SIAM J. IMAGING SCIENCES
Vol. 5, No. 2, pp. 652–687

On Affine Invariant Descriptors Related to SIFT*

R. Sadek†, C. Constantinopoulos‡, E. Meinhardt†, C. Ballester†, and V. Caselles†

Abstract. Using a classical result on algebraic invariants of the unimodular group, we present in this paper some basic geometric affine invariant quantities, and we use them to construct some distinctive descriptors for object detection. Although full affine invariance cannot be guaranteed due to noncommutativity of camera blur with affine maps and the domain problem (that is, the difficulty of finding an affine covariant domain), the proposed descriptors behave more robustly than SIFT with respect to affine deformations. This is supported by our comparisons both with the version of SIFT computed on an affine normalized neighborhood, and with ASIFT, which solves both the previously mentioned camera blur and domain problems by cleverly sampling the orbit of affine transformations of the images.

Key words. image matching, affine invariance, image descriptors, object recognition

AMS subject classifications. 68U10, 15A72, 14L24, 68T45

DOI. 10.1137/100798739

1. Introduction.

Object recognition in computer vision can be briefly defined as the task of finding a given object in an image or in a video sequence. Usually the object itself is defined by an image and may vary in scale and pose with respect to the image where the object is searched for. In recent years, an extensive huge literature has developed on object recognition motivated by its numerous applications, in particular, applications to object or image recognition in large databases, such as the web itself [51, 52, 8, 28, 50, 42], or to image classification [6, 7].

The starting point of most approaches is the computation of keypoints and their feature descriptors. In order to deal with real applications, computed feature descriptors have to be invariant with respect to a set of image distortions. Let us briefly comment on the main ones.

First, let us mention the invariance with respect to blur. Image blur is the result of many factors: out-of-focus camera, aperture diffraction, motion blur, pixel sampling, interpolation, etc. Inverting the blur is an ill-posed problem. A typical way to obtain features invariant to blur is to use a multiscale representation of the images. When the blur kernel has a small support, then at some scale the effect of the blur will disappear.

*Received by the editors June 14, 2010; accepted for publication (in revised form) February 17, 2012; published electronically June 5, 2012.

http://www.siam.org/journals/siims/5-2/79873.html

†Departament de Tecnologies de la Informació i les Comunicacions, Universitat Pompeu–Fabra, 08018 Barcelona, Spain (ridasadek@gmail.com, enric.meinhardt@upf.edu, coloma.ballester@upf.edu, vicent.caselles@upf.edu). The work of these authors was partially supported by MICINN project, reference MTM2009-08171, and by GRC, reference 2009 SGR 773, funded by the Generalitat de Catalunya. The last author's work was also partially supported by "ICREA Acadèmia" prize for excellence in research funded by the Generalitat de Catalunya.

‡Fundació Barcelona Media, 08017 Barcelona, Spain (konstantinos.konstantinopoulos@barcelonamedia.org). The work of this author was partially supported by the Torres Quevedo Program from the Ministry of Science and Innovation in Spain (MICINN), cofunded by the European Social Fund (ESF).

Second, ignoring the camera blur and assuming that the two images to be compared are taken from different positions and/or orientations, we need feature descriptors that are invariant with respect to projective transformations. Since this is a too general class, if the objects we are searching for can be locally approximated by a plane, then we can restrict the required invariance to the set of affine transformations. This approach is commonly used in practice and is the object of interest in this paper. There are many approaches that satisfy affine invariance with different degrees of approximation: from Euclidean invariance to approximately affine invariance. If we introduce the camera blur into the picture, then we need descriptors that are invariant with respect to affine transforms determined by the change of position of the camera. The trouble comes from the fact that affine transforms do not commute, in general, with the blur kernel [39]. Assuming, as we will do in this paper, that the blur kernel is radial, only rotations and translations commute with it. These parameters can be normalized by geometric affine invariant descriptors computed on the image. The remaining parameters cannot. They are the scale parameter corresponding to the camera zoom, and the camera axis longitude and latitude. Thus, following [38, 39], we are led to distinguish between geometric affine invariance (corresponding to an ideal pinhole camera, with no blur) and camera affine invariance that also models the camera blur. In order to get camera affine invariance, some authors have proposed simulating the remaining set of parameters by suitably sampling the affine orbits of both images. This is the case of Ferns [21, 41] or the variant of SIFT called ASIFT [38]. This may be very important in practice since in many cases there may be big scale and/or tilt changes between the two images that are compared.

Now, there is the invariance with respect to contrast changes which may be due to illumination variations, different gamma corrections, etc. Ignoring camera blur, exact invariance to contrast changes is assured by using morphological operations to construct the features and their descriptors, e.g., the level lines of the image [25, 10] or the gradient directions which describe the normals to the level lines [8, 28]. Another possibility, also used in the literature, is the binarization of the gray level comparison between pairs of pixels in a neighborhood of a keypoint [21, 41]. In practice, morphological operations are often complemented by weighting the results with a local contrast measure, such as the norm of the image gradient. When camera blur comes into play, it mixes level lines and changes their geometry. Since the two images may be taken from different viewpoints, this cannot be addressed by only simulating the blur with the scale space, and one also needs to simulate the camera axis longitude and latitude, as proposed in [39].

Noise, yet another distortion, is an inevitable artifact of image acquisition. It is due mainly to physical effects which are not taken into account and to the sampling and quantization of pixels. Noise invariance is probably impossible to achieve. The best we can aim for is noise robustness. This is achieved by defining suitable descriptors and a suitable metric for comparing these descriptors instead of comparing them by equality. This requires a threshold on the distance of two descriptors, or some other criterion to accept or reject matched features.

Finally, let us mention occlusions. They correspond to a basic operation in image formation and are perhaps the most important distortion. However, it is very difficult to model occlusions directly, so they are generally ignored as a source of invariance in the context of detection or recognition. The possibility to detect partially occluded objects is based on the use of local features that describe small parts of objects. Thus, we can say that occlusion

invariance is achieved by using only local features (points, curves, patches, etc.).

Henceforth, we will use the term *descriptor* as a set of numbers computed from a local feature. Thus, a descriptor is a vector in $\mathbf{R}^N$. Using this language, we can summarize a general scheme for many methods of object recognition. The input of these methods is often a pair of images u and v, and the output is a correspondence (or list of correspondences) between parts of each image. The generic method consists of the following steps:

Keypoint or feature extraction: Extract the features on both images. Thus, we form two sets of features F_u and F_v associated to each one of the input images.

Computation of descriptors: Compute the descriptors of each feature. A descriptor is computed on an image patch centered at an extracted feature.

