ICIC Express Letters
Volume **11**, Number **7**, July **2017**

 ISSN 1881-803X
pp. 1175–1183

EDGE-PRESERVING SMOOTHING BASED ON LOCAL GRADIENT-DOMAIN PROCESSING

Takanori Koga[1], Shota Furukawa[2], Noriaki Suetake[2] and Eiji Uchino[2]

[1]Department of Computer Science and Electronic Engineering
National Institute of Technology, Tokuyama College
Gakuendai, Shunan 745-8585, Japan
koga@tokuyama.ac.jp

[2]Graduate School of Sciences and Technology for Innovation
Yamaguchi University
Yoshida 1677-1, Yamaguchi 753-8512, Japan
{ t002wa; nsuetake; uchino }@yamaguchi-u.ac.jp

Received January 2017; accepted April 2017

Abstract. *As a technique for edge-preserving smoothing, gradient-domain processing is frequently used. Its operation is realized by modifying image gradients and then reconstructing the resulting image from the modified gradients. Generally, the reconstruction operation is realized by using an iterative or a direct method. The iterative methods might be unstable and affected by preconditioning, although they are frequently used and tend to be faster than the direct methods. On the other hand, generally, the direct methods require a huge memory space and are very time-consuming. Furthermore, it is difficult to apply them to large scale problems in spite of their stability and wide applicability. In this study, to cope with the problems, we propose a new edge-preserving smoothing filter which reconstructs the resulting image locally from the gradient-domain based on a direct method. The effectiveness of the proposed method is verified through a series of experiments regarding edge-preserving smoothing and noise removal using natural grayscale standard images.*

Keywords: Gradient-domain processing, Edge-preserving smoothing, Image gradient, Direct method

1. **Introduction.** Image smoothing is processing to obtain an image having solid contrasting density [1, 2]. As fundamental image smoothing methods, the mean filter and the Gaussian filter are frequently used. However, such simple smoothing methods not only smooth undesired contrasting density but also ruin the detailed part and edges in a processed image. In order to cope with this problem, edge-preserving smoothing has been studied [3, 4].

The bilateral filter is a representative edge-preserving smoothing filter [3]. This filter has two filtering weights; one is the spatial weight in the image space and the other is the range weight in the intensity space. Edge-preserving smoothing is realized by suppressing the smoothing effect of the filter if the differences between the intensity of a pixel of interest (POI) and those of its neighboring pixels are large.

On the other hand, edge-preserving smoothing method based on gradient-domain image processing such as Poisson image editing [5] was proposed and has been applied to various kinds of applications [6, 7, 8, 9]. In particular, as a simple and powerful edge-preserving smoothing, the gradient-domain filtering has been proposed [10]. This filtering method focuses on the gradient information; and it can also be applied to various kinds of image processing such as edge enhancement, line extraction and deletion. However, in the gradient-domain filtering, a resulting image in the original image space must be reconstructed from corresponding gradients by solving a Poisson's equation under a given

boundary condition. Generally, the reconstruction operation is realized by using an iterative method or a direct method. The iterative methods might be unstable and affected by preconditioning although they are frequently used and tend to be faster than the direct methods. On the other hand, generally, the direct methods require a huge memory space and are very time-consuming. Furthermore, it is difficult to apply them to large scale problems in spite of their stability and wide applicability.

In this study, to cope with those problems on memory space and computational costs of the direct methods, we propose a new edge-preserving smoothing filter which reconstructs the resulting image locally from the gradient-domain. Concretely, for edge-preserving smoothing, only the weak gradient is suppressed in the gradient-domain, and then the resulting image is reconstructed locally from the gradient information. Furthermore, in order to obtain the reconstructed image quickly, we solve the Poisson's equation by using a modified direct method suitable for the local processing. The proposed method makes users possible to take advantage of the features of the direct methods also in large-scale image processing. The effectiveness of the proposed method is verified through a series of experiments regarding edge-preserving smoothing and noise removal using natural grayscale standard images.

