p_o} \right\rangle + \frac{1}{2^{2p_1}} \sum_{p_1=0}^{2^{2p_1}-1} \alpha_{p_1} \otimes \left| x_{p_1+3} x_{p_1+2} x_{p_1+1} x_{p_1} \right\rangle + \ldots + \frac{1}{2^{2p_7}} \sum_{p_1=0}^{2^{2p_7}-1} \alpha_{p_7} \otimes \left| x_{p_7+3} x_{p_7+2} x_{p_7+1} x_{p_7} \right\rangle \tag{11}$$

where $i = p_o + p_1 + \ldots + p_7$ , $i, p_0 \ldots p_7 > 0$ and $\left|\alpha_{p_0}\right|^2 + \ldots + \left|\alpha_{p_7}\right|^2 = 1$.

The image pixel address is represented as $|\psi_0\rangle, |\psi_1\rangle, \cdots, |\psi_7\rangle$ . $|\psi_0\rangle$ denote those pixels where image information is absent. Similarly, $|\psi_1\rangle$ represent pixel addresses that do not have image information. Pixel addresses $|\psi_2\rangle, \cdots, |\psi_7\rangle$ used for image noise are transferred to the nearest of $|\psi_1\rangle$ . In noise, existing pixel addresses are in superposition states, whereas the image is present and absent using a rectilinear vector. Rectilinear states are either $|0\rangle$ or $|1\rangle$ at a time.

$$|\psi_0\rangle = \frac{1}{2^{2p_0}} \sum_{p_0=0}^{2^{2p_0}-1} \alpha_{p_0} \otimes \left| x_{p_0+3} x_{p_0+2} x_{p_0+1} x_{p_0} \right\rangle = |0\rangle \tag{12}$$

The image basis states (qubits) of a quantum image rotate by $\theta_{p_i (0 \le i \le 7)}$ to identify the basis states. The $|0\rangle$ and $|1\rangle$ rotate according to the Bloch sphere in their 2-dimensional base state. So, $|\psi_0\rangle$ can be written in polar coordinates as,

$$|\psi_0\rangle = \frac{1}{2^{2p_0}} \sum_{p_0=0}^{2^{2p_0}-1} \left( \cos(\theta_{p_0}) |0\rangle + i \sin(\theta_{p_0}) |1\rangle \right) \otimes \left| x_{p_0+3} x_{p_0+2} x_{p_0+1} x_{p_0} \right\rangle = |0\rangle \tag{13}$$

where, $|0\rangle$ and $|1\rangle$ are the two-dimensional basis states, $\left| x_{p_0+3} x_{p_0+2} x_{p_0+1} x_{p_0} \right\rangle$ is two-dimensional quantum state, $p_0 = 0,1,\ldots,2^{2n}-1, \theta_{p_0} \in \pi/2$ *and* $\theta = \left(\theta_0, \theta_1, \ldots, \theta_{2^{2n}-1}\right)$.

$$|\psi_1\rangle = \frac{1}{2^{2p_1}} \sum_{p_1=0}^{2^{2p_1}-1} \alpha_{p_1} \otimes \left| x_{p_1+3} x_{p_1+2} x_{p_1+1} x_{p_1} \right\rangle = |1\rangle \tag{14}$$

So, $|\psi_1\rangle$ can be written in polar coordinates as,

$$|\psi_1\rangle = \frac{1}{2^{2p_1}} \sum_{p_1=0}^{2^{2p_1}-1} \left( \cos\left(\theta_{p_1}\right) |0\rangle + i \sin\left(\theta_{p_1}\right) |1\rangle \right) \otimes \left| x_{p_1+3} x_{p_1+2} x_{p_1+1} x_{p_1} \right\rangle = |1\rangle \tag{15}$$

where, $|0\rangle$ and $|1\rangle$ are the two-dimensional basis states, $\left| x_{p_1+3} x_{p_1+2} x_{p_1+1} x_{p_1} \right\rangle$ is two-dimensional quantum state, $p_1 = 0,1,\ldots,2^{2n}-1, \theta_{p_1} \in 3\pi/2$ and $\theta = \left(\theta_0, \theta_1, \ldots, \theta_{2^{2n}-1}\right)$.

$$|\psi_2\rangle = \frac{1}{2^{2p_2}} \sum_{p_2=0}^{2^{2p_2}-1} \alpha_{p_2} \otimes \left| x_{p_2+3} x_{p_2+2} x_{p_2+1} x_{p_2} \right\rangle = \frac{1}{\sqrt{2}} \left( |0\rangle + i |1\rangle \right) \tag{16}$$

So, $|\psi_2\rangle$ can be written in polar coordinates as,

$$|\psi_2\rangle = \frac{1}{2^{2p_2}} \sum_{p_2=0}^{2^{2p_2}-1} \left( \cos\left(\theta_{p_2}\right) |0\rangle + i \sin\left(\theta_{p_2}\right) |1\rangle \right) \otimes \left| x_{p_2+3} x_{p_2+2} x_{p_2+1} x_{p_2} \right\rangle = \frac{1}{\sqrt{2}} (|0\rangle + i|1\rangle) \tag{17}$$

where, $|0\rangle$ and $|1\rangle$ are the two-dimensional basis states, $\left| x_{p_2+3} x_{p_2+2} x_{p_2+1} x_{p_2} \right\rangle$ is two-dimensional quantum state, $p_2 = 0,1,\ldots,2^{2n}-1, \theta_{p_2} \in 0$ and $\theta = \left(\theta_0, \theta_1, \ldots, \theta_{2^{2n}-1}\right)$.

$$|\psi_3\rangle = \frac{1}{2^{2p_3}} \sum_{p_3=0}^{2^{2p_3}-1} \alpha_{p_3} \otimes \left| x_{p_3+3} x_{p_3+2} x_{p_3+1} x_{p_3} \right\rangle = \frac{1}{\sqrt{2}} (|0\rangle - i|1\rangle) \tag{18}$$

So, $|\psi_3\rangle$ can be written in polar coordinates as,

$$|\psi_3\rangle = \frac{1}{2^{2p_3}} \sum_{p_3=0}^{2^{2p_3}-1} \left(\cos\left(\theta_{p_3}\right)|0\rangle + i\sin\left(\theta_{p_3}\right)|1\rangle\right) \otimes \left|x_{p_3+3}x_{p_3+2}x_{p_3+1}x_{p_3}\right\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - i|1\rangle\right) \tag{19}$$

where, $|0\rangle$ and $|1\rangle$ are the two-dimensional basis states, $\left|x_{p_3+3}x_{p_3+2}x_{p_3+1}x_{p_3}\right\rangle$ is two-dimensional quantum state, $p_3 = 0,1,\ldots,2^{2n}-1, \theta_{p_3} \in \pi$ and $\theta = \left(\theta_0, \theta_1, \ldots, \theta_{2^{2n}-1}\right)$.

$$|\psi_4\rangle = \frac{1}{2^{2p_4}} \sum_{p_4=0}^{2^{2p_4}-1} \alpha_{p_4} \otimes \left|x_{p_4+3}x_{p_4+2}x_{p_4+1}x_{p_4}\right\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \tag{20}$$

