

Paper

Efficient Constant-time Gaussian Filtering with Sliding DCT/DST-5 and Dual-domain Error Minimization

Kenjiro Sugimoto †, Sei-ichiro Kamata (member)†

Abstract This paper presents an efficient constant-time algorithm for Gaussian filtering and also Gaussian derivative filtering that provides a high approximate accuracy in a low computational complexity regardless of its filter window size. The proposed algorithm consists of two key techniques: second-order shift properties of the Discrete Cosine/Sine Transforms type-5 and dual-domain error minimization for finding optimal parameters. The former enables us to perform filtering in fewer number of arithmetic operations as compared than some state-of-the-art algorithms without integral images. The latter enables us to find the optimal filter size that provides the most accurate filter kernel approximation. Experiments show that the proposed algorithm clearly outperforms state-of-the-art ones in computational complexity, approximate accuracy, and accuracy stability.

Key words: constant-time Gaussian filtering, constant-time derivative Gaussian filtering, sliding DCT/DST, second-order shift property, frequency sampling method.

1. Introduction

The Gaussian filter (GF) is one of the essential tools in image processing and computer vision. Although one may think that this long-historical linear filter sounds old-fashion at the present day, it still plays fundamental roles in many modern applications. For example, its importance can be exemplified in the scale-space analysis[1], which is a successful method for dealing with some visual information of varied sizes contained in an image, and the bilateral filter[2) 4)], which is an efficient edge-preserving smoothing filter. These two techniques have achieved significant results in various tasks including object recognition[5], super resolution[6], high-dynamic range imaging[7], stereo matching[8], segmentation[9], visual saliency[10], edge detection[11)12)]. The former iteratively applies the GF to a target image many times; the latter is generally formed by iterations of GFs for fast filtering[13) 15)]. Thus, the GF is actively used in a wide variety of modern image processing applications.

A principal problem for the GF is the computational complexity proportional to its filter window size, which is generally determined from the scale parameter σ of the Gaussian kernel. This dependency on σ is a performance bottleneck for filtering high-resolutional images in particular because higher resolution requires large σ. A widely known classical solution to overcome this problem is to use the Fast Fourier Transform (FFT), based on convolution-multiplication property[16]. However, the FFT and its inverse require additional computational complexity and limit the size of target images, e.g., power-of-two. Hence, we discuss GF algorithms with σ-independent computational complexity by tolerating a slight sacrifice of approximate accuracy, known as a constant-time GF ($O(1)$ GF).

Many algorithms for the $O(1)$ GF have been proposed in the past and share the design concept that a Gaussian kernel is decomposed into a sum or product of several kernels able to be convolved in $O(1)$/pixel. We review three major categories in the $O(1)$ GF as follows:

Iterated box filtering[17]: A Gaussian kernel can be decomposed into a product of several box kernels, c.f., the central limit theorem. In short, applying box filters iteratively results in asymptoting a Gaussian-filtered image. Any box filter can be operated in $O(1)$ complexity by using a moving-sum approach. However, it has a difficulty to control approaching to a desired σ and adjusting the approximate accuracy of output images.

Recursive Gaussian filtering[18) 21)]: A Gaussian kernel can be represented as a sum or a product of two one-sided Gaussian kernels, which are filtered as two low-order recursive filters regardless of σ. This approach has produced the state-of-the-art performance in both computational complexity and approximate accuracy. However, as References[22)23)] observed, they tend to fail the kernel approximation for large σ.

Received June 14, 2014; Revised August 26, 2014; Accepted October 3, 2014

† Graduate School of Information, Production and Systems, Waseda University
(Fukuoka, Japan)

Filtering with extended integral images[24) 27)]: A Gaussian kernel can be approximated by a linear sum of splines[24)25)], polynomials[26)] or cosines[27)]. We can convolve each term of the linear sum in $O(1)$ by extending the integral image techniques[28)29)]. Above all, the Cosine Integral Image (CII)[27)] achieves the most efficient performance tradeoff in them. However, this category has an essential problem of high computational complexity caused by generating and restoring integral images.

A superiority of the CII underlies a spectrum sparsity of the Gaussian kernel. As classical methods with a similar concept, we must mention the frequency sampling method[30)] and the sliding Discrete Fourier Transform (sliding DFT)[31)], which had been actively studied in 1960-1980s. The former is a design method for approximating a finite-impulse response filter by sampling its significant frequency components, and the latter is a recursive method for computing short-time transform coefficients of a target sequence. For example, this combined algorithm was applied to the one-dimensional Gabor filter[32)]. Clearly, the CII and the sliding DFT with the frequency sampling method share many points in their concepts. In order to design a more efficient filtering algorithm, we should understand and discuss these methods comprehensively.

These existing algorithms have the following characteristics and drawbacks. First, the CII requires integral images, increasing computational complexity; by contrast, the sliding DFT runs without integral images. Second, the CII employs the DCT-1 for kernel decomposition; the sliding DFT employs the DFT. Since most cases in real applications of image processing deal with real data and real kernels, not complex, the DCT shows an advantage in computational complexity. However, the DCT-1 causes some offset error in kernel approximation as we demonstrate in Section 2. Third, in both algorithms, manually setting the filter window size without any consolidate theoretical criterion detracts their performance and usability. For example, it is common in the GF that the filter window size is manually determined, e.g., $2\lceil 3\sigma \rceil + 1$, which supports 99.7% area of the Gaussian kernel. However, we found that, in the CII etc., this manual approach actually has much room for improvements in approximate accuracy and computational complexity. It is clearly essential to find the optimal parameters to achieve the highest approximate accuracy. Because these discussions are extensively applicable to the Discrete Sine Transform (DST) and odd-symmetric filters such as the first-derivative GF, we collectively focus on the DCT and the DST hereafter.

This paper presents an efficient $O(1)$ filtering algorithm that provides a stably-high approximate accuracy in low computational complexity. Our algorithm targets one-dimensional, compact-support, narrowband kernels including Gaussian kernel and derivative Gaussian kernels. Note that, because of their separability, these one-dimensional kernels can compose multi-dimensional Gaussian, derivative Gaussian, and Laplacian-of-Gaussian (LoG) kernels, which are widely used in many applications. Our algorithm comprises two proposed techniques: second-order shift properties of the sliding DCT/DST-5 and dual-domain error minimization. The former requires the fewest number of arithmetic operations without integral images, extra components, and offset error as compared with the existing algorithms. The latter finds the optimal parameters for kernel approximation via error minimization both in spatial and frequency domains. These novel techniques enhance the performance of the $O(1)$ GF etc. in many aspects such as computational complexity, approximate accuracy, accuracy stability, and usability.

Major contributions of this paper are as follows:
1) We clarify the superiority of the DCT/DST-5 to the DCT/DST-1 in kernel approximation and derive their second-order shift properties for sliding transforms.
2) We formalize dual-domain error minimization, which is a theoretical solution to find the optimal parameters for our kernel approximation. This approach enhances accuracy and improves usability such as only one manual parameter and no complicated preprocessing.
3) We design an efficient constant-time algorithm for the GF and its derivative filters based on the above two techniques. Our algorithm outperforms the state-of-the-art ones such as the recursive GFs in computational complexity, approximate accuracy, and its stability.
The preliminary work for this paper are given in some references[23)33)34)].

