

# Approximated Scale Space for Efficient and Accurate SIFT Key-Point Detection



Ying Wang, Yiguang Liu, Zhenyu Xu, Yunan Zheng, and Weijie Hong

**Abstract** The SIFT (scale invariant feature transform) key-point serves as an indispensable role in many computer vision applications. This paper presents an approximation of the SIFT scale space for key-point detection with high efficiency while preserving the accuracy. We build the scale space by repeated averaging filters to approximate the Gaussian filters used in SIFT algorithm. The accuracy of the proposed method is guaranteed by that an image undergoes repeated smoothing with an averaging filter is approximately equivalent to the smoothing with a specified Gaussian filter, which can be proved by the *center limit theorem*. The efficiency is improved by using integral image to fast compute the averaging filtering. In addition, we also present a method to filter out unstable key-points on the edges. Experimental results demonstrate the proposed method can generate high repeatable key-points quite close to the SIFT with only about one tenth of computational complexity of SIFT, and concurrently the proposed method does outperform many other methods.

**Keywords** SIFT · Key-point detector · Approximated Gaussian · Repeated averaging filters

## 1 Introduction

The SIFT [1] key-point plays an important role in computer vision and pattern recognition tasks such as structure from motion, object matching, object recognition, and texture analysis [2–6]. Though many other key-points have been proposed [7–11], the SIFT key-point is still favored by many applications especially for those with high accuracy needed [12]. However, the SIFT key-point has also been criticized for the drawback of its heavy computational burden, thus many variational

---

Y. Wang · Y. Liu (✉) · Z. Xu · Y. Zheng · W. Hong  
College of Computer Science, Sichuan University, Chengdu, China  
e-mail: [liuyg@scu.edu.cn](mailto:liuyg@scu.edu.cn)

methods have been proposed to accelerate the computational speed, but at the cost of insufficient accuracy or degraded repeatability of the key-points. To efficiently compute the SIFT key-point while preserving the accuracy and repeatability is the goal of this paper. The SIFT key-point is obtained by the DOG blob detector which is an approximation of LOG detector. There are also some other blob detectors such as [13–15], but their performance is inferior to SIFT in terms of accuracy and repeatability, as the *scale space theory*[16, 17] proved that the Gaussian kernel is the only smoothing kernel that could be used to create the image scale space. Based on the center limit theorem, repeated averaging filtering can be used to approximate Gaussian filtering. We propose a bank of averaging filters that accurately approximate the Gaussian filter in the SIFT algorithm, and the averaging filters are implemented via integral images. Because integral image is computed in our method, a combined method for eliminating key-points on the edge is also presented by reusing the integral images, which is more efficient than the original one in SIFT algorithm.

## 2 A Brief Review of SIFT Scale Space Construction and Key-Point Detection

The SIFT scale space consists of two pyramids, the Gaussian pyramid and the DOG(difference of Gaussian) pyramid. The Gaussian pyramid consists of  $M$  octaves and each octave consists of  $N$  layers. Suppose  $L(a, b)$  denotes the  $a$ th layer in octave  $b$ , it is produced by the convolution of a Gaussian function with the input image  $I(x, y)$

$$L(a, b) = G(x, y, \sigma) * I(x, y) \quad (1)$$

where  $*$  stands for the convolution operation and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (2)$$

where  $\sigma$  is determined by  $a, b$  and the scale of the first layer  $\sigma_0$

$$\sigma = 2^{a+\frac{b}{N-3}} \cdot \sigma_0 \quad (3)$$

Since the computation complexity of Eq.(1) is proportion to  $\sigma^2$ , in practical implementation, smaller  $\sigma$  can be used due to the property of the Gaussian convolution that the convolution of two Gaussian functions is also a Gaussian with variance being the sum of the original variances. That is,

$$I(x, y) * G(x, y, \sigma_1) * G(x, y, \sigma_2) = I(x, y) * G(x, y, \sigma_3) \quad (4)$$

