Paper

Backward Filtering of Images by Iterating Forward Filters

AN-SHUI YU* Non-member, NAOKI ONO* Member
 KOHEI INOUE* Member, KENJI HARA* Non-member
 KIICHI URAHAMA†* Member

(Received October 1, 2015, revised December 1, 2015)

Abstract: Given a filtering operation for images, we consider a problem for inverting it. If the algorithm of the original, i.e. forward filter is explicitly expressed, its inversion is tractable. However, if its algorithm is unknown exactly, it is hard to be reversed. We present, in this paper, a simple solution for this problem only by iterating the forward filter. This method can be used for a wide class of image filters including ones built in photo-retouch softwares. Examples are shown for smoothing, sharpening, halftoning and resizing filters.

Keywords: Inverse filter, Iterative solution method, Contractive mapping theorem, Built-in filter

1. Introduction

Major classes of image filtering operation are smoothing and sharpening. The smoothing filter includes Gaussian, bilateral[1], non-local means[2], median filters and so on. Typical example of sharpening filter is the unsharp masking. In this paper, we consider the inverse of these filters. Except for linear filters such as the Gaussian filter, their inversion is not so simple. Furthermore, many filters implemented in general-purpose photo-retouch softwares such as the Photoshop and the Paintshop are executed by built-in programs of which algorithm is unknown for users. This closedness inhibits their inversion.

A method[3] has been presented for inverting some filters such as the bilateral filter. We, in this paper, present a similar simple method for general image filters and analyzed the behavior of the method. In this paper, a scalar is denoted by a lower-case letter, vectors by thick lower-case letters and matrices by upper-case letters.

2. Fundamental Theory

Let $\mathbf{x}$ be a vector of pixel values in an image and $\mathbf{y}$ be a vector outputted by a filter as is illustrated in Fig. 1 where we call the ordinary filter “forward filter” in contrast to its reverse called “backward filter”. The vectors $\mathbf{x}$ and $\mathbf{y}$ satisfy an equation $\mathbf{a}(\mathbf{x}, \mathbf{y}) = \mathbf{0}$ where $\mathbf{0}$ is the zero vector. In this paper, we classify $\mathbf{a}(\mathbf{x}, \mathbf{y})$ to the following three classes:

- type 1: $\mathbf{a}(\mathbf{x}, \mathbf{y})$ is nonlinear with respect to both of $\mathbf{x}$ and $\mathbf{y}$.
- type 2: $\mathbf{a}(\mathbf{x}, \mathbf{y}) = \mathbf{y} - \mathbf{b}(\mathbf{x})$, i.e. linear with respect to $\mathbf{y}$. The function $\mathbf{b}(\mathbf{x})$ takes nonnegative values because $\mathbf{y}$ is pixel values.
- type 3: $\mathbf{a}(\mathbf{x}, \mathbf{y}) = \mathbf{x} - \mathbf{c}(\mathbf{y})$ where $\mathbf{c}(\mathbf{y}) \geq \mathbf{0}$.

2.1 Iterative Methods Iterative methods are commonly used for solving nonlinear equations. Iterations are

Figure 1: Filter and its reverse.

generally expressed by $\mathbf{x}^{(\xi+1)} = \mathbf{f}(\mathbf{x}^{(\xi)})$ where ξ is the iteration counter $\xi = 0, 1, 2, \dots$ and $\mathbf{x}^{(\xi)}$ is called ξ th iterant. General theorems for their convergence are the Liapunov's theorem and the contractive mapping theorem[4], in which the latter one is mainly concerned in this paper except for the mode filter[5] for which the Liapunov's theorem is resorted. We use simple iterative methods for solving the equation $\mathbf{a}(\mathbf{x}, \mathbf{y}) = \mathbf{0}$ as follows.