2. Filtering via Sliding DCT/DST-5

2. 1 Kernel approximation via DCT/DST-5

Consider decomposing a real kernel via a DCT. Let $h_t \in \mathbb{R}$ be a finite-length filter kernel with domain $t \in \{-R, \ldots, R\} \subset \mathbb{N}$ where $R \in \mathbb{N}$ is the kernel radius. If h_t is even symmetric such as the Gaussian kernel, then it can be decomposed into a sum of cosine terms. Due to various different assumptions of the even symmetry, there exist eight patterns sharing the general form

$$h_t = \sum_{k=0}^{R} H^{(k)} \cos\left(\omega(k+\kappa)(t+\tau)\right), \qquad (1)$$

where $\omega \in \{\frac{\pi}{R}, \frac{2\pi}{2R+1}\}$, $\kappa \in \{0, \frac{1}{2}\}$, $\tau \in \{0, \frac{1}{2}\}$, and $H^{(k)}$ is the k-th weight coefficient. This decomposition is generally called an inverse DCT[35] but simply called a DCT in this paper. The variables ω, κ, and τ are determined from an employed assumption for the even symmetry. For example, $\omega = \frac{\pi}{R}$ corresponds to the DCT-1,2,3,4, and $\omega = \frac{2\pi}{2R+1}$ corresponds to the DCT-5,6,7,8. Here, we focus only on $\kappa = 0$ and $\tau = 0$ because they produce easily-computable phase starting from zero. In the remaining DCTs, the DCT-1 ($\omega = \frac{\pi}{R}$) assumes the symmetry of $\{f_{R-1}, \ldots, f_1, f_0, f_1, \ldots, f_R\}$ with the length of $2R$, and the DCT-5 ($\omega = \frac{2\pi}{2R+1}$) assumes the symmetry of $\{f_R, \ldots, f_1, f_0, f_1, \ldots, f_R\}$ with a length of $2R+1$. In signal processing, the DCT-1,2,3,4 (with an even-length of period) have been traditionally much focused on than the DCT/DST-5,6,7,8 (with an odd-length of period)[35], probably due to their affinity to butterfly computation.

In kernel approximation via the DCT-1 or the DCT-5, we consider eliminating a cosine term in (1) and reveal its negative effect. The total value of h_t is rewritten as

$$\sum_{t=-R}^{R} h_t = \sum_{k=0}^{R} H^{(k)} \sum_{t=-R}^{R} \cos(\omega k t)$$
$$= (2R+1)H^{(0)} + \begin{cases} 0 & \text{if } \omega = \frac{2\pi}{2R+1} \\ \sum_{k=1}^{R} (-1)^k H^{(k)} & \text{otherwise} \end{cases}.$$

This equation implies that, even if we eliminate a cosine term as $H^{(k)} \leftarrow 0$ except for $k = 0$, the total value unchanges in the DCT-5 but changes in the DCT-1. Evidently, the total area depends only on $H^{(0)}$ in the DCT-5 because of its orthogonality, which arises from a match between the period length of DCT-5 and the filter window length. On the other hand, the DCT-1 produces offset distortion of $(-1)^k H^{(k)}$ after eliminating the k-th cosine component. Even though it could be compensated by adjusting $H^{(0)}$, such a heuristic solution would break the direct relation to the DFT and would unguarantee the optimality in terms of least square error. Since the GF and many other linear filters in image processing generally have an odd-length filter window, the DCT-5 is consistent with these cases. As in the DCT, we can demonstrate the same fact on the DSTs.

Consider a one-dimensional convolution with the finite-length kernel and an input sequence $f_x \in \mathbb{R}$ ($x = \{0, 1, ..., N-1\}$) where N is the length of the input sequence. We first introduce $C_t^{(k)} = \cos(\omega k t)$ and $S_t^{(k)} = \sin(\omega k t)$ for shorthand notation. The finite-length kernel can be represented as

$$h_t = \frac{1}{2}a^{(0)} + \sum_{k=1}^{R} \left[a^{(k)} C_t^{(k)} + b^{(k)} S_t^{(k)}\right], \qquad (2)$$

where

$$a^{(k)} = \frac{\omega}{\pi} \sum_{t=-R}^{R} h_t C_t^{(k)}, \quad b^{(k)} = \frac{\omega}{\pi} \sum_{t=-R}^{R} h_t S_t^{(k)}. \quad (3)$$

Convolution between f_x and h_t can be described as *

$$\tilde{f}_x = \sum_{t=-R}^{R} h_t f_{x+t} = \sum_{k=0}^{R} \left[a^{(k)} A_x^{(k)} + b^{(k)} B_x^{(k)}\right], \quad (4)$$

where $A_k^{(x)}$ and $B_k^{(x)}$ are the k-th short-time DCT/DST coefficients of f_x at position x, respectively, defined by

$$A_x^{(k)} = \sum_{t=-R}^{R} f_{x+t} C_t^{(k)}, \quad B_x^{(k)} = \sum_{t=-R}^{R} f_{x+t} S_t^{(k)}. \quad (5)$$

Some useful kernels such as the Gaussian kernel or its derivative kernels possesses narrowband spectra, i.e., many $a^{(k)}$ and $b^{(k)}$ are zeros or near zeros (see Section 3 in detail). Hence, if the $A_x^{(k)}$ and $B_x^{(k)}$ are computable in $O(1)$ time complexity, this approach can be an efficient constant-time filtering algorithm.

2.2 2nd-order shift property of DCT/DST-5

The sliding DCT/DST is an efficient method for recursively computing $A_x^{(k)}$ and $B_x^{(k)}$ in $O(1)$ time complexity. Many sliding transforms have been proposed in the literature[31)36) 40)]. They utilize recurrence relations between two or three adjacent short-time transform coefficients of a target sequence, called first-order and second-order shift properties, respectively. In general, the second-order shift properties outperform the first-order ones in terms of computational complexity because the first-order ones require to compute both cosine and sine parts regardless of a symmetric or an asymmetric kernel. However no second-order shift properties of the DCT/DST-5 have not been explicitly derived yet. Only Wu[37] focused on the DCT/DST-5 but their first-order ones. Hence, we derive second-order shift properties of the DCT/DST-5 and design an efficient filtering algorithm with them.

Again, the second-order shift properties of the DCT/DST-5 indicate the relationships between three adjacent short-time coefficients $A_{x-1}^{(k)}$, $A_x^{(k)}$ and $A_{x+1}^{(k)}$.

Proposition 1. *A relationship between three adjacent*

* This definition is based on correlation filtering.

short-time transform coefficients are called a second-order shift property. The DCT-5 and the DST-5 have the following second-order shift properties:

$$A^{(k)}_{x-1} + A^{(k)}_{x+1} = 2C^{(k)}_1 A^{(k)}_x + C^{(k)}_R \Delta^{\mathrm{c}}_x, \quad (6)$$

$$B^{(k)}_{x-1} + B^{(k)}_{x+1} = 2C^{(k)}_1 B^{(k)}_x + S^{(k)}_R \Delta^{\mathrm{s}}_x, \quad (7)$$

where $\Delta^{\mathrm{c}}_x = f_{x-R-1} - f_{x-R} - f_{x+R} + f_{x+R+1}$ *and* $\Delta^{\mathrm{s}}_x = -f_{x-R-1} - f_{x-R} + f_{x+R} + f_{x+R+1}$.