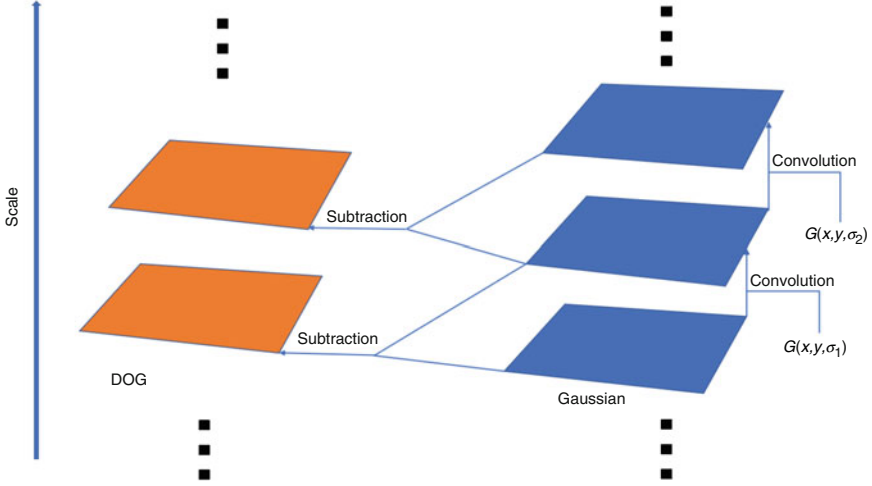


Fig. 1 The process to construct the SIFT scale space

where  $\sigma_1^2 + \sigma_2^2 = \sigma_3^2$ . So, if  $b \neq 0$ ,  $L(a, b)$  is computed as

$$L(a, b) = G(x, y, \sigma_{\text{gap}}) * L(a, b - 1) \tag{5}$$

where

$$\sigma_{\text{gap}} = \sqrt{\sigma(0, b)^2 - \sigma(0, b - 1)^2} \tag{6}$$

It can be seen from Eq. (6) that each octave shares the same group of  $\sigma$  to generate a successive layer, this is because when  $b = 0$  and  $a \neq 0$ , the  $L(a, b)$  is obtained by down sampling the layer  $L(a - 1, b + N - 2)$ . Once the Gaussian pyramid is computed, the difference of Gaussian pyramid is obtained via the subtraction of two successive layers in the Gaussian pyramid. Figure 1 shows the SIFT scale space construction processes. The SIFT key-point is detected by finding the extrema in the DOG pyramid, after that interpolation is performed to get sub-pixel precision if the key-point is not on the edge.

### 3 Approximation of Gaussian Smoothing Using Repeated Averaging Filter

In this section, we introduce the quantitative relationship between repeated averaging filtering and Gaussian filtering for a given image. The coefficients of an averaging filter can be viewed as the probability density of a random variable  $X$

which is uniformly distributed in a square area centered at the origin. Given an image  $I(x, y)$  and an averaging filter  $A(x, y, w)$ , where  $w$  is the window width of the averaging filter, repeated averaging filtering of the image can be mathematically expressed as:

$$I(x, y, n) = I(x, y) * \underbrace{A(x, y, w) * A(x, y, w) \dots A(x, y, w)}_n \quad (7)$$

Based on the *center limit theorem*, when  $n$  approaches to infinity,

$$\begin{aligned} \lim_{n \rightarrow \infty} I(x, y, n) &= I(x, y) * \lim_{n \rightarrow \infty} \underbrace{A(x, y, w) * A(x, y, w) \dots A(x, y, w)}_n \\ &= I(x, y) * G(x, y, \sigma_{\text{gau}}) \end{aligned} \quad (8)$$

where  $G(x, y, \sigma_{\text{gau}})$  is the Gaussian function with the variance  $\sigma_{\text{gau}}^2$ . Suppose the window width of an averaging filter is  $w$ , the variance of its corresponding discrete random variable is

$$\sigma_{\text{av}}^2 = \frac{w^2 - 1}{12} \quad (9)$$

the relation of  $\sigma_{\text{av}}$ ,  $\sigma_{\text{gau}}$ ,  $w$  and  $n$  can be deduced:

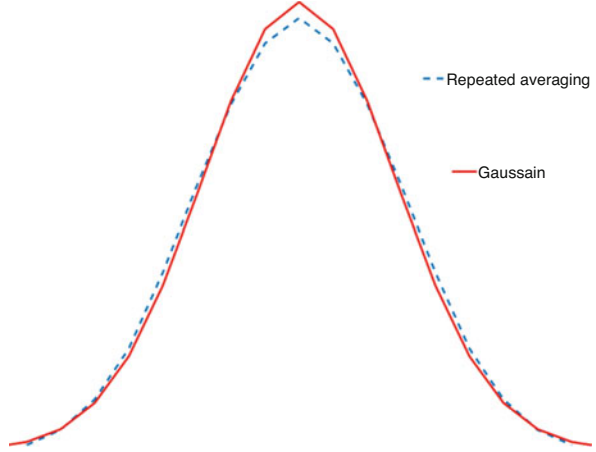
$$\sigma_{\text{gau}} = \sqrt{n\sigma_{\text{av}}^2} = \sqrt{\frac{nw^2 - n}{12}} \quad (10)$$

For discrete averaging filter and discrete Gaussian filter, the two sides of the first equal sign of Eq. (10) are very close to each other when  $n \not\ll 3$ , which indicates no less than 3 times repeated averaging filtering can well approximate a specified Gaussian filter. Figure 2 shows the two curves are close to each other.

## 4 Scale Space Construction via Repeated Averaging Filtering

To create the scale space introduced in Sect. 1, the key problem is to seek the initial scale  $\sigma_0$ , a group of Gaussian filters used to construct next layer from current layer and a group of averaging filters to approximate these Gaussian filters. Given a Gaussian kernel, it is not hard to find an optimal averaging filter  $A(x, y, w)$  and the times  $n$  needed to approximate the Gaussian kernel based on Eq. (10). However, there are more difficulties to get the optimal averaging filter and filtering times in real applications. Here we mainly focus on finding optimal averaging filters to construct the scale space introduced in Sect. 1.

**Fig. 2** Two similar curves: the red one is the Gaussian kernel with  $\sigma = 2\sqrt{2}$ , the blue one is its approximation resulted by 4 times convolution of an averaging filter with the window width  $w = 5$



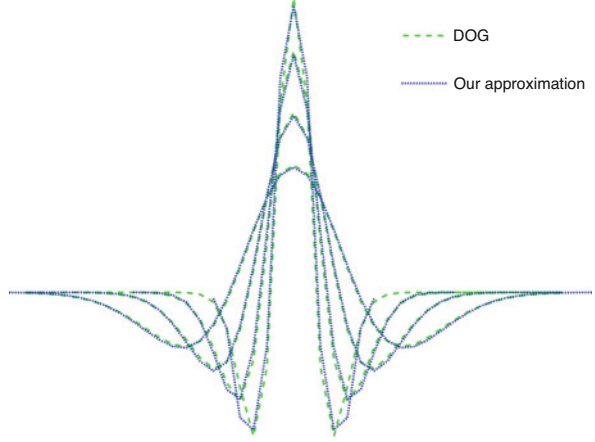
The constraint of Eq. (10) is that  $n$  is a positive integer not smaller than 3 and not bigger than 10, because if  $n < 3$  it is not enough to approximate the Gaussian kernel and if  $n > 10$  the computational complexity will have no advantage over the original Gaussian filtering. The variable  $w$  should be a positive odd number not smaller than 3, in order to satisfy that there could always be a center pixel the filtering result can be assigned to. Suppose the initial scale of the Gaussian pyramid is  $\sigma_0$ , as most key-points lie in the first several layer of the scale space, thus  $\sigma_0$  should not be large. According to the original SIFT algorithm where  $\sigma_0 = 1.6$ , here we set  $\sigma_0 < 2.0$ . The number of layers  $N$  in each octave depends on the sampling frequency  $F$ , and  $F = N - 3$ . In [1] the sampling frequency is set to 2, 3, or 4, which gains an acceptable balance between key-point repeatability and computational time. Constraints to  $\sigma_i$  under SIFT scale space is that  $\sigma_i = \sigma_0 \sqrt{2^{\frac{2(i+1)}{F}} - 2^{\frac{2i}{F}}}$ . Besides,  $\sigma_i = \sqrt{\frac{n_i w_i^2 - n_i}{12}}$  is needed for discrete averaging filters and discrete Gaussian filters. These constraints can be formulated as:

$$\sigma_i = \sigma_0 \sqrt{2^{\frac{2(i+1)}{F}} - 2^{\frac{2i}{F}}} = \sqrt{\frac{n_i w_i^2 - n_i}{12}} \quad (11)$$

