10.2478/msr-2013-0021

MEASUREMENT SCIENCE REVIEW, Volume 13, No. 3, 2013

# A Comparative Study of SIFT and its Variants

Jian Wu[1], Zhiming Cui[1], Victor S. Sheng[2], Pengpeng Zhao[1], Dongliang Su[1], Shengrong Gong[1]

[1]The Institute of Intelligent Information Processing and Application, Soochow University, Suzhou 215006, China, jianwu@suda.edu.cn

[2]Department of Computer Science, University of Central Arkansas, Conway 72035, USA

**SIFT is an image local feature description algorithm based on scale-space. Due to its strong matching ability, SIFT has many applications in different fields, such as image retrieval, image stitching, and machine vision. After SIFT was proposed, researchers have never stopped tuning it. The improved algorithms that have drawn a lot of attention are PCA-SIFT, GSIFT, CSIFT, SURF and ASIFT. In this paper, we first systematically analyze SIFT and its variants. Then, we evaluate their performance in different situations: scale change, rotation change, blur change, illumination change, and affine change. The experimental results show that each has its own advantages. SIFT and CSIFT perform the best under scale and rotation change. CSIFT improves SIFT under blur change and affine change, but not illumination change. GSIFT performs the best under blur change and illumination change. ASIFT performs the best under affine change. PCA-SIFT is always the second in different situations. SURF performs the worst in different situations, but runs the fastest.**

**Keywords: Image matching, local feature, SIFT, PCA-SIFT, GSIFT, CSIFT, SURF, ASIFT**

## 1. INTRODUCTION

IMAGE MATCHING is an important research direction in computer vision and image processing. It is also a necessary precondition to solve many practical problems. Many researchers are dedicated to improving the performance of image matching techniques, and have proposed a variety of algorithms [1]. The image matching algorithms can be divided into two categories: global feature-based matching algorithms and local feature-based matching algorithms [2]. Comparing with global feature-based matching algorithms, local feature-based matching algorithms are more stable. They have been applied successfully in many real-world applications, such as object recognition, texture recognition, image retrieval, robot localization, video data mining, building panoramas, and object category recognition [3]-[5].

Local feature-based matching algorithms include two stages: interest point detection and description. Good local features should have the following proper characteristics. Feature detection has a high repeatability rate and high speed. Feature description has a low feature dimension, which is easy to achieve quick matching and robustness to illumination, rotation, and viewpoint change. David G. Lowe proposed a local feature description algorithm SIFT (Scale-invariant Feature Transform) [6], [7] based on the analysis of existing invariance-based feature detection methods at that time. SIFT has good stability and invariance. It detects local keypoints, which contain a large amount of information. Because of its unique advantages, it has become a popular research topic. Many researchers constantly work hard to improve it.

Tuytelaars and Mikolajczyk presented a decent overview on most widely used local invariant feature detectors [8]. This survey article consists of two parts. After reviewing local invariant feature detectors, it distinguishes among corner detectors, blob detectors, and region detectors. Juan and Gwun [9] summarized the three robust feature detection methods: SIFT, PCA-SIFT, and SURF. The performance of the robust feature detection methods is compared for scale change, rotation change, blur change, illumination change, and affine transformations. Younes et al. [10] discussed three implementations of the SIFT algorithm, i.e., Lowe's, Hess's, and theirs, respecting the parameterization suggested by Lowe in 2004.

This paper makes a more in-depth analysis and comparisons on the SIFT algorithm and its most concerned five variants. In order to find out their advantages and disadvantages, we conduct experiments to evaluate their performance in different situations: scale change, rotation change, blur change, illumination change, and affine change. Their performance is evaluated in terms of a popular measure: matching correct rate. We also further investigate their time consumption. At the end of this paper, we discuss their advantages and disadvantages, and make conclusions on this study.

## 2. RELATED WORK

Since the SIFT algorithm was formally proposed, researchers have never stopped improving it. According to the statistics of references in Google Scholar, the article [7], which published the SIFT algorithm, has more than 12,000 references. Among its variants, the numbers of references of PCA-SIFT [11], GSIFT [12], CSIFT [13], SURF [14] and ASIFT [15] are relatively high. Thus, these algorithms are selected and investigated in this paper.

The procedure of SIFT mainly includes three steps: keypoint detection, descriptor establishing, and image feature matching. Researchers improve the performance of SIFT by adjusting these steps. Most of them just adjust one of the three steps. Detailed discussions are as follows.

In the phase of descriptor establishing, SIFT uses a 128-dimensional vector to describe each keypoint. This high dimension makes the following step of SIFT (image feature matching) slow. In order to reduce the dimensionality of describing each keypoint, Y. Ke [11] uses the Principal Component Analysis method to replace the histogram

method used in SIFT. This improved version is called PCA-SIFT.

In the phase of descriptor establishing, SIFT only describes local information and does not make use of global information. E. N. Mortensen [12] introduced a SIFT descriptor with global context (called GSIFT), which adds a global texture vector to the basis of SIFT.

In the phase of keypoint detection, SIFT only uses grayscale information of an image. A lot of color information is discarded for color images. A. A. Farag [13] proposed CSIFT, which adds color invariance to the basis of SIFT and intends to overcome the shortcoming of SIFT for color images.

H. Bay [14] proposed SURF, which is very similar to SIFT but it adopts different processing methods in every step. Details of these different processing methods will be discussed in Section 3. H. Bay claimed that SURF is an enhanced version of SIFT.

J. M. Morel [15] proposed Affine-SIFT (called ASIFT), which follows affine transformation parameters to correct images and intends to resist strong affine issues.

After analyzing the adjustments of each variant of SIFT, Fig.1. shows the big picture of the relationship of each variant with SIFT. The details of each variant of SIFT and itself will be discussed further in the following section.

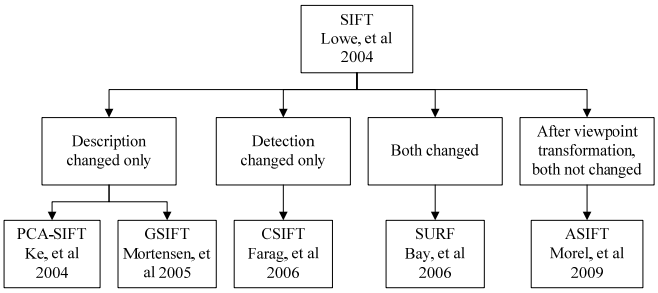



Fig.1. Relationship hierarchy of SIFT and its variants.

## 3. SIFT AND ITS VARIANTS

In this section, we will analyze SIFT first. Then we will further discuss the adjustments in each variant. At the end of this section, we will make an in-depth comparison among SIFT and its variants.

