IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 10 November 2022, accepted 17 November 2022, date of publication 28 November 2022,
date of current version 2 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3225411

RESEARCH ARTICLE

Reverse Image Filtering Using Total Derivative Approximation and Accelerated Gradient Descent

FERNANDO J. GALETTO AND GUANG DENG
Department of Engineering, La Trobe University, Melbourne, VIC 3086, Australia
Corresponding author: Fernando J. Galetto (f.galetto@latrobe.edu.au)

ABSTRACT In this paper, a new problem of reverse image filtering is addressed. The problem is to reverse the effect of an image filter given the observation $\boldsymbol{b} = g(\boldsymbol{x})$. The filter g is modelled as an available black box. An inverse method is proposed to recover the input image $\boldsymbol{x}$. The key idea is to formulate this inverse problem as minimizing a local patch-based cost function. To use gradient descent for the minimization, total derivative is used to approximate the unknown gradient. This paper presents a study of factors affecting the convergence and quality of the output in the Fourier domain when the filter is linear. It discusses the convergence property for nonlinear filters by using contraction mapping as a tool. It also presents applications of the accelerated gradient descent algorithms to three gradient-free reverse filters, including the one proposed in this paper. Results from extensive experiments are used to evaluate the complexity and effectiveness of the proposed algorithm. The proposed algorithm outperforms the state-of-the-art in two aspects. (1) It is at the same level of complexity as that of the fastest reverse filter, but it can reverse a larger number of filters. (2) It can reverse the same list of filters as that of a very complex reverse filter, but its complexity is much lower.

INDEX TERMS Inverse filtering, optimization, accelerated gradient descent, total derivative.

I. INTRODUCTION

Solving inverse problems is of critical importance in many science and engineering disciplines. In image processing, a typical problem is called image restoration [1] in which the distortion is modelled by a linear shift invariant (LSI) system and additive noise. Other well-known problems include: dehazing [2], [3] and denoising [4], [5], [6]. In these applications, an edge-aware filter such as [7], [8], [9], [10], [11], and [12] plays an important role. In general, solutions to such problems are model based. For example, in image restoration an observation model with a known or unknown blurring filter kernel is assumed, while in dehazing, a physical image formation model is also assumed. An essential part of such algorithms is to estimate model parameters such as the filter kernel in image restoration or the transmission map in dehazing. Recently, deep neural networks are used in solving inverse problem in imaging applications such as inverse tone mapping [13] and de-blurring [14], where a neural network is trained to solve a specific inverse problem. Other notable examples of solving inverse problems are computational imaging techniques, such computed tomography [15] and compressive sensing [16]. But they are not related to the work presented in this paper.

The increasing use of many image processing filters has led to a new formulation of the inverse problem. Let $g(.)$ represent an image filter which is usually available as a software tool. The user of the filter has no knowledge of the exact algorithm. As such, the filter can be regarded as an available black box which takes an input image $\boldsymbol{x}$ and produces an output image $\boldsymbol{b} = g(\boldsymbol{x})$. The new inverse problem is thus to accurately estimate the original image $\boldsymbol{x}$ given the observation $\boldsymbol{b}$ and the black box filter $g(.)$.

Two factors make the study of such a problem theoretically and practically important.

- Because traditional methods rely on using gradient information to solve optimization problems, they are not suitable for solving this new problem because the filter is unknown and is not limited to be linear. New algorithms, which do not directly rely on the gradient information, must be developed.
- The solution of this new inverse problem only relies on the filter as an available black box. Thus, it can

The associate editor coordinating the review of this manuscript and approving it for publication was Yiming Tang.

TABLE 1. Comparison of the 4 space domain methods and a frequency domain method. The complexity is measured in one iteration. The constant C represents the complexity of the filter $g(.)$.

Method	Update	Para.	Complexity	Filter calls
T [17]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{q}_k$	N/A	$\max(O(n), C)$	1
R [18]	$\boldsymbol{x}_{k+1} = \alpha\boldsymbol{x}_k + \lambda\boldsymbol{q}_k$	α, λ	$\max(O(n), C)$	1
P [19]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \frac{\|\|\boldsymbol{q}_k\|\|}{2\|\|\boldsymbol{p}_k\|\|}\boldsymbol{p}_k$	N/A	$\max(O(n^2), C)$	3
TDA	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \lambda\boldsymbol{t}_k$	$\lambda \leq 1$	$\max(O(n), C)$	2
F [20]	$\mathcal{F}(\boldsymbol{x}_{k+1}) = \frac{\mathcal{F}(\boldsymbol{b})}{\mathcal{F}(g(\boldsymbol{x}_k))\mathcal{F}(\boldsymbol{x}_k)}$	N/A	$\max(O(n \log n), C)$	1

potentially deal with a wide range of unknown linear and nonlinear filters.

A review of four methods on this new problem is presented in section II. They are three spatial domain methods: the T-method [17], R-method [18], and P-method [19] and a frequency domain method called F-method [20]. Except the T-method, which can be motivated as a fixed-point algorithm, the other three share a similar idea of formulating a cost function and use gradient approximation to derive the solution to the optimization problem. These algorithms have been used to reverse a wide range filters including some traditional filters [1] such as: Gaussian filter, disk averaging filter, motion blur filter, Laplace-of-Gaussian (LoG), and some edge-aware filters such as: rolling guidance filter (RGF) [21], adaptive manifolds filter (AMF) [22], structure extraction from texture via relative total variation (RTV) [23], image smoothing via iterative least squares (ILS) [24], image smoothing via L_0 minimization (L0) [25], bilateral filter (BF) [7], self-guided filter guided filter (GF) [8] and GF with the guidance image generated from filtering an image by a Gaussian filter.

This work is motivated by the P-method and the R-method. Our goal is to develop a principle-based new algorithm which is of low complexity and is highly effective (able to reverse the effect of a wide range of filters). The basic idea for our development is similar to that of the P- and R-method. The main differences are in the cost function formulation and gradient approximation. While authors of the P- and R-method use a global cost function, we use a patch-based local cost function. We have also used total derivative approximation (TDA) of the gradient. This differs from the approximation approach of the P- and R-method. The new algorithm is thus called the TDA-method. In addition, we have studied the application of well-known accelerated gradient descent (AGD) methods [26] which are listed in Table 2. AGD methods are commonly used in machine learning [27], but their applications have not been explored in this context. We have conducted extensive experiments to show that while the proposed TDA-method is of the same level of complexity as that of the T-method, it is as effective as the P-method but at a much lower complexity.

Key contributions of this work are as follows.

- In section 3, we present the main theoretical results.
  - The development of the TDA-method in section III-A.
  - A theoretical analysis of the convergent condition and factors related to the quality of the recovered image in section III-C for linear filters. We note that this is perhaps the only type of filter that allows an analytical analysis. The analysis shows why the T-method fails to reverse a linear filter and how the proposed TDA-method overcomes this limitation.
  - For nonlinear filters which do not have analytical representations, we point out that the reverse filtering methods can be regarded as fixed-point iterations and that contraction mapping is a theoretical tool to study the convergence in section III-C.
  - We discuss the relationship between the T-method, TDA-method, and P-method and their computational complexity in section III-D. We show that the TDA-method is related to the T-method in that the TDA-method has an extra correction step.
  - We apply AGD methods to 3 spatial domain reverse filtering algorithms, including the TDA-method in section III-E.
- In section 4 we show results of extensive experimental study of the proposed method.
  - The effect of parameter settings such as learning rate and number of iterations (section IV-A).
  - Qualitative and quantitative evaluations to demonstrate the performance of previous works and the TDA-method in reversing a wide range of filters (section IV-B).
  - A study of using AGD methods to accelerate the image reverse filtering algorithms (section IV-C).
  - A case study to illustrate the limitation of gradient-free image reversing algorithms (section IV-D).

We adopt the following notations. An image is represented as a matrix $\boldsymbol{x}$ and the filter denoted $g(.)$ operates on the image and produces the observed image $\boldsymbol{b} = g(\boldsymbol{x})$. The subscript is used to represent iteration index. Arithmetic operations are conducted pixel-wise. $||\boldsymbol{x}_k||$ is the 2-norm of the image at kth iteration. The Fourier transform of $\boldsymbol{x_k}$ is represented as $\mathcal{F}(\boldsymbol{x}_k)$. A pre-print version of this paper appeared at arXiv.org with the citation index "arXiv:2112.04121" on 8th Dec. 2021.

Finally, we remark that there is a fundamental difference between the neural network approach and the family of inverse filters studied in this paper. A neural network is often trained to solve a particular problem such as de-blurring an

image due to a particular filter. The training requires a huge dataset and fine-tuning of parameters. On the other hand, the family of inverse filters are designed to deal with a wide range of filters without training and without parameters.

II. PREVIOUS WORK

We follow the terminology used in [19] which called a particular reverse filter a method. We briefly comment on each of them and present a summary in Table 1 where we define the following variables.

- $\boldsymbol{q}_k = \boldsymbol{b} - g(\boldsymbol{x}_k)$.
- $\boldsymbol{p}_k = g(\boldsymbol{x}_k + \boldsymbol{q}_k) - g(\boldsymbol{x}_k - \boldsymbol{q}_k)$.
- $\boldsymbol{t}_k = g(\boldsymbol{x}_k + \boldsymbol{q}_k) - g(\boldsymbol{x}_k)$.

The T-method [17], can be formulated as solving a fixed-point iteration problem [28]

$$\boldsymbol{x} = \boldsymbol{x} + \boldsymbol{b} - g(\boldsymbol{x}). \tag{1}$$

It is the fastest method among all reverse filtering methods considered in this paper. When the contraction mapping condition (discussed in section III-B) is satisfied for a particular filter, the T-method produces good results. Otherwise, the iteration is unstable, leading to unbounded results. An unstable example is the reverse of an average filter. In addition, it was shown in [29] that the T-method can be motivated from Bregman's iteration point view. If a blurring filter is the result of solving a variational problem, then the T-method can completely recover the original image.

The R-method [18] is developed by minimizing the cost function:

$$J(\boldsymbol{x}) = \boldsymbol{x}^T(g(\boldsymbol{x}) - \boldsymbol{b}) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{x} \tag{2}$$

by using approximations/assumptions and gradient descent. The convergence condition is that $g(.)$ must be Lipschitz continuous [30]. The T-method can be considered as a special case of the R-method. This connection provides a gradient descent interpretation of the T-method, which permits applying the accelerated gradient descent algorithms. The R-method can reverse the effect of a wider range of operators than the T-method. However, because of the assumptions and approximations, the R-method only performs well for filters which mildly alter the original image. In a recent paper [31], similar ideas as that of the R- and T-method are used to develop an algorithm to remove mild defocus and motion blur from natural images.

The P-method and the S-method [19] are inspired by gradient descent methods studied by Polak [32] and Steffensen [33]. They deliver successful results and converge for a larger number of linear and non-linear filters than other methods. However, they have a high computational cost due to the calculation of the 2-norm of a matrix in each iteration. The 2-norm is the largest singular value of the matrix and has a computational complexity of $O(n^2)$. Because the P-method is more stable than the S-method [19], we only consider the P-method in this work.

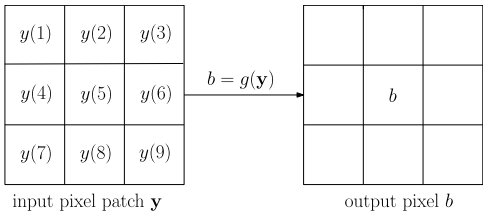


FIGURE 1. Illustration of filtering a 3 × 3 patch resulting a pixel.

The F-method [20] uses the Newton-Raphson technique to invert unknown linear filters in the frequency domain. Approximation of the gradient is also used. Although the F-method produces good results for some filters and is reported to have better performance than the T-method and the R-method, it only deals with linear shift invariant (LSI) filters. If the frequency response of the LSI filter has zeros in the frequency range of $[0, \pi]$, then the iteration is unstable. This is a classical problem in inverse filtering, which can be dealt with by using a Wiener filter in the Fourier domain [1]. A recent paper [34] described several iterative methods in the frequency domain. These methods are based on an assumption that the black box filter can be approximately modelled as a linear filter.

