Image Features Detection, Description and Matching

M. Hassaballah, Aly Amin Abdelmgeid and Hammam A. Alshazly

Abstract Feature detection, description and matching are essential components of various computer vision applications, thus they have received a considerable attention in the last decades. Several feature detectors and descriptors have been proposed in the literature with a variety of definitions for what kind of points in an image is potentially interesting (i.e., a distinctive attribute). This chapter introduces basic notation and mathematical concepts for detecting and describing image features. Then, it discusses properties of perfect features and gives an overview of various existing detection and description methods. Furthermore, it explains some approaches to feature matching. Finally, the chapter discusses the most used techniques for performance evaluation of detection and description algorithms.

Keywords Interest points · Feature detector · Feature descriptor · Feature extraction · Feature matching

1 Introduction

Over the last decades, image feature detectors and descriptors have become popular tools in the computer vision community and they are being applied widely in a large number of applications. Image representation [1], image classification and retrieval [2–5], object recognition and matching [6–10], 3D scene reconstruction [11], motion tracking [12–14], texture classification [15, 16], robot localization [17–19],

M. Hassaballah (✉) · H.A. Alshazly
Faculty of Science, Department of Mathematics, South Valley University,
Qena 83523, Egypt
e-mail: m.hassaballah@svu.edu.eg

H.A. Alshazly
e-mail: hammam.alshazly@sci.svu.edu.eg

A.A. Abdelmgeid
Faculty of Science, Department of Computer Science,
Minia University, El Minia, Egypt
e-mail: abdelmgeid@yahoo.com

A.I. Awad and M. Hassaballah (eds.), *Image Feature Detectors and Descriptors*,
Studies in Computational Intelligence 630, DOI 10.1007/978-3-319-28854-3_2

and biometrics systems [20–22], all rely on the presence of stable and representative features in the image. Thus, detecting and extracting the image features are vital steps for these applications.

In order to establish correspondences among a collection of images, where feature correspondences between two or more images are needed, it is necessary to identify a set of salient points in each image [8, 23]. In a classification task, feature descriptors of a query image are matched with all trained image features and the trained image giving maximum correspondence is considered the best match. In that case, feature descriptor matching can be based on distance measures such as Euclidean or Mahalanobis. In image registration, it is necessary to spatially align two or more images of a scene captured by different sensors at different times. The main steps involved in performing image registration or alignment tasks are: feature detection, feature matching, derivation of transformation functions based on corresponding features in images, and reconstruction of images based on the derived transformation functions [24]. In the context of matching and recognition, the first step of any matching/recognition system is to detect interest locations in the images and describe them. Once the descriptors are computed, they can be compared to find a relationship between images for performing matching/recognition tasks. Also, for online street-level virtual navigation application, we need a feature detector and a feature descriptor to extract features from planar images (panoramic images) [25].

The basic idea is to first detect interest regions (keypoints) that are covariant to a class of transformations. Then, for each detected regions, an invariant feature vector representation (i.e., descriptor) for image data around the detected keypoint is built. Feature descriptors extracted from the image can be based on second-order statistics, parametric models, coefficients obtained from an image transform, or even a combination of these measures. Two types of image features can be extracted form image content representation; namely global features and local features. Global features (e.g., color and texture) aim to describe an image as a whole and can be interpreted as a particular property of the image involving all pixels. While, local features aim to detect keypoints or interest regions in an image and describe them. In this context, if the local feature algorithm detects n keypoints in the image, there are n vectors describing each one's shape, color, orientation, texture and more. The use of global colour and texture features are proven surprisingly successful for finding similar images in a database, while the local structure oriented features are considered adequate for object classification or finding other occurrences of the same object or scene [26]. Meanwhile, the global features can not distinguish foreground from background of an image, and mix information from both parts together [27].

On the other hand, as the real time applications have to handle ever more data or to run on mobile devices with limited computational capabilities, there is a growing need for local descriptors that are fast to compute, fast to match, memory efficient, and yet exhibiting good accuracy. Additionally, local feature descriptors are proven to be a good choice for image matching tasks on a mobile platform, where occlusions and missing objects can be handled [18]. For certain applications, such as camera calibration, image classification, image retrieval, and object tracking/recognition, it is very important for the feature detectors and descriptors to be robust to changes in brightness or viewpoint and to image distortions (e.g., noise, blur, or illumination) [28]. While, other specific visual recognition tasks, such as face detection or recognition, requires the use of specific detectors and descriptors [29].

In the literature, a large variety of feature extraction methods have been proposed to compute reliable descriptors. Some of these feature descriptors were exclusively designed for a specific application scenario such as shape matching [29, 30]. Among these descriptors, the scale invariant feature transform (SIFT) descriptor [31] utilizing local extrema in a series of difference of Gaussian (DoG) functions for extracting robust features and the speeded-up robust features (SURF) descriptor [32] partly inspired by the SIFT descriptor for computing distinctive invariant local features quickly are the most popular and widely used in several applications. These descriptors represent salient image regions by using a set of hand-crafted filters and nonlinear operations. In the rest of the chapter, we give an overview for these methods and algorithms as well as their improvements proposed by developers. Furthermore, the basic notations and mathematical concepts for detecting and describing image features are introduced. We also explore in detail what is the quantitative relation between the detectors and descriptors as well as how to evaluate their performance.

2 Definitions and Principles

2.1 Global and Local Features

In image processing and computer vision tasks, we need to represent the image by features extracted therefrom. The raw image is perfect for the human eye to extract all information from; however that is not the case with computer algorithms. There are generally two methods to represent images, namely, global features and local features. In the global feature representation, the image is represented by one multidimensional feature vector, describing the information in the whole image. In other words, the global representation method produces a single vector with values that measure various aspects of the image such as color, texture or shape. Practically, a single vector from each image is extracted and then two images can be compared by comparing their feature vectors. For example, when one wants to distinguish images of a sea (blue) and a forest (green), a global descriptor of color would produce quite different vectors for each category. In this context, global features can be interpreted as a particular property of image involving all pixels. This property can be color histograms, texture, edges or even a specific descriptor extracted from some filters applied to the image [33]. On the other hand, the main goal of local feature representation is to distinctively represent the image based on some salient regions while remaining invariant to viewpoint and illumination changes. Thus, the image is represented based on its local structures by a set of local feature descriptors extracted from a set of image regions called interest regions (i.e., keypoints) as illustrated in Fig. 1. Most local features represent texture within the image patch.

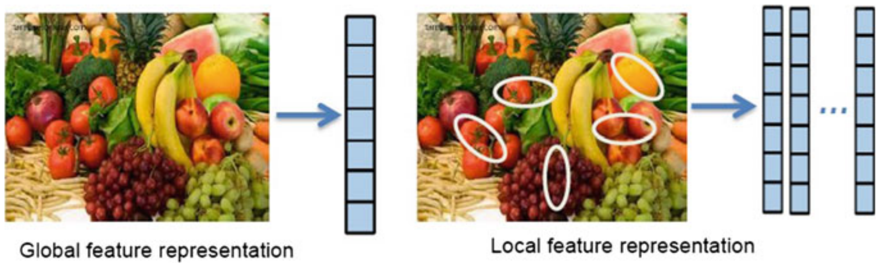

Fig. 1 Global and local image features representation

Generally, using what kind of features might greatly depend on the applications on hand. Developers prefer the most discriminative ones. For example, a person with a bigger nose and smaller eyes, and a person with a smaller nose and bigger eyes may have similar mug shot in terms of histogram or intensity distribution. Then, local features or the global pattern distilled from local feature clusters seem to be more discriminative. Whereas, for very large datasets in the web-scale image indexing application, it is appropriate to consider global features. Also, global features are useful in applications where a rough segmentation of the object of interest is available. The advantages of global features are that they are much faster and compact while easy to compute and generally require small amounts of memory. Nevertheless, the global representation suffers from well-known limitations, in particular they are not invariant to significant transformations and sensitive to clutter and occlusion. In some applications, such as copy detection, most of the illegal copies are very similar to the original; they have only suffered from compression, scaling or limited cropping. In contrast, the advantage of local features is their superior performance [34]. Meanwhile, using local features for large-scale image search have much higher performance than global features provide [35]. Besides, as the local structures are more distinctive and stable than other structures in smooth regions, it is expected to be more useful for image matching and object recognition. However, they usually require a significant amount of memory because the image may have hundreds of local features. As a solution for this problem, researchers suggest aggregating local image descriptors into a very compact vector representation and optimizing the dimensionality reduction of these vectors [35].

2.2 *Characteristics of Feature Detectors*

Tuytelaars and Mikolajczyk [27] define a local feature as "*it is an image pattern which differs from its immediate neighborhood*". Thus, they consider the purpose of local invariant features is to provide a representation that allows to efficiently match local structures between images. That is, we want to obtain a sparse set of local measurements that capture the essence of the underlying input images and encode their interesting structures. To meet this goal, the feature detectors and extractors must have certain properties keeping in mind that the importance of these properties depends on the actual application settings and compromises need to be made. The following properties are important for utilizing a feature detector in computer vision applications:

- Robustness, the feature detection algorithm should be able to detect the same feature locations independent of scaling, rotation, shifting, photometric deformations, compression artifacts, and noise.
- Repeatability, the feature detection algorithm should be able to detect the same features of the same scene or object repeatedly under variety of viewing conditions.
- Accuracy, the feature detection algorithm should accurately localize the image features (same pixel locations), especially for image matching tasks, where precise correspondences are needed to estimate the epipolar geometry.
- Generality, the feature detection algorithm should be able to detect features that can be used in different applications.
- Efficiency, the feature detection algorithm should be able to detect features in new images quickly to support real-time applications.
- Quantity, the feature detection algorithm should be able to detect all or most of the features in the image. Where, the density of detected features should reflect the information content of the image for providing a compact image representation.

2.3 *Scale and Affine Invariance*

Actually, finding correspondences based on comparing regions of fixed shape like rectangles or circles are not reliable in the presence of some geometric and photometric deformations as they affect the regions' shapes. Also, objects in digital images appear in different ways depending on the scale of observation. Consequently, scale changes are of important implications when analyzing image contents. Different techniques have been proposed to address the problem of detecting and extracting invariant image features under these conditions. Some are designed to handle scale changes, while others go further to handle affine transformations. In order to address the scale changes, these techniques assume that the change in scale is same in all directions (i.e., uniform) and they search for stable features across all possible scales using a continuous kernel function of scale known as scale space. Where, the image size is varied and a filter (e.g., Gaussian filter) is repeatedly applied to smooth subsequent layers, or by leaving the original image unchanged and varies only the filter size as shown in Fig. 2. More details about feature detection with scale changes can be found in [36], where a framework is presented for generating hypotheses about interesting scale levels in image data by searching for scale-space extrema in the scale normalized Laplacian of Gaussian (LoG).

On the other hand, in the case of an affine transformation the scaling can be different in each direction. The nonuniform scaling has an influence on the localization,

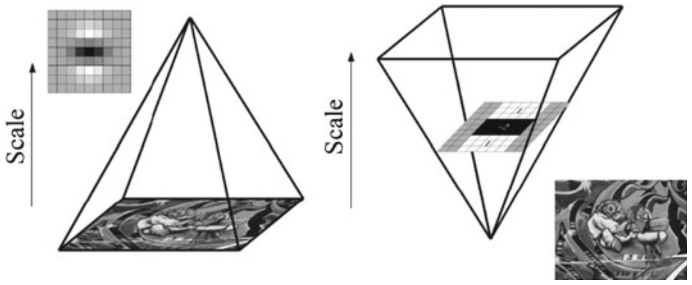

Fig. 2 Constructing the scale space structure

the scale and the shape of the local structure. Therefore, the scale invariant detectors fail in the case of significant affine transformations. Hence, detectors designed to detect the image features under uniform scaling need to be extended to be affine invariant detectors that can detect the affine shape of the local image structures. Thus, these affine invariant detectors can be seen as a generalization of the scale invariant detector.

