Principal Component Analysis for Acceleration of Color Guided Image Filtering

Yoshiki Murooka, Yoshihiro Maeda, Masahiro Nakamura, Tomohiro Sasaki, and Norishige Fukushima*
Nagoya Institute of Technology, Nagoya, Japan
*fukushima@nitech.ac.jp

***Abstract*—In this paper, we propose an acceleration of guided image filtering, which is one of the fastest edge-preserving filters. The proposed method converts RGB signals into a color space introduced by the principal component analysis. Then, the filtering signals are sub-sampled or approximated by simple box filtering in the biased color space. The experimental results show that the proposed method is superior to the conventional acceleration method in accuracy and computational time.**

***Index Terms*—guided image filtering, acceleration, principal component analysis, edge-preserving filtering**

I. Introduction

Edge-preserving filtering is essential tools in image processing and computer vision fields. The smoothing is a technology that realizes smoothing signals with keeping an outline of the input signals. For this characteristic, the edge-preserving filtering is utilized for various applications, such as HDR [1], haze remove [2], detail enhancement [3], stereo matching [4], [5], depth map filtering [6], optical flow [7], and so on.

The edge-preserving filtering is classified in term of the computational complexity and filtering attribute; Finite impulse response (FIR) filters, such as bilateral filtering [8] and non-local means filtering [9], are directly convoluting images. Frequency domain filtering, such as BM3D [10], DCT denoising [11], [12], and wavelet shrinkage [13], filters signals in the frequency domain and usually utilize redundant basis sets. Infinite impulse response (IIR) filtering, such as domain transform filtering [14], recursive bilateral filtering [15], IIR-based bilateral filtering [16], utilizes recursive filtering for efficient computation. Guided image filtering [17] and its variants [18], [19], [20], [21] utilize local linearity model.

One of the main issues in the edge-preserving filtering is its high computational cost. Accordingly, numerous acceleration methods are proposed in the various edge-preserving filtering [22], [23], [24], [25]. However, there is much demand for acceleration even now.

We focus on the guided image filtering, whose computational cost is constant per an image pixel. The fast guided filtering [26] is further accelerates the filter by sub-sampling input images. This acceleration significantly reduces computational cost, on the contrary, edge signals in input images are weakened. Also, the rate of sub-sampling is limited to the radius of the filtering patch. Therefore, the acceleration has a limitation in the trade-off between precision and speed.

This work was supported by KAKENHI JP17H01764.

TABLE I: Difference between the proposed and conventional methods. After hyphens in (Prop.) is the acronym for the sampling method of each channel.

Channel Index	1	2	3
Origin (Conv.)	Full-sample	Full-sample	Full-sample
Fast (Conv.)	Sub-sample	Sub-sample	Sub-sample
PCA-FSS (Prop.)	Full-sample	Sub-sample	Subsample
PCA-FFB (Prop.)	Full-sample	Full-sample	Box Filter
PCA-FSB (Prop.)	Full-sample	Sub-sample	Box Filter
PCA-SSB (Prop.)	Sub-sample	Sub-sample	Box Filter
PCA-FBB (Prop.)	Full-sample	Box Filter	Box Filter
PCA-SBB (Prop.)	Sub-sample	Box Filter	Box Filter

In this paper, we propose a new approximated acceleration for the guided image filtering by utilizing a bias of color information. The proposed method converts the RGB color space into a color space introduced by the principal component analysis (PCA). With this conversion, the importance of image signals in each channel is concentrated; thus, we can sub-sample more spatial information in signals of nonessential channels. When we largely sub-sampled signals, moreover, edge information is almost lost in low-rank channels, however, the signals have not significant amplitude. In this case, the guided image filtering can approximate by the simple box filter for acceleration. The Table I summarizes the difference between the proposed method and the conventional method. The proposed method reduces spatial information by various methods for each channel.

II. Guided image filtering

In this section, we summarize the guided image filtering and fast one. The guided image filtering converts local patches in an input image by a linear transformation of a guide image. Let the guide signal be I. The output q is assumed as follows;

$$q_{\boldsymbol{i}} = a_{\boldsymbol{k}} I_{\boldsymbol{i}} + b_{\boldsymbol{k}}, \forall i \in \omega_{\boldsymbol{k}} \tag{1}$$

where $\boldsymbol{k}$ indicates a center position of a rectangular patch $\omega_{\boldsymbol{k}}$, and $\boldsymbol{i}$ indicates a position of a pixel in the patch. $a_{\boldsymbol{k}}$ and $b_{\boldsymbol{k}}$ are coefficients for the linear transformation. The equation represents that guide signals in a patch are linearly converted by the coefficients.

