ANNALI DELL'UNIVERSITA' DI FERRARA (2022) 68:597–620
https://doi.org/10.1007/s11565-022-00445-2

Quantum median filter for total variation image denoising

Simone De Santis[1,2] · Damiana Lazzaro[1] · Riccardo Mengoni[2] · Serena Morigi[1]

Received: 6 May 2022 / Accepted: 5 September 2022 / Published online: 30 September 2022

Abstract
In this new computing paradigm, named quantum computing, researchers from all over the world are taking their first steps in designing quantum circuits for image processing, through a difficult process of knowledge transfer. This effort is named quantum image processing, an emerging research field pushed by powerful parallel computing capabilities of quantum computers. This work goes in this direction and proposes the challenging development of a powerful method of image denoising, such as the total variation (TV) model, in a quantum environment. The proposed quantum TV is described and its sub-components are analysed. Despite the natural limitations of the current capabilities of quantum devices, the experimental results show a competitive denoising performance compared to the classical variational TV counterpart.

Keywords Quantum image processing · Total variation model · Quantum median filter · Image denoising

Mathematics Subject Classification 65Y10 · 68Q12 · 65K10 · 92C55

✉ Serena Morigi
serena.morigi@unibo.it

Simone De Santis
simone.desantis@studio.unibo.it

Damiana Lazzaro
damiana.lazzaro@unibo.it

Riccardo Mengoni
r.mengoni@cineca.it

1 Department of Mathematics, University of Bologna, Piazza Porta San Donato, 40126 Bologna, Italy

2 CINECA, Organization, Via Magnanelli 6/3, Casalecchio di Reno, 40033 Bologna, Italy

1 Introduction

Digital image processing is an extremely, computationally demanding, strategic research field. This has led, over the years, to the development of many complex image processing algorithms on highly parallel, specialized hardware platforms. With the rapid progress of parallel hardware, suitably high performance is now available also for sophisticated imaging tasks. In the present study we focus on image denoising, an essential requirement for any other image processing, which refers to the recovery of a clean sharp image from a noisy observation.

In [20] Rudin, Osher and Fatemi presented one of the main mathematical models and algorithms for image denoising, the so-called Total Variation denoising (also known as *TV model*). Since then, TV model has become a quite popular technique, which usage allows to improve overall image quality when the images are affected by noise or corruption, while well preserving edges and details. Efficient solutions have been proposed over time for the numerical optimizations of the TV model [3, 4, 11, 25], but they are not able to fully exploit parallel computational resources. In [13], the authors designed a new optimization algorithm which is simple and highly parallelizable, and relies on median value computations, thus reducing computational effort to a sorting problem. Thanks to its low complexity, this algorithm is prone to be implemented on low-end devices or, more generally, in situations where a reduced amount of resources is available. That is the case of Quantum Image Processing (QIP), a novel and promising research field which goal is the development of image processing techniques for quantum computers, exploiting peculiar features from quantum world, like entanglement, superposition, interference [24]. Quantum computing is universally acknowledged for its ability to process data providing computational speedups compared to the classical paradigm. However, image processing is actually one of the most demanding applications in terms of resources for quantum computers. As a consequence QIP is still in its early stage, and thus is facing several fundamental problems, such as how to represent and store an image in quantum computers appropriately, and how to efficiently implement image processing algorithms [10, 16].

In the present days, quantum devices are subject to several issues (i.e. noise, absence of error correction, low amount of available qubits, etc.), which defines what researchers commonly refer to as *Noise intermediate-scale quantum* era, or just NISQ [18]. This is an intermediate development phase where, through the exhibition of quantum devices' limits, algorithms have been designed to be as optimized and simple as possible [2]. In this context, QIP is a complex subject to deal with because several difficulties may arise in the attempt of implementing the classical image processing algorithms on quantum devices (some of them will be discussed later in this work). Many proposals have been submitted during the past few years [22], in the attempt to provide QIP algorithms that fulfill NISQ requirements. QIP algorithms are usually tested using quantum simulators or similar execution environments.

Here along this line, the presented work aims to solve the TV model in a quantum environment. The result is a Quantum TV filter, which integrates a Quantum Median Filter proposal by Li et al. [10, 14]. This work focuses not only on the development of a QIP technique for image denoising which mimics its variational TV counterpart, but also on presenting an useful review of detected problems in the newly developed QIP field, and offering possible solutions and ideas for future developments and optimizations.

The paper is organized as follows. In Sect. 2 we introduce basic theory about Total Variation denoising technique and a median formula for efficiently solving an anisotropic TV problem. In Sect. 3 we discuss Quantum Image Processing concepts and related issues. In Sect. 4 we explain in detail how Quantum TV Filter works, highlighting its quantum components, and analysing its circuit complexity in Sect. 5. In Sect. 6 we focus on the analysis of experimental results, comparing both quantum and classical implementations of the TV algorithm. Section 7 reports conclusions and future works. In the Appendix we briefly report details on a few quantum modules used in the design of the proposed Quantum TV.

For basic notations and fundamental knowledge about quantum image processing computing, we refer the reader to [24].

2 Total variation image denoising

The goal of denoising is to obtain an image u^* not only with small variations in intensity between pixels but also close to the observation f. At this aim, the class of *variational* methods for image restoration relies on determining restored images $u^* \in \mathbb{R}^N$, given a noisy image $f \in \mathbb{R}^N$, as the minimizers of suitable cost functionals $J : \mathbb{R}^N \to \mathbb{R}$ such that, typically, restoration is casted as an optimization problem of the form:

$$u^* \;\leftarrow\; \arg\min_{u\in\mathbb{R}^N} \{\, J(u) := R(u) + \lambda\, F(u; f) \,\}\;, \tag{1}$$

where the functionals $R(u)$ and $F(u; f)$, commonly referred to as the *regularization* and the *fidelity* term, encode prior information on the clean image u and the observation model, respectively, with the so-called regularization parameter $\lambda > 0$ controlling the trade-off between the two terms. In particular, the functional form of the fidelity term is strictly connected to the characteristics of the noise corruption. A classical choice for the fidelity measures the data fitting in terms of the ℓ_2-norm, in formulas:

$$F(u; f) \;:=\; \|u - f\|_2^2. \tag{2}$$

For what regards the regularization term $R(u)$ in (1), a very popular choice is represented by the Total Variation, presented in the following two forms:

$$\text{[Isotropic TV]}\;\; R_{\text{iso}}(u) := \|Du\|_{2,1} = \|\sqrt{(D_X u)^2 + (D_Y u)^2}\|_1 \tag{3}$$

$$\text{[Anisotropic TV]}\;\; R_{\text{ani}}(u) := \|Du\|_1 = \|D_X u\|_1 + \|D_Y u\|_1 \tag{4}$$

where D_x, D_y denote the horizontal and vertical operators, respectively, and $D = (D_x, D_y)$ is the gradient operator ∇ in the discrete setting.

By substituting the TV regularizer $TV(u) := R_{iso}(u)$ in (3) or $TV(u) := R_{ani}(u)$ in (4) and the fidelity term (2) for R and F in (1), respectively, one obtains the so-called

TV-L$_2$ - restoration model, originally introduced in [20]. In formulas:

$$u^* \leftarrow \arg\min_{u \in \mathbb{R}^N} \left\{ J(u) = \mathrm{TV}(u) + \lambda \, \|u - f\|_2^2 \right\} . \tag{5}$$

2.1 Median formula for TV

Li and Osher in [13] proposed an efficient and highly parallelizable method for solving TV model (5) with anisotropic TV regularization (4). They proposed to solve iteratively N one-dimensional optimization problems to obtain an accurate solution of the N-dimensional optimization problem (5). In particular, for each pixel $u \in \mathbb{R}$, we consider the local minimization problem:

$$u^* = \arg\min_{u \in \mathbb{R}} \left\{ E(u) := \sum_{i=1}^{\kappa} w_i |u - u_i| + F(u) \right\} \tag{6}$$

where $F(u) = \lambda(f - u)^2$, u^*, $f \in \mathbb{R}$ are respectively the denoised and noisy versions of the same pixel, while u_i belongs to the set of κ neighboring pixels and $w_i \geq 0$ are given weights. In [13], a simple method for computing u^* in (6) is derived and here reported for self-consistency.