So, $|\psi_4\rangle$ can be written in polar coordinates as,

$$|\psi_4\rangle = \frac{1}{2^{2p_4}} \sum_{p_4=0}^{2^{2p_4}-1} \left(\cos\left(\theta_{p_4}\right)|0\rangle + i\sin\left(\theta_{p_4}\right)|1\rangle\right) \otimes \left|x_{p_4+3}x_{p_4+2}x_{p_4+1}x_{p_4}\right\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \tag{21}$$

where, $|0\rangle$ and $|1\rangle$ are the two-dimensional basis states, $\left|x_{p_4+3}x_{p_4+2}x_{p_4+1}x_{p_4}\right\rangle$ is two-dimensional quantum state, $p_4 = 0,1,\ldots,2^{2n}-1, \theta_{p_4} \in \left[\approx \frac{\pi}{4}\right]$ and $\theta = \left(\theta_0, \theta_1, \ldots, \theta_{2^{2n}-1}\right)$. State $|\psi_5\rangle$ is like $|\psi_4\rangle$ with $\theta_{p_5} \in \left[\approx \frac{3\pi}{4}\right]$.

So, Fig. 4 (a) is redesigned as,

| 0000 | ⋮ | 0011 | 0100 | 0101 | 0110 | 0111 | 1000 | 1001 | 1010 | ⋮ | 1111 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 0000 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | ⋯ |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_5\rangle$ | $\|\psi_4\rangle$ | $\|\psi_6\rangle$ | $\|\psi_3\rangle$ | $\|\psi_6\rangle$ | $\|\psi_4\rangle$ | $\|\psi_6\rangle$ | $\|\psi_5\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 0100 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_4\rangle$ | $\|\psi_2\rangle$ | $\|\psi_3\rangle$ | $\|\psi_6\rangle$ | $\|\psi_2\rangle$ | $\|\psi_7\rangle$ | $\|\psi_2\rangle$ | $\|\psi_7\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 0101 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_6\rangle$ | $\|\psi_3\rangle$ | $\|\psi_6\rangle$ | $\|\psi_1\rangle$ | $\|\psi_1\rangle$ | $\|\psi_3\rangle$ | $\|\psi_7\rangle$ | $\|\psi_5\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 0110 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_7\rangle$ | $\|\psi_5\rangle$ | $\|\psi_1\rangle$ | $\|\psi_4\rangle$ | $\|\psi_2\rangle$ | $\|\psi_1\rangle$ | $\|\psi_2\rangle$ | $\|\psi_6\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 0111 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_4\rangle$ | $\|\psi_7\rangle$ | $\|\psi_1\rangle$ | $\|\psi_1\rangle$ | $\|\psi_1\rangle$ | $\|\psi_1\rangle$ | $\|\psi_3\rangle$ | $\|\psi_4\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 1000 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_5\rangle$ | $\|\psi_2\rangle$ | $\|\psi_1\rangle$ | $\|\psi_2\rangle$ | $\|\psi_6\rangle$ | $\|\psi_1\rangle$ | $\|\psi_7\rangle$ | $\|\psi_7\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 1001 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_7\rangle$ | $\|\psi_7\rangle$ | $\|\psi_3\rangle$ | $\|\psi_5\rangle$ | $\|\psi_4\rangle$ | $\|\psi_7\rangle$ | $\|\psi_6\rangle$ | $\|\psi_7\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 1010 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_5\rangle$ | $\|\psi_7\rangle$ | $\|\psi_5\rangle$ | $\|\psi_7\rangle$ | $\|\psi_6\rangle$ | $\|\psi_6\rangle$ | $\|\psi_7\rangle$ | $\|\psi_5\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 1011 |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | ⋯ |
| $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | $\|\psi_0\rangle$ | 1111 |

Fig.5. Quantum image representation

$$|\psi_6\rangle = \frac{1}{2^{2p_6}} \sum_{p_6=0}^{2^{2p_6}-1} \alpha_{p_6} \otimes \left|x_{p_6+3}x_{p_6+2}x_{p_6+1}x_{p_6}\right\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \tag{22}$$

So, $|\psi_6\rangle$ can be written in polar coordinates as,

$$|\psi_6\rangle = \frac{1}{2^{2p_6}} \sum_{p_6=0}^{2^{2p_6}-1} \left(\cos\left(\theta_{p_6}\right)|0\rangle + i\sin\left(\theta_{p_6}\right)|1\rangle\right) \otimes \left|x_{p_6+3}x_{p_6+2}x_{p_6+1}x_{p_6}\right\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \tag{23}$$

where, $|0\rangle$ and $|1\rangle$ are the two-dimensional basis states, $|x_{p_6+3}x_{p_6+2}x_{p_6+1}x_{p_6}\rangle$ is two-dimensional quantum state, $p_6 = 0,1,\ldots,2^{2n}-1, \theta_{p_6} \in \left[\approx \frac{5\pi}{4}\right]$ and $\theta = \left(\theta_0, \theta_1, \ldots, \theta_{2^{2n}-1}\right)$. State $|\psi_7\rangle$ is like $|\psi_6\rangle$ with $\theta_{p_7} \in \left[\approx \frac{7\pi}{4}\right]$.

If $\left\||\psi_0\rangle\right|^2 = 1$ after measurement, then $\left\||\psi_1\rangle\right|^2 \ldots \left\||\psi_7\rangle\right|^2 = 0$. If $\left\||\psi_1\rangle\right|^2 = 1$ after measurement, then $\left\||\psi_0\rangle\right|^2$ and $\left\||\psi_2\rangle\right|^2 \ldots \left\||\psi_7\rangle\right|^2 = 0$. Like that other state measurement performed in Fig. 5.

Due to computation and quantum gate noise, the output image is not correct. The superposition contributes significantly to the increase in qubit noise during computation. During qubit transmission and computation, the original image pixel values were changed. Throughout computation, the noise varies in value, and the error rate decreases or increases. To reduce noise, the number of shots, which could be a real quantum computer computation or an Aer simulator, must increase exponentially. The QBER (Quantum Bit Error Rate) should be low after measurement. If the QBER is greater than 20%, it is hard to regain the original image after measurement.

Let $n = |x_{n+3}x_{n+2}x_{n+1}x_n\rangle$ where $n = 1\ldots3$ and $|\psi\rangle = \sum_{i=0}^{2^n-1}\sum_{j=0}^{2^n-1}\alpha_{i,j} \otimes |n_{i,j}\rangle$, where $n = 0\ldots15$ ( $2^n \times 2^n$ qubit pixel image). From $2^n \times 2^n$ qubit strings, extract $n$ qubit strings to know the quantum image. The $n$ is the quantum pixel image where it is lying, whereas the $2^n \times 2^n$ is a qubit pixel image. The x-coordinate is considered a row, while the y-coordinate is a column. From $2^n \times 2^n$ pixel image $n = |\psi_1\rangle$. The $(2^n \times 2^n) - n$ is in superposition, or $|0\rangle$ state. The $(2^n \times 2^n)$ qubit strings generate quantum noise (QBER). If $(2^n \times 2^n)$ increases, then the noise in the quantum image also increases to extract the required sequence of qubit strings. Noise propagates after measurement as a binary image. The 2-qubit (i.e. *CNOT*, *C-Z*, *C-S*, *C-T*, and *SWAP* ($C$ is controlled)) and more than 2-qubit gates (i.e. *CCNOT*) generate more noise than single qubit gates (i.e. Pauli-$X$, $Y$, $Z$).