The coefficients are calculated by a linear regression of the input signal p and (1).

$$\underset{a_{\boldsymbol{k}}, b_{\boldsymbol{k}}}{\arg\min} = \sum_{\boldsymbol{i} \in \omega_{\boldsymbol{k}}} \left(\left(a_{\boldsymbol{k}} I_{\boldsymbol{i}} + b_{\boldsymbol{k}} - p_{\boldsymbol{i}} \right)^2 + \epsilon a_{\boldsymbol{k}}^2 \right) \tag{2}$$

The coefficients are estimated as follows;

$$a_{\boldsymbol{k}} = \frac{cov_{\boldsymbol{k}}(I,p)}{var_{\boldsymbol{k}}(I)+\epsilon}, \quad b_{\boldsymbol{k}} = \bar{p}_{\boldsymbol{k}} - a_{\boldsymbol{k}}\bar{I}_{\boldsymbol{k}}, \tag{3}$$

where ϵ indicates a parameter of smoothing degree. $\bar{\cdot}_{\boldsymbol{k}}$, $cov_{\boldsymbol{k}}$ and $var_{\boldsymbol{k}}$ indicate mean, variance, and covariance values of the patch $\boldsymbol{k}$. The coefficients are over overlapping in the output signals; thus, these coefficient are averaged.

$$\bar{a}_{\boldsymbol{i}} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} a_{\boldsymbol{k}}, \quad \bar{b}_{\boldsymbol{i}} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} b_{\boldsymbol{k}}, \tag{4}$$

$|\cdot|$ indicates the number of elements in the set. Finally, the output is calculated as follows;

$$q_{\boldsymbol{i}} = \bar{a}_{\boldsymbol{i}} I_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}}, \tag{5}$$

For color filtering, let input, output and guidance signals be $\boldsymbol{p} = \{p^1, p^2, p^3\}$, q^n $(n = 1, 2, 3)$, and $\boldsymbol{I}$, respectively. The per channel filtering output is defined as follows;

$$q_{\boldsymbol{i}}^n = \bar{\boldsymbol{a}_{\boldsymbol{i}}^n}^{\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b_{\boldsymbol{i}}^n}, \tag{6}$$

$$\bar{\boldsymbol{a}_{\boldsymbol{i}}^n} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} \boldsymbol{a}_{\boldsymbol{k}}^n, \quad \bar{b_{\boldsymbol{i}}^n} = \frac{1}{|\omega|}\sum_{\boldsymbol{k}\in\omega_{\boldsymbol{i}}} b_{\boldsymbol{k}}^n, \tag{7}$$

The coefficients $\boldsymbol{a}_{\boldsymbol{k}}^n$, $b_{\boldsymbol{k}}^n$ for the linear transformation is obtained as follows;

$$\boldsymbol{a}_{\boldsymbol{k}}^n = \frac{cov_{\boldsymbol{k}}(\boldsymbol{I}, p^n)}{var_{\boldsymbol{k}}(\boldsymbol{I}) + \epsilon \boldsymbol{E}}, \quad b_{\boldsymbol{k}}^n = \bar{p_{\boldsymbol{k}}^n} - {\boldsymbol{a}_{\boldsymbol{k}}^n}^{\mathrm{T}} \bar{\boldsymbol{I}}_{\boldsymbol{k}}, \tag{8}$$

where $\boldsymbol{E}$ is an identity matrix. When the output signal is color image, $cov_{\boldsymbol{k}}$ is the covariance matrix of the patch in p and $\boldsymbol{I}$. Also, $var_{\boldsymbol{k}}$ is the variance of the R, G, and B components, which will be covariance matrix, in the patch of $\boldsymbol{I}$. The division of the matrix is calculated by multiplying the inverse matrix of the denominator from the left. The calculation results of per pixel mean, variance, and covariance are obtained from the box filter. The filter can be implemented with a recursive filter [27], which can work in a constant time per pixel.

The fast guided filtering, which is an approximated acceleration, is expressed as follows;

$$q_{\boldsymbol{i}}^n = \bar{\boldsymbol{a}_{\boldsymbol{i}\uparrow\downarrow}^n}^{\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b_{\boldsymbol{i}\uparrow\downarrow}^n}, \tag{9}$$

where $\uparrow$ and $\downarrow$ indicate up-sampling and sub-sampling operators, respectively. The fast guided filtering computes $\bar{\boldsymbol{a}}_{\boldsymbol{i}}$ and $\bar{b}_{\boldsymbol{i}}$ in sub-sampled image domain for acceleration, and then simply up-sampled the coefficients.