### *A. Methodology Analysis*

*SIFT analysis [7]*. The greatest characteristic of SIFT algorithm is scale invariance. In order to achieve scale invariance, SIFT uses a DoG (Difference of Gaussian) function, shown in formula (1), to do convolution on an image. It obtains different scale images by changing $\sigma$. Then, it subtracts the images which are adjacent in the same resolution to get a DoG pyramid. The DoG function is a kind of an improvement of a Gauss-Laplace algorithm [16], shown as formula (2).

$$G(x,y,\sigma)=\frac{1}{2\pi\sigma^2}\exp[-\frac{x^2+y^2}{2\sigma^2}], \tag{1}$$

$$\begin{aligned} D(x,y,\sigma) &= (G(x,y,k\sigma)-G(x,y,\sigma))*I(x,y) \\ &= L(x,y,k\sigma)-L(x,y,\sigma), \end{aligned} \tag{2}$$

where $I(x,y)$ denotes an input image, and $k$ denotes a scale coefficient of an adjacent scale-space factor. SIFT compares each point with its adjacent 26 pixels, which is the sum of eight adjacent pixels in the same layer and nine pixels in the upper and lower adjacent layers. If the point is minimum or maximum, the location and scale of this point are recorded. Therefore, SIFT gets all extreme points of DoG scale-space, and locates extreme points exactly. After that, it removes low contrast and unstable edge points. It further removes interference points, using $2\times2$ Hessian matrix obtained from adjacent difference images.

Next, in the scale of each keypoint, SIFT computes the gradient strength and direction of every neighborhood. According to gradient directions, SIFT votes in histogram for every neighborhood, and uses the summations as the gradient strengths of a keypoint. And the main direction of this keypoint is defined as the direction whose gradient strength is maximal. Then, SIFT uses the keypoint as a center to choose an adjacent $16\times16$ region. After the region is chosen, SIFT divides this region into $4\times4$ sub-regions, and sums the gradient strength in each sub-region. SIFT uses eight directions in each sub-region to generate an eight-dimensional vector. Thereby, SIFT gets a 128-dimensional feature description from 16 sub-regions, according to a certain order.

*PCA-SIFT analysis [11]*. PCA (Principle Component Analysis) is an effective and widely used data dimensionality reduction technique. It converts an original random vector, whose components have correlations, to a new random vector, whose components have no correlations, by an orthogonal transformation. PCA-SIFT uses PCA to replace the gradient histogram method in SIFT. Its description procedure can be divided into two sub-steps: projection matrix generating and descriptor establishing. It makes a new vector significantly smaller than a standard SIFT vector.

In the procedure of projecting high-dimension to low-dimension, an input vector has horizontal and vertical gradient mappings of a $41\times41$ region around the keypoint, whose dimensionality is $2\times39\times39=3042$. After the projection matrix is generated, the descriptor of PCA-SIFT can be computed.

*GSIFT analysis [12]*. GSIFT integrates global information into SIFT. Its main idea is to add global texture information, in order to make that descriptor include the wide range of curve shape information. For each detected keypoint, it establishes a vector that consists of two parts. One part is the SIFT descriptor of a local feature. Another part is a global texture vector used to distinguish similar local features. That is, a feature vector generated by GSIFT is as follows.

$$F=\begin{bmatrix}\omega S\\(1-\omega)G\end{bmatrix}, \tag{3}$$

where $S$ is a 128-dimensional vector of SIFT, $G$ is a 60-dimensional global vector, and $\omega$ is a relative weight factor. Similar to the local descriptor generation of SIFT, global texture also generates a histogram. GSIFT needs to compute the maximum curvature of each pixel, which is defined as the largest absolute eigenvalue of a Hessian matrix. For each keypoint, GSIFT establishes log-polar coordinates around it, divides a circular region, whose diameter is the length of an image diagonal, into a number of regions, and computes

discrete values of angles and radial distances of each keypoint. Thus, a curvature image can be calculated.

*CSIFT analysis [13].* CSIFT integrates color invariance [17] into the basis of SIFT. The color invariance describes optical potential radiation characteristics of objects based on a Kubelka-Munk theory, shown as follows.

$$E(\lambda, x) = e(\lambda, x)[1-\rho_f(x)]^2 R_\infty(\lambda, x) + e(\lambda, x)\rho_f(x), \quad (4)$$

where $\lambda$ denotes wavelength, $x$ is a two-dimensional vector defining an observation position, $e(\lambda, x)$ denotes spectral intensity, $\rho_f(x)$ denotes Fresnel reflection coefficient at $x$, $R_\infty(\lambda, x)$ denotes material reflectance, and $E(\lambda, x)$ denotes a reflection spectrum at the observation position.

Using formula (4), CSIFT generates the first derivative and second derivative of $E(\lambda, x)$, and defines the color image invariant $H$ as the ratio of the two derivatives:

$$H = \frac{E_\lambda}{E_{\lambda\lambda}}. \quad (5)$$

CSIFT defines $(E, E_\lambda, E_{\lambda\lambda})$ as a mapping relation to a RGB model, and $H$ is a statement of a color invariant. It has no relationship with the observation position, surface direction, light intensity, and reflection coefficient. Thus, after obtaining $H(x, y)$, it is feasible to replace $I(x, y)$ of SIFT with $H(x, y)$ in order to establish a DoG pyramid and detect extreme points in a DoG space.

*SURF analysis [14].* The basic idea of SURF is similar to that of SIFT, but SURF uses different methods for location detection and descriptor generation. Because there exists a large amount of data in an image database, and the time consumption of SIFT is rather high, Bay [14] proposed SURF to improve the detection and description efficiency of extreme points. In SURF, a quick Hessian matrix is adopted for detection, which has competitive advantages on speed and accuracy.

Meanwhile, an integral image algorithm is adopted to replace the procedure of constructing the Gaussian pyramid in SIFT. Besides, in the description phase, SURF first divides the neighborhood region of each extreme point into a number of $4\times4$ square sub-regions. Then, it computes a Haar wavelet response of each sub-region. Each response has a four-dimensional vector. Each keypoint is described with a 64-dimensional feature description of all sub-regions.

*ASIFT analysis [15].* SIFT does not perform well with images with affine change. In order to improve the performance in this situation, ASIFT simulates the rotation of camera's optical axis. It adopts an image affine transformation model resulting from the changes of viewpoint, which can be expressed as:

$$u(x, y) \rightarrow u(ax + by + e, cx + dy + f). \quad (6)$$

In the above affine model, there is an assumption that the camera is far away from the measured object. Starting from the opposite, the movement of camera may cause imaging deformation of the measured object. The angle, which is produced from the normal plane of the measured object and the mapping plane of the camera optical axis, is defined as a longitude angle. ASIFT first adds rotation transformation to an image. Then, it further obtains a series of affine images by a tilt transformation operation $u(x, y) \rightarrow u(tx, y)$ on the image in $x$ direction. Both rotation transformation and tilt transformation are achieved by means of changing the longitude angle and the latitude angle within a certain range. After these, ASIFT detects keypoints, and establishes description from the affine image.

