CHAPTER 3

Local Features: Detection and Description

Significant progress towards robustly recognizing objects has been made in the past decade through the development of *local invariant features*. These features allow the algorithm to find local image structures in a repeatable fashion and to encode them in a representation that is invariant to a range of image transformations, such as translation, rotation, scaling, and affine deformation. The resulting features then form the basis of approaches for recognizing both specific objects and object categories.

In this chapter, we will explain the basic ideas and implementation steps behind state-of-the-art local feature detectors and descriptors. A more extensive treatment of local features, including detailed comparisons and usage guidelines, can be found in [Tuytelaars and Mikolajczyk, 2007]. Systematic experimental comparisons are reported in [Mikolajczyk and Schmid, 2005, Mikolajczyk et al., 2005].

3.1 INTRODUCTION

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 that encode their *interesting* structure. To meet this goal, the feature extractors must fulfill two important criteria:

- The feature extraction process should be *repeatable and precise*, so that the same features are extracted from two images showing the same object.

- At the same time, the features should be *distinctive*, so that different image structures can be told apart from each other.

In addition, we typically require a sufficient number of feature regions to cover the target object, so that it can still be recognized under partial occlusion. This is achieved by the following feature extraction pipeline, illustrated in Figure 3.1:

1. Find a set of *distinctive keypoints*.

2. Define a region around each keypoint in a *scale- or affine-invariant* manner.

3. Extract and *normalize* the region content.

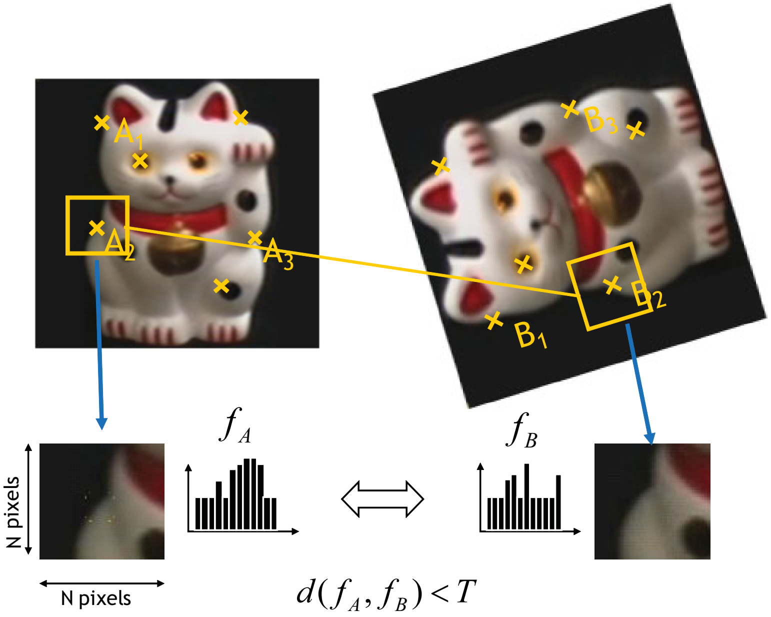

Figure 3.1: An illustration of the recognition procedure with local features. We first find distinctive keypoints in both images. For each such keypoint, we then define a surrounding region in a scale- and rotation-invariant manner. We extract and normalize the region content and compute a local descriptor for each region. Feature matching is then performed by comparing the local descriptors using a suitable similarity measure. Courtesy of Krystian Mikolajczyk.

4. Compute a *descriptor* from the normalized region.
5. Match the local descriptors.

In the remainder of this chapter, we will discuss the keypoint and descriptor steps in detail. Then in the following chapter, we will describe methods for computing candidate matches among the descriptors. Once we have candidate matches, we can then proceed to verify their geometric relationships, as we will describe in Chapter 5.

3.2 DETECTION OF INTEREST POINTS AND REGIONS

3.2.1 KEYPOINT LOCALIZATION

The first step of the local feature extraction pipeline is to find a set of distinctive keypoints that can be reliably localized under varying imaging conditions, viewpoint changes, and in the presence of noise. In particular, the extraction procedure should yield the same feature locations if the input image is translated or rotated. It is obvious that those criteria cannot be met for all image points. For instance, if we consider a point lying in a uniform region, we cannot determine its exact motion, since we cannot distinguish the point from its neighbors. Similarly, if we consider a point on a straight line, we can only measure its motion perpendicular to the line. This motivates us to focus on a particular subset of points, namely those exhibiting signal changes in two directions. In the following, we will present two keypoint detectors that employ different criteria for finding such regions: the *Hessian detector* and the *Harris detector*.

