INVITED PAPER *Special Section on Award-winning Papers*

XY-Separable Scale-Space Filtering by Polynomial Representations and Its Applications

Gou KOUTAKI[†a)] ***and*** **Keiichi UCHIMURA**[†]**,** ***Members***

SUMMARY In this paper, we propose the application of principal component analysis (PCA) to scale-spaces. PCA is a standard method used in computer vision. Because the translation of an input image into scale-space is a continuous operation, it requires the extension of conventional finite matrix-based PCA to an infinite number of dimensions. Here, we use spectral theory to resolve this infinite eigenvalue problem through the use of integration, and we propose an approximate solution based on polynomial equations. In order to clarify its eigensolutions, we apply spectral decomposition to Gaussian scale-space and scale-normalized Laplacian of Gaussian (sLoG) space. As an application of this proposed method, we introduce a method for generating Gaussian blur images and sLoG images, demonstrating that the accuracy of such an image can be made very high by using an arbitrary scale calculated through simple linear combination. Furthermore, to make the scale-space filtering efficient, we approximate the basis filter set using Gaussian lobes approximation and we can obtain XY-Separable filters. As a more practical example, we propose a new Scale Invariant Feature Transform (SIFT) detector.

key words: *scale-space, spectral decomposition, SIFT*

1. Introduction

Image features vary depending on object pose, camera view, and illumination; therefore, extracting invariant image features for such varying characteristics is an important task in computer vision applications. Scale-space filtering is an important and basic technique for acquiring a scale-invariant image feature. Previously, there has been extensive research on scale-space filtering [1]–[4]. The family of Gaussian scale-space filtering (i.e., Gaussian, derivative Gaussian, and Gaussian of Laplacian) are used to extract image features, such as multi-scale edges, keypoints, and saliency [5]–[7]. To obtain the scale-invariant filter's response, filter kernels are normalized by a scale parameter called the scale-normalized scale-space [8].

To construct the scale-space, multiple images are generated by filtering with multiple scales by performing discretization along the scale axis. However, increasing the number of scale-space images increases computational cost. Thus, a trade-off between the scale resolution and the computational cost exists. To construct the scale-space efficiently, Perona applied singular value decomposition (SVD) to scale-space images [9]. In their approach, an anisotropic directional filter's kernel, such as a 2D-Gabor filter, is decomposed (steerable filter [10]–[12] and scalable-steerable filter [13]). Then, multi-scale edges are extracted. This method has been improved by using a fast XY-separable filter [33]. By using scalable-steerable decomposition, a multiple-filter kernel can be represented by a small number of basis functions. Note that the decomposed filter functions are discretized with a unit step of scale, angle, and XY.

Koutaki *et al.* represented a scale-space filter as the continuous cubic polynomials of scale parameters using spectral decomposition [14]. A scale-space image with arbitrary scale can be represented by a simple linear combination of basis images and scale parameters. The analytical form of the eigen filter's kernel was given, and the Gaussian or sLoG images within scale range $s \in [1.0, 5.0]$ were accurately reconstructed from only 4-basis images [15], [16]. This cubic polynomial representation is reasonable for many vision applications because image features can be detected at the zero-crossing point along the scale parameter, such as $\partial I(x, y, s)/\partial s = 0$. Thus, it is easy to determine the optimal scale analytically. This scale-space filtering has been extended to Affine scale-space [17]. However, those obtained eigen filter is not XY-separable; therefore, computational bottleneck occurs when filtering in a practical application.

In this paper, we introduce the polynomial represented scale-space filtering using spectral decomposition of scale-space filters such as Gaussian and sLoG. Furthermore, we improve the filtering to XY-separable form by Gaussian lobes approximation. In conclusion, our contributions are as follows.

1. We propose and demonstrate a method for compressing scale-space images using continuous PCA (through spectral decomposition) to obtain numerical solutions.
2. We clarify the eigensolutions of Gaussian scale-space and sLoG space.
3. We demonstrate XY-separable scale-space filtering using Gaussian lobes approximation.
4. As a vision application, we introduce a new keypoint detector that is faster and has better repeatability than speeded up robust features (SURF) and scale invariant feature transform (SIFT).

1.1 Related Works

Many studies have explored image feature detection using scale-space, i.e., edge, keypoint, and saliency detection. Some examples of applications include object detec-

Manuscript received October 24, 2016.
Manuscript revised December 21, 2016.
Manuscript publicized January 11, 2017.

[†]The authors are with Kumamoto University, Kumamoto-shi, 860–8555 Japan.
a) E-mail: koutaki@cs.kumamoto-u.ac.jp

DOI: 10.1587/transinf.2016AWI0001

tion, tracking, image registration, 3D reconstruction, and image retrieval. In particular, the keypoint detector and descriptor for a corresponding search has been used frequently in recent years.

In the early 1980s, Marr et al. approximated the LoG as a Differential of Gaussian (DoG) [1]. In the 1990s, Lindberg proposed blob detector using 3D-local max search in sLoG space [8]. In 1999, Lowe proposed SIFT [18]. The SIFT detector is blob detector using a DoG. SIFT consists of a scale-orientation invariant detector and a descriptor. SIFT has been used in many applications. Other scale-invariant detectors have been proposed, e.g., the Hessian-Laplace and the Hessian-Harris detector [19].

After the rise of SIFT, many improved methods have also been proposed [20], [21]. SURF uses a fast Hessian that approximates the second derivative of a Gaussian as a combination of box filters [22]. In addition, box filtering can be applied quickly using an integral image technique. Features from accelerated segment test (FAST) is a previously proposed keypoint detector [23]. FAST does not use scale-space, and it must handle changes of scale by resizing an image. Oriented FAST and Rotated BRIEF (ORB) apply the same approach [24]. Those detectors are widely used with smartphones or microcomputers with low CPU power. KAZE features use non-linear scale-space [25], and the scale-invariant feature detector with error resilience (SIFER) uses a cosine-modulated Gaussian filter to construct scale-space [26], [27]. In addition, methods that employ direct polynomial approximation of filter kernels or multi-template images have also been proposed [28]–[30].

The remainder of this paper is organized as follows. Section 2 describes an analysis of scale-space filtering of Gaussian and a sLoG. Section 3 provides some numerical examples and the simulation results by proposed method. Section 4 presents a new improvement of the scale-space filtering using Gaussian lobes approximation for XY-separable filtering. Section 5 provides an application of the image matching, which demonstrated that the proposed method improves the repeatability and the computational time. Section 6 provides the concluding remarks.

2. Scale-Space Analysis

In this section, we analyze two-types of scale-spaces: Gaussian scale-space, sLoG space.