2. **Gradient-Domain Image Filtering.** In this section, the edge-preserving smoothing based on the gradient-domain image filtering [10] is explained. An image gradient is a directional change in the intensity or color in an image. It can be calculated as a difference value between a POI and its neighboring pixels. By letting a pixel value at a position (i, j) in an input image be $f(i, j)$, its gradient $\vec{g}(i, j)$ is calculated by the backward-difference as follows:

$$\vec{g}(i,j) = (g_i(i,j), g_j(i,j)) \ , \tag{1}$$

$$g_i(i,j) = f(i,j) - f(i,j-1) \ , \tag{2}$$

$$g_j(i,j) = f(i,j) - f(i-1,j) \ , \tag{3}$$

where $g_i(i, j)$ and $g_j(i, j)$ are horizontal and vertical gradients, respectively.

If the edge in the input image has a large variation in pixel value, then the gradient in the gradient-domain $|\vec{g}(i, j)|$ becomes a large value (i.e., strong gradient). On the other hand, regarding small fluctuations of contrast density, $|\vec{g}(i, j)|$ becomes a small value (i.e., weak gradient). Thus, the edge-preserving smoothing is realized by processing the gradient $\vec{g}(i, j)$ based on its degree of variation. Concretely, if $|\vec{g}(i, j)|$ is less than a threshold value th (i.e., it is weak gradient), its value is replaced by zeros. If $|\vec{g}(i, j)|$ is equal to or greater than th (i.e., strong gradient), its value is preserved. These are denoted by the following equation:

$$\vec{g}^*(i,j) = \begin{cases} \vec{g}(i,j) & \text{if } |\vec{g}(i,j)| \geq th \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where $|\cdot|$ denotes a Euclidean norm.

The output image is reconstructed from the obtained gradient-domain image $\vec{g}^*$ by solving a boundary value problem represented by a Poisson's equation. The Poisson's equation to be solved is described as follows:

$$\Delta \vec{f'} = \nabla \vec{g}^*, \tag{5}$$

$$\vec{f'}\big|_{b\Omega} = \vec{f}\big|_{b\Omega}, \tag{6}$$

where Ω is the target region, $b\Omega$ the boundary, and $\vec{f'}$ the output image reconstructed from a gradient image $\vec{g}^*$.

The intensities of the reconstructed image $\vec{f'}$ at the boundary $b\Omega$ are equal to those of the original image; and those are $\Delta \vec{f'} = \nabla \vec{g}^*$ in the processed region Ω. Here, $\Delta \vec{f'}$ is the

second-order partial derivatives (i.e., Laplacian) of the reconstructed image $\vec{f'}$ denoted as $\Delta\vec{f'} = \frac{\partial^2 \vec{f'}}{\partial i^2} + \frac{\partial^2 \vec{f'}}{\partial j^2}$. Similarly, $\nabla\vec{g}^*$ is denoted by $\nabla\vec{g}^* = \frac{\partial g_i^*}{\partial i} + \frac{\partial g_j^*}{\partial j}$. In this regard, $\nabla\vec{g}^*$ is calculated by using the forward difference because the gradient image $\vec{g}$ is calculated by using the backward difference. This Poisson's equation can be solved by an iterative method or a direct method [11].

2.1. **Iterative method.** The iterative methods are ways to obtain an approximate solution successively by iterative processing. The following is a concrete description of the successive over-relaxation (SOR) which is a variant of the Gauss-Seidel method for solving a linear system of equations. Now, by letting a pixel value at a position (i, j) of an output image be $f'(i, j)$, the solution of the Poisson's equation given by Equations (5) and (6) is calculated by the iterative expression of SOR as follows:

$$\begin{aligned} f'^{n+1}(i,j) = (1-\omega)f'^{n}(i,j) + \omega\,(f'^{n+1}(i,j-1) + f'^{n+1}(i-1,j) \\ + f'^{n}(i+1,j) + f'^{n}(i,j+1) - \nabla\vec{g}^*(i,j))\,/4, \end{aligned} \tag{7}$$

where n is an iteration count, ω the relaxation factor. Generally, ω is set to be from 1 to 2; and it is frequently set to be 1.95 to obtain good convergence properties. However, the iterative methods take much computational cost for solving the Poisson's equation due to the iterative calculation.