3.2.1.1 The Hessian Detector

The *Hessian detector* [Beaudet, 1978] searches for image locations that exhibit strong derivatives in two orthogonal directions. It is based on the matrix of second derivatives, the so-called Hessian. As derivative operations are sensitive to noise, we always use *Gaussian derivatives* in the following, *i.e.*, we combine the derivative operation with a Gaussian smoothing step with smoothing parameter σ.

$$\mathbf{H}(\mathbf{x}, \sigma) \; = \; \begin{bmatrix} I_{xx}(\mathbf{x}, \sigma) & I_{xy}(\mathbf{x}, \sigma) \\ I_{xy}(\mathbf{x}, \sigma) & I_{yy}(\mathbf{x}, \sigma) \end{bmatrix}. \tag{3.1}$$

The detector computes the second derivatives I_{xx}, I_{xy}, and I_{yy} for each image point and then searches for points where the determinant of the Hessian becomes maximal:

$$\det(\mathbf{H}) = I_{xx} I_{yy} - I_{xy}^2. \tag{3.2}$$

This search is usually performed by computing a result image containing the Hessian determinant values and then applying *non-maximum suppression* using a 3 × 3 window. In this procedure, the search window is swept over the entire image, keeping only pixels whose value is larger than the values of all 8 immediate neighbors inside the window. The detector then returns all remaining locations whose value is above a pre-defined threshold θ. As shown in Figure 3.7(top left), the resulting detector responses are mainly located on corners and in strongly textured image areas.

3.2.1.2 The Harris Detector

The popular *Harris/Förstner detector* [Förstner and Gülch, 1987, Harris and Stephens, 1988] was explicitly designed for geometric stability. It defines keypoints to be "points that have locally maximal self-matching precision under translational least-squares template matching" [Triggs, 2004]. In practice, these keypoints often correspond to corner-like structures. The detection procedure is visualized in Figure 3.2.

The Harris detector proceeds by searching for points $\mathbf{x}$ where the second-moment matrix $\mathbf{C}$ around $\mathbf{x}$ has two large eigenvalues. The matrix $\mathbf{C}$ can be computed from the first derivatives in a window around $\mathbf{x}$, weighted by a Gaussian $G(\mathbf{x}, \tilde{\sigma})$:

$$\mathbf{C}(\mathbf{x}, \sigma, \tilde{\sigma}) = G(\mathbf{x}, \tilde{\sigma}) \star \begin{bmatrix} I_x^2(\mathbf{x}, \sigma) & I_x I_y(\mathbf{x}, \sigma) \\ I_x I_y(\mathbf{x}, \sigma) & I_y^2(\mathbf{x}, \sigma) \end{bmatrix}. \tag{3.3}$$

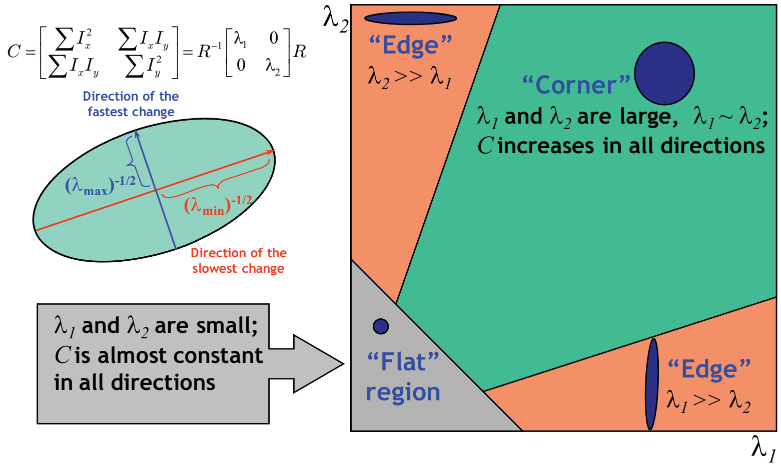
$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$
Direction of the fastest change
$(\lambda_{\text{max}})^{-1/2}$
$(\lambda_{\text{min}})^{-1/2}$
Direction of the slowest change
λ_1 and λ_2 are small; C is almost constant in all directions
λ_2
"Edge" $\lambda_2 >> \lambda_1$
"Corner" λ_1 and λ_2 are large, $\lambda_1 \sim \lambda_2$; C increases in all directions
"Flat" region
"Edge" $\lambda_1 >> \lambda_2$
λ_1

Figure 3.2: The Harris detector searches for image neighborhoods where the second-moment matrix **C** has two large eigenvalues, corresponding to two dominant orientations. The resulting points often correspond to corner-like structures. Courtesy of Dennis Simakov and Darya Frolova.

In this formulation, the convolution with the Gaussian $G(\mathbf{x}, \tilde{\sigma})$ takes the role of summing over all pixels in a circular local neighborhood, where each pixel's contribution is additionally weighted by its proximity to the center point.

Instead of explicitly computing the eigenvalues of **C**, the following equivalences are used

$$\det(\mathbf{C}) = \lambda_1 \lambda_2 \tag{3.4}$$
$$\text{trace}(\mathbf{C}) = \lambda_1 + \lambda_2 \tag{3.5}$$

to check if their ratio $r = \frac{\lambda_1}{\lambda_2}$ is below a certain threshold. With

$$\frac{\text{trace}^2(\mathbf{C})}{\det(\mathbf{C})} = \frac{(\lambda_1 + \lambda_2)^2}{\lambda_1 \lambda_2} = \frac{(r\lambda_2 + \lambda_2)^2}{r\lambda_2^2} = \frac{(r+1)^2}{r} \tag{3.6}$$

this can be expressed by the following condition

$$\det(\mathbf{C}) - \alpha \text{trace}^2(\mathbf{C}) > t, \tag{3.7}$$

which avoids the need to compute the exact eigenvalues. Typical values for α are in the range of $0.04 - 0.06$. The parameter $\tilde{\sigma}$ is usually set to 2σ, so that the considered image neighborhood is slightly larger than the support of the derivative operator used.