$$\bar{\boldsymbol{a}_{\boldsymbol{i}\uparrow\downarrow}^n} := (\bar{\boldsymbol{a}_{\boldsymbol{i}\downarrow}^n})\uparrow, \quad \bar{b_{\boldsymbol{i}\uparrow\downarrow}^n} := (\bar{b_{\boldsymbol{i}\downarrow}^n})\uparrow, \tag{10}$$

$$\bar{\boldsymbol{a}_{\boldsymbol{i}\downarrow}^n} = \frac{1}{|\omega'|}\sum_{\boldsymbol{k}\in\omega'_{\boldsymbol{i}}} \boldsymbol{a}_{\boldsymbol{k}\downarrow}^n, \quad \bar{b_{\boldsymbol{i}\downarrow}^n} = \frac{1}{|\omega'|}\sum_{\boldsymbol{k}\in\omega'_{\boldsymbol{i}}} b_{\boldsymbol{k}\downarrow}^n, \tag{11}$$

$$\boldsymbol{a}_{\boldsymbol{k}\downarrow}^n = \frac{cov_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow, {p^n}_\downarrow)}{var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow) + \epsilon \boldsymbol{E}}, \quad b_{\boldsymbol{k}\downarrow}^n = \bar{p_{\boldsymbol{k}\downarrow}^n} - {\boldsymbol{a}_{\boldsymbol{k}\downarrow}^n}^{\mathrm{T}} \bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}, \tag{12}$$

Note that the filtering radius is reduced according to the rate of sub-sampling. ω' indicates a small rectangular patch, which is reshaped to fit the rate of sub-sampling.

III. Principal component analysis

The principal component analysis is a technique to synthesize variables that hold the most information from many correlation variables. In many cases, the color distribution of an RGB image is not uniformly distributed. The bias of color information is obtained by converting the pixel distribution in the RGB color space to the color space obtained by PCA.

Let input signals be $\boldsymbol{p} = \{p^1, p^2, p^3\}$. The first principal component $\hat{p}^1$ is defined as follows;

$$\hat{p}^1 = P_{11}p^1 + P_{12}p^2 + P_{13}p^3, \tag{13}$$

where $\boldsymbol{P}_1 = \{P_{11}, P_{12}, P_{13}\}^{\mathrm{T}}$ is a transformation matrix. The variance of $\hat{p}^1$ is expressed as follows;

$$var(\hat{p}^1) = \frac{1}{N}\sum_{i=0}^{N}(\hat{p}_{\boldsymbol{i}}^1 - \bar{\hat{p}}^1)^2, \tag{14}$$

where $\bar{\hat{p}}^1$ is a average of $\bar{\hat{p}}^1$. N indicates the sum of total pixels in the image. For simplicity, let (13) be $\boldsymbol{P}_1\boldsymbol{p}$. Let $\bar{\boldsymbol{A}}$ is $\bar{\boldsymbol{A}} = \{p^1 - \bar{p}^1, p^2 - \bar{p}^2, p^3 - \bar{p}^3\}$. The formula obtained by assigning $\boldsymbol{P}_1\bar{\boldsymbol{A}}$ is transformed as follows;

$$var(\hat{p}^1) = \frac{1}{N}(\boldsymbol{P}_1\bar{\boldsymbol{A}})^{\mathrm{T}}(\boldsymbol{P}_1\bar{\boldsymbol{A}}) \tag{15}$$

$$= \boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{P}_1(\frac{1}{N}\bar{\boldsymbol{A}}^{\mathrm{T}}\bar{\boldsymbol{A}}) \tag{16}$$

$$= \boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{P}_1 var(\bar{\boldsymbol{A}}). \tag{17}$$

We have to obtain the $\boldsymbol{P}_1$ that maximizes the $var(\hat{p}^1)$. When $\boldsymbol{P}_1$ is a unit vector, it can be treated as a problem of the constrained maximum. Thus, (17) is calculated by a method of Lagrange multiplier.

$$F(\boldsymbol{P}_1, \lambda) = \boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{P}_1 var(\bar{\boldsymbol{A}}) - \lambda(\boldsymbol{P}_1^{\mathrm{T}}\boldsymbol{P}_1 - 1), \tag{18}$$

where λ is a undetermined multiplier. Solving the partial differential equation, (18) is as follows;

$$\frac{\partial F}{\partial \boldsymbol{P}_1} = 2\boldsymbol{P}_1 var(\bar{\boldsymbol{A}}) - 2\lambda\boldsymbol{P}_1, \tag{19}$$

$$var(\bar{\boldsymbol{A}})\boldsymbol{P}_1 = \lambda\boldsymbol{P}_1, \tag{20}$$

where $var(\bar{\boldsymbol{A}})$ is a covariance matrix of p^1, p^2 and p^3. $\hat{p}^1$ is obtained by computing the eigenvector from the eigenvalues of the matrix. The eigenvalues of the covariance correspond to the first, second and third principal components in descending order of values. Then, the transformation vectors to the second and third principal components are also obtained from the eigenvalues. By projecting with the transformation vector, the PCA-based color space is obtained.