The definite basis states that are extract is,

$$|\psi_1\rangle = \frac{1}{2^{2n}}\sum_{i=0}^{2^n-1}\sum_{j=0}^{2^n-1}\alpha_{(i,j)} \otimes \left|x_{(i,j)+3}x_{(i,j)+2}x_{(i,j)+1}x_{(i,j)}\right\rangle \tag{24}$$

from,

$$|\psi\rangle = \frac{1}{2^{2n}}\sum_{i=0}^{2^n-1}\sum_{j=0}^{2^n-1}\alpha_{(i,j)} \otimes |\psi_i\rangle \tag{25}$$

To mitigate the quantum image's surrounding noise, we need to extract $|\psi_1\rangle$ basis states (pixels) from $|\psi\rangle$. The IBM Aer simulator and real time quantum computer are shot (iterated) up to 10,000 times. So, $2^n \times 2^n$ the quantum image result could be as follows (see appendix for more information):

$$|\psi\rangle = \frac{1}{2^{2n}}\sum_{i=0}^{2^n-1}\sum_{j=0}^{2^n-1}\alpha_{(i,j)} \otimes |\psi_i\rangle = 10000\text{times} \tag{26}$$

If shots or iterations increase, then noise mitigates more, as shown in Fig. 6 (c) and Fib. 7 (a). Most of the pixel basis states are in a superposition of eq. (26). So, noise also increases at the output end after measurement. The non-superposition states generate low or negligible noise for quantum images. Aero simulator shots 10000 distributed among eq. (26) basis states randomly depending upon superposition and non-superposition basis states. The eq. (26) Aer simulator shot outcomes are present in appendix (line number 36).

The Aer simulator shots are repeated $2^n\left(n=4\right)$ times, and the results of each simulator shot are represented in a matrix as $M$ (appendix line number 39). The Aer simulator calculates the $C$ count vector, and 10000 runs it. Any vector in Hilbert space has an amplitude probability. After measurement, the amplitude cannot be 0 or 1. The base state amplitude of a quantum circuit is generated while it runs. Statistical variation amplitude values should be generated for each run. The several statistical variant values generated and organized in a matrix $M$. Now, multiply $C$ and $M$ to find out what the quantum image basis states are.

$$C_{noise} = M.C \tag{27}$$

$$C_{noise} = \begin{bmatrix} 629.1456 & 625.3512 & 627.3864 & 627.9256 & \cdots & 608.4352 & 630.2032 & 619.7192 \end{bmatrix}^T \tag{28}$$

The $C_{noise}$ predicted outcomes of Bell's states are more accurate than the outcomes of every $2^n \times 2^n$ basis state in Hilbert space. The $C_{noise}$ is not ideal data for predicting the correct quantum noise. So, find $C_{ideal}$ by inverse matrix $M$ as,

$$M = M^{-1} \tag{29}$$

$$M^{-1} = \begin{bmatrix} -28.27994 & -9.54925 & -64.40336 & 26.88759 & \cdots & 114.11443 & 166.24966 & 2.98106 \\ -24.5931 & -14.11132 & -12.34712 & 33.32099 & \cdots & -97.07508 & 206.93766 & -6.72851 \\ 40.50503 & -57.91316 & 92.93689 & -7.31078 & \cdots & -74.29684 & 264.95367 & -186.61303 \\ 214.80532 & 161.69231 & 70.19809 & 36.43661 & \cdots & -183.52281 & 136.87904 & -177.74321 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 164.53833 & 156.80997 & 151.00659 & -240.38081 & \cdots & -178.84033 & -188.4026 & 44.6147 \\ -121.35112 & -5.56762 & 73.98244 & 41.66303 & \cdots & 125.56655 & -29.2616 & 79.16682 \\ 136.68909 & 155.36677 & 144.58613 & -113.30337 & \cdots & -168.95268 & 71.64501 & -97.64539 \end{bmatrix} \tag{30}$$

After matrix inverse $M$ computation and computation of $C_{noise}$, calculate the noise mitigation value as,

$$C_{mitigated} = M^{-1}.C_{noise} \tag{31}$$

To save space, $C_{mitigated}$ has been preserved in the form of a transposition as,

$$C_{mitigated} = \begin{bmatrix} 0 & 0 & 200 & 300 & 6600 & 2900 & \cdots & 0 & 0 & 0 \end{bmatrix}^T \tag{32}$$

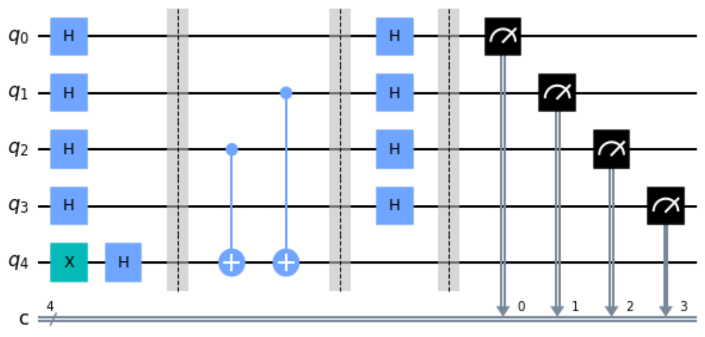
One more time, the Aer simulator runs 10,000 times for matrix calibration as

$$M = \begin{bmatrix} 0.961 & 0.0093 & 0.0094 & 0 & \cdots & 0 & 0 & 0 \\ 0.0106 & 0.9611 & 0.0003 & 0.0086 & \cdots & 0.0001 & 0 & 0 \\ 0.0085 & 0.0002 & 0.961 & 0.008 & \cdots & 0 & 0.0001 & 0 \\ 0.0001 & 0.0091 & 0.0094 & 0.9627 & \cdots & 0 & 0 & 0.0004 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0.0001 & 0 & 0 & \cdots & 0.9622 & 0 & 0.0096 \\ 0 & 0 & 0.0003 & 0 & \cdots & 0.0003 & 0.9598 & 0.0095 \\ 0 & 0 & 0 & 0.0003 & \cdots & 0.0098 & 0.0101 & 0.9592 \end{bmatrix} \tag{33}$$

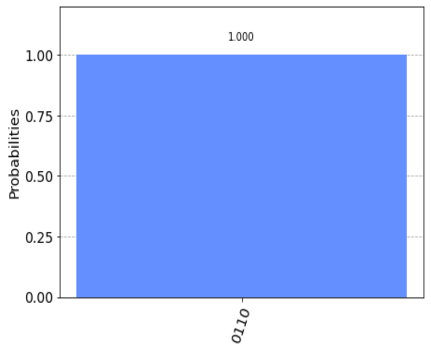
Quantum circuits are built using quantum image basis states. In addition, the simulator and real time quantum computer are used to run quantum image basis states 10,000 times. The simulator and real time computer generate statistical variance values on each run. The generated values are organized in a matrix. Matrix multiplication, inverse, and unitary actions were employed to determine image noise and error. When state shots increase in a quantum circuit, processing takes longer. The image basis specifies that distinct values are created without delay in small images. The noise is minimized as the shot value increases, but the computation time increases.