Figure 3.7(top right) shows the results of the Harris detector and compares them to those of the Hessian. As can be seen, the returned locations are slightly different as a result of the changed selection criterion. In general, it can be stated that Harris locations are more specific to corners,

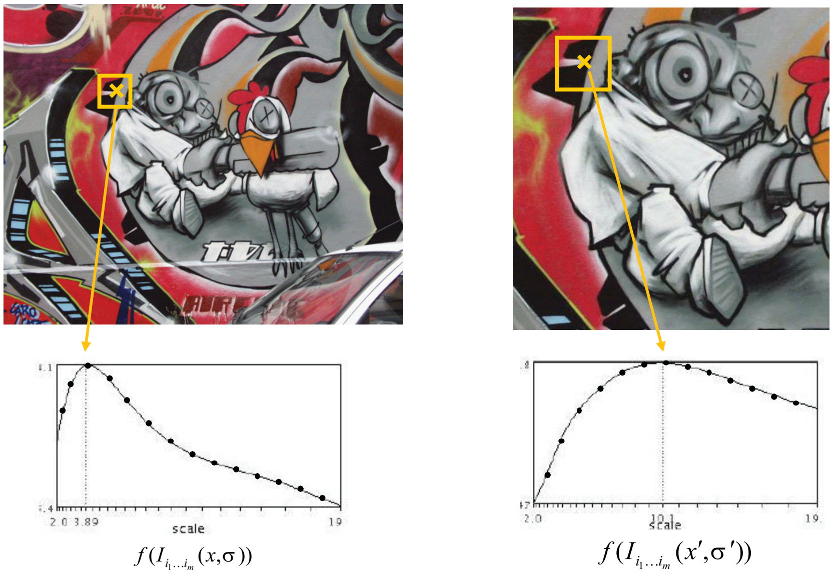

Figure 3.3: The principle behind automatic scale selection. Given a keypoint location, we evaluate a scale-dependent signature function on the keypoint neighborhood and plot the resulting value as a function of the scale. If the two keypoints correspond to the same structure, then their signature functions will take similar shapes and corresponding neighborhood sizes can be determined by searching for scale-space extrema of the signature function *independently in both images*. Courtesy of Krystian Mikolajczyk.

while the Hessian detector also returns many responses on regions with strong texture variation. In addition, Harris points are typically more precisely located as a result of using first derivatives rather than second derivatives and of taking into account a larger image neighborhood.

Thus, Harris points are preferable when looking for exact corners or when precise localization is required, whereas Hessian points can provide additional locations of interest that result in a denser coverage of the object.

3.2.2 SCALE INVARIANT REGION DETECTION

While shown to be remarkably robust to image plane rotations, illumination changes, and noise [Schmid et al., 2000], the locations returned by the Harris and Hessian detectors are only repeatable up to relatively small scale changes. The reason for this is that both detectors rely on Gaussian derivatives computed at a certain fixed base scale σ. If the image scale differs too much between the test images, then the extracted structures will also be different. For scale invariant feature extraction, it is thus necessary to detect structures that can be reliably extracted under scale changes.

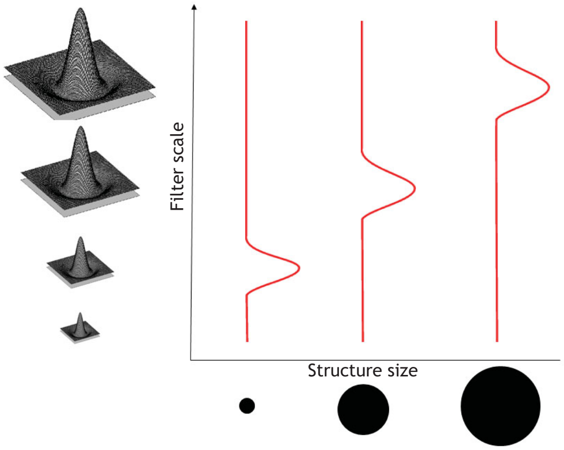

Figure 3.4: The (scale-normalized) Laplacian-of-Gaussian (LoG) is a popular choice for a scale selection filter. Its 2D filter mask takes the shape of a circular center region with positive weights, surrounded by another circular region with negative weights. The filter response is therefore strongest for circular image structures whose radius corresponds to the filter scale.

3.2.2.1 Automatic Scale Selection

The basic idea behind automatic scale selection is visualized in Figure 3.3. Given a keypoint in each image of an image pair, we want to determine whether the surrounding image neighborhoods contain the same structure up to an unknown scale factor. In principle, we could achieve this by sampling each image neighborhood at a range of scales and performing $N \times N$ pairwise comparisons to find the best match. This is, however, too expensive to be of practical use. Instead, we evaluate a *signature function* on each sampled image neighborhood and plot the result value as a function of the neighborhood scale. Since the signature function measures properties of the local image neighborhood at a certain radius, it should take a similar qualitative shape if the two keypoints are centered on corresponding image structures. The only difference will be that one function shape will be squashed or expanded compared to the other as a result of the scaling factor between the two images. Thus, corresponding neighborhood sizes can be detected by searching for extrema of the signature function *independently in both images*. If corresponding extrema σ and σ' are found in both cases, then the scaling factor between the two images can be obtained as $\frac{\sigma'}{\sigma}$.