Proof. Expand $A^{(k)}_{x-1}$ and $A^{(k)}_{x+1}$ as

$$A^{(k)}_{x-1} = \left[\sum_{t=-R}^{R} f_{x+t} C^{(k)}_{t+1}\right] - f_{x+R} C^{(k)}_{R+1} + f_{x-R-1} C^{(k)}_{-R},$$

$$A^{(k)}_{x+1} = \left[\sum_{t=-R}^{R} f_{x+t} C^{(k)}_{t-1}\right] + f_{x+R+1} C^{(k)}_{R} - f_{x-R} C^{(k)}_{-R-1}.$$

Since $C^{(k)}_R = C^{(k)}_{-R} = C^{(k)}_{R+1} = C^{(k)}_{-R-1}$, we obtain (6). Likewise, $B^{(k)}_{x-1}$ and $B^{(k)}_{x+1}$ are expanded as

$$B^{(k)}_{x-1} = \left[\sum_{t=-R}^{R} f_{x+t} S^{(k)}_{t+1}\right] - f_{x+R} S^{(k)}_{R+1} + f_{x-R-1} S^{(k)}_{-R},$$

$$B^{(k)}_{x+1} = \left[\sum_{t=-R}^{R} f_{x+t} S^{(k)}_{t-1}\right] + f_{x+R+1} S^{(k)}_{R} - f_{x-R} S^{(k)}_{-R-1}.$$

Since $S^{(k)}_R = -S^{(k)}_{-R} = -S^{(k)}_{R+1} = S^{(k)}_{-R-1}$, (7) holds. □

Next, we design a filtering algorithm based on the sliding DCT-5 with (6) and the sliding DST-5 with (7). The first and second coefficients $A^{(k)}_0$ and $A^{(k)}_1$ are explicitly computed from the original definition (5). In addition, for $k = 0$, the following sliding transform has lower computational complexity:

$$A^{(0)}_{x+1} = A^{(0)}_x + f_{x+R+1} - f_{x-R}. \quad (8)$$

In the second-order shift properties of them, they are closed under cosine components for (6) or sine components for (7), not mixed. These forms are important for kernel approximation because most of kernels are symmetric, which are represented by cosine or sine components only. Thus, this sliding technique requires low-computational complexity. Our algorithm precomputes all the cosines/sins as look-up tables prior to filtering.

We can reduce more multiplications by utilizing the look-up tables. In many cases of image filtering, a filter kernel is statically convolved to a whole image. Since $a^{(k)}$ and $b^{(k)}$ can be interpreted as constants while filtering in this scenario, we can reformulate (4) as

$$\tilde{f}_x = \sum_{k=0}^{R} \left[\alpha^{(k)}_x + \beta^{(k)}_x\right]. \quad (9)$$

where $\alpha^{(k)}_x = a^{(k)} A^{(k)}_x$ and $\beta^{(k)}_x = b^{(k)} B^{(k)}_x$. The sliding transforms (6) and (7) are also replaced to

$$\alpha^{(k)}_{x-1} + \alpha^{(k)}_{x+1} = \left\{2C^{(k)}_1\right\} \alpha^{(k)}_x + \left\{a^{(k)} C^{(k)}_R\right\} \Delta^{\mathrm{c}}_x, \quad (10)$$

$$\beta^{(k)}_{x-1} + \beta^{(k)}_{x+1} = \left\{2C^{(k)}_1\right\} \beta^{(k)}_x + \left\{b^{(k)} S^{(k)}_R\right\} \Delta^{\mathrm{s}}_x, \quad (11)$$

where $\{\cdot\}$ indicates the stored values in the look-up tables. Through this formulation, the $4K$ multiplications contained in (4) are moved to (10) and (11) as the look-up tables. Thus, our algorithm processes a cosine/sin component by two multiplications per pixel.

Algorithm 1 Proposed $O(1)$ Gaussian Filter

```
1: ▷ f: target sequence, σ: scale, K, R: parameters
2: function ProposedGaussianFilter(f, σ, K, R)
3:     ▷ Calculating the first and second terms
4:     a^(0) ← 1/(2R+1)
5:     A^(0) ← Σ_{t=-R}^{R} f_t
6:     for k ← 1 to K do
7:         a^(k) ← (2/(2R+1)) exp(-(1/2)ω²σ²k²)                ▷ See (20).
8:         α^(k)_{-1} ← Σ_{t=-R}^{R} {a^(k) cos(ωkt)} f_t
9:         α^(k)_0 ← Σ_{t=-R}^{R} {a^(k) cos(ωkt)} f_{t+1}
10:    end for
11:
12:    ▷ Filtering via Our Sliding DCT-5
13:    f̃_0 ← a^(0) A^(0) + Σ_{k=1}^{K} α^(k)_{-1}
14:    A^(0) ← A^(0) + (f_{R+1} - f_{-R})
15:    for x ← 1 to N - 1 do
16:        ▷ Calculating the filter output for position x
17:        f̃_x ← a^(0) A^(0) + Σ_{k=1}^{K} α^(k)_0
18:        ▷ Updating short-time DCT coefficients for the next
19:        A^(0) ← A^(0) + (f_{x+R+1} - f_{x-R})
20:        Δ^c ← f_{x-R-1} - f_{x-R} - f_{x+R} + f_{x+R+1}
21:        for k ← 1 to K do
22:            α^(k)_{+1} ← {2 cos(ωk)} α^(k)_0 - α^(k)_{-1} + {a^(k) cos(ωkR)} Δ^c
23:            α^(k)_{-1} ← α^(k)_0, α^(k)_0 ← α^(k)_{+1}
24:        end for
25:    end for
26: end function
```

2. 3 Algorithm Procedure and Analysis

Algorithm 1 shows a general procedure of our algorithm for the GF where $\{\cdot\}$ denotes precomputed values stored in look-up tables. The computational complexity is quantified by counting the number of arithmetic operations: multiplication/division (Mul/Div) and addition/subtraction (Add/Sub). We target the main filtering routine only (see line 15–25) because the precomputing routine has a negligible complexity as compared with the the main one. Our algorithm requires $2K + 1$ Muls and $3K + 5$ Adds/Subs for each element. Specifically, line 17 has 1 Mul and K Adds, line 19 has 2 Adds/Subs, line 20 has 3 Adds/Subs, and line 22 has $2K$ Muls and $2K$ Adds/Subs. Incidentally, there exist two effective techniques for an efficient implementation: loop unrolling for small-sized loops and ring buffers for updating short-time DCT/DST coefficients. Unrolling loops with respect to k such as line 21–24 eliminates loop

Table 1 The number of arithmetic operations per element in one-dimensional filtering.