In the matching phase, in contrast with SIFT, ASIFT not only detects more feature points, but also has relatively fewer mismatching points.

### *B. Detailed Comparisons*

The main steps of SIFT and its variants are keypoint detection and description. Based on the above methodological analysis, we briefly summarize the characteristics of SIFT and its variants in Table 1.

Table 1. Detailed comparisons of SIFT and its variants.

| | Keypoint Detection | | Keypoint Description | | |
|---|---|---|---|---|---|
| | Scale-space | Selection | Main direction | Feature Extraction | #Dimensions |
| SIFT | Different-scale images convoluted with a Gaussian function | Detect extrema in DoG space; do non-maxima suppression | Calculate a gradient amplitude of a square area; regard the direction with the maximum gradient strength as the main direction | Divide a 16×16 region into 4×4 sub-regions; create a gradient histogram for each sub-region | 128 |
| PCA-SIFT | Same as SIFT | Same as SIFT | Same as SIFT | Extract a 41×41 patch; form a 3042-dimension vector; use a project matrix to multiply with it | 20 or less |
| GSIFT | Same as SIFT | Same as SIFT | Same as SIFT | For each keypoint, create a vector consisting of SIFT description and a global texture vector | 188 |
| CSIFT | Replace grayscale with color invariant; convolute with a Gaussian function | Same as SIFT | Same as SIFT | Same as SIFT | 384 |
| SURF | Different-scale box filter convoluting with an original image | Use a Hessian matrix to determine candidate keypoints; do non-maxima suppression | Calculate a Haar wavelet response in x and y directions of each sector in a circular area; regard the direction with maximum norm as the main direction | Divide a 20×20s region into 4×4s sub-regions; calculate a Haar wavelet response | 64 |
| ASIFT | After a preprocessing - viewpoint transformation, follow SIFT's steps (i.e., the same as SIFT) | | | | |

## 4. EXPERIMENTS

In this section, we conduct experiments to investigate the performance of SIFT and its variants in different situations: scale and rotation change, blur change, illumination change, and affine change. We also investigate the time consumption of each algorithm in different situations.

### *A. Experiment Description*

SIFT and its variants are implemented in Matlab 2010a and executed on a Dell PC. It has a Pentium(R) Dual-Core CPU E5300@2.60GHz, and 4G memory, running Windows 7. In order to conduct empirical comparative analysis of SIFT and its variants, we use image data sets [18] provided by Visual Geometry Group, belonging to the Department of Engineering Science, University of Oxford. Within the data sets, a set of *boat* images is used to test scale and rotation invariance, a set of *bike* images is used to test blur invariance, a set of *Leuven* images is used to test illumination invariance, and a set of *graffiti* images is used to test affine invariance.

In all experiments, we follow the traditional common approach, using KNN (k-nearest neighbor algorithm) to match keypoints and RANSAC (random sample consensus) [19] to eliminate mismatches, in SIFT and its variants. Specifically, they use KNN to match keypoints on a KD-tree (short for *k-dimensional tree*).

In keypoint matching, target image keypoints are used as a benchmark. The goal of SIFT and its variants is to search keypoints, which are the nearest neighbor and the second nearest neighbor to target image keypoints, from the keypoints of a comparing image.

There are a lot of mismatched points in the keypoint matching process. Thus, RANSAC is used to eliminate mismatch. RANSAC is a robust parameter estimation method. Essentially, RANSAC is a process with repeated tests and constant iterations. Its basic idea is as follows: firstly, design a model according to specific issues; then repeatedly extract the smallest point set to estimate the initial values of parameters in the model; use these initial values to divide all the data into "inner point" and "outer point"; and use all the inner points to finally calculate and estimate model parameters. In keypoint matching, the model is the projective relationship, projecting keypoints on a plane to keypoints on another plane. The reflection is a projection matrix. RANSAC is used in SIFT and its variants. It is to find out a projective matrix and to make keypoints fit the projective relationship as much as possible.

### *B. Performance Evaluation under Different Situations*

*Comparisons under scale and rotation invariance.* The first set of experiments is conducted on a set of boat images [18] for investigating the performance of SIFT and its variants under scale and rotation change. The set of images is shown in Fig.2. This set has six images in total. Each represents a different scale and rotation, respectively. Our experiments match the image A1 with others (A2-A6), respectively. The experimental results are shown in Fig.3.

Fig.3. shows the matching correct rate of each algorithm between the image A1 and the others (A2-A6), respectively. From Fig.3., we can see that the performance of CSIFT is the same as SIFT. Comparing with CSIFT and SIFT, both GSIFT and ASIFT have lower matching correct rate, especially on the image A6. PCA-SIFT performs a little better than GSIFT and ASIFT on images A3-A5. However, it performs the best on the image A6. The matching correct rate of SURF is always the lowest.

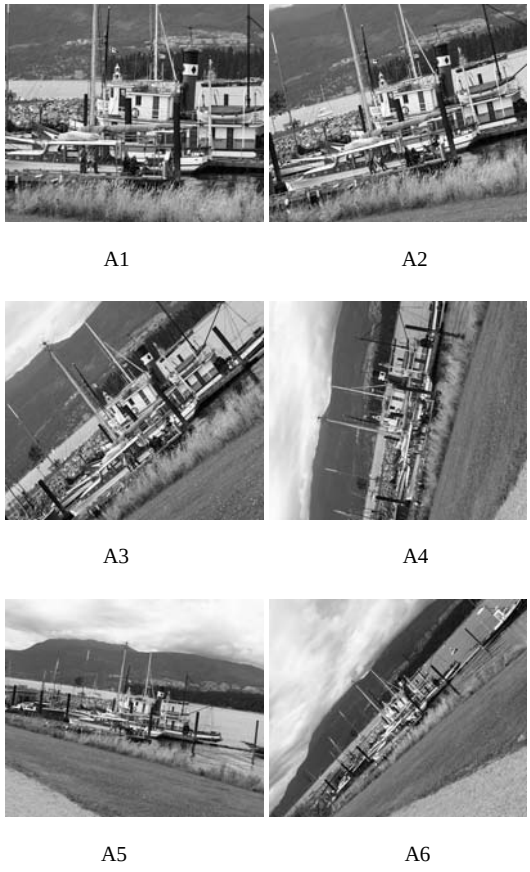



Fig.2. A boat image set for investigating scale and rotation invariance.