type 1:

forward filter :

we convert $\mathbf{a}(\mathbf{x}, \mathbf{y}) = \mathbf{0}$ to $\mathbf{y} = \mathbf{y} - k\mathbf{a}(\mathbf{x}, \mathbf{y})$ where $k > 0$. The iteration is written by

$$\mathbf{y}^{(\xi+1)} = \mathbf{y}^{(\xi)} - k\mathbf{a}(\mathbf{x}, \mathbf{y}^{(\xi)}) \quad (1)$$

backward filter :

We convert $\mathbf{a}(\mathbf{x}, \mathbf{y}) = \mathbf{0}$ to $\mathbf{x} = \mathbf{x} + k\mathbf{a}(\mathbf{x}, \mathbf{y})$ for which iteration is

$$\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} + k\mathbf{a}(\mathbf{x}^{(\xi)}, \mathbf{y}) \quad (2)$$

type 2:

forward filter :

The equation $\mathbf{a}(\mathbf{x}, \mathbf{y}) = \mathbf{y} - \mathbf{b}(\mathbf{x}) = \mathbf{0}$ leads to $\mathbf{y} = \mathbf{b}(\mathbf{x})$ which needs no iteration.

† Corresponding author: urahama@design.kyushu-u.ac.jp

* Department of Communication Design Science, Kyushu University
 4-9-1, Shiobaru, Minami-ku, Fukuoka 815-8540, Japan

backward filter :

We convert $\mathbf{b}(\mathbf{x}) = \mathbf{y}$ to $\mathbf{x} = \mathbf{x} - k[\mathbf{b}(\mathbf{x}) - \mathbf{y}]$ and solve it with the iteration

$$\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} - k[\mathbf{b}(\mathbf{x}^{(\xi)}) - \mathbf{y}] \quad (3)$$

type 3: In this type, $\mathbf{x}$ and $\mathbf{y}$ are exchanged from the above type 2. The forward filter needs the iteration $\mathbf{y}^{(\xi+1)} = \mathbf{y}^{(\xi)} - k[\mathbf{c}(\mathbf{y}^{(\xi)}) - \mathbf{x}]$, while the backward filter is explicitly expressed as $\mathbf{x} = \mathbf{c}(\mathbf{y})$.

Some examples of these three types and their convergence are presented in the following sections where the role of the coefficient k is explained.

3. Smoothing Filters

Let a value of pixel (i, j) in an input image be x_{ij} , and an output of a filter be y_{ij} . Arrays of these pixel values are $\mathbf{x} = [x_{ij}]$ and $\mathbf{y} = [y_{ij}]$. Some filters are derived from a statistical estimation as

$$y_{ij}^* = \arg \min_{y_{ij}} d(\mathbf{x}, y_{ij}) \quad (4)$$

3.1 Mode Filter In the mode filter[5], the objective function in eq.(4) is

$$d(\mathbf{x}, y_{ij}) = - \sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij} - x_{i+l, j+m})^2} \quad (5)$$

where $s_{lm} = e^{-\alpha(l^2+m^2)}$. There exists a unique solution for eq.(4) because this $d(\mathbf{x}, y_{ij})$ is convex downward with respect to y_{ij} and is given by

$$\begin{aligned} a(\mathbf{x}, y_{ij}) &= \frac{\partial d(\mathbf{x}, y_{ij})}{\partial y_{ij}} / \sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij} - x_{i+l, j+m})^2} \\ &= y_{ij} - \frac{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij} - x_{i+l, j+m})^2} x_{i+l, j+m}}{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij} - x_{i+l, j+m})^2}} = 0 \end{aligned} \quad (6)$$

This $a(\mathbf{x}, y_{ij})$ is nonlinear with respect to both of $\mathbf{x}$ and y_{ij} , hence this mode filter is of type 1 mentioned above.

forward filter :