III. THE TDA-METHOD, ANALYSIS AND ACCELERATIONS

A. THE PROPOSED TDA-METHOD

To develop the proposed algorithm, we assume the filter only acts on a patch of the pixels. For notational simplicity, we represent a patch of pixels as a vector $\boldsymbol{y}$ such that the output pixel is: $b = g(\boldsymbol{y})$. We illustrate this in Fig. 1. The vector is

$$\boldsymbol{y} = [y(1), y(2), y(3), y(4), y(5), y(6), y(7), y(8), y(9)]^T \tag{3}$$

where the pixel to be estimated is $y(5)$ and b is the observed pixel corresponding to $y(5)$. For this particular case, the filter function has the property: $g : \mathbb{R}^9 \to \mathbb{R}$.

We would like to minimize the following cost function to determine $y(5)$:

$$f(\boldsymbol{y}) = \frac{1}{2}\{b - g(\boldsymbol{y})\}^2 \tag{4}$$

Using gradient descent, we can write

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k - \lambda \nabla f(\boldsymbol{y}_k) \tag{5}$$

where

$$\nabla f(\boldsymbol{y_k}) = -u_k(5)\boldsymbol{d}, \tag{6}$$

and

$$u_k(5) = b - g(\boldsymbol{y}_k), \tag{7}$$

The nth element of the vector $\boldsymbol{d}$ is defined as

$$d(n) = \frac{\partial g}{\partial y(n)}(\boldsymbol{y}_k), \tag{8}$$

Here the notation $\frac{\partial g}{\partial y(n)}(\boldsymbol{y}_k)$ represents the partial derivative evaluated at the point $\boldsymbol{y}_k$. Because in each patch we are only

interested in recovering the center pixel of the patch $y(5)$, from equation (5) we have

$$y_{k+1}(5) = y_k(5) + \lambda u_k(5)\frac{\partial g}{\partial y(5)}(\boldsymbol{y}_k). \tag{9}$$

Since the filter function g is unknown, we cannot calculate its derivative and have to resort to approximation. We use the concept of total derivative for the approximation. More specifically, the total derivative of a function $c(\boldsymbol{x}) : \mathbb{R}^N \to \mathbb{R}$ at a point $\boldsymbol{a}$ can be approximated as

$$c(\boldsymbol{a} + \boldsymbol{\delta}) - c(\boldsymbol{a}) \approx \sum_{n=1}^{N} \delta(n)\frac{\partial c}{\partial x(n)}(\boldsymbol{a}) \tag{10}$$

where $x(n)$ and $\delta(n)$ are the nth element of $\boldsymbol{x}$ and $\boldsymbol{\delta}$ respectively.

Using this approximation, we can write the following for the patch of pixels:

$$g(\boldsymbol{y}_k + \boldsymbol{u}_k) - g(\boldsymbol{y}_k) \approx \sum_{n=1}^{9} u_k(n)\frac{\partial g}{\partial y(n)}(\boldsymbol{y}_k) \tag{11}$$

where values of elements of $\boldsymbol{u}_k = [u_k(1), u_k(2), \ldots, u_k(9)]$ are assumed to be small. Except $u_k(5) = b - g(\boldsymbol{y_k})$, other elements are not used in the following approximation.

A comparison equations (9) with (11) suggests a solution of avoiding the calculation of the derivative of an unknown function $\frac{\partial g}{\partial y(5)}$. The solution is through the two-step approximation:

$$u_k(5)\frac{\partial g}{\partial y(5)} \to \sum_{n=1}^{9} u_k(n)\frac{\partial g}{\partial y(n)} \approx g(\boldsymbol{y}_k + \boldsymbol{u}_k) - g(\boldsymbol{y}_k) \tag{12}$$

where the symbol "$\to$" represents the operation of "replaced by". Replacing $u_k(5)\frac{\partial g}{\partial y(5)}$ by the local average $\sum_{n=1}^{9} u_k(n)\frac{\partial g}{\partial y(n)}$ can be regarded as using a smoothed estimate which is further approximated by the total derivative.

Using this approximation in equation (9), we can write

$$y_{k+1}(5) = y_k(5) + \lambda(g(\boldsymbol{y}_k + \boldsymbol{u}_k) - g(\boldsymbol{y}_k)) \tag{13}$$

We now generalize the above result to the whole image. The key idea is to replace the image patch $\boldsymbol{y}_k + \boldsymbol{u}_k$ by the image $\boldsymbol{x}_k + \boldsymbol{q}_k$ where $\boldsymbol{q}_k = \boldsymbol{b} - g(\boldsymbol{x}_k)$. Using this generalization, we can omit the reference to pixel location in (13) and use our general notation of $\boldsymbol{x}_k$ to represent an image in the kth iteration. We can write

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \lambda(g(\boldsymbol{x}_k + \boldsymbol{q}_k) - g(\boldsymbol{x}_k)) \tag{14}$$

Equation (14) is the proposed algorithm formally presented in Algorithm 1. It is thus called total derivative approximation (TDA).

One interesting observation is: the TDA method has a similar form as that of gradient descent where one may assume that $g(\boldsymbol{x}_k + \boldsymbol{q}_k) - g(\boldsymbol{x}_k)$ is the negative gradient of a certain cost function. So, what is the corresponding cost function and under what condition is such assumption true?

Algorithm 1: The TDA Algorithm

Input: g – the filter as a black box, $\boldsymbol{b}$ – observed image to be processed, $\lambda > 0$ – a user defined parameter, and N – number of iterations

Output: $\boldsymbol{x}$

```
k = 0
x_0 = b
while k < N do
    x_{k+1} = x_k + λ(g(x_k + b − g(x_k)) − g(x_k))
    k = k + 1
```

To answer these questions, we start with a case where the signal is 1d and the filter is linear such that the filter model $\boldsymbol{b} = g(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$, where $\boldsymbol{A}$ is matrix defined by the impulse response of the linear filter. We then consider a commonly used cost function

$$J(\boldsymbol{x}) = \frac{1}{2}\|\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}\|_2^2 \tag{15}$$

Its gradient is given by

$$\nabla J(\boldsymbol{x}) = -\boldsymbol{A}^T(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}) \tag{16}$$

Next, we perform the following calculation

$$g(\boldsymbol{x}_k + \boldsymbol{q}_k) - g(\boldsymbol{x}_k) = \boldsymbol{A}(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}) \tag{17}$$

Comparing the above two equations, we can clearly see that the TDA can be regarded as gradient descent for the 1d linear filter case when the filter matrix $\boldsymbol{A}$ is symmetric.

However, we should point out that the above analysis is only true for the 1d case. Although two dimensional linear filters can be represented in the general form of matrix-vector multiplication, the matrix and the vector no longer have the same simple formation as that of the 1d case, see for example, reference [35] pages 29-30. The gradient of the cost function is completely different and is rather complicated. As such, we will not pursue this question further.

B. CONVERGENCE FOR LINEAR FILTERS

It is in general a very difficult task to analyse gradient free algorithms studied in this work. This is because the large number nonlinear filters make the analysis non-tractable. In the following, we focus on studying the properties of the T-method and the TDA-method for the mathematically simplest class of filters–linear filters. We use frequency domain representation such that $X = \mathcal{F}(\boldsymbol{x})$ and $B = \mathcal{F}(\boldsymbol{b}) = \mathcal{F}(g(\boldsymbol{x})) = GX$ where G is the frequency response of a linear filter.

1) FREQUENCY DOMAIN REPRESENTATIONS

We can write the T-method in the frequency domain as follows

$$X_{k+1} = X_k + B - GX_k = B + (1 - G)X_k \tag{18}$$

After k iterations, we can write the above equation in term of the unknown image X as

$$X_k = X - H_0^k I \tag{19}$$

where $H_0 = 1 - G$, $I = (1 - G)X$, and H_0^k represent an element-wise raising to the power of k. The result has two terms: the original image X and the result of iteratively filtering the image (I) k times by the same filter H_0.

For the TDA-method, we perform similar calculations and have the following result:

$$X_k = X - J_0^k I \tag{20}$$

where $J_0 = 1 - G^2$, and $I = (1 - G)X$. The result also has two terms: X and the result of iteratively filtering the image (I) k times by using the filter J_0. In the calculation, we set $\lambda = 1$ such that the TDA-method is consistent with the T-method which is parameter-free. Appendix presents the derivation of equations (19) and (20).

2) ANALYSIS

From the above results, we can write the two iterative reverse filters in a unified form

$$X_k = X - H_k I \tag{21}$$

where $H_k = H_0^k$ and $H_0 = 1 - G$ (T-method), $H_0 = 1 - G^2$ (TDA-method). In order to recover the original image, a sufficient condition is $|H_k| \to 0$ such that $X_k \to X$.

Since H_k is a function of the unknown linear filter G, to perform a concrete analysis we consider the filter G to be a class of non-causal symmetric linear low pass filters such as average filters and Gaussian filters whose coefficients sum to 1. The frequency response of such filters, denoted $G(\omega_1, \omega_2)$, is real and symmetric and has the property $G(\omega_1, \omega_2) \leq 1$.

Using these notations, the amplitude response of the filter is represented as

$$|H_k(\omega_1, \omega_2)| = |H_0(\omega_1, \omega_2)|^k \tag{22}$$

where

$$H_0(\omega_1, \omega_2) = 1 - G(\omega_1, \omega_2) \tag{23}$$

and

$$H_0(\omega_1, \omega_2) = 1 - G^2(\omega_1, \omega_2) \tag{24}$$

for the T- and TDA-method, respectively. Let the maximum value of the amplitude response be

$$T_0(\omega_1^*, \omega_2^*) = \max_{(\omega_1, \omega_2)} |H_0(\omega_1, \omega_2)| \tag{25}$$

where (ω_1^*, ω_2^*) represent frequencies where the amplitude response is at maximum. The maximum amplitude response after k iterations, denoted $T_k(\omega_1^*, \omega_2^*)$, is $T_k(\omega_1^*, \omega_2^*) = T_0^k(\omega_1^*, \omega_2^*)$. The superscript k represents raising to the power of k.

We consider the following three cases.

- Case 1: $T_0(\omega_1^*, \omega_2^*) > 1$.
  $T_k(\omega_1^*, \omega_2^*)$ increases exponentially with k. As a result, the frequency component of the image $I(\omega_1^*, \omega_2^*)$ will be amplified by an unbounded factor. The reverse filter is unstable.
- Case 2: $T_0(\omega_1^*, \omega_2^*) < 1$.
  $T_k(\omega_1^*, \omega_2^*)$ decreases exponentially with k, which approaches the limit of zero. As a result, the amplitude response for all frequencies will approach 0, i.e., $|H_k(\omega_1, \omega_2)| \to 0$, and the original image is recovered.
- Case 3: $T_0(\omega_1^*, \omega_2^*) = 1$.
  $|H_0(\omega_1, \omega_2)| < 1$ $(\omega_1 \neq \omega_1^*, \omega_2 \neq \omega_2^*)$ and $H_0(\omega_1^*, \omega_2^*) = 1$. The amplitude response decreases exponentially $|H_k(\omega_1, \omega_2)| \to 0$ $(\omega_1 \neq \omega_1^*, \omega_2 \neq \omega_2^*)$. As a result, the original image can be approximately recovered except at that particular frequency (ω_1^*, ω_2^*).

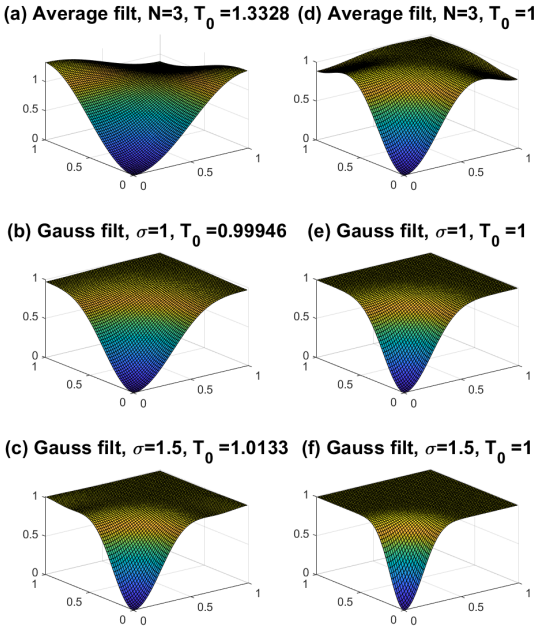


FIGURE 2. Amplitude response $|H_0|$ due to 3 linear low pass filters. Left column shows the case for the T-method where $H_0 = 1 - G$. Right column shows the case for the TDA-method where $H_0 = 1 - G^2$. The maximum value of $|H_0|$ is also shown as T_0.