2.1 Gaussian Scale-Space

For a given input image $f(x, y)$, its corresponding scale-space image $I(x, y, s)$ with scale parameter s ($s_1 \le s \le s_2$) can be defined using convolution with a Gaussian kernel $g(x, y, s)$:

$$I(x', y', s) = \iint g(x, y, s) f(x - x', y - y') dx dy. \quad (1)$$

The 2D-Gaussian kernel $g(x, y, s)$ is defined using:

$$g(x, y, s) = \frac{1}{2\pi s^2} \exp\left(-\frac{x^2 + y^2}{2s^2}\right), s_1 \le s \le s_2.$$

This can be expanded in a series of eigenfunctions $\varphi_i(s)$ in the scale parameter s:

$$g(x, y, s) = \sum_{i=0}^{\infty} \left(\int_{s_1}^{s_2} g(x, y, t) \varphi_i(t)\, dt \right) \varphi_i(s).$$

The series in the equation above can be approximated by truncating it to N terms:

$$g(x, y, s) \approx \sum_{i=0}^{N} \left(\int_{s_1}^{s_2} g(x, y, t) \varphi_i(t)\, dt \right) \varphi_i(s). \quad (2)$$

Substituting this into Eq. (1), we obtain:

$$\begin{aligned} I(x', y', s) \approx & \iint \sum_{i=0}^{N} \left(\int_{s_1}^{s_2} g(x, y, t) \varphi_i(t)\, dt \right) \varphi_i(s) \\ & \cdot f(x - x', y - y') dx dy. \end{aligned}$$

By then changing the order of integration of $dxdy$ and dt, we obtain:

$$\begin{aligned} I(x', y', s) &\approx \sum_{i=0}^{N} \left\{ \iint \left(\int_{s_1}^{s_2} g(x, y, t) \varphi_i(t)\, dt \right) \right. \\ &\quad \left. \cdot f(x - x', y - y') dx dy \right\} \varphi_i(s) \\ &= \sum_{i=0}^{N} \varphi_i(s) \cdot \\ &\quad \left\{ \iint F_i(x, y) f(x - x', y - y') dx dy \right\} \\ &\equiv \sum_{i=0}^{N} \varphi_i(s)\, q_i(x', y'). \end{aligned} \quad (3)$$

where $F_i(x, y)$ is defined as:

$$F_i(x, y) = \int_{s_1}^{s_2} g(x, y, t) \varphi_i(t)\, dt. \quad (4)$$

Here, $F_i(x, y)$, which can be considered as a 2D-image, is called an *eigenimage*. Equation (3) can be interpreted as a Gaussian blurred image of scale s obtained by a linear combination of q_i and $\varphi_i(s)$, where the q_i are obtained by convolving the input image f and N-eigenimages $F_i(x, y)$.

To calculate the eigenfunctions, we try applying PCA to the Gaussian kernel. In the field of computer vision, PCA is generally understood as a standard method of compressing data and is used in processes such as the eigenface method or the subspace method. In the subspace method, for example, the eigenfunctions are obtained by solving the following $N \times N$ matrix eigenvalue problem:

$$\mathbf{C}\varphi = \lambda\varphi. \quad (5)$$

The factor $\mathbf{C}$ above represents a covariance matrix defined by N images $g_1, g_2, \ldots, g_N$:

$$\mathbf{C} = \begin{bmatrix} \langle g_1, g_1 \rangle & \langle g_1, g_2 \rangle & \cdots & \langle g_1, g_N \rangle \\ \langle g_2, g_1 \rangle & \langle g_2, g_2 \rangle & \cdots & \langle g_2, g_N \rangle \\ \vdots & \vdots & \vdots & \vdots \\ \langle g_N, g_1 \rangle & \langle g_N, g_2 \rangle & \cdots & \langle g_N, g_N \rangle \end{bmatrix}. \quad (6)$$

where $\langle g_i, g_j \rangle$ is the inner product of g_i and g_j.

However, because the scale parameter s is continuous, it is difficult to apply this matrix-based PCA to scale-space compression. In the case where $N \to \infty$, it is necessary to expand the eigenvalue problem; in the functional analysis of mathematics, this approach is known as spectral theory. By applying spectral theory to Eq. (5), the matrix eigenvalue problem can be transformed into the following Fredholm integral equation:

$$\int_{s_1}^{s_2} K(t,s)\,\varphi(t)\,dt = \lambda\varphi(s), \quad (7)$$

where $K(t, s)$ is the integral kernel and is defined as:

$$\begin{aligned} K(s,t) &= \iint g(x,y,s)\,g(x,y,t)\,dxdy \\ &= \frac{1}{2\pi\left(s^2+t^2\right)}. \end{aligned} \quad (8)$$

If the integral kernel is non-zero, symmetric, and finite, Eq. (7) has a unique solution; nevertheless, the integral equation remains difficult to be solved exactly except with a set of specific integral kernels. Therefore, we propose a solution by using a polynomial approximation:

$$\begin{aligned} \varphi_i(s) &= a_i^0 + s a_{i,1} + s^2 a_{i,2} + \cdots + s^N a_{i,N} \\ &= \left(1, s, s^2, \cdots, s^N\right) \cdot \boldsymbol{a}_i. \end{aligned} \quad (9)$$

By multiplying both sides of Eq. (7) by the polynomials $1, s, s^2, \cdots, s^N$ and then integrating, Eq. (7) is transformed into the following generalized eigenvalue problem of an $(N+1) \times (N+1)$ matrix:

$$\mathbf{K}\boldsymbol{a} = \lambda \mathbf{S}\boldsymbol{a}. \quad (10)$$

The elements of $\mathbf{K}$, $\mathbf{S}$ here are defined as:

$$K_{i+1\,j+1} = \frac{1}{2\pi}\iint \frac{s^j t^i}{s^2+t^2} ds dt, \quad (11)$$

$$S_{i+1\,j+1} = \int s^{i+j} ds = \frac{s^{1+i+j}}{1+i+j}. \quad (12)$$

Here, $0 \le i, j \le N$. By solving for the $(N+1)$ eigenvalues λ_i and the eigenvector $\boldsymbol{a}_i$ in Eq. (10), the eigenfunctions $\varphi_i(s)$ in Eq. (9) can be obtained.

To calculate the eigenimage F_i the following equation can be obtained by substituting Eq. (4) into Eq. (9):

$$\begin{aligned} F_i(x,y) &= \int_{s1}^{s2} g(x,y,s)\varphi_i(s)\,ds \\ &= -\sum_{n=0}^{N} \frac{a_{i,n}}{2^{3/2}\pi r}\left(\frac{r}{2^{1/2}}\right)^n \Gamma\left(\frac{1-n}{2}, \frac{r^2}{2s_1^2}, \frac{r^2}{2s_2^2}\right), \end{aligned} \quad (13)$$

where $r = \sqrt{x^2+y^2}$ and Γ is a complete gamma function defined as:

$$\Gamma(p, t_1, t_2) = \int_{t_1}^{t_2} t^{p-1}\exp(-t)\,dt, \quad (14)$$

which can be calculated accurately using a continued fraction expansion [31].