If we set $k = 1$, eq.(1) leads to

$$y_{ij}^{(\xi+1)} = \frac{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij}^{(\xi)} - x_{i+l, j+m})^2} x_{i+l, j+m}}{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij}^{(\xi)} - x_{i+l, j+m})^2}} \quad (7)$$

which converges for arbitral initial values $y_{ij}^{(0)}$ because $d(\mathbf{x}, y_{ij}^{(\xi)})$ decreases monotonically[4], i.e. $d(\mathbf{x}, y_{ij}^{(\xi)})$ is the Liapunov's function for this iteration.

backward filter :

Similarly, by setting $k = 1$, eq.(2) becomes

$$x_{ij}^{(\xi+1)} = x_{ij}^{(\xi)} + y_{ij} - \frac{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij} - x_{i+l, j+m}^{(\xi)})^2} x_{i+l, j+m}^{(\xi)}}{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij} - x_{i+l, j+m}^{(\xi)})^2}} \quad (8)$$

which is written in a vector form as $\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} + \mathbf{y} - H(\mathbf{x}^{(\xi)}, \mathbf{y})\mathbf{x}^{(\xi)} = [I - H(\mathbf{x}^{(\xi)}, \mathbf{y})]\mathbf{x}^{(\xi)} + \mathbf{y}$ where I is the identity matrix and $H(\mathbf{x}, \mathbf{y})$ is the coefficient matrix which is positive definite and its eigenvalues satisfy $0 < \lambda \leq 1$. The eigenvalues of $I - H(\mathbf{x}^{(\xi)}, \mathbf{y})$ satisfy $0 \leq 1 - \lambda < 1$, hence the right hand side of eq.(8) is a contractive function of $\mathbf{x}^{(\xi)}$, which ensures the convergence of eq.(8) for arbitral initial values $\mathbf{x}^{(0)}$ [4].

3.2 Bilateral Filter In the bilateral filter (BF)[1], the objective function in eq.(4) is

$$d(\mathbf{x}, y_{ij}) = \sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij} - x_{i+l, j+m})^2} (y_{ij} - x_{i+l, j+m})^2 \quad (9)$$

from which

$$\begin{aligned} a(\mathbf{x}, y_{ij}) &= \frac{\partial d(\mathbf{x}, y_{ij})}{\partial y_{ij}} / \sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij} - x_{i+l, j+m})^2} \\ &= y_{ij} - \frac{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij} - x_{i+l, j+m})^2} x_{i+l, j+m}}{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij} - x_{i+l, j+m})^2}} = 0 \end{aligned} \quad (10)$$

forward filter :

This function $a(\mathbf{x}, y_{ij})$ is linear with respect to y_{ij} , hence this BF is of type 2. From eq.(10), we obtain

$$y_{ij}^* = \frac{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij} - x_{i+l, j+m})^2} x_{i+l, j+m}}{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij} - x_{i+l, j+m})^2}} \quad (11)$$

backward filter :

If we set $k = 1$, eq.(3) leads to

$$x_{ij}^{(\xi+1)} = x_{ij}^{(\xi)} - \frac{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij}^{(\xi)} - x_{i+l, j+m}^{(\xi)})^2} x_{i+l, j+m}^{(\xi)}}{\sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(x_{ij}^{(\xi)} - x_{i+l, j+m}^{(\xi)})^2}} + y_{ij} \quad (12)$$

which converges because its right hand side is a contractive function of $\mathbf{x}^{(\xi)}$. This convergence is ensured for an arbitral initial value, while we set $x_{ij}^{(0)} = y_{ij}$ for avoiding wasteful computation.

Figure 2: Frequency response of iterant $\mathbf{x}^{(\xi)}$.