Theorem 1 *Supposing the w_i are non-negative and the u_i are sorted as $u_1 \leq u_2 \leq \ldots \leq u_\kappa$, the function F is strictly convex and differentiable and F' is bijective; then the minimizer of* (6) *is a median:*

$$u^* = \mathrm{median}\{u_1, u_2, \ldots, u_\kappa, p_0, p_1, \ldots, p_\kappa\} \tag{7}$$

where $p_i = (F')^{-1}(W_i)$ and

$$W_i = -\sum_{j=1}^{i} w_j + \sum_{j=i+1}^{\kappa} w_j, \quad i = 0, 1, \ldots, \kappa. \tag{8}$$

In our formulation, the neighborhood of the current pixel u are simply u_u, u_d, u_l, u_r, the vertical and horizontal direct neighbors pixels, respectively. The adopted 4-neighbors strategy allows us to apply the median formula in parallel on multiple pixels at one time using a proper configuration, since each pixel is directly affected only by its 4 neighbors.

The fidelity is defined as $F(u) = \lambda(f - u)^2$, and consequently

$$F'(u) = -2\lambda(f - u) \Rightarrow (F')^{-1}(W) = f + \frac{W}{2\lambda},$$

where W is a sum of weights previously defined in (8). The denoised pixel u^* is then obtained as the median value:

$$u^* = \mathrm{median}\{u_l, u_r, u_u, u_d, p_0, p_1, p_2, p_3, p_4\}, \tag{9}$$

where the p-values p_i are calculated following Theorem 1, with $w_i = 1$ and $\kappa = 4$:

$$\begin{array}{ll} W_0 = 4 \Rightarrow p_0 = f + \frac{4}{2\lambda} \Rightarrow & p_0 = f + \frac{2}{\lambda} \\ W_1 = 2 \Rightarrow p_1 = f + \frac{2}{2\lambda} \Rightarrow & p_1 = f + \frac{1}{\lambda} \\ W_2 = 0 \Rightarrow p_2 = f + \frac{0}{2\lambda} \Rightarrow & p_2 = f \\ W_3 = -2 \Rightarrow p_3 = f + \frac{-2}{2\lambda} \Rightarrow & p_3 = f - \frac{1}{\lambda} \\ W_4 = -4 \Rightarrow p_4 = f + \frac{-4}{2\lambda} \Rightarrow & p_4 = f - \frac{2}{\lambda} \end{array} \tag{10}$$

The image denoising problem can hence be solved by iteratively computing (7) pixel-by-pixel over the whole image until convergence, which is guaranteed by the following result.

Theorem 2 *The algorithm defined by repeatedly applying* (7) *at the* j*th pixel, converges, i.e.* $u_j^{(k+1)} = \arg\min_{u_j \in \mathbb{R}} E^{(k)}(u_j)$, *hence*

$$u^{(k)} \Rightarrow \arg\min_{u \in \mathbb{R}^N} J(u).$$

This means that, after k iterations, with $k \to \infty$, we obtain the minimizer of problem (5).

For the numerical implementation the stopping criterion considered is computed as follows: given a small tolerance value ϵ, process stops when, at iteration k

$$\frac{\|u^{(k-1)} - u^{(k)}\|_2}{\|u^{(k-1)}\|_2} \leq \epsilon \tag{11}$$

where $u^{(k-1)}$ and $u^{(k)}$ are consecutive processed images.

The resulting algorithm is described in Algorithm 1.

Algorithm 1 Anisotropic TV algorithm

```
Input: $f \in \mathbb{R}^N$, $\lambda > 0$
Output: $u^* \in \mathbb{R}^N$
Initialize image$^{(0)} = f, k = 0$
while(!convergence)
    For each pixel in image$^{(k)}$ do:
      compute $W \in \mathbb{R}^5$ as in (10)
      set $N_{pixel}$ = get_neighbors(pixel)    $\%N_{pixel} = (u_l, u_r, u_d, u_u)$
      $v \leftarrow$ sort($N_{pixel}$) in ascending order
      compute $p \in \mathbb{R}^5$, with $p_j$ = pixel $+ \frac{1}{\lambda} W_j$,    $j = 0, .., 4$
      pixel* $\leftarrow$ median($v_0, v_1, v_2, v_3, p_0, p_1, p_2, p_3, p_4$)
      tmp_image $\leftarrow$ pixel*
    end
image$^{(k+1)}$ =tmp_image
$k = k + 1$
end
$u^*$ = image$^{(k+1)}$
```

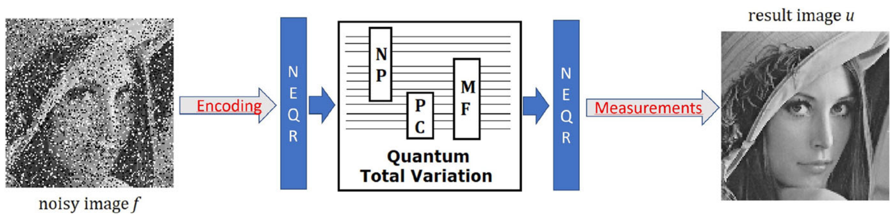


Fig. 1 Scheme illustrating Quantum TV (QTV) denoising process

3 Quantum image processing

QIP aims to design quantum algorithms that, once constructed a quantum state which encodes an image, implement image processing techniques in a quantum environment. QIP is a novel topic in quantum computing and many issues are far from being solved, as explained in [19]. A central issue regards the image encoding in a quantum environment, which will be discussed in Sect. 3.1. Another limiting issue is that, nowadays, QIP represents a demanding branch of quantum computing, as it needs a lot of resources that are far from being offered by current or medium term devices. At the moment, researchers are proposing many different approaches to the problem, even though only a limited number of these are commonly accepted and used [22–24]. In order to tackle the memory restrictions, we subdivide the image into a set of sub-images as described in Sect. 3.2.

The denoising QIP here proposed consists of three steps, illustrated in Fig. 1. The image is first encoded in the quantum environment, then it is processed by quantum circuits to perform the denoising task, and finally the denoised image is measured to convert it in a classical image format. The used measurement process is detailed in Sect. 3.3.

3.1 Quantum image representation

A quantum image encoding is defined Quantum Image Representation (QIR). Unlike classical image processing, where a set of well-known standard formats are available and well-assessed, in QIP many encoding QIR techniques were proposed and tested [14], but on the other hand none nowadays has distinguished itself as standard. The most used QIR technique is the Novel Enhanced Quantum Representation (NEQR).

This kind of representation needs to encode the following image's data:

- **Pixel coordinates**, encoded by qubits $|XY\rangle$. An image of dimension $D_X \times D_Y$, usually needs to use $d_X = \lceil \log_2 D_X \rceil$ qubits for horizontal coordinates and $d_Y = \lceil \log_2 D_Y \rceil$ qubits for vertical ones. In this way $|XY\rangle = |X_0 X_1 ... X_{d_x - 1} Y_0 Y_1 ... Y_{d_y - 1}\rangle$.
- **Pixel value**, encoded by one or more qubits $|C\rangle$. It uses q qubits for representing $N_q = 2^q$ possible values in a binary encoding.

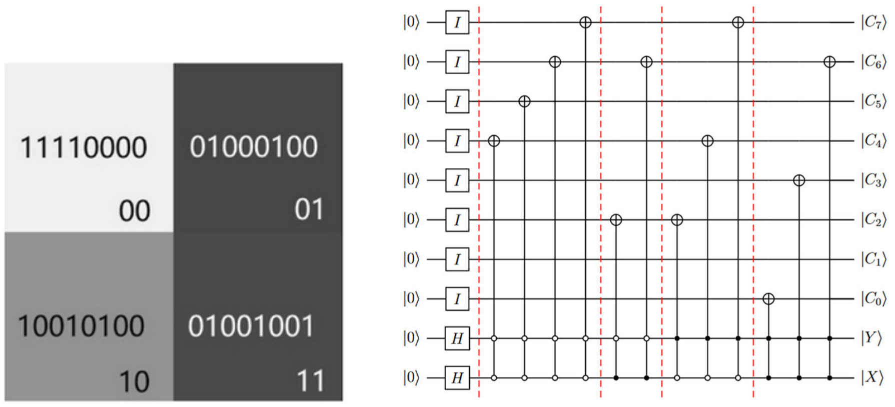


Fig. 2 NEQR representation of an image: (left) the 2×2 image; (right) NEQR circuit, encoding pixels 00, 01, 10, 11 (left to right). Figure on the left from [24]