Matching: For each feature $p_i \in F_u$ find a matching feature $q_j \in F_v$ such that their descriptors are as close as possible in the descriptor metric. Thus, we form a set M of pairs of features (p_i, q_j).

By analyzing this set of correspondences we may extract a geometric transformation between the two images. This transformation helps to group several correspondences in a geometrically consistent way. This is usually done by the following step:

Clustering: Search in M for clusters of pairs of features that can be matched by the same affinity (or some predefined kind of correspondence).

Finally, the last step is the following:

Verification: Verify that each cluster actually corresponds to an instance of u that appears in v.

Clearly, this scheme is a simplification but gives a summarized description of the structure of many methods. This paper is devoted to the generation of geometric affine invariant quantities and their use in building up descriptors that behave more robustly than SIFT with respect to affine transformations.

Many descriptors have been proposed to address the above set of invariances. Let us briefly mention two of them: SIFT and Ferns. They will be reviewed in more detail in section 2. Other variants will also be briefly discussed. The scale invariant feature transform (SIFT) [27, 28] is a descriptor defined in terms of weighted histograms of gradient directions around a given keypoint. The use of gradient directions provides robustness with respect to illumination changes. Usually keypoints are maxima of some measure (e.g., corners, edge points, blobs) often computed at different scales. The histograms of orientations (in the neighborhood of a keypoint) used in the SIFT descriptor are computed on the Gaussian scale space of the image. This normalizes the scale parameter. Since the method is also invariant to translations and rotations of the image plane around the camera axis, four out of the six parameters of an affine transform are normalized. The other two, the longitude and latitude of the camera axis, cannot be normalized since their associated transformation does not commute with the Gaussian kernel (assuming that the camera blur can be well approximated by a Gaussian). The computation of orientation histograms provides partial invariance with respect to the angle between the object's plane and the optical axis (the latitude). In [28], Lowe studied the robustness of SIFT with respect to latitude changes and provided some practical bounds on their maximum variation, around 50°, for SIFT to work.

It is important to note that the (geometric or camera) affine invariance is related not only to the descriptor but also to the domain where it is computed. We will call this the domain

problem. Notice that, due to the camera blur, the generation of domains that are related by an affine transformation when the two images are taken by a camera from two different viewpoints is not an obvious question. Thus we can also distinguish between geometric and camera domain problems. Mikolajczyk and Schmid [31] addressed the computation of an affine covariant domain, and by doing that they enforced the camera affine invariance of SIFT. They proposed an affine normalization of the keypoint neighborhood based on an iterative computation and normalization of the second moment matrix [14, 24, 23]. In this process, the selection of the affine domain alternates with a simulation of the corresponding blur. Thus, they attempt to compute a camera affine invariant domain; however, no proof of this is provided. After this normalization, SIFT is computed, although other descriptors could be used. Let us refer to this version of SIFT as SIFT+NN.

Using a different approach to impose the camera affine invariance and solve the domain problem, Morel and Yu [38] proposed generating an orbit of affine deformations of one (or both) images and apply SIFT to the simulated images, obtaining results that outperform SIFT. Using the invariance properties of SIFT, the orbit was reduced to a simulation of the longitude and latitude (tilt) parameters. Indeed, in [38] the authors propose a precise and careful sampling of the orbit that takes into account the SIFT performance in scale and angular (latitude) changes. In this way, they guarantee that the query image will have a positive matching with one of the orbit images, specifically the nearest one. Moreover, they proved mathematically that the resulting method is camera affine invariant, up to an arbitrary precision. On the other hand, by providing the image orbits, a set of domains is generated, and one of them will be similar to the domain in the query image. The feature descriptors computed on these domains are relieved from the responsibility of being highly robust to big affine transformations and, indeed, become camera affine invariant in this setting (modulo the sampling of the orbit). This method is known as ASIFT [38].

Ferns [22, 41, 21] also addresses the problem of illumination and camera affine invariant recognition. Although we leave the detailed description of it for section 2.4, let us point out that, as in ASIFT, the camera affine invariance of the Ferns descriptor is obtained by the construction of an orbit of affine deformations of the model image [22, 41, 21].

Finally, a different special affine invariant region detector based on the computation of sufficiently contrasted level lines, the maximally stable extremal regions (MSER), was introduced in [29]. Although the domain is normalized with respect to all six parameters of the affine transform, this normalization is not perfect, since level lines change when the amount of blur changes. Thus, MSER are geometric affine invariant but not camera affine invariant. In practice they are camera affine robust for moderate affine camera motions, but only if the scale change is not big. The MSER method provides a complementary point of view since it directly addresses the domain problem and computes geometric affine invariant domains that are based on contrasted level sets (on which arbitrary affine invariant descriptors can later be computed). We retain from it the observation that, discarding boundary effects, the level sets of the image are the natural geometric affine invariant domains on which descriptors can be computed. As pointed out above, MSER select the most contrasted level lines in a precise sense that will be reviewed in section 2.5.

In the previous paragraphs we have discussed one of the key elements for the construction of camera affine invariant descriptors around keypoints: the need to have a camera affine

covariant domain; that is, the computed neighborhoods around corresponding keypoints of two images related by an affinity should match under the affine map. The domain problem is perhaps the most difficult one, and we have reviewed several proposals for solving it. Both the computation of an intrinsic affine normalized neighborhood [31] and the generation of an orbit [38] aim to solve the domain problem and the compensation of the partially missing invariance of SIFT with respect to out-of-plane rotations. Our paper does not address this problem but rather the computation of geometric affine invariant quantities assuming that the domain problem has been solved. When combined with an orbit, these quantities also give camera affine invariant descriptors. Then the following question arises:

Do we need geometric affine invariant quantities if we have normalized the domain or simulated its affine distortions?

From the above discussion, due to camera blur, we need camera affine invariant quantities. Our descriptors are only geometric affine invariant, although they can be converted to camera affine invariant if we use an image orbit, as in ASIFT, or approximately camera affine invariant if we use an intrinsic affine normalized neighborhood, as in [31]. In relation to ASIFT [38], there is still room for improvement because the images generated constitute a sampling of the affine orbit of the given images, and the performance of ASIFT depends on the number of simulated images. In relation to SIFT+NN [31], there is no theoretical guarantee that SIFT+NN provides a normalized camera affine invariant neighborhood, but it seems to work in practice, at least as a good approximation. In any case, the comparison of our descriptors with SIFT in both of the contexts SIFT+NN and ASIFT shows that the proposed quantities permit us to improve the results. Let us also mention that the performance of any quantity depends on its discriminative power. The proposed quantities are different from those of SIFT (although they use the same organization) and exhibit more discriminative power, improving on the performance of SIFT.