Effectively, this procedure builds up a *scale space* [Witkin, 1983] of the responses produced by the application of a local kernel with varying scale parameter σ. In order for this idea to work, the signature function or kernel needs to have certain specific properties. It can be shown that the only operator that fulfills all necessary conditions for this purpose is the scale-normalized Gaussian kernel $G(\mathbf{x}, \sigma)$ and its derivatives [Lindeberg, 1994, 1998].

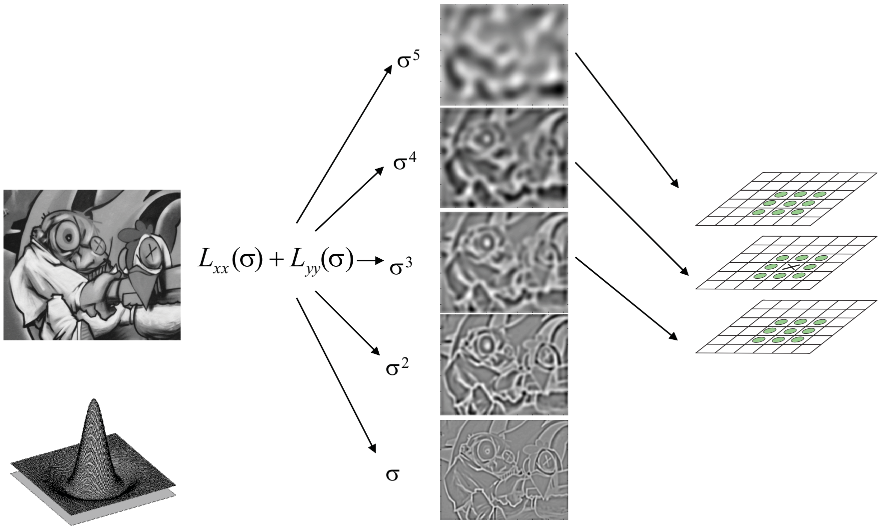

Figure 3.5: The Laplacian-of-Gaussian (LoG) detector searches for 3D scale space extrema of the LoG function. Courtesy of Krystian Mikolajczyk.

3.2.2.2 The Laplacian-of-Gaussian (LoG) Detector

Based on the above idea, Lindeberg proposed a detector for blob-like features that searches for scale space extrema of a scale-normalized *Laplacian-of-Gaussian* (LoG) [Lindeberg, 1998]:

$$L(\mathbf{x}, \sigma) \quad = \quad \sigma^2 \left(I_{xx}(\mathbf{x}, \sigma) + I_{yy}(\mathbf{x}, \sigma) \right). \tag{3.8}$$

As shown in Figure 3.4, the LoG filter mask corresponds to a circular center-surround structure, with positive weights in the center region and negative weights in the surrounding ring structure. Thus, it will yield maximal responses if applied to an image neighborhood that contains a similar (roughly circular) blob structure at a corresponding scale. By searching for scale-space extrema of the LoG, we can therefore detect circular blob structures.

Note that for such blobs, a repeatable keypoint location can also be defined as the blob center. The LoG can thus both be applied for finding the *characteristic scale* for a given image location and for directly detecting *scale-invariant regions* by searching for 3D (location + scale) extrema of the LoG. This latter procedure is visualized in Figure 3.5 and resulting interest regions are shown in Figure 3.7(bottom left).

3.2.2.3 The Difference-of-Gaussian (DoG) Detector

As shown by Lowe [2004], the scale-space Laplacian can be approximated by a difference-of-Gaussian (DoG) $D(\mathbf{x}, \sigma)$, which can be more efficiently obtained from the difference of two adjacent

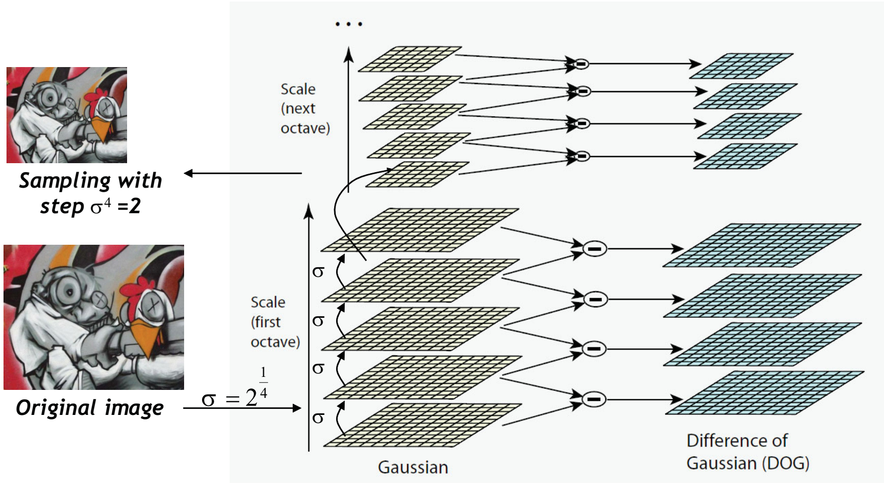

Figure 3.6: The Difference-of-Gaussian (DoG) provides a good approximation for the Laplacian-of-Gaussian. It can be efficiently computed by subtracting adjacent scale levels of a Gaussian pyramid. The DoG region detector then searches for 3D scale space extrema of the DoG function. From Tuytelaars and Mikolajczyk [2007].

scales that are separated by a factor of k:

$$D(\mathbf{x}, \sigma) \quad = \quad (G(\mathbf{x}, k\sigma) - G(\mathbf{x}, \sigma)) \star I(\mathbf{x}) \ . \tag{3.9}$$

Lowe [2004] shows that when this factor is constant, the computation already includes the required scale normalization. One can therefore divide each scale octave into an equal number K of intervals, such that $k = 2^{1/K}$ and $\sigma_n = k^n \sigma_0$. For more efficient computation, the resulting scale space can be implemented with a Gaussian pyramid, which resamples the image by a factor of 2 after each scale octave. See Figure 3.6.