Without loss of generality, we consider grayscale images I with a square $2^n \times 2^n$ domain and values in the range $[0, 2^q - 1]$. In this way we consider $|XY\rangle$ with $2n$ qubits, where $n = \lceil \log_2 D \rceil$, and $|C\rangle$ using $q = \lceil \log_2 N_q \rceil$ qubits to represent 2^q
state of $2n + q$ qubits:

$$|I(n)\rangle = \frac{1}{2^n} \sum_{X=0}^{2^n-1} \sum_{Y=0}^{2^n-1} |C_{XY}\rangle \otimes |XY\rangle, \tag{12}$$

where

$$|C_{XY}\rangle = |C_{XY}^{q-1} \dots C_{XY}^{2} C_{XY}^{1} C_{XY}^{0}\rangle \in \{0, 1, \dots, 2^q - 1\} \tag{13}$$

and each $|C_{XY}^i\rangle$ qubit is $|0\rangle$ or $|1\rangle$. A simple example of NEQR representation for a 2×2 grayscale image is illustrated in Fig. 2(left), where each corner number denotes pixel's coordinates, while centered value indicates pixel's intensity [24]. The quantum wave function for this image in NEQR is hence the following:

$$\begin{aligned} |I\rangle &= \frac{1}{2}(|C_{00}\rangle \otimes |00\rangle + |C_{01}\rangle \otimes |01\rangle + |C_{10}\rangle \otimes |10\rangle |C_{11}\rangle \otimes |11\rangle) \\ &= \frac{1}{2}(|11110000\rangle \otimes |00\rangle + |01000100\rangle \otimes |01\rangle \\ &\quad + |10010100\rangle \otimes |10\rangle + |01001001\rangle \otimes |11\rangle). \end{aligned} \tag{14}$$

This is obtained by putting the coordinate qubits $|X\rangle$ and $|Y\rangle$ in a superposition state using Hadamard gates, then entangle them with qubits $|C_{XY}\rangle$ using a series of CNOT gates. The resulting NEQR circuit is illustrated in Fig. 2(right) for the image in Fig. 2(left).

NEQR is a very versatile representation for image computation. However, it presents some drawbacks. As pixels are encoded one by one, large images produce long NEQR

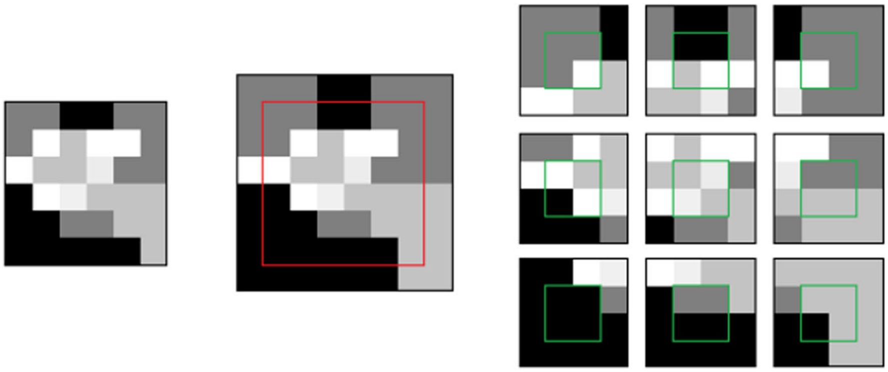


Fig. 3 Image pre-processing: image (left), padded image (center) and extracted patches (right)

circuits; this means that circuit depth grows linearly with image size. As the image size grows, the number of coordinate qubits become larger and the control qubits needed for encoding quickly become more than two. A Toffoli gate with more than two control qubits is defined Multiple CNOT gate, or just MCX: this operator is decomposed before execution, using many Toffoli gates and some auxiliary qubits called *ancilla*. From an efficiency point of view, more coordinate qubits means larger MCX to be used, which leads to a larger amount of gates.

3.2 Image pre-processing

In order to reduce the memory usage, we split a pre-padded image (one-pixel bordered) into many smaller overlapped patches, 4×4 sub-images. Once extracted, a patch of pixels will be processed by the quantum TV algorithm. At the end of each iteration, will be necessary to reassemble the resulting image from the output patches.

Figure 3 illustrates the image pre-processing procedure for a sample image: red square frames original image, while green squares in patches highlight the processed pixels.

In order to speed up quantum circuits generation, we have subdivided workload using multi-threading: each thread is tasked to assemble a QTV for each image patch, using pre-assembled circuits and generating remaining ones. This approach considerably accelerated the execution process for what regards the generation phase.

3.3 Image measurement

The image extraction from the QIR format is a not-trivial process and its performance depends on the quantum representation used in the algorithm.

A quantum representation collects all image's data in a single quantum state. Due to the nature of this particular quantum state, the extraction is not a deterministic process: for each measurement, one of the possible pixel coordinate-value association is randomly obtained as outcome from the *collapse* of the quantum state. For a complete recovery of an image, it is necessary to execute the same algorithm many times.

The image measurement is the last step in Fig. 1.

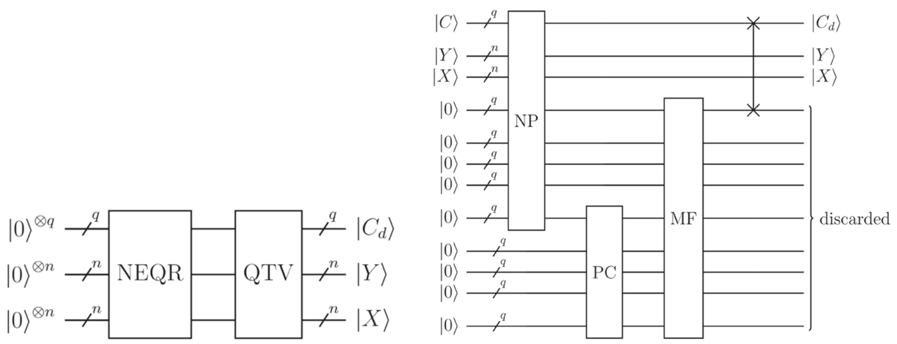


Fig. 4 Quantum TV general scheme (left); a detail of the QTV sub-circuits (right)

4 Quantum TV filter

In this section we introduce the Quantum TV Filter, named QTV, a quantum implementation of the TV regularization algorithm described in Sect. 2 applied to qubits involved in the NEQR quantum image representation. This work extends and improves the work in [10] where the authors proposed a quantum solution for implementing a simple median filter for image processing.

The proposed circuit processes an image input in NEQR representation and provides a denoised image in output in the same NEQR form, according to the scheme in Fig. 4(left).

According to the TV algorithm presented in Sect. 2 we have to iterate a core process for each pixel of the input image. Considering a four-pixel neighborhood configuration, this process is composed by three steps defined as three different quantum operators acting for each pixel:

1. *Neighborhood Preparation* (NP): collect neighboring pixels and extract their values u_u, u_d, u_l, u_r;
2. *P-values Computation* (PC): compute weighted values p_0, p_1, p_2, p_3, p_4;
3. *Median Function* (MF): extract median value from set $\{u_u, u_d, u_l, u_r, p_0, p_1, p_2, p_3, p_4\}$

The quantum operators are assembled into the QTV structure illustrated in Fig. 4(right).

In the following, we will describe in detail the three operands which characterize the Quantum TV. Each one is composed of many sub-circuits, or modules.

4.1 Neighborhood preparation

This operand is in charge of extract neighboring pixels from NEQR representation. At this aim, we used Cycle-Shift (CS) module to change a coordinate register's value, allowing us to shift an image up, down, left or right; see Appendix A for details.

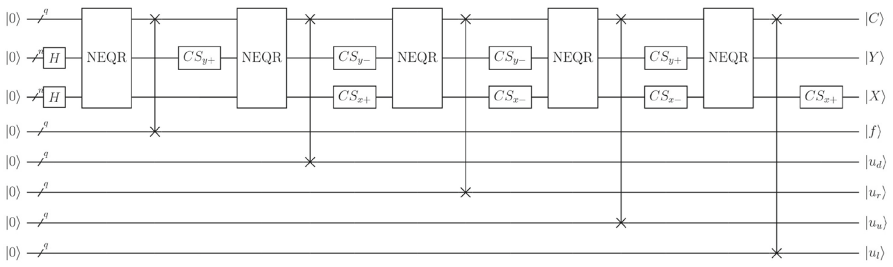
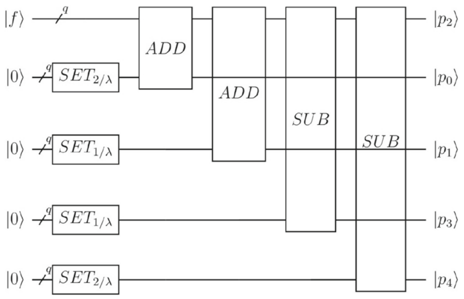


Fig. 5 Neighborhood Preparation module

Fig. 6 P-values computation module



The structure of the NP operand, which output is a quantum superposition state of the central f, up u_u, down u_d, left u_l and right u_r pixel values, is illustrated in Fig. 5. Specifically, once obtained a pixel value from NEQR, we use CS to shift coordinate values, then we re-apply NEQR to gather a new pixel value corresponding to the new position data. With the exception of the first NEQR for image encoding, we avoid the usage of H-gates applied to coordinate registers, as we want to extract a specific (and not random) color outcome.