Equation (12) is written in a vector form as

$$\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} - H(\mathbf{x}^{(\xi)})\mathbf{x}^{(\xi)} + \mathbf{y} \quad (13)$$

We analyze the convergence process of this iteration. For simplifying the analysis, we approximate $H(\mathbf{x})$ to constant H and simplifying eq.(13) to

$$\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} - H\mathbf{x}^{(\xi)} + H\mathbf{x} \quad (14)$$

with substitution of $\mathbf{y} = H\mathbf{x}$. From the graph spectral analysis[6], the frequency of signals is given by $\omega = 1 - \lambda$ where λ is the eigenvalue of H . From a straightforward manipulation of eq.(14), the frequency response of $\mathbf{x}^{(\xi)}$ is derived as $1 - \omega^{\xi+1}$ which is plotted for first three iterants $\mathbf{x}^{(0)}$, $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ in Fig. 2. These graphs reveal that each $\mathbf{x}^{(\xi)}$ is a low-pass filter which converges to an all-pass filter except for the highest frequency $\omega = 1$ where the response is always fixed to zero. Thus it is confirmed that the iteration of eq.(14) converges to the original $\mathbf{x}$ inputted to the forward filter in Fig. 1, i.e. $\mathbf{x}^{(\infty)} = \mathbf{x}$.

Note the difference between the iteration of eq.(13) and the repeated unsharp masking:

$$\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} - H(\mathbf{x}^{(\xi)})\mathbf{x}^{(\xi)} + \mathbf{x}^{(\xi)} = 2\mathbf{x}^{(\xi)} - H(\mathbf{x}^{(\xi)})\mathbf{x}^{(\xi)} \quad (15)$$

which diverges because the eigenvalue of $2I - H(\mathbf{x}^{(\xi)})$ is $2 - \lambda$ which is larger than 1. Fixation of y_{ij} in eq.(12) is prerequisite for the convergence of $\mathbf{x}^{(\xi)}$. We will consider the unsharp masking again in section 4.

Equation (11) is the standard BF. If we set the denominator in eq.(11) to a constant value, we obtain a new type of BF called the unnormalized BF[7].

3.3 Regularized Nonlinear Diffusion Filter The regularized nonlinear diffusion filter (RNDF)[8] is defined by

$$d(x_{ij}, \mathbf{y}) = (x_{ij} - y_{ij})^2 - \phi \sum_{l=-p}^p \sum_{m=-p}^p s_{lm} e^{-\beta(y_{ij} - y_{i+l, j+m})^2} \quad (16)$$

Figure 3: Regularized nonlinear diffusion filter.

Figure 4: Inversion of nonlinear smoothing filter.

from which we get

$$\begin{aligned} a(x_{ij}, \mathbf{y}) &= \partial d(x_{ij}, \mathbf{y}) / \partial y_{ij} \\ &= y_{ij} - x_{ij} + \beta \phi \sum_{l=-p}^p \sum_{m=-p}^p s_{lm} (y_{ij} - y_{i+l, j+m}) e^{-\beta(y_{ij} - y_{i+l, j+m})^2} \end{aligned} \quad (17)$$

which is linear with respect to x_{ij} , hence this RNDF is of type 3.

forward filter :

From eq.(17), $a(x_{ij}, \mathbf{y}) = 0$ leads to the iteration

$$y_{ij}^{(\xi+1)} = \frac{x_{ij} + \beta \phi \sum_{l,m \neq 0} s_{lm} y_{i+l, j+m}^{(\xi)} e^{-\beta(y_{ij}^{(\xi)} - y_{i+l, j+m}^{(\xi)})^2}}{1 + \beta \phi \sum_{l,m \neq 0} s_{lm} e^{-\beta(y_{ij}^{(\xi)} - y_{i+l, j+m}^{(\xi)})^2}} \quad (18)$$

where $\sum_{l,m \neq 0}$ means the sum for l and m skipping the center pixel where $l = 0$ and $m = 0$. An output of this RNDF for image in Fig. 3(a) is shown in Fig. 3(b) where we set $p = 5, \alpha = 0.01, \beta = 0.01, \phi = 0.02$.

backward filter :

From $a(\mathbf{x}, y_{ij}) = 0$ in eq.(17), we obtain

$$x_{ij} = y_{ij} + \beta \phi \sum_{l=-p}^p \sum_{m=-p}^p s_{lm} (y_{ij} - y_{i+l, j+m}) e^{-\beta(y_{ij} - y_{i+l, j+m})^2} \quad (19)$$

by which we can get the image in Fig. 3(a) from Fig. 3(b). Equation (19) is a sharpening filter called

Figure 5: Inversion of anisotropic smoothing filter.