Category	Method	Mul/Div*	Add/Sub*
	Convolution	$R+1$	$2R+1$
Recurs.	Deriche[18]	$4M$	$4M-2$
Recurs.	Farneback and Westin[21]	$4M$	$4M-2$
Recurs.	Young and van Vliet[19]	10	6
Recurs.	van Vliet et al.[20]	$2M$	$2M+2$
DCT-1	CII[27]	$4K$	$6K+2$
DCT-5	**Ours**	$2K+1$	$3K+5$

*In general, $R=\lceil 3\sigma \rceil$, $M=3$ and $K=2$.

counting routine and condition judgement. Ring buffers with the length of two eliminate the cyclic substitutions in line 23. Both of the techniques surely reduce the actual running time without a loss of accuracy.

Table 1 lists the number of the arithmetic operations of various algorithms for the GF: convolution ($R=\lceil 3\sigma \rceil$ is commonly-used), recursive Gaussian filters[18) 21)] ($M=3$ in general), and DCT-based algorithms including the CII[27)] ($K=3$) and our algorithm ($K=2$, as discussed later). Under the general parameter values, our algorithm achieves the fewest Muls/Divs in the constant-time Gaussian filters followed by the recursive Gaussian filter proposed by van Vliet et al.[20)] In two-dimensional filtering, our algorithm requires only 14 Muls per pixel; whereas van Vliet's algorithm requires 16 Muls per pixel, regardless of σ. In comparison with the CII, our algorithm reduces the number of arithmetic operations nearly by 50%. As another advantage of our algorithm, it enables us to perform filtering in one-pass process per dimension, which contributes to reduce the number of memory accesses. Any recursive Gaussian filter has a two-pass processing caused by its two feedback systems and the CII also has an extra process to construct integral images in advance. Thus, our algorithm clearly outperforms the state-of-the-art algorithms in terms of computational complexity.

3. Dual-domain Error Minimization

This section describes how to find an optimal parameter for our algorithm, called dual-domain error minimization. As useful examples, we specify cases of Gaussian kernel and Gaussian derivative kernels here.

3. 1 Spatial/Frequency truncation errors

Our algorithm requires two parameters: filter window radius R and the number of significant frequency components K. The former involves a truncation in spatial domain and the latter involves a truncation in (short-time) frequency domain. Since both truncations cause approximate error of kernels, we should determine them adequately. In general, convolution requires R only and it is common to calculate it from σ, e.g., $R=\lceil 3\sigma \rceil$, which supports most area of any Gaussian kernel. The CII[27)] carefully sets K in terms of computational complexity but fixes $R=\lceil \pi\sigma \rceil$. However, this R-fixed parameter determination has room for improvement in approximate accuracy. This is a major motivation of dual-domain error minimization.

Consider an approximate kernel truncated at K-th components in frequency domain. In order to facilitate mathematical analysis, (2) is redefined as $t \in \mathbb{R}$ by

$$\hat{h}(K,R,t)=\frac{1}{2}a^{(0)}+\sum_{k=1}^{K}\left[a^{(k)}C_t^{(k)}+b^{(k)}S_t^{(k)}\right]. \quad (12)$$

The spatial truncation operator is also given by

$$\mathscr{T}_R\left[h(t)\right]=\begin{cases} h(t) & \text{if } |t| \leqq R+\frac{1}{2} \\ 0 & \text{otherwise} \end{cases}.$$

Note that the period length after $\mathscr{T}_R[\cdot]$ is $2R+1$ and still $\omega=\frac{2\pi}{2R+1}$. We quantify the approximate error as

$$E(K,R)=\frac{\int_{-\infty}^{\infty}\left\{h(t)-\mathscr{T}_R[\hat{h}(K,R,t)]\right\}^2 dt}{\int_{-\infty}^{\infty}h^2(t)dt}, \quad (13)$$

which is interpreted as the relative error energy. More importantly, the approximate error consists of spatial truncation error and frequency truncation error. By decomposing the integral range of the numerator into $[-(R+\frac{1}{2}), R+\frac{1}{2}]$ and the other, we obtain

$$E_s(R)=\frac{\int_{-\infty}^{-(R+\frac{1}{2})}h^2(t)dt+\int_{R+\frac{1}{2}}^{\infty}h^2(t)dt}{\int_{-\infty}^{\infty}h^2(t)dt}, \quad (14)$$

$$E_f(K,R)=\frac{\int_{-(R+\frac{1}{2})}^{R+\frac{1}{2}}\left\{h(t)-\hat{h}(K,R,t)\right\}^2 dt}{\int_{-\infty}^{\infty}h^2(t)dt}, \quad (15)$$

Using the Parseval equation, (15) can be rewritten as

$$E_f(K,R)=\frac{\frac{\pi}{\omega}\sum_{k=K+1}^{\infty}\left[\left(a^{(k)}\right)^2+\left(b^{(k)}\right)^2\right]}{\int_{-\infty}^{\infty}h^2(t)dt}, \quad (16)$$

This representation clearly shows the role of frequency truncation. We observe that, if K is given, there is a tradeoff between $E_s(R)$ and $E_f(K,R)$ because larger R causes smaller $E_s(R)$ but larger $E_f(K,R)$. This implies that the optimal R corresponding to the given K exists, which is found via the error minimization problem

$$\arg\min_{R\in\mathbb{N}} \; E_s(R)+E_f(K,R), \quad \text{subject to a given } K.$$

After determining a desired K, e.g., from acceptable computational complexity, we can solve this problem by full search because $R \in \mathbb{N}$ is less than the half size of

an image. However, it seems time-consuming to naively compute $E_s(R)$ and $E_f(K,R)$ for each iteration.

3.2 Analysis on truncation errors

In order to clarify the properties of $E_s(R)$ and $E_f(K,R)$ and simplify them, we analyse the cases of the Gaussian kernel and the Gaussian derivative kernels. Note that we use dot accents such as $\dot{g}$ and $\ddot{g}$ for notational distinctions between them.

Definition 2. Original, first-derivative, and second-derivative Gaussian kernels, respectively, are defined by

$$g(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{t^2}{2\sigma^2}}, \tag{17}$$

$$\dot{g}(t) = \frac{d}{dt} g(t) = -\frac{t}{\sqrt{2\pi}\sigma^3} e^{-\frac{t^2}{2\sigma^2}}, \tag{18}$$

$$\ddot{g}(t) = \frac{d^2}{dt^2} g(t) = \frac{t^2-\sigma^2}{\sqrt{2\pi}\sigma^5} e^{-\frac{t^2}{2\sigma^2}}, \tag{19}$$

where $\sigma \in \mathbb{R}_+$ is the scale parameter.