To illustrate the role of $H_0(\omega_1, \omega_2)$ in determining the behavior of the two reverse filters, we consider three low pass filters:

- a (3×3) average filter,
- a Gaussian filter of standard deviation $\sigma = 1$, and
- a Gaussian filter of standard deviation $\sigma = 1.5$.

In Fig. 2, we plot $|H_0(\omega_1, \omega_2)|$ for $0 \leq \omega_1, \omega_2 \leq \pi$. In this figure, the two frequency axes are normalized for easy visualization. The left column shows the results for the T-method, while the right column shows the results of the TDA-method. We have the following observations and explanations.

- The T-method is unstable for a simple average filter. This is because the frequency response G can be negative at certain frequency band(s) leading to the amplitude response $|H_0| = |1 - G| > 1$ at those frequency band(s). For a symmetric Gaussian filters, theoretically

its frequency response is also a Gaussian, which is always positive under the assumption that the length of the filter is infinite. However, in actual implementation the length of the filter is usually set as $2 \times ceil(2\sigma) + 1$, where $ceil(x)$ is the ceiling function. The fixed length Gaussian filter can be regarded as a truncation of the ideal infinite length Gaussian filter. The truncation can create negative value in G leading to $|H_0| = |1 - G| > 1$. Experimentally we found that when $\sigma \leq 1$, $T_0(\omega_1^*, \omega_2^*) \leq 1$ which results in a stable filter and a prefect recovery of the original image. When $\sigma > 1$, $T_0(\omega_1^*, \omega_2^*) > 1$ which leads to an unstable filter.

- The proposed TDA-method is always stable for all three filters. This is because $0 \leq G^2 \leq 1$ and the maximum value of the amplitude response is less than 1, i.e.,

$$T_0(\omega_1^*, \omega_2^*) = |1 - G^2(\omega_1^*, \omega_2^*)| \leq 1 \tag{26}$$

- For stable filters, the quality of the recovered image depends on the range of the frequency $(\omega_1, \omega_2) \in \Omega$ such that $|H_k(\omega_1, \omega_2)| < 1$. More specifically, for the case $T_0(\omega_1^*, \omega_2^*) < 1$, if Ω covers the whole frequency plane $[0, \pi] \times [0, \pi]$, then the original image can be perfectly recovered. For the case $T_0(\omega_1^*, \omega_2^*) = 1$, if Ω covers more areas of the frequency plane, then the recovered image is of better quality because more frequency components of the image I are nulled. In addition, the number of iterations required to force $|H_k(\omega_1, \omega_2)|$ to approach zero for frequency $(\omega_1, \omega_2) \in \Omega$ depends on the shape of $|H_0(\omega_1, \omega_2)|$. For example, compared to Fig.2 (d) and (e), Fig.2(f) has a larger flat area of the frequency plane satisfying $|H_0(\omega_1, \omega_2)| \approx 1$. As such it will take more iterations to recover an image of better quality for the case of Fig.2(f).

The above analysis is supported by experimental results of an image (the image ''pepper.png'' in MATLAB) shown in Fig. 3. We used the above 3 linear filters to blur the original image and used the T-method and the proposed TDA-method to recover the original image. We use mean square error (MSE) to measure the quality of the recovered image as a function of the number of iterations. From Fig. 3(a-c), we can see that the T-method is not stable for the average filter and Gaussian filter with $\sigma = 1.5$. In fact, this is the case for any average filter and for a Gaussian filter with $\sigma > 1$. The T-method can recover the original image perfectly for a Gaussian with $\sigma \leq 1$, which can be attributed to $T_0(\omega_1^*, \omega_2^*) < 1$ for this family of filters.

From Fig. 3(d-f), we can see that the TDA-method is stable for all three filters. This is true for any average filter of any size and a Gaussian filter of any setting of σ. The required number of iterations to achieve a certain MSE depends on the filter kernel, which determines to what degree the image is smoothed. For example, we can see that for the lightly smoothed image (Fig. 3(d-e) the TDA-method requires about 2000 iterations to achieve a MSE of 10^{-5}, while for the heavily smoothed image (Fig. 3(f)) the TDA-method only achieves a MSE of 10^{-4} at 2000 iterations. Such observation

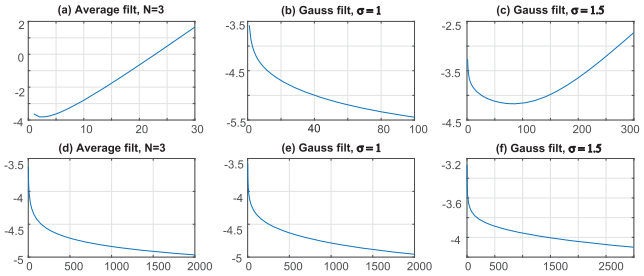


FIGURE 3. Mean square error (in $\log_{10}$ scale, vertical axis) of the recovered image as a function of number of iterations (horizontal axis). Top row shows the case for the T-method, while the bottom row shows the case for the TDA-method.

can be further explained in terms of the area of the frequency plane where the amplitude response $|H_0(\omega_1, \omega_2| \approx 1$. The larger the area such as Fig. 2(f), the more iterations are required to recover a good quality approximation of the original image.

C. CONVERGENCE FROM FIXED-POINT ITERATION POINT OF VIEW

In this section, we present some general results of the convergence of reverse filtering methods from a fixed-point iteration point of view. Specifically, a fixed point of $\mathcal{T}(x)$ is defined as $x^* = \mathcal{T}(x^*)$ which can be determined by the iteration $x_{k+1} = \mathcal{T}(x_k)$. To have a fixed-point, $\mathcal{T}$ must be a contraction mapping which is formally defined as the following. Let (X, d) be a complete metric space. Then a map $\mathcal{T} : X \to X$ is called a contraction mapping on X if there exists $q \in [0, 1)$ such that $d(\mathcal{T}(x), \mathcal{T}(y)) \leq qd(x, y)$ for all $x, y \in X$.

We can clearly see that the reverse filtering methods can be regarded as fixed-point iterations. The convergence of such methods can be theoretically studied from the contraction mapping perspective. For example, for the T-method we have $\mathcal{T}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{b} - g(\boldsymbol{x})$. Whether or not the mapping is a contraction depends on the filter g which can be complicated and does not have an analytical form. Therefore, we do not pursue such study further.

In addition, the fixed-point perspective allows us to study these methods from the point of view of solving a system of nonlinear equations, i.e., finding the solution of the filter equation $g(\boldsymbol{x}) = \boldsymbol{b}$. Specifically, let us assume there is a fixed-point $\boldsymbol{x}^* = \boldsymbol{x}_{k+1} = \boldsymbol{x}_k$, when the iteration converges. For the T-method, it is easy to see that $g(\boldsymbol{x}^*) = \boldsymbol{b}$ which means the fixed-point is a solution of the filter equation.

For the P-method, the fixed-point satisfies the equation

$$\frac{||h(\boldsymbol{x}^*)||(g(\boldsymbol{x}^* + h(\boldsymbol{x}^*)) - g(\boldsymbol{x}^* - h(\boldsymbol{x}^*))}{||g(\boldsymbol{x}^* + h(\boldsymbol{x}^*)) - g(\boldsymbol{x}^* - h(\boldsymbol{x}^*))||} = 0 \tag{27}$$

where $h(\boldsymbol{x}^*) = \boldsymbol{b} - g(\boldsymbol{x}^*)$. The above equation implies that $h(\boldsymbol{x}^*) = 0$ leading to $g(\boldsymbol{x}^*) = \boldsymbol{b}$ which is also a solution of filter equation.

For the TDA-method, the fixed-point satisfies the equation: $g(\boldsymbol{x}^* + \boldsymbol{b} - g(\boldsymbol{x}^*)) = g(\boldsymbol{x}^*)$. In order for the fixed-point to be a solution of the filter equation, it requires a further assumption

that the filter is injective: if $g(\boldsymbol{x}) = g(\boldsymbol{y})$, then $\boldsymbol{x} = \boldsymbol{y}$. The requirement is filter dependent and is a limitation of the TDA-method compared to the other two methods which do not require such an assumption.

However, from a mathematical point of view, the fixed-point that satisfies the filter equation may not be the same as the original image. This is because there could be multiple solutions of the filter equation. For the fixed-point $\boldsymbol{x}^*$ to be the same as the original image denoted $\boldsymbol{x}_o$, a sufficient condition is that the filter is injective such that $g(\boldsymbol{x}^*) = g(\boldsymbol{x}_o)$ leads to $\boldsymbol{x}^* = \boldsymbol{x}_o$. From this point of view, all three methods require the injective assumption to theoretically guarantee the recovery of the original image. We point out that it is beyond the scope of this paper to further discuss the following issues. For a particular filter whether or not the filter equation has a unique solution. If not, how to find a solution which is close to the "desired" image. We must also point out that our extensive experiments demonstrate that although many nonlinear filters do not satisfy the injective condition, these methods still produce good results.

D. RELATION AND COMPLEXITY

We can see in Table 1 that the T-method is a special case of the R-method and is the simplest of all methods. When the filter $g(.)$ is linear, we have $g(\boldsymbol{x}_k + \boldsymbol{q}_k) - g(\boldsymbol{x}_k) = g(\boldsymbol{q}_k)$. Thus, compared to the T-method, the TDA-method has a further step of filtering of $\boldsymbol{q}_k$ and has a scale factor λ.

In addition, we can rewrite the TDA method as follows

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{b} - g(\boldsymbol{x}_k) + g(\boldsymbol{y}_{k+1}) - \boldsymbol{b} \tag{28}$$

$$= \boldsymbol{y}_{k+1} + g(\boldsymbol{y}_{k+1}) - \boldsymbol{b} \tag{29}$$

where $\boldsymbol{y}_{k+1} = T(\boldsymbol{x}_k) = \boldsymbol{x}_k + \boldsymbol{b} - g(\boldsymbol{x}_k)$. The TDA-method can then be represented as a two-step process as follows

$$\text{T-step:} \quad \boldsymbol{y}_{k+1} = T(\boldsymbol{x}_k) \tag{30}$$

and

$$\text{C-step:} \quad \boldsymbol{x}_{k+1} = \boldsymbol{y}_{k+1} + \boldsymbol{e}_{k+1} \tag{31}$$

where $\boldsymbol{e}_{k+1} = g(\boldsymbol{y}_{k+1}) - \boldsymbol{b}$. The C-step adds the error $(\boldsymbol{e}_{k+1})$ due to the $(k+1)$th T-step back to correct the result $\boldsymbol{y}_{k+1}$ and uses it as a new estimate of $\boldsymbol{x}_{k+1}$. The term $\boldsymbol{e}_{k+1}$ is an error because the goal in this case is to estimate a signal $\hat{\boldsymbol{x}}$ such that $g(\hat{\boldsymbol{x}}) \approx \boldsymbol{b}$ or a certain measurement of the difference $g(\hat{\boldsymbol{x}}) - \boldsymbol{b}$ is as small as possible. The result at $(k+1)$th T-step is $\boldsymbol{y}_{k+1}$ and the error is thus $\boldsymbol{e}_{k+1}$. Therefore, the TDA-method can be regarded as a generalization of the T-method with an extra correction step.

The P-method and the TDA-method are related through the different approximations for the gradient. The computationally expensive ratio of matrix norms in the P-method can be regarded as playing the role of a scale parameter λ in the TDA-method.

To illustrate their difference in computational complexity, a MATLAB implementation of the 3 methods is used to reverse a Gaussian filter with a kernel size of 7×7 pixels and $\sigma = 1$. Running time for one iteration vs image size is presented in Fig. 4. The P-method is significantly slower than the T- and TDA-method. The difference in running time of the T- and TDA-method is due to number of Gaussian filters per iteration. The complexity is summarized in the last two columns of Table1.