### *4.2. Result and Suggestions*

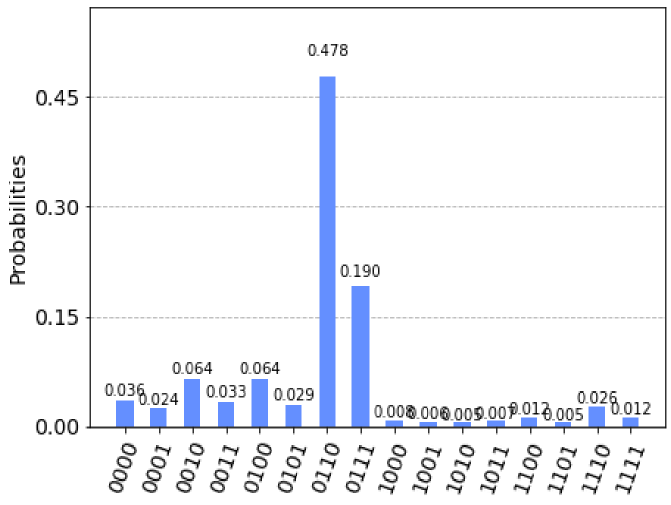
As indicated in Fig 2, image representation in quantum basis states (in classical called pixels). The $|0110\rangle$, $|1001\rangle$, $|1111\rangle$, and $|1001\rangle$ are qubit strings, a subset of a quantum image. From the $2^n \times 2^n$ image matrix (qubit strings set space), extract only the required qubit strings. Fig. 6 (a) indicates the quantum circuit of the $|0110\rangle$ qubit sequence where Hadamard $(H)$, Pauli-X $(X)$, CNOT, and measurement are used for basis states $|q_0 \ldots q_4\rangle$. The $|q_0 \ldots q_4\rangle$ are initialized by $|0\rangle$, according to the basis. The $|q_4\rangle$ acts as an ancilla qubit to control the outcome of the $|0110\rangle$. Fig. 6 (b) is the outcome as a histogram of the quantum circuit in Fig. 6 (a). Fig. 6 (b) shows the outcome of $|0110\rangle$ is 100% without any quantum computation noise. The rest of the qubit sequence mean $2^{2n}-1$ the generated outcome is 0%. However, due to superposition quantum noise, which exists on every $2^{2n}$ basis state, So, Fig. 6 (b) outcome prediction is incorrect at all. Among the available IBM simulators, we used an Aer simulator to generate the outcome of the basis state $|0110\rangle$. Fig. 6 (a) run 10,000 the quantum circuits in the Aer simulator and plot the histogram as shown in Fig. 6 (c). Each $2^{2n}$ basis state has some noise, as shown in Fig. 6(c). The $|0110\rangle$ histogram output value is greater than the other basis states (i.e., $|0110\rangle$ = 0.478). Other basis state output values that have an effect on the required qubit sequence $(|0110\rangle)$. For example, the $|0111\rangle$ qubit sequence value is greater than the rest of the undesired basis states. It mean, the IBM Aer simulator does not correctly generate the output of every basis state from $2^{2n}$. There is a need to mitigate the noise of undesired qubit sequence strings. So, we used the IBM real time quantum computer for the computation of the required qubit sequence and corresponding noise ratio. When the same qubit sequence $(|0111\rangle)$ is shot by the IBM real time quantum computer, then Fig. 7 (a) is the outcome generated by it. Real time quantum computers suppress the noise of undesired qubit sequence strings. When comparing Fig. 6(c) and Fig. 7(a), the Aer simulator noise is greater than that of a real time quantum computer.



(a)

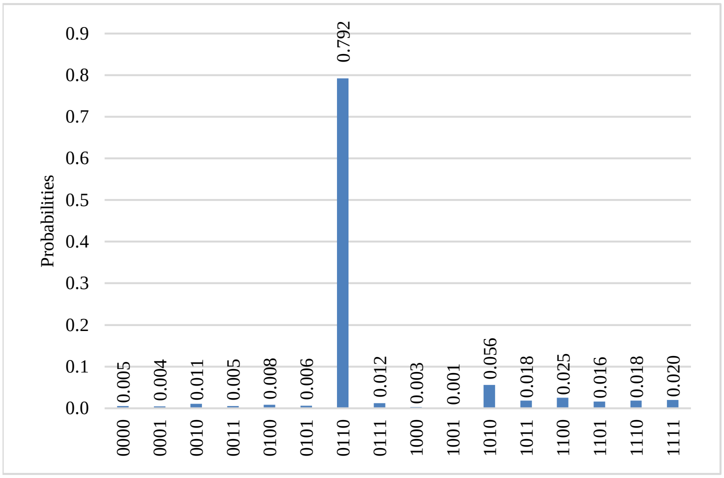


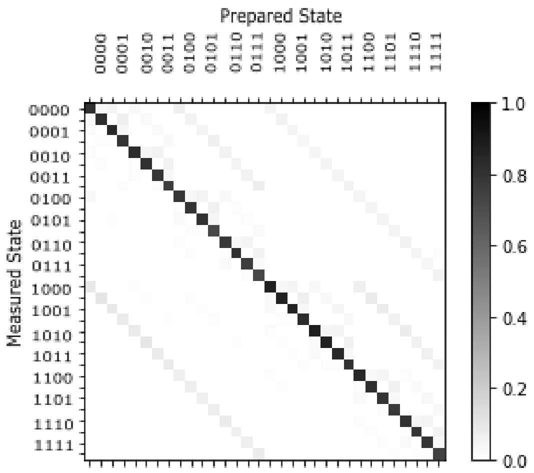
(b)



(c)

Fig.6. (a) A quantum circuit for qubit strings $|0110\rangle$ (b) Quantum circuit outcome after measurement (c) The IBM Aer simulator produced a quantum image of Fig. 2(c) after 10,000 runs





(a) (b)

Fig.7. Qubit states actual output probabilities. (a) The IBM real time quantum computer produces a quantum image of Fig. 2(c) after running it 10,000 times. (b) Figure 7 (a) shows the measured and prepared states

Table 1. Noise in the quantum image qubit basis states and their mitigated values

| Qubit States | Mitigated | Noise |
|---|---|---|
| 0110 | 0.06282 | 0.0626 |
| 1001 | 0.06036 | 0.0606 |
| 1111 | 0.06439 | 0.0642 |
| 1001 | 0.06036 | 0.0606 |

Table 1 takes the data from Fig. 2(c) only and plots the histogram of qubit states $|0110\rangle$, $|1001\rangle$, $|1111\rangle$, and $|1001\rangle$. The noise mitigation is greater than the noise probabilities in IBM's real time quantum computer. So, it is clear that IBM's real time quantum computer mitigates noise more than the IBM Aer simulator. If the noise mitigation is greater than the output image, the output image will be clearer. The image output by the IBM Aer simulator presents more noise, as shown in Fig. 9, than the IBM real time quantum computer, as shown in Fig. 10.