Figure 6: Inverse median filter for noisy $\mathbf{x}$.

a weighted unsharp masking induced from the above mentioned unnormalized BF. Standard sharpening filters are analyzed below in section 4.

3.4 Built-in Filters In addition to the above filters, many filters such as the median filter and the non-local means filter are of type 2. We can also use similar filters implemented in photo-retouch software or online at a webpage. Among them, we experimented a nonlinear smoothing filter in a photo-retouch software “Paintshop”. Its output $\mathbf{y}$ is shown in Fig. 4(b) for the input image $\mathbf{x}$ in Fig. 4(a) which is an ear of Koala. Iterants in the iteration of eq.(3) are shown in Fig. 4(c) and (d). The iterant $\mathbf{x}^{(\epsilon)}$ sufficiently converges at the fifth iteration. Another example of smoothing filter is shown in Fig. 5 where Fig. 5(b) is an output of the anisotropic filter[9] for Fig. 5(a). The algorithm in this filter is too complex to be analytically inverted. Iterants in the iteration of eq.(3) are shown in Fig. 5(c) and (d). This filter smoothes images strongly, hence its inversion needs more iteration than Fig. 4.

Figure 7: Inversion of image enhancer.

3.5 Finally Converged Value The iteration converges for all of these filters. The iterant $\mathbf{x}^{(\epsilon)}$ almost converges to the original $\mathbf{x}$, thus the filters can be inverted. In some highly nonlinear filters, however, the final iterant $\mathbf{x}^{(\infty)}$ deviates from the original $\mathbf{x}$. This is because such filters remove some components in $\mathbf{x}$ completely. If we decompose $\mathbf{x}$ into a robust signal component $\mathbf{x}_r$ and a fragile component $\mathbf{x}_f$ as $\mathbf{x} = \mathbf{x}_r + \mathbf{x}_f$, then $\mathbf{b}(\mathbf{x}_f) = \mathbf{0}$ hence $\mathbf{b}(\mathbf{x}) = \mathbf{b}(\mathbf{x}_r + \mathbf{x}_f) = \mathbf{b}(\mathbf{x}_r)$. The iteration for inverting the forward filter $\mathbf{y} = \mathbf{b}(\mathbf{x})$ converges to $\mathbf{x}^{(\infty)} = \mathbf{x}_r$. An image with addition to 30% random noise to Fig. 3(a) is shown in Fig. 6(a) for which output of the median filter of 5×5 window is shown in Fig. 6(b). Noise in Fig. 6(a) is almost removed while some textures also disappear. Iterants are shown in Fig. 6(c),(d),(e) where the lost textures are gradually restored while noise still remains removed. Thus this iteration method is useful for removal of noise from images while maintaining textures.

4. Sharpening Filters

As is seen above, the coefficient k in the iterations in section 2.1 can be safely set to 1 for ensuring their convergence for smoothing filters. This is intrinsically owing to the contractiveness of these filters. Sharpening filters, however, does not satisfy this condition as is explained in this section. Typical sharpening filter is the unsharp masking (UM), e.g. if we write eq.(11) in a vector form as $\mathbf{y} = H(\mathbf{x})\mathbf{x}$, the unsharp masking derived from it is expressed by

$$\mathbf{y} = \mathbf{x} + \mu[\mathbf{x} - H(\mathbf{x})\mathbf{x}] = [(1 + \mu)I - \mu H(\mathbf{x})]\mathbf{x} \quad (20)$$

with $\mu > 0$. The eigenvalue of $(1 + \mu)I - \mu H(\mathbf{x})$ is $1 + \mu - \mu\lambda$ where λ is the eigenvalue of $H(\mathbf{x})$. This eigenvalue $1 + \mu - \mu\lambda$ exceeds 1 because $0 < \lambda \leq 1$. this eigenvalue larger than 1 means that this filter magnifies high frequency components in $\mathbf{x}$, i.e. UM is a sharpening filter. This UM is of type 2

Figure 8: Inverse halftoning.