IV. Proposed methods

The proposed method, firstly, converts RGB input signals $\boldsymbol{p}$ into PCA-based color space $\hat{\boldsymbol{p}}$.

$$\hat{\boldsymbol{p}} = \boldsymbol{P}\boldsymbol{p}, \quad \hat{\boldsymbol{p}} = \{\hat{p}^1, \hat{p}^2, \hat{p}^3\}, \tag{21}$$

where $\boldsymbol{P}$ is a transformation matrix (3×3) derived from PCA of the RGB input image. $\hat{p}^1$, $\hat{p}^2$, and $\hat{p}^3$ represent the first,

second and third principal components, respectively. Since $\hat{p}^1$ is important information, $\hat{p}^1$ should be kept high or full sampling ratio. On the other hand, $\hat{p}^2$ and $\hat{p}^3$ can be done in a down-sampled domain. The $\hat{p}^2$ and $\hat{p}^3$ are then up-sampled by cubic interpolation. The coefficients for the linear transformation is obtained as follows;

$$\boldsymbol{a}_{\boldsymbol{k}}^n = \frac{cov_{\boldsymbol{k}}(\boldsymbol{I}, \hat{p}^n)}{var_{\boldsymbol{k}}(\boldsymbol{I}) + \epsilon \boldsymbol{E}}, \quad b_{\boldsymbol{k}}^n = \bar{\hat{p}}_{\boldsymbol{k}}^n - \boldsymbol{a}_{\boldsymbol{k}}^{n\,\mathrm{T}} \bar{\boldsymbol{I}}_{\boldsymbol{k}}, \quad (n = 1) \tag{22}$$

$$\boldsymbol{a}_{\boldsymbol{k}\downarrow}^n = \frac{cov_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow, \hat{p}_\downarrow^n)}{var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow) + \epsilon \boldsymbol{E}}, \quad b_{\boldsymbol{k}\downarrow}^n = \bar{\hat{p}}_{\boldsymbol{k}\downarrow}^n - \boldsymbol{a}_{\boldsymbol{k}\downarrow}^{n\,\mathrm{T}} \bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}, \quad (n = 2, 3) \tag{23}$$

The coefficients are processed by the average in (4) and (11). The output $\hat{q}_{\boldsymbol{i}}^n$ is calculated as follows;

$$\hat{q}_{\boldsymbol{i}}^n = \begin{cases} \bar{\boldsymbol{a}}_{\boldsymbol{i}}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}}^n \quad (n = 1), & (24) \\ \bar{\boldsymbol{a}}_{\boldsymbol{i}\uparrow\downarrow}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}\uparrow\downarrow}^n \quad (n = 2, 3). & (25) \end{cases}$$

Note that $\hat{q}^n$ is introduced from input signal in the PCA-based color space, and we do not convert guidance signal channels. If we use the PCA-based color space directly for the guided image filtering, the filtering attribute is changed in the case of shearing the input and guidance signals, which is no additional guidance information case. Thus, we utilize that the guided image filtering as a joint filter [28], [29], even if the filtering is the non-joint filtering case. We use the unconverted signal for guidance images $\boldsymbol{I}$ for joint filtering to maintain the characteristics. Finally, $\hat{q}_{\boldsymbol{i}}^n$ are converted back into RGB signals by inverse color conversion (26). We call this approximation PCA-FSS (Full-Sub-Sub sampling).

$$\boldsymbol{q} = \boldsymbol{P}^{-1} \hat{\boldsymbol{q}} \tag{26}$$

The coefficients $\boldsymbol{a}_{\boldsymbol{k}}^n$ and $b_{\boldsymbol{k}}^n$ in (8) have $var_{\boldsymbol{k}}(\boldsymbol{I})$ and $\bar{\boldsymbol{I}}_{\boldsymbol{k}}$, whose values are the same for all channels of the input signal n. Since this term depends only on the guide signal, the calculation result of $n = 1, 2, 3$ can be reused. In the proposed method, however, the term that does not hold the characteristics, because we switch the computing method depending on n as in (22) and (23). The calculation result of $n = 1$, therefore, cannot be used for $n = 2, 3$. This problem is caused by the difference in image size between $\hat{p}^1$ and $\hat{p}^2$, $\hat{p}^3$. The calculation $var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow)$ and $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$ increased by the proposed method becomes recalculation for the sub-sampled signal. The processing time of this calculation is not bottlenecked, but we examined the acceleration of this processing for further speed-up. We solve this problem by down-sampling the covariance matrix of the higher resolution, which method is introduced in image up-sampling method with local covariance values [30]. We considered the following three cases for the calculation. First, we recalculate $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$ and $var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow)$.