Our contribution. Thus, the purpose of this paper is to propose a set of geometric affine invariant quantities to be used in the construction of feature descriptors. They will effectively improve upon the robustness of SIFT to affine transformations. The basic quantities were introduced in [3, 4] in the context of affine invariant image segmentation. In the context of the present section we will sometimes omit the word geometric when talking about affine invariants. In [3, 4] the authors proposed an affine covariant quantity associated to a given finite length curve Γ, namely the quantity

$$\int_0^1 \int_0^1 |c'(s) \wedge c'(t)|\, ds\, dt,$$

where $c : [0,1] \to \mathbf{R}^2$ is a parameterization of Γ and $c'(s) \wedge c'(t) := \det(c'(s), c'(t))$. By integrating on the level lines of the image, this quantity can be translated to images u defined on the plane as

$$\int_{\mathbf{R}^2} \int_{\mathbf{R}^2} |\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|\, d\mathbf{x} d\mathbf{y}, \tag{1.1}$$

where $\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y}) := \det(\nabla u(\mathbf{x}), \nabla u(\mathbf{y}))$ is just the area determined by the two vectors $\nabla u(\mathbf{x})$ and $\nabla u(\mathbf{y})$. Note that, if $\mathcal{T}\mathbf{x} = A\mathbf{x} + \mathbf{x_0}$, where A is a 2×2 matrix, $\mathbf{x}_0$ is a given point

in $\mathbf{R}^2$, and $u_{\mathcal{T}}(\mathbf{x}) = u(\mathcal{T}\mathbf{x})$, we have $\nabla u_{\mathcal{T}}(\mathbf{x}) \wedge \nabla u_{\mathcal{T}}(\mathbf{y}) = \det(A)\, \nabla u(\mathcal{T}\mathbf{x}) \wedge \nabla u(\mathcal{T}\mathbf{y})$ and, as can be easily checked, the quantity (1.1) is affine covariant (that is, affine invariant modulo a scale factor that is a power of the determinant of the affinity). It can be used as a basis for constructing geometric affine invariant descriptors on keypoints.

If we had at our disposal an affine covariant neighborhood $V_{\mathbf{x}}(u)$ of a keypoint $\mathbf{x}$ (that is, such that $\mathcal{T}V_{\mathbf{x}}(u_{\mathcal{T}}) = V_{\mathcal{T}\mathbf{x}}(u)$), we could define the affine invariant quantity associated to $\mathbf{x}$:

$$\int_{V_{\mathbf{x}}(u)} |\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|\, d\mathbf{y}.$$

The quantity $\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})$ is one of the basic covariant quantities that we will introduce in section 3. This and other examples are based on a classical result on algebraic invariants of the unimodular group [53]. Thus, we discuss in section 3 a generic method to construct other geometric affine covariant quantities and obtain affine invariants using combinations of them. Using these geometric affine invariant quantities, we implement in section 4 a set of descriptors with an algorithmic structure similar to SIFT (see section 4.2). Since we will be referring to them later on, let us call them $\mathcal{AD}$ descriptors. At this point, note that we can compute them in any given neighborhood.

Although we use geometric affine invariant quantities even if there is no camera blur, the geometric affine invariance is guaranteed only if we have an affine covariant domain. As we said above, we do not address the domain problem in this paper. Either we can use the affine normalized neighborhood of Mikolajczyk and Schmid [31], or we may use an orbit of affine deformations as in [22, 41, 38] or simply take advantage of the level sets of the image as it is proposed by MSER [29], although MSER do not guarantee camera affine invariance. Thus, we compared the proposed $\mathcal{AD}$ descriptors both to SIFT+NN and to ASIFT. When comparing to SIFT+NN, we used the same normalized neighborhood. Our experiments show that the results obtained using $\mathcal{AD}$ improve the results obtained by SIFT+NN. When comparing to ASIFT, we used the same orbit and the same square neighborhood. We did experiments with three orbit sizes, and in all three cases we are able to obtain results that improve those obtained by ASIFT (especially when we use a reduced orbit). In both experiments, we see that the camera affine invariance is reinforced. Indeed, note that, by the arguments in [38], geometric affine invariants computed using the Gaussian scale space become camera affine invariant in the sense of [38] after simulating an orbit as in ASIFT.

Inspired by MSER (see section 2.5), we also explored a further variant based on the observation that level sets are geometric affine covariant domains. Let us first describe it in the context of SIFT. Assume that $\mathbf{x}$ is a keypoint of an image u and $\mathcal{N}_{\mathbf{x}}$ is a given neighborhood of $\mathbf{x}$ (as in common SIFT). As a first step we quantize the image u in $\mathcal{N}_{\mathbf{x}}$. To avoid the effect of illumination changes we equalize the image u in $\mathcal{N}_{\mathbf{x}}$ (that is, we compute $H_{\mathbf{x}}(u)$, where $H_{\mathbf{x}}(\cdot)$ is the distribution function of u in $\mathcal{N}_{\mathbf{x}}$) and then define the (bi)level sets $\mathcal{N}_{\mathbf{x}}^{u,j} := \{\mathbf{y} \in \mathcal{N}_{\mathbf{x}} : j\Delta \le H_{\mathbf{x}}(u)(\mathbf{y}) < (j+1)\Delta\}$, where Δ is a quantization step and $j = 0, 1, \ldots, \frac{256}{\Delta} - 1$. In the case of SIFT, the new descriptor can be described as a modification of the SIFT descriptor that takes into account the level sets $\mathcal{N}_{\mathbf{x}}^{u,j}$ (in practice we use $\Delta = 64$ and $j = 0, 1, 2, 3$). It is formed by the concatenation of vectors v_j, $j = 0, 1, \ldots, \frac{256}{\Delta} - 1$. While in the case of common SIFT each pixel $y \in \mathcal{N}_{\mathbf{x}}$ contributes to a coordinate of a vector $v \in \mathbf{R}^{128}$, now each pixel $y \in \mathcal{N}_{\mathbf{x}}^{u,j}$ contributes to the corresponding coordinate of $v_j \in \mathbf{R}^{128}$. When $\Delta = 256$ and $j = 0$,

we recover the standard SIFT. By weighting the histogram of orientations in SIFT with the geometric affine invariant quantities generated by the method described in section 3, we have a set of descriptors that reinforce affine invariance.

Thus, for each quantity we can use the similar algorithmic structure of SIFT in $\mathcal{N}_{\mathbf{x}}$ to get the descriptors that we called $\mathcal{AD}$, or we may organize them as we described in the previous paragraph, taking into account the level sets $\mathcal{N}_{\mathbf{x}}^{u,j}$, $j = 0, 1, \ldots, \frac{256}{\Delta} - 1$. Let us refer to them as the quantized level set version of $\mathcal{AD}$, or simply as $\mathcal{AD}$+QLS.