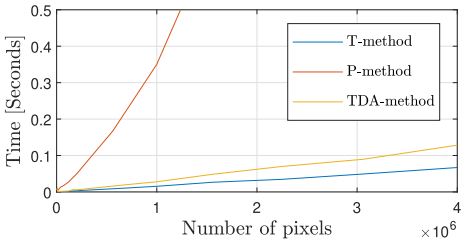


FIGURE 4. Comparison of running time vs image size when reversing a Gaussian filter with a kernel size of 7 × 7 pixels and a standard deviation $\sigma = 1$ applying one iteration of T-, P- and TDA-method.

TABLE 2. AGD methods.

Gradient descent	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \lambda \nabla f(\boldsymbol{x}_k)$
MGD [36]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{v}_k$ $\boldsymbol{v}_k = \beta \boldsymbol{v}_{k-1} - \lambda \nabla f(\boldsymbol{x}_k)$
NAG [37]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{v}_k$ $\boldsymbol{v}_k = \beta \boldsymbol{v}_{k-1} - \lambda \nabla f(\boldsymbol{x}_k + \beta \boldsymbol{v}_{k-1})$
RMSProp [38]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \frac{\lambda}{\sqrt{\boldsymbol{v}_k + \epsilon}} \nabla f(\boldsymbol{x}_k)$ $\boldsymbol{v}_k = \beta \boldsymbol{v}_{k-1} + (1-\beta) \nabla f(\boldsymbol{x}_k)^2$
Adadelta [39]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \Delta \boldsymbol{x}_k$ $\Delta \boldsymbol{x}_k = \frac{\sqrt{\boldsymbol{u}_{k-1} + \epsilon}}{\sqrt{\boldsymbol{v}_k + \epsilon}} \nabla f(\boldsymbol{x}_k)$ $\boldsymbol{v}_k = \beta \boldsymbol{v}_{k-1} + (1-\beta) \nabla f(\boldsymbol{x}_k)^2$ $\boldsymbol{u}_k = \beta \boldsymbol{u}_{k-1} + (1-\beta) \Delta \boldsymbol{x}_k^2$
ADAM [40]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \lambda \frac{\hat{\boldsymbol{m}}_k}{\sqrt{\hat{\boldsymbol{v}}_k + \epsilon}}$ $\boldsymbol{m}_k = \beta_1 \boldsymbol{m}_{k-1} + (1-\beta_1) \nabla f(\boldsymbol{x}_k)$ $\boldsymbol{v}_k = \beta_2 \boldsymbol{v}_{k-1} + (1-\beta_2) \nabla f(\boldsymbol{x}_k)^2$ $\hat{\boldsymbol{m}}_k = \frac{\boldsymbol{m}_k}{1-\beta_1}, \hat{\boldsymbol{v}}_k = \frac{\boldsymbol{v}_k}{1-\beta_2}$

E. USING ACCELERATED GRADIENT DESCENT

Accelerated gradient descent (AGD) methods [26], which are summarised in Table 2, were developed to tackle problems related to gradient descent. They are widely used for training deep neural networks and have the potential to reduce the number of iterations needed to achieve convergence and to increase the robustness against noisy gradients [27]. Since the T-, P- and the proposed TDA-method can be regarded as gradient descent algorithms, we test these AGD methods to see if they can improve the performance of the three algorithms. The tests are conducted by replacing the gradient

of each AGD method with the following approximations and results are presented in section IV-C.

- TDA-method: $\nabla f(\boldsymbol{x_k}) = -\boldsymbol{t}_k$
- T-method: $\nabla f(\boldsymbol{x_k}) = -\boldsymbol{q}_k$
- P-method: $\nabla f(\boldsymbol{x_k}) = -\frac{||\boldsymbol{q}_k||}{2||\boldsymbol{p}_k||}\boldsymbol{p}_k$

IV. EXPERIMENTS AND RESULTS

In this section, we present an experimental study of the proposed TDA-method, including

- effects of setting the learning rate λ and the number of iterations (section IV-A),
- validation of its effectiveness in reversing a wide range of filters (section IV-B),
- a comparison with the four existing methods (section IV-B2),
- an experimental study (section IV-C) of using accelerated gradient descent algorithms, and
- a demonstration of the limitation of all current methods failing to reverse the effect of a median filter (section IV-D).

When it is not specified, the running time test is conducted using MATLAB r2021a running in a PC with Intel i7-3930k CPU and 40GB RAM.

A. NUMBER OF ITERATIONS, LEARNING RATE AND IMAGE QUALITY

We present two case studies in which the TDA-method is convergent and non-convergent, respectively.

1) CONVERGENT CASE

We use a successful example of reversing the effect of a bilateral filter (BF) [7] to demonstrate the relationship. Results are shown in Fig. 5 where the smoothed image is produced by using range and spatial parameters $\sigma_r = 0.3$ and $\sigma_s = 4$. We can see that the proposed TDA-method is convergent and details from the original image are gradually restored as the number of iterations increases. In this figure, we also show the importance of setting the learning rate. A larger value leads to a faster improvement in image quality.

2) NON-CONVERGENT CASE

We have shown in last section, the condition for proposed TDA-method to be stable in the linear filter case. However, there is no guarantee that the proposed TDA-method, like previous works, will be convergent to a satisfactory result for any filters. Therefore, a practical problem is to determine the criteria for stopping the iteration at a point where the quality of the output image is the highest.

How do we quantify the output image quality without knowing the original image? Since we have the input image $\boldsymbol{b} = g(\boldsymbol{x})$ and the filter $g(.)$ as a black box, we can calculate the following relative error:

$$e_k = \frac{||\boldsymbol{b} - g(\boldsymbol{x}_k)||_2^2}{||\boldsymbol{b}||_2^2} \tag{32}$$

If $||\boldsymbol{b} - g(\boldsymbol{x}_k)||_2^2$ is small, then under the bi-Lipschitz condition[1] we can expect that $||\boldsymbol{x} - \boldsymbol{x}_k||_2^2$ will also be small. Thus, the relative error is a non-referenced metric which can be used to determine the stopping point of the iteration to achieve the best outcome.

Indeed, we can use a two-pass process to determine the optimum stopping point. In the first pass, we run the algorithm for a fixed number of iterations, record the sequence $\{e_k\}$, and find n such that $n = \min_k\{e_k\}$. In the second pass, we run the algorithm again for n iterations to determine the best outcome. We remark that image quality assessment in general and non-reference quality assessment in particular is an active research area [41]. Incorporation results from this research area to deal with the stopping of the iteration is beyond the scope of this work.

In addition, it is a common practice in evaluating image processing algorithms that the original image is assumed to be known so that metrics such as peak-signal-to-noise ratio (PNSR) and structural similarity index (SSIM) can be used. Compared with the relative error, both PSNR and SSIM are referenced metrics. They are used to evaluate the performance of the algorithm, especially in comparison of performance of different algorithms.

In Fig.6, we present a non-convergent example. We use the above 3 metrics in evaluating the performance of reversing a RGF [21] by the TDA-method with $\lambda = 1$. The smoothed image shown in Fig.6(e) is produced by 4 iterations of a bilateral filter with $\sigma_s = 3$ and $\sigma_r = 0.05$. We can see that the TDA-method keeps improving output image quality as measured by the three metrics up to a point. After that, the output image quality is getting worse. This is revealed in the top row of the figure, where there is an optimal number of iteration for each metric. We want to point out that the optimal number of iterations determined by the relative error ($k = 57$) is fairly close to those determined by using PSNR and SSIM ($k = 53$ and $k = 72$, respectively). Thus, this experiment supports the use of the relative error in determining the number of iterations. In the bottom row of Fig.6, we can visually compare the results of 500 iterations and 57 iterations. The former (Fig.6(f)) has strong ringing artifacts, while the latter (Fig.6(g)) is free of such artifacts and restores an acceptable level of details which can be found in the original image.

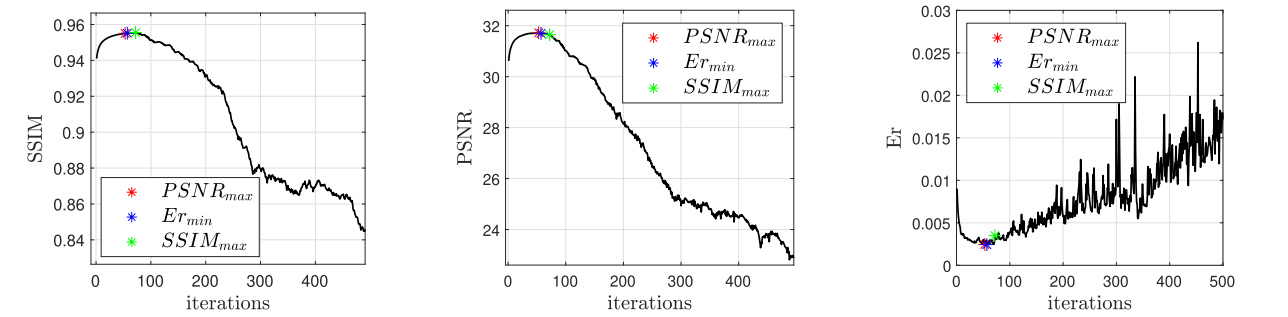
3) SUMMARY

When the proposed TDA-method converges, setting a larger value of λ helps speed up the convergence rate. When it does not converge to a useful result, we can use the relative error to stop the iteration. Adaptively changing the learning rate will be discussed in section IV-C.

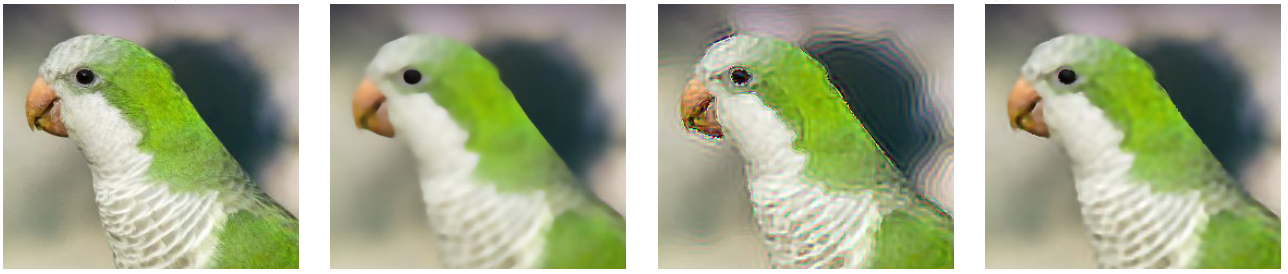
[1] We regard the filter $g(.)$ as a function. When it is bi-Lipschitz [30], it satisfies the condition $\frac{1}{K}d_X(x_1, x_2) \le d_Y(y_1, y_2) \le Kd_X(x_1, x_2)$. Here the function is defined as $y = g(x)$, X and Y are sets for x and y, $K > 1$, and d_X and d_Y are metric on sets X and Y, respectively. An equivalent definition is that both g and its inverse g^{-1} are Lipschitz.

(a) Ground truth (b) Smoothed (c) $N_{iter} = 5$ (d) $N_{iter} = 100$ (e) $N_{iter} = 500$ (f) PSNR vs Iter

FIGURE 5. Results of reversing a bilateral filter using the TDA-method with λ = 1. (a) Original image. (b) Smoothed image with BF (25.3dB). (c) Result after 5 iterations (29.9dB). (d) Result after 100 iterations (37.0dB). (e) Result after 500 iterations (40.9dB). (f) PSNR as a function of N_{iter} for three settings of λ.



(a) SSIM (b) PSNR (c) Relative error



(d) Original image (e) Input image (f) $N_{iter} = 500$ (g) $N_{iter} = 57$

FIGURE 6. Example of finding the optimum number of iterations when reversing a RGF [21] as non-convergent example with the TDA-method. (a) SSIM per iteration. (b) PSNR per iteration. (c) Relative error per iteration, (d) original image. (e) Smoothed image (30.22dB). (f) Result after 500 iterations (22.93dB). (g) Result after 57 iterations (31.70dB).

B. EVALUATION AND COMPARISONS

We first evaluate the effectiveness of the TDA-method by comparing its performance in reversing 12 commonly used filters with that of T- and P-method. We then study three particular cases to highlight and visualize the performance of the TDA-method. We do not make experimental comparison with methods proposed in reference [34], because these methods assume that the black box filter $g(.)$ is locally linear. This assumption may not be true for highly nonlinear filters.