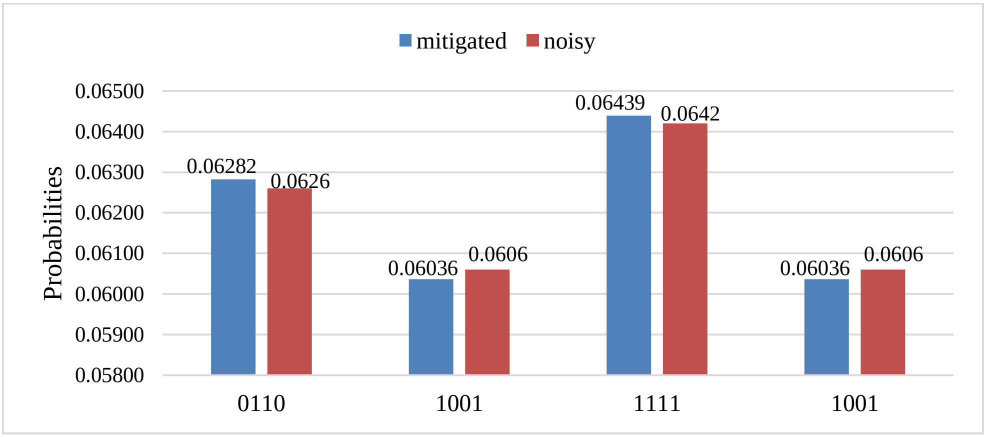


Fig.8. Mitigated and noisy probabilities in IBM real time quantum computer for $|0110\rangle$, $|1001\rangle$, $|1111\rangle$, and $|1001\rangle$ qubit sequence

As an input, we've used a classical image. An extractor extracts the locations and colors of pixels in a classical image. The color code determiner determines color, whereas the quantum image geometric function determines pixel position. Some noise and errors are generated as a result of the quantum image transformation of classical to quantum images. The IBM simulator reduces the amplitude of undesirable basis states while probabilistically amplifying the required amplitude. Qubit noise and error are decreased even more when repetitive processing (shots) is employed. After qubit measurement, as seen in Fig. 9, the image returns to its original form. The IBM real time computer, as shown in Fig. 10, employs the same approach.

A quantum image transform is performed in qubit basis states by the quantum component. Using basis states, create a quantum circuit. After computing the circuit basis state, measurements were made. They generate statistical variation amplitude for each basis state probabilistically by measurement. In simulator and real time quantum computer, the amplitude probabilities are modified. Image noise and errors can be detected using amplitude probabilities.

Qubits are known to be present during and prior to computation. The algorithm must be repeated until all noise and errors have been reduced to insignificant levels.

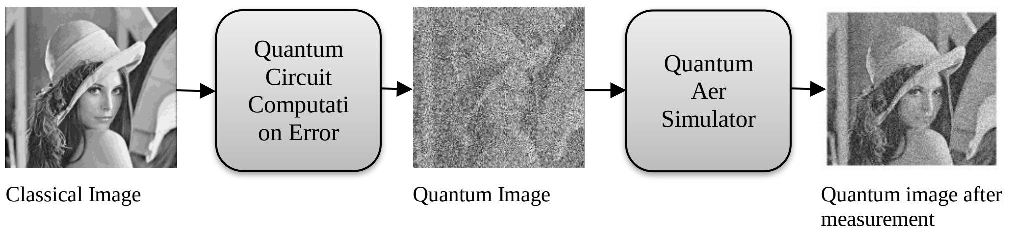

Quantum Circuit Computati on Error
Quantum Aer Simulator
Classical Image
Quantum Image
Quantum image after measurement

Fig. 9. Classical to quantum image transformation by IBM simulator Aer when shots are 10000

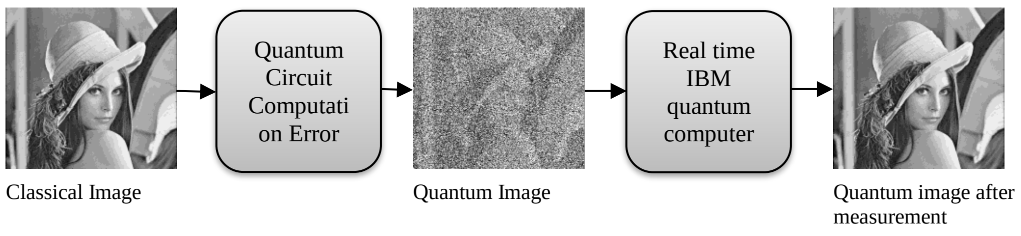

Quantum Circuit Computati on Error
Real time IBM quantum computer
Classical Image
Quantum Image
Quantum image after measurement

Fig.10. Classical to quantum image transformation by IBM real time quantum computer when shots are 10000

## 5. Proposed Algorithm Background

Quantum algorithms were developed using the IBM Qiskit and Anaconda Python platforms. Using an algorithm, we created a quantum circuit for four vector sets. After the circuit is designed, it is processed to determine the amount of noise and errors. For various operations, the generated probable values are matrices. In addition, the Aer simulator and IBM real time computer functions are included in a noise and quantum error detection algorithm.

## 6. Conclusion

Quantum image processing is a combination of quantum physics, computers, and mathematics. The superposition feature speeds up quantum computation a million times faster than classical computers. Complex image processing and the extraction of sub-images from complex images are quite time-consuming tasks for classical systems. Hence, there is a need to switch to another alternative. Due to superposition, it is an appropriate alternative for the processing of classical images within the desired timeframe. We have taken a small image for analysis through quantum mechanics. After analysis and processing by the IBM Aer simulator and a real time quantum computer, the image noise is more present in the IBM Aer than on the IBM real time quantum computer. As a result, there are numerous challenges that we must overcome in order to create a commercially viable product. After analysis and quantum image representation, superposition plays a significant role during computation. Quantum gates of either 1-qubit, 2-qubits, or more than 2-qubits, generate noise in an image. We need to control superposition qubits to mitigate quantum image noise.

Superposition of qubits and probabilistic behavior are essential for quantum image processing. Quantum image representation, operation, geometry, and color transformations are all based on quantum mechanics properties such as superposition. Superposition is well known for its efficiency in computing. This acceleration can be used to tackle challenges like analyzing complicated images and solving problems that have previously been unresolved. Because of the superposition, these problems can be solved in polynomial time. Quantum images generate noise and errors as a result of the superposition feature. For noise mitigation and error minimization, we presented a quantum method. Quantum error reduction and noise mitigation are the subjects of ongoing research. Low photon radiation intensity is of primary importance in the medical field.