2.2 Scale Normalized LoG Space

In the same way as Sect. 2.1, we show the eigensolutions of Scale normalized LoG space (sLoG). sLoG is used for scale invariant edge detection and scale invariant feature transform (SIFT). It is important on computer vision's application.

The sLoG space is defined as the following equation which is second-order differentiation and normalization constant s^2 for the Gaussian kernel.

$$\begin{aligned} &I^{LoG}(x', y', s) = \\ &\iint s^2 \nabla^2 g(x,y,s) f(x-x', y-y') dxdy, \end{aligned} \quad (15)$$

Here, $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. Then, by using the relationship of the diffusion equation,

$$s\nabla^2 g(x,y,s) = \frac{\partial}{\partial s} g(x,y,s),$$

Equation (15) is transformed to the following equation.

$$\begin{aligned} &I^{LoG}(x', y', s) = \\ &\iint s\frac{\partial g(x,y,s)}{\partial s} f(x-x', y-y') dxdy. \end{aligned} \quad (16)$$

In the same way as Eq. (1)~Eq. (4), Eq. (16) can be expanded by eigenfunctions. The integral kernel of sLoG (equivalent to Eq. (8)) is defined as:

$$\begin{aligned} &K^{LoG}(s,t) \\ &= \iint st \frac{\partial g(x,y,s)}{\partial s}\frac{\partial g(x,y,t)}{\partial t} dxdy \\ &= \frac{4s^2t^2}{\pi\left(s^2+t^2\right)^3}. \end{aligned} \quad (17)$$

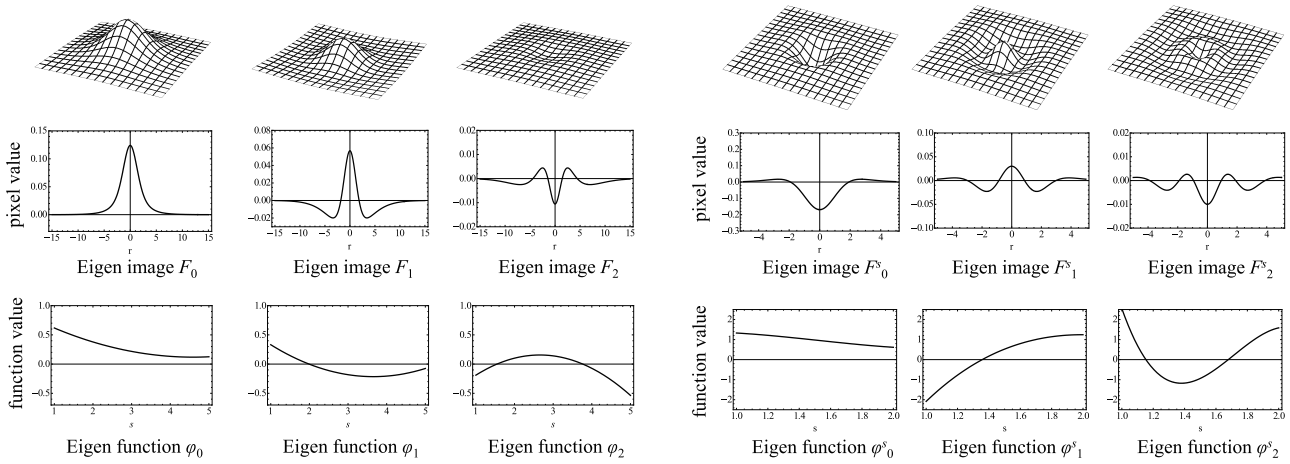
In order to solve the above integral equation, we transform the integral equation to the matrix-based generalized eigenvalue problem by the polynomial approximation. Then the elements of the matrix are obtained as:

$$K^{LoG}_{i+1\,j+1} = \frac{4}{\pi}\iint \frac{s^{j+2}t^{i+2}}{\left(s^2+t^2\right)^3} ds dt, \quad (18)$$

$$S^{LoG}_{i+1\,j+1} = \int s^{i+j} ds = \frac{s^{1+i+j}}{1+i+j}. \quad (19)$$

Here, $0 \le i, j \le N$. Then, the eigenimage of sLoG F_i^{LoG} is defined as follows.

$$F_i^{LoG}(x,y) = \int_{s1}^{s2} s\frac{\partial g(x,y,s)}{\partial s}\varphi_i^{LoG}(s)\,ds$$



(a) Eigen solutions of Gaussian space

(b) Eigen solutions of sLoG space

Fig. 1 Left: Eigenimages and eigenfunctions of Gaussian scale-space. Right: Eigenimages and eigenfunctions of sLoG space.

$$= -\sum_{n=0}^{N} \frac{a_{i,n}^{LoG}}{2^{1/2}\pi r}\left(\frac{r}{2^{1/2}}\right)^n \times \left[-\Gamma\left(\frac{1-n}{2}, \frac{r^2}{2s_1^2}, \frac{r^2}{2s_2^2}\right) + \Gamma\left(\frac{3-n}{2}, \frac{r^2}{2s_1^2}, \frac{r^2}{2s_2^2}\right)\right]. \quad (20)$$

Here, $a_{i,n}^{LoG}$ is the coefficient of polynomial of eigensolution $\varphi_i^{LoG}(s)$ obtained by solving the generalized eigenvalue problem.

3. Numerical Examples

In this section, we show numerical examples of eigensolutions of Eq. (7) and show the linear generation of Gaussian and sLoG image.

3.1 Gaussian Scale-Space

In order to approximate the eigenfunction of Eq. (9), we use second or third-order polynomials ($N = 2$ or $N = 3$) and set the integral range of the scale parameter s to $s_1 = 1.0$, $s_2 = 5.0$. Based on this, we solve the 3×3 or 4×4 matrix generalized eigenvalue problem of Eq. (10). The solutions $a_{i,j}$ and eigenvalues λ_i ($0 \le i \le N$) are shown in Tables 1 and 2.