As in the case of the LoG detector, DoG interest regions are defined as locations that are simultaneously extrema in the image plane and along the scale coordinate of the $D(\mathbf{x}, \sigma)$ function. Such points are found by comparing the $D(\mathbf{x}, \sigma)$ value of each point with its 8-neighborhood on the same scale level, and with the 9 closest neighbors on each of the two adjacent levels (as depicted in the right side of Figure 3.5).

Since the scale coordinate is only sampled at discrete levels, it is important in both the LoG and the DoG detector to interpolate the responses at neighboring scales in order to increase the accuracy of detected keypoint locations. In the simplest version, this could be done by fitting a second-order polynomial to each candidate point and its two closest neighbors. A more exact approach was introduced by Brown and Lowe [2002]. This approach simultaneously interpolates both the location and scale coordinates of detected peaks by fitting a 3D quadric function.

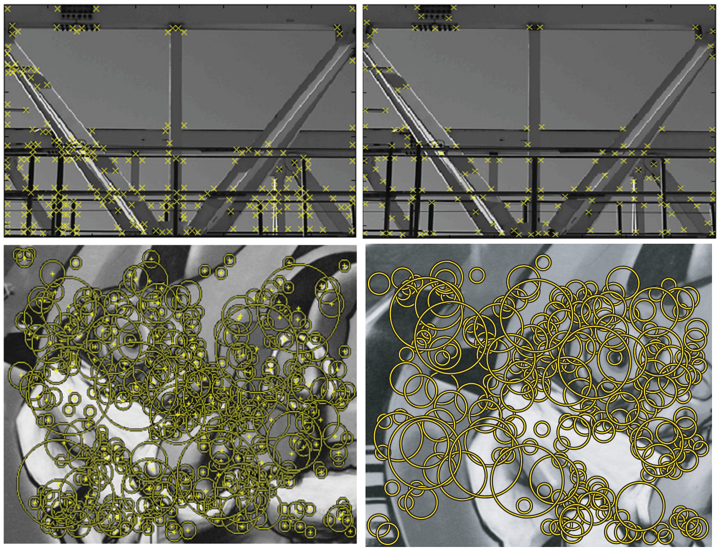

Figure 3.7: Example results of the (top left) Hessian detector; (top right) Harris detector; (bottom left) Laplacian-of-Gaussian detector; (bottom right) Difference-of-Gaussian detector. Courtesy of Krystian Mikolajczyk, from Tuytelaars and Mikolajczyk [2007].

Finally, those regions are kept that pass a threshold t and whose estimated scale falls into a certain scale range $[s_{min}, s_{max}]$. The resulting interest point operator reacts to blob-like structures that have their maximal extent in a radius of approximately 1.6σ of the detected points (as can be derived from the zero crossings of the modeled Laplacian). In order to also capture some of the surrounding structure, the extracted region is typically larger (most current interest region detectors choose a radius of $r = 3\sigma$ around the detected points). Figure 3.7(bottom right) shows the result regions returned by the DoG detector on an example image. It can be seen that the obtained regions are very similar to those of the LoG detector. In practice, the DoG detector is therefore often the preferred choice since it can be computed far more efficiently.

3.2.2.4 The Harris-Laplacian Detector

The Harris-Laplacian operator [Mikolajczyk and Schmid, 2001, 2004] was proposed for increased discriminative power compared to the Laplacian or DoG operators described so far. It combines the Harris operator's specificity for corner-like structures with the scale selection mechanism by Lindeberg [1998]. The method first builds up two separate scale spaces for the Harris function and the Laplacian. It then uses the Harris function to localize candidate points on each scale level and selects those points for which the Laplacian simultaneously attains an extremum over scales.

The resulting points are robust to changes in scale, image rotation, illumination, and camera noise. In addition, they are highly discriminative, as several comparative studies show

[Mikolajczyk and Schmid, 2001, 2003]. As a drawback, however, the original Harris-Laplacian detector typically returns a much smaller number of points than the Laplacian or DoG detectors. This is not a result of changed threshold settings, but of the additional constraint that each point has to fulfill two different maxima conditions simultaneously. For many practical object recognition applications, the lower number of interest regions may be a disadvantage, as it reduces robustness to partial occlusion. This is especially the case for object categorization, where the potential number of corresponding features is further reduced by intra-category variability.