2.2. **Direct method.** In this subsection, as the other solution, the direct method is explained. Now, suppose that a two-dimensional image $\vec{f}$ with $m \times n$ pixels is made into a one-dimensional signal. Let the reshaped one-dimensional version of the image $\vec{f}$ be $\hat{\vec{f}}$ as follows:

$$\hat{f}(k) = f(i,j). \tag{8}$$

Here, the index k is denoted as follows:

$$k = i + m(j-1). \tag{9}$$

Now, the Laplacian of the image $\vec{f}$, i.e., $\Delta\vec{f}$, is denoted by using $\hat{\vec{f}}$ as follows:

$$\Delta\vec{f} = \vec{L}\hat{\vec{f}}, \tag{10}$$

where $\vec{L}$ is the square matrix having the filtering coefficients of the Laplacian filter with $m\times n$ rows and $m\times n$ columns.

$$\vec{L} = \begin{pmatrix} \vec{A} & 0 & \cdots & & 0 \\ \vec{C} & \vec{B} & \vec{C} & & \\ 0 & \vec{C} & \vec{B} & \vec{C} & \vdots \\ \vdots & & & \ddots & 0 \\ & & & \vec{C} & \vec{B} & \vec{C} \\ 0 & & & \cdots & 0 & \vec{A} \end{pmatrix}. \tag{11}$$

In Equation (11), $\vec{A}$, $\vec{B}$, and $\vec{C}$ are denoted as follows:

$$\vec{A} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & 1 \end{pmatrix}, \tag{12}$$

$$\vec{B} = \begin{pmatrix} 1 & 0 & \cdots & & & 0 \\ -1 & 4 & -1 & & & \\ 0 & -1 & 4 & -1 & & \vdots \\ \vdots & & & \ddots & & 0 \\ & & & -1 & 4 & -1 \\ 0 & & & \cdots & 0 & 1 \end{pmatrix}, \tag{13}$$

$$\vec{C} = \begin{pmatrix} 0 & 0 & \cdots & & 0 \\ 0 & -1 & & & \\ \vdots & & \ddots & & \vdots \\ & & & -1 & 0 \\ 0 & & \cdots & 0 & 0 \end{pmatrix}. \tag{14}$$

Here, $\vec{A}$, $\vec{B}$, and $\vec{C}$ are all $m \times m$ square matrices. $\vec{A}$ is an identity matrix; the diagonal elements of the first and m-th rows in $\vec{B}$ are 1, and the diagonal elements of the first and m-th rows in $\vec{C}$ are 0. This is because, as shown in Equation (6), it is taken into consideration that the values at the boundary are the same to those of the input image. From Equations (5) and (10), the reconstructed one-dimensional image $\hat{\vec{f}}'$ is denoted as follows:

$$\hat{\vec{f}}' = \vec{L}^{-1}\hat{\vec{G}}\,, \tag{15}$$

$$\hat{\vec{G}} = \begin{pmatrix} \nabla\vec{g}^*(1,1) \\ \nabla\vec{g}^*(2,1) \\ \vdots \\ \nabla\vec{g}^*(i,j) \\ \vdots \\ \nabla\vec{g}^*(m,n) \end{pmatrix}, \tag{16}$$

where $\hat{\vec{G}}$ is the one-dimensional version of $\nabla\vec{g}^*$ as a similar manner to Equation (8). In this regard, the size of $\vec{L}$ becomes the square of the image size; it becomes too large with increasing the size of the image. The inverse matrix calculation for such large-scale matrix requires too much computational memory; and it is difficult to obtain the inverse matrix with general purpose computers.

3. **Proposed Method.** The direct method requires too much memory to calculate the inverse matrix of a large-scale $\vec{L}$. To cope with this problem, we propose a local block-wise image reconstruction processing. The following is concrete explanation of the local reconstruction method.

