REVERSE IMAGE FILTERING USING TOTAL DERIVATIVE APPROXIMATION AND ACCELERATED GRADIENT DESCENT

A PREPRINT

Fernando J. Galetto*
Department of Engineering
La Trobe University
Bundoora, VIC 3086, Australia
f.galetto@latrobe.edu.au

Guang Deng
Department of Engineering
La Trobe University
Bundoora, VIC 3086, Australia
d.deng@latrobe.edu.au

December 8, 2021

ABSTRACT

In this paper, we address a new problem of reversing the effect of an image filter, which can be linear or nonlinear. The assumption is that the algorithm of the filter is unknown and the filter is available as a black box. We formulate this inverse problem as minimizing a local patch-based cost function and use total derivative to approximate the gradient which is used in gradient descent to solve the problem. We analyze factors affecting the convergence and quality of the output in the Fourier domain. We also study the application of accelerated gradient descent algorithms in three gradient-free reverse filters, including the one proposed in this paper. We present results from extensive experiments to evaluate the complexity and effectiveness of the proposed algorithm. Results demonstrate that the proposed algorithm outperforms the state-of-the-art in that (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, and (2) it can reverse the same list of filters as that of the very complex reverse filter, but its complexity is much smaller.

***K*eywords** Inverse filtering, optimization, accelerated gradient descent, total derivative.

1 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 [4–6]. In these applications, an edge-aware filter such as [7–12] 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 the model parameter 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 our current research.

The increasing use of many image processing filters in many areas 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 which it implements. 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 estimate the original image given the observation $\boldsymbol{b}$ and the black box filter $g(.)$. Two factors make the study of such a problem theoretically and practically important.

*Corresponding author.

- Traditional methods, which rely on using gradient information to solve optimization problems, will face a challenge of lack of such information because the filter is unknown. New algorithms, which do not directly rely on gradient information, must be developed.
- While classical image restoration techniques are mostly dealing with known or estimated linear filter kernels for the inversion, solutions of the new inverse problem, which do not rely on the model of the filter to be inverted, can potentially deal with a wide range of unknown linear and nonlinear filters.

A brief review of current work on this new problem is presented in section 2 which includes 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 (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 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). This is in contrast to a deep neural network approach in which a network is trained to deal with a specific problem. 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 application 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 development of the TDA-method (section 3.1) and provide a theoretical analysis of the convergent condition and factors related to the quality of the recovered image (section 3.2). We then discuss applying AGD methods to 3 spatial domain reverse filtering algorithms, including the TDA-method (section 3.3).
- In section 4 we show results of extensive experimental study of the proposed method, including: the effect of parameter settings such as learning rate and number of iterations (section 4.1), qualitative and quantitative evaluations and comparisons to demonstrate the performance of current works and the TDA-method in reversing a wide range of filters (section 4.2) and a study of using AGD methods (section 4.3). We also present an intriguing case study where all methods failed to reverse the effect of a simple median filter to illustrate the limitation of this class of gradient-free algorithms (section 4.4).

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 and 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 $\mathcal{F}(\boldsymbol{x}_k)$.

2 Previous work

We follow the terminology used in [19] which called a particular reverse filter a method and briefly comment on each of them in this section. We present a summary of related previous work in Table 1 where we define the following variables which will be used in rest of this paper.

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

The T-method, which is originally called zero-order reverse filter [17], can be formulated as solving a fixed point iteration problem [28] $\boldsymbol{x} = \boldsymbol{x} + \boldsymbol{b} - g(\boldsymbol{x})$. It is the fastest method among all reverse filtering methods considered in this paper. When the contraction mapping condition ($|1 - g'(x)| < 1$ in 1D case) 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 using the T-method. 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, which is originally called the rendition algorithm [18], is developed by minimizing the cost function: $\boldsymbol{x}^T(g(\boldsymbol{x}) - \boldsymbol{b}) - \frac{1}{2}\boldsymbol{x}^T\boldsymbol{x}$ 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 when $\alpha = 1$ and $\beta = 1$. 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.

The F-method, which is formulated in the frequency domain [20], uses the Newton-Raphson technique to invert unknown linear filters. 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 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].

Table 1: A comparison of the 4 space domain methods, including the proposed TDA-method 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**
T [17]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{q}_k$	N/A	$O(n) + C$
R [18]	$\boldsymbol{x}_{k+1} = \alpha\boldsymbol{x}_k + \lambda\boldsymbol{q}_k$	α, λ	$O(n) + C$
P [19]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \frac{\|\|\boldsymbol{q}_k\|\|}{2\|\|\boldsymbol{p}_k\|\|}\boldsymbol{p}_k$	N/A	$O(n^2) + C$
TDA	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \lambda\boldsymbol{t}_k$	$\lambda \leq 1$	$O(n) + C$
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	$O(n \log n) + C$

3 The TDA-method, analysis and accelerations

3.1 The proposed TDA-method

To develop the proposed algorithm, we assume the filter only acts on a patch of the pixels. For notation 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 $\boldsymbol{y} = [y(1), y(2), y(3), y(4), y(5), y(6), y(7), y(8), y(9)]^T$ where the pixel to be estimated is $y(5)$ and b is the observed pixel corresponding to $y(5)$.

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

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

Using gradient descent, we can write

$$\boldsymbol{y}_{k+1} = \boldsymbol{y}_k - \lambda\nabla f(\boldsymbol{y}_k) \tag{2}$$

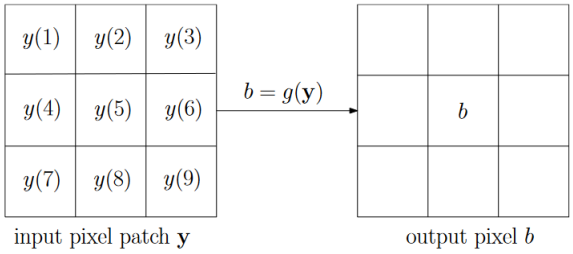


Figure 1: Illustration of filtering a 3×3 patch resulting a pixel.

where $\nabla f(\boldsymbol{y}) = -u_k(5)\boldsymbol{d}$, $u_k(5) = b - g(\boldsymbol{y}_k)$ and $\boldsymbol{d}$ is a vector with its nth element defined as $d(n) = \frac{\partial g}{\partial y(n)}$. Because in each patch we are only interested in recovering the center pixel of the patch $y(5)$, from equation (2) we have

$$y_{k+1}(5) = y_k(5) + \lambda u_k(5)\frac{\partial g}{\partial y(5).} \tag{3}$$

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 $c(\boldsymbol{y}) : \mathbb{R}^N \rightarrow \mathbb{R}$ at a point $\boldsymbol{a}$ can be approximated as

$$c(\boldsymbol{a} + \boldsymbol{\delta}) - c(\boldsymbol{a}) \approx \sum_{n=1}^{N} \delta(n)\frac{\partial c}{\partial y(n)} \tag{4}$$

where $y(n)$ and $\delta(n)$ are the nth element of the vector $\boldsymbol{y}$ and $\boldsymbol{\delta}$, respectively. With this approximation, we can write the following for the patch:

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

where values of elements of $\boldsymbol{u}_k = [u_k(1), u_k(2), ..., u_k(9)]$ 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 (3) with (5) 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:

$$u_k(5)\frac{\partial g}{\partial y(5)} \rightarrow \sum_{n=1}^{9} u_k(n)\frac{\partial g}{\partial y(n)} \approx g(\boldsymbol{y}_k + \boldsymbol{u}_k) - g(\boldsymbol{y}_k) \tag{6}$$

where the symbol "$\rightarrow$" represents the operation of "replacing 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 difference. Using this approximation in equation (3), we can write

$$y_{k+1}(5) = y_k(5) + \lambda(g(\boldsymbol{y}_k + \boldsymbol{u}_k) - g(\boldsymbol{y}_k)) \tag{7}$$

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 (7) and use our general notation of $\boldsymbol{x}_k$ to represent an image in the kth iteration. We can write

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \lambda(g(\boldsymbol{x}_k + \boldsymbol{q}_k) - g(\boldsymbol{x}_k)) \tag{8}$$

Equation (8) is the proposed algorithm, which is thus called total derivative approximation (TDA).

3.2 Convergence condition and image quality

It is in general a very difficult task to analyse gradient free algorithms studied in this work, because the large number nonlinear filters make the analysis non-tractable. In this work, we focus on studying the properties of the T-method and the TDA-method for the mathematically simplest class of filters–LSI low-pass 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 the linear filter.

3.2.1 Frequency domain representations

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

$$X_{k+1} = X_k + B - GX_k = B + (1 - G)X_k \tag{9}$$

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

$$X_k = X - H_0^k I \tag{10}$$

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:

$$X_k = X - J_0^k I \tag{11}$$

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.

3.2.2 Analysis

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

$$X_k = X - H_k I \tag{12}$$

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 commonly used image processing 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

$$|H_k(\omega_1, \omega_2)| = |H_0(\omega_1, \omega_2)|^k \tag{13}$$

where $H_0(\omega_1, \omega_2) = 1 - G(\omega_1, \omega_2)$ for the T-method and $H_0(\omega_1, \omega_2) = 1 - G^2(\omega_1, \omega_2)$ for the TDA-method. Let the maximum value of the amplitude response be

$$T_0(\omega_1^*, \omega_2^*) = \max_{(\omega_1, \omega_2)} |H_0(\omega_1, \omega_2)| \tag{14}$$

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 super-script k represents raising to the power of k. We consider the following three cases.

- If $T_0(\omega_1^*, \omega_2^*) > 1$, then $T_k(\omega_1^*, \omega_2^*)$ increases exponentially with k, which is unbounded. 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.
- If $T_0(\omega_1^*, \omega_2^*) < 1$, then $T_k(\omega_1^*, \omega_2^*)$ 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.
- If $T_0(\omega_1^*, \omega_2^*) = 1$, then $|H_0(\omega_1, \omega_2)| < 1$ $(\omega_1 \neq \omega_1^*, \omega_2 \neq \omega_2^*)$ 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^*).

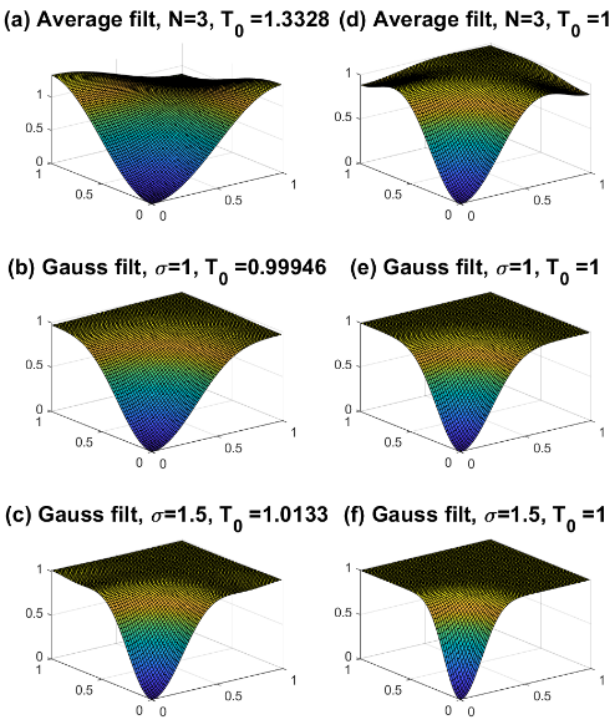


Figure 2: The amplitude response $|H_0|$ due to 3 linear low pass filters. Left column shows the case for the T-method where $H_0 = 1 - G$. Right column shows the case for the TDA-method where $H_0 = 1 - G^2$. The maximum value of $|H_0|$ is also shown as T_0.

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: (1) a (3×3) average filter, (2) a Gaussian low pass filter of standard deviation $\sigma = 1$, and (3) a Gaussian low pass 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 the figure, the two frequency axes are normalized for easy visualization) for the three filters. 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 stable for all three filters. This is because $0 \leq G^2 \leq 1$ and the maximum value of the amplitude response $|H_0| = |1 - G^2| \leq 1$ is attained at frequency where G^2 is minimum which is denoted $U_0(\omega_1^*, \omega_2^*) = \min_{(\omega_1, \omega_2)} G^2(\omega_1, \omega_2)$. The maximum value of the amplitude response is thus given by $T_0(\omega_1^*, \omega_2^*) = 1 - U_0(\omega_1^*, \omega_2^*)$. For the three filters, we have $U_0(\omega_1^*, \omega_2^*) = 0$ leading to $T_0(\omega_1^*, \omega_2^*) = 1$.
- 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).

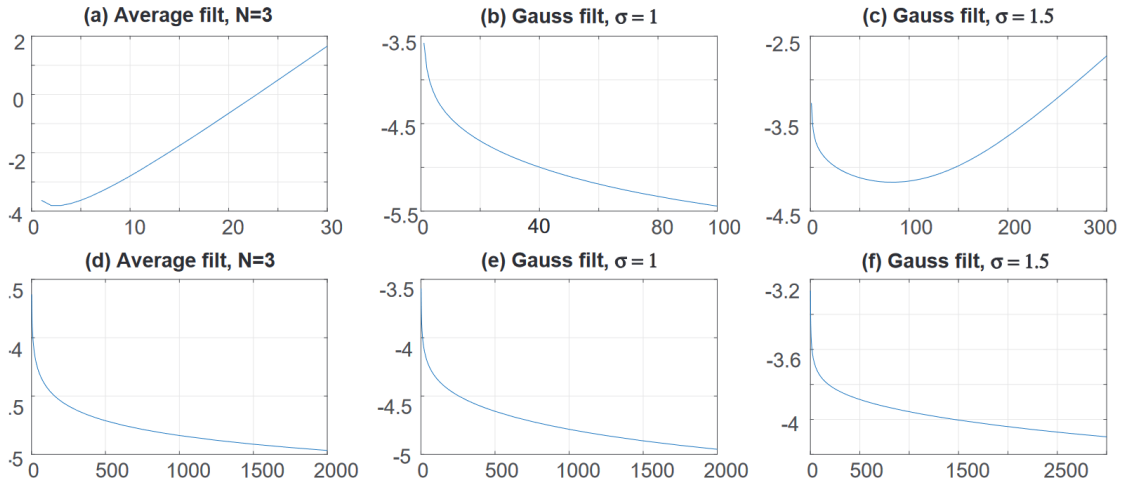


Figure 3: The mean square error (in $\log_{10}$ scale, vertical axis) of the recovered image as a function of number of iterations (horizontal axis). Top row shows the case for the T-method, while the bottom row shows the case for the TDA-method.