$$\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow} := \bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}, \quad var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow) := var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow) \tag{27}$$

This method is indicated by (recalc_var), and is no approximation. Next, $\bar{\boldsymbol{I}}_{\boldsymbol{k}}$ and $var_{\boldsymbol{k}}(\boldsymbol{I})$ are obtained from sub-sampling full resolution values for acceleration.

$$\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow} \approx (\bar{\boldsymbol{I}}_{\boldsymbol{k}})\downarrow, \quad var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow) \approx (var_{\boldsymbol{k}}(\boldsymbol{I}))\downarrow \tag{28}$$

This approximation is indicated by (resize_$\bar{I}$, resize_var). In the final case, we recalculate only $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$, and sub-sample $var_{\boldsymbol{k}}(\boldsymbol{I})$ is used.

$$\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow} := \bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}, \quad var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow) \approx (var_{\boldsymbol{k}}(\boldsymbol{I}))\downarrow \tag{29}$$

This approximation is indicated by (recalc_$\bar{I}$, resize_var).

Furthermore, we focus on the fact that the intensity of the signal $\hat{p}^3$ is small. Processing three channels with different sub-sampling sizes lead to increase processing time for var matrix. We notice that when sub-sampling the signal largely, the output cannot maintain the edge. Thus, there is no significant difference between the output of the box filter and guided image filter. Beside, importance of $\hat{p}^3$ is low for reconstructed images; therefore, $\hat{q}_{\boldsymbol{i}}^3$ in (25) can be approximated as follows;

$$\hat{q}_{\boldsymbol{i}}^n = \begin{cases} \bar{\boldsymbol{a}}_{\boldsymbol{i}}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}}^n, \quad (n = 1) & (30) \\ \bar{\boldsymbol{a}}_{\boldsymbol{i}\uparrow\downarrow}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 2) & (31) \\ \bar{\hat{p}}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 3) & (32) \end{cases}$$

This method utilizes full sampling for $\hat{p}^1$ and sub-sampling for $\hat{p}^2$, and sub-sampling and filtering with a box filter for $\hat{p}^3$. Then, the $\hat{p}^2, \hat{p}^3$ are up-sampled. We call this approximation, PCA-FSB (Full-Sub-Box). The approach can also extend to the fast guided filter by sub-sampling $\hat{p}^1$ as shown in (25).

$$\hat{q}_{\boldsymbol{i}}^n = \begin{cases} \bar{\boldsymbol{a}}_{\boldsymbol{i}\uparrow\downarrow}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 1, 2) & (33) \\ \bar{\hat{p}}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 3) & (34) \end{cases}$$

This approximation is indicated by PCA-SSB (Sub-Sub-Box). $\hat{q}_{\boldsymbol{i}}^2$ can also be approximated as follows;

$$\hat{q}_{\boldsymbol{i}}^n = \begin{cases} \bar{\boldsymbol{a}}_{\boldsymbol{i}\uparrow\downarrow}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 1) & (35) \\ \bar{\hat{p}}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 2, 3) & (36) \end{cases}$$

This approximation is indicated by PCA-SBB (Sub-Box-Box). When the fast guided filtering is described like the proposed method, it becomes RGB-SSS (Sub-Sub-Sub).

When approximating with box filter without sub-sampling, it is as follows.

$$\hat{q}_{\boldsymbol{i}}^n = \begin{cases} \bar{\boldsymbol{a}}_{\boldsymbol{i}}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}}^n, \quad (n = 1, 2) & (37) \\ \bar{\hat{p}}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 3) & (38) \end{cases}$$

$$\hat{q}_{\boldsymbol{i}}^n = \begin{cases} \bar{\boldsymbol{a}}_{\boldsymbol{i}}^{n\,\mathrm{T}} \boldsymbol{I}_{\boldsymbol{i}} + \bar{b}_{\boldsymbol{i}}^n, \quad (n = 1) & (39) \\ \bar{\hat{p}}_{\boldsymbol{i}\uparrow\downarrow}^n, \quad (n = 2, 3) & (40) \end{cases}$$

This approximation is indicated by PCA-FFB (Full-Full-Box) and PCA-FBB (Full-Box-Box), respectively.