Let the position of a target pixel be (i, j), a local region of an input image $\vec{f}_l$, the window size of the local region $2r+1$, the one-dimensional version of the region $\hat{\vec{f}}_l$, and the corresponding square matrix having the filtering coefficients of a Laplacian filter as shown in Equations (11)-(14) $\vec{L}_l$. Now, the Laplacian of the image $\vec{f}_l$, i.e., $\Delta\vec{f}_l$, is denoted by using $\hat{\vec{f}}_l$ as follows:

$$\Delta\vec{f}_l = \vec{L}_l\,\hat{\vec{f}}_l. \tag{17}$$

The reconstructed local one-dimensional image $\hat{\vec{f}}'_l$ is denoted as follows:

$$\hat{\vec{f}}'_l = \vec{L}_l^{-1}\hat{\vec{G}}_l, \tag{18}$$

$$\hat{\vec{G}}_l = \begin{pmatrix} \nabla\vec{g}^*(i-r, j-r) \\ \nabla\vec{g}^*(i-r+1, j-r) \\ \vdots \\ \nabla\vec{g}^*(i, j) \\ \vdots \\ \nabla\vec{g}^*(i+r, j+r) \end{pmatrix}. \tag{19}$$

Here, $\vec{L}_l$ is equal to the case of Equation (11) with the size of $(2r+1)^2$ rows and $(2r+1)^2$ columns. Thus, Equations (12)-(14), which are the elements of the coefficients, similarly, are also square matrices with $(2r+1)$ rows and $(2r+1)$ columns. This processing is applied to the entire image by using a sliding window manner as shown in Figure 1.

As shown in the figure, multiple processing results from the sliding windows are obtained for a POI. Those results for a POI are merged by calculating the mean value of the results as the final output value. In this regard, the values at boundary $b\Omega$ in the local regions are obtained by using only the values in the region Ω because those values are the same to those of the input image.

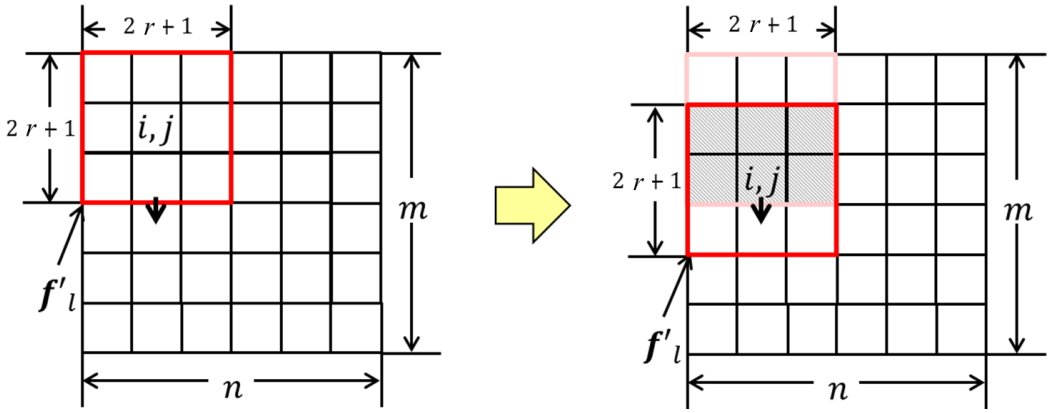


FIGURE 1. Filtering process with a sliding window manner

This method has two parameters, the window size r of a local region and the threshold value th in the gradient-domain processing. The former parameter and the latter one control the degree of smoothing and the amplitude of the edge should be preserved, respectively.