Generally, affine transformations are constructed using sequences of translations, scales, flips, rotations, and shears. An affine transformation (affinity) is any linear mapping that preserves collinearity and ratios of distances. In this sense, affine indicates a special class of projective transformations that do not move any object from the affine space $\mathbb{R}^3$ to the plane at infinity or conversely. Briefly, the affine transformation of $\mathbb{R}^n$ is a map $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ of the form

$$f(Y) = \mathbb{A}Y + \mathbb{B} \tag{1}$$

for all $Y \in \mathbb{R}^n$, where $\mathbb{A}$ is a linear transformation of $\mathbb{R}^n$. In some special cases, if $\det(\mathbb{A}) > 0$, the transformation is called orientation-preserving, while, if $\det(\mathbb{A}) < 0$, it is orientation-reversing. It is well known that a function is invariant under a certain family of transformations if its value does not change when a transformation from this family is applied to its argument. The second moment matrix provides the theory for estimating affine shape of the local image features. Examples of affine invariant detectors are Harris-affine, Hessian-affine, and maximally stable extremal regions (MSER). It must be borne in mind that the detected features by these detectors are transformed from circles to ellipses.

3 Image Feature Detectors

Feature detectors can be broadly classified into three categories: single-scale detectors, multi-scale detectors, and affine invariant detectors. In a nutshell, single scale means that there is only one representation for the features or the object contours using detector's internal parameters. The single-scale detectors are invariant to image transformations such as rotation, translation, changes in illuminations and addition of noise. However, they are incapable to deal with the scaling problem. Given two images of the same scene related by a scale change, we want to determine whether same interest points can be detected or not. Therefore, it is necessary to build multi-scale detectors capable of extracting distinctive features reliably under scale changes. Details of single-scale and multi-scale detectors as well as affine invariant detectors are discussed in the following sections.

3.1 Single-Scale Detectors

3.1.1 Moravec's Detector

Moravec's detector [37] is specifically interested in finding distinct regions in the image that could be used to register consecutive image frames. It has been used as a corner detection algorithm in which a corner is a point with low self-similarity. The detector tests each pixel in a given image to see if a corner is present. It considers a local image patch centered on the pixel and then determines the similarity between the patch and the nearby overlapping patches. The similarity is measured by taking the sum of square differences (SSD) between the centered patch and the other image patches. Based on the value of SSD, three cases need to be considered as follows:

- If the pixel in a region of uniform intensity, then the nearby patches will look similar or a small change occurs.
- If the pixel is on an edge, then the nearby patches in a parallel direction to the edge will result in a small change and the patches in a direction perpendicular to the edge will result in a large change.
- If the pixel is on a location with large change in all directions, then none of the nearby patches will look similar and the corner can be detected when the change produced by any of the shifts is large.

The smallest SSD between the patch and its neighbors (horizontal, vertical and on the two diagonals) is used as a measure for cornerness. A corner or an interest point is detected when the SSD reaches a local maxima. The following steps can be applied for implementing Moravec's detector:

1. Input: gray scale image, window size, threshold T
2. For each pixel (x, y) in the image compute the intensity variation V from a shift (u, v) as

$$V_{u,v}(x, y) = \sum_{\forall a,b \in window} [I(x + u + a, y + v + b) - I(x + a, y + b)]^2 \quad (2)$$

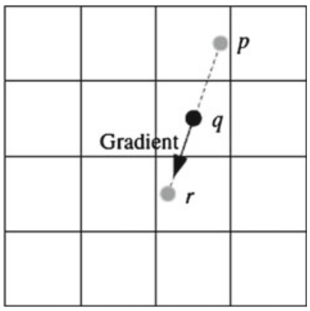

Fig. 3 Performing the non-maximum suppression

3. Construct the cornerness map by calculating the cornerness measure $C(x, y)$ for each pixel (x, y)
$$C(x, y) = \min(V_{u,v}(x, y)) \tag{3}$$
4. Threshold the cornerness map by setting all $C(x, y)$ below the given threshold value T to zero.
5. Perform non-maximum suppression to find local maxima. All non-zero points remaining in the cornerness map are corners.

For performing non-maximum suppression, an image is scanned along its gradient direction, which should be perpendicular to an edge. Any pixel that is not a local maximum is suppressed and set to zero. As illustrated in Fig. 3, p and r are the two neighbors along the gradient direction of q. If the pixel value of q is not larger than the pixel values of both p and r, it is suppressed. One advantage of Moravec's detector is that, it can detect majority of the corners. However, it is not isotropic; intensity variation is calculated only at a discrete set of shifts (i.e., the eight principle directions) and any edge is not in one of the eight neighbors' directions is assigned a relatively large cornerness measure. Thus, it is not invariant to rotation resulting in the detector is of a poor repeatability rate.

3.1.2 Harris Detector

Harris and Stephens [38] have developed a combined corner and edge detector to address the limitations of Moravec's detector. By obtaining the variation of the auto-correlation (i.e., intensity variation) over all different orientations, this results in a more desirable detector in terms of detection and repeatability rate. The resulting detector based on the auto-correlation matrix is the most widely used technique. The 2×2 symmetric auto-correlation matrix used for detecting image features and describing their local structures can be represented as

$$M(x, y) = \sum_{u,v} w(u, v) * \begin{bmatrix} I_x^2(x, y) & I_x I_y(x, y) \\ I_x I_y(x, y) & I_y^2(x, y) \end{bmatrix} \tag{4}$$

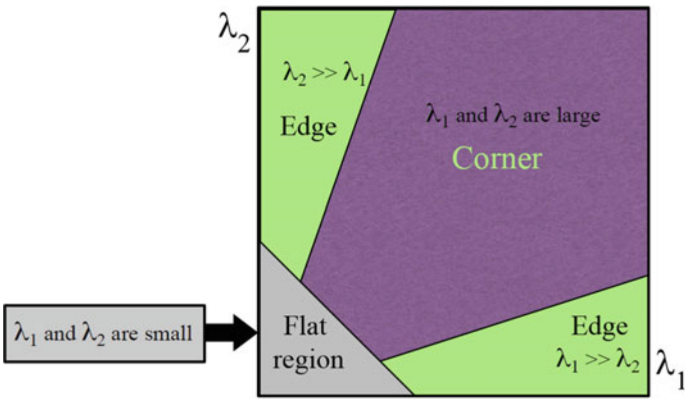

Fig. 4 Classification of image points based on the eigenvalues of the autocorrelation matrix M

where I_x and I_y are local image derivatives in the x and y directions respectively, and $w(u, v)$ denotes a weighting window over the area (u, v). If a circular window such as a Gaussian is used, then the response will be isotropic and the values will be weighted more heavily near the center. For finding interest points, the eigenvalues of the matrix M are computed for each pixel. If both eigenvalues are large, this indicates existence of the corner at that location. An illustrating diagram for classification of the detected points is shown in Fig. 4. Constructing the response map can be done by calculating the cornerness measure $C(x, y)$ for each pixel (x, y) using

$$C(x, y) = \det(M) - K(trace(M))^2 \tag{5}$$

where

$$\det(M) = \lambda_1 * \lambda_2, \quad and \quad trace(M) = \lambda_1 + \lambda_2 \tag{6}$$

The K is an adjusting parameter and λ_1, λ_2 are the eigenvalues of the auto-correlation matrix. The exact computation of the eigenvalues is computationally expensive, since it requires the computation of a square root. Therefore, Harris suggested using this cornerness measure that combines the two eigenvalues in a single measure. The non-maximum suppression should be done to find local maxima and all non-zero points remaining in the cornerness map are the searched corners.

3.1.3 SUSAN Detector

Instead of using image derivatives to compute corners, Smith and Brady [39] introduced a generic low-level image processing technique called SUSAN (Smallest Univalue Segment Assimilating Nucleus). In addition to being a corner detector, it has been used for edge detection and image noise reduction. A corner is detected by placing a circular mask of fixed radius to every pixel in the image. The center pixel is referred to as the nucleus, where pixels in the area under the mask are compared with the nucleus to check if they have similar or different intensity values. Pixels having almost the same brightness as the nucleus are grouped together and the resulting area is termed USAN (Univalue Segment Assimilating Nucleus). A corner is found at locations where the number of pixels in the USAN reaches a local minimum and below a specific threshold value T. For detecting corners, the similar comparison function $C(r, r_0)$ between each pixel within the mask and mask's nucleus is given by

$$C(r, r_0) = \begin{cases} 1, & \text{if } |I(r) - I(r_0)| \leq T, \\ 0, & \text{otherwise,} \end{cases} \tag{7}$$

and the size of USAN region is

$$n(r_0) = \sum_{r \in c(r0)} C(r, r_0) \tag{8}$$

where r_0 and r are nucleus's coordinates and the coordinates of other points within the mask, respectively. The performance of SUSAN corner detector mainly depends on the similar comparison function $C(r, r_0)$, which is not immune to certain factors impacting imaging (e.g., strong luminance fluctuation and noises).

SUSAN detector has some advantages such as: (i) no derivatives are used, thus, no noise reductions or any expensive computations are needed; (ii) High repeatability for detecting features; and (iii) invariant to translation and rotation changes. Unfortunately, it is not invariant to scaling and other transformations, and a fixed global threshold is not suitable for general situation. The corner detector needs an adaptive threshold and the shape of mask should be modified.

3.1.4 FAST Detector

FAST (Features from Accelerated Segment Test) is a corner detector originally developed by Rosten and Drummondn [40, 41]. In this detection scheme, candidate points are detected by applying a segment test to every image pixel by considering a circle of 16 pixels around the corner candidate pixel as a base of computation. If a set of n contiguous pixels in the Bresenham circle with a radius r are all brighter than the intensity of candidate pixel (denoted by I_p) plus a threshold value t, $I_p + t$, or all darker than the intensity of candidate pixel minus the threshold value $I_p - t$, then p is classified as a corner. A high-speed test can be used to exclude a very large number of non-corner points; the test examines only the four pixels 1, 5, 9 and 13. A corner

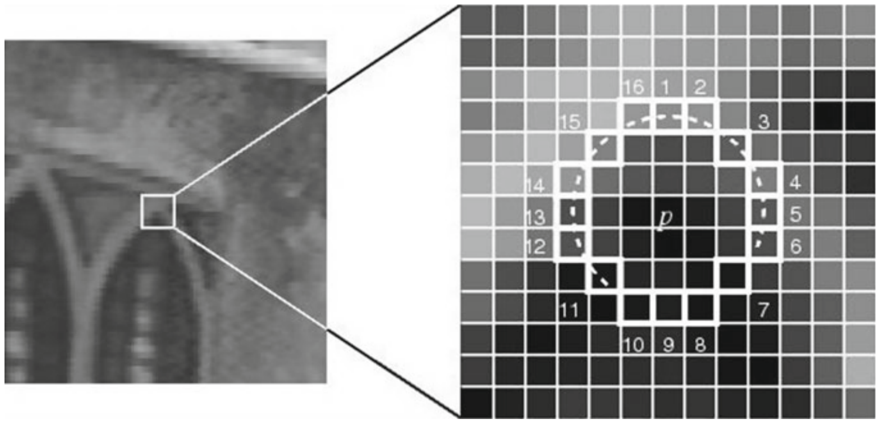

Fig. 5 Feature detection in an image patch using FAST detector [41]

can only exist if three of these test pixels are brighter than $I_p + t$ or darker than $I_p - t$ and the rest of pixels are then examined for final conclusion. Figure 5 illustrates the process, where the highlighted squares are the pixels used in the corner detection. The pixel at p is the center of a candidate corner. The arc is indicated by the dashed line passes through 12 contiguous pixels which are brighter than p by a threshold. The best results are achieved using a circle with $r = 3$ and $n = 9$.