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 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.

3.3 Using accelerated gradient descent

Accelerated gradient descent (AGD) methods [26], which are summarised in Table 2, were developed to tackle problems related gradient descend. They are widely used 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 4.3.

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

3.4 Summary and comparison

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. The P-method and the TDA-method are related through the

Table 2: AGD methods.

Gradient descent	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \lambda \nabla f(\boldsymbol{x}_k)$
MGD [34]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \lambda \nabla f(\boldsymbol{x}_k) + \beta \boldsymbol{v}_{k-1}$
NAG [35]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \lambda \nabla f(\boldsymbol{x}_k + \beta \boldsymbol{v}_{k-1}) + \beta \boldsymbol{v}_{k-1}$
RMSProp [36]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \frac{\lambda}{\sqrt{\boldsymbol{v}_k + \epsilon}} \nabla f(\boldsymbol{x}_k)$ $\boldsymbol{v}_k = \beta \boldsymbol{v}_{k-1} + (1-\beta) \nabla f(\boldsymbol{x}_k)^2$
Adadelta [37]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \Delta \boldsymbol{x}_k$ $\Delta \boldsymbol{x}_k = \frac{\sqrt{\boldsymbol{u}_{k-1} + \epsilon}}{\sqrt{\boldsymbol{v}_k + \epsilon}} \nabla f(\boldsymbol{x}_k)$ $\boldsymbol{v}_k = \beta \boldsymbol{v}_{k-1} + (1-\beta) \nabla f(\boldsymbol{x}_k)^2$ $\boldsymbol{u}_k = \beta \boldsymbol{u}_{k-1} + (1-\beta) \Delta \boldsymbol{x}_k^2$
ADAM [38]	$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k - \lambda \frac{\hat{\boldsymbol{m}}_k}{\sqrt{\hat{\boldsymbol{v}}_k + \epsilon}}$ $\boldsymbol{m_k} = \beta_1 \boldsymbol{m_{k-1}} + (1-\beta_1) \nabla f(\boldsymbol{x}_k)$ $\boldsymbol{v_k} = \beta_2 \boldsymbol{v_{k-1}} + (1-\beta_2) \nabla f(\boldsymbol{x}_k)^2$ $\hat{\boldsymbol{m}}_{\boldsymbol{k}} = \frac{\boldsymbol{m_k}}{1-\beta_1}, \hat{\boldsymbol{v}}_{\boldsymbol{k}} = \frac{\boldsymbol{v_k}}{1-\beta_2}$

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 running time difference between the T-method and TDA-method is because the latter requires two Gaussian filters per iteration, while the former only requires one.

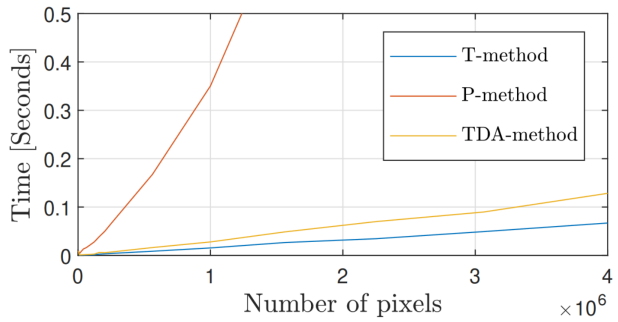
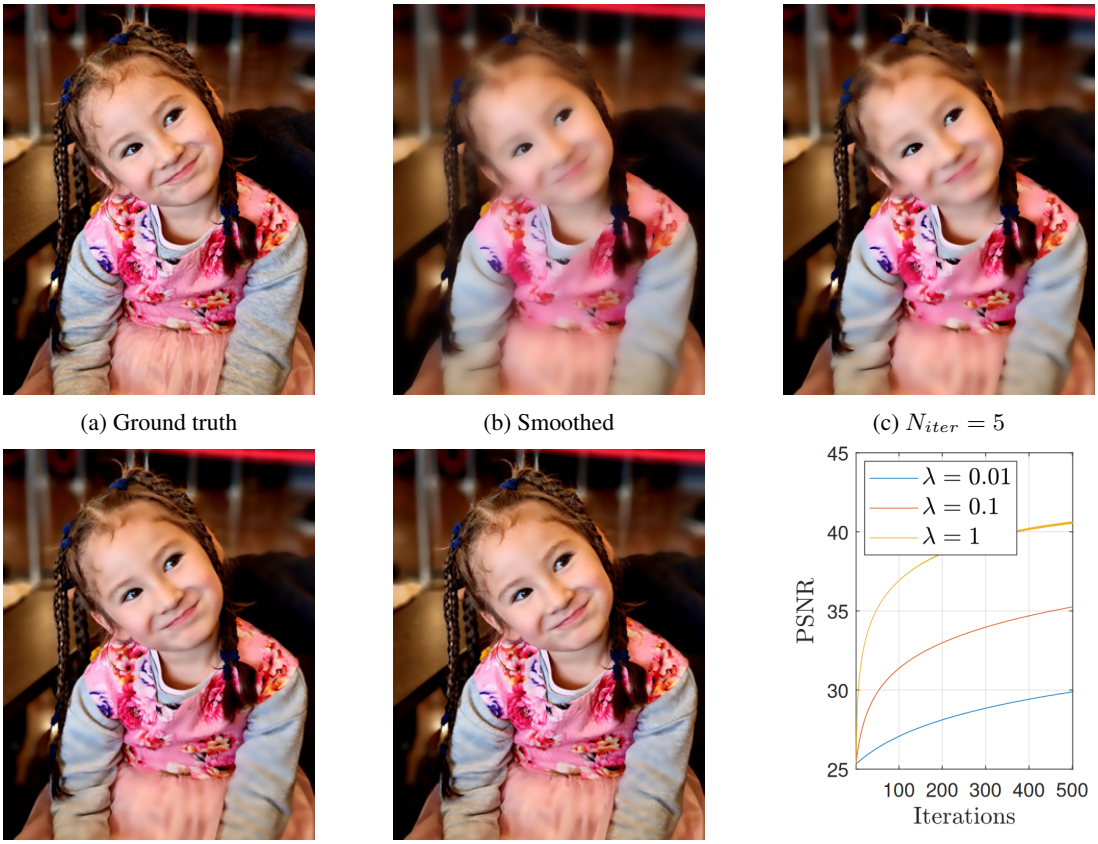


Figure 4: Comparison of running time vs image size when reversing a Gaussian filter with a kernel size of 7×7 pixels and a standard deviation $\sigma = 1$ applying one iteration of T-, P- and TDA-method.