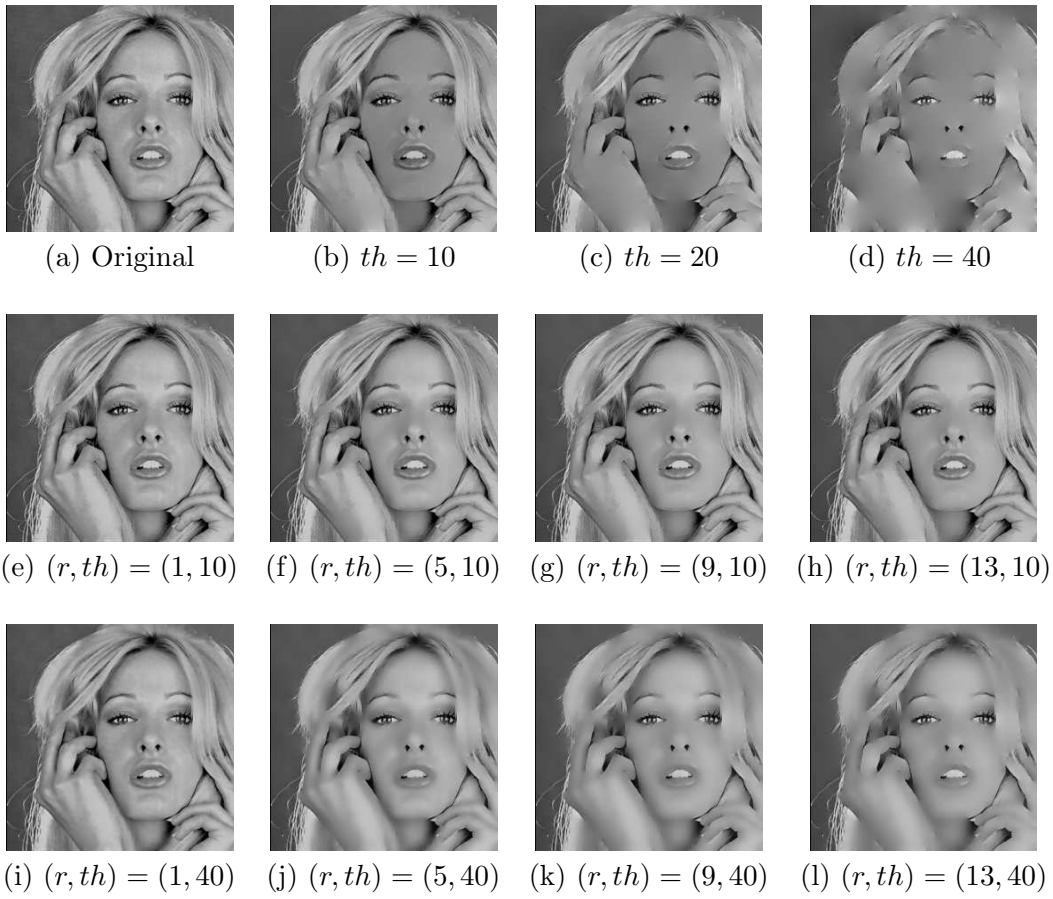
Figure 2 shows the differences of the smoothing results between the gradient-domain filtering [10] and the proposed method with various parameters. It can be seen that the proposed method shows superb edge-preserving smoothing effect in the case of the same threshold th. This is because the gradient-domain filtering processes the entire image at once without considering the local context of the input image. On the other hand, the proposed method can control the degree of smoothing by changing the parameters. As shown in Figure 2, both the window size and the threshold can control the degree of smoothing.

Furthermore, the proposed method can be applied to various kinds of image processing by changing the processing method of the gradients. For example, a typical example is as follows:

$$\vec{g}^*(i,j) = \begin{cases} \alpha\vec{g}(i,j) & \text{if } |\vec{g}(i,j)| \geq th \\ \beta\vec{g}(i,j) & \text{otherwise.} \end{cases} \tag{20}$$

Here, α and β are the parameters to control the degrees of enhancement of the strong and the weak gradients, respectively. Figure 3 shows the output images for various parameter pairs α and β. In this case, the other parameters r and th were set to be 2 and 20,

respectively. It can be confirmed that edge and gradation in flat part are enhanced by changing α and β, respectively.