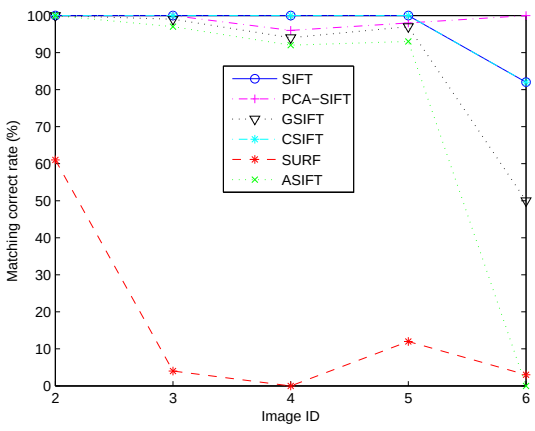
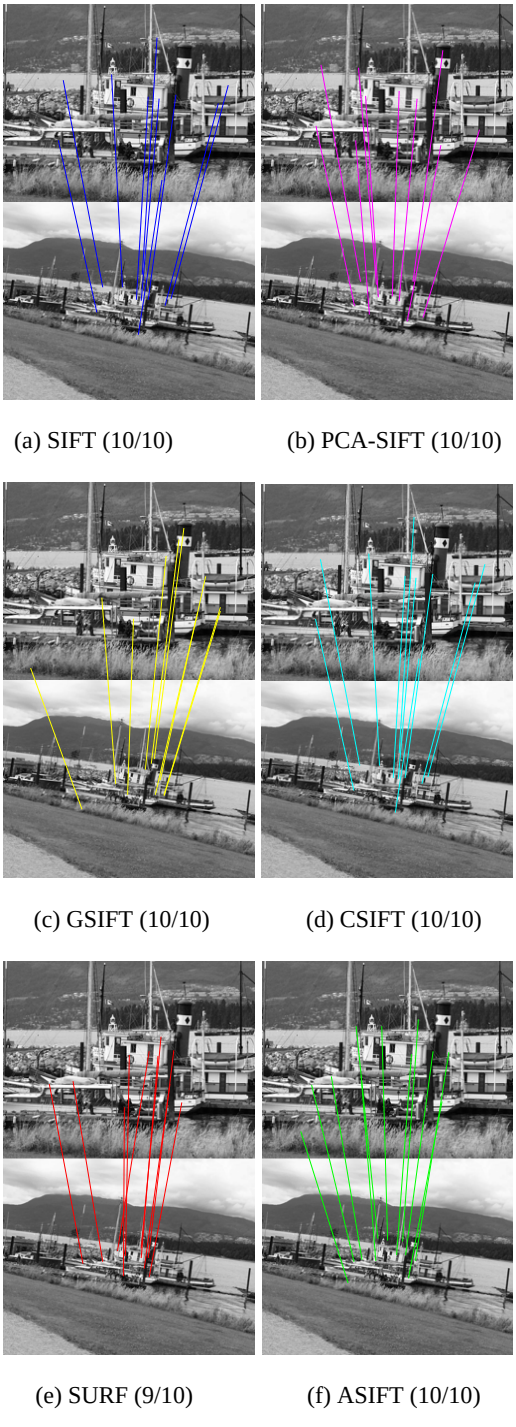



Fig.3. The experimental results under scale and rotation invariance.

Fig.4. shows the details of the first 10 matching keypoints between images A1 and A5 for SIFT and its variants, respectively. Note that we cannot show all the matching keypoints in the figure. For the sake of clarity, we only choose the first 10 matching keypoints. From this figure, we can see that the matching correct rates for SIFT and its variants are 1.0 for the first 10 matching keypoints, except SURF. It can only achieve 0.9. This further shows us that SURF does not perform well in terms of matching correct rate.



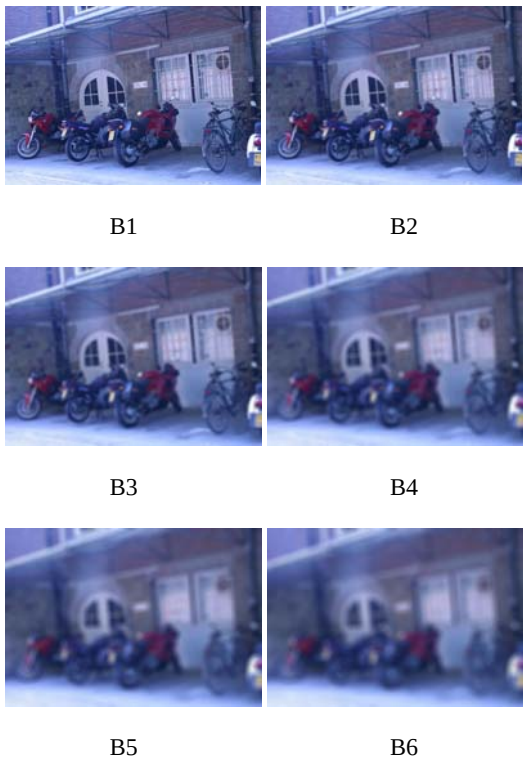
(a) SIFT (10/10) (b) PCA-SIFT (10/10)

(c) GSIFT (10/10) (d) CSIFT (10/10)

(e) SURF (9/10) (f) ASIFT (10/10)

Fig.4. The first 10 matching keypoints between images A1 and A5 under scale and rotation invariance.

*Comparisons under blur invariance*. The second set of experiments is conducted on a set of bike images for investigating the performance of SIFT and its variants under blur invariance. The set of images is shown in Fig.5. It has six images in total. Image B1 is the original one from the image set [18]. We use it to produce 10 different blur images with different fuzzy radii (from 1 to 10, refer to the horizontal axis in Fig.6)., which represent different degrees of blur. Fig.5. shows the original image and five produced images with their corresponding blur radius. The experimental results are shown in Fig.6.



B1 B2

B3 B4

B5 B6

Fig.5. A bike image set for investigating blur invariance.