4 Experiments and results

In this section, we present an experimental study of the proposed TDA-method, including effects setting the learning rate λ and number of iterations (section 4.1), validation of its effectiveness in reversing a wide range of filters (section 4.2), and a comparison (section 4.2.2) with of the four existing methods reviewed in section 2. We also present an experimental study (section 4.3) of using accelerated gradient descent algorithms. The limitation (section 4.4) of all current methods is demonstrated through an example in which all of them failed to reverse the effect of a median filter. The running time tests were conducted using MATLAB r2021a running in a PC with Intel i7-3930k CPU and 40GB RAM.



(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 $\lambda = 1$. (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.

4.1 Number of iterations, learning rate and image quality

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

4.1.1 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 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 $\lambda \leq 1$. We observed that, for this convergent case, a larger value leads to a faster improvement in image quality.

4.1.2 Non-convergent case

Although we have shown in last section, the condition for proposed TDA-method to be stable in the linear filter case, the same analysis for a nonlinear filter is in general very difficult. 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.

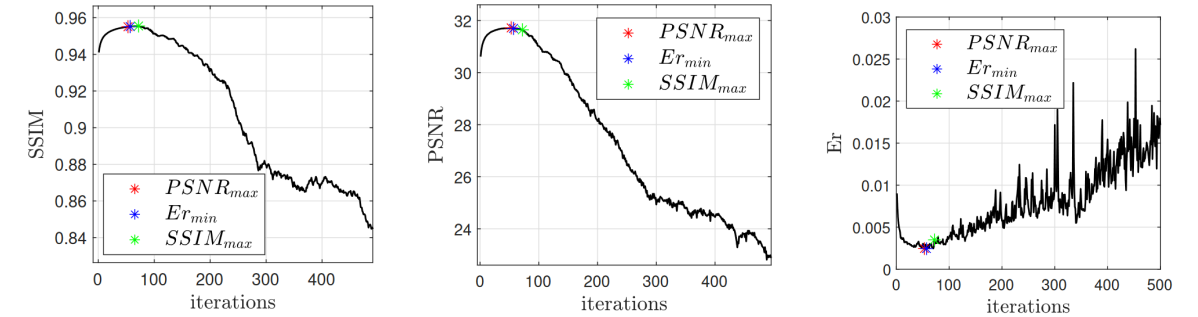
How do we quantify the output image quality without knowing the original image? Since we have the input image $\boldsymbol{b} = g(\boldsymbol{x})$ and the filter $g(.)$ as a black box, we can calculate the following relative error:

$$e_k = \frac{||\boldsymbol{b} - g(\boldsymbol{x}_k)||_2^2}{||\boldsymbol{b}||_2^2} \tag{15}$$

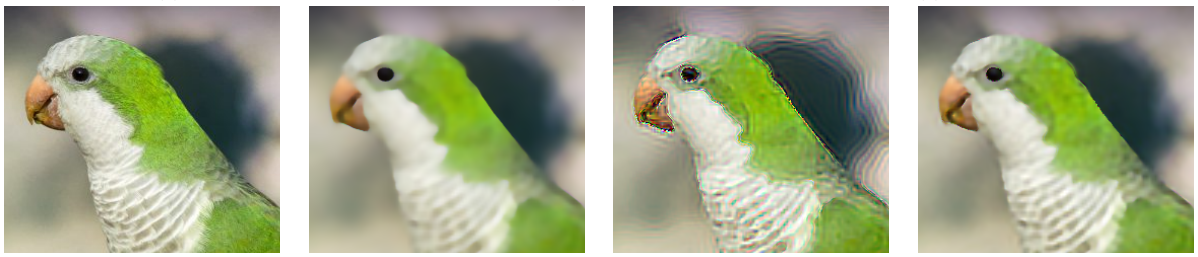
If $||\boldsymbol{b} - g(\boldsymbol{x}_k)||_2^2$ is small, then under the bi-Lipschitz condition[2] we can expect that $||\boldsymbol{x} - \boldsymbol{x}_k||_2^2$ will also be small. Thus, the relative error is a non-referenced metric which can be used to determine when the iteration can be stopped to achieve the best outcome. Indeed, we can use a two-pass process to determine the optimum iterations. 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 [39]. 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 which 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 as those determined by using PSNR and SSIM ($k = 53$ and $k = 72$, respectively). Thus, this experiment support 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.



(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).

[2]We regard the filter $g(.)$ as a function. When it is bi-Lipschitz [30], it satisfies the condition $\frac{1}{K}d_X(x_1, x_2) \leq d_Y(y_1, y_2) \leq Kd_X(x_1, x_2)$ where we define $y = g(x)$, X and Y are set 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.

4.1.3 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 4.3.

4.2 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.

4.2.1 Evaluation based on 12 filters

We performed experiments using 300 natural images from the BSD300 data set [40]. 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. The average PSNR for each filter in each iteration is calculated over the 300 images. Since different filters produce different levels of degradation, we use the percentage of improvement given by the following equation to compare the performance for different filters

$$\bar{p}_k = \frac{p_k - p_0}{p_0} \times 100 \tag{16}$$

where p_k and p_0 represent respectively the average PSNR value for a filter over 300 images at the kth iteration and 0th iteration (the input image).

Table 3: Parameter settings for the 12 filters

Filter	**Parameters**
RGF [21]	$\sigma_s = 3, \quad \sigma_r = 0.05, \quad N_{iter} = 4$
Gauss.	$\sigma = 5$
LoG	$k = 7 \times 7, \quad \sigma = 0.4$
AMF [22]	$\sigma_s = 7, \quad \sigma_r = 0.4$
RTV [23]	$\lambda = 0.05, \quad \sigma = 3, \quad \epsilon = 0.05, \quad N_{iter} = 2$
ILS [24]	$\lambda = 1, \quad p = 0.8, \quad \epsilon = 1 \times 10^{-4}, \quad N_{iter} = 4$
L0 [25]	$\lambda = 0.01, \quad \kappa = 2$
BF [7]	$\epsilon = 0.05, \quad \sigma_s = 3$
Disk averaging filter	$r = 3$
Motion blur filter	$l = 20, \quad \theta = 45^\circ$
GF [8]	$w_{size} = 5 \times 5, \quad \epsilon = 0.1$
GF+Gauss.	$w_{size} = 5 \times 5, \quad \epsilon = 0.1, \quad \sigma = 5$

The results for the TDA-method are shown in Fig. 7a ($\lambda = 0.5$) and Fig. 7b ($\lambda = 1$). We can see that the performance of 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. We can also observe that, while setting $\lambda = 1$ helps faster improvement in image quality for most filters, such setting does not work in reversing the L0 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.