Although we use geometric affine invariant quantities, if we use a square neighborhood $\mathcal{N}_{\mathbf{x}}$ to compute the sets $\mathcal{N}_{\mathbf{x}}^{u,j}$, we break the geometric affine covariance of the domain. But we have experimentally observed that we gain distinctness with this proposal when compared to common SIFT. We have also considered the descriptors in $\mathcal{AD}$+QLS computed on the normalized affine invariant neighborhood of [31] and we obtain better results than with SIFT+NN and also better results than by using $\mathcal{AD}$. The comparison of $\mathcal{AD}$+QLS with ASIFT (using a square neighborhood and the same orbit for both) does not show an improvement over using $\mathcal{AD}$. The reason for this may be that the information brought by the orbit is sufficient to cancel the benefits gained by the QLS strategy. Indeed, the results obtained with $\mathcal{AD}$ and $\mathcal{AD}$+QLS are similar except in the case of a reduced orbit, where the QLS version improves over $\mathcal{AD}$. The experiments will be shown in section 5. Summarizing our comparisons, we observe that the proposed descriptors are more robust to affine deformations than SIFT.

Let us finally explain the plan of the paper. In section 2 we review the main descriptors of interest in the context of this paper, namely SIFT, Ferns, MSER, and their variants. In section 3 we recall a basic result of the theory of algebraic invariants of the unimodular group, and we use it to generate some geometric affine invariant quantities. In section 4 we introduce the descriptors that we use and briefly describe their implementation in consonance with SIFT. In section 5 we show some experiments comparing our descriptors with SIFT using Harris affine keypoints and its normalized neighborhood (SIFT+NN) [31], and with ASIFT [38]. Section 6 summarizes our conclusions.

2. Review of several descriptors for object recognition.

The purpose of this section is to review SIFT and some other relevant descriptors which have been introduced in the literature and are connected with our discussion here. Since we are searching for affine invariant descriptors, it will be useful to recall the decomposition of an affine matrix in terms of geometric parameters related to the observation of a plane in the scene by an affine camera [16].

2.1. Geometric description of an affine map.

Image distortions arising from viewpoint changes can be locally approximated by affine planar transforms, assuming that the objects we are searching for can be locally approximated by planes. Thus we restrict ourselves to studying the invariance of descriptors with respect to planar affine transformations.

Let us describe an affine transformation in terms of intrinsic parameters that have a geometric significance. Assume that images u and v are defined in $\mathbf{R}^2$ and are related by an affine map, so that

$$v(\mathbf{x}) = u(\mathcal{T}\mathbf{x}),$$

where

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix},$$

and

$$\mathcal{T}\mathbf{x} := A\mathbf{x} + \mathbf{x}_0 := \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}.$$

Since we will compare two keypoints, we may assume that we have already corrected the relative translation, and we may assume that $\mathbf{x}_0 = 0$. We also assume that the affine map is orientation preserving, so that A has a positive determinant. Then A has a unique decomposition

$$A = H_\lambda R_1(\psi) T_t R_2(\phi) = \lambda \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}, \tag{2.1}$$

where $\lambda > 0$, $R_1(\psi), R_2(\phi)$ are rotations of angles $\psi, \phi \in [0, \pi]$, respectively, and T_t is a tilt, namely a diagonal matrix whose eigenvalues are $(t, 1)$, $t \geq 1$. Figure 1 shows the interpretation of this affine decomposition in terms of the relative position of the (affine) camera and the object's plane where ϕ and $\theta = \arccos 1/t$ are the viewpoint angles, θ represents the angle between the optical axis and the normal to the object's plane, ϕ represents the relative rotation between the optical axis and a fixed axis on the plane, and ψ parameterizes a rotation of the camera plane. The angles θ and ϕ are called latitude and longitude, respectively. Finally, we can change the focal length or the distance of the camera to the scene, and this is reflected in the zoom parameter λ.

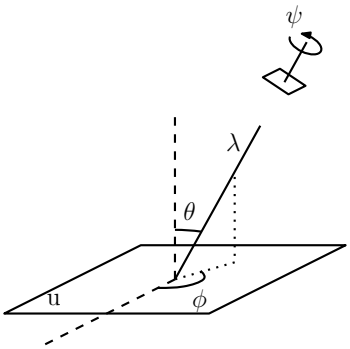


Figure 1. *Geometric interpretation of the decomposition formula (2.1). This figure illustrates the four main parameters in the affine image deformation. The angle θ, the latitude, is the angle between the optical axis and the normal to the image plane. The plane containing the normal and the optical axis makes an angle ϕ with a fixed axis in the object's plane. This angle is called the longitude. The camera can also rotate around its optical axis (rotation parameter ψ). Finally, the camera can move forward or backward, or change its focal length. This is the zoom parameter λ.*

2.2. Affine camera model. Let u_0 be an infinite resolution frontal view image of a flat object. Following [38], we model digital images acquired by a camera by the relation

$$u = S_1 G_\alpha \mathcal{T} u_0, \tag{2.2}$$

where $\mathcal{T}(\mathbf{x}) = A\mathbf{x} + \mathbf{x}_0$, A is a linear map with positive determinant, $\mathbf{x}_0 \in \mathbf{R}^2$, G_α is a Gaussian convolution with standard deviation $\alpha > 0$ modeling the optical blur and ensuring that there is no aliasing by 1-sampling, and, finally, S_1 is the sampling operator on a regular

grid with 1-spacing. Notice that, following the usual simplifying assumption, the camera blur is modeled by a Gaussian kernel. Since we assume that images are well sampled, from now on we do not consider the sampling operation.

For later use, we say that the image u is a frontal snapshot if we can write it as (2.2), where $\mathcal{T}$ is a similarity, that is, the tilt parameter $t = 1$.

The presence of the camera blur obliges one to distinguish between geometric affine invariant descriptors and camera affine invariant ones. If two images are related by an affine map, as $u(x)$ and $u_{\mathcal{T}}(\mathbf{x}) = u(\mathcal{T}\mathbf{x})$, a geometric affine invariant descriptor would give the same descriptors for u and $u_{\mathcal{T}}$. If we work with two digital images, say $u = G_\alpha \mathcal{T}_1 u_0$ and $v = G_\beta \mathcal{T}_2 u_0$, a camera affine invariant descriptor would give the same quantities for u and v (this will be made precise below). Note that only rotations and translations commute with camera blur, while the other three parameters do not commute. Thus, Euclidean invariant descriptors permit one to normalize these parameters. Scale, longitude, and latitude cannot be normalized and have to be simulated. SIFT, in addition to normalizing rotations and translations, simulates the scale and, thus, is a similarity invariant descriptor. The proposal of ASIFT is to normalize the remaining two parameters by simulating an orbit of transformations, proving that in this way one gets a camera affine invariant descriptor. Let us review both methods.