1) EVALUATION BASED ON 12 FILTERS

We performed experiments using 300 natural images from the BSD300 dataset [42]. We filtered each of them using one of the 12 commonly used image filters listed in Table 3. We then applied the TDA-method of 200 iterations to restore the original image. Since different filters produce different levels of degradation, we use the percentage of improvement of PSNR given by the following equation to compare the performance for different filters

$$\bar{p}_k = \frac{p_k - p_0}{p_0} \times 100 \tag{33}$$

where p_k and p_0 represent respectively the PSNR value for a filter at the kth iteration and 0th iteration (the input image). The results for the 300 images are then averaged and their standard deviations are calculated. We also used the SSIM directly to measure the output image quality at an iteration point for each filter. The result is summarized as average values and standard deviations over the 300 images.

The PSNR results for the TDA-method are shown in Fig. 7. Sub-figures (a) and (b) show the results using $\lambda = 0.5$ and $\lambda = 1$, respectively. We can see that the performance of the

TABLE 3. Parameter settings for the 12 filters.

Filter	Parameters
RGF [21] Rolling guidance filter	$\sigma_s = 3, \quad \sigma_r = 0.05, \quad N_{iter} = 4$
Gauss. Gaussian low pass filter	$\sigma = 5$
LoG Laplacian of Gaussian	$k = 7 \times 7, \quad \sigma = 0.4$
AMF [22]	$\sigma_s = 7, \quad \sigma_r = 0.4$
RTV [23] Relative total variation	$\lambda = 0.05, \quad \sigma = 3, \quad \epsilon = 0.05, \quad N_{iter} = 2$
ILS [24] Iterative least squares	$\lambda = 1, \quad p = 0.8, \quad \epsilon = 1 \times 10^{-4}, \quad N_{iter} = 4$
L0 [25] L_0 norm smoothing	$\lambda = 0.01, \quad \kappa = 2$
BF [7] Bilateral filter	$\epsilon = 0.05, \quad \sigma_s = 3$
Disk averaging filter	$r = 3$
Motion blur filter	$l = 20, \quad \theta = 45°$
GF [8] Guided filter	$w_{size} = 5 \times 5, \quad \epsilon = 0.1$
GF+Gauss.	$w_{size} = 5 \times 5, \quad \epsilon = 0.1, \quad \sigma = 5$

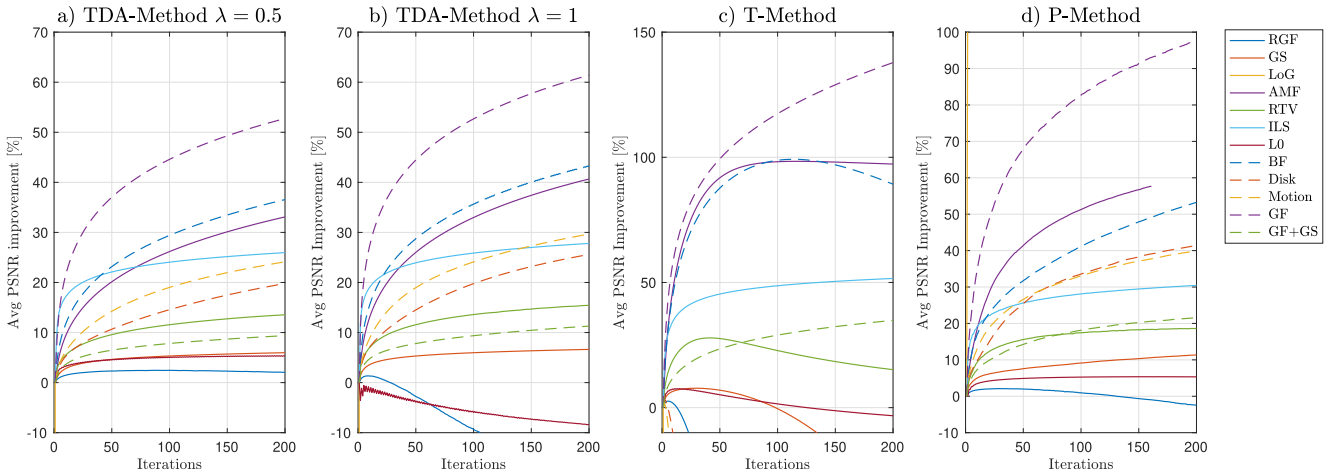


FIGURE 7. Improvement of the PSNR per iteration when reversing 12 filters. The result is the average of the PSNR improvements for 300 images in the BSD300 dataset [42]. (a) TDA-method setting $\lambda = 0.5$. (b) TDA-method setting $\lambda = 1$. (c) T-method. (d) P-method.

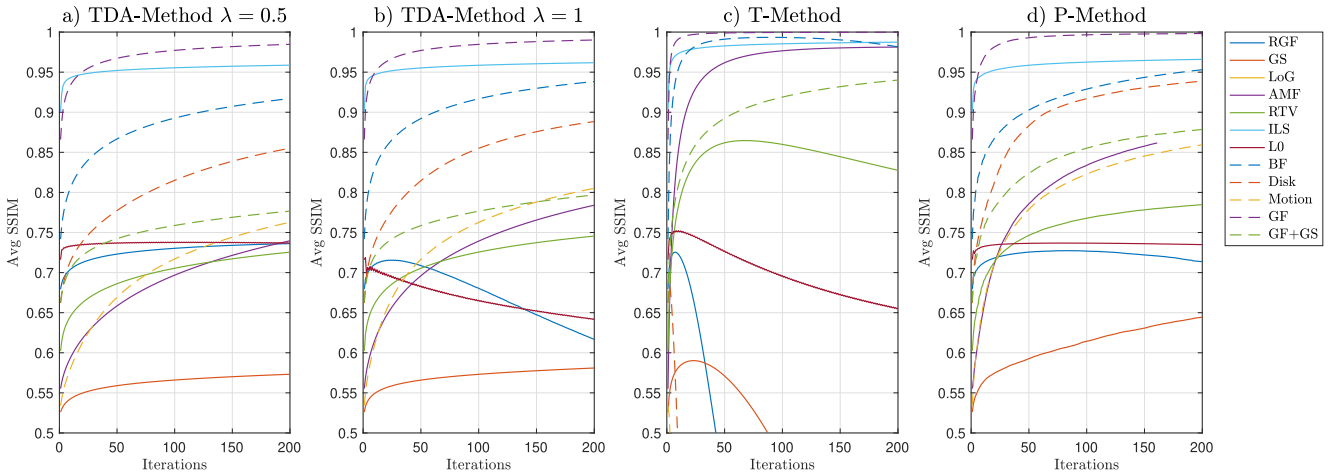
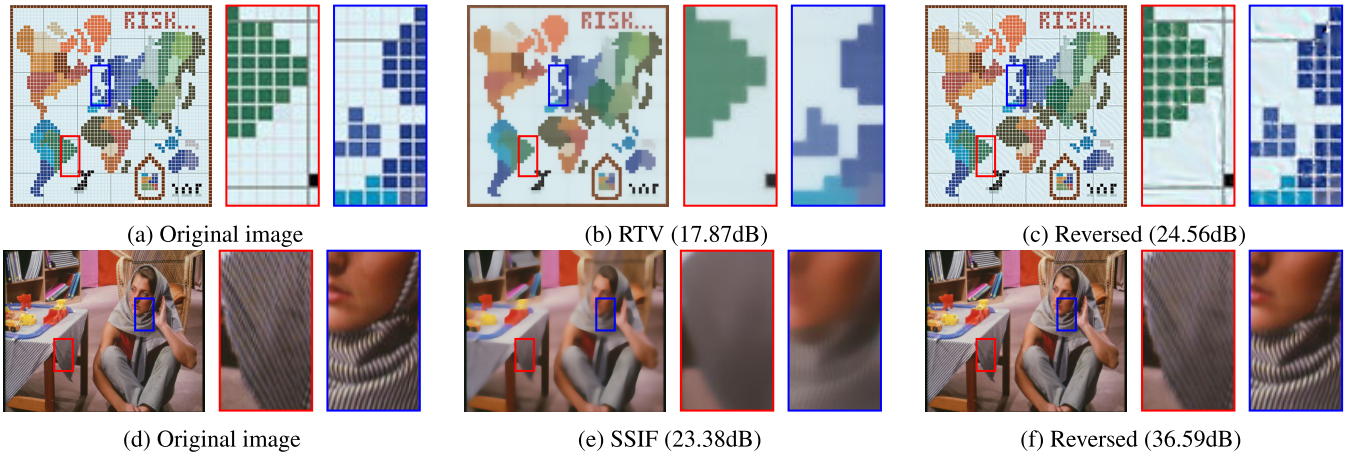


FIGURE 8. SSIM per iteration when reversing 12 filters. The result is the average of SSIM value for 300 images in the BSD300 dataset [42]. (a) TDA-method setting $\lambda = 0.5$. (b) TDA-method setting $\lambda = 1$. (c) T-method. (d) P-method.

TDA-method depends on the filter being reversed. While it can achieve a substantial improvement for some filters such as GF and BF after a few iterations, it requires a lot of iterations for filters such as Gaussian filter and rolling guided filter. Setting $\lambda = 0.5$ produces acceptable average improvements. Setting $\lambda = 1$ achieves faster improvement in image quality for most filters. However such a setting does not work in reversing the L_0 filter. Overall, after 200 iterations, for 6 out of the 12 filters tested for both settings of λ, the TDA-method achieved quality improvement greater than 10%. For $\lambda = 0.5$, improvements have been achieved for all 12 filters after 200 iterations, although some improvements are quite small.

TABLE 4. Average PSNR improvement (Avg.PSNR Impr.) and Standard deviation (SD) after 200 iterations on BSD300 dataset. Entries in red color indicate that the reverse filter method fails.

	TDA ($\lambda = 0.5$)		TDA ($\lambda = 1$)		T-method		P-method	
	Avg.PSNR Impr.[%]	SD	Avg.PSNR Impr.[%]	SD	Avg.PSNR Impr.[%]	SD	Avg.PSNR Impr.[%]	SD
RGF [21]	2.1	1.5	-19.5	8.3	-77.7	2.9	-2.5	6.2
Gauss.	6	2.3	6.6	2.5	-38	10.6	11.3	3.8
LoG	NaN	NaN	NaN	NaN	NaN	NaN	255.7	1920.3
AMF [22]	33.1	20.2	40.7	24.2	97.3	36	NaN	NaN
RTV [23]	13.5	5.4	15.4	5.8	15.2	9.1	18.6	6.6
ILS [24]	26	8.2	27.8	9	51.6	14.4	30.4	10.6
L0 [25]	5.3	2.3	-8.4	3.2	-3.3	4.3	5.3	2.2
BF [7]	36.6	15	43.3	17.5	89.3	49.6	53.2	23.9
Disk	19.8	2.8	25.6	3.5	-703.2	89.8	41.4	11.8
Motion	24.1	4.1	29.7	4.8	-1503	185.8	40	9.9
GF [8]	52.7	20.1	61.4	23.7	137.8	57.5	97.9	49.5
GF+Gauss.	9.4	2.8	11.3	3.2	34.9	7.5	21.6	5.4



(a) Original image (b) RTV (17.87dB) (c) Reversed (24.56dB)

(d) Original image (e) SSIF (23.38dB) (f) Reversed (36.59dB)

FIGURE 9. Recovering small-scale image texture by reversing detail removal filters. (a) Original image 1. (b) Filtered image with RTV ($\lambda = 0.05, \sigma = 3, \epsilon = 0.05, N_{iter} = 2$). (c) Reversed with TDA-method ($\lambda = 1, N_{iter} = 3000$). (d) Original image 2. (e) Filtered image with SSIF ($r = 5, \kappa = 0.1, \epsilon = 0.1, N_{iter} = 2$). (f) Reversed with TDA-method ($\lambda = 1, N_{iter} = 500$).

To compare the performance of TDA-method with those of the T- and P-method, we conducted the same experiment and presented the results in sub-figures (c) (T-method) and (d) (P-method). The T-method produces very good results for 5 filters (>20% PSNR improvement), but it also fails to reverse 5 filters. The P-method, like the TDA-method, is able to reverse all the filters to some extent but at a very high computational cost. Therefore, the proposed TDA method is better than the T-method in terms of its ability to reverse a wider variety of filters, and is also better than the P-method in terms of its much lower computational cost.