For this reason, an updated version of the Harris-Laplacian detector has been proposed based on a less strict criterion [Mikolajczyk and Schmid, 2004]. Instead of searching for *simultaneous* maxima, it selects scale maxima of the Laplacian at locations for which the Harris function also attains a maximum *at any scale*. As a result, this modified detector yields more interest points at a slightly lower precision, which results in improved performance for applications where a larger absolute number of interest regions is required [Mikolajczyk et al., 2005].

3.2.2.5 The Hessian-Laplace Detector

As in the case of the Harris-Laplace, the same idea can also be applied to the Hessian, leading to the *Hessian-Laplace* detector. As with the single-scale versions, the Hessian-Laplace detector typically returns more interest regions than Harris-Laplace at a slightly lower repeatability [Mikolajczyk et al., 2005].

3.2.3 AFFINE COVARIANT REGION DETECTION

The approaches discussed so far yield local features that can be extracted in a manner that is invariant to translation and scale changes. For many practical problems, however, it also becomes important to find features that can be reliably extracted under large viewpoint changes. If we assume that the scene structure we are interested in is locally planar, then this would boil down to estimating and correcting for the perspective distortion a local image patch undergoes when seen from a different viewpoint. Unfortunately, such a perspective correction is both computationally expensive and error-prone since the local feature patches typically contain only a small number of pixels. It has, however, been shown by a number of researchers [Matas et al., 2002, Mikolajczyk and Schmid, 2004, Schaffalitzky and Zisserman, 2002, Tuytelaars and Van Gool, 2000, 2004] that a local affine approximation is sufficient in such cases.

We therefore aim to extend the region extraction procedure to *affine covariant* regions[1]. While a scale- and rotation-invariant region can be described by a circle, an affine deformation transforms this circle to an ellipse. We thus aim to find local regions for which such an ellipse can be reliably and repeatedly extracted purely from local image properties.

[1] The literature speaks of affine *covariant* extraction here in order to emphasize the property that extracted region *shapes* vary according to the underlying affine deformation. This is required so that the region *content* will be *invariant*.

3.2.3.1 Harris and Hessian Affine Detectors

Both the Harris-Laplace and Hessian-Laplace detectors can be extended to yield affine covariant regions. This is done by the following iterative estimation scheme. The procedure is initialized with a circular region returned by the original scale-invariant detector. In each iteration, we build up the region's second-moment matrix and compute the eigenvalues of this matrix. This yields an elliptical shape (as shown in Figure 3.2), corresponding to a local affine deformation. We then transform the image neighborhood such that this ellipse is transformed to a circle and update the location and scale estimate in the transformed image. This procedure is repeated until the eigenvalues of the second-moment matrix are approximately equal.

As a result of this iterative estimation scheme, we obtain a set of elliptical regions which are adapted to the local intensity patterns, so that the same object structures are covered despite the deformations caused by viewpoint changes.

3.2.3.2 Maximally Stable Extremal Regions (MSER)

A different approach for finding affine covariant regions has been proposed by Matas et al. [2002]. In contrast to the above methods, which start from keypoints and progressively add invariance levels, this approach starts from a segmentation perspective. It applies a watershed segmentation algorithm to the image and extracts homogeneous intensity regions which are stable over a large range of thresholds, thus ending up with *Maximally Stable Extremal Regions* (MSER). By construction, those regions are stable over a range of imaging conditions and can still be reliably extracted under viewpoint changes. Since they are generated by a segmentation process, they are not restricted to elliptical shapes, but can have complicated contours. In fact, the contour shape itself is often a good feature, which has led to the construction of specialized contour feature descriptors [Matas et al., 2002]. For consistency with the other feature extraction steps discussed here, an elliptical region can however also easily be fitted to the Maximally Stable regions by computing the eigenvectors of their second-moment matrices.

3.2.3.3 Other Interest Region Detectors

Several other interest region detectors have been proposed that are not discussed here. Tuytelaars & Van Gool introduced detectors for affine covariant *Intensity Based Regions* (IBR) and *Edge Based Regions* (EBR) [Tuytelaars and Van Gool, 2004]. Kadir & Brady proposed a *Salient Regions* detector that was later on also extended to affine covariant extraction [Kadir and Brady, 2001, Kadir et al., 2004]. An overview over those detectors and a discussion of their merits can be found in [Tuytelaars and Mikolajczyk, 2007].

3.2.4 ORIENTATION NORMALIZATION

After a scale-invariant region has been detected, its content needs to be normalized for rotation invariance. This is typically done by finding the region's *dominant orientation* and then rotating the region content according to this angle in order to bring the region into a canonical orientation.

Lowe [2004] suggests the following procedure for the orientation normalization step. For each detected interest region, the region's scale is used to select the closest level of the Gaussian pyramid, so that all following computations are performed in a scale invariant manner. We then build up a gradient orientation histogram with 36 bins covering the 360° range of orientations. For each pixel in the region, the corresponding gradient orientation is entered into the histogram, weighted by the pixel's gradient magnitude and by a Gaussian window centered on the keypoint with a scale of 1.5σ. The highest peak in the orientation histogram is taken as the dominant orientation, and a parabola is fitted to the 3 adjacent histogram values to interpolate the peak position for better accuracy.

In practice, it may happen that multiple equally strong orientations are found for a single interest region. In such cases, selecting only one of them would endanger the recognition procedure since small changes in the image signal could cause one of the other orientations to be chosen instead, which could lead to failed matches. For this reason, Lowe suggests to create a separate interest region for each orientation peak that reaches at least 80% of the dominant peak's value [Lowe, 2004]. This strategy significantly improves the region detector's repeatability at a relatively small additional cost (according to Lowe [2004], only about 15% of the points are assigned multiple orientations).