The output of the NP operand is a quantum state obtained starting from $|\psi\rangle = |0\rangle^{\otimes(q+2n+5q)}$, which reads as

$$|\psi_{NC}\rangle = \frac{1}{2^n}\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_l\rangle|u_u\rangle|u_r\rangle|u_d\rangle|f\rangle|0\rangle|X\rangle|Y\rangle \tag{15}$$

where each qubit in the $|C\rangle$ register is reset to the $|0\rangle$ state.

More in detail, following the scheme in Fig. 5, we have

$$\begin{aligned}
\text{NEQR}\cdot \text{H}_{XY}|\psi\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|f\rangle|X\rangle|Y\rangle)\otimes|0\rangle^{\otimes 5} &&= |\psi_1\rangle\\
\text{SWAP}|\psi_1\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|f\rangle|0\rangle|X\rangle|Y\rangle)\otimes|0\rangle^{\otimes 4} &&= |\psi_2\rangle\\
\text{CS}_{y+}|\psi_2\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|f\rangle|0\rangle|X\rangle|Y+1\rangle)\otimes|0\rangle^{\otimes 4} &&= |\psi_3\rangle
\end{aligned}$$

$$
\begin{aligned}
\text{NEQR}|\psi_3\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|f\rangle|u_d\rangle|X\rangle|Y+1\rangle)\otimes|0\rangle^{\otimes 4} &&= |\psi_4\rangle\\
\text{SWAP}|\psi_4\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_d\rangle|f\rangle|0\rangle|X\rangle|Y+1\rangle)\otimes|0\rangle^{\otimes 3} &&= |\psi_5\rangle\\
\text{CS}_{y-}\text{CS}_{x+}|\psi_4\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_d\rangle|f\rangle|0\rangle|X+1\rangle|Y\rangle)\otimes|0\rangle^{\otimes 3} &&= |\psi_6\rangle\\
\text{NEQR}|\psi_6\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_d\rangle|f\rangle|u_r\rangle|X+1\rangle|Y\rangle)\otimes|0\rangle^{\otimes 3} &&= |\psi_7\rangle\\
\text{SWAP}|\psi_7\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_r\rangle|u_d\rangle|f\rangle|0\rangle|X+1\rangle|Y\rangle)\otimes|0\rangle^{\otimes 2} &&= |\psi_8\rangle\\
\text{CS}_{y-}\text{CS}_{x-}|\psi_8\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_r\rangle|u_d\rangle|f\rangle|0\rangle|X\rangle|Y-1\rangle)\otimes|0\rangle^{\otimes 2} &&= |\psi_9\rangle\\
\text{NEQR}|\psi_9\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_r\rangle|u_d\rangle|f\rangle|u_u\rangle|X\rangle|Y-1\rangle)\otimes|0\rangle^{\otimes 2} &&= |\psi_{10}\rangle\\
\text{SWAP}|\psi_{10}\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_u\rangle|u_r\rangle|u_d\rangle|f\rangle|0\rangle|X\rangle|Y-1\rangle)\otimes|0\rangle &&= |\psi_{11}\rangle\\
\text{CS}_{y+}\text{CS}_{x-}|\psi_{11}\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_u\rangle|u_r\rangle|u_d\rangle|f\rangle|0\rangle|X-1\rangle|Y\rangle)\otimes|0\rangle &&= |\psi_{12}\rangle\\
\text{NEQR}|\psi_{12}\rangle &= \tfrac{1}{2^n}(\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_u\rangle|u_r\rangle|u_d\rangle|f\rangle|u_l\rangle|X-1\rangle|Y\rangle)\otimes|0\rangle &&= |\psi_{13}\rangle\\
\text{SWAP}|\psi_{13}\rangle &= \tfrac{1}{2^n}\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_l\rangle|u_u\rangle|u_r\rangle|u_d\rangle|f\rangle|0\rangle|X-1\rangle|Y\rangle &&= |\psi_{14}\rangle\\
\text{CS}_{x+}|\psi_{14}\rangle &= \tfrac{1}{2^n}\textstyle\sum_{X=0}^{2^n-1}\sum_{Y=0}^{2^n-1}|u_l\rangle|u_u\rangle|u_r\rangle|u_d\rangle|f\rangle|0\rangle|X\rangle|Y\rangle &&= |\psi_{NC}\rangle
\end{aligned}
$$

4.2 P-values computation

Starting from the obtained neighborhood values, we compute the p-values according to relations (10). This reduces to adding some constant values to f, the central pixel.

However the QTV algorithm only works in unsigned integer arithmetic, while the TV algorithm works with floating point numbers, thus getting more accurate and precise results. Therefore, instead of the p-values $p_i = f + \frac{W_i}{2\lambda}$, we force approximated rounded values

$$p_i = f + round(\frac{W_i}{2\lambda}). \tag{16}$$

The p-values are then given by:

$$
\begin{array}{ll}
W_0 = 4, & p_0 = f + round(2/\lambda)\\
W_1 = 2, & p_1 = f + round(1/\lambda)\\
W_2 = 0, & p_2 = f\\
W_3 = -2, & p_3 = f - round(1/\lambda)\\
W_4 = -4, & p_4 = f - round(2/\lambda).
\end{array} \tag{17}
$$

The final design of the P-values Computation module is hence illustrated in Fig. 6, and it is composed by three sub-circuits for setting, adding and subtracting the mentioned constants. In particular:

- SETTER module: assign the round to the nearest integer of a given value to a register, which is used to encode our constants;
- ADDER module: add two values encoded in two quantum registers. We refer to the Appendix A for more details;

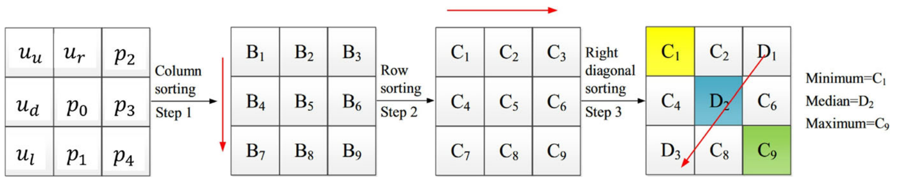
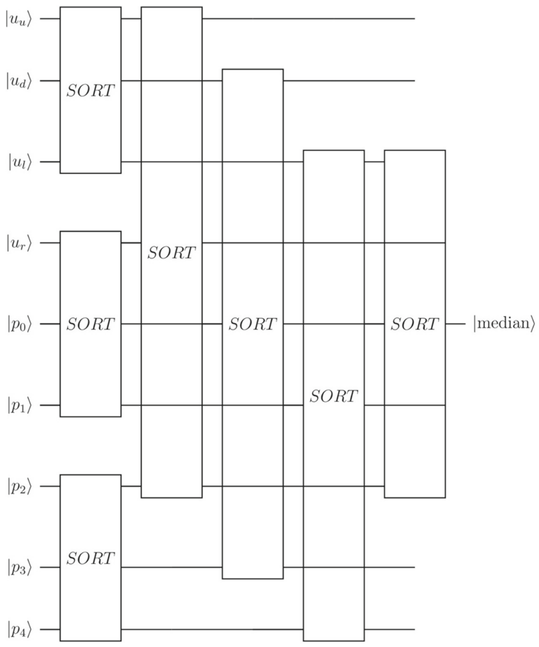


Fig. 7 The basic sorting strategy

Fig. 8 Median function module



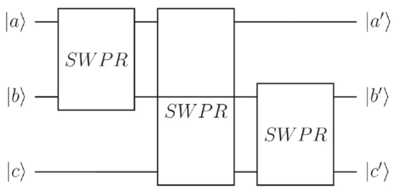
- SUBTRACTOR module: subtract two values by using an adder module, according to $\alpha - \beta \Rightarrow \overline{\overline{\alpha} + \beta}$.