From the tables, it can be seen that $\lambda_2 \approx 0.0007$ is only 1 [%] of $\lambda_0 = 0.070$. From this rapid decrease, it is apparent that the original Gaussian function can be approximated by using a low-order of series expansion. The eigenimages for $N = 2$ are shown in the left of Fig. 1. The upper-part of the figure shows the eigenimages on the xy-plane, while the middle-part of the figure shows a graph of the eigenimages on $r = \sqrt{x^2 + y^2}$. As they depend only on r, these eigenimages are isotropic functions. The lower-part of the figure shows the eigenfunctions. The first-order's eigenimage looks like Gaussian and the second and third-order's eigenimage looks like Laplacian, however it is slight difference.

Table 1 Eigen solutions $\vec{a}$ of Gaussian scale-space at $N = 2$

i	$a_{i,0}$	$a_{i,1}$	$a_{i,2}$	λ_i
0	−1.51664	0.63295	−0.07352	0.07028
1	−1.98457	1.51593	−0.21841	0.01003
2	1.41248	−1.44794	0.29090	0.00077

Table 2 Eigen solutions $\vec{a}$ of Gaussian scale-space at $N = 3$

i	$a_{i,0}$	$a_{i,1}$	$a_{i,2}$	$a_{i,3}$	λ_i
0	−1.96331	1.47595	−0.40397	0.03729	0.07055
1	−2.52465	3.31488	−1.09824	0.11097	0.01088
2	2.20392	−3.77154	1.58800	−0.18386	0.00140
3	1.04991	−2.15040	1.14305	−0.16560	0.00008

Table 3 Eigen solutions $\vec{a}$ of sLog space at $N = 2$

i	$a_{i,0}^{LoG}$	$a_{i,1}^{LoG}$	$a_{i,2}^{LoG}$	λ_i^{LoG}
0	−1.66680	0.66306	−0.07074	0.09065
1	−2.45391	1.77823	−0.25326	0.02621
2	1.86269	−1.70701	0.32655	0.00354

Table 4 Eigen solutions $\vec{a}$ of sLog space at $N = 3$

i	$a_{i,0}^{LoG}$	$a_{i,1}^{LoG}$	$a_{i,2}^{LoG}$	$a_{i,3}^{LoG}$	λ_i^{LoG}
0	−1.78134	0.80365	−0.12157	0.00560	0.09067
1	−4.48103	4.32614	−1.19007	0.10394	0.02773
2	6.27885	−7.62290	2.65264	−0.27408	0.00624
3	4.07331	−5.69794	2.35606	−0.29145	0.00054

3.2 sLoG Space

Tables 3 and 4 show an example of solution (coefficient of polynomial and eigenvalue) of sLoG for $N = 2$ and $N = 3$, $s_1 = 1.0$, $s_2 = 5.0$. Then, the right-up part of Fig. 1 shows the eigenimages and eigenfunctions of sLoG.

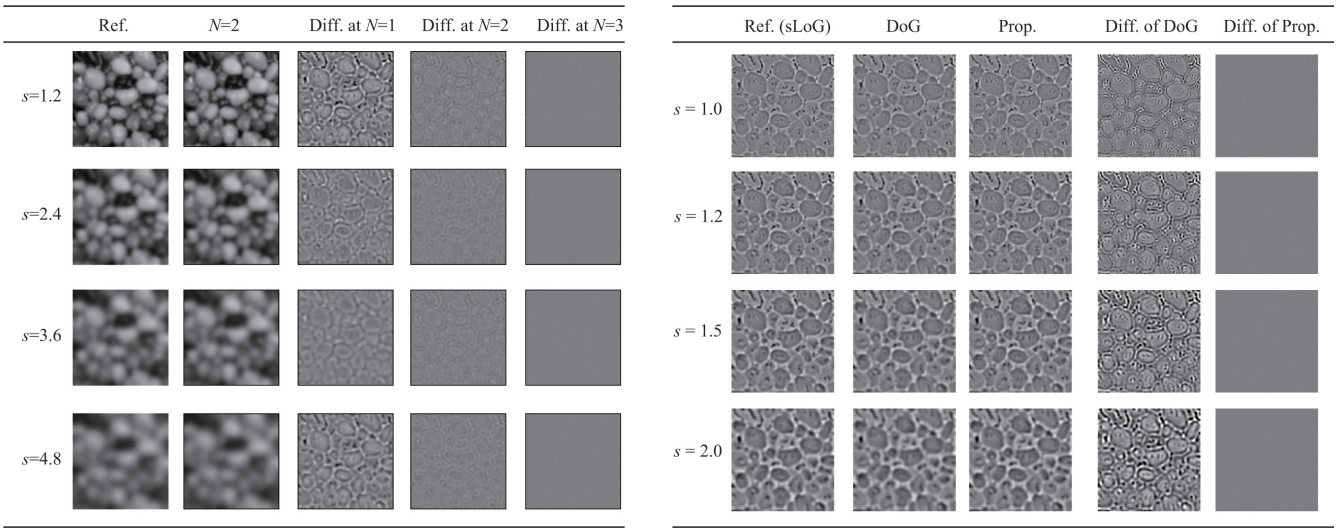


Fig. 3 Left: Generated Gausian blurred images for some scales. Right: Generated sLoG images for some scales.

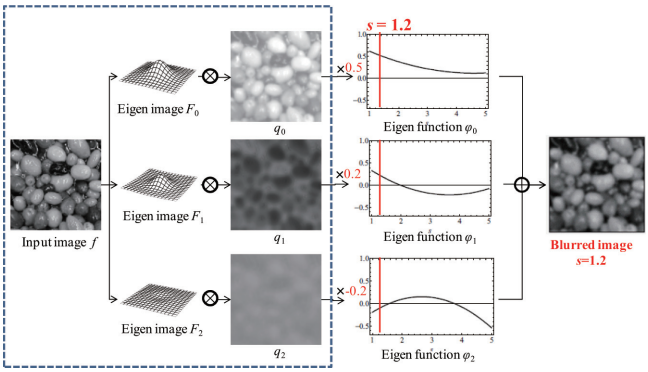


Fig. 2 Flowchart of Gaussian blurred image generation. Gaussian blurred image with arbitrary scale cen be obtained by simple linear combinations of q_i.

3.3 Linear Generation of Gaussian Image

In this section, we introduce a method of Gaussian blur image generation with an arbitrary scale as an application of scale-space compression.

A Gaussian blur image of scale s can be defined as:

$$I(x', y', s) = \sum_{i,j=0}^{N} q_i(x', y') s^j a_{i,j}. \tag{21}$$

Here, $q_i \equiv f * F_i$. This equation means that Gaussian blur image with a scale s can be obtained by a linear combination of q_i and $a_{i,j}$. The factors q_i can be obtained by convolving the eigenimage F_i into an input image f.

Figure 2 shows a flowchart detailing the steps of image generation at scale $s = 1.2$. The blue window on the left shows the step in which q_i is calculated; this indicates that a Gaussian blur image with an arbitrary scale s can be obtained immediately by linear combination once q_i is calculated.