V. Experimental results

In the experiment, firstly, we investigated the calculation of $var_{\boldsymbol{k}}(\boldsymbol{I}_\downarrow)$ and $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$ in PCA-FSS and PCA-FSB. We compared the methods (resize_$\bar{I}$, resize_var), (recalc_$\bar{I}$, resize_var) and (recalc_var). The approximation accuracy was evaluated with the peak signal noise ratio (PSNR) of the average of each RGB channel between the approximation and naïve implementation. We employed 24 images (768×512), which are Kodak test images, for evaluation. We use Intel Core i5-7500 3.4 GHz

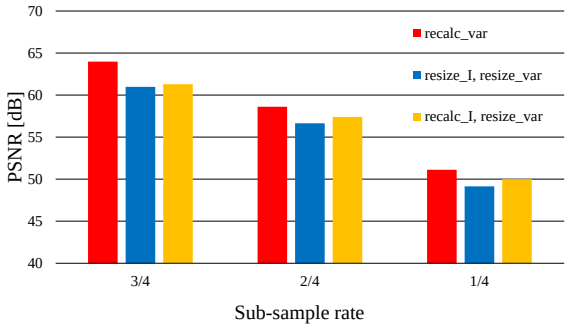

Fig. 1: PSNR for the calculation of $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$ and $var_{\boldsymbol{k}}(\boldsymbol{I}_{\downarrow})$ (r = 4, $\epsilon = 0.2^2$).

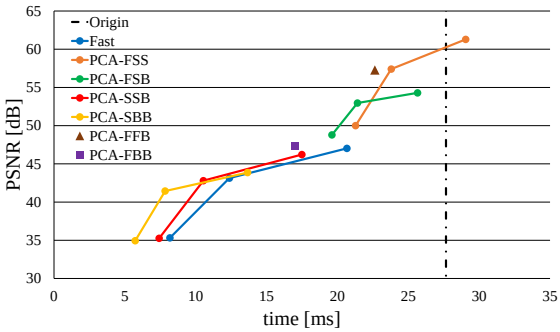

Fig. 2: Processing time and PSNR (r = 4, $\epsilon = 0.2^2$).

TABLE II: Processing time for the calculation of $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$ and $var_{\boldsymbol{k}}(\boldsymbol{I}_{\downarrow})$. [ms]

sub-sample rate	recalc_var		resize_$\bar{I}$, resize_var		recalc_$\bar{I}$, resize_var	
	small	large	small	large	small	large
3/4	2.81	30.39	2.45	20.05	2.03	17.03
2/4	0.93	14.41	0.79	8.30	0.76	8.16
1/4	0.36	4.10	0.68	5.20	0.33	3.82

(4 threads). The code is written in C++ (Visual Studio 2015 C++). The filters are vectorized by AVX and parallelized by four threads. Experimental results are shown in Fig. 1 and Table II. Figure 1 shows the approximation accuracy. (resize_$\bar{I}$, resize_var) and (recalc_$\bar{I}$, resize_var) have lower accuracy than (recalc_var), but deterioration is not high. Table II shows the processing time. The calculation time was the average of 10000 trials. We employed the large image (1920×1200) and the small image (512×512). Table II shows that the method of sub-sampling $var_{\boldsymbol{k}}(\boldsymbol{I})$ is effective regardless of the image size. Also, the table shows that the recalculation is effective in $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$. Computing resize_$\bar{I}$ requires large input images; thus computing throughput is down. We employed, therefore, _$\bar{I}$, resize_var which is a method of recalculating $\bar{\boldsymbol{I}}_{\boldsymbol{k}\downarrow}$ and sub-sampling $var_{\boldsymbol{k}}(\boldsymbol{I})$ for acceleration.

Also, we compared the proposed method with the fast guided filtering. The calculation time was the average of 10000 trials. The approximation accuracy was also evaluated with PSNR of the average of each RGB channel between the approximation and naïve implementation. We employed 24 images (768×512), which are Kodak test images, for evaluation. Experimental results are shown in Fig. 2. Figure 2 shows the trade-off between computational time and PSNR. The plots with connected lines are the same filtering method varying down-sampling ratio. The changing down-sampling rates are three-quarter, two-quarter, and one-quarter with the fixed radius of the filtering, which was 4. The proposed method indicates the better trade-off than the conventional method. The PCA-FSS, PCA-FSB, and PCA-FFB have higher accuracy than the conventional approximation. In particular, the PCA-SSB and PCA-SBB achieve higher speeds than the conventional acceleration approximation with keeping the same accuracy level. Here, Table III shows the processing time of the PCA part. We can find that the related processing time of PCA is minimal.

TABLE III: Computational cost [ms] of PCA parts. $\boldsymbol{Pp}$ contains the calculation of the covariance matrix for computing $\boldsymbol{P}$ and $\boldsymbol{P}^{-1}$.