3.2.5 SUMMARY OF LOCAL DETECTORS

Summarizing the above, we have seen the following local feature detectors so far. If precisely localized points are of interest, we can use the *Harris* and *Hessian* detectors. When looking for scale-invariant regions, we can choose between the *LoG* or *DoG* detectors, both of which react to blob-shaped structures. In addition, we can combine the Harris and Hessian point detectors with the Laplacian scale selection idea to obtain the *Harris-Laplacian* and *Hessian-Laplacian* detectors. Finally, we can further generalize those detectors to affine covariant region extraction, resulting in the *Harris-Affine* and *Hessian-Affine* detectors. The affine covariant region detectors are complemented by the *MSER* detector, which is based on maximally stable segmentation regions. All of those detectors have been used in practical applications. Detailed experimental comparisons can be found in [Mikolajczyk and Schmid, 2004, Tuytelaars and Mikolajczyk, 2007].

3.3 LOCAL DESCRIPTORS

Once a set of interest regions has been extracted from an image, their content needs to be encoded in a descriptor that is suitable for discriminative matching. The most popular choice for this step is the SIFT descriptor [Lowe, 2004], which we present in detail in the following.

3.3.1 THE SIFT DESCRIPTOR

The *Scale Invariant Feature Transform* (SIFT) was originally introduced by Lowe as combination of a DoG interest region detector and a corresponding feature descriptor [Lowe, 1999, 2004]. However, both components have since then also been used in isolation. In particular, a series of studies has

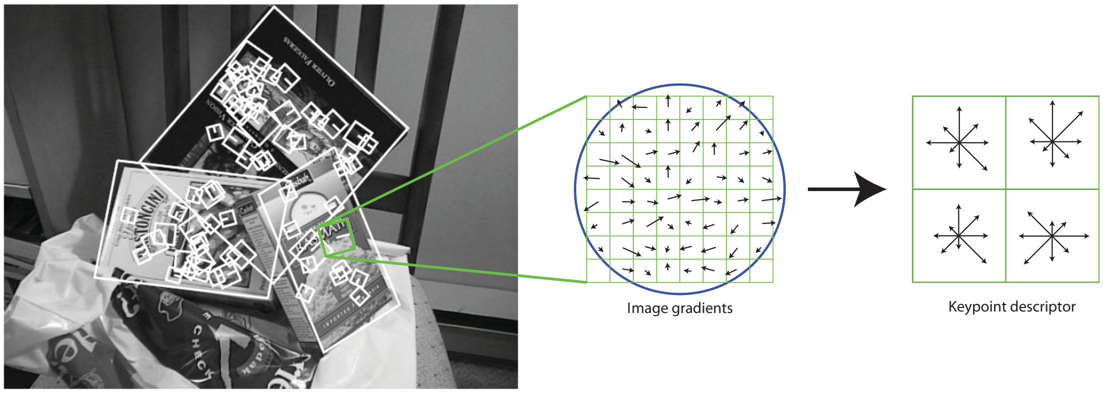

Figure 3.8: Visualization of the SIFT descriptor computation. For each (orientation-normalized) scale invariant region, image gradients are sampled in a regular grid and are then entered into a larger 4 × 4 grid of local gradient orientation histograms (for visibility reasons, only a 2 × 2 grid is shown here). Based on Lowe [1999] and Lowe [2004].

confirmed that the SIFT descriptor is suitable for combination with all of the above-mentioned region detectors and that it achieves generally good performance [Mikolajczyk and Schmid, 2005].

The SIFT descriptor aims to achieve robustness to lighting variations and small positional shifts by encoding the image information in a *localized set of gradient orientation histograms*. The descriptor computation starts from a scale and rotation normalized region extracted with one of the above-mentioned detectors. As a first step, the image gradient magnitude and orientation is sampled around the keypoint location using the region scale to select the level of Gaussian blur (*i.e.*, the level of the Gaussian pyramid at which this computation is performed). Sampling is performed in a regular grid of 16 × 16 locations covering the interest region. For each sampled location, the gradient orientation is entered into a coarser 4 × 4 grid of gradient orientation histograms with 8 orientation bins each, weighted by the corresponding pixel's gradient magnitude and by a circular Gaussian weighting function with a σ of half the region size. The purpose of this Gaussian window is to give higher weights to pixels closer to the middle of the region, which are less affected by small localization inaccuracies of the interest region detector.

This procedure is visualized for a smaller 2 × 2 grid in Figure 3.8. The motivation for this choice of representation is that the coarse spatial binning allows for small shifts due to registration errors without overly affecting the descriptor. At the same time, the high-dimensional representation provides enough discriminative power to reliably distinguish a large number of keypoints.

When computing the descriptor, it is important to avoid all boundary effects, both with respect to spatial shifts and to small orientation changes. Thus, when entering a sampled pixel's gradient information into the 3-dimensional spatial/orientation histogram, its contribution should be smoothly distributed among the adjoining histogram bins using trilinear interpolation.