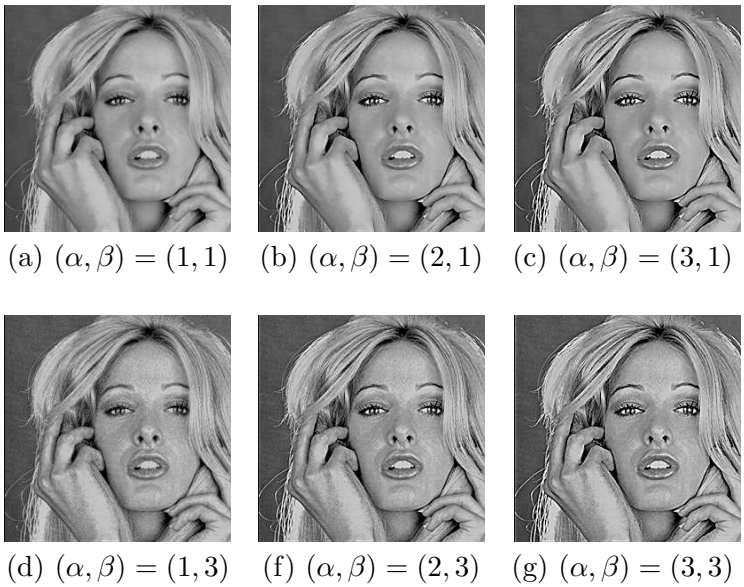


(a) Original (b) $th = 10$ (c) $th = 20$ (d) $th = 40$

(e) $(r, th) = (1, 10)$ (f) $(r, th) = (5, 10)$ (g) $(r, th) = (9, 10)$ (h) $(r, th) = (13, 10)$

(i) $(r, th) = (1, 40)$ (j) $(r, th) = (5, 40)$ (k) $(r, th) = (9, 40)$ (l) $(r, th) = (13, 40)$

FIGURE 2. Differences between the gradient-domain filtering and the proposed method: (a) original image, (b)-(d) results by the gradient-domain filtering, (e)-(l) results by the proposed method



(a) $(\alpha, \beta) = (1, 1)$ (b) $(\alpha, \beta) = (2, 1)$ (c) $(\alpha, \beta) = (3, 1)$

(d) $(\alpha, \beta) = (1, 3)$ (f) $(\alpha, \beta) = (2, 3)$ (g) $(\alpha, \beta) = (3, 3)$

FIGURE 3. Enhancement results by the proposed method with various parameters

4. **Experiments.** The edge-preserving smoothing and noise removal performance of the proposed method was verified quantitatively and qualitatively through a series of experiments. In the experiments, twelve images included in the standard image database (SIDBA) [12] were used. All the images have 256×256 pixels and are 8-bit grayscale ones.

The input images were generated by superimposing Gaussian noise (with standard deviation $\sigma = 5, 10, 15$) on the original images. The quantitative evaluation was conducted by using the mean squared error (MSE) between the original images and the corresponding resulting images. The smaller MSE value means that the resulting images are better and more similar to the original images. As methods for comparison, the conventional gradient-domain filtering [10] was employed. Furthermore, the bilateral filter was employed just for reference. The parameters of those methods were set so that those methods show the best MSE values.