Lemma 3. *The transform coefficients of $g(t)$, $\dot{g}(t)$, and $\ddot{g}(t)$, respectively, are approximated by*

$$a^{(k)} \approx \frac{\omega}{\pi} e^{-\frac{1}{2}\omega^2\sigma^2k^2}, \qquad b^{(k)} = 0, \tag{20}$$

$$\dot{b}^{(k)} \approx -\frac{\omega^2}{\pi} k e^{-\frac{1}{2}\omega^2\sigma^2k^2}, \qquad \dot{a}^{(k)} = 0, \tag{21}$$

$$\ddot{a}^{(k)} \approx \frac{\omega^3}{\pi} k^2 e^{-\frac{1}{2}\omega^2\sigma^2k^2}, \qquad \ddot{b}^{(k)} = 0. \tag{22}$$

Proof. Derive $a^{(k)}$ and $b^{(k)}$ first. By substituting (17) for (3) and using the Gaussian integral, we obtain

$$(2R+1)a^{(k)} = \frac{1}{\sqrt{2\pi}\sigma} \sum_{t=-R}^{R} e^{-\frac{t^2}{2\sigma^2} - i\omega kt} \approx e^{-\frac{1}{2}\sigma^2\omega^2k^2},$$

where

$$\sum_{t=-R}^{R} e^{-\frac{t^2}{2\sigma^2} - i\omega kt} = e^{-\frac{1}{2}\sigma^2\omega^2k^2} \sum_{t=-R}^{R} e^{-\frac{1}{2\sigma^2}\left(t+i\sigma^2\omega k\right)^2}$$
$$\approx e^{-\frac{1}{2}\sigma^2\omega^2k^2} \int_{-\infty}^{\infty} e^{-\frac{t^2}{2\sigma^2}} dt = \sqrt{2\pi}\sigma e^{-\frac{1}{2}\sigma^2\omega^2k^2}.$$

In the derivative cases, from differentiating (12) with respect to t, we obtain $\dot{a}_k = \omega k b_k$ and $\dot{b}_k = -\omega k a_k$; likewise, $\ddot{a}_k = \omega k \dot{b}_k$ and $\ddot{b}_k = -\omega k \dot{a}_k$ hold. □

Lemma 3 reveals that the three kernels have many zero (from shape symmetry) or almost-zero (from exponential attenuation) coefficients. Hence, we can sufficiently approximate them using small K.

Next, their truncation errors are clarified as follows:

Proposition 4. *If we assume that $1 \leqq \sigma$, then the spatial/frequency truncation errors of $g(t)$, $\dot{g}(t)$, and $\ddot{g}(t)$ are simplified or approximated as follows:*
i) Gaussian kernel:

$$E_s(R) = \operatorname{erfc}(\phi), \qquad E_f(K,R) \approx \operatorname{erfc}(\psi), \tag{23}$$

ii) First-derivative Gaussian kernel:

$$\dot{E}_s(R) = \operatorname{erfc}(\phi) + \frac{2}{\sqrt{\pi}}\phi e^{-\phi^2}, \tag{24}$$

$$\dot{E}_f(K,R) \approx \operatorname{erfc}(\psi) + \frac{2}{\sqrt{\pi}}\psi e^{-\psi^2}, \tag{25}$$

iii) Second-derivative Gaussian kernel:

$$\ddot{E}_s(R) = \operatorname{erfc}(\phi) + \frac{2}{\sqrt{\pi}}\phi e^{-\phi^2}\left(\frac{2}{3}\phi^2 - \frac{1}{3}\right), \tag{26}$$

$$\ddot{E}_f(K,R) \approx \operatorname{erfc}(\psi) + \frac{2}{\sqrt{\pi}}\psi e^{-\psi^2}\left(\frac{2}{3}\psi^2 + 1\right), \tag{27}$$

where $\phi = \frac{2R+1}{2\sigma}$, $\psi = \pi\sigma\frac{2K+1}{2R+1}$, and $\operatorname{erfc}(\cdot)$ is the complementary error function.

Proof. We use the following formulae in this proof:

$$\int_u^v a^{2n}x^{2n}e^{-a^2x^2}dx = \frac{1}{2a}\left[\Gamma\left(n+\frac{1}{2}, a^2x^2\right)\right]_v^u,$$

where $\Gamma(\cdot,\cdot)$ is the incomplete Gamma function, which has the recurrence formulae

$$\Gamma\left(\frac{1}{2}, a^2x^2\right) = \sqrt{\pi}\operatorname{erfc}(ax),$$
$$\Gamma\left(m+1, a^2x^2\right) = m\Gamma\left(m, a^2x^2\right) + a^{2m}x^{2m}e^{-a^2x^2}.$$

We first derive (23). Its normalization factor for (13) is

$$\int_{-\infty}^{\infty} g^2(t)dt = \frac{1}{2\pi\sigma^2}\int_{-\infty}^{\infty} e^{-\frac{t^2}{\sigma^2}}dt = \frac{1}{2\sqrt{\pi}\sigma}.$$

The spatial truncation error is simplified as

$$E_s(R) = \frac{2\int_{R+\frac{1}{2}}^{\infty} g^2(t)dt}{\int_{-\infty}^{\infty} g^2(t)dt} = \frac{2}{\sqrt{\pi}\sigma}\int_{R+\frac{1}{2}}^{\infty} e^{-\frac{t^2}{\sigma^2}}dt$$
$$= \frac{1}{\sqrt{\pi}}\left[\Gamma\left(\frac{1}{2}, \frac{t^2}{\sigma^2}\right)\right]_{\infty}^{R+\frac{1}{2}} = \operatorname{erfc}\left(\frac{2R+1}{2\sigma}\right).$$

The frequency truncation error is approximated via continuous relaxation by

$$E_f(K,R) \approx \frac{\frac{\pi}{\omega}\int_{K+\frac{1}{2}}^{R+\frac{1}{2}} a_k^2 dk}{\int_{-\infty}^{\infty} g^2(t)dt} \approx \frac{2\omega\sigma}{\sqrt{\pi}}\int_{K+\frac{1}{2}}^{R+\frac{1}{2}} e^{-\omega^2\sigma^2k^2}dk$$
$$= \frac{1}{\sqrt{\pi}}\left[\Gamma\left(\frac{1}{2}, \omega^2\sigma^2k^2\right)\right]_{R+\frac{1}{2}}^{K+\frac{1}{2}} \approx \operatorname{erfc}\left(\pi\sigma\frac{2K+1}{2R+1}\right),$$

where $\operatorname{erfc}(\pi\sigma) \approx 0$ is negligible under $1 \leqq \sigma$.

The cases of the other kernels are also derived in the same manner. For the first-derivative Gaussian kernel,

$$\dot{E}_s(R) = \frac{2}{\sqrt{\pi}}\left[\Gamma\left(\frac{3}{2}, \frac{t^2}{\sigma^2}\right)\right]_{\infty}^{R+\frac{1}{2}},$$
$$\dot{E}_f(K,R) \approx \frac{2}{\sqrt{\pi}}\left[\Gamma\left(\frac{3}{2}, \omega^2\sigma^2k^2\right)\right]_{R+\frac{1}{2}}^{K+\frac{1}{2}}.$$

Likewise, the second-derivative Gaussian kernel creates

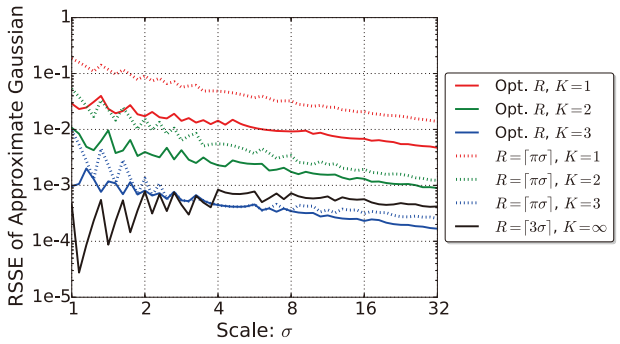

Fig. 1 As compared with the approximation employed in CII ($R = \lceil \pi\sigma \rceil$), the dual-domain error minimization (Opt. R) can significantly reduce approximate error, in particular, for smaller K. For $\sigma \geqq 4$, it provides less error than a common truncated Gaussian kernel ($R = \lceil 3\sigma \rceil$).