Process	Processing time
$\boldsymbol{Pp}$	0.57
$\boldsymbol{P}^{-1}\hat{\boldsymbol{q}}$	0.17

Figure 3 shows the output images, which sub-sample rate is $2/4$. The proposed method (c) - (f) is subjectively well. In Fig. 2, the fast guided filtering, PCA-SSB, and PCA-SBB have the lowest accuracy. Figure 4 shows the too sub-sampling point in the fast guided filtering. We can find that edge signals in an image are weakened.

VI. Conclusion

In this paper, the guided image filtering could be accurately accelerated by the principal component analysis with down-sampling or approximating by box filtering. Experimental results showed that the proposed method was superior to the conventional acceleration of the fast guided image filtering [26].

References

[1] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," *ACM Trans. on Graphics*, vol. 21, no. 3, pp. 257–266, 2002.

[2] K. He, J. Sun, and X. Tang, "Single image haze removal using dark channel prior," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009, pp. 2341–2353.

[3] Z. Farbman, R. Fattal, D. Lischinski, and R. Szeliski, "Edge-preserving decompositions for multi-scale tone and detail manipulation," *ACM Trans. on Graphics*, vol. 27, no. 3, 2008.

[4] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011, pp. 3017–3024.

[5] T. Matsuo, S. Fujita, N. Fukushima, and Y. Ishibashi, "Efficient edge-awareness propagation via single-map filtering for edge-preserving stereo matching," in *Proc. IS&T/SPIE Electronic Imaging, Three-Dimensional Image Processing, Measurement, and Applications*, 2015, p. 93930S.

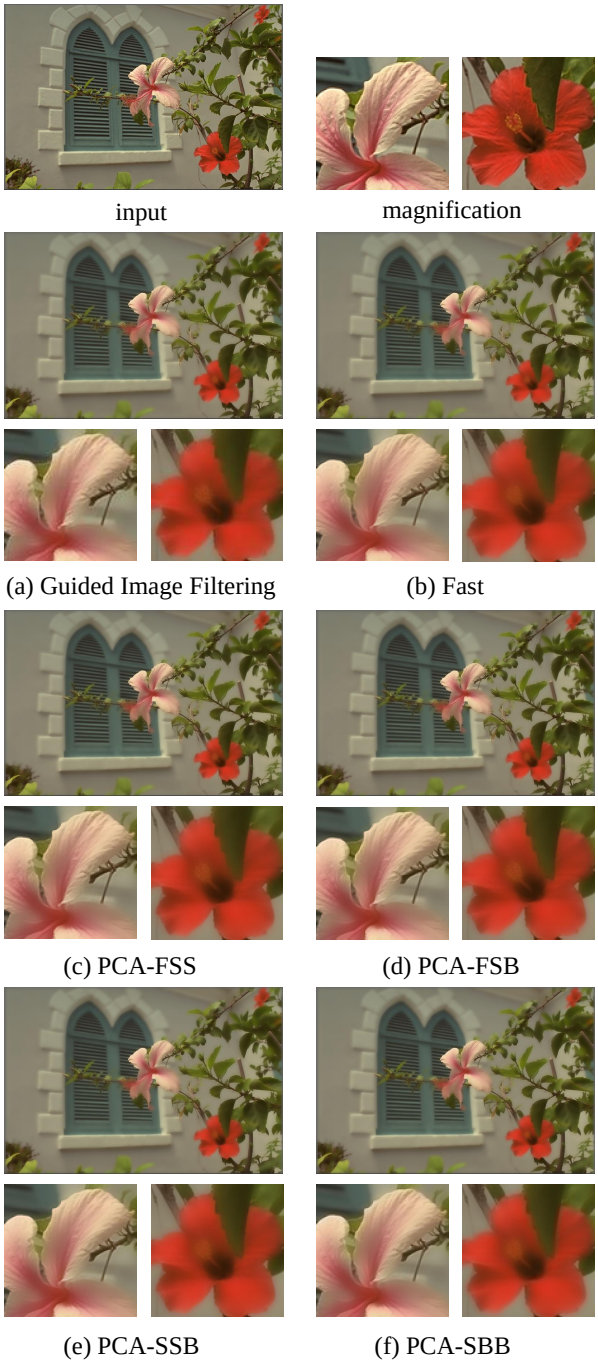

Fig. 3: Filtering results of various methods.

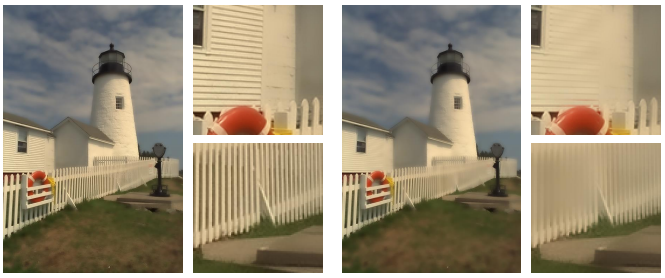

Fig. 4: Example of sub-sampling the input signal too much (sub-sample rate $1/4$).