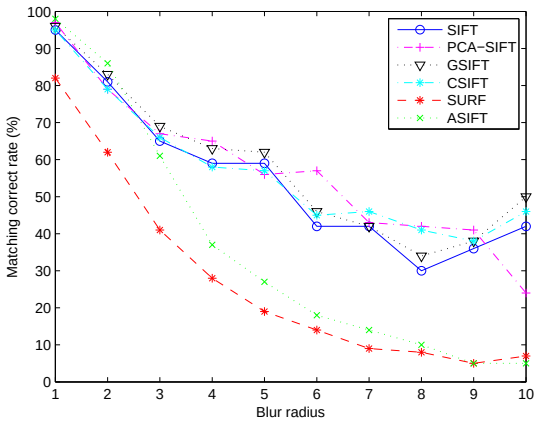
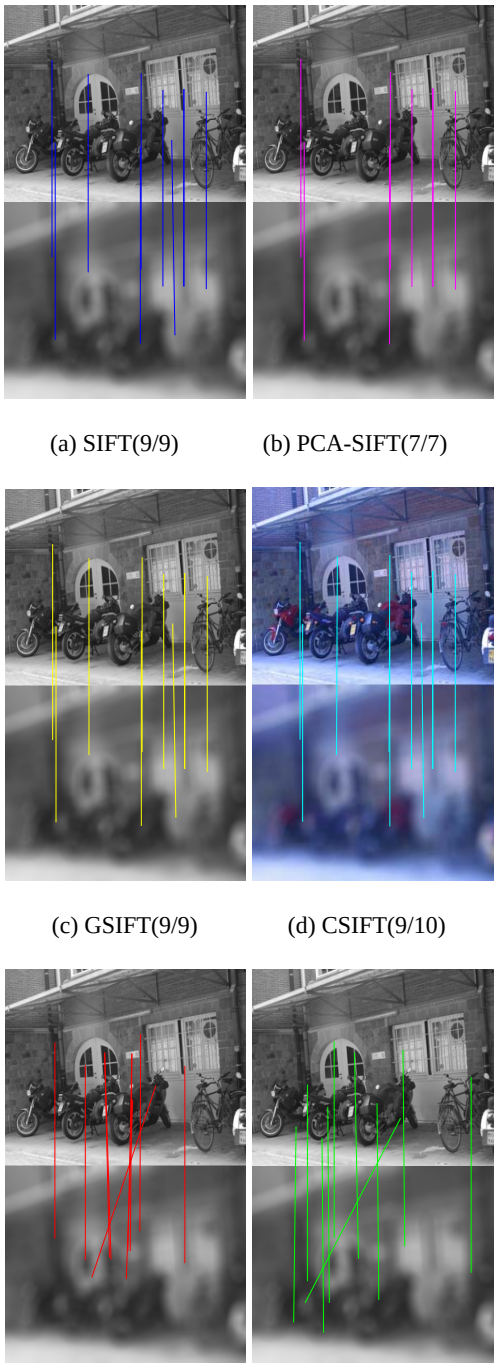



Fig.6. The experimental results under blur invariance.

Fig.6. shows the matching correct rate of each algorithm between the original image B1 and the 10 generated images, respectively. From Fig.6., we can see that the performance of PCA-SIFT, GSIFT, and CSIFT is very competitive. Among the three competitive algorithms, none consistently performs better. In general, GSIFT performs a litter better than the other two. All of the three algorithms perform better than SIFT, much better than ASIFT and SURF. Again, SURF performs the worst.



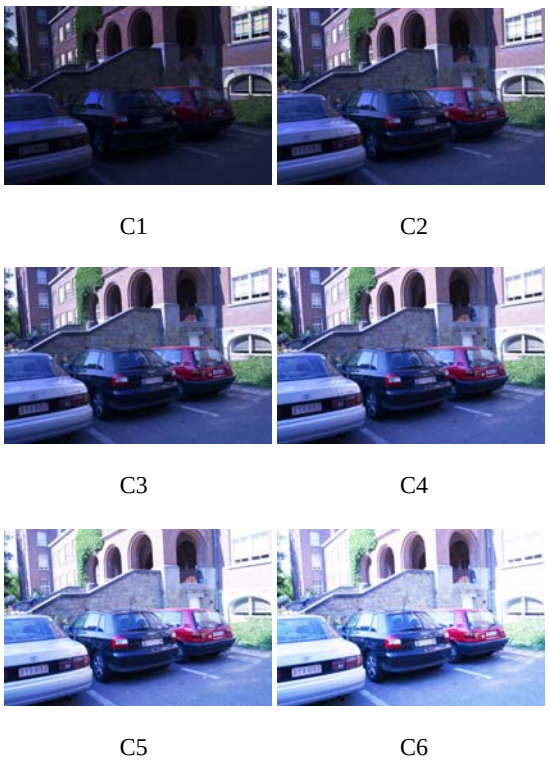
(a) SIFT(9/9) (b) PCA-SIFT(7/7)

(c) GSIFT(9/9) (d) CSIFT(9/10)

(e) SURF(8/10) (f) ASIFT(10/10)

Fig.7. The first 10 matching keypoints between images B1 and B5 under blur invariance.

Fig.7. shows the details of the first 10 matching keypoints between the original image B1 and the generated image B5 for SIFT and its variants, respectively. Note that we cannot show all the matching keypoints in the figure. For the sake of clarity, we only choose the first 10 matching keypoints. As we can see from the figure, due to high blur in B5, it caused that the number of extracted keypoints of SIFT, PCA-SIFT and GSIFT are 9, 7, and 9, respectively (less than 10). But among all the extracted keypoints, there occurs no keypoint mismatching. That means that the extracted keypoints can be matched successfully. However, there exists keypoint mismatching on CSIFT and SURF. They have one and two mismatching keypoints, respectively. We can also conclude that the performance of SURF is the worst under blur invariance.



C1 C2

C3 C4

C5 C6

Fig.8. A set of Leuven images for investigating illumination invariance.

*Comparisons under illumination invariance.* The third set of experiments is conducted on a set of Leuven images for investigating the performance of SIFT and its variants under illumination invariance. The set of images is shown in Fig.8. Please note that the image C3 is the original image [18]. We use the original image C3 to produce 10 different illumination images by decreasing and increasing brightness intensity (from -150 to 150, refer to the horizontal axis in Fig.9). The brightness intensity represents a degree of illumination. Fig.8. shows the original image C3 and five produced images (C1-C2, and C4-C5). Note that we order

the images according to their brightness. Images C1 and C2 are darker than the original C3. The images C4-C5 are brighter than the original one. The experimental results are shown in Fig.9.

Fig.9. shows the matching correct rate of each algorithm between the original image C3 and the 10 generated images, respectively. From Fig.9., we can see that the performance of PCA-SIFT and GSIFT is very competitive. They bother perform a little better than SIFT. Between PCA-SIFT and GSIFT, we can see that GSIFT performs slightly better on most images. CSIFT performs very well when images are brighter. However, its performance drops gradually when images become darker and darker. But it still performs better than ASIFT and SURF. Between ASIFT and SURF, SURF performs worse. Overall, it performs the worst.