Once all orientation histogram entries have been completed, those entries are concatenated to form a single $4 \times 4 \times 8 = 128$ dimensional feature vector. A final illumination normalization completes the extraction procedure. For this, the vector is first normalized to unit length, thus adjusting for changing image contrast. Then all feature dimensions are thresholded to a maximum value of 0.2 and the vector is again normalized to unit length. This last step compensates for non-linear illumination changes due to camera saturation or similar effects.

3.3.2 THE SURF DETECTOR/DESCRIPTOR

As local feature detectors and descriptors have become more widespread, efficient implementations are getting more and more important. Consequently, several approaches have been proposed in order to speed up the interest region extraction and/or description stages [Bay et al., 2008, 2006, Cornelis and Van Gool, 2008, Rosten and Drummond, 2008]. Among those, we want to pick out the SURF ("Speeded-Up Robust Features") approach, which has been designed as an efficient alternative to SIFT [Bay et al., 2008, 2006].

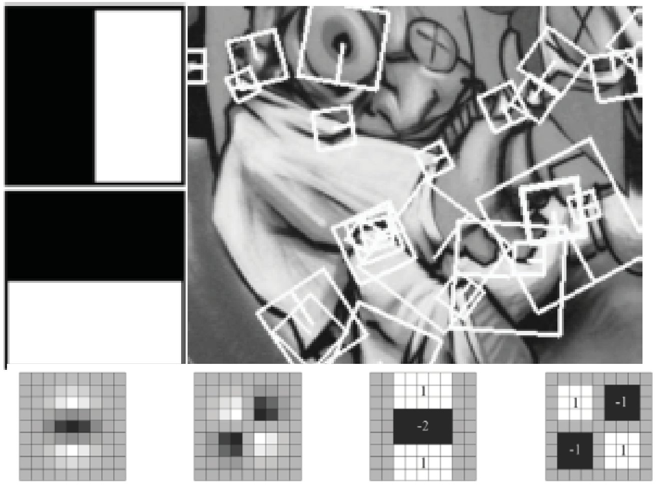

Figure 3.9: The SURF detector and descriptor were designed as an efficient alternative to SIFT. Instead of relying on ideal Gaussian derivatives, their computation is based on simple 2D box filters, which can be efficiently evaluated using integral images. Based on Bay et al. [2008].

SURF combines a Hessian-Laplace region detector with its own gradient orientation-based feature descriptor. Instead of relying on Gaussian derivatives for its internal computations, it is however based on simple 2D box filters ("Haar wavelets"), as shown in Figure 3.9. Those box filters approximate the effects of the derivative filter kernels, but they can be efficiently evaluated using integral images [Viola and Jones, 2004]. In particular, this evaluation requires the same constant number of lookups regardless of the image scale, thus removing the need for a Gaussian pyramid.

Despite this simplification, SURF has been shown to achieve comparable repeatability as detectors based on standard Gaussian derivatives, while yielding speedups of more than a factor of five compared to standard DoG.

The SURF descriptor is also motivated by SIFT and pursues a similar spatial binning strategy, dividing the feature region into a 4 × 4 grid. However, instead of building up a gradient orientation histogram for each bin, SURF only computes a set of summary statistics $\sum \mathrm{d}x$, $\sum |\mathrm{d}x|$, $\sum \mathrm{d}y$, and $\sum |\mathrm{d}y|$, resulting in a 64-dimensional descriptor, or a slightly extended set resulting in a 128-dimensional descriptor version.

Motivated by the success of SURF, a further optimized version has been proposed by Cornelis and Van Gool [2008] that takes advantage of the computational power available in current CUDA enabled graphics cards. This GPUSURF implementation has been reported to perform feature extraction for a 640 × 480 image at frame rates up to 200 Hz (*i.e.*, taking only 5ms per frame), thus making feature extraction a truly affordable processing step.

3.4 CONCLUDING REMARKS

The first step in the specific-object recognition pipeline is to extract local features from both the training and test images. For this, we can use any of the feature detectors and descriptors described in this chapter. The appropriate choice of detector class (single-scale, scale-invariant, or affine-invariant) mainly depends on the type of viewpoint variations foreseen for the target application.

For many practical recognition applications, scale invariant features (in particular SIFT [Lowe, 2004]) have proven a good compromise, since they are fast to extract, are robust to moderate viewpoint variations, and retain enough discriminative power to allow for reliable matching. When dealing with larger viewpoint changes, as in wide-baseline stereo matching applications, affine invariance becomes important in order to still establish correspondences. However, the added invariance comes at the price of reduced discriminative power (since several different elliptical regions can be mapped onto the same circular descriptor neighborhood) and generally also a smaller number of features (since not all regions have a characteristic affine-covariant neighborhood). Whenever possible, it is therefore advisable to use the simpler feature representation.

The development of local invariant features has had an enormous impact in many areas of computer vision, including wide-baseline stereo matching, image retrieval, object recognition, and categorization. They have provided the basis for many state-of-the-art algorithms and have led to a number of new developments. Moreover, efficient implementations for all detectors discussed in this chapter are freely available [GPU, 2008, Oxf, 2004, SUR, 2006], making them truly building blocks that other researchers can build on.

In the next two chapters, we discuss how to compute candidate matching descriptors, and then describe how the geometric consistency of those candidate matches are verified to perform specific object recognition. In later chapters we will again draw on local descriptors to build models for generic object categorization.