How is the performance of TDA-method when compared to those of the T- and P-method? We conducted the same experiment and presented the results in Fig.7(c) (T-method) and (d) (P-method). The T-method produces very good results for 5 filters ($> 20\%$ 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 numerous filters, and is also better than the P-method in terms of its much smaller computational cost.

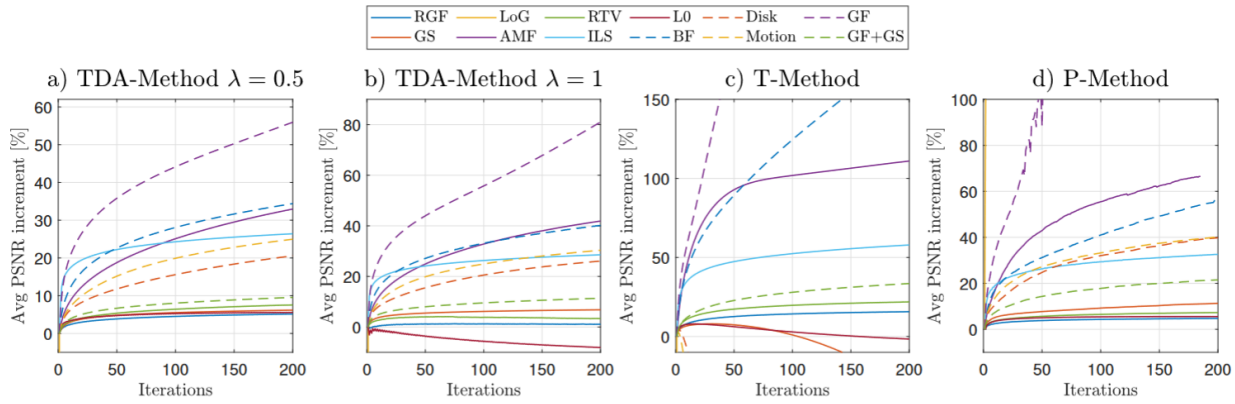


Figure 7: Curves of increment of the Average PSNR per iteration when reversing 12 well known filters using the the BSD300 dataset [40]. (a) TDA-method setting $\lambda = 0.5$. (b) TDA-method setting $\lambda = 1$. (c) T-method. (d) P-method.

4.2.2 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 of 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, but in some cases, those details are not completely removed but 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 [41] or the SSIF algorithm [42]. These two algorithms are highly nonlinear and are well known for their ability to smooth out texture. Results are shown in Fig. 8 which show that the texture information has been recovered to some extent in both images. To produce the results shown in Figures 8(c) and (f) we used 3000 and 500 iterations of the TDA-method respectively.

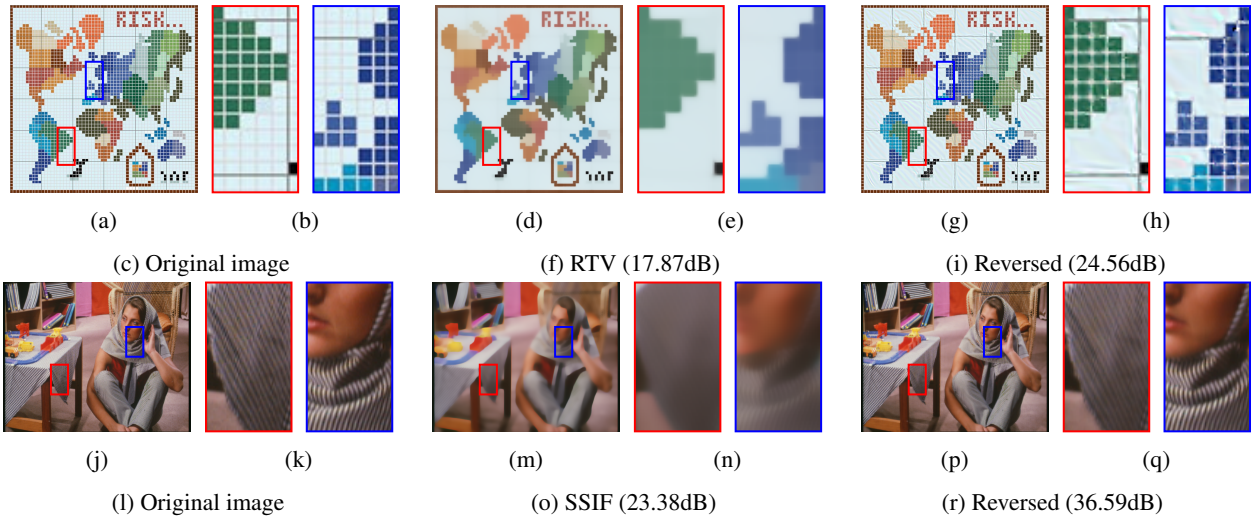


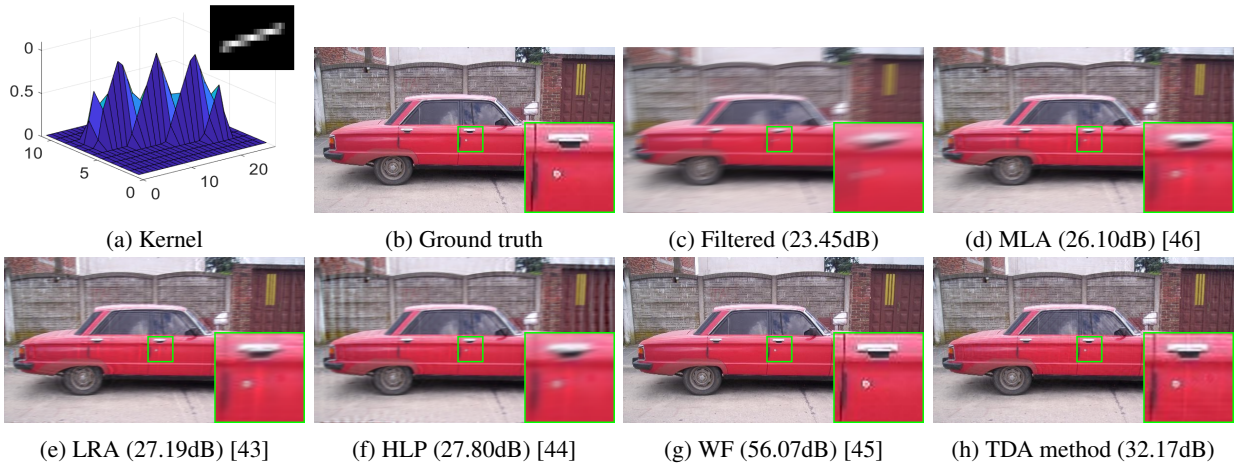
Figure 8: Recovering small-scale image texture by reversing detail removal filters. (a) Original image 1. (b) Filtered image with RTV ($\lambda = 0.05, \sigma = 3, \epsilon = 0.05, N_{iter} = 2$). (c) Reversed with TDA-method ($\lambda = 1, N_{iter} = 3000$). (d) Original image 2. (e) Filtered image with SSIF ($r = 5, \kappa = 0.1, \epsilon = 0.1, N_{iter} = 2$). (f) Reversed with TDA-method ($\lambda = 1, N_{iter} = 500$).