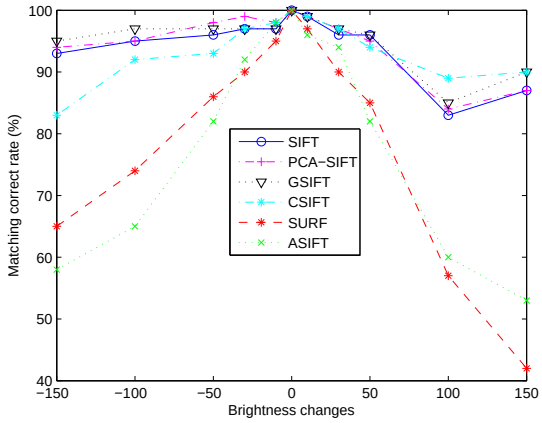
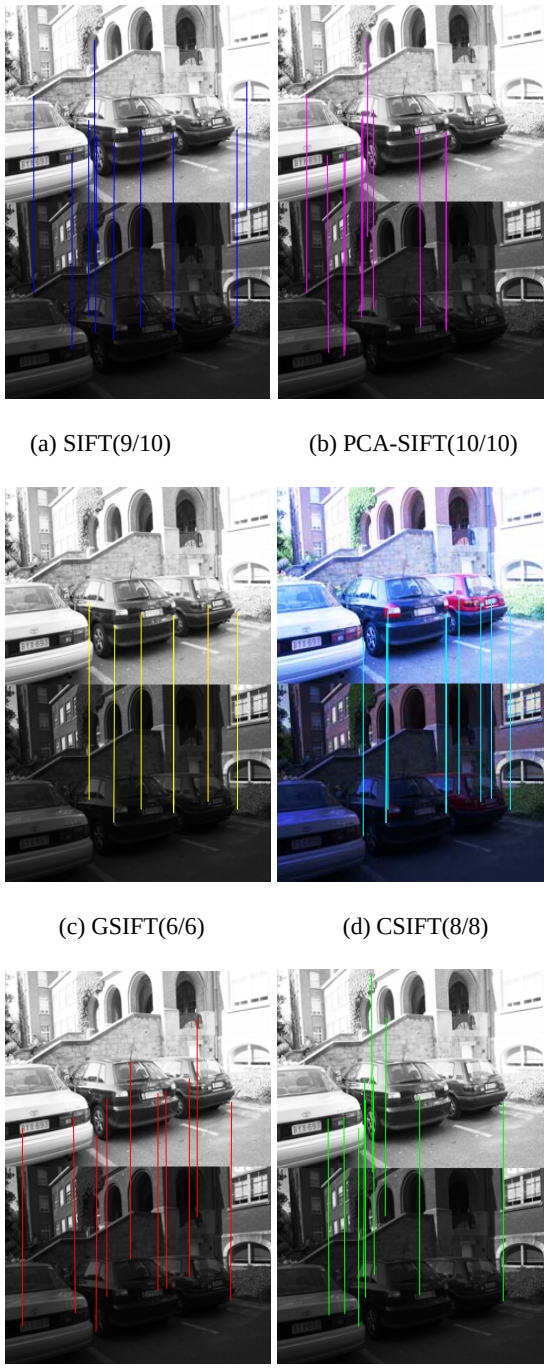



Fig.9. The experimental results under illumination invariance.

Fig.10. shows the details of the first 10 matching keypoints between the darkest image C1 and the brightest image C6 among the images shown in Fig.8., for SIFT and its variants, respectively. Note that we cannot show all the matching keypoints in the figure. For the sake of clarity, we only choose the first 10 matching keypoints. As we can see from the figure, due to the big difference in brightness intensity, the number of extracted keypoints of both GSIFT and CSIFT are 6 and 8, respectively (less than 10). Except SIFT, all keypoints extracted by the other five algorithms can be matched successfully.

*Comparisons under affine invariance*. The fourth set of experiments is conducted on the set of graffiti images [18] for investigating the performance of SIFT and its variants under affine invariance. This set has six images in total, shown in Fig.11. Each image has a different viewpoint. The experimental results are shown in Fig.12.

Fig.12. shows the matching correct rate of each algorithm between the image D1 and the others (D2-D6), respectively. From Fig.12., we can see that the performance of ASIFT is the best, followed by CSIFT, followed by PCA-SIFT and SIFT. SURF performs the worst. GSIFT performs a little better than SURF, but worse than SIFT. From Fig.12., we can also see that SIFT and its variants (except SURF) perform very well when the viewpoint angle is smaller than 30 degrees. Their performance becomes worse when the viewpoint angle is greater than 30 degrees. However, the matching correct rate of ASIFT is stable under different angles (even 60 degrees). It maintains a very high matching correct rate. It is much better than the others when the viewpoint angle increases. The performance of SURF is always relatively low. It has a small advantage over PCA-SIFT, GSIFT, and CSIFT when the viewpoint angle is greater than 50 degrees. Overall, it performs the worst.



(a) SIFT(9/10) (b) PCA-SIFT(10/10)

(c) GSIFT(6/6) (d) CSIFT(8/8)

(e) SURF(10/10) (f) ASIFT(10/10)

Fig.10. The first 10 matching keypoints between the darkest image C1 and the brightest image C6 (shown in Fig.8) under illumination invariance.

D1 D2

D3 D4

D5 D6

Fig.11. A set of graffiti images for investigating affine invariance.

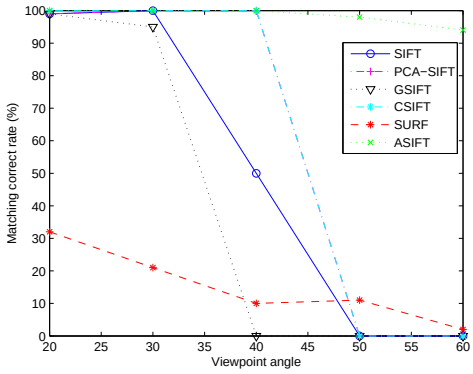
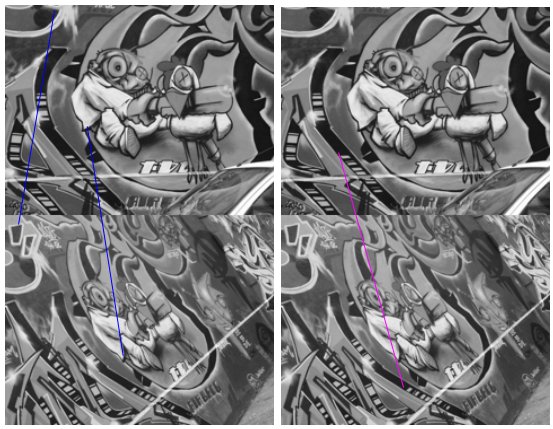


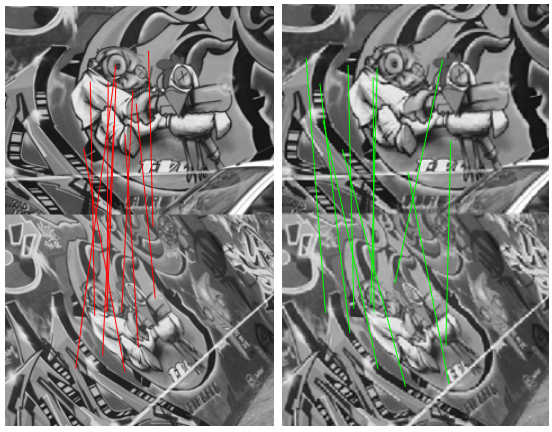

Fig.12. The experimental results under affine invariance.