Quantum image representation and its mathematical formulation are examined in this study. On the IBM Qiskit Anaconda Python platforms, quantum images have been used to design a circuit. The Aer simulator and IBM's real time quantum computer are used to run the vector. These simulators and real time quantum computers have produced statistical variation values for each run. The noise, on the other hand, decreases and mitigates the image error on every run. Between desired and undesired vectors, there isn't much of a difference in experience. Image noise is reduced as the number of shots on image vectors grows, but calculation time grows as well.

The decrease of noise and error in quantum color images can help researchers go even further. For complicated greyscale and color quantum images, IBM simulators and real time quantum computers may be used. The proposed approach can be used to eliminate quantum image errors and mitigate noise in complicated images.

## References

[1] https://qiskit.org/textbook/ch-quantum-hardware/measurement-error-mitigation.html (accessed on March 2022)

[2] https://qiskit.org/textbook/ch-applications/image-processing-frqi-neqr.html (accessed on March 2022)

[3] Shor, Peter W. "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer." *SIAM review* 41.2 (1999): 303-332.

[4] Le, Phuc Q., Fangyan Dong, and Kaoru Hirota. "A flexible representation of quantum images for polynomial preparation, image compression, and processing operations." *Quantum Information Processing* 10.1 (2011): 63-84.

[5] Li, Hai-Sheng, et al. "Multi-dimensional color image storage and retrieval for a normal arbitrary quantum superposition state." *Quantum Information Processing* 13.4 (2014): 991-1011.

[6] Li, Hai-Sheng, et al. "Image storage, retrieval, compression and segmentation in a quantum system." *Quantum information processing* 12.6 (2013): 2269-2290.

[7] Yuan, Suzhen, et al. "SQR: a simple quantum representation of infrared images." *Quantum Information Processing* 13.6 (2014): 1353-1379.

[8] Zhang, Yi, et al. "A novel quantum representation for log-polar images." *Quantum information processing* 12.9 (2013): 3103-3126.

[9] Zhang, Yi, et al. "NEQR: a novel enhanced quantum representation of digital images." *Quantum information processing* 12.8 (2013): 2833-2860.

[10] H.-S. Li, Z. Qingxin, S. Lan, C.-Y. Shen, R. Zhou, and J. Mo, ‘‘Image storage, retrieval, compression and segmentation in a quantum system,’’ (in English), Quantum Inf. Process., vol. 12, no. 6, pp. 2269–2290, Jun. 2013.

[11] S. Yuan, X. Mao, Y. Xue, L. Chen, Q. Xiong, and A. Compare, ‘‘SQR: A simple quantum representation of infrared images,’’ (in English), Quantum Inf. Process., vol. 13, no. 6, pp. 1353–1379, Jun. 2014.

[12] S. Caraiman and V. Manta, ‘‘Image representation and processing using ternary quantum computing,’’ in Adaptive and Natural Computing Algorithms, vol. 7824. Springer, Apr. 2013, pp. 366–375. [Online]. Available: https://link.springer.com/chapter/10.1007%2F978-3-642-37213-1_38

[13] B. Sun, A. M. Iliyasu, F. Yan, F. Dong, and K. Hirota, ‘‘An RGB multichannel representation for images on quantum computers,’’ J. Adv. Comput. Intell. Intell. Inform., vol. 17, no. 3, pp. 404–417, 2013.

[14] N. Jiang and L. Wang, ‘‘Quantum image scaling using nearest neighbor interpolation,’’ (in English), Quantum Inf. Process., vol. 14, no. 5, pp. 1559–1571, May 2015.

[15] H.-S. Li, P. Fan, H.-Y. Xia, H.-L. Peng, and S. X. Song, ‘‘Quantum implementation circuits of quantum signal representation and type conversion,’’ IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 66, no. 1, pp. 341–354, Jan. 2019.

[16] J. Sang, S. Wang, and Q. Li, ‘‘A novel quantum representation of color digital images,’’ (in English), Quantum Inf. Process., vol. 16, no. 2, p. 14, Feb. 2017.

[17] H.-S. Li, X. Chen, H. Xia, Y. Liang, and Z. Zhou, ‘‘A quantum image representation based on bitplanes,’’ (in English), IEEE Access, vol. 6, pp. 62396–62404, 2018.

[18] N. Zhou, X. Yan, H. Liang, X. Tao, and G. Li, ‘‘Multi-image encryption scheme based on quantum 3D arnold transform and scaled zhongtang chaotic system,’’ (in English), Quantum Inf. Process., vol. 17, no. 12, p. 36, Dec. 2018.

[19] E. Sahin and I. Yilmaz, ‘‘QRMW: Quantum representation of multi wavelength images,’’ (in English), Turkish J. Electr. Eng. Comput. Sci., vol. 26, no. 2, pp. 768–779, Mar. 2018.

[20] M. Abdolmaleky, M. Naseri, J. Batle, A. Farouk, and L.-H. Gong, ‘‘Redgreen-blue multi-channel quantum representation of digital images,’’ (in English), Optik, vol. 128, pp. 121–132, Jan. 2017

[21] B. Wang, M.-Q. Hao, P.-C. Li, and Z.-B. Liu, ‘‘Quantum representation of indexed images and its applications,’’ (in English), Int. J. Theor. Phys., vol. 59, no. 2, pp. 374–402, Feb. 2020.

[22] L. Wang, Q. Ran, and J. Ma, ‘‘Double quantum color images encryption scheme based on DQRCI,’’ Multimedia Tools Appl., vol. 79, nos. 9–10, pp. 6661–6687, Mar. 2020.

[23] G. Xu, X. Xu, X. Wang, and X. Wang, ‘‘Order-encoded quantum image model and parallel histogram specification,’’ (in English), Quantum Inf. Process., vol. 18, no. 11, p. 346, Nov. 2019.

[24] X. Liu, D. Xiao, W. Huang, and C. Liu, ‘‘Quantum block image encryption based on arnold transform and sine chaotification model,’’ IEEE Access, vol. 7, pp. 57188–57199, 2019.

[25] R. A. Khan, ‘‘An improved flexible representation of quantum images,’’ Quantum Inf. Process., vol. 18, no. 7, p. 201, Jul. 2019.

[26] P. C. Li and X. D. Liu, ‘‘Color image representation model and its application based on an improved FRQI,’’ (in English), Int. J. Quantum Inf., vol. 16, no. 1, p. 25, Feb. 2018

[27] L. Wang, Q. Ran, J. Ma, S. Yu, and L. Tan, ‘‘QRCI: A new quantum representation model of color digital images,’’ (in English), Opt. Commun., vol. 438, pp. 147–158, May 2019.

[28] Shah, Younis A., Irshad A. Mir, and Uzair M. Rathe. "Quantum Mechanics Analysis: Modeling and Simulation of some simple systems." *Int. J. Math. Sci. Comput.(IJMSC)* 2.1 (2016): 23-40.