Although the high speed test yields high performance, it suffers from several limitations and weakness as mentioned in [41]. An improvement for addressing these limitations and weakness points is achieved using a machine learning approach. The ordering of questions used to classify a pixel is learned by using the well-known decision tree algorithm (ID3), which speeds this step up significantly. As the first test produces many adjacent responses around the interest point, an additional criteria is applied to perform a non-maximum suppression. This allows for precise feature localization. The used cornerness measure at this step is

$$C(x, y) = max(\sum_{j \in S_{bright}} |I_{p \to j} - I_p| - t, \sum_{j \in S_{dark}} |I_p - I_{p \to j}| - t) \tag{9}$$

where $I_{p \to j}$ denotes the pixels laying on the Bresenham circle. In this way, the processing time remains short because the second test is performed only on a fraction of image points that passed the first test.

In other words, the process operates in two stages. First, corner detection with a segment test of a given n and a convenient threshold is performed on a set of images (preferably from the target application domain). Each pixel of the 16 locations on the circle is classified as darker, similar, or brighter. Second, employing the ID3 algorithm on the 16 locations to select the one that yields the maximum information gain. The non-maximum suppression is applied on the sum of the absolute difference between the pixels in the contiguous arc and the center pixel. Notice that the corners detected using the ID3 algorithm may be slightly different from the results obtained with segment test detector due to the fact that decision tree model depends on the training data, which could not cover all possible corners. Compared to many existing detectors, the FAST corner detector is very suitable for real-time video processing applications because of its high-speed performance. However, it is not invariant to scale changes and not robust to noise, as well as it depends on a threshold, where selecting an adequate threshold is not a trivial task.

3.1.5 Hessian Detector

The Hessian blob detector [42, 43] is based on a 2×2 matrix of second-order derivatives of image intensity $I(x, y)$, called the Hessian matrix. This matrix can be used to analyze local image structures and it is expressed in the form

$$H(x, y, \sigma) = \begin{bmatrix} I_{xx}(x, y, \sigma) & I_{xy}(x, y, \sigma) \\ I_{xy}(x, y, \sigma) & I_{yy}(x, y, \sigma) \end{bmatrix} \tag{10}$$

where I_{xx}, I_{xy}, and I_{yy} are second-order image derivatives computed using Gaussian function of standard deviation σ. In order to detect interest features, it searches for a subset of points where the derivatives responses are high in two orthogonal directions. That is, the detector searches for points where the determinant of the Hessian matrix has a local maxima

$$\det(H) = I_{xx}I_{yy} - I_{xy}^2 \tag{11}$$

By choosing points that maximize the determinant of the Hessian, this measure penalizes longer structures that have small second derivatives (i.e., signal changes) in a single direction. Applying non-maximum suppression using a window of size 3×3 over the entire image, keeping only pixels whose value is larger than the values of all eight immediate neighbors inside the window. Then, the detector returns all the remaining locations whose value is above a pre-defined threshold T. The resulting detector responses are mainly located on corners and on strongly textured image areas. While, the Hessian matrix is used for describing the local structure in a neighborhood around a point, its determinant is used to detect image structures that exhibit signal changes in two directions. Compared to other operators such as Laplacian, the determinant of the Hessian responds only if the local image pattern contains significant variations along any two orthogonal directions [44]. However, using second order derivatives in the detector is sensitive to noise. In addition, the local maxima are often found near contours or straight edges, where the signal changes in only one direction [45]. Thus, these local maxima are less stable as the localization is affected by noise or small changes in neighboring pattern.

3.2 Multi-scale Detectors

3.2.1 Laplacian of Gaussian (LoG)

Laplacian-of-Gaussian (LoG), a linear combination of second derivatives, is a common blob detector. Given an input image $I(x, y)$, the scale space representation of the image defined by $L(x, y, \sigma)$ is obtained by convolving the image by a variable scale Gaussian kernel $G(x, y, \sigma)$ where

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{12}$$

and

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}} \tag{13}$$

For computing the Laplacian operator, the following formula is used

$$\nabla^2 L(x, y, \sigma) = L_{xx}(x, y, \sigma) + L_{yy}(x, y, \sigma) \tag{14}$$

This results in strong positive responses for dark blobs and strong negative responses for bright blobs of size $\sqrt{2\sigma}$. However, the operator response is strongly dependent on the relationship between the size of the blob structures in the image domain and the size of the smoothing Gaussian kernel. The standard deviation of the Gaussian is used to control the scale by changing the amount of blurring. In order to automatically capture blobs of different size in the image domain, a multi-scale approach with automatic scale selection is proposed in [36] through searching for scale space extrema of the scale-normalized Laplacian operator.

$$\nabla^2_{norm} L(x, y, \sigma) = \sigma^2 (L_{xx}(x, y, \sigma)_+ L_{yy}(x, y, \sigma) \tag{15}$$

Which can also detect points that are simultaneously local maxima/minima of $\nabla^2_{norm} L(x, y, \sigma)$ with respect to both space and scale. The LoG operator is circularly symmetric; it is therefore naturally invariant to rotation. The LoG is well adapted to blob detection due to this circular symmetry property, but it also provides a good estimation of the characteristic scale for other local structures such as corners, edges, ridges and multi-junctions. In this context, the LoG can be applied for finding the characteristic scale for a given image location or for directly detecting scale-invariant regions by searching for 3D (location + scale) extrema of the LoG function as illustrated in Fig. 6. The scale selection properties of the Laplacian operator are studied in detail in [46].

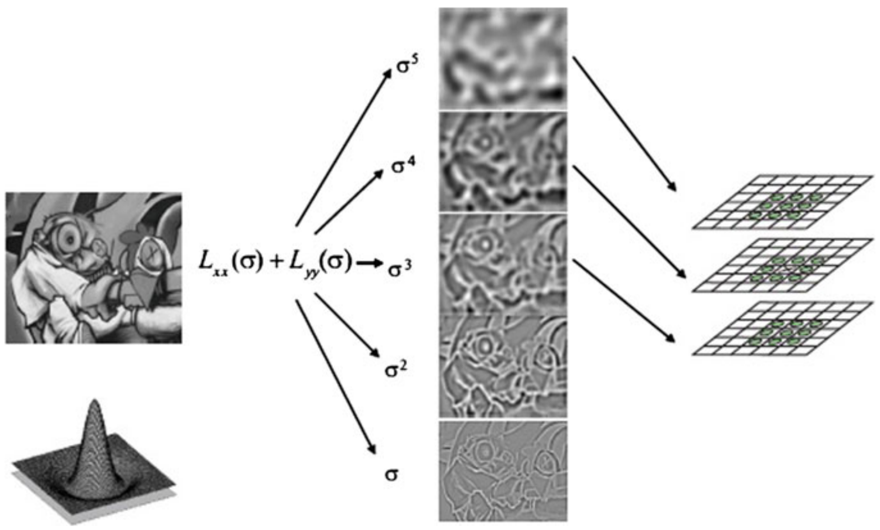

Fig. 6 Searching for 3D scale space extrema of the LoG function

3.2.2 Difference of Gaussian (DoG)

In fact, the computation of LoG operators is time consuming. To accelerate the computation, Lowe [31] proposed an efficient algorithm based on local 3D extrema in the scale-space pyramid built with Difference-of-Gaussian(DoG) filters. This approach is used in the scale-invariant feature transform (SIFT) algorithm. In this context, the DoG gives a close approximation to the Laplacian-of-Gaussian (LoG) and it is used to efficiently detect stable features from scale-space extrema. The DoG function $D(x, y, \sigma)$ can be computed without convolution by subtracting adjacent scale levels of a Gaussian pyramid separated by a factor k.

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{16}$$

Feature types extracted by DoG can be classified in the same way as for the LoG operator. Also, the DoG region detector searches for 3D scale space extrema of the DoG function as shown in Fig. 7. The computation of LoG operators is time consuming. The common drawback of both the LoG and DoG representations is that the local maxima can also be detected in neighboring contours of straight edges, where the signal change is only in one direction, which make them less stable and more sensitive to noise or small changes [45].

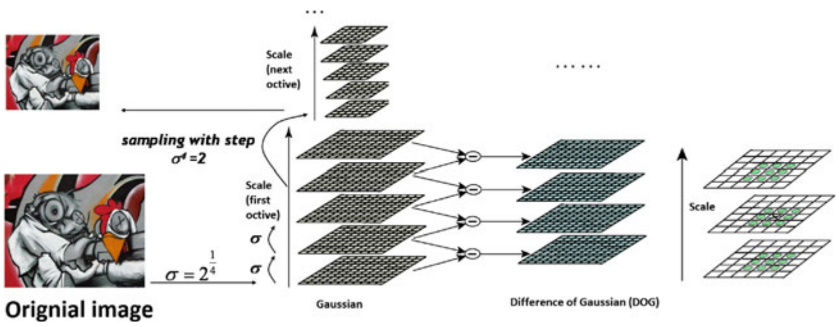
Scale (next octive)
sampling with step $\sigma^4 = 2$
Scale (first octive)
σ
σ
$\sigma = 2^{\frac{1}{4}}$
Orignial image
Gaussian
Difference of Gaussian (DOG)
Scale

Fig. 7 Searching for 3D scale space extrema in the DoG function [31]

3.2.3 Harris-Laplace

Harris-Laplace is a scale invariant corner detector proposed by Mikolajczyk and Schmid [45]. It relies on a combination of Harris corner detector and a Gaussian scale space representation. Although Harris-corner points have been shown to be invariant to rotation and illumination changes, the points are not invariant to the scale. Therefore, the second-moment matrix utilized in that detector is modified to make it independent of the image resolution. The scale adapted second-moment matrix used in the Harris-Laplace detector is represented as

$$M(x, y, \sigma_I, \sigma_D) = \sigma_D^2 \; g(\sigma_I) \begin{bmatrix} I_x^2(x, y, \sigma_D) & I_x I_y(x, y, \sigma_D) \\ I_x I_y(x, y, \sigma_D) & I_y^2(x, y, \sigma_D) \end{bmatrix} \tag{17}$$

where I_x and I_y are the image derivatives calculated in their respective direction using a Gaussian kernel of scale σ_D. The parameter σ_I determines the current scale at which the Harris corner points are detected in the Gaussian scale-space. In other words, the derivative scale σ_D decides the size of gaussian kernels used to compute derivatives. While, the integration scale σ_I is used to performed a weighted average of derivatives in a neighborhood. The multi-scale Harris cornerness measure is computed using the determinant and the trace of the adapted second moment matrix as

$$C(x, y, \sigma_I, \sigma_D) = det[M(x, y, \sigma_I, \sigma_D)] - \alpha.trace^2[M(x, y, \sigma_I, \sigma_D)] \tag{18}$$

The value of the constant α is between 0.04 and 0.06. At each level of the representation, the interest points are extracted by detecting the local maxima in the 8-neighborhood of a point (x, y). Then, a threshold is used to reject the maxima of small cornerness, as they are less stable under arbitrary viewing conditions

$$C(x, y, \sigma_I, \sigma_D) > Threshold_{Harris} \tag{19}$$

In addition, the Laplacian-of-Gaussian is used to find the maxima over the scale. Where, only the points for which the Laplacian attains maxima or its response is above a threshold are accepted.

$$\sigma_I^2 |L_{xx}(x, y, \sigma_I) + L_{yy}(x, y, \sigma_I)| > Threshold_{Laplacian} \tag{20}$$

The Harris-Laplace approach provides a representative set of points which are characteristic in the image and in the scale dimension. It also dramatically reduces the number of redundant interest points compared to Multi-scale Harris. The points are invariant to scale changes, rotation, illumination, and addition of noise. Moreover, the interest points are highly repeatable. However, the Harris-Laplace detector returns a much smaller number of points compared to the LoG or DoG detectors. Also, it fails in the case of affine transformations.