Figure 10: Iterative regularization.

Figure 9: Inverse resizing.

and eq.(3) reads

$$\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} - k[(1 + \mu)I - \mu H(\mathbf{x}^{(\xi)})]\mathbf{x}^{(\xi)} + k\mathbf{y} \quad (21)$$

which converges if $-1 \leq 1 - k(1 + \mu - \mu\lambda) \leq 1$ which holds if $0 < k \leq 2/(1 + \mu)$. The iterant $\mathbf{x}^{(\xi)}$ converges with oscillation if $-1 \leq 1 - k(1 + \mu - \mu\lambda) \leq 0$. We cannot stop such oscillating iteration at its midway due to errors. Hence $0 \leq 1 - k(1 + \mu - \mu\lambda) \leq 1$ is required for monotonic convergence which can be safely stopped at any midpoint. Therefore, $0 < k \leq 1/(1 + \mu)$ is desired for smooth convergence.

Various sharpening filters are also included in photo-

retouch software. The magnification factor is unknown for those built-in filters, for such filters k should be safely set to a small value, typically $k = 0.5$.

An example of foggy image as $\mathbf{x}$ is shown in Fig. 7(a) and its enhanced output $\mathbf{y}$ by an online software “One-Shot-Enhancer” in the webpage <http://www.picturetopeople.org> is shown in Fig. 7(b). Iterants $\mathbf{x}^{(\xi)}$ are shown in Fig. 7(c) and (d) where we set $k = 0.5$.

5. Relaxed Iterative Methods

The above smoothing and sharpening are fundamental processing of images, hence its inversion is not so difficult. However, if filters are more advanced, the iteration formula should be extended.

5.1 Quantizing Filters By quantizing $\mathbf{x}$ to discrete levels, $\mathbf{y}$ takes discrete values, extremely only two levels, black and white. For such quantization, halftoning techniques such as the error diffusion are used. Inversion of such quantizing filters becomes an integer programming (IP) problem which is hard to solve. We approximate it by a nonlinear programming (NP) problem by smoothing the discrete image $\mathbf{y}$.

Let a quantizing operation be denoted by $\mathbf{y} = \mathbf{q}(\mathbf{x})$ which is rewritten as $\mathbf{q}(\mathbf{x}) - \mathbf{y} = \mathbf{0}$. We relax this equation to $\mathbf{s}(\mathbf{q}(\mathbf{x}) - \mathbf{y}) = \mathbf{0}$ where $\mathbf{s}(\cdot)$ is a smoothing filter. Next we convert $\mathbf{s}(\mathbf{q}(\mathbf{x}) - \mathbf{y}) = \mathbf{0}$ to $\mathbf{x} = \mathbf{x} - k[\mathbf{s}(\mathbf{q}(\mathbf{x}) - \mathbf{y})]$. The iteration

$$\mathbf{x}^{\xi+1} = \mathbf{x}^{\xi} - k[\mathbf{s}(\mathbf{q}(\mathbf{x}^{\xi}) - \mathbf{y})] \quad (22)$$

needs no complex computation demanded for many inverse halftoning methods[10] [11] even without explicit programming by utilizing a photo retouch software. We experimented this iteration for the images in Fig. 8 where (a) is an input grayscale image $\mathbf{x}$ and (b) is $\mathbf{y}$ with the use of the

Floyd-Steinberg error diffusion as $\mathbf{q}(\cdot)$. Some iterants are shown in Fig. 8 (c),(d) where we used the Gaussian filter as $\mathbf{s}(\cdot)$.