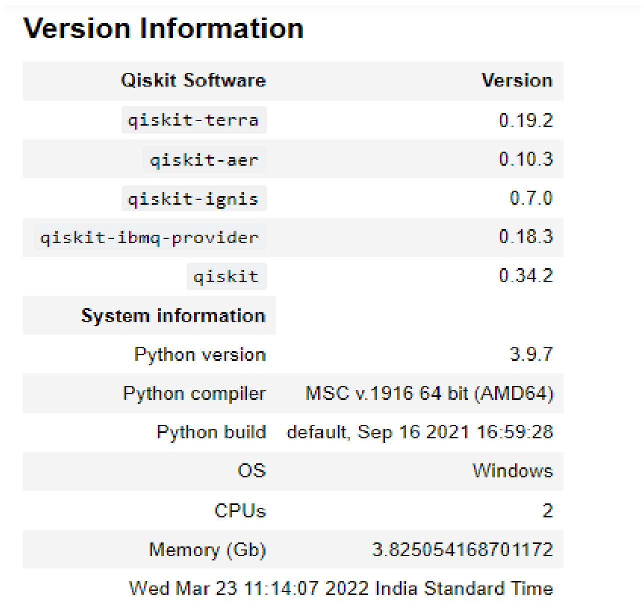
[29] Sihare, Shyam, and V. Nath. "Revisited Quantum Protocols." *International Journal of Mathematical Sciences and Computing (IJMSC)* 3.2 (2017): 11-21.

[30] Sihare, Shyam R., and V. V. Nath. "Application of quantum search algorithms as a web search engine." *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*. IEEE, 2016.

[31] Sihare, Shyam R., and V. V. Nath. "Analysis of quantum algorithms with classical systems counterpart." *International Journal of Information Engineering and Electronic Business* 9.2 (2017): 20.

## Appendix-A Quantum circuit design and image processing algorithm

```
from qiskit import *
%matplotlib inline
from qiskit import QuantumCircuit, QuantumRegister, Aer, transpile, assemble
from qiskit.visualization import array_to_latex
from qiskit.tools.visualization import plot_histogram
from qiskit.providers.aer.noise import NoiseModel
from qiskit.providers.aer.noise.errors import pauli_error, depolarizing_error
def get_noise(p):
    error_meas = pauli_error([('X',p), ('I', 1 - p)])
    noise_model = NoiseModel()
    noise_model.add_all_qubit_quantum_error(error_meas, "measure") # measurement error is applied to measurements
    return noise_model

noise_model = get_noise(0.01)
aer_sim = Aer.get_backend('aer_simulator')
circuit=QuantumCircuit(4+4,4)
circuit.h([0,1,2,3])
circuit.x([4,5,6,7])
circuit.h([4,5,6,7])
circuit.barrier()
i=0
j=4
for state in ['0110','1001','1111','1001']:
    circuit = QuantumCircuit(4+4,4)
    if state[0]=='1':
        circuit.cx(i+3,j)
    if state[1]=='1':
        circuit.cx(i+2,j)
    if state[2]=='1':
        circuit.cx(i+1,j)
    if state[3]=='1':
        circuit.cx(i,j)
    j=j+1
    circuit.measure([0,1,2,3],[0,1,2,3])
    t_qc = transpile(circuit, aer_sim)
    qobj = assemble(t_qc)
    counts = aer_sim.run(qobj, noise_model=noise_model, shots=10000).result().get_counts()
    print(state+' becomes', counts)
import numpy as np

M =
[[0.06240,0.06270,0.06420,0.06000,0.06230,0.06250,0.06620,0.06380,0.06350,0.06060,0.06040,0.06440,0.06250,0.06150,0.06040,0.06260],
[0.06380,0.06620,0.06100,0.06100,0.06170,0.06040,0.06080,0.06030,0.06620,0.06120,0.06060,0.06130,0.06530,0.06270,0.06060,0.06690],
[0.06020,0.06300,0.06510,0.06120,0.06230,0.06320,0.06400,0.06230,0.06380,0.06180,0.06060,0.06040,0.06110,0.06260,0.06570,0.06270],
[0.06210,0.06510,0.06060,0.06510,0.06090,0.06300,0.06100,0.06250,0.06590,0.06580,0.06020,0.06050,0.06160,0.06220,0.06340,0.06010],
[0.06110,0.06300,0.06100,0.06120,0.06170,0.06110,0.06080,0.06120,0.06430,0.06610,0.06010,0.06660,0.06400,0.06220,0.06340,0.06220],
[0.06070,0.06540,0.06250,0.06240,0.06320,0.06010,0.06080,0.06540,0.06590,0.06310,0.06030,0.06130,0.06020,0.06360,0.06030,0.06480],
[0.06290,0.06400,0.06320,0.06300,0.06120,0.06580,0.06120,0.06140,0.06580,0.06450,0.06050,0.06030,0.06100,0.06360,0.06120,0.06040],
[0.06230,0.06290,0.06290,0.06120,0.06460,0.06060,0.06480,0.06060,0.06520,0.06270,0.06040,0.06170,0.06280,0.06500,0.06080,0.06150],
[0.06070,0.06350,0.06410,0.06300,0.06480,0.06370,0.06460,0.06290,0.06200,0.06010,0.06090,0.06040,0.06460,0.06020,0.06100,0.06350],
[0.06630,0.06280,0.06080,0.06100,0.06240,0.06210,0.06500,0.06180,0.06140,0.06060,0.06690,0.06120,0.06030,0.06010,0.06120,0.06610],
[0.06650,0.06490,0.06300,0.06190,0.06100,0.06040,0.06180,0.06180,0.06140,0.06070,0.06530,0.06170,0.06400,0.06230,0.06150,0.06180],
[0.06460,0.06040,0.06030,0.06180,0.06530,0.06070,0.06430,0.06200,0.06100,0.06400,0.06090,0.06170,0.06620,0.06010,0.06090,0.06580],
[0.06180,0.06060,0.06060,0.06570,0.06430,0.06500,0.06450,0.06100,0.06150,0.06340,0.06300,0.06280,0.06020,0.06120,0.06420,0.06020],
[0.06380,0.06050,0.06380,0.06620,0.06350,0.06050,0.06020,0.06610,0.06180,0.06130,0.06090,0.06250,0.06180,0.06200,0.06420,0.06090],

[0.06390,0.06480,0.06120,0.06130,0.06220,0.06230,0.06300,0.06200,0.06210,0.06250,0.06220,0.06410,0.06110,0.06030,0.06310,0.06390],
[0.06540,0.06870,0.06020,0.06140,0.06120,0.06230,0.06120,0.06540,0.06120,0.06230,0.06010,0.06020,0.06410,0.06170,0.06230,0.06240]]

Cideal = [[0],
        [0],
        [0],
        [0],
        [0],
        [0],
        [3216],
        [0],
        [0],
        [4216],
        [0],
        [0],
        [0],
        [0],
          [0],
         [2568]]
Cnoisy = np.dot(M, Cideal)
array_to_latex(Cnoisy, prefix="\\text{C}_\\text{noisy} = ")
import scipy.linalg as la

M =
[[0.06240,0.06270,0.06420,0.06000,0.06230,0.06250,0.06620,0.06380,0.06350,0.06060,0.06040,0.06440,0.06250,0.06150,0.06040,0.06260],
[0.06380,0.06620,0.06100,0.06100,0.06170,0.06040,0.06080,0.06030,0.06620,0.06120,0.06060,0.06130,0.06530,0.06270,0.06060,0.06690],
[0.06020,0.06300,0.06510,0.06120,0.06230,0.06320,0.06400,0.06230,0.06380,0.06180,0.06060,0.06040,0.06110,0.06260,0.06570,0.06270],
[0.06210,0.06510,0.06060,0.06510,0.06090,0.06300,0.06100,0.06250,0.06590,0.06580,0.06020,0.06050,0.06160,0.06220,0.06340,0.06010],
[0.06110,0.06300,0.06100,0.06120,0.06170,0.06110,0.06080,0.06120,0.06430,0.06610,0.06010,0.06660,0.06400,0.06220,0.06340,0.06220],
[0.06070,0.06540,0.06250,0.06240,0.06320,0.06010,0.06080,0.06540,0.06590,0.06310,0.06030,0.06130,0.06020,0.06360,0.06030,0.06480],
[0.06290,0.06400,0.06320,0.06300,0.06120,0.06580,0.06120,0.06140,0.06580,0.06450,0.06050,0.06030,0.06100,0.06360,0.06120,0.06040],
[0.06230,0.06290,0.06290,0.06120,0.06460,0.06060,0.06480,0.06060,0.06520,0.06270,0.06040,0.06170,0.06280,0.06500,0.06080,0.06150],
[0.06070,0.06350,0.06410,0.06300,0.06480,0.06370,0.06460,0.06290,0.06200,0.06010,0.06090,0.06040,0.06460,0.06020,0.06100,0.06350],
[0.06630,0.06280,0.06080,0.06100,0.06240,0.06210,0.06500,0.06180,0.06140,0.06060,0.06690,0.06120,0.06030,0.06010,0.06120,0.06610],
[0.06650,0.06490,0.06300,0.06190,0.06100,0.06040,0.06180,0.06180,0.06140,0.06070,0.06530,0.06170,0.06400,0.06230,0.06150,0.06180],
[0.06460,0.06040,0.06030,0.06180,0.06530,0.06070,0.06430,0.06200,0.06100,0.06400,0.06090,0.06170,0.06620,0.06010,0.06090,0.06580],
[0.06180,0.06060,0.06060,0.06570,0.06430,0.06500,0.06450,0.06100,0.06150,0.06340,0.06300,0.06280,0.06020,0.06120,0.06420,0.06020],
[0.06380,0.06050,0.06380,0.06620,0.06350,0.06050,0.06020,0.06610,0.06180,0.06130,0.06090,0.06250,0.06180,0.06200,0.06420,0.06090],
[0.06390,0.06480,0.06120,0.06130,0.06220,0.06230,0.06300,0.06200,0.06210,0.06250,0.06220,0.06410,0.06110,0.06030,0.06310,0.06390],
[0.06540,0.06870,0.06020,0.06140,0.06120,0.06230,0.06120,0.06540,0.06120,0.06230,0.06010,0.06020,0.06410,0.06170,0.06230,0.06240]]

Minv = la.inv(M)
array_to_latex(Minv)
Cmitigated = np.dot(Minv, Cnoisy)
array_to_latex(Cmitigated, prefix="\\text{C}_\\text{mitigated}=")
Cmitigated = np.dot(Minv, Cnoisy)
array_to_latex(Cmitigated, prefix="\\text{C}_\\text{mitigated}=")
t_qc = transpile(meas_calibs, aer_sim)
qobj = assemble(t_qc, shots=10000)
cal_results = aer_sim.run(qobj, noise_model=noise_model, shots=10000).result()
meas_fitter = CompleteMeasFitter(cal_results, state_labels, circlabel='mcal')
array_to_latex(meas_fitter.cal_matrix)
noise_model = get_noise(0.1)
  t_qc = transpile(meas_calibs, aer_sim)
  qobj = assemble(t_qc, shots=10000)
  cal_results = aer_sim.run(qobj, noise_model=noise_model, shots=10000).result()
  meas_fitter = CompleteMeasFitter(cal_results, state_labels, circlabel='mcal')
  array_to_latex(meas_fitter.cal_matrix)
  circuit=QuantumCircuit(4+4,4)
  circuit.h([0,1,2,3])
  circuit.x([4,5,6,7])
  circuit.h([4,5,6,7])
  circuit.cx(2,4)
  circuit.cx(1,4)
  circuit.cx(3,5)
  circuit.cx(0,5)
  circuit.cx(3,6)
  circuit.cx(2,6)
  circuit.cx(1,6)
  circuit.cx(0,6)
  circuit.cx(3,7)
  circuit.cx(0,7)
  circuit.measure([0,1,2,3],[0,1,2,3])

  t_qc = transpile(circuit, aer_sim)
  qobj = assemble(t_qc, shots=10000)
  results = aer_sim.run(qobj, noise_model=noise_model, shots=10000).result()
  noisy_counts = results.get_counts()
  print(noisy_counts)
  # Get the filter object
  meas_filter = meas_fitter.filter

  # Results with mitigation
  mitigated_results = meas_filter.apply(results)
  mitigated_counts = mitigated_results.get_counts()
  from qiskit.visualization import plot_histogram
  noisy_counts = results.get_counts()
  #print(noisy_counts)
  print(mitigated_counts)
  plot_histogram([noisy_counts, mitigated_counts], legend=['noisy', 'mitigated'])
```