3.2.4 Hessian-Laplace

Similar to Harris-Laplace, the same idea can also be applied to the Hessian-based detector, leading to a scale invariant detector termed, Hessian-Laplace. At first, we build an image scale-space representation using Laplacian filters or their approximations DoG filters. Then, use the determinant of the Hessian to extract scale invariant blob-like features. Hessian-Laplace detector extracts large number of features that cover the whole image at a slightly lower repeatability compared to its counterpart Harris-Laplace. Furthermore, the extracted locations are more suitable for scale estimation due to the similarity of the filters used in spatial and scale localization, both based on second order Gaussian derivatives. Bay et al. [32] claimed that Hessian-based detectors are more stable than the Harris-based counterparts. Likewise, approximating LoG by DoG for acceleration, the Hessian determinant is approximated using integral images technique [29] resulting in the Fast Hessian detector [32].

3.2.5 Gabor-Wavelet detector

Recently, Yussof and Hitam [47] proposed a multi-scale interest points detector based on the principle of Gabor wavelets. The Gabor wavelets are biologically motivated convolution kernels in the shape of plane waves restricted by a Gaussian envelope function, whose kernels are similar to the response of the two-dimensional receptive field profiles of the mammalian simple cortical cell. The Gabor wavelets take the form of a complex plane wave modulated by a Gaussian envelope function

$$\psi_{u,v}(z) = \frac{||K_{u,v}||^2}{\sigma^2} \; e^{(\frac{||K_{u,v}||^2 ||z||^2}{2\sigma^2})} \left[e^{izK_{u,v}} - e^{\frac{\sigma^2}{2}} \right] \tag{21}$$

where $K_{u,v} = K_v e^{i\phi u}$, $z = (x, y)$, u and v define the orientation and scale of the Gabor wavelets, $K_v = K_{max}/f^v$ and $\phi_u = \pi u/8$, K_{max} is the maximum frequency, and $f =$

$\sqrt{2}$ is the spacing factor between kernels in the frequency domain. The response of an image I to a wavelet ψ is calculated as the convolution

$$G = I * \psi \tag{22}$$

The coefficients of the convolution, represent the information in a local image region, which should be more effective than isolated pixels. The advantage of Gabor wavelets is that they provides simultaneous optimal resolution in both space and spatial frequency domains. Additionally, Gabor wavelets have the capability of enhancing low level features such as peaks, valleys and ridges. Thus, they are used to extract points from the image at different scales by combining multi orientations of image responses. The interest points are extracted at multiple scales with a combination of uniformly spaced orientation. The authors proved that the extracted points using Gabor-wavelet detector have high accuracy and adaptability to various geometric transformations.

3.3 Affine Invariant Detectors

The feature detectors discussed so far exhibit invariance to translations, rotations and uniform scaling; assuming that the localization and scale are not affected by an affine transformation of the local image structures. Thus, they partially handle the challenging problem of affine invariance, keeping in mind that the scale can be different in each direction rather than uniform scaling. Which in turn makes the scale invariant detectors fail in the case of significant affine transformations. Therefore, building a detector robust to perspective transformations necessitates invariance to affine transformations. An affine invariant detector can be seen as a generalized version of a scale invariant detector. Recently, some features detectors have been extended to extract features invariant to affine transformations. For instance, Schaffalitzky and Zisserman [48] extended the Harris-Laplace detector by affine normalization. Mikolajczyk and Schmid [45] introduced an approach for scale and affine invariant interest point detection. Their algorithm simultaneously adapts location, scale and shape of a point neighborhood to obtain affine invariant points. Where, Harris detector is adapted to affine transformations and the affine shape of a point neighborhood is estimated based on the second moment matrix. This is achieved by following the iterative estimation scheme proposed by Lindberg and Gårding [49] as follows:

1. Identify initial region points using scale-invariant Harris-Laplace detector.
2. For each initial point, normalize the region to be affine invariant using affine shape adaptation.
3. Iteratively estimate the affine region; selection of proper integration scale, differentiation scale and spatially localize interest points.
4. Update the affine region using these scales and spatial localizations.
5. Repeat step 3 if the stopping criterion is not met.

Similar to Harris-affine, the same idea can also be applied to the Hessian-based detector, leading to an affine invariant detector termed as Hessian-affine. For a single image, the Hessian-affine detector typically identifies more reliable regions than the Harris-affine detector. The performance changes depending on the type of scene being analyzed. Further, the Hessian-affine detector responds well to textured scenes in which there are a lot of corner-like parts. However, for some structured scenes, like buildings, the Hessian-affine detector performs very well. A thorough analysis of several state-of-the-art affine detectors have been done by Mikolajczyk and Schmid [50].

There are several other feature detectors that are not discussed in this chapter due to space limitation. Fast Hessian or the Determinant of Hessian (DoH) [32], MSER [51, 52], are some examples. A more detailed discussion of these detectors can be found in [44, 45, 53].

4 Image Feature Descriptor

Once a set of interest points has been detected from an image at a location $p(x, y)$, scale s, and orientation θ, their content or image structure in a neighborhood of p needs to be encoded in a suitable descriptor for discriminative matching and insensitive to local image deformations. The descriptor should be aligned with θ and proportional to the scale s. There are a large number of image feature descriptors in the literature; the most frequently used descriptors are discussed in the following sections.

4.1 Scale Invariant Feature Transform (SIFT)

Lowe [31] presented the scale-invariant feature transform (SIFT) algorithm, where a number of interest points are detected in the image using the Difference-of-Gaussian (DOG) operator. The points are selected as local extrema of the DoG function. At each interest point, a feature vector is extracted. Over a number of scales and over a neighborhood around the point of interest, the local orientation of the image is estimated using the local image properties to provide invariance against rotation. Next, a descriptor is computed for each detected point, based on local image information at the characteristic scale. The SIFT descriptor builds a histogram of gradient orientations of sample points in a region around the keypoint, finds the highest orientation value and any other values that are within 80 % of the highest, and uses these orientations as the dominant orientation of the keypoint.

The description stage of the SIFT algorithm starts by sampling the image gradient magnitudes and orientations in a 16×16 region around each keypoint using its scale to select the level of Gaussian blur for the image. Then, a set of orientation histograms is created where each histogram contains samples from a 4×4 subregion of the original neighborhood region and having eight orientations bins in each. A

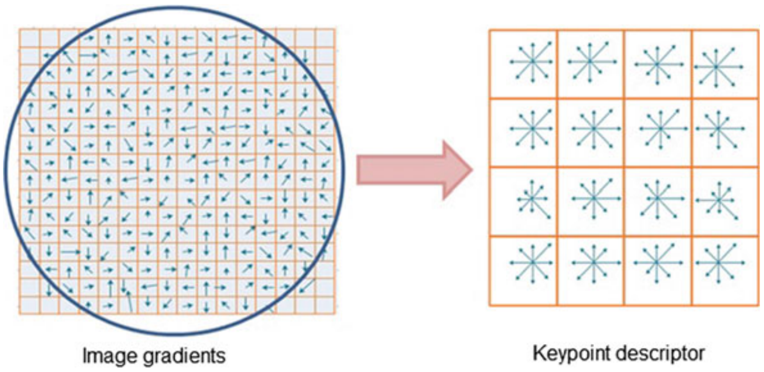

Fig. 8 A schematic representation of the SIFT descriptor for a 16 × 16 pixel patch and a 4 × 4 descriptor array

Gaussian weighting function with σ equal to half the region size is used to assign weight to the magnitude of each sample point and gives higher weights to gradients closer to the center of the region, which are less affected by positional changes. The descriptor is then formed from a vector containing the values of all the orientation histograms entries. Since there are 4×4 histograms each with 8 bins, the feature vector has $4 \times 4 \times 8 = 128$ elements for each keypoint. Finally, the feature vector is normalized to unit length to gain invariance to affine changes in illumination. However, non-linear illumination changes can occur due to camera saturation or similar effects causing a large change in the magnitudes of some gradients. These changes can be reduced by thresholding the values in the feature vector to a maximum value of 0.2, and the vector is again normalized. Figure 8 illustrates the schematic representation of the SIFT algorithm; where the gradient orientations and magnitudes are computed at each pixel and then weighted by a Gaussian falloff (indicated by overlaid circle). A weighted gradient orientation histogram is then computed for each subregion.

The standard SIFT descriptor representation is noteworthy in several respects: the representation is carefully designed to avoid problems due to boundary effects-smooth changes in location, orientation and scale do not cause radical changes in the feature vector; it is fairly compact, expressing the patch of pixels using a 128 element vector; while not explicitly invariant to affine transformations, and the representation is surprisingly resilient to deformations such as those caused by perspective effects. These characteristics are evidenced in excellent matching performance against competing algorithms under different scales, rotations and lighting. On the other hand, the construction of the standard SIFT feature vector is complicated and the choices behind its specific design are not clear resulting in the common problem of SIFT its high dimensionality, which affects the computational time for computing the descriptor (significantly slow). As an extension to SIFT, Ke and Sukthankar [54] proposed PCA-SIFT to reduce the high dimensionality of original SIFT descriptor using the standard Principal Components Analysis (PCA) technique to the normalized gradient image patches extracted around keypoints. It extracts a 41×41 patch at the given scale and computes its image gradients in the vertical and horizontal directions and creates a feature vector from concatenating the gradients in both directions. Therefore, the feature vector is of length $2 \times 39 \times 39 = 3042$ dimensions. The gradient image vector is projected into a pre-computed feature space, resulting a feature vector of length 36 elements. The vector is then normalized to unit magnitude to reduce the effects of illumination changes. Also, Morel and Yu [55] proved that the SIFT is fully invariant with respect to only four parameters namely zoom, rotation and translation out of the six parameters of the affine transform. Therefore, they introduced affine-SIFT (ASIFT), which simulates all image views obtainable by varying the camera axis orientation parameters, namely, the latitude and the longitude angles, left over by the SIFT descriptor.