where  $i = 1, 2, 3, \dots, F + 2$ ,  $\sigma_0 < 2$ ,  $n_i \in \{3, 4, 5, \dots, 10\}$ ,  $w_i \in \{3, 5, 7, \dots\}$ . It can be verified that the only solution to Eq. (11) is

$$\begin{cases} \sigma_0 = \sqrt{2}, \sigma_1 = 2, \sigma_2 = 2\sqrt{2}, \sigma_3 = 4, \sigma_4 = 4\sqrt{2} \\ n_1 = 3, n_2 = 6, n_3 = 4, n_4 = 4 \\ w_1 = 3, w_2 = 3, w_3 = 5, w_4 = 7 \\ F = 2 \end{cases} \quad (12)$$

**Fig. 3** Approximated DOG detector for SIFT key-point detection



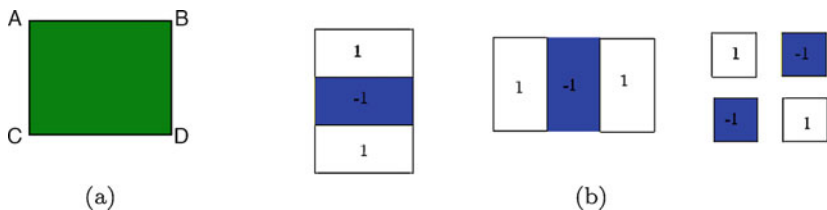
The approximated SIFT scale space is constructed in the same procedure as it is in the original SIFT algorithm, the Gaussian filters are replaced by averaging filters, and their quantitative relation is given in Eq. (12). Note that the key-point detector in SIFT is actually the DOG detector obtained from subtraction of two Gaussian kernels, Fig. 3 shows the approximated DOG and DOG curves in one octave of the DOG pyramid.

## 5 Key-Point Detection

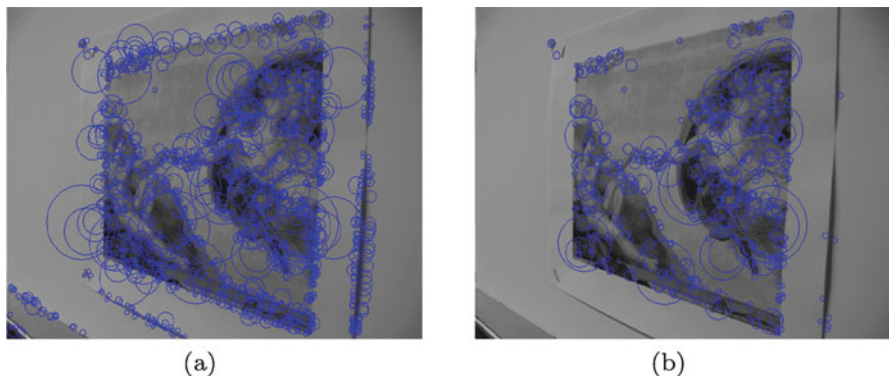
The key-points are detected by searching the local extrema in a  $3 \times 3$  region of the approximated DOG pyramid. However, the DOG detector is sensitive to image edges, and the key-point on the edge should be removed since it is unstable. As integral image is computed in our method (depicted in Fig. 4a), we propose to use fast Hessian measure to reject key-points on the edge. The principal curvature is used to determine whether a point is on the edge, which is computed by the eigenvalues of scale adapted Hessian matrix:

$$H(X, \sigma) = \begin{bmatrix} L_{xx}(X, \sigma), & L_{xy}(X, \sigma) \\ L_{xy}(X, \sigma), & L_{yy}(X, \sigma) \end{bmatrix}, \quad (13)$$

The partial derivatives in Eq. (13) are computed by integral image and a group of box filters as illustrated in Fig. 4. The images of before and after filtering out the key-point on the edge are shown in Fig. 5. After filtering out the key-points on edge, the scale space quadratic interpolation is performed for every key-point to get a sub-pixel accurate location (Fig. 5).