In order to evaluate the proposed method we compared generated blur images in the range $1 \le s \le 5$ with references generated by convolving the Gaussian kernel $g(x, y, s)$.

The left of Fig. 3 shows the generated images for 128×128 Fruit image. The figure shows references, the blur images generated by proposed method for $N = 2$ and $N = 3$, and the difference images between the generated images and references for $s = 1.2, 2.4, 3.6$, and 4.8. It can be seen that the results for $N = 2$ and $N = 3$ show few errors.

The left of Fig. 4 shows the PSNR between the generated and reference images at scales ranging from $N = 1, 2$ and 3, $s = 1.0$ to $s = 5.0$ for Lenna and Fruit. Then, the graph shows the PSNR of Scalable Filter (5-kernels used) [32]. The average of PSNR error of our method for $N = 3$ is 68 [dB], and it can be seen from this that the proposed method can generate Gaussian blur images accurately by using simple linear operations.

3.4 Linear Generation of sLoG Image

In the case of $N = 3$, the sLoG images $I^{LoG}(x, y, s)$ can be obtained as follows:

$$\begin{aligned} &I^{LoG}(x', y', s) \\ &= \sum_{i=0}^{3} q_i^{LoG} \left(a_{i,0}^{LoG} + s a_{i,1}^{LoG} + s^2 a_{i,2}^{LoG} + s^3 a_{i,3}^{LoG} \right). \end{aligned} \tag{22}$$

Here, $q_i^{LoG} \equiv f * F_i^{LoG}$. The right of Fig. 3 shows the sLoG images generated by proposed method, and conventional (scale-normalized) DoG images. Here, DoG image is obtained as follows:

$$\mathrm{DoG}(x, y) \equiv \frac{g(x, y, k\sigma) - g(x, y, \sigma)}{1 - k}. \tag{23}$$

In this case, $k = 1.2$ is used. For better visibility, the pixel value of those figures are enhanced. The right two

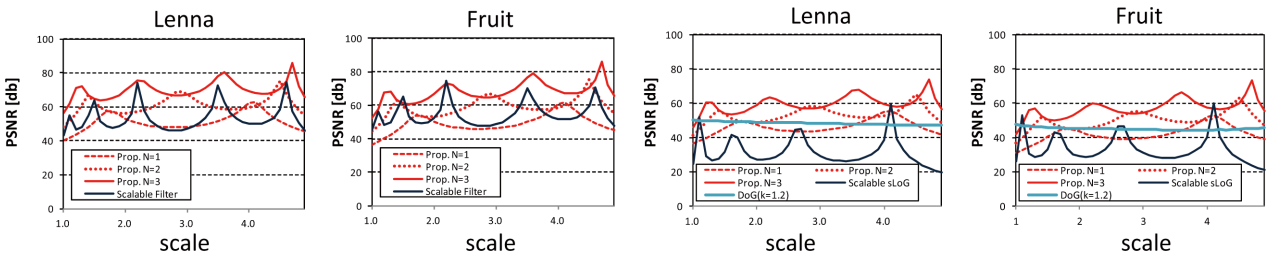


Fig. 4 PSNR evaluation of Gausian blurred image and sLoG image.

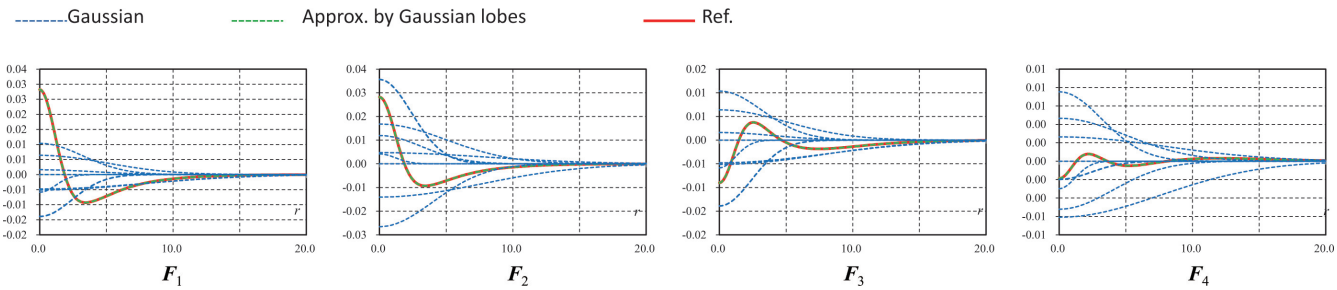


Fig. 5 The decomposition filters F_i can be approximated well by the linear combination of Gaussian lobes with different sigma.

columns of the figure shows the difference of a reference $s\nabla^2 g * f$ (Ref. of the figure) and the generated image. The figure shows that DoG image have more errors because of a backward difference approximation Eq. (23). On the other hand, the proposed method can approximate the sLoG images with various scale.

The right of Fig. 4 shows a numerical accuracy of the above approximation for Lenna and Fruit. Scalable sLoG filter is implemented by the same analogy of scalable filter and 5-kernes is used. The average of PSNR error of our method for $N = 3$ is 56 [dB], The proposed method can approximate the sLoG accurately with arbitrary scale by linear combination of only four images q_i^{LoG}.

4. Gaussian Lobes Approximation

The decomposition filter F_i of sLoG using polynomials is not an XY-separable filter. Previously, to convolute such filters, a 2D-FFT operation was required [16], which requires significant computational time. In order to accelerate filtering, we approximate the decomposition filter by a linear combination of some Gaussian kernels with different scale (referred to as Gaussian lobes approximation) as follows:

$$F_i(x, y) \approx \sum_j^T w_{ij} g(x, y, t_j). \tag{24}$$

Here, w_{ij} are weight values and they can be computed by the least squared fitting to an original basis functions of $F_i(r)$. Using the above equation, the approximated decomposition filter is written as follows:

$$\hat{g}(x, y, s) \approx \sum_i^L \left(\sum_j^T w_{ij} g(x, y, t_j) \right) \varphi_i(s) \tag{25}$$

$$= \sum_j^T g(x, y, t_j) \varphi'_j(s). \tag{26}$$

Here, φ'_i is the transformed eigen function defined as follows:

$$\varphi'_i = \sum_i^L w_{ij} \varphi'_i. \tag{27}$$

Thus, decomposition filtering can be achieved by some number of Gaussian filtering. Such Gaussian filtering is XY-separable, and there are extensive, efficient, and accurate filtering algorithms, such as a cascade algorithm or recursive filtering [33]–[36]. Such algorithms are used in spatial domains and do not require FFT operations.