4.2 Gradient Location-Orientation Histogram (GLOH)

Gradient location-orientation histogram (GLOH) developed by Mikolajczyk and Schmid [50] is also an extension of the SIFT descriptor. GLOH is very similar to the SIFT descriptor, where it only replaces the Cartesian location grid used by the SIFT with a log-polar one, and applies PCA to reduce the size of the descriptor. GLOH uses a log-polar location grid with 3 bins in radial direction (radius is set to 6, 11, and 15) and 8 in angular direction, resulting in 17 location bins as illustrated in Fig. 9. GLOH descriptor builds a set of histograms using the gradient orientations in 16 bins, resulting in a feature vector of $17 \times 16 = 272$ elements for each interest point. The 272-dimensional descriptor is reduced to 128 dimensional one by computing the covariance matrix for PCA and the highest 128 eigenvectors are selected for description. Based on the experimental evaluation conducted in [50], GLOH has

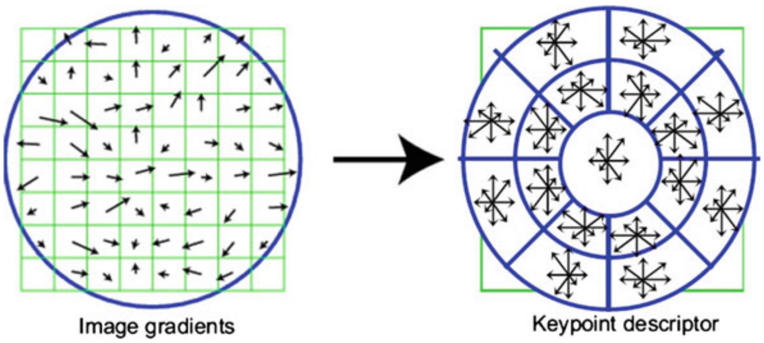

Fig. 9 A schematic representation of the GLOH algorithm using log-polar bins

been reported to outperform the original SIFT descriptor and gives the best performance, especially under illumination changes. Furthermore, it has been shown to be more distinctive but also more expensive to compute than its counterpart SIFT.

4.3 Speeded-Up Robust Features Descriptor (SURF)

The Speeded-Up Robust Features (SURF) detector-descriptor scheme developed by Bay et al. [32] is designed as an efficient alternative to SIFT. It is much faster, and more robust as opposed to SIFT. For the detection stage of interest points, instead of relying on ideal Gaussian derivatives, the computation is based on simple 2D box filters; where, it uses a scale invariant blob detector based on the determinant of Hessian matrix for both scale selection and locations. Its basic idea is to approximate the second order Gaussian derivatives in an efficient way with the help of integral images using a set of box filters. The 9×9 box filters depicted in Fig. 10 are approximations of a Gaussian with $\sigma = 1.2$ and represent the lowest scale for computing the blob response maps. These approximations are denoted by D_{xx}, D_{yy}, and D_{xy}. Thus, the approximated determinant of Hessian can be expressed as

$$\det(H_{approx}) = D_{xx}D_{yy} - (wD_{xy})^2 \tag{23}$$

where w is a relative weight for the filter response and it is used to balance the expression for the Hessian's determinant. The approximated determinant of the Hessian represents the blob response in the image. These responses are stored in a blob response map, and local maxima are detected and refined using quadratic interpolation, as with DoG. Finally, do non-maximum suppression in a $3 \times 3 \times 3$ neighborhood to get steady interest points and the scale of values.

The SURF descriptor starts by constructing a square region centered around the detected interest point, and oriented along its main orientation. The size of this window is $20s$, where s is the scale at which the interest point is detected. Then, the interest region is further divided into smaller 4×4 sub-regions and for each sub-region the Harr wavelet responses in the vertical and horizontal directions (denoted

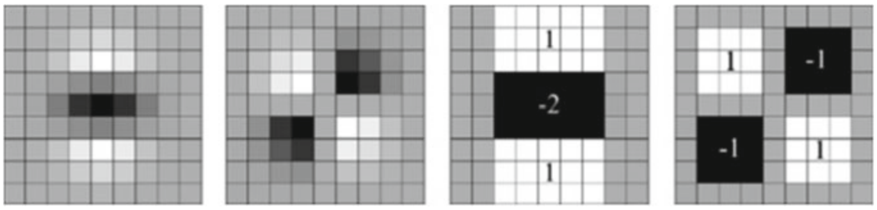

Fig. 10 *Left* to *right* Gaussian second order derivatives in y- (D_{yy}), xy-direction (D_{xy}) and their approximations in the same directions, respectively

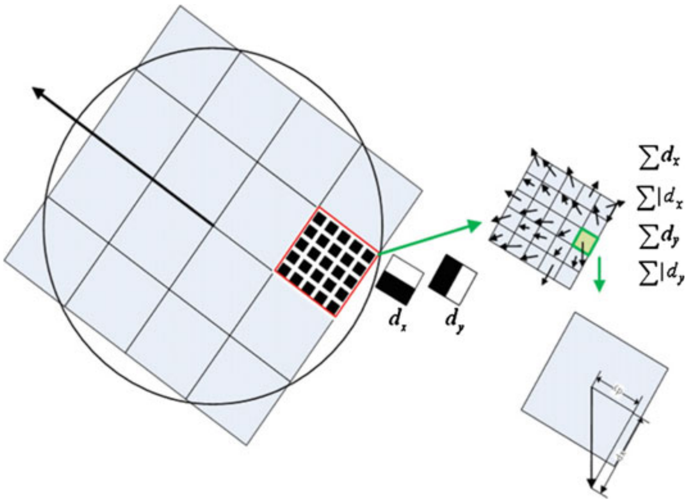

Fig. 11 Dividing the interest region into 4 × 4 sub-regions for computing the SURF descriptor

d_x and d_y, respectively) are computed at a 5 × 5 sampled points as shown in Fig. 11. These responses are weighted with a Gaussian window centered at the interest point to increase the robustness against geometric deformations and localization errors. The wavelet responses d_x and d_y are summed up for each sub-region and entered in a feature vector v, where

$$v = \left(\sum d_x, \sum |d_x|, \sum d_y, \sum |d_y|\right) \tag{24}$$

Computing this for all the 4 × 4 sub-regions, resulting a feature descriptor of length $4 \times 4 \times 4 = 64$ dimensions. Finally, the feature descriptor is normalized to a unit vector in order to reduce illumination effects.

The main advantage of the SURF descriptor compared to SIFT is the processing speed as it uses 64 dimensional feature vector to describe the local feature, while SIFT uses 128. However, the SIFT descriptor is more suitable to describe images affected by translation, rotation, scaling, and other illumination deformations. Though SURF shows its potential in a wide range of computer vision applications, it also has some shortcomings. When 2D or 3D objects are compared, it does not work if rotation is violent or the view angle is too different. Also, the SURF is not fully affine invariant as explained in [56].

4.4 *Local Binary Pattern (LBP)*

Local Binary Patterns (LBP) [57, 58] characterizes the spatial structure of a texture and presents the characteristics of being invariant to monotonic transformations of the gray-levels. It encodes the ordering relationship by comparing neighboring pixels with the center pixel, that is, it creates an order based feature for each pixel by comparing each pixel's intensity value with that of its neighboring pixels. Specifically, the neighbors whose feature responses exceed the central's one are labeled as '1' while the others are labeled as '0'. The co-occurrence of the comparison results is subsequently recorded by a string of binary bits. Afterwards, weights coming from a geometric sequence which has a common ratio of 2 are assigned to the bits according to their indices in strings. The binary string with its weighted bits is consequently transformed into a decimal valued index (i.e., the LBP feature response). That is, the descriptor describes the result over the neighborhood as a binary number (binary pattern). On its standard version, a pixel c with intensity $g(c)$ is labeled as

$$S(g_p - g_c) = \begin{cases} 1, & \textit{if} \;\; g_p \geq g_c \\ 0, & \textit{if} \;\; g_p < g_c \end{cases} \tag{25}$$

where pixels p belong to a 3×3 neighborhood with gray levels $g_p (p = 0, 1, \ldots, 7)$. Then, the LBP pattern of the pixel neighborhood is computed by summing the corresponding thresholded values $S(g_p - g_c)$ weighted by a binomial factor of 2^k as

$$LBP = \sum_{k=0}^{7} S(g_p - g_c).2^k \tag{26}$$

After computing the labeling for each pixel of the image, a 256-bin histogram of the resulting labels is used as a feature descriptor for the texture. An illustration example for computing LBP of a pixel in a 3×3 neighborhood and an orientation descriptor of a basic region in an image is shown in Fig. 12. Also, the LBP descriptor is calculated in its general form as follows

$$LBP_{RN}(x, y) = \sum_{i=0}^{N-1} S(n_i - n_c).2^i, \quad S(x) = \begin{cases} 1, & x \geq 0, \\ 0, & \textit{otherwise} \end{cases} \tag{27}$$

where n_c corresponds to the gray level of the center pixel of a local neighborhood and n_i is the gray levels of N equally spaced pixels on a circle of radius R. Since correlation between pixels decreases with the distance, a lot of the texture information can be obtained from local neighborhoods. Thus, the radius R is usually kept small. In practice, the signs of the differences in a neighborhood are interpreted as a N-bit binary number, resulting in 2^N distinct values for the binary pattern as shown in Fig. 13. The binary patterns are called uniform patterns, where they contain at most two bitwise transitions from 0 to 1. For instance, "11000011" and "00001110" are two uniform patterns, while "00100100" and "01001110" are non-uniform patterns.

Several variations of LBP have been proposed, including the center-symmetric local binary patterns (CS-LBP), the local ternary pattern (LTP), the center-symmetric local ternary pattern (CS-LTP) based on the CS-LBP, and orthogonal symmetric local

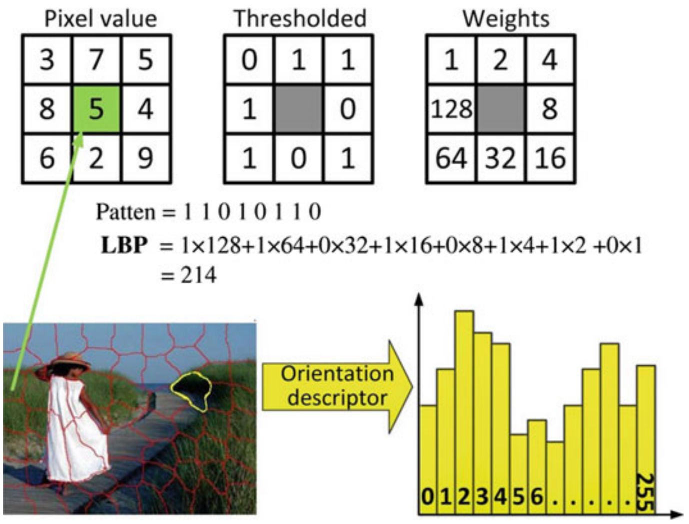

Fig. 12 Computing LBP descriptor for a pixel in a 3 × 3 neighborhood [59] © 2014 IEEE

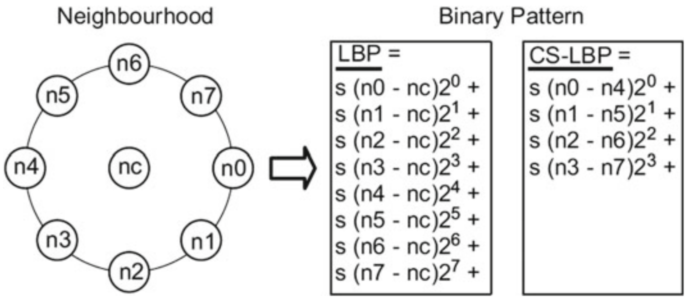

Fig. 13 LBP and CS-LBP features for a neighborhood of 8 pixels [58] © 2009 Elsevier

ternary pattern (OS-LTP) [60]. Unlike the LBP, the CS-LBP descriptor compares gray-level differences of center-symmetric pairs (see Fig. 13). In fact, the LBP has the advantage of tolerance of illumination changes and computational simplicity. Also, the LBP and its variants achieve great success in texture description. Unfortunately, the LBP feature is an index of discrete patterns rather than a numerical feature, thus it is difficult to combine the LBP features with other discriminative ones in a compact descriptor [61]. Moreover, it produces higher dimensional features and is sensitive to Gaussian noise on flat regions.