$$\ddot{E}_s(R) = \frac{4}{3\sqrt{\pi}} \left[\Gamma\left(\frac{5}{2}, \frac{t^2}{\sigma^2}\right) - 2\Gamma\left(\frac{3}{2}, \frac{t^2}{\sigma^2}\right) + \Gamma\left(\frac{1}{2}, \frac{t^2}{\sigma^2}\right) \right]_{\infty}^{R+\frac{1}{2}},$$

$$\ddot{E}_f(K, R) \approx \frac{4}{3\sqrt{\pi}} \left[\Gamma\left(\frac{5}{2}, \omega^2\sigma^2k^2\right) \right]_{R+\frac{1}{2}}^{K+\frac{1}{2}}.$$

Expanding all the above $\Gamma(\cdot, \cdot)$ by its recurrence formulae, we obtain (24), (25), (26) and (27). □

Evidently, each Gaussian-based kernel shows a tradeoff between both truncation errors with respect to R, i.e., $E(K, R)$ is unimodal. Thereby, in our dual-domain error minimization, the optimal R corresponding to a desired K can be found efficiently by the binary search.

4. Experiments and Discussion

This section validates the efficiency of our algorithm through some experiments using natural images. The test image is the standard image "N2" (grayscale, 2560×2048) in the ISO/JIS-SCID[41]. Our test environment mounts on an Intel Core i5 2.67GHz CPU and 8GB main memory. The competitors are convolution, van Vliet's recursive filter[20], and our algorithm. We used cv::GaussianBlur function in OpenCV 2.4.5[42] for the convolution and self-produced codes written in C++ for the others. Note that all the algorithms showed the almost same tendencies regardless of target images since the performance are independent of image contents.

4.1 Effectivity of Parameter Optimization

Figure 1 plots the relationship between σ and root-sum-square-error (RSSE) of the Gaussian kernel defined by (2). Evidently, the dual-domain error minimization (Opt. R) produces less approximate error than the approximation employed in the CII ($R = \lceil \pi\sigma \rceil$) , in addition to provide an even higher accuracy than a common truncated Gaussian kernel ($R = \lceil 3\sigma \rceil$) for $\sigma \geqq 4$. Hence, our algorithm surely improves approximate accuracy, particularly for smaller K.

Figure 2 shows the root truncation error for Gaussian kernel and Gaussian derivative kernels in our algorithm. For a fair comparison, we normalize the area of each kernel to one in advance to equalize the dynamic range of output images. If a sufficient accuracy is assumed as 2.5% tolerance (gray-dashed), we set $K = 2$ for Gaussian kernel and $K = 3$ for first- and second-derivative Gaussian kernels. These parameter values are used in the successive experiments. Moreover, our algorithm has a high usability because of its accuracy control method of determining K only.

4.2 Evaluation of accuracy

We confirm the quality of the actual filter output by using natural images. Figure 3 plots the relationship between scale σ and output accuracy (the peak signal-to-noise ratio; PSNR [dB]). We assume that $\pm 5\sigma$-supported convolution outputs an exact one. In all the filters, our algorithm shows the most stable performance. It is clearly higher than the recursive filter, which faces accuracy dropping over $\sigma > 16$. This stability of accuracy is an advantage for the scale-space analysis and high-resolutional image filtering.

Figure 4 lists actual output images and their 100×amplified error image to facilitate visual assessment. Note that they achieve almost the same PSNR. Convolution and van Vliet's recursive GF have errors around edges. Our algorithm shows high-frequency error over an entire image. Evidently, this is caused by the frequency truncation.

4.3 Evaluation of running time

Figure 5 plots the relationship between scale σ versus the filtering time. For any kernel, our algorithm shows stably-high performance regardless of σ, comparable to convolution for $\sigma = 1$ and approximately 2.5× faster than the recursive filtering. The reason why our algorithm achieved higher performance rate expected in Table 1 as compared with van Vliet's algorithm is memory access cost as discussed in the preceding section. Thus, our algorithm achieves significantly faster filtering.

5. Conclusions

This paper presented an efficient $O(1)$ GF and its derivative ones that provide high accuracy in low computational complexity over a wide range of σ. They utilized the spectrum sparsity of the kernels and optimal parameter determination via dual-domain error minimization. In experiments of image filtering, our al-

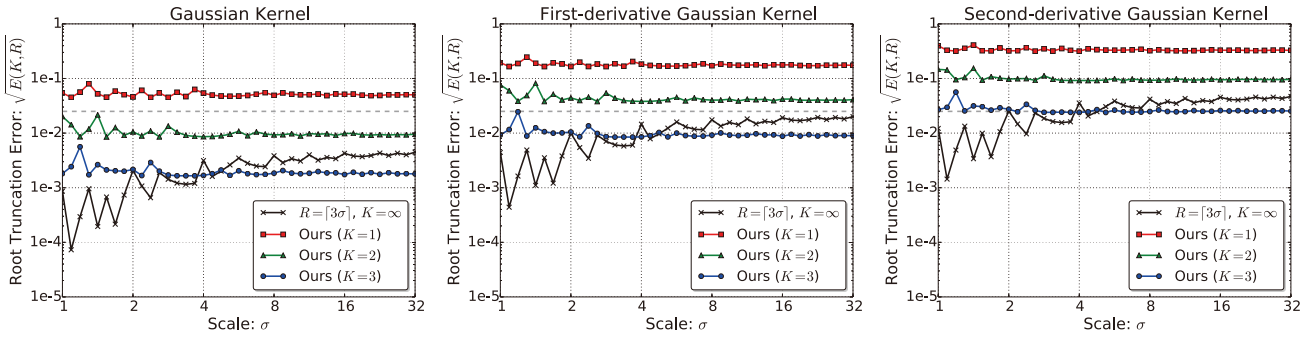

Fig. 2 Scale σ versus kernel truncation error. If a sufficient accuracy is assumed as 2.5% tolerance (gray-dashed), we set $K = 2$ for the GF and $K = 3$ for the first- and second-derivative GF.