5.2 Resizing Filters When $\mathbf{r}(\cdot)$ is a resizing operation, $\mathbf{y} = \mathbf{r}(\mathbf{x})$ cannot be directly inverted because image size is different for $\mathbf{x}$ and $\mathbf{y}$. Similarly to the above quantization, we relax $\mathbf{r}(\mathbf{x}) - \mathbf{y} = \mathbf{0}$ to $\mathbf{b}(\mathbf{r}(\mathbf{x}) - \mathbf{y}) = \mathbf{0}$ where $\mathbf{b}(\cdot)$ is the inverse resizing of $\mathbf{r}(\cdot)$, i.e. if $\mathbf{r}(\cdot)$ changes the size $M \times N$ to $M' \times N'$, $\mathbf{b}(\cdot)$ alters $M' \times N'$ to $M \times N$. We convert $\mathbf{b}(\mathbf{r}(\mathbf{x}) - \mathbf{y}) = \mathbf{0}$ to $\mathbf{x} = \mathbf{x} - \mathbf{b}(\mathbf{r}(\mathbf{x}) - \mathbf{y})$ and solve it by the iteration $\mathbf{x}^{\xi+1} = \mathbf{x}^{\xi} - \mathbf{b}(\mathbf{r}(\mathbf{x}^{\xi}) - \mathbf{y})$. This iteration resembles the back-projection method for image super-resolution[12] [13].

An example image shown in Fig.9(a) is shrunk to Fig.9(b) of which width and height are 1/4 of those of Fig.9(a). Jaggies obstruct the perceptual quality of Fig.9(b). Enlargement of Fig.9(b) to Fig.9(a) is an example called super-resolution. Iterants $\mathbf{x}^{(\xi)}$ are shown in Fig.9(c) and (d) where we set $k = 0.5$. We used the flow filter in Fig. 5 for interpolating enlarged images. Jaggies are weakened in Fig. 9(d).

5.3 Iterative Regularization Equation (22) is effective even when $\mathbf{q}(\mathbf{x}) = \mathbf{x}$. Image denoising is an operation for recovering $\mathbf{x}$ from $\mathbf{y} = \mathbf{x} + \mathbf{n}$ where $\mathbf{n}$ is noise. A conventional way of denoising is smoothing of $\mathbf{y}$ and outputting $\mathbf{s}(\mathbf{y})$ where $\mathbf{s}(\cdot)$ is a smoothing filter. This smoothing removes noise but also smoothes out signal components slightly.

A method called the iterative regularization[14] restores lost signal components by iterating

$$\mathbf{x}^{(\xi+1)} = \mathbf{x}^{(\xi)} + \mathbf{s}(\mathbf{y} - \mathbf{x}^{(\xi)}) \quad (23)$$

where its initial value is set to $\mathbf{x}^{(0)} = \mathbf{s}(\mathbf{y})$. Note that this iteration should be stopped at its midway because $\mathbf{x}$ approaches gradually to $\mathbf{y}$ and eventually $\mathbf{x}^{(\infty)} = \mathbf{y}$.

Addition of Gaussian noise of standard deviation 10 to the image in Fig.3(a) yields Fig.10(a). Blurring this Fig. 10(a) by a Gaussian filter, we get Fig. 10(b) where noise is reduced while textures disappear slightly. The iterant $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(5)}$ of eq.(23) are shown in Fig.10(c),(d) where lost textures are partially recovered. We set $\mathbf{s}(\cdot)$ as the Gaussian filter of the standard deviation 3.

6. Conclusion

We have presented a method for inverting image filters by simply iterating them. An outstanding benefit of the proposed method lies in its applicability to wide class of filters including built-in softwares of which algorithm is unknown exactly. We have experimented this method for smoothing and enhancing filters, image halftoning and resizing operations. Automatic setting rule of the coefficient in the iteration is under study for image sharpening and enhancing filters.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP16K00241.