4.5 Binary Robust Independent Elementary Features (BRIEF)

Binary robust independent elementary features (BRIEF), a low-bitrate descriptor, is introduced for image matching with random forest and random ferns classifiers

[62]. It belongs to the family of binary descriptors such as LBP and BRISK, which only performs simple binary comparison test and uses Hamming distance instead of Euclidean or Mahalanobis distance. Briefly, for building a binary descriptor, it is only necessary to compare the intensity between two pixel positions located around the detected interest points. This allows to obtain a representative description at very low computational cost. Besides, matching the binary descriptors requires only the computation of Hamming distances that can be executed very fast through XOR primitives on modern architectures.

The BRIEF algorithm relies on a relatively small number of intensity difference tests to represent an image patch as a binary string. More specifically, a binary descriptor for a patch of pixels of size $S \times S$ is built by concatenating the results of the following test

$$\tau = \begin{cases} 1, & \text{if } I(P_j) > I(P_i), \\ 0, & \text{otherwise}, \end{cases} \tag{28}$$

where $I(p_i)$ denotes the (smoothed) pixel intensity value at p_i, and the selection of the location of all the p_i uniquely defines a set of binary tests. The sampling points are drawn from a zero-mean isotropic Gaussian distribution with variance equal to $\frac{1}{25}S^2$. For increasing the robustness of the descriptor, the patch of pixels is pre-smoothed with a Gaussian kernel with variance equal to 2 and size equal to 9×9 pixels. The BRIEF descriptor has two setting parameters: the number of binary pixel pairs and the binary threshold.

The experiments conducted by authors showed that only 256 bits, or even 128 bits, often suffice to obtain very good matching results. Thus, BRIEF is considered to be very efficient both to compute and to store in memory. Unfortunately, BRIEF descriptor is not robust against a rotation larger than 35° approximately, hence, it does not provide rotation invariance.

4.6 Other Feature Descriptors

A large number of other descriptors have been proposed in the literature and many of them have been proved to be effective in computer vision applications. For instance, color-based local features are four color descriptors based on color information proposed by Weijer and Schmid [63]. The Gabor representation or its variation [64, 65] has been also shown to be optimal in the sense of minimizing the joint two-dimensional uncertainty in space and frequency. Zernike moments [66, 67] and Steerable filters [68] are also considered for feature extraction and description.

Inspired by Weber's Law, a dense descriptor computed for every pixel depending on both the local intensity variation and the magnitude of the center pixel's intensity called Weber Local Descriptor (WLD) is proposed in [28]. The WLD descriptor employs the advantages of SIFT in computing the histogram using the gradient and its orientation, and those of LBP in computational efficiency and smaller support regions. In contrast to the LBP descriptor, WLD first computes the salient micro-patterns (i.e., differential excitation), and then builds statistics on these salient patterns along with the gradient orientation of the current point.

Two methods for extracting distinctive features from interest regions based on measuring the similarity between visual entities from images are presented in [69]. The idea of these methods combines the powers of two well-known approaches, the SIFT descriptor and Local Self-Similarities (LSS). Two texture features called Local Self-Similarities (LSS, C) and Fast Local Self-Similarities (FLSS, C) based on Cartesian location grid, are extracted, which are the modified versions of the Local Self-Similarities feature based on Log-Polar location grid (LSS, LP). The LSS and FLSS features are used as the local features in the SIFT algorithm. The proposed LSS and FLSS descriptors use distribution-based histogram representation in each cell rather than choosing the maximal correlation value in each bucket in the Log-Polar location grid in the natural (LSS, LP) descriptor. Thus, they get more robust geometric transformations invariance and good photometric transformations invariance. A local image descriptor based on Histograms of the Second Order Gradients, namely HSOG is introduced in [70] for capturing the curvature related geometric properties of the neural landscape. Dalal and Triggs [71] presented the Histogram of Oriented Gradient (HOG) descriptor, which combines both the properties of SIFT and GLOH descriptors. The main difference between HOG and SIFT is that the HOG descriptor is computed on a dense grid of uniformly spaced cells with overlapping local contrast normalization.

Following a different direction, Fan et al. [72] proposed a method for interest region description, which pools local features based on their intensity orders in multiple support regions. Pooling by intensity orders is not only invariant to rotation and monotonic intensity changes, but also encodes ordinal information into a descriptor. By pooling two different kinds of local features, one based on gradients and the other on intensities, two descriptors are obtained: the Multisupport Region Order-Based Gradient Histogram (MROGH) and the Multisupport Region Rotation and Intensity Monotonic Invariant Descriptor (MRRID). The former combines information of intensity orders and gradient, while the latter is completely based on intensity orders, which makes it particularly suitable to large illumination changes. Several image features are analyzed

In spite of the fact that, a large number of image feature descriptors have been introduced recently, several of these descriptors are exclusively designed for a specific application scenario such as object recognition, shape retrieval, or LADAR data processing [73]. Furthermore, the authors of these descriptors evaluated their performance on a limited number of benchmarking datasets collected specifically for particular tasks. Consequently, it is very challenging for researchers to choose an appropriate descriptor for their particular application. In this respect, some recent studies compare the performance of several descriptors; interest region descriptors [50], binary descriptors [74], local colour descriptors [75], and the 3D descriptors [76, 77]. In fact, claims that describing image features is a solved problem are overly bold and optimistic. On the other hand, claims that designing a descriptor for general real-world scenarios is next to impossible are simply too pessimistic, given the success of the aforementioned descriptors in several applications. Finally, there is much work to be done in order to realize description algorithms that can be used for general applications. We argue for further research towards using new modalities captured by 3D data and color information. For real time applications, a further path of future research would be the implementation of the algorithms on parallel processing units such as GPU.

5 Features Matching

Features matching or generally image matching, a part of many computer vision applications such as image registration, camera calibration and object recognition, is the task of establishing correspondences between two images of the same scene/object. A common approach to image matching consists of detecting a set of interest points each associated with image descriptors from image data. Once the features and their descriptors have been extracted from two or more images, the next step is to establish some preliminary feature matches between these images as illustrated in Fig. 14.

Without losing the generality, the problem of image matching can be formulated as follows, suppose that p is a point detected by a detector in an image associated with a descriptor

$$\Phi(p) = \{\phi_k(P) \mid k = 1, 2, \ldots, K\} \tag{29}$$

where, for all K, the feature vector provided by the k-th descriptor is

$$\phi_k(p) = (f^k_{1p}, f^k_{2p}, \ldots, f^k_{n_k p}) \tag{30}$$

The aim is to find the best correspondence q in another image from the set of N interest points $Q = \{q_1, q_2, \ldots, q_N\}$ by comparing the feature vector $\phi_k(p)$ with those of the points in the set Q. To this end, a distance measure between the two interest points descriptors $\phi_k(p)$ and $\phi_k(q)$ can be defined as

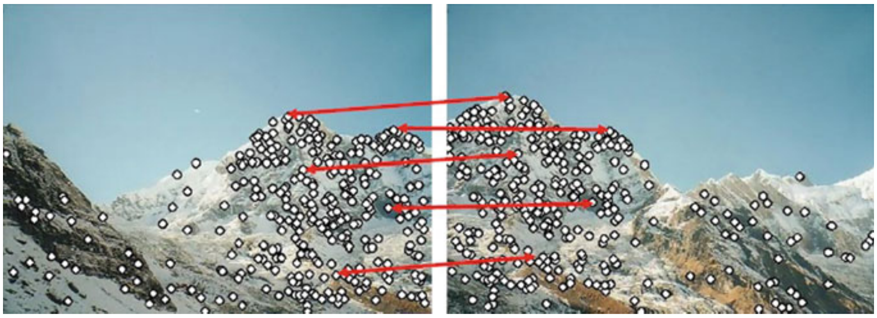

Fig. 14 Matching image regions based on their local feature descriptors [79] © 2011 Springer

$$d_k(p,q) = |\phi_k(p) - \phi_k(q)| \tag{31}$$

Based on the distance d_k, the points of Q are sorted in ascending order independently for each descriptor creating the sets

$$\Psi(p,Q) = \{\psi_k(p,Q) \mid k = 1,2,\ldots,k\} \tag{32}$$

Such that,

$$\psi_k(p,Q) = \left\{(\psi_k^1, \psi_k^2, \ldots, \psi_k^N) \in Q \mid d_k(p,\psi_k^i) \leq d_k(p,\psi_k^j), \forall i > j\right\} \tag{33}$$

A match between the pair of interest points (p,q) is accepted only if (i) p is the best match for q in relation to all the other points in the first image and (ii) q is the best match for p in relation to all the other points in the second image. In this context, it is very important to devise an efficient algorithm to perform this matching process as quickly as possible. The nearest-neighbor matching in the feature space of the image descriptors in Euclidean norm can be used for matching vector-based features. However, in practice, the optimal nearest neighbor algorithm and its parameters depend on the data set characteristics. Furthermore, to suppress matching candidates for which the correspondence may be regarded as ambiguous, the ratio between the distances to the nearest and the next nearest image descriptor is required to be less than some threshold. As a special case, for matching high dimensional features, two algorithms have been found to be the most efficient: the randomized k-d forest and the fast library for approximate nearest neighbors (FLANN) [78].

On the other hand, these algorithms are not suitable for binary features (e.g., FREAK or BRISK). Binary features are compared using the Hamming distance calculated via performing a bitwise XOR operation followed by a bit count on the result. This involves only bit manipulation operations that can be performed quickly. The typical solution in the case of matching large datasets is to replace the linear search with an approximate matching algorithm that can offer speedups of several orders of magnitude over the linear search. This is, at the cost that some of the nearest neighbors returned are approximate neighbors, but usually close in distance to the exact neighbors. For performing matching of binary features, other methods can be employed such as [80–82].

Generally, the performance of matching methods based on interest points depends on both the properties of the underlying interest points and the choice of associated image descriptors [83]. Thus, detectors and descriptors appropriate for images contents shall be used in applications. For instance, if an image contains bacteria cells, the blob detector should be used rather than the corner detector. But, if the image is an aerial view of a city, the corner detector is suitable to find man-made structures. Furthermore, selecting a detector and a descriptor that addresses the image degradation is very important. For example, if there is no scale change present, a corner detector that does not handle scale is highly desirable; while, if image contains a higher level of distortion, such as scale and rotation, the more computationally intensive SURF feature detector and descriptor is a adequate choice in that case. For greater accuracy, it is recommended to use several detectors and descriptors at the same time. In the area of feature matching, it must be noticed that the binary descriptors (e.g., FREAK or BRISK) are generally faster and typically used for finding point correspondences between images, but they are less accurate than vector-based descriptors [74]. Statistically robust methods like RANSAC can be used to filter outliers in matched feature sets while estimating the geometric transformation or fundamental matrix, which is useful in feature matching for image registration and object recognition applications.