We also report the average SSIM values per iteration for each method in Fig.8. These results agree with those on Fig.7. In addition, we summarize all results at the 200th iteration in Tables 4 (average improvement in PSNR over 300 images and standard deviation) and 5 (average SSIM over 300 images and the standard deviation).

We notice that in Table 4, a negative or NAN value for average PSNR improvement means the method fails to reverse the effect of the filter. A higher average value means a better quality of the restored image. The variance provides information about fluctuation of image quality improvement within the image dataset. A smaller value means the performance of the reverse filter is less affected by content of the image when the number of iteration is fixed. We have a similar interpretation of of the SSIM result presented in Table 4. The SSIM value is in the interval [0, 1]. When the average SSIM value is close to 1, the reverse filter produces output image of better quality. When the average SSIM value is close to 0, the reverse filter fails to restore the image. Comparing values of these two tables, we can see that they provide consistent information about the performance of each reversing filters.

2) SUBJECTIVE AND OBJECTIVE COMPARISON EXAMPLES

We present details of three examples for comparing different methods subjectively through visualization and objectively through PSNR. The purpose of the 1st example is to visualize the performance of the TDA-method in reversing two highly nonlinear filters. The 2nd example provides further results the comparing the TDA-method with 4 current methods. The last example demonstrates the effectiveness of the TDA-method in image restoration by comparing it with some classical blind and non-blind algorithms.

Example 1: An image filter can sometimes irreversibly remove details from an image. However, in some cases, those details are not completely removed, but are just diminished at a point of being visually imperceptible. In this example, we demonstrate the effectiveness of the TDA-method in recovering the texture information which is imperceptible after the image is smoothed by the RTV algorithm [23] and the SSIF algorithm [43]. These two algorithms are highly nonlinear and are used to smooth out texture. Fig. 9 show that the texture information has been recovered to some extent in both images. To produce the results shown in Figures 9(c) and (f) we used 3000 and 500 iterations of the TDA-method, respectively.

Example 2: We compare the performance of the TDA-method with that of the 4 methods described in section II in reversing a self-guided filter [8]. This filter is chosen because it can be reversed by all methods. The filter was configured to produce a texture smoothing effect by setting the window size to 15×15 pixels and $\epsilon = 0.01$. The ground truth and filtered image are shown in Figures 12 (a) and (b). We can see that textures have been smoothed while the main structure of the image is preserved. Figures 12 (c) to (f) show the result for each method. Although the difference between results of all methods is quite small in terms of visual inspection, the TDA-method has achieved the best improvement in PSNR.

Example 3: We study the performance of the TDA-method in de-convolution applications [1]. When the filter kernel is known, the problem is non-blind. Classical non-blind image restoration algorithms, which require the knowledge of the kernel, include: Lucy Richardson algorithm (LRA) [45], fast image de-convolution using hyper-Laplacian priors (HLP) [46], and Wiener filter (WF) [47]. A well-known blind restoration algorithm is the maximum likelihood algorithm (MLA) [44], which does not require the knowledge of the kernel. The proposed TDA-method is applicable in a situation where the kernel is unknown but is available as black-box. As such, it can be regarded as semi-blind. We should point out that there is a vast literature on the subject of image restoration, we only pick some classical methods in our experiments.

In Fig. 10 we show an example of smoothing using a Gaussian kernel of size 11×11 pixels and a standard deviation $\sigma = 10$. In Fig. 11 we repeat the same experiment using a kernel which models the blurring due to the linear motion of a camera by 20 pixels with an angle of 10 degrees in a counter-clockwise direction. In both cases, the TDA-method can reverse the effect of the linear filter, small details are recovered and edges are well defined (refer to the green box on Figures 10 and 11). The TDA-method clearly produces a more appealing result than MLA and HLP since it does not produce artifacts and the output image is sharper. The LRA algorithm produces very good results but requires 10000 iterations to achieve the result.[2] The Wiener filter is the winner among all methods. However, it requires the blurring kernel as the input. The impact of not knowing the exact kernel can be seen in the relatively inferior results of the MLA algorithm in both cases. Since the exact knowledge of the kernel may not be available in practice, the proposed method can be useful because it attempts to reverse the effect of any available filter as a black box without the need to know its internal operations.

[2]It takes more than 10 minutes on a MacBook Pro computer with 16GB RAM and a M1 Pro CPU.

TABLE 6. AGD methods, parameter settings.

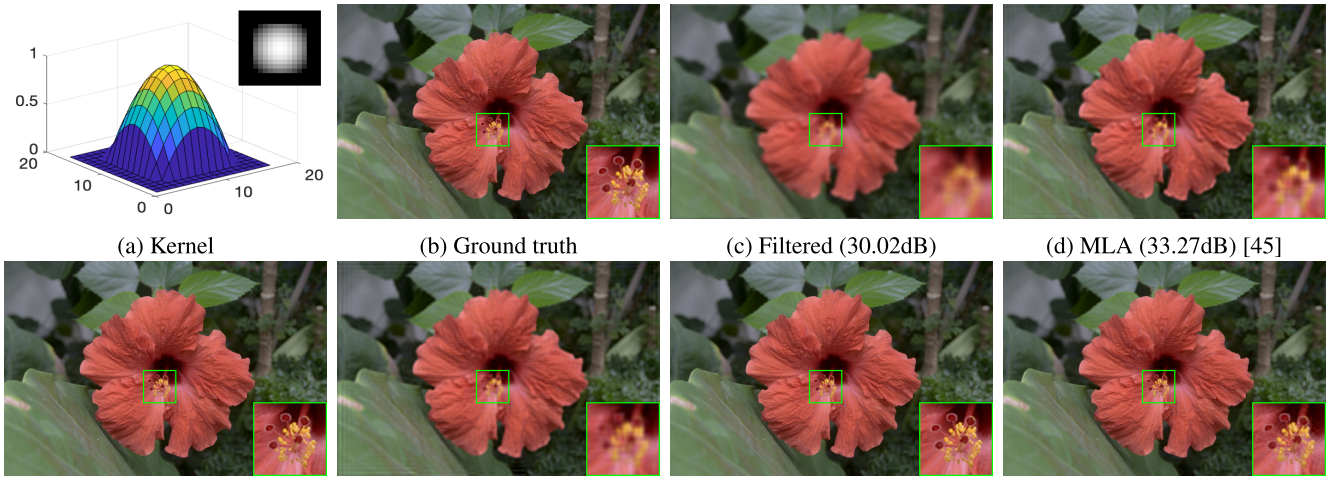
Method	Parameter
GD	$\lambda = 1$
MGD	$\lambda = 1, \quad \beta = 0.9, \quad v_0 = 0$
NAG	$\lambda = 1, \quad \beta = 0.9, \quad v_0 = 0$
RMSprop	$\lambda = 1, \quad \beta = 0.9, \quad v_0 = 0$
ADAM	$\lambda = .1, \quad m_0 = 0, \quad v_0 = 0, \quad \beta_1 = 0.9, \quad \beta_2 = 0.999$
Adadelta	$\lambda = 1, \quad \beta = 0.9, \quad v_0 = 0, \quad u_0 = 0$

C. RESULTS OF USING ACCELERATED GRADIENT DESCENT METHODS

In this section, we present a study of applying AGD methods to the three reverse filtering algorithms: the proposed TDA-method, the T-method and the P-method. We summarize the parameter setting of each AGD method in Table 6. We aim to study the following questions.

Question 1. When the filter can be reversed, how is the performance of the reverse filter changed by each of the AGD methods? We conduct an experiment in which all 3 reverse filter methods can successfully recover the original image. We smooth the *"cameraman"* image using a Gaussian filter with a kernel size of 7×7 and a standard deviation $\sigma = 1$. The three reverse filtering methods and its corresponding AGD variants are then applied to process the image and the PSNR values calculated at each iteration are recorded. Results are presented in Fig. 13. Original methods (without AGD and referred to as GD) are represented by thick dashed black lines to simplify the comparison. We can clearly see that for the T-method, applying the AGD methods of MGD, NAG and ADAM results in significant improvement, while applying the other two AGD methods, RMSprop and Adadelta, does not produce notable improvement. For the P-method, there is no improvement when using any one of the AGD methods. Some AGD methods even have a negative impact on its performance. For the TDA-method, both MGD and NAG results in notable improvement, while the other AGD methods lead to roughly the same performance as that of without using AGD. We note that the superior performance of the T-method for this particular case has been explained in terms of the frequency response of the iterative filter in section III-B2 and is demonstrated in figure 3

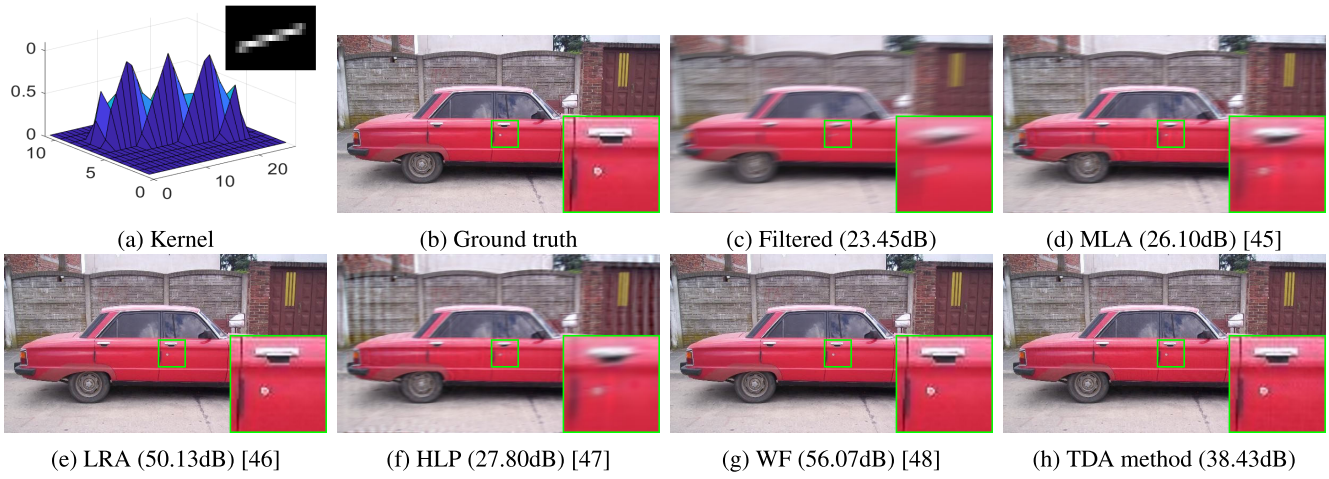
Question 2. When the filter cannot be reversed by a particular method because the iteration is an unstable process, does the AGD help to stabilize the iteration? We conduct an experiment which is aimed at reversing a motion blur filter. An image shown in Fig. 15(a) was smoothed by a filter which



(a) Kernel (b) Ground truth (c) Filtered (30.02dB) (d) MLA (33.27dB) [45]

(e) LRA (49.93dB) [46] (f) HLP (33.87dB) [47] (g) WF (56.39dB) [48] (h) TDA method (42.19dB)

FIGURE 10. Reversing a gaussian filter of size 11 × 11 and $\sigma = 10$. (a) Kernel (Re-scaled for better visualization). (b) Original image. (c) Filtered image. (d) MLA. (e) LRA. (f) HLP. (g) Wiener filter. (h) TDA-method.