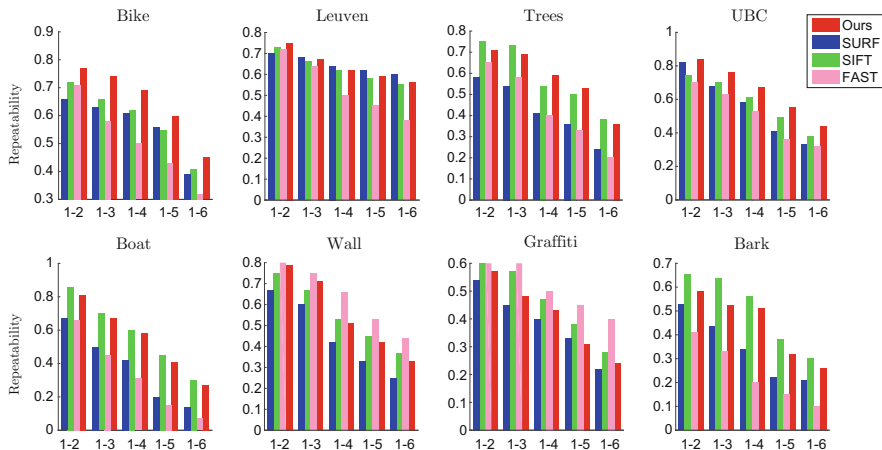
**Fig. 4** Using integral image and box filters to compute the partial derivatives of Hessian matrix. (a) The integral image:  $SUM = A + D - B - C$ . (b) Box filters to compute the partial derivatives of the Hessian matrix



**Fig. 5** Before-and-after filtering out the edge key-points. (a) Before filtering out the edge key-points. (b) After filtering out the edge key-points

## 6 Experimental Results

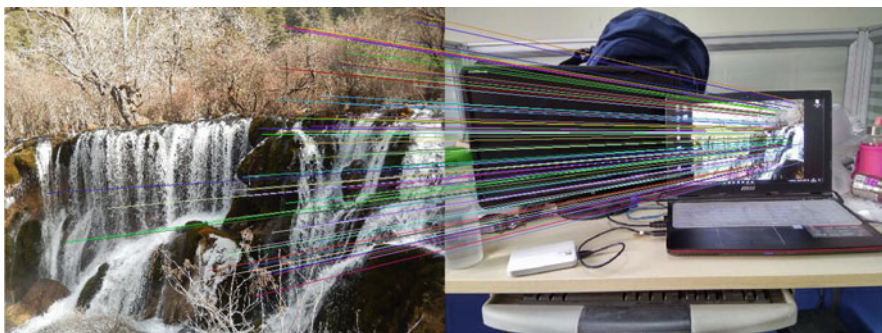
The accuracy of our method is validated by the performance comparison of the approximated SIFT detector, SIFT detector, SURF detector, and FAST detector. The SIFT code implemented by Rob Hess [18] is used in our experiment. For the SURF algorithm, we use the original implementation released by its authors. The FAST detector in our experiment is implemented in OpenCV 3.2.0. The datasets proposed by Mikolajczyk and Schmid [19] are used for the evaluation. There are totally 8 datasets, and each dataset has six images with increasing amount of deformation from a reference image. The deformations, covering zoom and rotation (Boat and Bark sequences), view-point change (Wall and Graffiti sequences), brightness changes (Leuven sequence), blur (Trees and Bikes sequence) as well as JPEG compression (UBC sequence), are provided with known ground truth which can be used to identify the correspondence. The *repeatability score* introduced in [19] is used to measure the reliability of the detectors on detecting the same feature point under different deformations of the same scene. The repeatability score is defined as the ratio between the number of corresponding features and the smaller number of features in one image in the image pair. To give a fair comparison, the thresholds of these detectors are adjusted to give approximately equal number of feature points



**Fig. 6** The repeatability of several key-point detectors

**Table 1** Timing results of SIFT, SURF, FAST and our method for key-point detection on the first image of the Wall sequence (size:  $1000 \times 700$  pixels)

Detector	SIFT	SURF	FAST	OURS
Time (ms)	332	97	11	35



**Fig. 7** Real-world image matching using the approximated SIFT detector with SURF descriptor

as SURF detector detected. The results in Fig. 6 show that the SIFT detector has the best performance in most cases, and the proposed method is close to SIFT and better than SURF and FAST detector.