6 Performance Evaluation

6.1 Benchmarking Data Sets

There are a wide variety of data sets available on the Internet that can be used as a benchmark by researchers. One popular and widely used for performance evaluation of detectors and descriptors is the standard Oxford data set [84]. The dataset consists of image sets with different geometric and photometric transformations (viewpoint change, scale change, image rotation, image blur, illumination change, and JPEG compression) and with different scene types (structured and textured scenes). In the cases of illumination change, the light changes are introduced by varying the camera aperture. While in the case of rotation, scale change, viewpoint change, and blur, two different scene types are used. One scene type contains structured scenes which are homogeneous regions with distinctive edge boundaries (e.g., graffiti, buildings), and the other contains repeated textures of different forms. In this way, the influence of image transformation and scene type can be analyzed separately. Each image set contains 6 images with a gradual geometric or photometric distortion where the first image and the remaining 5 images are compared. Sample images from the Oxford data set are shown in Fig. 15.

6.2 Evaluation Criterion

To judge whether two image features are matched (i.e., belonging to the same feature or not), Mikolajczyk et al. [44] proposed an evaluation procedure based on the repeatability criterion by comparing the ground truth transformation and the detected region overlap. The repeatability can be considered as one of the most important criteria used for evaluating the stability of feature detectors. It measures the ability of a detector to extract the same feature points across images irrespective of imaging conditions. The repeatability criterion measures how well the detector determines corresponding scene regions. In this evaluation procedure, two regions of interest **A** and **B** are deemed to correspond if the overlap error ε is sufficiently small as shown

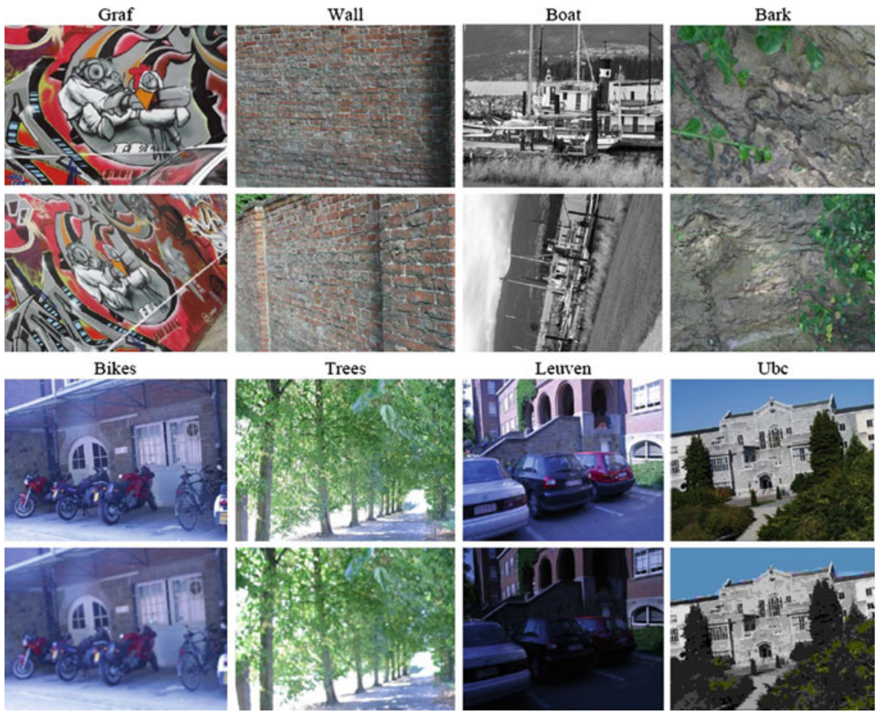

Fig. 15 Test images *Graf* (viewpoint change, structured scene), *Wall* (viewpoint change, textured scene), *Boat* (scale change + image rotation, structured scene), *Bark* (scale change + image rotation, textured scene), *Bikes* (image blur, structured scene), *Trees* (image blur, textured scene), *Leuven* (illumination change, structured scene), and *Ubc* (JPEG compression, structured scene)

in Fig. 16. This overlap error measures how well the regions correspond under a homography transformation **H**. It is defined by the ratio of the intersection and union of the two regions, that is the error in the image area covered by the two regions,

$$\varepsilon = 1 - \frac{\mathbf{A} \cap (\mathbf{H}^{\mathbf{T}} \ \mathbf{B} \ \mathbf{H})}{\mathbf{A} \cup (\mathbf{H}^{\mathbf{T}} \ \mathbf{B} \ \mathbf{H})} \tag{34}$$

This approach counts the total number of pixels in the union and the intersection of the two regions. Also, a match is correct if the error in the image area covered by two corresponding regions is less than 50 % of the region union, that is, $\varepsilon < 0.5$. The overlap error is computed numerically based on homography **H** and the matrices defining the regions. Thus, to evaluate feature detectors performance, the repeatability score for a given pair of images is computed as the ratio between the number of region to region correspondences and the smaller number of regions in the pair of images.

On the other hand, the performance of a region descriptor is measured by the matching criterion, i.e., how well the descriptor represents a scene region. It is based

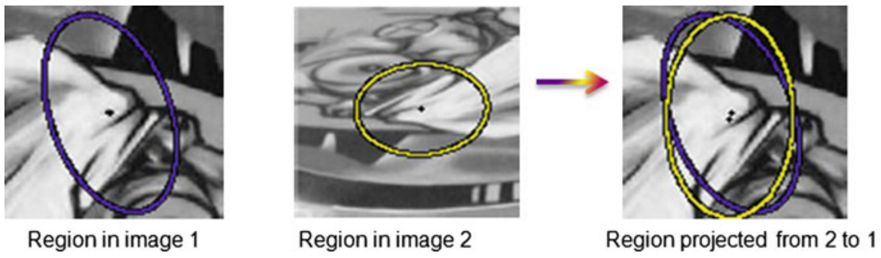

Fig. 16 Illustration of overlap error for a region projected onto the corresponding region (*ellipse*)

on the number of correct matches and the number of false matches obtained for the image pair. This is measured by comparing the number of corresponding regions obtained with the ground truth and the number of correctly matched regions. Matches are the nearest neighbors in the descriptor space [50]. In this case, the two regions of interest are matched if the Euclidean distance between their descriptors D_A and D_B is below a threshold τ. The results are presented with *recall* versus *1-precision*. Each descriptor from the reference image is compared with each descriptor from the transformed one and counting the number of correct matches as well as the number of false matches.

$$recall = \frac{No.\ \ correct\ \ matches}{Total\ \ No.\ \ correspondences}, \tag{35}$$

$$1 - precision = \frac{No.\ \ false\ \ matches}{Total\ \ No.\ \ all\ \ matches} \tag{36}$$

where, *No. correspondences* refers to the number of matching regions between image pairs. While, *recall* is the number of correctly matched regions with respect to the number of corresponding regions between two images of the same scene. An ideal descriptor gives a *recall* equal to 1 for any *precision* value. In order to obtain the curve, the value of τ is varied. Practically, *recall* increases for an increasing distance threshold τ because noise introduced by image transformations and region detection increases the distance between similar descriptors. A slowly increasing curve indicates that the descriptor is affected by the image noise. If the obtained curves corresponding to different descriptors are far apart or have different slopes, then the distinctiveness and robustness of these descriptors are different for the investigated image transformation or scene type [50].

7 Conclusions

The objective of this chapter is to provide a straight-forward, brief introduction for new researchers to the image feature detection and extraction research field. It introduces the basic notations and mathematical concepts for detecting and extracting image features, then describes the properties of perfect feature detectors. Various existing algorithms for detecting interest points are discussed briefly. The most frequently used description algorithms such as SIFT, SURF, LBP, WLD,…etc are also discussed and their advantages/disadvantages are highlighted. Furthermore, it explains some approaches to feature matching. Finally, the chapter discusses the techniques used for evaluating the performance of these algorithms.

References

1. Yap, T., Jiang, X., Kot, A.C.: Two-dimensional polar harmonic transforms for invariant image representation. IEEE Trans. Pattern Anal. Mach. Intell. **32**(7), 1259–1270 (2010)
2. Liu, S., Bai, X.: Discriminative features for image classification and retrieval. Pattern Recogn. Lett. **33**(6), 744–751 (2012)
3. Rahmani, R., Goldman, S., Zhang, H., Cholleti, S., Fritts, J.: Localized content-based image retrieval. IEEE Trans. Pattern Anal. Mach. Intell. **30**(11), 1902–1912 (2008)
4. Stöttinger, J., Hanbury, A., Sebe, N., Gevers, T.: Sparse color interest points for image retrieval and object categorization. IEEE Trans. Image Process. **21**(5), 2681–2691 (2012)
5. Wang, J., Li, Y., Zhang, Y., Wang, C., Xie, H., Chen, G., Gao, X.: Bag-of-features based medical image retrieval via multiple assignment and visual words weighting. IEEE Trans. Med. Imaging **30**(11), 1996–2011 (2011)
6. Andreopoulos, A., Tsotsos, J.: 50 years of object recognition: directions forward. Comput. Vis. Image Underst. **117**(8), 827–891 (2013)
7. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: an evaluation of the state of the art. IEEE Trans. Pattern Anal. Mach. Intell. **34**(4), 743–761 (2012)
8. Felsberg, M., Larsson, F., Wiklund, J., Wadströmer, N., Ahlberg, J.: Online learning of correspondences between images. IEEE Trans. Pattern Anal. Mach. Intell. **35**(1), 118–129 (2013)
9. Miksik, O., Mikolajczyk, K.: Evaluation of local detectors and descriptors for fast feature matching. In: International Conference on Pattern Recognition (ICPR 2012), pp. 2681–2684. Tsukuba, Japan, 11–15 Nov 2012
10. Kim, B., Yoo, H., Sohn, K.: Exact order based feature descriptor for illumination robust image matching. Pattern Recogn. **46**(12), 3268–3278 (2013)
11. Moreels, P., Perona, P.: Evaluation of features detectors and descriptors based on 3D objects. Int. J. Comput. Vis. **73**(3), 263–284 (2007)
12. Takacs, G., Chandrasekhar, V., Tsai, S., Chen, D., Grzeszczuk, R., Girod, B.: Rotation-invariant fast features for large-scale recognition and real-time tracking. Sign. Process. Image Commun. **28**(4), 334–344 (2013)
13. Tang, S., Andriluka, M., Schiele, B.: Detection and tracking of occluded people. Int. J. Comput. Vis. **110**(1), 58–69 (2014)
14. Rincón, J.M., Makris, D., Uruňuela, C., Nebel, J.C.: Tracking human position and lower body parts using Kalman and particle filters constrained by human biomechanics. IEEE Trans. Syst. Man Cybern. Part B **41**(1), 26–37 (2011)
15. Lazebnik, S., Schmid, C., Ponce, J.: A sparse texture representation using local affine regions. IEEE Trans. Pattern Anal. Mach. Intell. **27**(8), 1265–1278 (2005)