[6] T. Matsuo, N. Fukushima, and Y. Ishibashi, "Weighted joint bilateral filter with slope depth compensation filter for depth map refinement," in *Proc. The 8th International Conference on Computer Vision Theory and Applications (VISAPP)*, 2013, pp. 300–309.

[7] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski, "A database and evaluation methodology for optical flow," *International Journal of Computer Vision*, vol. 92, no. 1, pp. 1–31, 2011.

[8] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *IEEE International Conference on Computer Vision*, 1998, pp. 839–846.

[9] A. Buades, B. Coll, and J. M. Morel, "A non-local algorithm for image denoising," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 60–65.

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

[11] S. Fujita, N. Fukushima, M. Kimura, and Y. Ishibashi, "Randomized redundant dct: Efficient denoising by using random subsampling of dct patches," in *SIGGRAPH Asia 2015 Technical Briefs*. ACM, 2015.

[12] Y. Kawasaki, Y. Maeda, and N. Fukushima, "Parallelized and vectorized implementation of dct denoising with fma instructions," in *Proc. International Workshop on Advanced Image Technology (IWAIT)*, 2018.

[13] D. L. Donoho, I. M. Johnstone, G. Kerkyacharian, and D. Picard, "Wavelet shrinkage: Asymptopia," in *With Discussion*, 1995.

[14] E. S. L. Gastal and M. M. Oliveira, "Domain transform for edge-aware image and video processing," *ACM Trans. on Graphics*, vol. 30, no. 4, pp. 69:1–69:12, 2011.

[15] Q. Yang, "Recursive bilateral filtering," in *Proc. European Conference on Computer Vision (ECCV)*, 2012, pp. 399–413.

[16] N. Fukushima, K. Sugimoto, and S. Kamata, "Complex coefficient representation for iir bilateral filter," in *in Proc. International Conference on Image Processing (ICIP)*, 2017.

[17] K. He, J. Shun, and X. Tang, "Guided image filtering," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.

[18] J. Zheng, Z. Li, Z. Zhu, W. Yao, and S. Wu, "Weighted guided image filtering," *IEEE Trans. on Image Processing*, vol. 24, no. 1, pp. 120–129, 2015.

[19] C. Wen, F. Kou, W. Chen, and Z. Li, "Gradient domain guided image filtering," *IEEE Trans. on Image Processing*, vol. 24, no. 11, pp. 4528–4539, 2015.

[20] S. Fujita and N. Fukushima, "High-dimensional guided image filtering," in *Proc. International Conference on Computer Vision Theory and Applications (VISAPP)*, 2016, pp. 27–34.

[21] ——, *Extending Guided Image Filtering for High-Dimensional Signals*. Springer International Publishing, 2017, vol. 693, pp. 439–453.

[22] F. Durand and J. Dorsey, "Fast bilateral filtering for the display of high-dynamic-range images," *ACM Trans. on Graphics*, vol. 21, no. 3, pp. 257–266, 2002.

[23] K. Chaudhury, "Constant-time filtering using shiftable kernels," *IEEE Signal Processing Letters*, vol. 18, no. 11, pp. 651–654, 2011.

[24] N. Fukushima, S. Fujita, and Y. Ishibashi, "Switching dual kernels for separable edge-preserving filtering," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2015.

[25] K. Sugimoto and S.-I. Kamata, "Compressive bilateral filtering," *IEEE Trans. on Image Processing*, vol. 24, no. 11, pp. 3357–3369, 2015.

[26] K. He and J. Sun, "Fast guided filter," *CoRR*, vol. abs/1505.00996, 2015.

[27] F. C. Crow, "Summed-area tables for texture mapping," in *Proc. ACM SIGGRAPH*, 1984, pp. 207–212.

[28] G. Petschnigg, M. Agrawala, H. Hoppe, R. Szeliski, M. Cohen, and K. Toyama, "Digital photography with flash and no-flash image pairs," *ACM Trans. on Graphics*, vol. 23, no. 3, pp. 664–672, 2004.

[29] E. Eisemann and F. Durand, "Flash photography enhancement via intrinsic relighting," *ACM Trans. on Graphics*, vol. 23, no. 3, pp. 673–678, 2004.

[30] X. Li and M. T. Orchard, "New edge-directed interpolation," *IEEE Trans. on Image Processing*, vol. 10, no. 10, pp. 1521–1527, 2001.