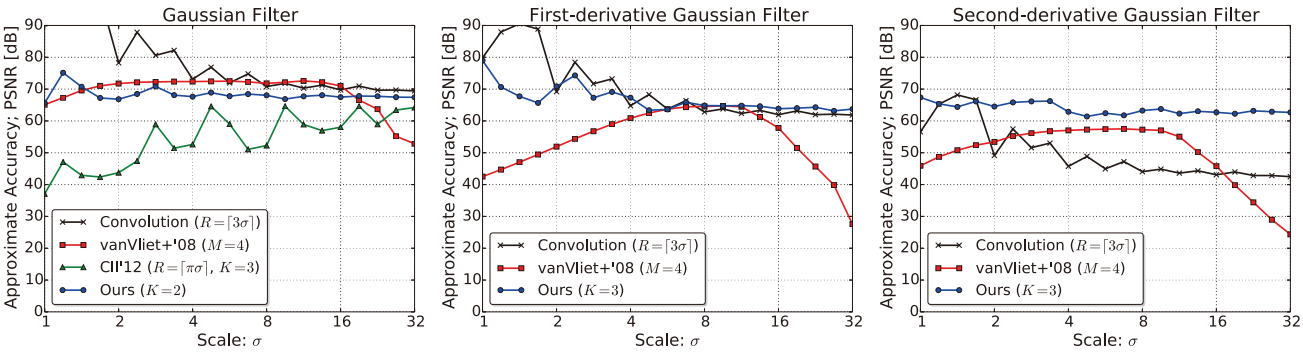

Fig. 3 Scale σ versus output accuracy (PSNR) [dB] in image filtering. We assume $\pm 5\sigma$-supported convolution as an exact output. Our algorithm produces a stably-high accuracy over wide range of σ; whereas van Vliet's recursive filter shows an accuracy degradation over $16 < \sigma$.

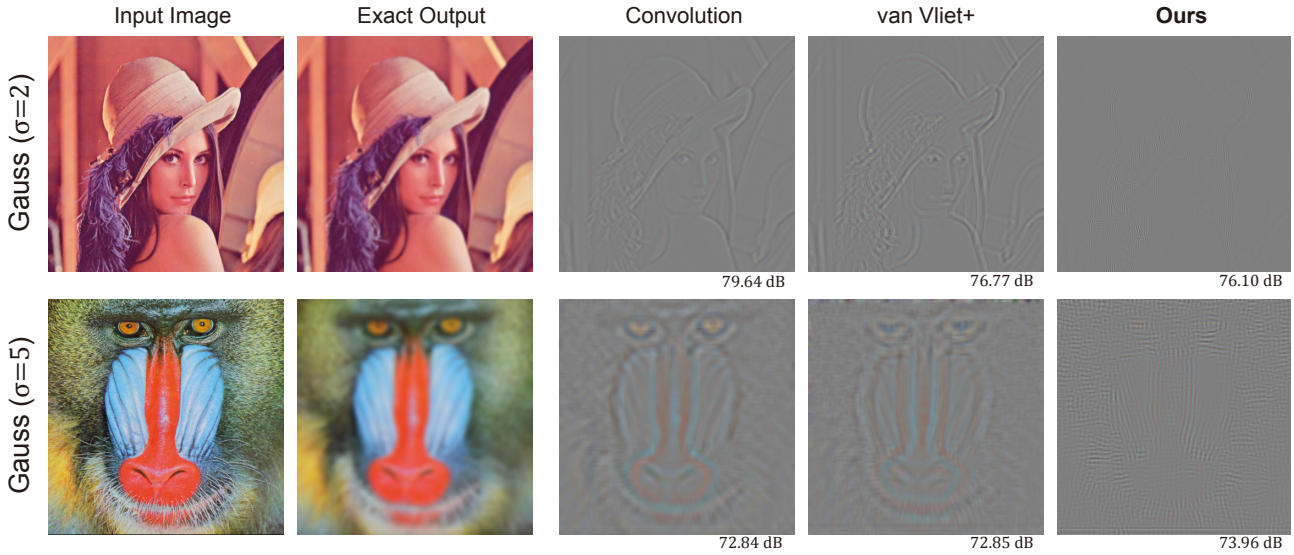

Fig. 4 Visual assessment of GF algorithms. The error images are 100× amplified to facilitate visualization. The parameters values are $R = \lceil 3\sigma \rceil$ for convolution, $M = 4$ for van Vliet's recursive GF, and $K = 3$ for ours to achieve almost the same PSNR.

gorithm showed superiority to the recursive Gaussian filters in computational complexity, approximate accuracy, and accuracy stability. We believe that this improvement contributes to various modern algorithms in many image processing applications. As future work, we will extend this idea to more generalized filters such as the bilateral filter[2) 4)].

Acknowledgement

This work was supported by Super Cluster Program in Japan Science and Technology Agency.

References

1) T. Lindeberg, "Scale-space for discrete signals," IEEE Trans. Pattern Anal. Mach. Intell., vol.12, no.3, pp.234–254, 1990.

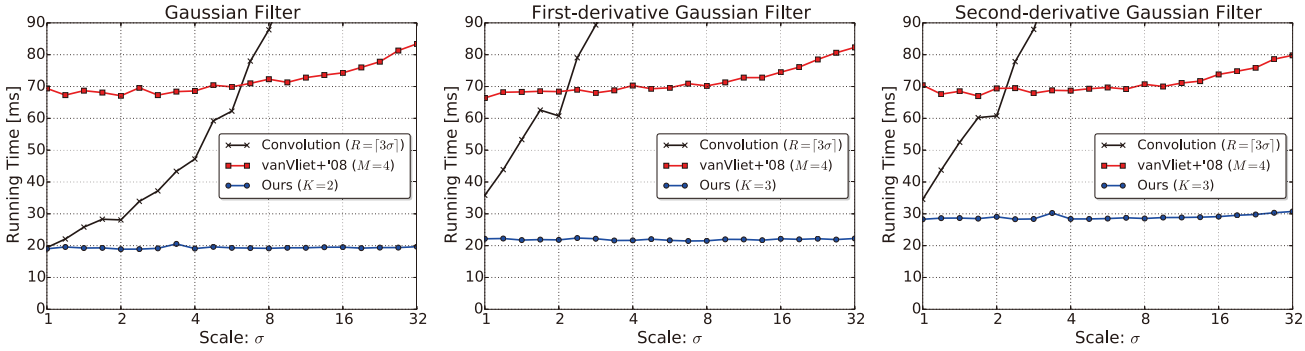

Fig. 5 The runnning time of our algorithm is stable over a wide range of σ, comparable to a $\pm 3\sigma$-supported convolution of $\sigma = 1$, and approximately 2.5× faster than van Vliet's recursive filter.