Example 2. We study the performance of the TDA-method in de-convolution applications [1]. When the filter kernel is known, the problem is non-blind. Otherwise, the problem is blind. Classical non-blind image restoration algorithms,

(a) Kernel (b) Ground truth (c) Filtered (30.02dB) (d) MLA (33.27dB) [46]

(e) LRA (34.16dB) [43] (f) HLP (33.87dB) [44] (g) WF (56.39dB) [45] (h) TDA method (37.67dB)

Figure 9: Reversing a Gaussian filter of size 7×7 and $\sigma = 1$. (a) Kernel. (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) [46]

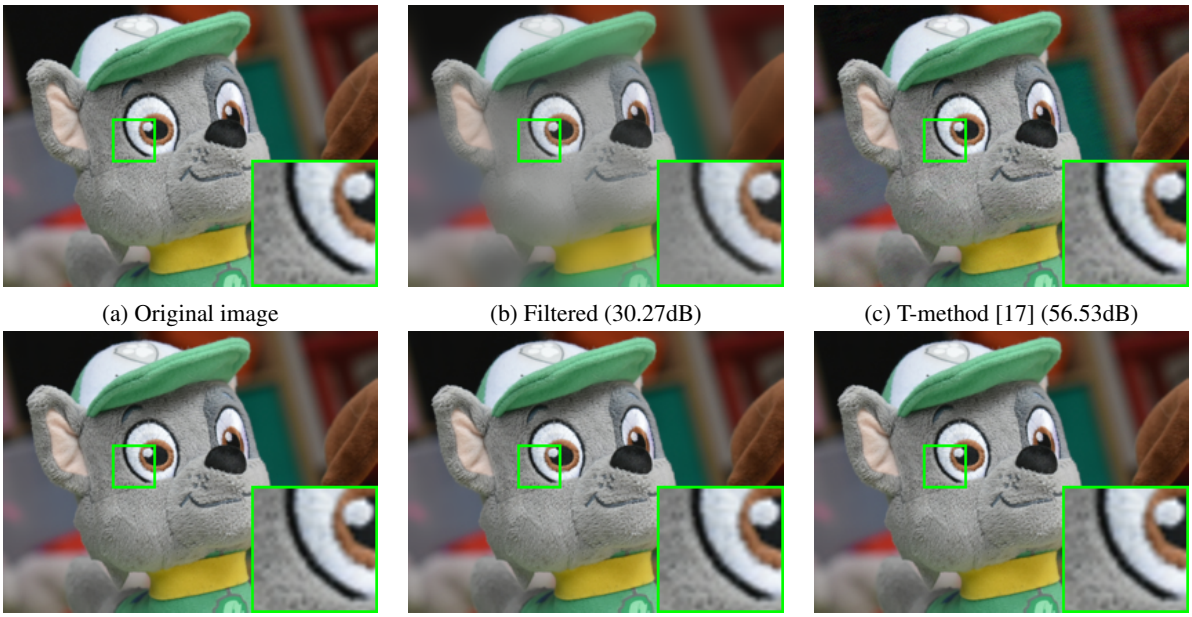
(e) LRA (27.19dB) [43] (f) HLP (27.80dB) [44] (g) WF (56.07dB) [45] (h) TDA method (32.17dB)

Figure 10: Reversing a motion blur filter. (a) Kernel. (b) Original image. (c) Filtered image. (d) MLA. (e) LRA. (f) HLP. (g) Wiener filter. (h) TDA-method.

which require the knowledge of the kernel, include: Lucy Richardson algorithm (LRA) [43], fast image de-convolution using hyper-Laplacian priors (HLP) [44] and Wiener filter (WF) [45]. A well-known blind restoration algorithm is the maximum likelihood algorithm (MLA) [46], which does not require the knowledge of the kernel. The proposed TDA-method is applicable for 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. 9 we show an example of smoothing using a Gaussian kernel of size 7×7 pixels and a standard deviation $\sigma = 1$. In Fig. 10 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 9 and 10). The TDA-method clearly produces a more appealing result than MLA, LRA, and HLP since it does not produce artifacts and the output image is sharper. The Wiener filter is the winner of 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.

Example 3. We compare the performance of TDA-method with that of the 4 methods described in section 2 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 11 (a) and (b), where we can see that textures have been smoothed while the main structure of the image is preserved. Figures 11 (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 had achieved the best improvement in PSNR.



(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 11: 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 ($N_{iter} = 500$).

4.3 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 4. We aim to study the following questions.

Table 4: 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	$\lambda = 1, \quad \beta = 0.9, \quad v_0 = 0, \quad u_0 = 0$

Question 1. When the filter can be reversed, how is the performance of the reverse filter changed by each of AGD method? 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. 12. 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, the AGD methods of MGD, NAG and ADAM result in significant improvement, while the other two AGD methods, RMSprop and Adadelta do not produce improvement. For the P-method, there is no improvement

when using any of the AGD method. 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.

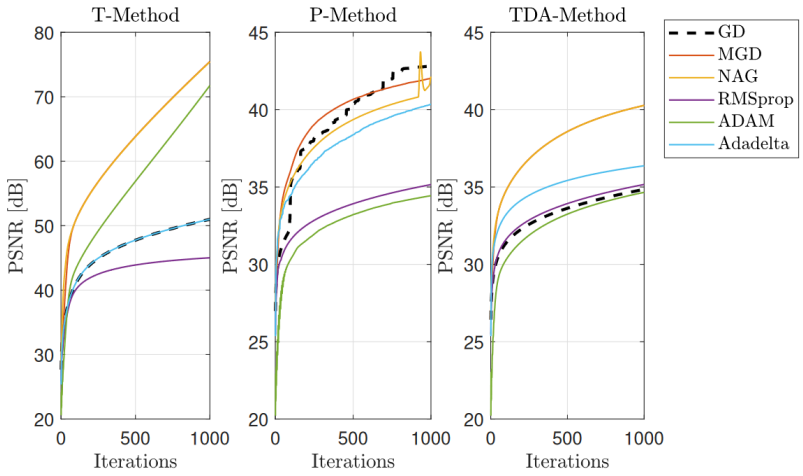


Figure 12: PSNR per iteration when reversing Gaussian filter with T-, P- and TDA-method applying accelerated gradient descent.