The efficiency of the proposed method is also compared to SIFT, SURF, and FAST. The times of detection of the key-points in the first image of the wall sequence is shown in Table 1, where we can see that the FAST detector is most efficient among all methods, but it merely detects key-points in one scale, while others detect them in more than 10 scales. The proposed method is about 3 times faster than SURF and 10 times faster than SIFT. Real-world image matching using the approximated SIFT detector with SURF descriptor is shown in Fig. 7.



## 7 Conclusion

In this paper we have proposed a framework to efficiently and accurately approximate the SIFT scale space for key-point detection. The quantitative relation between repeated averaging filtering and Gaussian filtering has been analyzed, and a group of averaging filters has been found to accurately approximate the DOG detector. Experimental results demonstrate that the proposed method is about 10 times faster than SIFT detector, while preserving the high accuracy and high repeatability of the SIFT detector.

**Acknowledgements** This work is supported by NSFC under Grant 61860206007 and 61571313, by National Key Research and Development Program under Grant 2016YFB0800600 and 2016YFB0801100 and by funding from Sichuan Province under Grant 18GJHZ0138, and by joint funding from Sichuan University and Lu-Zhou city under 2016CDLZ-G02-SCU.

## References

1. Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2), 91–110.
2. St-Charles, P. L., Bilodeau, G. A., & Bergevin, R. (2016). Fast image gradients using binary feature convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 1–9).
3. Loncomilla, P., Ruiz-del-Solar, J., & Martnez, L. (2016). Object recognition using local invariant features for robotic applications: A survey. *Pattern Recognition*, 60, 499–514.
4. Ma, W., Wen, Z., Wu, Y., Jiao, L., Gong, M., Zheng, Y., et al. (2016). Remote sensing image registration with modified SIFT and enhanced feature matching. *IEEE Geoscience and Remote Sensing Letters*, 14(1), 3–7.
5. Yang, C., Wanyu, L., Yanli, Z., & Hong, L. (2016). The research of video tracking based on improved SIFT algorithm. In *2016 IEEE International Conference on Mechatronics and Automation (ICMA)* (pp. 1703–1707). Piscataway, NJ: IEEE.
6. Stcharles, P. L., Bilodeau, G. A., & Bergevin, R. (2016). Fast image gradients using binary feature convolutions. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* (pp. 1074–1082). Piscataway, NJ: IEEE.
7. Bay, H., Ess, A., Tuytelaars, T., & Van Gool, L. (2008). Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3), 346–359.
8. Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). Brief: Binary robust independent elementary features. In *European Conference on Computer Vision* (pp. 778–792). Berlin: Springer.
9. Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. R. (2011). ORB: An efficient alternative to SIFT or SURF. In *2011 IEEE International Conference on Computer Vision (ICCV)* (pp. 2564–2571). Piscataway, NJ: IEEE.
10. Leutenegger S, Chli M, Siegwart R. Y. (2011). BRISK: Binary robust invariant scalable keypoints. In *2011 IEEE International Conference on Computer Vision (ICCV)* (pp. 2548–2555). Piscataway, NJ: IEEE.
11. Alahi, A., Ortiz, R., & Vanderghaynst, P. (2012). Freak: Fast retina keypoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 510–517). Piscataway, NJ: IEEE.

12. Mitsugami, I. (2013). Bundler: Structure from motion for unordered image collections. *Journal of the Institute of Image Information & Television Engineers*, 65(4), 479–482.
13. Agrawal, M., Konolige, K., & Blas, M. R. (2008). CenSurE: Center surround extremas for realtime feature detection and matching. In *European Conference on Computer Vision (ECCV)* (pp. 102–115). Berlin: Springer.
14. Matas, J., Chum, O., Urban, M., & Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, 22(10), 761–767.
15. Grabner, M., Grabner, H., & Bischof, H. (2006). Fast approximated SIFT. In *Asian Conference on Computer Vision* (pp. 918–927). Berlin: Springer.
16. Crowley, J. L., & Parker, A. C. (1984). A representation for shape based on peaks and ridges in the difference of low-pass transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2), 156–170.
17. Lindeberg, T. (1994). Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, 21(2), 225–270, 21(2):224–270.
18. Hess, R. (2010). An open-source SIFTLibrary. In *Proceedings of the Inter National Conference on Multimedia* (pp. 1493–1496). New York, NY: ACM.
19. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., et al. (2005). A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1–2), 43–72.