2.3. Review of SIFT and some relatives.

Common SIFT and its invariance properties. Let us briefly summarize SIFT descriptors [28] using the generic scheme mentioned in section 1. Its input is a pair of images. To fix ideas, let us say that the first image u is a query image, while the second image v is a target image. If the query image contains a certain object, the output of the method gives the corresponding object in image v, in case it is present, and the corresponding affine map between the parts of the images u and v containing the object. The limitations of SIFT concerning its affine invariance are described in [28] and further analyzed in [38]. Since it is based on the comparison of descriptors computed on keypoints, the translation invariance is guaranteed. The invariance with respect to scale changes is obtained by simulating them. Since $G_\delta H_\lambda u_0 = H_\lambda G_{\lambda\delta} u_0$ for any $\delta, \lambda > 0$, this can be achieved using the Gaussian scale space. Indeed, to compute the "SIFT keypoints" of an image u, first one computes a Gaussian scale space pyramid of u (with scales sampled exponentially) and then tries to capture the intrinsic scale of the keypoint by finding the scale space local maxima of the Laplacian of the Gaussian. Selecting keypoints as local maxima in the scale space makes it possible that a point x may appear several times, at several different scales. Thus, the method is (modulo discretizations) invariant with respect to scale. Finally, for each keypoint found, a more stable positioning with subpixel accuracy is computed by fitting a three-dimensional quadratic function to the keypoint. This determines the interpolated location of the maximum.

Now, given a keypoint $\mathbf{x}$, its SIFT descriptor is based on the computation of histograms of gradient orientations in a neighborhood of $\mathbf{x}$. To compute the gradient directions, one first computes the dominant orientation of the gradient in $\mathbf{x}$ (estimated from the gradients around this point) and takes it as the new x-axis. The gradient orientations are then referred to this axis. In this way, the method is invariant to rotations in the image plane. Then, the SIFT descriptor is computed in a neighborhood of $\mathbf{x}$, typically of size 16×16 in the standard SIFT implementation. This neighborhood is divided into 4×4 blocks of size 4×4. On each

of them, a weighted histogram of directions quantized in 8 angular bins is computed. The weights of this histogram depend both on the modulus of the gradient at the pixel and on the distance to the central keypoint. At the end we have a descriptor which is a vector v with 128 coordinates. Each coordinate of v contains the weight contributions of all pixels of a given block with a given angular orientation. Finally, in practice, to avoid quantization effects, each orientation occurrence is distributed in several neighboring bins. We note that the use of scale space provides some invariance to camera blur, and, since SIFT uses gradient directions, it is robust to illumination changes. Furthermore, the descriptor is first normalized, then large values are thresholded and finally normalized again, which makes SIFT robust to saturation effects.

As proved in [39, Theorem 1], for two frontal snapshots of the same full resolution flat image u_0, SIFT descriptors are identical if both camera blurs coincide; otherwise, they become similar as soon as the scale of the simulated Gaussian grows. In conclusion, as we have mentioned, the translation, rotation, and scale parameters can be fixed, although one has to pay attention to the sampling issues [38]. Now, taking (2.1) into account, there are still two parameters to be fixed in order to get camera affine invariance, namely, the angle between the normal to the object's plane and the optical axis of the camera (the latitude), and the relative rotation between the optical axis and a fixed axis on the object's plane (the longitude); see Figure 1. Let us mention the two main methods that have been proposed to address this problem: either by computing an intrinsic neighborhood of the keypoint which aims to achieve camera affine invariance [31], or by simulating the remaining affine deformations of the query image, related to the longitude and latitude angles, as proposed in ASIFT [38] (see also [43] for a first step toward this method). A full discussion of this problem and the relevance of respecting Shannon sampling theorem when simulating the affine deformations can be found in [38].

A related variation of SIFT proposed in [20], which somewhat improves its rotation invariance but does not tackle invariance under arbitrary affinities, consists of using circular neighborhoods instead of square ones. In any case, this is a minor modification, since the square neighborhoods of common SIFT are rotated toward a principal direction, and the points inside the neighborhood are weighted using a circular distribution around the center of the square.

ASIFT: Simulating the tilt and longitude parameters. Although the method in [31] produces good results, better results are obtained by simulating all possible affine distortions of the image and applying SIFT to each of them [38]. By our previous discussion (see [38]) it suffices to simulate the affine deformations determined by the two parameters ϕ and $t = |\frac{1}{\cos\theta}|$. Given an image u, one first simulates the rotations with respect to the longitude parameter. An important point for digital images, as discussed in [38], is that the tilt involves a subsampling of factor t in the x-direction (after a rotation by ϕ), and therefore its simulation requires the previous application of an anti-aliasing filter, namely the convolution by a Gaussian with standard deviation $c\sqrt{t^2-1}$ (for good anti-aliasing, $c \sim 0.6$) [38]. Thus, a digital tilt in the direction x (resp., y) is simulated by the map $T_t^x G_{c\sqrt{t^2-1}}$ (resp., $T_t^x G_{c\sqrt{t^2-1}}$). Since $T_t^x G_{c\sqrt{t^2-1}} G_c = G_c T_t^x$ [38, Theorem 1], by simulating digital tilts we are able to commute tilts with Gaussian blur, and this is the main ingredient in proving that ASIFT is camera affine invariant [38, Theorem 2]. On the other hand, it is not geometric affine invariant. The

conclusion from [38] is that, by simulating tilts, longitudes, and scales, one can convert a descriptor that is invariant to similarity transforms into a camera affine invariant one.

SIFT on affine neighborhoods. We have mentioned the invariance of SIFT with respect to the parameters λ and ψ in (2.1). The lack of invariance with respect to the image deformations induced by the angles θ and ϕ has to be compensated for. In [31], the authors proposed computing an affine invariant interest point detector, called Harris affine, which detects a keypoint and provides it with an affine region given by a 2×2 matrix $\mathbf{M}$ representing this region. The Harris interest point detector is first applied at several scales [15], followed by an iterative selection of the scale and the location (as an extremum over scale of the Laplacian of the Gaussian) [31]. This provides a set of points that are robust to scale changes. Then, an affine normalization of the point neighborhood is computed as a fixed point by alternating the selection of the affine domain with a simulation of the corresponding blur. One can interpret this as an approximate way of obtaining a camera affine invariant domain. The method and the algorithm for finding such keypoints and their associated normalized neighborhoods are discussed in depth in [31]. The output is a set of keypoints where for each of them we have its position, an affine covariant neighborhood, and its dominant orientation which is obtained as in SIFT but using the normalized neighborhood. The neighborhood is expressed by the shape adaptation matrix which is a symmetric positive-definite 2×2 matrix $\mathbf{M}$ whose eigenvectors express the axis directions of the corresponding elliptical shape $\{\mathbf{x} \in \mathbf{R}^2 : \langle \mathbf{M}\mathbf{x}, \mathbf{x} \rangle \leq 1\}$. Having $\mathbf{M}$, one can normalize the image on the affine neighborhood of the keypoint by mapping it to an image on the unit circle. With this, one compensates for the affine transform caused by the camera change of position. Applying SIFT to the normalized image, one obtains in principle a camera affine invariant descriptor. Although the authors of [31] do not rigorously demonstrate this camera affine invariance, it seems to work in practice.