4.3 Median function

To determine a median from a set of values, we need to sort them. The sorting strategy here adopted follows the proposal in [10] for a set of 9 sortable elements. If we re-arrange these values in a matrix form, then we simply need to follow these three steps in pipeline: sort each column, sort each row, and sort the right diagonal. This will guarantee us to store the median value in the central quantum register. The strategy is illustrated in Fig. 7.

The corresponding quantum circuit is shown in Fig. 8.

The Sort module is then the core of Median Function module. Sort module has to order its three input registers. The total ordering of three elements is a trivial problem, as it is reduced to compare two positive integer values and, if not ordered, swap them. A sequence of three comparisons is necessary and sufficient to reach a correct ordering.



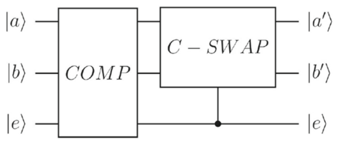


Fig. 9 Sort module (left); Swapper module (right)

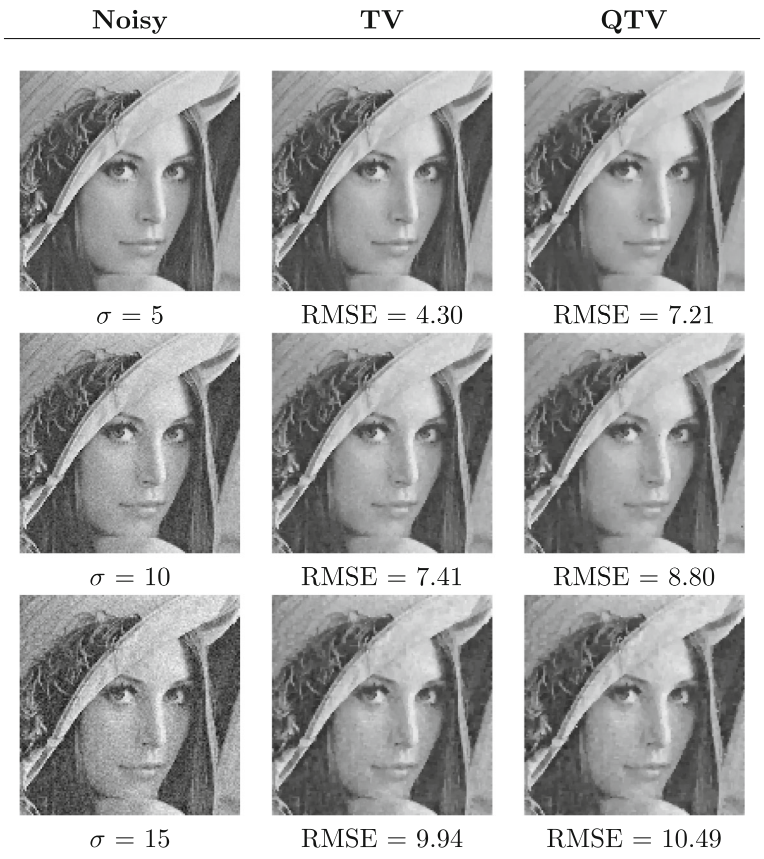


Fig. 10 Lena denoising results for AWGN

A Sort module is then a sequence of three sub-circuits, named Swapper (SWPR). A Swapper module is in turn composed by two sub-circuits: a Comparator (COMP) and a Controlled Swap (C-SWAP). Sort and Swapper modules are shown in Fig. 9.

The Comparator module evaluates two register values a, b and provides, on an auxiliary qubit e, the result of the comparison $a > b$, as follows

$$\text{if} \quad a \leq b \quad \text{then} \quad e = |0\rangle \quad \text{else} \quad e = |1\rangle. \tag{18}$$

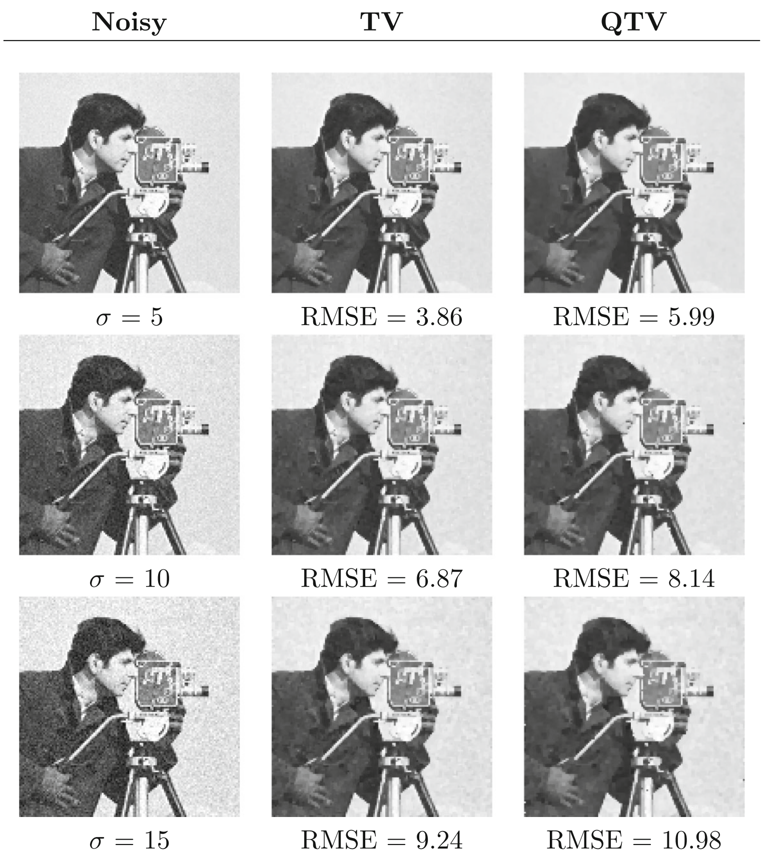


Fig. 11 Cameraman denoising results for AWGN

This result will be next used to control a C-SWAP gate, so that if $e = |1\rangle$, then the values a and b are swapped.

The Comparator module is described in Appendix A. The designed quantum circuit for the Comparator module is more efficient with respect to other proposals. For example, in case $q = 8$, this circuit uses less quantum elementary gates than other existing methods: depth = 64 with respect to the Sort module applied in [10] which has depth = 1.091.767.

5 Circuit complexity analysis

In order to estimate a quantum circuit efficiency, we have to look at its depth, that is the longest path in it. The path length is always an integer number, representing the number of gates it has to execute in that path. At this aim, we are going to analyze each operand of Quantum TV to derive an approximate estimation of its depth.

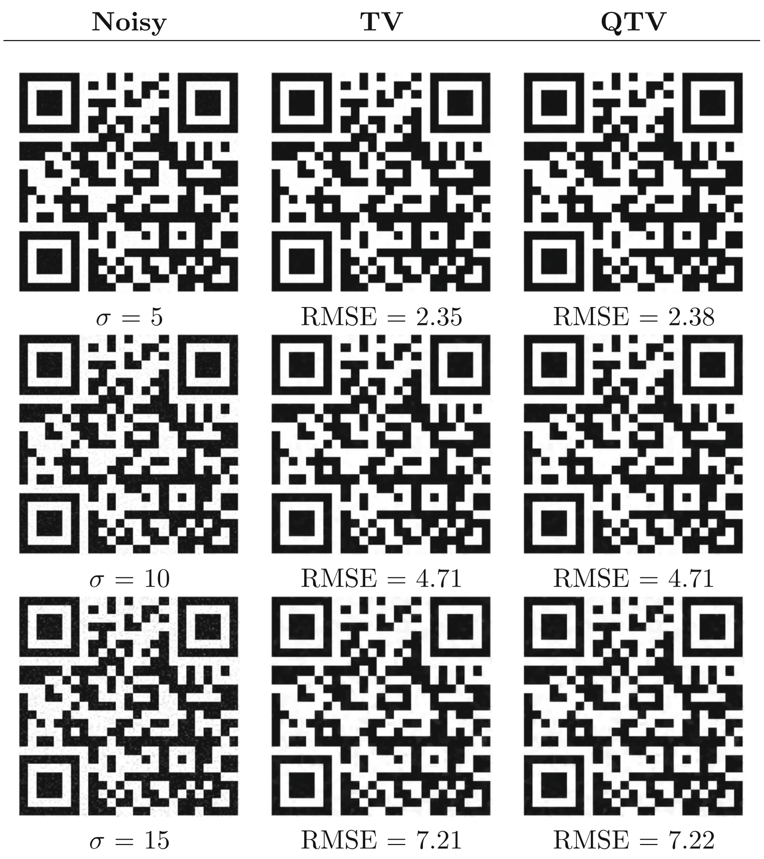


Fig. 12 QRCode denoising results for AWGN

Neighborhood Preparation is for sure the most demanding operand of all filtering algorithm, because it uses multiple instances of NEQR circuit, which length is variable according to the number of pixel to encode. Considering a NEQR implementation that commits the least amount of MCX gates and encodes N pixels, its depth grows polynomially with N as follows,

$$\mathrm{NEQR}_{\mathrm{depth}} = N(4(2\log_2 \sqrt{N} - 1) + 8) \Rightarrow \mathcal{O}(8N\log_2 \sqrt{N} + 4N).$$

Other modules involved are Swap and Cycle Shift. Swap depth depends on the number q of color qubits, thus its depth is always equal to that value. Cycle Shift depth, instead, depends on coordinate register size and it is equal to $\log_2 \sqrt{N} \cdot (\log_2 \sqrt{N} - 1)$.