## Version Information

| Qiskit Software | Version |
|---|---|
| qiskit-terra | 0.19.2 |
| qiskit-aer | 0.10.3 |
| qiskit-ignis | 0.7.0 |
| qiskit-ibmq-provider | 0.18.3 |
| qiskit | 0.34.2 |
| **System information** | |
| Python version | 3.9.7 |
| Python compiler | MSC v.1916 64 bit (AMD64) |
| Python build | default, Sep 16 2021 16:59:28 |
| OS | Windows |
| CPUs | 2 |
| Memory (Gb) | 3.825054168701172 |
| | Wed Mar 23 11:14:07 2022 India Standard Time |

## Author's Profile

**Shyam R. Sihare** has completed Ph. D. candidate in Raksha Shakti University, Ahmedabad, India. He took up his Master's degree in Computer Science at Nagpur University, Nagpur, India in 2003 and obtained M. Phil. in Computer Science at Madurai Kamraj University, Madurai, India. He cleared Professor Eligibility Test GSLET (Gujarat) and MS-SET (Maharashtra), India in 2011 and 2018 respectively. He obtained MCA at IGNOU, New Delhi, India in 2011. He is currently working as Asstt. Professor in Computer Science and Application in Dr. APJ Abdul Kalam Govt. College, Silvassa, Dadra & Nagar Haveli(UT), India. His research interests include Quantum Computer, Quantum Algorithms, Quantum Cryptography, and Classical Computer Algorithms.

**How to cite this paper:** Shyam R. Sihare, "Transformation of Classical to Quantum Image, Representation, Processing and Noise Mitigation", International Journal of Image, Graphics and Signal Processing(IJIGSP), Vol.14, No.5, pp. 10-27, 2022. DOI:10.5815/ijigsp.2022.05.02