(a) Kernel (b) Ground truth (c) Filtered (23.45dB) (d) MLA (26.10dB) [45]

(e) LRA (50.13dB) [46] (f) HLP (27.80dB) [47] (g) WF (56.07dB) [48] (h) TDA method (38.43dB)

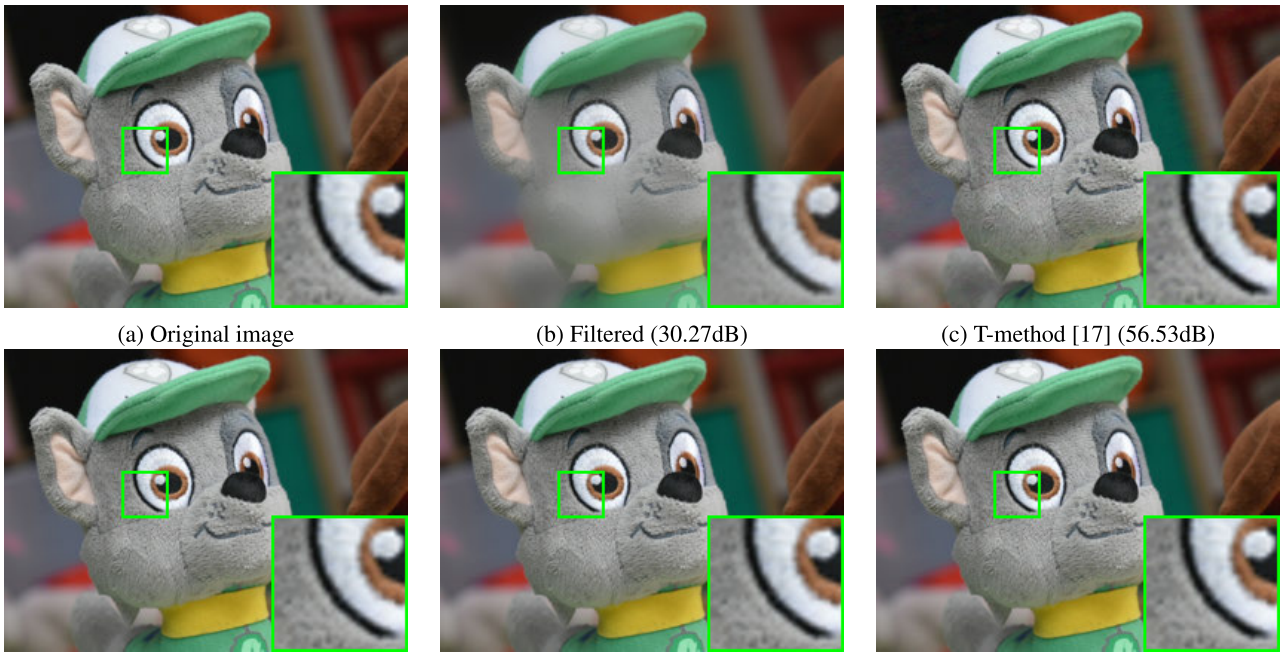
FIGURE 11. Reversing a motion blur filter. (a) Kernel (Re-scaled for better visualization). (b) Original image. (c) Filtered image. (d) MLA. (e) LRA. (f) HLP. (g) Wiener filter. (h) TDA-method.

TABLE 5. Average SSIM value and standard deviation (SD) after 200 iterations on BSD300 dataset. Entries in red color indicate that the reverse filter method fails.

	TDA ($\lambda = 0.5$)		TDA ($\lambda = 1$)		T-method		P-method	
	Avg. SSIM	SD	Avg. SSIM	SD	Avg. SSIM	SD	Avg. SSIM	SD
RGF [21]	0.74	0.12	0.62	0.14	0.02	0.01	0.62	0.14
Gauss.	0.57	0.15	0.58	0.15	0.18	0.17	0.58	0.15
LoG	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
AMF [22]	0.74	0.14	0.78	0.13	0.98	0.02	0.78	0.13
RTV [23]	0.73	0.12	0.75	0.11	0.83	0.05	0.75	0.11
ILS [24]	0.96	0.03	0.96	0.02	0.99	0.01	0.96	0.02
L0 [25]	0.74	0.09	0.64	0.1	0.66	0.1	0.64	0.1
BF [7]	0.92	0.05	0.94	0.04	0.98	0.02	0.94	0.04
Disk	0.86	0.06	0.89	0.04	0	0	0.89	0.04
Motion	0.76	0.07	0.81	0.06	0	0	0.81	0.06
GF [8]	0.98	0.01	0.99	0.01	1	0	0.99	0.01
GF+Gauss.	0.78	0.1	0.8	0.09	0.94	0.03	0.8	0.09

approximates a linear motion of 20 pixels in a 45 degrees angle. Results are shown in Fig. 14. The T-method fails to recover the image and none of the AGD methods helps to make the T-method produce useful results.

Question 3. When the T-method fails, how does the AGD help improve the performance of P- and TDA-method? We can see in Fig. 14 that the performance of the P-method and T-method are improved by using MGD, NAG and ADAM.



(a) Original image (b) Filtered (30.27dB) (c) T-method [17] (56.53dB)

(d) F-method [20] (43.41dB) (e) P-method [19] (56.46dB) (f) TDA-method (57.48dB)

FIGURE 12. Reversing the guided filter [8]. (a) Original image. (b) Filtered or input image. (c) T-method ($N_{iter} = 10$). (d) F-method ($N_{iter} = 20$). (e) P-method ($N_{iter} = 500$). (f) Proposed TDA-method ($N_{iter} = 500$).

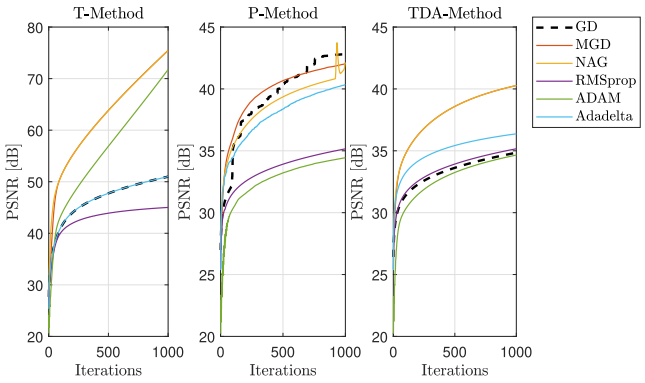


FIGURE 13. PSNR per iteration when reversing Gaussian filter with T-, P- and TDA-method applying accelerated gradient descent.

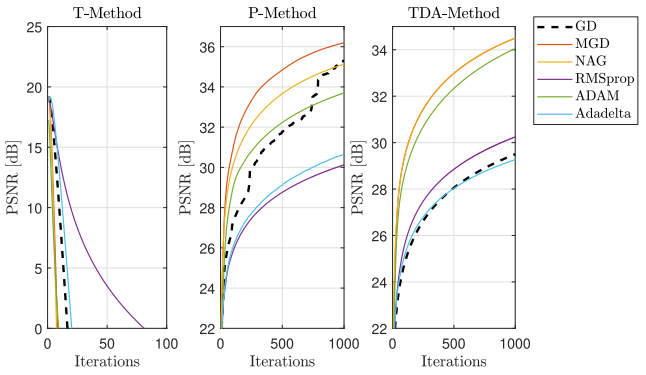


FIGURE 14. PSNR per iteration when reversing motion blur filter using T-, P- and TDA-method with accelerated gradient descent.

To provide further evidence of the improvement and to further compare these two methods, we present in Fig. 15 results after 1000 iterations of algorithms with and without AGD. It can be seen that the PSNR improved by about 5dB when the TDA-method is combined with an AGD method, while that of the P-method is improved by about 1dB. However, in 1000 iterations the P-method produces 36.54dB, while the TDA-method produces 34.76dB. We should point out that due to the huge difference in complexity, to complete 1000 iterations, the TDA-method and the P-method take about 3.27 seconds and 43.57 seconds, respectively. For the TDA-method with NAG to produce the result of 36.5dB, it takes only 14.60 seconds to complete 3048 iterations. This is about 1/3 of the time which the P-method needs to produce the same result. As a further comparison, in 14.60 seconds the P-method could only perform 335 iterations achieving a PSNR of 30.72dB. Therefore, compared with the P-method, the TDA-method not only benefited more from the AGD, but is also faster due to its much lower complexity.

Question 4. For the three methods, what are the benefits when using AGD over a wide range of filters to be reversed? This is a further study of that presented in IV-B1 by comparing the performance of the 3 methods with AGD. We conducted experiments using the ‘‘cameraman’’ image which is smoothed by 11 filters mentioned in section IV-B1. We have excluded the LoG filter because all methods with AGD failed to reverse this filter. We recorded the PSNR value of the image produced by each filter and the PSNR

(a) Original (b) Blurred (18.84dB)
(c) P w/o AGD (35.56dB) (d) TDA w/o AGD (29.71dB)
(e) P w/ MGD (36.45dB) (f) TDA w/ NAG (34.76dB)

FIGURE 15. Example of reversing a motion blur filter with P-method and TDA-method by applying accelerated gradient descent. (a) Original image. (b) Blurred image. (c) P-method. (d) TDA-method. (e) P-method with MGD. (f) TDA-method with NAG.

TABLE 7. PSNR improvement of the TDA-method with AGD techniques.

Filter	Input	GD	MGD	NAG	RMSprop	ADAM	Adadelta
RGF [21]	**25**	23	-7	2	**25**	9	**25**
Gauss.	25	31	**33**	**33**	31	29	32
AMF [22]	20	29	34	**35**	31	33	34
RTV [23]	22	25	24	25	25	23	**26**
ILS [24]	33	39	38	39	39	33	**40**
L0 [25]	27	27	22	23	**28**	24	27
BF [7]	28	34	**39**	32	34	35	38
Disk	22	26	**30**	30	27	28	27
Motion	19	23	**27**	**27**	24	26	23
GF [8]	32	45	47	**48**	40	40	47
GF+Gauss.	23	25	**26**	**26**	25	24	**26**

of the image after applying each reverse filter method at 50 iterations. The results for the TDA-, P- and T-method are shown in Tables 7, 8 and 9 respectively, where a negative number indicates a non-convergent iteration. We can see that among the AGD methods MGD, NAG and Adadelta generally produce the best results. Table 7 shows that, for all smoothing filters, applying AGD to TDA-method results in improved image quality. In addition, Tables 8 and 9 show that, in some cases, reversing nonlinear filters such as AMF

TABLE 8. PSNR improvement of the P-method with AGD techniques.

Filter	Input	GD	MGD	NAG	RMSprop	ADAM	Adadelta
RGF [21]	**25**	**25**	0	11	**25**	13	**25**
Gauss.	25	31	**34**	**34**	31	29	**34**
AMF [22]	20	**34**	33	28	31	**34**	33
RTV [23]	22	25	24	25	25	23	**26**
ILS [24]	33	**40**	36	39	39	33	**40**
L0 [25]	27	27	24	27	**28**	25	27
BF [7]	28	34	38	37	34	35	**40**
Disk	22	27	**31**	**31**	27	28	28
Motion	19	24	**28**	27	25	26	24
GF [8]	32	48	47	**48**	41	40	47
GF+Gauss.	23	26	**27**	**27**	25	24	26

TABLE 9. PSNR improvement of the T-method with AGD techniques.

Filter	Input	GD	MGD	NAG	RMSprop	ADAM	Adadelta
RGF [21]	**25**	14	-4	0	14	6	15
Gauss.	25	38	46	**47**	37	39	38
AMF [22]	20	**44**	40	42	41	39	**44**
RTV [23]	22	29	29	**30**	29	28	29
ILS [24]	33	**43**	41	41	41	37	**43**
L0 [25]	27	28	25	25	**29**	24	**29**
BF [7]	28	**46**	35	35	40	39	**46**
Disk	**22**	-14	-90	-106	5	-18	-10
Motion	**19**	-52	-146	-176	4	-19	-40
GF [8]	32	53	58	**73**	45	43	53
GF+Gauss.	23	29	**34**	**34**	29	30	29

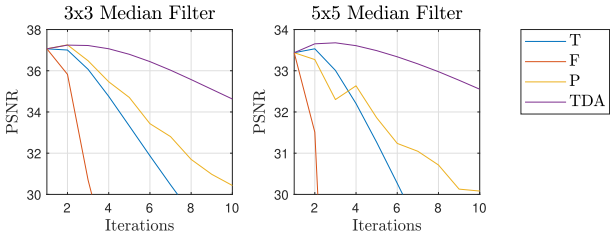
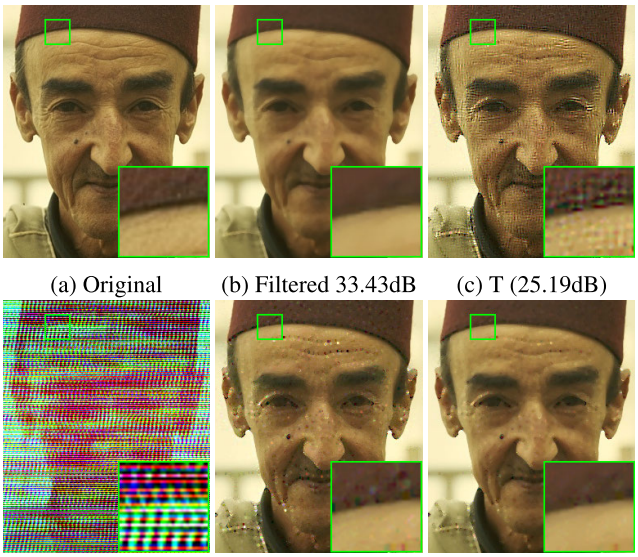


FIGURE 16. Values of PSNR per iteration obtained when reversing a Median filter using the T-, F-, P- and TDA- methods. (a) 3 × 3 Median filter. (b) 5 × 5 Median filter.

and RTV, the T- and P-method do not benefit from using the AGD.