Fig.13. shows the details of the first 10 matching keypoints between the images D1 and D5, for SIFT and its variants, respectively. Note that we cannot show all the matching keypoints in the figure. For the sake of clarity, we only choose the first 10 matching keypoints. As we can see from the figure, the numbers of matched keypoints extracted by SIFT, PCA-SIFT, GSIFT are 2, 1, and 0, respectively, which are much fewer than the other three algorithms (i.e., CSIFT, SURF and ASIFT). The numbers of matched keypoints extracted by CSIFT, SURF and ASIFT are 7, 5, and 10, respectively.



(a) SIFT(1/2) (b)PCA-SIFT(1/1)

(c) GSIFT(0/0) (d) CSIFT(7/7)



(e) SURF(5/10) (f) ASIFT(10/10)

Fig.13. The first 10 matching keypoints between images D1 and D5 under affine invariance.

### *C. Comparisons on Time Consumption*

Because of large computation of SIFT and its variants, it is necessary to empirically investigate their time consumption. In order to investigate the time consumption of each algorithm in different situations (i.e., scale and rotation change, blur change, illumination change, and affine change), we have four pairs of images from the above four image sets. Specifically, we have a pair of images (A1 and

A5) for investigating the time consumption of each algorithm under scale and rotation change, pair of images (the original image B1 and the generated image B5) for investigating the time consumption of each algorithm under blur change, a pair of images (the darkest image C1 and the brightest image C6 among the images shown in Fig.8.) for investigating the time consumption of each algorithm under illumination change, and a pair of images (D1 and D5) for investigating the time consumption of each algorithm under affine change.

Table 2. Comparisons on time consumption (in milliseconds).

| Algorithm | Step | Boat | Bikes | Leuven | Graffiti | **Avg.** |
|---|---|---|---|---|---|---|
| SIFT | Feature Extraction | 0.9935 | 1.0014 | 0.7166 | 1.0186 | 0.9325 |
| | Image Matching | 1.5922 | 1.1457 | 0.3946 | 1.6921 | 1.2062 |
| | **Total** | **2.5857** | **2.1471** | **1.1112** | **2.7107** | **2.1387** |
| PCA-SIFT | Feature Extraction | 2.0280 | 1.7195 | 1.7068 | 2.3556 | 1.9525 |
| | Image Matching | 0.5070 | 0.1910 | 0.3012 | 0.7852 | 0.4461 |
| | **Total** | **2.535** | **1.9105** | **2.008** | **3.1408** | **2.3986** |
| GSIFT | Feature Extraction | 2.4032 | 2.6617 | 1.1051 | 2.4148 | 2.1462 |
| | Image Matching | 1.0572 | 1.4134 | 0.5002 | 1.2198 | 1.0477 |
| | **Total** | **3.4604** | **4.0751** | **1.6053** | **3.6346** | **3.1939** |
| CSIFT | Feature Extraction | 2.5374 | 2.5493 | 1.8223 | 2.4545 | 2.3409 |
| | Image Matching | 3.0959 | 3.5783 | 2.3132 | 2.4709 | 2.8646 |
| | **Total** | **5.6333** | **6.1276** | **4.1355** | **4.9254** | **5.2055** |
| SURF | Feature Extraction | 1.8048 | 1.8857 | 1.3853 | 1.8061 | 1.7205 |
| | Image Matching | 0.0970 | 0.1394 | 0.0536 | 0.1029 | 0.09823 |
| | **Total** | **1.9018** | **2.0251** | **1.4389** | **1.909** | **1.8187** |
| ASIFT | Feature Extraction | 4.7647 | 5.1204 | 3.4687 | 4.4752 | 4.4573 |
| | Image Matching | 3.0420 | 2.2801 | 2.1562 | 1.7897 | 2.3170 |
| | **Total** | **7.8067** | **7.4005** | **5.6249** | **6.2649** | **6.7743** |

Table 2. shows the time consumption of each algorithm running on each pair of images. In order to see the details of the time consumed by each algorithm, we also show feature extraction time and matching time separately in the table. The average time of each algorithm running on the four pairs is shown in the last column. Note that the time in the table is in milliseconds. Please note also that the time in the table is a relative value for comparisons only, since we reduced the resolution of the images in the experiments.

From the last column of Table 2., we can see that SURF is the fastest one, followed by SIFT, followed by PAC-SIFT, followed by GSIFT, and followed by CSIFT. ASIFT is the slowest. If we have a closer look, we can see that SURF saves time on image matching. Its image matching time is always the lowest in all situations. However, its feature extraction time is much higher than that of SIFT. SIFT has the lowest feature extraction time in all situations. ASIFT is the slowest in all situations, because of its highest feature extraction time in all situations. Among all six algorithms, CSIFT is the second slowest. The reason is that its image matching time is the highest in all situations, and its feature extraction time is the second slowest in all situations.

Table 2. also shows that the order of the six algorithms, based on their time consumption, does not change with the different situations.

### *D. Experiment Summarization*

In the above sub-sections, we empirically compared the performance of SIFT and its variants in four different situations, i.e., scale and rotation invariance, blur invariance, illumination invariance, and affine invariance. We also investigated the time consumption of each algorithm in the above four situations. In this sub-section, we analyze the experimental results qualitatively. Thus, we can have general ideas of the performance of each algorithm in different situations. We rate the experimental results in four grades, i.e., Best, Better, Good, and Common. The qualitative analysis is shown in Table 3.