Figure 5 shows the decomposition filters and approximated filters by Gaussian lobes for a Gaussian kernel. Here, the scale range is $s = [1.0, 8.0]$, the number of Gaussian lobes is $T = 8$, and their sigma parameters are $\mathbf{t} = \{1.0, 1.2, 2.4, 3.2, 4.0, 5.0, 7.6, 8.0\}$. As is shown in the figure, Gaussian lobes can approximate the decomposition filters accurately.

5. Application: Spectral SIFT

We propose a new SIFT detector (Spectral SIFT) using the sLoG space compression. The SIFT keypoints are detected by finding the local extremum of sLoG. It is enough to find the zero position of the partial differential of sLoG. In the proposed model, because sLoG image is represented by the polynomial of s, it is easy to find the exact local extremum

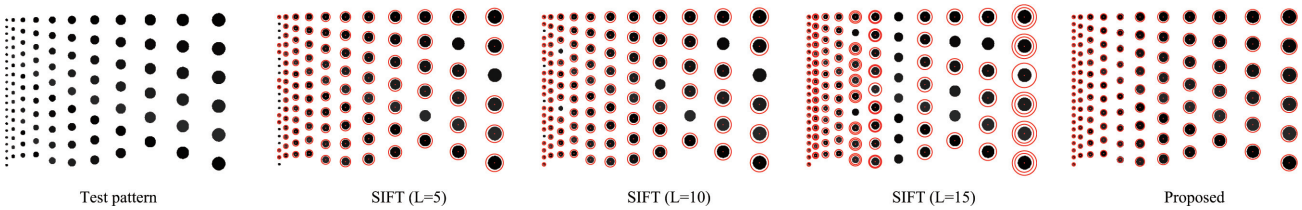
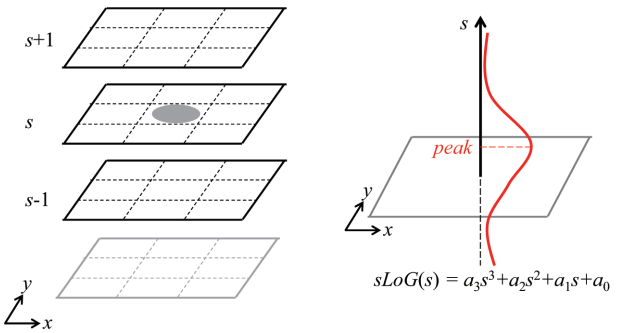


Fig. 7 Results for simple test pattern. Red circles are the detected keypoints and the radius means scale. Conventional SIFT cannot detect the circles correctly even an easy case is.



(a) 27 neighbor pixels extrema (b) Polynomial representation

Fig. 6 Scale detection. In conventional SIFT, scale-space is discretized. In our method, images of scale-space are represented by polynimials of scale parameter s.

of sLoG by solving the following quadric equation.

$$\begin{aligned}&\partial I^{LoG}(x', y', s)/\partial s \\ &= \sum_{i=0}^{3} q_i^{LoG}\left(a_{i,1}^{LoG} + 2sa_{i,2}^{LoG} + 3s^2 a_{i,3}^{LoG}\right) \\ &\equiv as^2 + bs + c = 0.\end{aligned} \tag{28}$$

Then, the optimal scales can be detect at $s = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$. The keypoint with $2as + b > 0$ is a bright keypoint, and $2as + b < 0$ is dark keypoint. After detecting the scale, then 27-neighbor pixels of sLoG are checked for the XY-scale extremum.

The conventional SIFT requires the 27-neightbor pixel check for all scale layer (Fig. 6 (a)). It is time consuming step of SIFT, especially if increasing the number of scale layer L. On the other hand, the proposed method can detect the optimal scale by simple algebraic operation (Fig. 6 (b)). It is fast and accurate because it does not include the discretization error of scale layer and the interpolation artifact.

5.1 Simple Pattern Testing

We evaluate our method for simple test pattern in Fig. 7. The 640×480 input image has black circle patterns with from small to large radius (about 2~15[pix]). Figure shows the results of three conventional SIFTs and the proposed method. The conventional SIFT has different number of scale-layer $L = 5, 10$ and 15. Generally, $L = 5$ is used for one octave. The same detection parameters such as threshold were used. The SIFT at $L = 5$ could not detect the small radius circles in the left of the image and could not detect some large radius circles. In the case of increasing the number of scale layer L, the scale-resolution was improved and more number of the small circles were detected. However, more number of the large circles were missed by the scale discretization artifact. On the other hand, the proposed method can detect all circles correctly.

5.2 Evaluation of Detector Repeatablity

We evaluated the repeatability of the proposed detector with an Oxford dataset[†]. In this evaluation, we used six scenes, i.e., two viewpoint change scene (Wall and Graffiti), one zoom and rotation (Boat) scene, a lighting change (Leuven) a scene, a blurring change (Trees) scene, and a JPEG compression rate change (UBC) scene. Each scene includes six different images, and the homography H, which relates to the viewing of two images, is given.

The repeatability score was computed by Mikolajczyk's method [37]. This score indicates whether the detector localizes the keypoints of the same region in two different images. The region of interest of a keypoint is defined as a circle with a radius $3s$ (s is the determined scale of the keypoint).

In the evaluation, after detecting keypoints for two images, each region of the keypoints in one image was transformed by the given homography H. At this time, the circular region of interest was transformed to an ellipsoid region and non-visible keypoints in the two images were removed from the evaluation. Then, the overlap error ϵ of two regions of interest was computed as $\epsilon = 1 - \frac{a \cap A^T b A}{a \cup A^T b A}$. Here, a and b are the two regions of interest, and A is the linearization of the homography H. In this evaluation, we counted the keypoint pairs with $\epsilon < 0.5$ as correspondings.

To compare the proposed detector, we used two DoG detectors from SIFT (SIFT ($L = 6$) and SIFT ($L = 10$), where L is the number of octave layer for DoG), the fast hessian detector from SURF, and Spectral SIFT (SSIFT without Gaussian lobes (GL); $N = 3$), which is the LoG detector with polynomial representation. Spectral SIFT using Gaussian lobes approximation (SSIFT with GL) represents the proposed method with $N = 3$.