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. 14(a) was smoothed by a filter which approximates a linear motion of 20 pixels in an 45 degrees angle. Results are shown in Fig. 13. The T-method fails to recover the image and none of the AGD methods helps to make the T-method produce useful results.

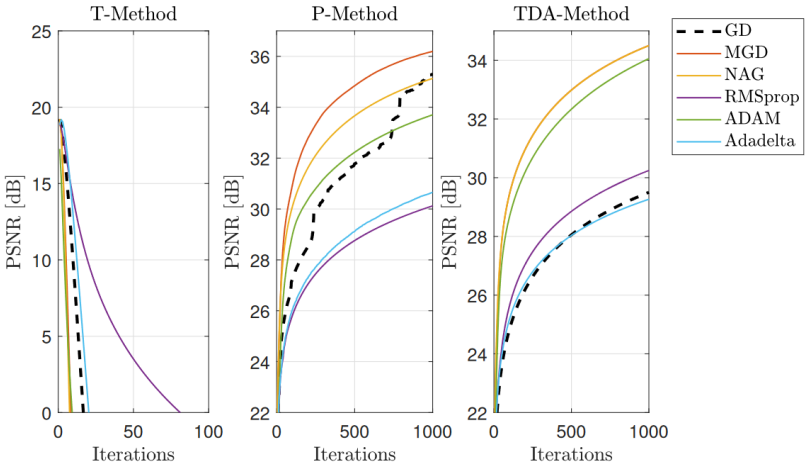


Figure 13: PSNR per iteration when reversing motion blur filter using T-, P- and TDA-method with accelerated gradient descent.

Question 3. When the T-method failed, how does the AGD help improve the performance of P- and TDA-method? We can see in Fig. 13 that performance of the P-method and T-method are improved by using MGD, NAG and ADAM. To provide further evidence of the improvement and to further compare these two methods, we present in Fig. 14 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 second, 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, comparing with the P-method, the TDA-method not only benefited more from the AGD, but is also faster due to its much lower complexity.

(a) Original (b) Blurred (18.84dB) (c) P without AGD (35.56dB)

(d) TDA without AGD (29.71dB) (e) P with MGD (36.45dB) (f) TDA with NAG (34.76dB)

Figure 14: 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.

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 4.2.1 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 4.2.1. We have excluded LoG filter because all methods with AGD failed to reverse this filter. We recorded the input PSNR value produced by each filter and the PSNR obtained after applying each reverse filter method after 50 iterations. The results for the TDA-, P- and T-method are shown in Tables 5, 6 and 7 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 5 shows that, for all smoothing filters, applying AGD to TDA-method results in improved image quality. In addition, Tables 6 and 7 show that, in some cases, reversing nonlinear filters such as AMF and RTV, the T- and P-method do not benefit from using the AGD.

4.4 Limitation

We provide a case study in which all algorithms failed to reverse the effect of a simple median filter. 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 15 and 16 which show that the PSNR decreases after each iteration and the visual quality of the image is also decreasing.

Table 5: 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**

Table 6: 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 7: 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

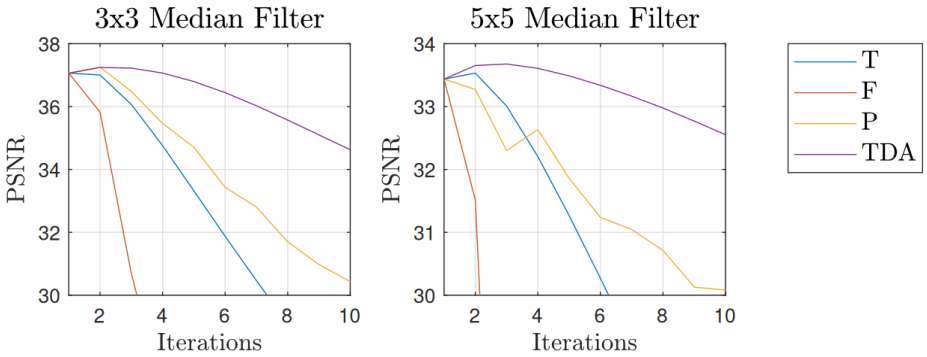
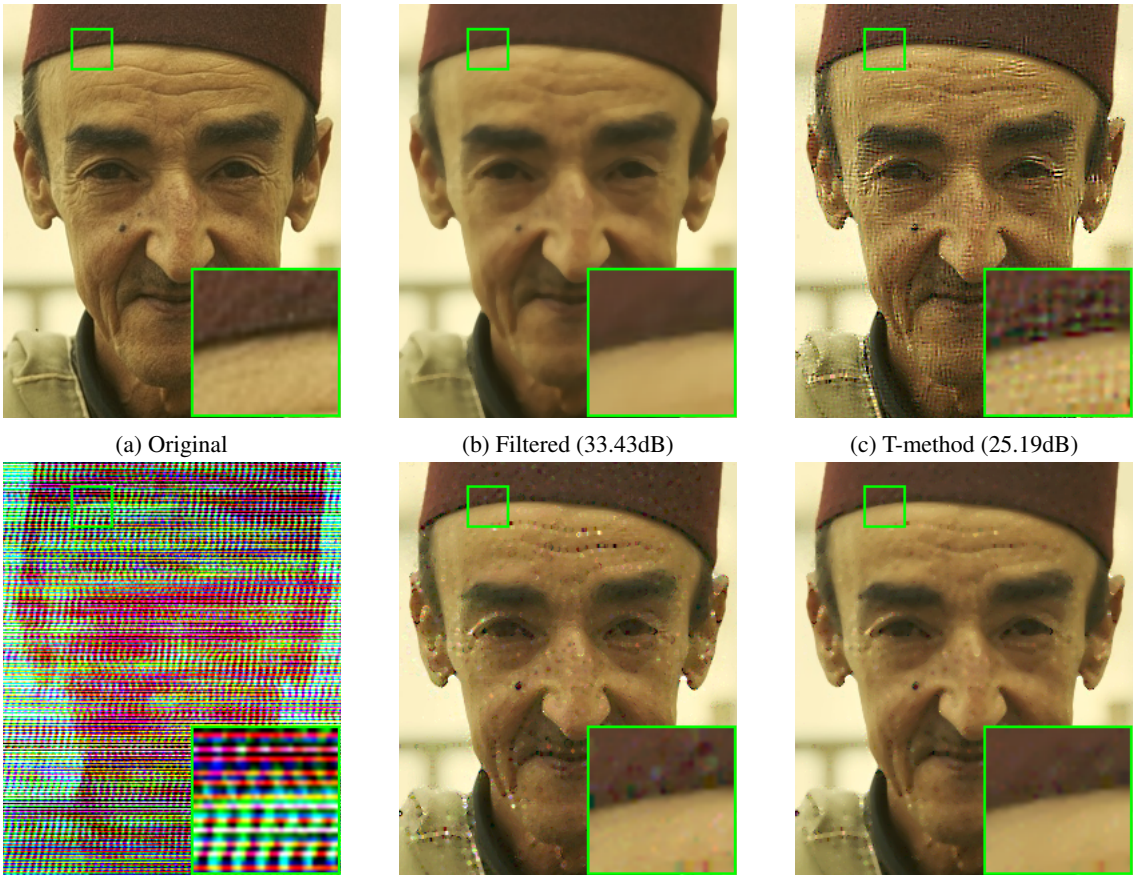


Figure 15: Values of PSNR per iteration obtained when reversing a Median filter using the T-, F-, P- and TDA-methods. (a) 3x3 Median filter. (b) 5x5 Median filter.