Table 3. Qualitative Summarization.

| Algorithm | Scale & Rotation | Blur | Illumination | Affine | Time Cost |
|---|---|---|---|---|---|
| SIFT | Best | Good | Better | Good | Better |
| PCA-SIFT | Better | Better | Better | Good | Better |
| GSIFT | Good | Best | Best | Good | Better |
| CSIFT | Best | Better | Good | Better | Good |
| SURF | Common | Common | Common | Common | Best |
| ASIFT | Good | Common | Common | Best | Common |

As we can see from the above table, SIFT keeps the invariance on scale and rotation change, and illumination change. It maintains a certain degree of stability for image blur and affine transformation. PCA-SIFT replaces the histogram method used inside SIFT with a principal component analysis method. It has a good performance in four different situations. It also has reasonable time consumption (a little slower than SIFT). GSIFT integrates a global texture vector with the descriptors generated by SIFT. It has significant advancements under blur and illumination changes. CSIFT takes full advantage of image color information, and uses color invariance based on SIFT. It maintains a good stability in scale, rotation, blur, and affine transformation. Based on an integral image, SURF uses a quick Hessian matrix, which has an advantage in speed and accuracy while detecting keypoints. Thus, its computational efficiency is greatly improved. Compared with the other algorithms, it has obvious advantages in terms of speed. ASIFT has a full sense of affine invariance. Not only it is able to detect more keypoints, but also its keypoint mismatching is relatively low.

## 5. CONCLUSIONS

This paper systematically analyzed the major members of the SIFT family, including SIFT, PCA-SIFT, GSIFT, CSIFT, SURF and ASIFT. They are image local feature description algorithms based on scale-space. In this paper, we empirically evaluated their performance in different situations: scale and rotation change, blur change, illumination change, and affine change. Because of large computation of SIFT and its variants, we also investigated their time consumption empirically in different situations. Based on our experimental results, we qualitatively analyzed the performance of each algorithm, which provided the general ideas and suggestions of how to choose the best algorithm for a specific real-world problem.

The experimental results show that each algorithm has its own advantage. SIFT and CSIFT perform the best under scale and rotation change. CSIFT improves SIFT under blur and affine changes, but not illumination change. GSIFT performs the best under blur and illumination changes. ASIFT performs the best under affine change. PCA-SIFT is always the second in different situations. SURF performs the worst in different situations, but runs the fastest.

Image matching techniques based on feature keypoints have important theoretical and practical significance. These methods can be used for image registration, image retrieval, and so on. Local feature description algorithms for complex situations are constantly proposed. After years of development, they have better and better performance and robustness. As we know, local features and global features have advantages and disadvantages, respectively. Thus, it is promising to develop novel algorithms that combine local features with global features (such as color, texture, and spatial relationships). We can imagine that these novel algorithms will have better practical values in target recognitions.

## ACKNOWLEDGMENTS

This research was partially supported by the Natural Science Foundation of China under grant No. 61003054, 61170020, and 61170124, the Program for Postgraduates Research Innovation in Jiangsu Province in 2011 under grant No. CXLX11_0072, the Beforehand Research Foundation of Soochow University, and the National Science Foundation (IIS-1115417).

## REFERENCES

[1] Ouyang, W., Tombari, F., Mattoccia, S., Di Stefano, L., Cham, W.-K. (2012). Performance evaluation of full search equivalent pattern matching algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34 (1), 127-143.

[2] Birinci, M., Diaz-de-Maria, F., Abdollahian, G. (2011). Neighborhood matching for object recognition algorithms based on local image features. In *IEEE Digital Signal Processing Workshop and IEEE Signal Processing Education Workshop (DSP/SPE)*, 4-7 January 2011. IEEE, 157-162.

[3] Mian, A., Bennamoun, M., Owens, R. (2010). On the repeatability and quality of keypoints for local feature-based 3D object retrieval from cluttered scenes. *International Journal of Computer Vision*, 89 (2-3), 348-361.

[4] Mikulka, J., Gescheidtova, E., Bartusek, K. (2012). Soft-tissues image processing: Comparison of traditional segmentation methods with 2D active contour methods. *Measurement Science Review*, 12 (4), 153-161.

[5] Kim, D., Rho, S., Hwang, E. (2012). Local feature-based multi-object recognition scheme for surveillance. *Engineering Applications of Artificial Intelligence*, 25 (7), 1373–1380.

[6] Lowe, D.G. (1999). Object recognition from local scale invariant features. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, 20-27 September 1999. IEEE, Vol. 2, 1150-1157.

[7] Lowe, D.G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60 (2), 91-110.

[8] Tuytelaars, T., Mikolajczyk, K. (2008). Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3 (3), 177-280.

[9] Juan, L., Gwun, O. (2009). A comparison of SIFT, PCA-SIFT and SURF. *International Journal of Image Processing*, 3 (4), 143-152.

[10] Younes, L., Romaniuk, B., Bittar, E. (2012). A comprehensive and comparative survey of the SIFT algorithm - feature detection, description, and characterization. In *Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP)*. SciTePress, Vol. 1, 467-474.

[11] Ke, Y., Sukthankar, R. (2004). PCA-SIFT: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition (CVPR 2004)*, 27 June – 2 July 2004. IEEE, Vol. 2, 506-513.

[12] Mortensen, E.N., Deng, H., Shapiro, L. (2005). A SIFT descriptor with global context. In *Computer Vision and Pattern Recognition (CVPR 2005)*, 20-25 June 2005. IEEE, Vol. 1, 184-190.

[13] Abdel-Hakim, A.E., Farag, A.A. (2006). CSIFT: A SIFT descriptor with color invariant characteristics. In *Computer Vision and Pattern Recognition (CVPR 2006)*, 17-22 June 2006. IEEE, Vol. 2, 1978-1983.

[14] Bay, H., Tuytelaars, T., Gool, L.V. (2006). SURF: Speeded up robust features. In *Computer Vision – ECCV 2006 : 9th European Conference on Computer Vision*, 7-13 May 2006. Springer, Part II, 404-417.

[15] Morel, J.M., Yu, G. (2009). ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, 2 (2), 438-469.

[16] Rabbani, H. (2011). Statistical modeling of low SNR magnetic resonance images in wavelet domain using Laplacian prior and two-sided Rayleigh noise for visual quality improvement. *Measurement Science Review*, 11 (4), 125-130.

[17] Benveniste, R., Unsalan, C. (2011). A color invariant for line stripe-based range scanners. *The Computer Journal*, 54 (5), 738-753.

[18] Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A. (2005). A comparison of affine region detectors. *International Journal of Computer Vision*, 65 (1/2), 43-72.

[19] Wu, Z., Radke, R.J. (2012). Using scene features to improve wide-area video surveillance. In *Computer Vision and Pattern Recognition Workshops (CVPRW)*, 16-21 June 2012. IEEE, 50-57.

Received July 16, 2012.
Accepted June 11, 2013.