The code for SIFT, SURF, and the computation of repeatability was developed using OpenCV 2.4.6. The implementation of SSIFT is also based on OpenCV. In SSIFT,

[†]http://www.robots.ox.ac.uk/~vgg/data/data-aff.html

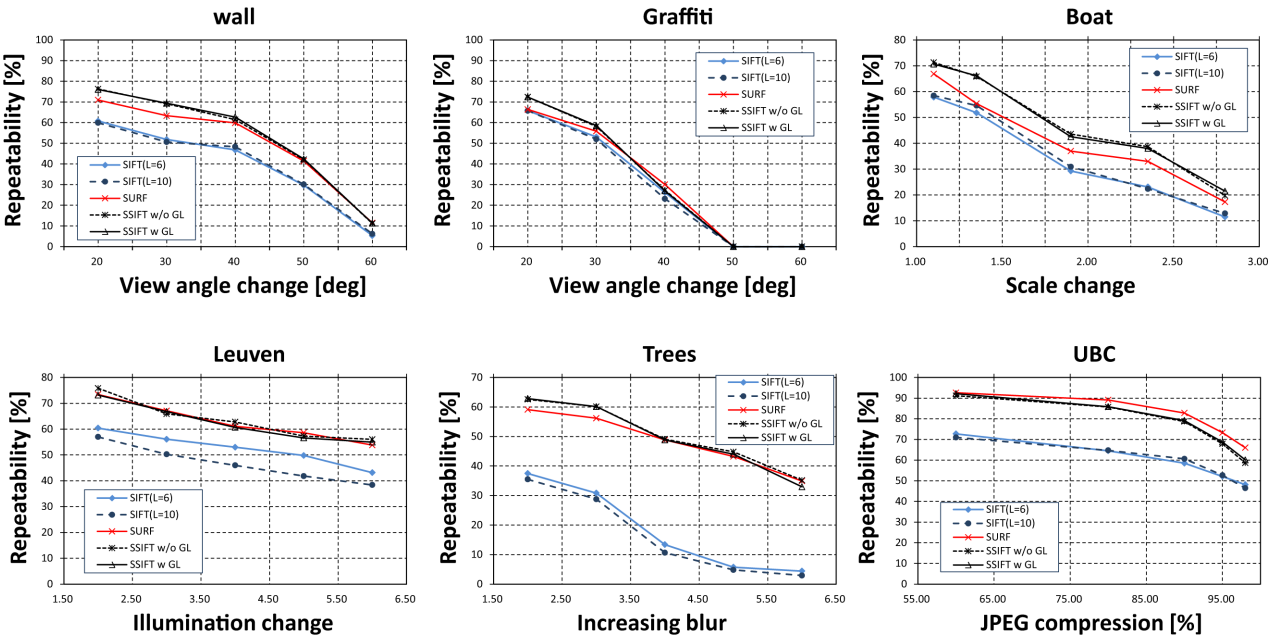


Fig. 8 Repeatability evaluation for 6 scnens of Oxford dataset.

the scale range per octave was $s = [1.6, 6.4]$ and $T = 8$. This scale range specification is wide compared to SIFT and SURF; therefore, SSIFT can reduce the time required for image pyramid processing.

To evaluate repeatability with the same number of keypoints, the detected keypoints were sorted according to keypoint response, such as absolute amplitude of the DoG, LoG, or Hessian response. The top-1000 keypoints were then selected for evaluation. For the computation of overlap of two regions, we must closely consider scale because larger ellipsoid regions demonstrate superior repeatability; thus, targeted over-estimation of scale improves repeatability. In this study, the scale of the SIFT and SSIFT detectors are defined as the local maximum of DoG and LoG, and the scale of SURF can be transformed as $s = 0.13 \times \text{MaskSize}$.

Figure 8 shows the repeatability of the six scenes. SSIFT demonstrated slightly better performance than SURF with the view angle change scenes (i.e., Wall and Graffiti), and in the scale change scene (i.e., Boat), SSIFT outperformed SURF. For other scenes (i.e., Leuven, Tree, and UBC), SSIFT performance was comparable to SURF. In all scenes, it was observed that SSIFT outperformed standard SIFT. Note that SSIFT and SSIFT with GL demonstrated approximately equal repeatability. This indicates that Gaussian lobes can approximate the decomposition filter of polynomials of sLoG.

It should be noted that increasing the number of octave layers reduces repeatability; this has been established previously [38]. SSIFT can detect a continuous optimal scale without discretization of the scale layer, similar to SIFT and SURF; thus, results obtained with the scale change scene (i.e., Boat) indicate good repeatability.

Table 5 Computational time for Boat-6 850×680 [pix]

	#keypoints	total time[msec]
SIFT (L = 6)	2260	228
SIFT (L = 10)	2260	319
SURF	2329	71
SSIFT	2369	217
SSIFT with GL	2355	39

Table 6 Computational time for Trees-1 1000×700 [pix]

	#keypoints	total time [msec]
SIFT (L = 6)	5570	297
SIFT (L = 10)	5537	413
SURF	5573	128
SSIFT GL	5539	141
SSIFT with GL	5719	65

5.3 Computational Time

We also measured computational time for each detector. Table 5 and Table 6 show the total computational times and number of detected keypoints. To measure computational time, we used an Intel Core-i7 4770K 3.4 GHz, with 8 GB memory running 64-bit Windows 7. Note that SURF was implemented professionally by Intel TBB as multiprocessor programming, and SSIFT was implemented as multiprocessor programming using OpenMP. The edge threshold for each method was adjusted manually to equal the number of detected keypoints for each method. Conventional SIFT required approximately 200~400 [msec], and increasing the number of octave layers slowed performance. This indicates that detection of the local min/max by 26-neighbor pixel search results in a bottleneck relative to computational time. SSIFT does not require a 26-neighbor search ; thus, its

computational times can be accelerated. SSIFT without GL was observed to be comparable to SURF, and SSIFT with GL performed twice as fast as SURF.

6. Conclusions

In this paper, we proposed a method for applying PCA to scale-spaces. PCA is the standard method used in computer vision for tasks. However, in order to apply the method to scale-spaces it is necessary to extend conventional square matrix-based finite PCA to an infinite number of dimensions. To resolve this infinite eigenvalue problem, we used spectral theory to develop integral equations for which approximate solutions could be developed using polynomial equations. We applied spectral decomposition to some scale-spaces and clarified its eigensolutions. Furthermore, to make the scale-space filtering efficient, we approximated the basis filter set using Gaussian lobes approximation and XY-Separable filters were obtained.

As an application of this proposed method, we introduced a method for generating Gaussian blur images and sLoG images of arbitrary scale that can be calculated through simple linear combination. As a more practical example, we proposed spectral SIFT detector using the spectral theory. The experimental results showed that the proposed detector outperformed the previous SIFT and SURF in repeatablity and computational cost. Because the scale-space processing is a basic technique, our method can apply to many existing scale-space processing.