References

- [1] C. Tomasi and R. Manduchi: "Bilateral filtering for gray and color images", *Proc. ICCV*, pp.839-846, 1998.
- [2] A. Buades, B. Coll and J. Morel: "Image denoising by non-local averaging", *Proc. ICASSP*, pp.25-28, 2005.
- [3] L. Hang, K. Inoue and K. Urahama: "Inverse cross-bilateral filter for cross-sharpening of images", *proc. GCCE*, pp.924-929, 2014.
- [4] J. Ortega and W. Rheinboldt: "Iterative solutions of non-linear equations in several variables", *Society for Industrial Mathematics*, 1987.
- [5] L. D. Griffin: "Mean, median and mode filtering of images", *Proc. R. Soc. Lond. A*, vol.456, pp.2995-3004, 2000.
- [6] A. Gadde, S. K. Narang and A. Ortega: "Bilateral filter: graph spectral interpretation and extensions", *Proc. ICIP*, pp.1222-1226, 2013.
- [7] P. Milanfar and H. T. Esfandarani: "A new class of image filters without normalization", *Proc. ICIP*, pp.3294-3298, 2016.
- [8] P. Perona and J. Malik: "Scale-space and edge detection using anisotropic diffusion", *IEEE Trans. Patt. Anal. Mach. Intell.*, vol.12, no.7, pp.629-639, 1990.
- [9] H. Kang, S. Lee and C. Chui: "Flow-based image abstraction", *IEEE Trans. Vis. Comput. Graph.*, vol.15, no.1, pp.62-76, 2009.
- [10] Z. Xiong, M. T. Orchard and K. Ramchandran: "Inverse halftoning using wavelets", *IEEE Trans. Image Process.*, vol. 8, no.10, pp.1479-1483, 1999.
- [11] J. Puzicha, M. Held, J. Ketterer, J. M. Buhmann and D. W. Fellner: "On spatial quantization of color images", *IEEE Trans. Image Process.* vol.9, no.4, pp.666-682, 2000.
- [12] M. Irani and S. Peleg: "Motion analysis for image enhancement: resolution, occlusion, and transparency", *J. Visual Comm., Image Repres.*, vol.4, no.4, pp.324-335, 1993.
- [13] S. Dai M. Han, Y. Wu and Y. Gong: "Bilateral back-projection for single image super resolution", *Proc. ICME*, pp.1039-1042, 2007.
- [14] M. R. Charest and P. Milanfar: "On iterative regularization and its applications", *IEEE Trans. Circ. Syst. Video Tech.*, vol.18, no.3, pp.406-411, 2008.

An-shui Yu (Non-member) He received M.F.A.degrees from Kyushu University in 2016. He is now studying for a doctor's degree in the Kyushu University. His research interests include omnidirectional spherical image and foreground enlargement.

Naoki Ono (Member) He was born in Saga, Japan, in 1961. He received a Ph.D. degree in engineers from Kyushu University in 1997, and is presently an associate professor at Kyushu University. He has worked on image processing and pattern recognition. He is member of IIAE and IEICE.

Kohei Inoue (Member) He received B.Des., M.Des. and D.Eng. degrees from Kyushu Institute of Design in 1996, 1998 and 2000, respectively. He is currently an Associate Professor in Kyushu University. His research interests include pattern recognition and image processing.

Kenji Hara (Non-member) He received the BE and ME degrees from Kyoto University in 1987 and 1989, respectively, and the PhD degree from Kyushu University in 1999. He is currently an Associate Professor in Kyushu University. His research interests include physics-based vision and geometric modeling.

Kiichi Urahama (Member) He received M.Eng. and D.Eng. degrees from Kyushu University in 1976 and 1980. From 1980 to 1995 he was an Associate Professor in Kyushu Institute of Technology. He is now a Professor in Kyushu University. His research interests include pattern recognition, image processing and computer graphics.