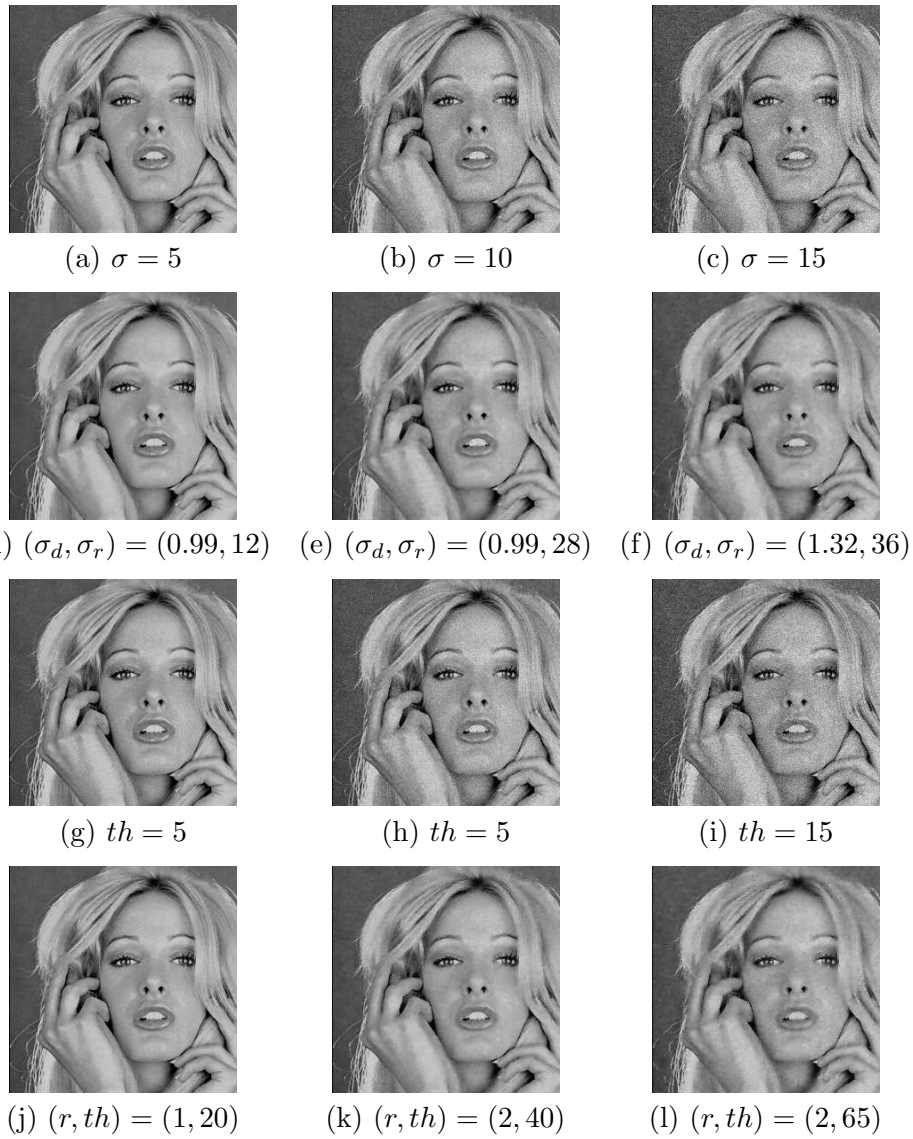
Table 1 shows the MSEs for each image by each method. In addition, Figure 4 shows some resulting images for "Woman". As shown in Table 1, the bilateral filter shows the best performance for all the images and the amplitudes of Gaussian noise. The proposed method and the gradient-domain filtering take the second and the third places, respectively. In particular, the MSE values of the gradient-domain filtering become greater than those of the input image. This can be considered that the gradient-domain filtering is applied to the entire image at the same time and its filtering effect is slightly strong.

As shown in Figure 4, all the results by the gradient-domain filtering show the obvious residual noise. On the other hand, the other methods show less residual noise than the gradient-domain filtering in the cases of small amplitudes; and the results are almost equivalent with each other. However, in the case of large amplitude noise, the proposed method shows more residual noise than the bilateral filter.

TABLE 1. MSE evaluation ($\sigma = 5$, 10, 15): input image (Input), bilateral filter (BF), gradient-domain filtering (Grad.), proposed method (Prop.)

	$\sigma = 5$				$\sigma = 10$				$\sigma = 15$			
Image Name	Input	BF	Grad.	Prop.	Input	BF	Grad.	Prop.	Input	BF	Grad.	Prop.
Airplane	24.9	13.2	25.0	17.3	100.9	37.8	99.2	52.0	224.8	68.6	224.4	100.4
Barbara	24.9	17.8	25.0	22.0	100.0	54.6	100.0	72.9	223.8	100.5	223.6	140.0
Boat	25.2	12.4	25.1	17.3	100.0	34.1	98.7	50.7	225.3	58.1	221.1	86.9
Bridge	25.0	21.9	25.1	24.5	99.4	69.2	99.4	89.0	220.3	125.1	220.3	176.4
Building	23.9	16.6	24.3	21.3	96.8	44.6	97.1	65.0	216.4	76.3	216.5	109.4
Cameraman	25.0	11.8	24.9	15.3	96.4	34.1	92.9	47.8	212.3	60.3	203.9	86.6
Girl	25.0	12.8	25.1	19.2	98.1	29.2	98.0	45.0	218.6	46.1	215.6	65.7
Lax	25.3	20.7	25.4	24.8	100.7	60.1	100.7	81.8	224.1	108.9	224.0	153.2
Lenna	25.1	12.5	25.1	16.7	100.0	33.4	99.9	46.9	225.4	57.4	224.9	84.7
Lighthouse	23.9	15.8	24.1	19.5	95.3	48.3	95.4	64.4	213.3	89.3	213.4	125.0
Text	23.8	18.0	24.4	21.9	93.9	51.6	94.0	71.0	211.7	90.7	212.0	133.1
Woman	25.0	13.5	25.2	18.4	98.6	35.1	98.6	51.1	223.7	60.7	222.5	89.1
Average	24.8	15.6	24.9	19.8	98.3	44.3	97.8	61.5	220.0	78.5	218.5	112.5