Other descriptors related to SIFT. Inspired by SIFT, many other related descriptors have been proposed with the purpose of improving or speeding up SIFT. In particular, let us mention SURF detector and descriptor (speeded-up robust features) [5], PCA (principal component analysis) SIFT [18], and FAST (features from accelerated segment test) [44]. Also, other descriptors specially tailored for operating on a dense set of keypoints have been proposed, such as HOG (histograms of oriented gradients) [13], GLOH (gradient location and orientation histograms) [32], LESH (local energy based shape histograms) [48], and, in some sense, spin images [17]. Let us note that it is also possible, and useful, to compute SIFT descriptors on dense keypoints [26].

2.4. Review of Ferns method. A different method which also addresses the problems of illumination and affine invariant recognition was proposed in [22, 21, 41]. Given a query and a target image, the camera affine invariance of the method is given, as in ASIFT, by the construction of an orbit of affine and blur deformations of the target image. Then a set of keypoints (using a cornerness measure) are computed on this orbit. Only those keypoints that are stable under the deformations are used as effective keypoints. Each of them determines a class which is characterized by the comparisons (encoded in a binary vector) of the gray levels of a selected random set of pairs of points in its neighborhood. These comparisons determine the posterior probability that a given point belongs to a class and may be used for classification using a naive Bayes approach. Thus, assuming that the priors of each class are uniform, we are

led to the computation of the likelihood of each class for each set of answers to the questions. Let us mention the Ferns method [41], of which the randomized tree method is a variant with a different organization. Since one cannot assume that each question is independent of the others, the proposal in [41] is to assume that the set of questions is organized in subsets, called Ferns, which are independent. Thus, a Fern is determined by a random selection of a subset of pairs of pixels where the values of the gray levels are compared. Several Ferns are used to characterize a keypoint. Each comparison produces a binary answer. This guarantees the morphological invariance of the method, which amounts to its invariance with respect to illumination changes [49]. The probability of each Fern, given the class, is learned offline from the keypoints computed in the orbit. In order to compute it, a tree structure is used. The product of the likelihoods of Ferns gives the likelihood of the set of answers that characterizes each class. This method does not require the storage of the orbit of affine deformations since, for each Fern, the probability computations can be organized as a tree that can be learned offline. A full comparison of the Ferns method with SIFT is beyond the scope of the present paper.

2.5. Review of MSER method. A completely different approach to affine invariance is that of maximally stable extremal regions (MSER) [29]. The main idea behind MSER, as opposed to SIFT, is to build up affine covariant domains on which arbitrary affine invariant descriptors (or arbitrary descriptors, if one first performs an affine normalization [29]) will be computed, thus giving a solution to the domain problem. Two key observations in [29] lead to the choice of MSER as robust elements for establishing image correspondences under severe viewpoint changes for automatic reconstruction of three-dimensional scenes. In the wide-baseline stereo problem, local image deformations cannot be realistically approximated by Euclidean motions, and a full affine model is required (as an approximation to the projective transformation between the images). On the other hand, the elements should be robust against illumination changes modeled here as monotonic transformation of image intensities. Thus, MSER are defined as the most contrasted connected components of upper and lower level sets of the image [29]. Let us point out that MSER are only geometric, and not camera, affine invariant. If we take the camera blur into account, MSER achieves invariance only with respect to translation and rotation. The other three parameters (zoom and camera axis longitude and latitude) cannot be perfectly normalized since they do not commute with the image blur and have to be simulated [38].

Let us review the definition of MSER. We recall it here using the language of the trees of connected components of upper and lower level sets [49, 45, 11, 12] (an analogous notion for the tree of shapes [37] can also be defined; see [37, 12]). Let Ω be the image domain, usually a closed rectangle in $\mathbf{R}^2$. Let $u : \Omega \to \mathbf{R}$ be a given image (modeled as an upper semicontinuous function). Let us first introduce some basic notation that will also be of use later in section 3. Given $\lambda \in \mathbf{R}$, we denote $\{u \geq \lambda\} := \{\mathbf{x} \in \Omega : u(\mathbf{x}) \geq \lambda\}$, which is the upper level set of u determined by the height λ. We denote by $\mathcal{CC}(\{u \geq \lambda\})$ the family of connected components of $\{u \geq \lambda\}$. In order to define the MSER we fix a threshold value $\delta > 0$. If u takes values in a discrete set of integers, we may take $\delta = 1$. This is the more relevant case in applications since we deal with digital (sampled and quantized) images. For each connected component of an upper level set of u, $X_\lambda \in \mathcal{CC}(\{u \geq \lambda\})$, we consider $X_{\lambda-\delta} \in \mathcal{CC}(\{u \geq \lambda-\delta\})$,

$X_{\lambda+\delta} \in \mathcal{CC}(\{u \geq \lambda + \delta\})$ such that $X_{\lambda+\delta} \subseteq X_\lambda \subseteq X_{\lambda-\delta}$. We define the function

$$F_\delta^u(\lambda) := \frac{\text{Area}(X_{\lambda-\delta}) - \text{Area}(X_{\lambda+\delta})}{\text{Area}(X_\lambda)}. \tag{2.3}$$

We say that X_λ is a maximally stable extremal (upper) region if X_λ achieves a local minimum of $F_\delta(\lambda)$. The function F_δ is well defined on the maximal branches of the tree of connected components of upper level sets of u, where no bifurcation takes place [12]. When X_λ contains a bifurcation, the connected component $X_{\lambda+\delta} \in \mathcal{CC}([u \geq \lambda + \delta])$ is not uniquely defined. We have dismissed those elements.

In a similar way we can define a maximally stable extremal (lower) region this time using the connected components of the lower level sets of u. We shall refer to both of these regions as MSER.