Neighborhood preparation uses NEQR, SWAP and CS five times in a row. Hence the NP module overall depth is polynomial in N and we can estimate the NP depth as:

$$\mathrm{NP}_{\mathrm{depth}} = 5 \cdot (8N\log_2 \sqrt{N} + 4N + q + \log_2 \sqrt{N} \cdot \log_2 \sqrt{N} - 1))$$

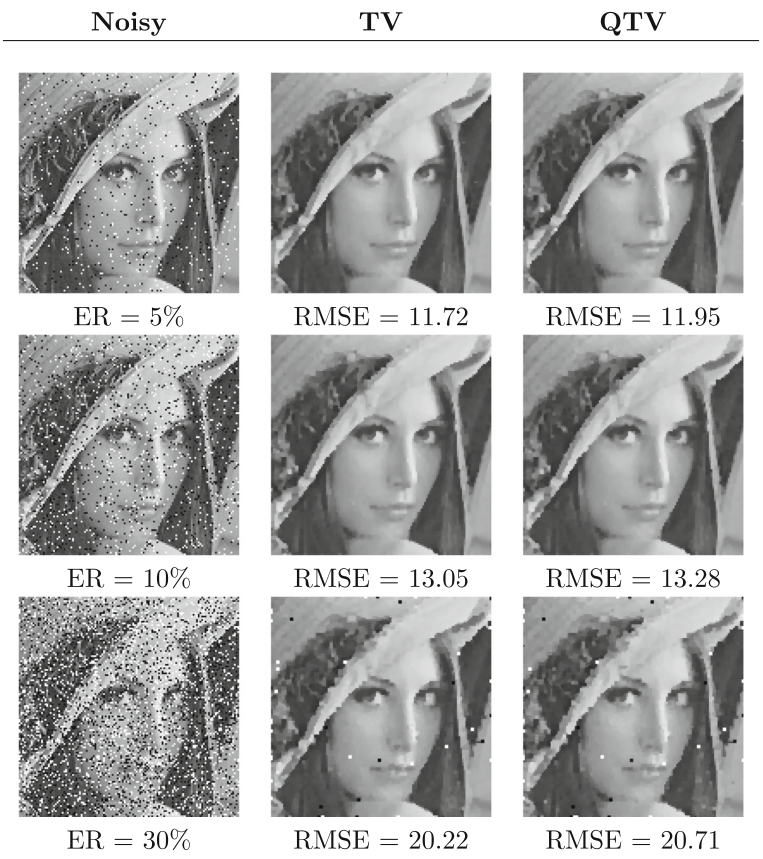


Fig. 13 Lena denoising results for SPN

For what concerns the P-values Computation depth, the setting phase involves four SET modules, which however are applied simultaneously, so they count as a single one. An Adder module depth instead is dependent on the q value. Full-Adder is composed of q Half-Adder, with constant depth (considering also Reset gates); then it is followed by $4q + 1$ sequentially placed gates. Thus we have:

$$\mathrm{ADD}_{\mathrm{depth}} = 9q + 4q + 1 = 15q + 1$$
$$\mathrm{SUB}_{\mathrm{depth}} = \mathrm{ADD}_{\mathrm{depth}} + 2 = 15q + 3$$

From these results, we can derive the P-values Computation module total depth which is polynomial in q:

$$\mathrm{PC}_{\mathrm{depth}} = 60q + 8 \Rightarrow \mathcal{O}(poly(q))$$

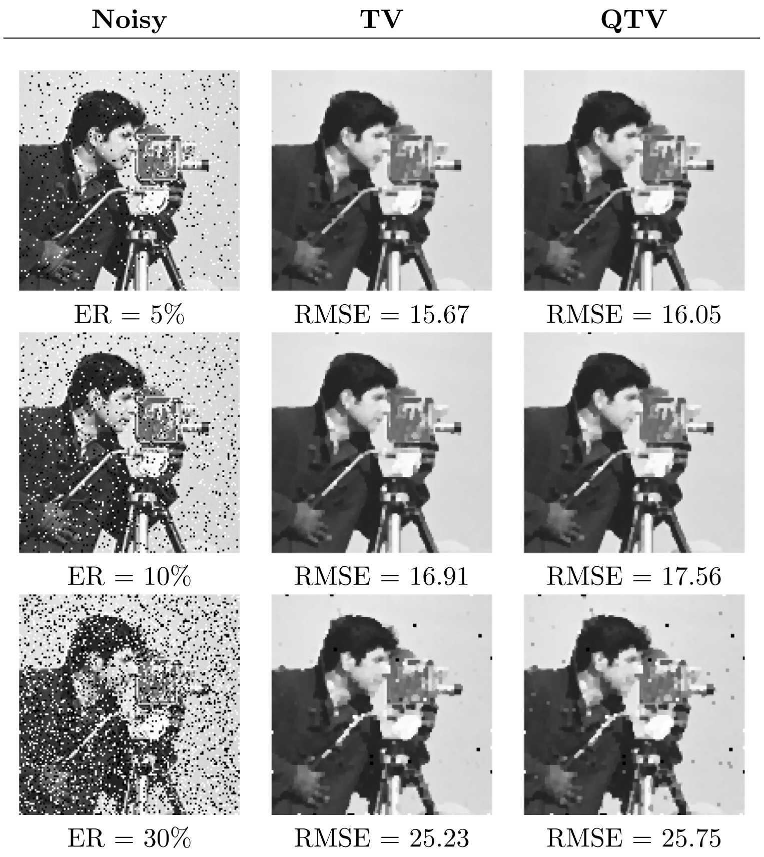


Fig. 14 Cameraman denoising results for SPN

To compute Median Filter depth, we just need to estimate SWPR module depth and count its occurrences in the circuit. To do so, we add up Comparator and C-SWAP depths.

Comparator's depth depends on q, as for each color qubit it uses 6 gates. Although Toffoli gate used in this module counts as three, as two additional X-gates need to be added in order to work. This means that module depth is estimated as $8q$. C-SWAP depth is the same as SWAP. This means that Swapper's total depth is estimated as $8q + q = 9q$.

In Median Filter there are multiple occurrences of SWPR module, which can be further reduced if row and column sorting are executed simultaneously:

$$\mathrm{MF_{depth}} = 9 \cdot 9q = 81q \Rightarrow \mathcal{O}(poly(q)).$$

From this analysis we have derived that Quantum TV algorithm implements the NP module with polynomial complexity in the number of pixels N. The PC and MF