References

[1] D. Marr and E. Hildreth, "Theory of edge detection," Proceedings of the Royal Society of London, vol.207, no.1167, pp.187–217, 1980.
[2] A.P. Witkin, "Scale-space filtering," International Joint Conference on Artificial Intelligence, pp.1019–1022, 1983.
[3] S. Ranganath, "Image filtering using multiresolution representations," IEEE Trans. Pattern Anal. Mach. Intell., vol.13, no.5, pp.426–440, 1991.
[4] A.L. Yuille and T.A. Poggio, "Scaling theorems for zero crossings," IEEE Trans. Pattern Anal. Mach. Intell., vol.PAMI-8, no.1, pp.15–25, 1986.
[5] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," International Journal of Computer Vision, vol.30, no.2, pp.117–156, 1998.
[6] P. Burt and E. Adelson, "The Laplacian Pyramid as a Compact Image Code," IEEE Trans. Pattern Anal. Mach. Intell., vol.31, no.4, pp.532–540, 1983.
[7] 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.
[8] T. Lindeberg, "Feature detection with automatic scale selection," International Journal of Computer Vision, vol.30, no.2, pp.79–116, 1998.
[9] P. Perona, "Deformable kernels for early vision," The IEEE International Conference on Computer Vision and Pattern Recognition, pp.222–227, 1991.
[10] E. Simoncelli, W. Freeman, E. Adelson, and D. Heeger, "Shiftable multi-scale transforms," IEEE Trans. Inf. Theory, vol.38, no.2, pp.587–607, 1992.
[11] A.A. Bharath, "Steerable filters from erlang functions," British Machine Vision Conference, pp.144–153, 1998.
[12] W.T. Freeman and E.H. Adelson, "The design and use of steerable filters," IEEE Trans. Pattern Anal. Mach. Intell., vol.13, no.9, pp.891–906, 1991.
[13] P. Perona, "Steerable-scalable kernels for edge detection and junction analysis," The IEEE European Conference on Computer Vision, vol.588, pp.3–18, 1992.
[14] G. Koutaki and K. Uchimura, "Scale-space compression and its application using spectral theory," The IEEE International Conference on Image Processing, pp.820–823, 2013.
[15] G. Koutaki and K. Uchimura, "Applications to pattern matching using spectral theory and its performance evaluation," IEICE Trans. Inf. & Syst. (Japanese Edition), vol.J96-D, no.8, pp.1664–1674, 2013.
[16] G. Koutaki and K. Uchimura, "Scale-space processing using polynomial representations," The IEEE International Conference on Computer Vision and Pattern Recognition, pp.2744–2751, 2014.
[17] T. Hasegawa, M. Ambai, K. Ishikawa, G. Koutaki, Y. Yamauchi, T. Yamashita, and H. Fujiyoshi, "Multiple-hypothesis affine region estimation with anisotropic log filters," The IEEE International Conference on Computer Vision (ICCV), pp.585–593, Dec. 2015.
[18] D.G. Lowe, "Object recognition from local scale-invariant features," The IEEE International Conference on Computer Vision, vol.2, pp.1150–1157, 1999.
[19] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors," International Journal of Computer Vision, vol.60, pp.63–86, 2004.
[20] Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," The IEEE International Conference on Computer Vision and Pattern Recognition, pp.506–513, 2004.
[21] J.-M. Morel and G. Yu, "ASIFT: A new framework for fully affine invariant image comparison," SIAM J. Img. Sci., vol.2, no.2, pp.438–469, April 2009.
[22] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," CVIU, vol.110, no.3, pp.346–359, 2008.
[23] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," IEEE Trans. Pattern Anal. Mach. Intell., vol.32, no.1, pp.105–119, 2010.
[24] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: an efficient alternative to SIFT or SURF," The IEEE International Conference on Computer Vision, pp.2564–2571, 2011.
[25] P.F. Alcantarilla, A. Bartoli, and A.J. Davison, "Kaze features," The IEEE European Conference on Computer Vision, vol.7577, pp.214–227, 2012.
[26] P. Mainali, G. Lafruit, Q. Yang, B. Geelen, L.V. Gool, and R. Lauwereins, "Sifer: Scale-invariant feature detector with error resilience," International Journal of Computer Vision, vol.104, no.2, pp.172–197, 2013.
[27] P. Mainali, G. Lafruit, K. Tack, L. Van Gool, and R. Lauwereins, "Derivative-based scale invariant image feature detector with error resilience," IEEE Trans. Image Process., vol.23, no.5, pp.2380–2391, 2014.
[28] S. Omachi and M. Omachi, "Fast template matching with polynomials," IEEE Trans. Image Process., vol.16, no.8, pp.2139–2149, 2007.
[29] R. van den Boomgaard and J. van De Weijer, "Least squares and robust estimation of local image structure," Proceedings of Scale-Space, Berlin Heidelberg, vol.2695, pp.237–254, 2003.
[30] L. Florack, B.T.H. Romeny, M. Viergever, and J. Koenderink, "The gaussian scale-space paradigm and the multiscale local jet," International Journal of Computer Vision, vol.18, pp.61–75, 1996.
[31] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery, Numerical Recipes in C, 2nd ed., Cambridge University Press, 1992.
[32] D. Shy and P. Perona, "X-y separable pyramid steerable scalable filters," IEEE International Conference on Computer Vision and Pattern Recognition, pp.237–244, 1994.
[33] 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, 1986.
[34] E. Elboher and M. Werman, "Cosine integral images for fast spatial and range filtering," The IEEE International Conference on Image Processing, pp.89–92, 2011.
[35] K. Sugimoto and S. Kamata, "Fast image filtering by DCT-based kernel decomposition and sequential sum update," The IEEE International Conference on Image Processing, pp.125–128, 2012.
[36] K. Sugimoto and S. Kamata, "Fast gaussian filter with second-order shift property of DCT-V," The IEEE International Conference on Image Processing, pp.514–518, 2013.
[37] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," IEEE Trans. Pattern Anal. Mach. Intell., vol.27, no.10, pp.1615–1630, 2005.
[38] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, vol.60, no.2, pp.91–110, 2004.

Gou Koutaki received his M.E. and Ph.D. degrees from Kumamoto University, Japan, in 2004 and 2007, respectively. He joined the Central Research Laboratory, Hitachi, Ltd. in 2007. He is presently an Assistant Professor of Kumamoto University, Japan. His research interests are image processing and machine vision. He is also a member of the IEEE and ACM.

Keiichi Uchimura received his M.S degree from Kumamoto University in 1977 and Ph.D. degree from Tohoku University in 1987. He became a Research Associate at Kumamoto National College of Technology in 1977. He moved to Kumamoto University in 1980 and is now a Professor. He was a Visiting Professor at McMaster University (Canada) in 2010. His research interests are image processing and recognition, advanced road traffic systems, optimization, and other fields. He is a member of the IPSJ.