16. Liu, L., Fieguth, P.: Texture classification from random features. IEEE Trans. Pattern Anal. Mach. Intell. **34**(3), 574–586 (2012)
17. Murillo, A., Guerrero, J., Sagues, C.: SURF features for efficient robot localization with omnidirectional images. In: International Conference on Robotics and Automation, pp. 3901–3907. Rome, Italy, 10–14 Apr, 2007
18. Valgren, C., Lilienthal, A.J.: SIFT, SURF & seasons: appearance-based long-term localization in outdoor environments. Rob. Auton. Syst. **58**(2), 149–156 (2010)
19. Campos, F.M., Correia, L., Calado, J.M.F.: Robot visual localization through local feature fusion: an evaluation of multiple classifiers combination approaches. J. Intell. Rob. Syst. **77**(2), 377–390 (2015)
20. Farajzadeh, N., Faez, K., Pan, G.: Study on the performance of moments as invariant descriptors for practical face recognition systems. IET Comput. Vis. **4**(4), 272–285 (2010)
21. Mian, A., Bennamoun, M., Owens, R.: Keypoint detection and local feature matching for textured 3D face recognition. Int. J. Comput. Vis. **79**(1), 1–12 (2008)
22. Jain, A.K., Ross, A.A., Nandakumar, K.: Introduction to Biometrics, 1st edn. Springer (2011)
23. Burghardt, T., Damen, D., Mayol-Cuevas, W., Mirmehdi, M.: Correspondence, matching and recognition. Int. J. Comput. Vis. **113**(3), 161–162 (2015)
24. Bouchiha, R., Besbes, K.: Comparison of local descriptors for automatic remote sensing image registration. SIViP **9**(2), 463–469 (2015)
25. Zhao, Q., Feng, W., Wan, L., Zhang, J.: SPHORB: a fast and robust binary feature on the sphere. Int. J. Comput. Vis. **113**(2), 143–159 (2015)
26. Zhang, S., Tian, Q., Huang, Q., Gao, W., Rui, Y.: USB: ultrashort binary descriptor for fast visual matching and retrieval. IEEE Trans. Image Process. **23**(8), 3671–3683 (2014)
27. Tuytelaars, T., Mikolajczyk, K.: Local invariant feature detectors: a survey. Found. Trends Comput. Graph. Vis. **3**(3), 177–280 (2007)
28. Chen, J., Shan, S., He, C., Zhao, G., Pietikäinen, M., Chen, X., Gao, W.: WLD: a robust local image descriptor. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1705–1720 (2010)
29. Viola, P., Jones, M.J.: Robust real-time face detection. Int. J. Comput. Vis. **57**(2), 137–154 (2004)
30. Janan, F., Brady, M.: Shape description and matching using integral invariants on eccentricity transformed images. Int. J. Comput. Vis. **113**(2), 92–112 (2015)
31. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
32. Bay, H., Ess, A., Tuytelaars, T., Gool, L.: Speeded-up robust features (SURF). Comput. Vis. Image Underst. **110**(3), 346–359 (2008)
33. Oliva, A., Torralba, A.: Modeling the shape of the scene: a holistic representation of the spatial envelope. Int. J. Comput. Vis. **42**(3), 145–175 (2001)
34. Bianco, S., Mazzini, D., Pau, D., Schettini, R.: Local detectors and compact descriptors for visual search: a quantitative comparison. Digital Signal Process. **44**, 1–13 (2015)
35. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating local descriptors into a compact codes. IEEE Trans. Pattern Anal. Mach. Intell. **34**(9), 1704–1716 (2012)
36. Lindeberg, T.: Feature detection with automatic scale selection. Int. J. Comput. Vis. **30**(2), 79–116 (1998)
37. Moravec, H.P.: Towards automatic visual obstacle avoidance. In: 5th International Joint Conference on Artificial Intelligence, pp. 584–594 (1977)
38. Harris, C., Stephens, M.: A combined corner and edge detector. In: The Fourth Alvey Vision Conference, pp. 147–151. Manchester, UK (1988)
39. Smith, S.M., Brady, J.M.: A new approach to low level image processing. Int. J. Comput. Vis. **23**(1), 45–78 (1997)
40. Rosten, E., Drummond, T.: Fusing points and lines for high performance tracking. In: International Conference on Computer Vision (ICCV'05), pp. 1508–1515. Beijing, China, 17–21 Oct 2005

41. Rosten, E., Drummond, T.: Machine learning for high speed corner detection. In: 9th European Conference on Computer Vision (ECCV'06), pp. 430–443. Graz, Austria, 7–13 May 2006
42. Beaudet, P.R.: Rotationally invariant image operators. In: International Joint Conference on Pattern Recognition, pp. 579–583 (1978)
43. Lakemond, R., Sridharan, S., Fookes, C.: Hessian-based affine adaptation of salient local image features. J. Math. Imaging Vis. **44**(2), 150–167 (2012)
44. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.: A comparison of affine region detectors. Int. J. Comput. Vis. **65**(1/2), 43–72 (2005)
45. Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. Int. J. Comput. Vis. **60**(1), 63–86 (2004)
46. Lindeberg, T.: Scale selection properties of generalized scale-space interest point detectors. J. Math. Imaging Vis. **46**(2), 177–210 (2013)
47. Yussof, W., Hitam, M.: Invariant Gabor-based interest points detector under geometric transformation. Digital Signal Process. **25**, 190–197 (2014)
48. Schaffalitzky, F., Zisserman, A.: Multi-view matching for unordered image sets. In: European Conference on Computer Vision (ECCV), pp. 414–431. Copenhagen, Denmark, 28–31 May 2002
49. Lindeberg, T., Gårding, J.: Shape-adapted smoothing in estimation of 3-D shape cues from affine deformations of local 2-D brightness structure. Image Vis. Comput. **15**(6), 415–434 (1997)
50. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. IEEE Trans. Pattern Anal. Mach. Intell. **27**(10), 1615–1630 (2005)
51. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide baseline stereo from maximally stable extremal regions. In. In British Machine Vision Conference (BMV), pp. 384–393 (2002)
52. Matas, J., Ondrej, C., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image Vis. Comput. **22**(10), 761–767 (2004)
53. Li, J., Allinson, N.: A comprehensive review of current local features for computer vision. Neurocomputing **71**(10–12), 1771–1787 (2008)
54. Ke, Y., Sukthankar, R.: PCA-SIFT: a more distinctive representation for local image descriptors. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04), pp. 506–513. Washington, DC, USA, 27 June–2 July 2004
55. Morel, J., Yu, G.: ASIFT: a new framework for fully affine invariant image comparison. SIAM J. Imaging Sci. **2**(2), 438–469 (2009)
56. Pang, Y., Li, W., Yuan, Y., Pan, J.: Fully affine invariant SURF for image matching. Neurocomputing **85**, 6–10 (2012)
57. Ojala, T., Pietikäinen, M., Mäenpää, M.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 971–987 (2002)
58. Heikkiläa, M., Pietikäinen, M., Schmid, C.: Description of interest regions with local binary patterns. Pattern Recogn. **42**(3), 425–436 (2009)
59. Tian, H., Fang, Y., Zhao, Y., Lin, W., Ni, R., Zhu, Z.: Salient region detection by fusing bottom-up and top-down features extracted from a single image. IEEE Trans. Image Process. **23**(10), 4389–4398 (2014)
60. Huang, M., Mu, Z., Zeng, H., Huang, S.: Local image region description using orthogonal symmetric local ternary pattern. Pattern Recogn. Lett. **54**(1), 56–62 (2015)
61. Hong, X., Zhao, G., Pietikäinen, M., Chen, X.: Combining LBP difference and feature correlation for texture description. IEEE Trans. Image Process. **23**(6), 2557–2568 (2014)
62. Calonder, M., Lepetit, V., Özuysal, M., Trzcinski, T., Strecha, C., Fua, P.: BRIEF: computing a local binary descriptor very fast. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1281–1298 (2012)
63. Van De Weijer, J., Schmid, C.: Coloring local feature extraction. In: European Conference on Computer Vision (ECCV), pp. 334–348. Graz, Austria, 7–13 May 2006
64. Subrahmanyam, M., Gonde, A.B., Maheshwari, R.P.: Color and texture features for image indexing and retrieval. In: IEEE International Advance Computing Conference (IACC), pp. 1411–1416. Patiala, India, 6–7 Mar 2009

65. Zhang, Y., Tian, T., Tian, J., Gong, J., Ming, D.: A novel biologically inspired local feature descriptor. Biol. Cybern. **108**(3), 275–290 (2014)
66. Chen, Z., Sun, S.: A Zernike moment phase-based descriptor for local image representation and matching. IEEE Trans. Image Process. **19**(1), 205–219 (2010)
67. Chen, B., Shu, H., Zhang, H., Coatrieux, G., Luo, L., Coatrieux, J.: Combined invariants to similarity transformation and to blur using orthogonal Zernike moments. IEEE Trans. Image Process. **20**(2), 345–360 (2011)
68. Freeman, W., Adelson, E.: The design and use of steerable filters. IEEE Trans. Pattern Anal. Mach. Intell. **13**(9), 891–906 (1991)
69. Liu, J., Zeng, G., Fan, J.: Fast local self-similarity for describing interest regions. Pattern Recogn. Lett. **33**(9), 1224–1235 (2012)
70. Huang, D., Chao, Z., Yunhong, W., Liming, C.: HSOG: a novel local image descriptor based on histograms of the second-order gradients. IEEE Trans. Image Process. **23**(11), 4680–4695 (2014)
71. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05), pp. 886–893. San Diego, CA, USA, 20–26 June 2005
72. Fan, B., Wu, F., Hu, Z.: Rotationally invariant descriptors using intensity order pooling. IEEE Trans. Pattern Anal. Mach. Intell. **34**(10), 2031–2045 (2012)
73. Al-Temeemy, A.A., Spencer, J.W.: Invariant chromatic descriptor for LADAR data processing. Mach. Vis. Appl. **26**(5), 649–660 (2015)
74. Figat, J., Kornuta, T., Kasprzak, W.: Performance evaluation of binary descriptors of local features. Lect. Notes Comput. Sci. (LNCS) **8671**, 187–194 (2014)
75. Burghouts, G., Geusebroek, J.M.: Performance evaluation of local color invariantss. Comput. Vis. Image Underst. **113**(1), 48–62 (2009)
76. Moreels, P., Perona, P.: Evaluation of features detectors and descriptors based on 3D objects. Int. J. Comput. Vis. **73**(2), 263–284 (2007)
77. Guo, Y., Bennamoun, M., Sohel, F., Lu, M., Wan, J., Kwok, N.M.: A comprehensive performance evaluation of 3D local feature descriptors. Int. J. Comput. Vis. First online, 1–24 (2015)
78. Muja, M., Lowe, D.G.: Scalable nearest neighbor algorithms for high dimensional data. IEEE Trans. Pattern Anal. Mach. Intell. **36**(11), 2227–2240 (2014)
79. Szeliski, R.: Computer Vision: Algorithms and Applications. Springer, USA (2011)
80. Muja, M., David G. Lowe: Fast matching of binary features. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12), pp. 404–410. Toronto, ON, USA, 28–30 May 2012
81. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06), pp. 2161–2168. Washington, DC, USA, 17–22 June 2006
82. Yan, C.C., Xie, H., Zhang, B., Ma, Y., Dai, Q., Liu, Y.: Fast approximate matching of binary codes with distinctive bits. Frontiers Comput. Sci. **9**(5), 741–750 (2015)
83. Lindeberg, T.: Image matching using generalized scale-space interest points. J. Math. Imaging Vis. **52**(1), 3–36 (2015)
84. The oxford data set is available at (last visit, Oct. 2015) http://www.robots.ox.ac.uk/~vgg/data/data-aff.html