2) V. Aurich and J. Weule, "Non-linear Gaussian filters performing edge preserving diffusion," Mustererkennung 1995, 17. DAGM-Symposium, pp.538–545, 1995.
3) S.M. Smith and J.M. Brady, "SUSAN — a new approach to low level image processing," Int. J. Comput. Vis. (IJCV), vol.23, no.1, pp.45–78, 1997.
4) C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," Proc. IEEE Int. Conf. Comput. Vis. (ICCV), pp.839–846, Jan. 1998.
5) D.G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Comput. Vis. (IJCV), vol.60, no.2, pp.91–110, Nov. 2004.
6) S. Baker and T. Kanade, "Limits on super-resolution and how to break them," IEEE Trans. Pattern Anal. Mach. Intell., vol.24, no.9, pp.1167–1183, Sept. 2002.
7) E. Reinhard, M. Stark, P. Shirley, and J. Ferwerda, "Photographic tone reproduction for digital images," ACM Trans. Graph. (Proc. SIGGRAPH), pp.267–276, July 2002.
8) Q. Yang, R. Yang, J. Davis, and D. Nister, "Spatial-depth super resolution for range images," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), June 2007.
9) R. Marfil, L. Molina-Tanco, A. Bandera, J. Rodríguez, and F. Sandoval, "Pyramid segmentation algorithms revisited," Pattern Recognition, vol.39, no.8, pp.1430–1451, Aug. 2006.
10) L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," IEEE Trans. Pattern Anal. Mach. Intell., vol.20, no.11, pp.1254–1259, 1998.
11) J. Canny, "A computational approach to edge detection," IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-8, no.6, pp.679–698, Nov. 1986.
12) M. Basu, "Gaussian-based edge-detection methods — a survey," IEEE Trans. Syst. Man Cybern. C, Appl. Rev., vol.32, no.3, pp.252–260, Aug. 2002.
13) Q. Yang, K.H. Tan, and N. Ahuja, "Real-time O(1) bilateral filtering," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.557–564, IEEE, June 2009.
14) K.N. Chaudhury, D. Sage, and M. Unser, "Fast O(1) bilateral filtering using trigonometric range kernels," IEEE Trans. Image Process., vol.20, no.12, pp.3376–82, Dec. 2011.
15) K.N. Chaudhury, "Acceleration of the shiftable O(1) algorithm for bilateral filtering and nonlocal means," IEEE Trans. Image Process., vol.22, no.4, pp.1291–1300, April 2013.
16) R.N. Bracewell, The Fourier Transform and Its Applications, McGraw-Hill, New York, 1965.
17) W.M. Wells, "Efficient synthesis of Gaussian filters by cascaded uniform filters," IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-8, no.2, pp.234–239, March 1986.
18) R. Deriche, "Fast algorithms for low-level vision," IEEE Trans. Pattern Anal. Mach. Intell., vol.12, no.1, pp.78–87, 1990.
19) I.T. Young and L.J. van Vliet, "Recursive implementation of the Gaussian filter," Signal Process., vol.44, no.2, pp.139–151, 1995.
20) L. van Vliet, I. Young, and P. Verbeek, "Recursive Gaussian derivative filters," Proc. Int. Conf. Pattern Recognition (ICPR), pp.509–514, 1998.
21) G. Farnebäck and C. Westin, "Improving Deriche-style recursive Gaussian filters," J. Math. Imaging and Vision, vol.26, no.3, pp.293–299, Nov. 2006.
22) A. Bernardino and J. Santos-Victor, "Fast IIR isotropic 2-D complex Gabor filters with boundary initialization," IEEE Trans. Image Process., vol.15, no.11, pp.3338–3348, Nov. 2006.
23) K. Sugimoto and S. Kamata, "Fast Gaussian filter with second-order shift property of DCT-5," Proc. IEEE Int. Conf. Image Process. (ICIP), pp.514–518, Sept. 2013.
24) M. Unser, A. Aldroubi, and M. Eden, "B-spline signal processing. II. efficiency design and applications," IEEE Trans. Signal Process., vol.41, no.2, pp.834–848, 1993.
25) Y.P. Wang and S. Lee, "Scale-space derived from B-splines," IEEE Trans. Pattern Anal. Mach. Intell., vol.20, no.10, pp.1040–1055, 1998.
26) M. Hussein, F. Porikli, and L. Davis, "Kernel integral images: A framework for fast non-uniform filtering," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.1–8, June 2008.
27) E. Elboher and M. Werman, "Cosine integral images for fast spatial and range filtering," Proc. IEEE Int. Conf. Image Process. (ICIP), pp.89–92, Sept. 2011.
28) F.C. Crow, "Summed-area tables for texture mapping," ACM Trans. Graph., vol.18, no.3, pp.207–212, 1984.
29) P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), pp.511–518, 2001.
30) C. Rader and B. Gold, "Digital filter design techniques in the frequency domain," Proc. IEEE, vol.55, no.2, pp.149–171, 1967.
31) R. Lyons, "The sliding DFT," IEEE Signal Processing Mag., vol.20, no.2, pp.74–80, March 2003.
32) H. Kikuchi, M. Nakashizuka, H. Watanabe, S. Watanabe, and N. Tomisawa, "Fast wavelet transform and its application to detecting detonation," IEICE Trans. Fundamentals, vol.E75-A, no.8, pp.980–987, 1992.
33) K. Sugimoto and S. Kamata, "Fast image filtering by DCT-based kernel decomposition and sequential sum update," Proc. IEEE Int. Conf. Image Process. (ICIP), pp.125–128, Oct. 2012.
34) K. Sugimoto and S. Kamata, "Fast Gaussian filter using DCT-V," Int. Workshop on Advanced Image Technology (IWAIT), pp.338–343, Jan. 2013.
35) G. Strang, "The discrete cosine transform," SIAM Review, vol.41, no.1, pp.135–147, Jan. 1999.
36) P. Yip and K. Rao, "On the shift property of DCT's and DST's," IEEE Trans. Acoust., Speech, Signal Process., vol.35, no.3, pp.404–406, March 1987.
37) L.N. Wu, "Comments on "On the shift property of DCT's and DST's"," IEEE Trans. Acoust., Speech, Signal Process., vol.38, no.1, pp.186–188, Jan. 1990.
38) N. Murthy and M. Swamy, "On the computation of running discrete cosine and sine transform," IEEE Trans. Signal Process., vol.40, no.6, pp.1430–1437, June 1992.
39) J. Xi and J. Chicharo, "Computing running DCT's and DST's based on their second-order shift properties," IEEE Trans. Circuits Syst. I, vol.47, no.5, pp.779–783, May 2000.
40) V. Kober, "Fast algorithms for the computation of sliding discrete sinusoidal transforms," IEEE Trans. Signal Process., vol.52, no.6, pp.1704–1710, June 2004.
41) "ISO/JIS-SCID: graphic technology —prepress digital data exchange— standard colour image data," 1995.
42) "Open Source Computer Vision Library (OpenCV)."

Kenjiro Sugimoto received the B.E. and M.E. degrees in 2007 and 2009 from Kurume National College of Technology, Japan, and Waseda University, Japan, respectively. He is currently pursuing a Ph.D. degree at Graduate School of Information, Production and Systems, Waseda Univeristy, and had been a Research Fellow of the Japan Society for the Promotion of Science (JSPS) from 2010 to 2012. His current research interests are in image processing and pattern recognition. He is a student member of IEEE and IEICE.

Sei-ichiro Kamata received the M.S. degree in computer science from Kyushu University, Fukuoka, Japan, in 1985, and the Doctor of Computer Science, Kyushu Institute of Technology, Kitakyushu, Japan, in 1995. From 1985 to 1988, he was with NEC, Ltd., Kawasaki, Japan. In 1988, he joined the faculty at Kyushu Institute of Technology. From 1996 to 2001, he has been an Associate Professor in the Department of Intelligent System, Graduate School of Information Science and Electrical Engineering, Kyushu University. Since 2003, he has been a professor in the Graduate School of Information, Production and Systems, Waseda University. In 1990 and 1994, he was a Visiting Researcher at the University of Maine, Orono. His research interests include image processing, pattern recognition, image compression, and space-filling curve application. He is a member of IEEE, IEICE and ITE in Japan.