D. A CHALLENGING QUESTION

We demonstrate a limitation of the reverse filtering process by considering median filters. We filter an image using a median filter with two patch sizes of 3 × 3 and 5 × 5. We then apply 10 iterations of the T-, F-, P- and TDA-method to try to reverse the filter effect. Results are shown in Figures 16 and 17 which show that the PSNR decreases after each iteration and the visual quality of the image is also decreasing. Thus, all algorithms failed to reverse the effect of a median filter which removes some information from the image. This study raised a challenging question: does such a family of reversing filters recover lost information due to the



(a) Original (b) Filtered 33.43dB (c) T (25.19dB)

(d) F (6.93dB) (e) P (29.35dB) (f) TDA (32.33dB)

FIGURE 17. Limitation example. Results of reversing a 5 × 5 median filter using 10 iterations. (a) Original image. (b) Filtered image. (c) T-method. (d) F-method. (e) P-method. (f) TDA-method.

original filter? If so, what is the maximum information loss that a particular reverse filter can tolerate?

V. CONCLUSION AND FURTHER RESEARCH

In this paper, we presented a new method to solve the semi-blind inverse filtering problem where the filter is available as a black box. The proposed algorithm is based on solving a local patch-based minimization problem by gradient descent where the unknown gradient is approximated by total derivative. We study issues related to convergence and output image quality for the case of a linear low pass filter. Results provide new insights into the proposed method and the T-method. Because the proposed method and two other existing methods can be regarded as gradient descent algorithms, we study the application of some widely used accelerated gradient descent (AGD) algorithms. We have demonstrated that applying AGD can usually lead to better image quality in a smaller number of iterations. However, the success of each AGD method depends on the filter being reversed. So the optimum AGD method needs to be found empirically. Through extensive experiments and comparisons, we have demonstrated that the proposed TDA-method has achieved a good balance between efficiency in terms of complexity and effectiveness in terms of its ability to reverse a larger number of filters.

The failure of all gradient-free algorithms to reverse the effect of a median filter reveals a limitation of this family of methods and calls for further investigation into the assumptions being made in the algorithmic development and possible ways to develop new algorithms. Another direction for further research is to apply numerical techniques such as those developed for the acceleration of fixed-point iterations to accelerate iterative reverse filters. In [48], we have presented some results based on both fixed-point and gradient descent accelerations.

APPENDIX

We present derivation of equations (19) and (20). We use the following facts: $B = GX$ and $X_0 = B = GX$. We can write T-method iteration in the Fourier domain as the following

$$\begin{aligned} X_{k+1} &= X_k + B - GX_k && (34)\\ &= X - (1-G)(X - X_k) && (35)\\ &= X - H_0(X - X_k) && (36) \end{aligned}$$

where $H_0 = (1 - G)$. We can rewrite the above equation as

$$X_{k+1} - X = H_0(X_k - X) \quad (37)$$

Therefore, we have

$$\begin{aligned} X_k - X &= H_0^k(X_0 - x) && (38)\\ &= H_0^k(GX - X) && (39)\\ &= -H_0^k I && (40) \end{aligned}$$

where $I = (1 - G)X$. Equation (19) is derived.

In the Fourier domain, we can write the TDA-method iteration as the following

$$\begin{aligned} X_{k+1} &= X_k + G(X_k + B - GX_k) - GX_k && (41)\\ &= GB + (1 - G^2)X_k && (42)\\ &= G^2X + (1 - G^2)X_k && (43)\\ &= X - (1 - G^2)X + (1 - G^2)X_k && (44)\\ &= X - (1 - G^2)(X - X_k) && (45)\\ &= X - J_0(X - X_k) && (46) \end{aligned}$$

where $J_0 = (1 - G^2)$. Equation (20) can be similarly derived by using the above method.

REFERENCES

[1] R. Gonzalez, R. Woods, and S. Eddins, *Digital Image Processing Using MATLAB*. Hoboken, NJ, USA: Pearson Prentice-Hall, 2004.

[2] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 12, pp. 2341–2353, Sep. 2010.

[3] Z. G. Li and J. H. Zheng, "Single image de-hazing using globally guided image filtering," *IEEE Trans. Image Process.*, vol. 27, no. 1, pp. 442–450, Jan. 2018.

[4] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, Jun. 2005, pp. 60–65.

[5] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Trans. Image Process.*, vol. 16, no. 8, pp. 2080–2095, Aug. 2007.

[6] P. Jain and V. Tyagi, "A survey of edge-preserving image denoising methods," *Inf. Syst. Frontiers*, vol. 18, no. 1, pp. 159–170, 2016.

[7] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th Int. Conf. Comput. Vis.*, 1998, pp. 839–846.

[8] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[9] X. Wei, Q. Yang, and Y. Gong, "Joint contour filtering," *Int. J. Comput. Vis.*, vol. 126, no. 11, pp. 1245–1265, Nov. 2018.

[10] H. Yin, Y. Gong, and G. Qiu, "Side window guided filtering," *Signal Process.*, vol. 165, pp. 315–330, Dec. 2019.

[11] Z. Sun, B. Han, J. Li, J. Zhang, and X. Gao, "Weighted guided image filtering with steering kernel," *IEEE Trans. Image Process.*, vol. 29, pp. 500–508, 2020, doi: 10.1109/TIP.2019.2928631.
[12] Z. Li, J. Zheng, Z. Zhu, W. Yao, and S. Wu, "Weighted guided image filtering," *IEEE Trans. Image Process.*, vol. 24, no. 1, pp. 120–129, Jan. 2015.
[13] Y. Endo, Y. Kanamori, and J. Mitani, "Deep reverse tone mapping," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 1–10, Nov. 2017.
[14] J. Koh, J. Lee, and S. Yoon, "Single-image deblurring with neural networks: A comparative survey," *Comput. Vis. Image Understand.*, vol. 203, Feb. 2021, Art. no. 103134.
[15] A. C. Kak and M. Slaney, *Principles of Computerized Tomographic Imaging*. Piscataway, NJ, USA: IEEE Press, 1988.
[16] J. Romberg, "Imaging via compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 14–20, Mar. 2008.
[17] X. Tao, C. Zhou, X. Shen, J. Wang, and J. Jia, "Zero-order reverse filtering," in *Proc. ICCV*, 2017, pp. 222–230.
[18] P. Milanfar, "Rendition: Reclaiming what a black box takes away," 2018, *arXiv:1804.08651*.
[19] A. G. Belyaev and P.-A. Fayolle, "Two iterative methods for reverse image filtering," *Signal, Image Video Process.*, vol. 15, pp. 1–9, Apr. 2021.
[20] L. Dong, J. Zhou, C. Zou, and Y. Wang, "Iterative first-order reverse image filtering," in *Proc. ACM Turing Celebration Conf. China*, 2019, pp. 1–5.
[21] Q. Zhang, X. Shen, L. Xu, and J. Jia, "Rolling guidance filter," in *Proc. ECCV*. Cham, Switzerland: Springer, 2014, pp. 815–830.
[22] E. S. L. Gastal and M. M. Oliveira, "Adaptive manifolds for real-time high-dimensional filtering," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–13, Aug. 2012.
[23] L. Xu, Q. Yan, Y. Xia, and J. Jia, "Structure extraction from texture via relative total variation," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 1–10, 2012.
[24] W. Liu, P. Zhang, X. Huang, J. Yang, C. Shen, and I. Reid, "Real-time image smoothing via iterative least squares," *ACM Trans. Graph.*, vol. 39, no. 3, pp. 1–24, Jun. 2020.
[25] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via L_0 gradient minimization," *ACM Trans. Graph.*, vol. 30, no. 6, pp. 1–12, 2011.
[26] M. J. Kochenderfer and T. A. Wheeler, *Algorithms for Optimization*. Cambridge, MA, USA: MIT Press, 2019.
[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[28] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*. Philadelphia, PA, USA: SIAM, 2000.
[29] G. Deng and P. Broadbridge, "Bregman inverse filter," *Electron. Lett.*, vol. 55, no. 4, pp. 192–194, Feb. 2019.
[30] N. Weaver, *Lipschitz Algebras*. Singapore: World Scientific, 2018.
[31] M. Delbracio, I. Garcia-Dorado, S. Choi, D. Kelly, and P. Milanfar, "Polyblur: Removing mild blur by polynomial reblurring," *IEEE Trans. Comput. Imag.*, vol. 7, pp. 837–848, 2021.
[32] B. T. Polyak, "Minimization of unsmooth functionals," *USSR Comput. Math. Math. Phys.*, vol. 9, no. 3, pp. 14–29, Jan. 1969.
[33] J. Steffensen, "Remarks on iteration," *Scandin. Actuarial J.*, vol. 1933, no. 1, pp. 64–72, 1933.
[34] A. G. Belyaev and P.-A. Fayolle, "Black-box image deblurring and defiltering," *Signal Process., Image Commun.*, vol. 108, Oct. 2022, Art. no. 116833.
[35] A. K. Jain, *Fundamentals of Image Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, 1989.
[36] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Comput. Math. Math. Phys.*, vol. 4, no. 5, pp. 1–17, 1964.
[37] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate $O(1/k^2)$," *Doklady Akademii Nauk SSSR*, vol. 269, no. 3, pp. 543–547, 1983.
[38] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop, coursera: Neural networks for machine learning," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Tech. Rep. 6, 2012.
[39] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*.
[40] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[41] G. Zhai and X. Min, "Perceptual image quality assessment: A survey," *Sci. China Inf. Sci.*, vol. 63, no. 11, Nov. 2020, Art. no. 211301.
[42] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th IEEE Int. Conf. Comput. Vis. (ICCV)*, vol. 2. Jul. 2001, pp. 416–423.
[43] G. Deng, F. Galetto, M. Alnasrawi, and W. Waheed, "A guided edge-aware smoothing-sharpening filter based on patch interpolation model and generalized gamma distribution," *IEEE Open J. Signal Process.*, vol. 2, pp. 119–135, 2021.
[44] T. J. Holmes, S. Bhattacharyya, J. A. Cooper, D. Hanzel, V. Krishnamurthi, W.-C. Lin, B. Roysam, D. H. Szarowski, and J. N. Turner, "Light microscopic images reconstructed by maximum likelihood deconvolution," in *Handbook of Biological Confocal Microscopy*. Springer, 1995, pp. 389–402.
[45] W. H. Richardson, "Bayesian-based iterative method of image restoration," *J. Opt. Soc. Amer.*, vol. 62, no. 1, pp. 55–59, Jan. 1972.
[46] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-Laplacian priors," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 22, 2009, pp. 1033–1041.
[47] N. Wiener, *Extrapolation, Interpolation, and Smoothing of Stationary Time Series: With Engineering Applications*, vol. 8. Cambridge, MA, USA: MIT Press, 1964.
[48] F. Galetto and G. Deng, "Fast image reverse filters through fixed point and gradient descent acceleration," 2022, *arXiv:2206.10124*.

FERNANDO J. GALETTO received the B.Eng. degree in electronics engineering from the National Technological University, Córdoba, Argentina, in 2015, and the M.Eng. degree in electronics engineering from La Trobe University, Melbourne, VIC, Australia, in 2019, where he is currently pursuing the Ph.D. degree. His research interests include computer vision and underwater image processing.

GUANG DENG is currently a Reader and an Associate Professor of electronic engineering with La Trobe University, Melbourne, VIC, Australia. His current research interests include Bayesian signal processing, lossless image compression, and generalized linear systems.