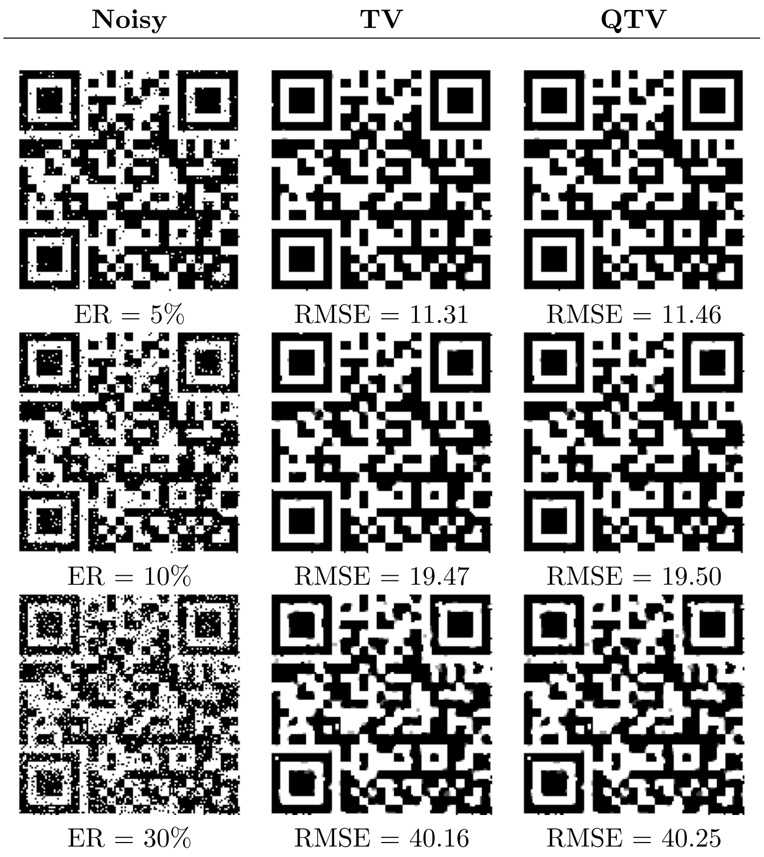


Fig. 15 QRCode denoising results for SPN

modules instead have a polynomial complexity in $q = \lceil \log_2 N_q \rceil$, i.e. the logarithm of the number of colors, hence providing an exponential speedup.

6 Experimental results

In this section we evaluate the performance of the quantum TV algorithm (QTV) on denoising grayscale images, and we present some preliminary results from the comparison with the variational anisotropic TV in Algorithm 1.

The reference images used for the test have dimension 128×128 pixels and grayscale values (8-bit color depth). The discrete model of the image degradation process under noise corruptions can be written as:

$$f = \mathcal{N}(\bar{u}) \tag{19}$$

where $\bar{u}, f \in \mathbb{R}^N$ represent vectorized forms of the unknown clean image and of the observed corrupted image, respectively, while $\mathcal{N}(\cdot)$ denotes the noise corruption operator, which in most cases is of random nature.

In this work we considered two important types of noise, namely the additive (zero-mean) white Gaussian noise (AWGN) and the impulsive salt and pepper noise (SPN), which models saturated or dead pixels.

Denoting by $\Omega := \{1, \ldots, N\}$ the set of all pixel positions in the images, for these two kinds of noise the general degradation model in (19) reads as

$$\begin{array}{ll} \text{AWGN :} & \text{SPN :} \\ f_i \;=\; \bar{u}_i + n_i \quad \forall\, i \in \Omega\,; & f_i \;=\; \begin{cases} \bar{u}_i & \text{for} \quad i \in \Omega_0 \subseteq \Omega \\ n_i \in \{V_{min}, V_{max}\} & \text{for} \quad i \in \Omega_1 := \Omega \setminus \Omega_0. \end{cases} \end{array}$$

In case of SPN, only a subset Ω_1 of the pixels is corrupted by noise, whereas the complementary subset Ω_0 is noise-free. In particular, the corrupted pixels can take only the two possible extreme values V_{min}/V_{max} , where in our case assume 0 and 255 values, with the same probability. The amount of noise can be measured with an *error rate* computed as follows:

$$ER_{\%} = \frac{\text{number of corrupted pixels}}{\text{number of pixels in image}} \times 100.$$

For what concerns AWGN, the additive corruptions $n_i \in \mathcal{G}(\sigma, 0)$, $i \in \Omega$, represent independent realizations from the same univariate Gaussian distribution with zero mean and standard deviation σ.

The performance has been evaluated by the Root-Mean-Square Error (RMSE) metric, defined as:

$$RMSE(\bar{u}, u) := \sqrt{\frac{\sum_{i=1}^{N} (\bar{u}_i - u_i)^2}{N}},$$

where $\bar{u}$ is the original reference image and u is the denoised output image. A lower RMSE indicates a more precise reconstruction.

For the remaining part of this work, when you come across the terms classical/quantum algorithm, we are referring to their respective implementation.

The selection of the regularization parameter λ, which exerts a crucial effect on the solution, has been carried out, for each test, by running the TV algorithm for a range of λ values in order to select by the trial-error strategy the optimal regularization parameter. Then the estimated selected optimal λ has been used in the quantum TV algorithm in order to compare the results of the two algorithms with the same optimal parameter.

A fundamental concept we must keep in mind when analyzing the obtained results, is that classical TV denoising uses floating point numbers, usually along with a value normalization, to be as accurate as possible. This leads to a more precise outcome and a *finer* quantization of the output image. On the other hand, our quantum computation uses integer numbers, so an approximation had to be applied (as previously described in Sect. 4). This difference has an impact on the output images, which present *coarser* improvements than the classical ones.

For testing purpose, we created an implementation of Quantum Median Filter using Qiskit, a Python library for quantum computing simulation [9, 15, 17]. The simulations of the quantum algorithm has ran on the Galileo100 supercomputer (CINECA), with the following cluster configuration:

- Nodes: 348 standard nodes
- Processors: 2xCPU x86 Intel Xeon Platinum 8276-8276L (2.4Ghz)
- Cores: 16704 (48 cores/node)
- RAM: 384GB

6.1 Example 1: AWGN denoising

We consider the problem of denoising the three test images *lena*, *QR*, *cameraman*, corrupted only by AWG noise with standard deviation $\sigma = \{5, 10, 15\}$, as shown in the first column of Figs. 10, 11, 12.

The denoised images obtained by applying TV and QTV algorithms are illustrated in Figs. 10, 11, 12, column-wise for the three images. For each denoised image the RMSE obtained value is reported below.

From a visual inspection the denoised images from TV and QTV present a comparable quality, even if the RMSE highlights the lost in accuracy due to the mentioned integer arithmetic representation followed by QTV.

Moreover, the quantum TV algorithm is able to denoise images corrupted by severe Gaussian noise, as illustrated in the last row of Figs. 10, 11, and Fig. 12 for $\sigma = 15$. Unlike, the quantum median filter proposed in [10], as any standard median filtering, is not appropriate to remove this kind of noise. The median filter is instead well-known to be an excellent image denoiser in case of salt-and-pepper noise because it does not blur the image, as a mean filter would do. However, despite its name, the median filter is not a filter because it does not respect the linearity property.

6.2 Example 2: SPN denoising

In this example we applied TV and QTV for the denoising of the three test images *lena*, *QR*, *cameraman*, corrupted by SPN noise with error rate $\mathrm{ER}_{\%} = \{5, 10, 30\}$.

The noisy images are shown in the first column of Figs.13, 14, 15, together with the denoised images in the second (TV) and third (QTV) columns, along with the associated RMSE values, reported in the bottom.

From these results, we can see how well quantum algorithm performs when compared to its variational counterpart. QTV results present excellent qualitative performance, with minimal RMSE differences.

However, by a visual inspection of the denoised images, we notice that, for both the algorithms, some pixel clusters were not completely denoised. This can be due to either to the limited pixel neighborhood considered (4 pixels), or to the L_2-norm metric fidelity in our model (5), when it is well known that SPN can be better treated by a L_1-norm fidelity, which, anyway, leads to a non-differentiable fidelity term.

7 Conclusions and discussion

In this work a quantum approach is proposed for total variation denoising, and its corresponding quantum circuit is designed. Specifically the quantum TV implements the anisotropic median formula presented in [13]. The main idea of the approach is that first the classical image is converted into a quantum version based on the quantum representation (NEQR) of digital images, and then three quantum modules are applied to realize the neighboring collection for each pixel in the image, weight calculation, and median extraction. Finally, an image measurement process collapses the quantum state into a resulting denoised image. From the complexity analysis in Section 5 we derived a polynomial complexity in the number of pixels N for the NP module, and a polynomial complexity in logarithm of the number of colors $N_q = 2^q$ for the PC and MF modules.

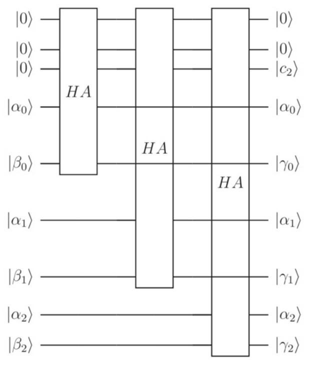
The experimental results show that the quantum TV performance is comparable to the classical variational TV approach. However, we highlighted several issues that need to be addressed to make the proposal a competitive QIP algorithm, as its variational counterpart. For example the Neighborhood collection module is an expensive operand due to the NEQR image representation. Even though NEQR is still one of the most used representation methods in QIP and the most suitable choice for this work, future developments should definitely search for other QIR alternatives, or develop more efficient versions of the same representation, such as parametric quantum circuits that take advantage of data structure for improving image processing activities.