As in section 2.3, we dismiss the problems caused by boundary effects, and we assume that the image u is defined in $\mathbf{R}^2$. If $\mathcal{T}$ is an affine transformation and we define $u_\mathcal{T}(\mathbf{x}) = u(\mathcal{T}\mathbf{x})$, then the trees of u and $u_\mathcal{T}$ have the same structure. Since the function $F_\delta^u(\lambda) = F_\delta^{u_\mathcal{T}}(\lambda)$, we have that the MSER of u and $u_\mathcal{T}$ are related by $\mathcal{T}$ in a covariant way. Thus, they are invariant under affine transformations. Its invariance under contrast changes comes from the fact that we are using connected components of level sets.

As we mentioned previously, the affine invariance of MSER is only geometric, and camera invariance does not hold. This means that MSER may fail under large scale changes or large tilts, or when well-contrasted shapes are not present [38]. In these cases, the image shape boundaries tend to mix. Following [38], camera affine invariance can be obtained after simulating the scale, longitude, and latitude parameters.

Finally, let us say that a related descriptor based on level lines was proposed in [25, 10]. As MSER, it is geometric affine invariant and photometric invariant.

Descriptors for MSER. Following the general scheme mentioned in section 1, MSER are geometric affine covariant domains on which many different descriptors can be computed, for example, SIFT. The descriptors originally proposed in [29] are based on constructing one or several *distinguished regions* (DR) around each MSER (e.g., an ellipse, the MSER itself, or several enlarged/reduced copies of the MSER or its convex hull). Then, rotationally invariant complex moments of the image inside the DR are computed after applying an affine transformation that normalizes the DR (e.g., so that the covariance matrix of the transformed DR becomes diagonal) [29]. Alternatively, one can build the descriptor using standard geometric affine invariant moments of the color values inside the DR [34, 29]. An analogous approach using the shapes of the image [37, 12] can be found in [36].

3. Generation of geometric affine invariant descriptors. Our purpose in this section is the generation of geometric affine invariant quantities. The quantities we propose here will be used in section 4 to generate new descriptors which are geometric affine invariant. When used with an affine normalized neighborhood as in [31], the descriptors become approximately camera affine invariant, although no formal proof of this exists. Camera affine invariance, up to an arbitrary level of precision, can be obtained by simulating scales, longitude, and latitude parameters (which are the three parameters that do not commute with radial camera blurs) as in ASIFT [38]. Although this camera affine invariance can be obtained if we start from any

Euclidean invariant descriptor, such as SIFT, the experiments displayed in section 5 comparing our descriptors with SIFT+NN and with ASIFT using their normalization strategies show that the proposed descriptors are more robust to affine perturbations.

We have distinguished above between geometric and camera affine invariance. To simplify our expressions, in the rest of the paper, when we say affine invariant or covariant we mean geometric affine invariant or covariant, respectively. When we refer to camera affine invariance (resp., covariance) we will say it explicitly.

Let us denote by $GL(2,\mathbf{R})^+$ the set of all 2×2 matrices with positive determinant. To avoid boundary effects we assume in this section that images are defined on $\mathbf{R}^2$. If $A \in GL(2,\mathbf{R})^+$ and $u : \mathbf{R}^2 \to \mathbf{R}$ is a given image, we denote $u_A(\mathbf{x}) = u(A\mathbf{x})$, $\mathbf{x} \in \mathbf{R}^2$.

Let $\mathcal{L}$ be a class of images (e.g., continuous, smooth, etc.). We say that $\mathcal{L}$ is $GL(2,\mathbf{R})^+$ invariant if $u_A \in \mathcal{L}$ for any $u \in \mathcal{L}$ and any $A \in GL(2,\mathbf{R})^+$. Notice that we are identifying the word image with a function from $\mathbf{R}^2$ or a domain of $\mathbf{R}^2$ to $\mathbf{R}$ (gray level image), ignoring for the moment the presence of the blur kernel.

Definition 3.1 (affine invariant and covariant descriptors). *Let $\mathcal{L}$ be a $GL(2,\mathbf{R})^+$ invariant class of images and let $\mathcal{F}$ be a class of allowed subsets of $\mathbf{R}^2$. Let $H(u,R)$ be a quantity which can be computed for any image $u \in \mathcal{L}$ and any subset $R \in \mathcal{F}$. Let $k \in \mathbf{R}$. We say that the quantity H is affine k-covariant if $H(u_A, A^{-1}(R)) = (\det(A))^k H(u,R)$ for any image u, any subset $R \in \mathcal{F}$, and any $A \in GL(2,\mathbf{R})^+$. When $k = 0$, we say that H is affine invariant.*

In other words, affine k-covariant quantities are quantities that are invariant to affine transformations of the image, up to a scale factor that is a power of the affine matrix determinant.

There are many ways to build new covariant quantities from old ones. Functions of covariants (of possibly different degree) can be made covariant, provided that the functions have a suitable degree of homogeneity. Arbitrary homogeneous functions of covariants are very general. They include, for instance, taking limits, integrals, maxima, and minima of sets of covariants.

All of these constructions produce new invariant quantities defined over the same class $\mathcal{F}$. A more interesting way to produce new invariants is by extending the class $\mathcal{F}$ on which an invariant is defined. The following lemma extends any invariant defined on upper level sets to an invariant defined on level curves.

Let $\mathcal{F}_u$ denote the family of connected components of all upper level sets of the image u.

Lemma 3.2. *Let $\mathcal{L}$ be a $GL(2,\mathbf{R})^+$ invariant class of continuous images. Let $k \in \mathbf{R}$ and let H be an affine k-covariant quantity defined on pairs (u,X), where $u \in \mathcal{L}$ and $X \in \mathcal{F}_u$. For each $\lambda \in \mathbf{R}$ we define $F(u,\partial X_\lambda) := \lim_{\delta\to 0+} \frac{H(u,X_{\lambda-\delta})-H(u,X_\lambda)}{\delta}$, assuming that the limit exists, where $X_\lambda \in \mathcal{CC}(\{u \geq \lambda\})$, $X_{\lambda-\delta} \in \mathcal{CC}(\{u \geq \lambda-\delta\})$, and $X_\lambda \subseteq X_{\lambda-\delta}$. Then $F(u,\partial X_\lambda)$ is an affine k-covariant quantity defined on the boundaries of upper level sets.*

We have written the previous lemma in an informal way. We observe that if $X_\lambda \in \mathcal{CC}(\{u \geq \lambda\})$ and δ is small enough, then there is only one connected component $X_{\lambda-\delta} \in \mathcal{CC}(\{u \geq \lambda-\delta\})$ containing X_λ. Then the lemma follows essentially by observing that $F_\delta(u,\partial X_\lambda) := \frac{H(u,X_{\lambda-\delta})-H(u,X_\lambda)}{\delta}$, $\delta > 0$, is affine k-covariant and passing to the limit as $\delta \to 0^+$.