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

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

Figure 16: 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.

5 Conclusions

In this paper, we presented a new solution to a relatively new image inverse problem in which image filter to be reversed is assumed as an available black box. The development of our solution is based on formulating a local patch-based minimization problem and solving it using gradient descent. The problem of unknown gradient due to the unknown filter is solved by using a total derivative based gradient approximation. We study issues related to convergence and output image quality for the case of a linear low pass filter. Results provide new insights into 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 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 with state-of-the-art we have demonstrated that the proposed method has achieved the best trade-off in terms of computational complexity and effectiveness. The proposed method is of the same complexity as the T-method which is the fastest method, and it is more effective than the T-method in that it can reverse a larger number of filters. The proposed is as effective as that of the P-method in that they can reverse the same set of filters, and it is of much lower complexity than the P-method.

The failure of all gradient-free algorithms to reverse the effect of a median filter calls for further investigation into the assumptions being made in the algorithmic development and possible ways to develop new algorithms which are capable of reversing this family of filters, including order statistics filters. Another possible direction of future work is to combine gradient-free algorithm with a deep neural network which can reverse the effect of not only one filter, but a wide range of filters. The idea of algorithm unrolling [47] can be used.

References

[1] R. Gonzalez, R. Woods, and S. Eddins, *Digital Image Processing Using MATLAB*. 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, 2010.

[3] Z. Li and J. Zheng, "Single image de-hazing using globally guided image filtering," *IEEE Trans. Image Process.*, vol. 27, no. 1, pp. 442–450, 2017.

[4] A. Buades, B. Coll, and J. . Morel, "A non-local algorithm for image denoising," in *Proc. CVPR*, vol. 2, pp. 60–65, IEEE, 2005.

[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, 2007.

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

[7] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. ICCV*, pp. 839–846, IEEE, 1998.

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

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

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

[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, 2014.

[13] Y. Endo, Y. Kanamori, and J. Mitani, "Deep reverse tone mapping," *ACM Trans. Graph. (Proc. of SIGGRAPH ASIA 2017)*, vol. 36, Nov. 2017.

[14] J. Koh, J. Lee, and S. Yoon, "Single-image deblurring with neural networks: A comparative survey," *Comput. Vis. Image Underst.*, vol. 203, p. 103134, 2021.

[15] A. C. Kak and M. Slaney, *Principles of Computerized Tomographic Imaging*. IEEE Press, 1988.

[16] J. Romberg, "Imaging via compressive sampling," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 14–20, 2008.

[17] X. Tao, C. Zhou, X. Shen, J. Wang, and J. Jia, "Zero-order reverse filtering," in *Proc. ICCV*, pp. 222–230, IEEE, 2017.

[18] P. Milanfar, "Rendition: Reclaiming what a black box takes away," *arXiv preprint arXiv:1804.08651*, 2018.

[19] A. G. Belyaev and P.-A. Fayolle, "Two iterative methods for reverse image filtering," *Signal, Image and Video Process.*, pp. 1–9, 2021.

[20] L. Dong, J. Zhou, C. Zou, and Y. Wang, "Iterative first-order reverse image filtering," in *Proceedings of the ACM Turing Celebration Conference-China*, pp. 1–5, 2019.

[21] Q. Zhang, X. Shen, L. Xu, and J. Jia, "Rolling guidance filter," in *Proc. ECCV*, pp. 815–830, Springer, 2014.

[22] E. S. Gastal and M. M. Oliveira, "Adaptive manifolds for real-time high-dimensional filtering," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 1–13, 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, 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*. The MIT Press, 2019.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. The MIT Press, 2016.

[28] J. M. Ortega and W. C. Rheinboldt, *Iterative Solution of Nonlinear Equations in Several Variables*. SIAM, 2000.

[29] G. Deng and P. Broadbridge, "Bregman inverse filter," *Electron. Lett.*, vol. 55, no. 4, pp. 192–194, 2019.

[30] N. Weaver, *Lipschitz Algebras*. World Scientific Publishing Co. Pte. Ltd., 2018.

[31] M. Delbracio, I. Garcia-Dorado, S. Choi, D. Kelly, and P. Milanfar, "Polyblur: Removing mild blur by polynomial reblurring," *IEEE Trans. Comput. Imaging*, vol. 7, pp. 837–848, 2021.

[32] B. T. Polyak, "Minimization of unsmooth functionals," *USSR Computational Mathematics and Mathematical Physics*, vol. 9, no. 3, pp. 14–29, 1969.

[33] J. Steffensen, "Remarks on iteration," *Scandinavian Actuarial Journal*, vol. 1933, no. 1, pp. 64–72, 1933.

[34] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *Ussr Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, 1964.

[35] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate o (1/kˆ 2)," in *Dokl. Akad. Nauk Sssr*, vol. 269, pp. 543–547, 1983.

[36] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop, coursera: Neural networks for machine learning," *University of Toronto, Technical Report*, 2012.

[37] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[39] G. Zhai and X. Min, "Perceptual image quality assessment: a survey," *Sci. China Inf. Sci.*, vol. 63, pp. 211301:1–211301:52, 2020.

[40] 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. ICCV*, vol. 2, pp. 416–423, IEEE, 2001.

[41] L. Xu, Q. Yan, Y. Xia, and J. Jia, "Structure extraction from texture via relative total variation," *ACM Trans. Graph. (TOG)*, vol. 31, no. 6, pp. 1–10, 2012.

[42] G. Deng, F. Galetto, M. Al-nasrawi, and W. Waheed, "A guided edge-aware smoothing-sharpening filter based on patch interpolation model and generalized gamma distribution," *IEEE Open Journal of Signal Process.*, vol. 2, pp. 119–135, 2021.

[43] W. H. Richardson, "Bayesian-based iterative method of image restoration," *J. Acoust. Soc. Am.*, vol. 62, no. 1, pp. 55–59, 1972.

[44] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-laplacian priors," *Advances in Neural Information Processing Systems*, vol. 22, pp. 1033–1041, 2009.

[45] N. Wiener *et al.*, *Extrapolation, interpolation, and smoothing of stationary time series: with engineering applications*, vol. 8. The MIT Press, 1964.

[46] 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*, pp. 389–402, Springer, 1995.

[47] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Process. Mag.*, vol. 38, no. 2, pp. 18–44, 2021.