Acknowledgements This work is the result of a collaboration activity with CINECA High Performance Computing center, located in Bologna, Italy. This work was supported in part by the National Group for Scientific Computation (GNCS-INDAM), Research Projects 2020, and in part by MIUR RFO projects.

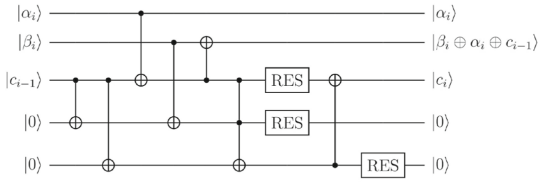
Funding Open access funding provided by Alma Mater Studiorum - Università.

Declarations

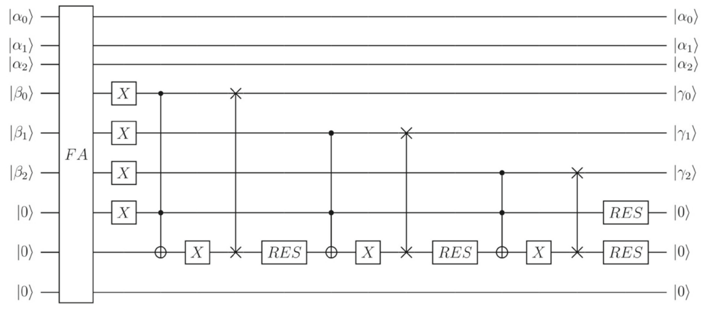
Conflict of interest The authors declare that there are no conflict of interest regarding the publication of this paper.



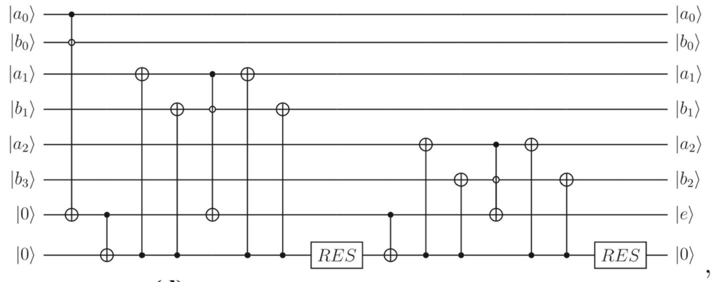
(a) Full adder (FA)



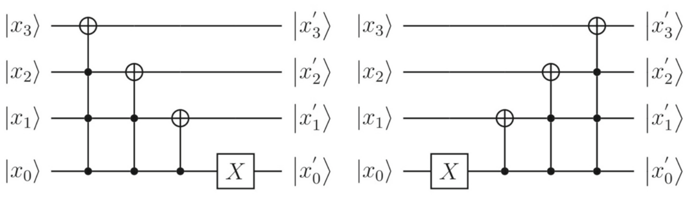
(b) Half adder (HA)



(c) Adder for 3-qubits registers



(d) Comparator module for 3-qubits registers



(e) Cycle Shift modules: CS+ (left); CS- (right)

Fig. 16 Quantum modules used in QTV

Appendix A Quantum gates for QTV

In this appendix we illustrate a few modules composed of basic quantum gates used in the design of QTV algorithm.

We used classical logic for designing a quantum version of a Half-Adder (HA) and a Full-Adder (FA), as illustrated in Fig. 16a, b. These sub-circuits need three additional auxiliary qubits, for storing temporary results needed for summing. FA does not provide a minimum/maximum cap over result: to solve this, we can use the temporary result stored in auxiliary qubits to manually fix the output to the correct value, as illustrated in Fig. 16c.

The Comparator module is illustrated in Fig. 16d. This module uses a very small amount of elementary gates and a reduced number of auxiliary qubits: if Reset option (RES) is not available, only q ancillas are needed for Comparator to work.

A Cycle Shift module (CS) is essentially a modulo-2 adder or subtractor, defined as CS+ and CS− respectively. CS module implementation is illustrated in Fig. 16e.

References

1. Avila, A., Maron, A., Reiser, R., Pilla, M., Yamin, A.: GPU-aware distributed quantum simulation. In: Proceedings of the 29th Annual ACM symposium on applied computing, pp. 860–865 (2014)
2. Bharti, K., Cervera-Lierta, A., Kyaw, T. H., Haug, T., et al.: Noisy intermediate-scale quantum (NISQ) algorithms. Reviews of Modern Physics, American Physical Society. 94(1) (2022)
3. Chambolle, A., Pock, T.: A first-order primal-dual algorithm for convex problems with applications to imaging. J. Math Imaging Vis. **40**, 120–145 (2011)
4. Chan, R.H., Tao, M., Yuan, X.M.: Constrained total variational deblurring models and fast algorithms based on alternating direction method of multipliers. SIAM J. Imag. Sci. **6**, 680–697 (2013)
5. Deutsch, D., Josza, R.: Rapid solutions of problems by quantum computation. Proc. Roy. Soc. London Se. A **439**, 553–558 (1992)
6. Gyongyosi, L., Imre, S.: A Survey on Quantum Computing Technology. Elsevier, Computer Science Review (2018)
7. Grover, L. K. : A fast quantum mechanical algorithm for database search. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of Computing (STOC '96). Association for Computing Machinery, New York, NY, USA, pp. 212–219, (1996)
8. Harris, C.R., Millman, K.J., van der Walt, S.J., et al.: Array programming with NumPy. Nature **585**, 357–362 (2020)
9. IBM Q TEAM, Qiskit: An open-source framework for quantum computing. https://doi.org/10.5281/zenodo.2562110
10. Jiang, S., Zhou, R.G., Hu, W., et al.: Improved quantum image median filtering in the spatial domain. Int. J. Theor. Phys. **58**, 2115–2133 (2019)
11. Lanza, A., Morigi, S., Pragliola, M., Sgallari, F.: Space-Variant TV Regularization for Image Restoration. In: Lecture notes in computational vision and biomechanics. **27**, 160–169 (2018)
12. Levental, M.: Tensor networks for simulating quantum circuits on FPGAs. arXiv:2108.06831 (2021)
13. Li, Y., Osher, S.: A new median formula with applications to PDE based denoising. Commun. Math. Sci. **7**(3), 741–753 (2009)
14. Li, P., Liu, X., Xiao, H.: Quantum image median filtering in the spatial domain. Quantum Inform. Process. **17**(49), 1 (2018)
15. Markov, I.L., Shi, Y.: Simulating quantum computation by contracting tensor networks. SIAM J. Comput. **38**(3), 963–981 (2008)
16. Mengoni, R., Incudini, M., Di Pierro, A.: Facial expression recognition on a quantum computer. Quantum Mach. Intell. **3**(8), 1–11 (2021)
17. Pilch, J., Długopolski, J.: An FPGA-based real quantum computer emulator. J. Comput. Electron. **18**, 329–342 (2019)

18. Preskill, J.: Quantum Computing in the NISQ era and beyond. Quantum, Verein zur Förderung des Open Access Publizierens in den Quantenwissenschaften **2**, 79–99 (2018)
19. Ruan, Y., Xue, X., Shen, Y.: Quantum image processing: opportunities and challenges. Math. Probl. Eng., pp. 1–8, (2021)
20. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Phys. D **60**, 259–268 (1992)
21. Shor, P. W. :Algorithms for quantum computation: discrete logarithms and factoring. In: Proceedings 35th Annual symposium on foundations of computer science, pp. 124–134, (1994)
22. Yan, F., Iliyasu, A.M., Jiang, Z.: Quantum computation-based image representation, processing operations and their applications. Entropy **16**(10), 5290–5338 (2014)
23. Yao, X.-W., Wang, H., Liao, Z., et al.: Quantum image processing and its application to edge detection: theory and experiment. Phys. Rev. X **7**(3), 031041 (2017)
24. Wang, Z., Xu, M., Zhang, Y.: Review of quantum image processing. Arch Comput. Methods. Eng. **29**, 737–761 (2022)
25. Wu, C., Tai, X.-C.: Augmented lagrangian method, dual methods, and split bregman iteration for ROF, vectorial TV, and high order models. SIAM J. Imaging Sci. **3**(3), 300–339 (2010)