5. **Conclusions.** In gradient-domain image filtering, an output image in the original space must be reconstructed from a gradient domain image by solving a Poisson's equation under a given boundary condition. To solve the equation by a direct method requires too huge memory space and takes much computational cost. To cope with the problems on memory size and computational cost, we proposed the local-reconstruction-based gradient-domain image filtering especially for edge-preserving smoothing. Furthermore, in order to obtain the reconstructed image quickly, we solved the Poisson's equation by using the



(a) $\sigma = 5$ (b) $\sigma = 10$ (c) $\sigma = 15$

(d) $(\sigma_d, \sigma_r) = (0.99, 12)$ (e) $(\sigma_d, \sigma_r) = (0.99, 28)$ (f) $(\sigma_d, \sigma_r) = (1.32, 36)$

(g) $th = 5$ (h) $th = 5$ (i) $th = 15$

(j) $(r, th) = (1, 20)$ (k) $(r, th) = (2, 40)$ (l) $(r, th) = (2, 65)$

FIGURE 4. Noise removal results for "Woman": (a)-(c) noise-corrupted images, (d)-(f) results by the bilateral filter, (g)-(i) results by the gradient-domain filtering, (j)-(l) results by the proposed method

modified direct method. Concretely, only the weak gradient is suppressed in the gradient-domain, and then the resulting image is reconstructed and merged locally.

In the experiments, we compared the performance of the proposed method on noise removal with some methods. As methods for comparison, the conventional gradient-domain filter was employed; and the results by the bilateral filter was shown just for reference. The proposed method showed the better performance than the conventional gradient-domain filtering; and this means the problems on the memory-space and the computational cost of the gradient-domain filtering were solved.

The future work includes fine adjustment of the suppression of the weak gradient in edge-preserving smoothing and application to other kinds of image processing.

REFERENCES

[1] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd Edition, Prentice-Hall Inc., 2006.
[2] W. Burger and J. Burger, *Principles of Digital Image Processing: Core Algorithms*, Springer, 2009.
[3] C. Tomasi and R. Manduchi, Bilateral filtering for gray and color images, *Proc. of IEEE Int. Conf. Computer Vision*, pp.839-846, 1998.

[4] H. Harashima, K. Odajima, Y. Shishikui and H. Miyakawa, ε-separating nonlinear digital filter and its applications, *Electronics and Communications in Japan, Part 1*, vol.65, no.4, pp.11-19, 1982.

[5] P. Patrick, G. Michel and B. Andrew, Poisson image editing, *ACM Trans. Graphics (SIGGRAPH 2003)*, vol.22, no.3, pp.313-318, 2003.

[6] R. Fattal, D. Lischinski and M. Werman, Gradient domain high dynamic range compression, *ACM Trans. Graphics (SIGGRAPH 2002)*, vol.21, no.3, pp.249-256, 2002.

[7] M. Hua, X. Bie, M. Zhang and W. Wang, Edge-aware gradient domain optimization framework for image filtering by local propagation, *Proc. of IEEE Conf. Comput. Vis. Pattern Recognit.*, pp.2838-2845, 2014.

[8] F. Kou, W. Chen, C. Wen and Z. Li, Gradient domain guided image filtering, *IEEE Trans. Image Process.*, vol.24, no.11, pp.4528-4539, 2015.

[9] T. Shibata, M. Tanaka and M. Okutomi, Gradient-domain image reconstruction framework with intensity-range and base-structure constraints, *Proc. of IEEE Conf. Comput. Vis. Pattern Recognit.*, pp.2745-2753, 2016.

[10] S. Miyaoka, Gradient-domain image filtering, *IPSJ Journal*, vol.52, no.2, pp.901-909, 2011 (in Japanese).

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*, Cambridge University Press, 2007.

[12] M. Sakauchi, Y. Ohsawa, M. Sone and M. Onoe, Management of the standard image database for image processing researches (SIDBA), *ITEJ Technical Report*, vol.8, no.38, pp.7-12, 1984 (in Japanese).