The introduction of Lemma 3.2 was motivated by the next example.

Example 1. Let us compute the affine invariant quantity on level lines associated to the

area function defined on the upper level sets of an image u. Assume first that u is smooth and the integrals converge. As usual, $\nabla u(\mathbf{x})$ denotes the gradient of u at the point $\mathbf{x}$ and $|\nabla u(\mathbf{x})|$ its modulus. Then for each $\mu \in \mathbf{R}$, if X_μ is a connected component of $\{u \geq \mu\}$ and we denote by $u|_{X_\mu}$ the restriction of u to X_μ, by the coarea formula (see [2]), we have

$$H(X_\mu) := \text{Area}(X_\mu) = \int_\mu^\infty \int_{\partial\{u|_{X_\mu} \geq \eta\}} \frac{1}{|\nabla u(\mathbf{x})|}\, d\mathcal{H}^1(\mathbf{x})\, d\eta,$$

where $d\mathcal{H}^1$ denotes the one-dimensional Hausdorff measure, i.e., the arc length on $\partial\{u|_{X_\mu} \geq \eta\}$ in the above integral. Hence

$$F(u, \lambda, \delta) := \frac{1}{\delta} \int_{\lambda-\delta}^{\lambda+\delta} \int_{\partial\{u|_{X_\mu} \geq \eta\}} \frac{1}{|\nabla u(\mathbf{x})|}\, d\mathcal{H}^1(\mathbf{x})\, d\eta \approx 2 \int_{\partial X_\lambda} \frac{1}{|\nabla u(\mathbf{x})|}\, d\mathcal{H}^1(\mathbf{x}).$$

If $g^u(x) = \frac{1}{|\nabla u(\mathbf{x})|}$, and, for any set of finite perimeter E, we define the weighted perimeter $P_{g^u}(E) := \int_{\partial^* E} g_u(\mathbf{x}) d\mathcal{H}^1(\mathbf{x})$, where $\partial^* E$ denotes the essential boundary of E [2], then we have

$$F(u, \lambda, \delta) \to 2P_{g^u}(X_\lambda) \quad \text{as } \delta \to 0+.$$

This is the affine invariant quantity on level lines associated to the area of the level sets of u.

Remark 1. Using the previous example, we may redefine maximally stable upper regions of the image u as the local minimizers in the tree of upper connected components of

$$G(\lambda) := \frac{P_{g^u}(X_\lambda)}{|X_\lambda|} = \lim_{\delta \to 0+} F_\delta^u(\lambda),$$

where $F_\delta^u(\lambda)$ is defined in (2.3), $\lambda \in \mathbf{R}$. Note that the above quotient is a perimeter/area ratio; hence we may interpret MSER as local Cheeger sets (with respect to the weighted perimeter P_{g^u}) of the image domain, when we restrict the family of sets to the connected components of upper (or lower) level sets of the image. Similarly, we may redefine maximally stable lower regions of the image u (or the maximally stable shapes; see [30, 12]). The same analysis has been given in [19], where other interesting affine invariant measures for shape selection are also derived.

Our purpose in this section is to describe some other basic rules to generate affine invariants and covariants. In the language of Definition 3.1, these new invariants are defined over the same class of subsets by combining vector-valued invariants using simple algebraic rules.

Our discussion will be restricted to $\mathbf{R}^2$. Vectors of $\mathbf{R}^2$ will be designed by boldface roman letters $\mathbf{x}$ and $\mathbf{y}$, sometimes with subindices. Covectors of $\mathbf{R}^2$, that is, elements of the dual space, will be denoted by greek letters ξ, $\hat{\xi}$, etc. By $\langle \xi, \mathbf{x} \rangle$ we denote the standard dual pairing between the vector $\mathbf{x}$ and the covector ξ. In what follows, we fix the standard canonical basis $\mathbf{e}_1 = (1, 0)$, $\mathbf{e}_2 = (0, 1)$ of $\mathbf{R}^2$ and its dual basis ξ_1, ξ_2 (so that $\xi_i(e_j) = \delta_{ij}$, $i, j = 1, 2$, where $\delta_{ij} = 1$ if $i = j$, and 0 if $i \neq j$), and the action of a covector ξ of coordinates (ξ_x, ξ_y) on a vector $\mathbf{x} = (x, y)$ will be denoted by $\langle \xi, \mathbf{x} \rangle = \xi_x x + \xi_y y$, which is the standard scalar product. Clearly, given two vectors $\mathbf{x}, \mathbf{y} \in \mathbf{R}^2$, the determinant of the matrix whose columns are $\mathbf{x}$ and $\mathbf{y}$ is affine 1-covariant. This is the basic covariant made of vectors, and the others can

be deduced from it. Let us denote by $\mathbf{x} \wedge \mathbf{y}$ this determinant. Notice that vectors transform cogradiently, i.e., as $\mathbf{x} \to A\mathbf{x}$, while covectors transform contragradiently, i.e., as $\xi \to A^{-t}\xi$, by the group $GL(2,\mathbf{R})^+$ [53]. Note that when using coordinates, we can identify covectors as elements of $\mathbf{R}^2$, although we have to keep in mind their transformation rules.

Let $SL(2,\mathbf{R})$ be the unimodular group in $\mathbf{R}^2$, that is, the set of 2×2 matrices of determinant 1. Then we specify to the case $N = 2$ the following result proved in [53, Theorem 2.6.A].

Theorem 3.3 (see [53, Theorem 2.6.A]). *Let $\mathbf{x}, \mathbf{y} \in \mathbf{R}^2$ be vectors and $\xi, \hat{\xi}$ be two covectors. Then $\mathbf{x} \wedge \mathbf{y}$, $\langle \xi, \mathbf{x} \rangle$, and $\xi \wedge \hat{\xi}$ are the basic invariants for the unimodular group. Thus any other algebraic invariant is a polynomial in those basic elements.*

Observe that $\mathbf{x} \wedge \mathbf{y}$ generates an affine 1-covariant for the group $GL(2,\mathbf{R}^+)$, $\langle \xi, \mathbf{x} \rangle$ is an affine invariant, and $\xi \wedge \hat{\xi}$ generates an affine -1-covariant.

Definition 3.4. *Let $\mathcal{L}$ be a $GL(2,\mathbf{R})^+$ invariant class of images. Let $k \in \mathbf{R}$. Let $H(u, \mathbf{x}_1, \ldots, \mathbf{x}_p)$ be a quantity which can be computed for any image $u \in \mathcal{L}$ and any points $\mathbf{x}_1, \ldots, \mathbf{x}_p \in \mathbf{R}^2$. We say that the quantity H is an affine k-covariant density if $H(u_A, \mathbf{x}_1, \ldots, \mathbf{x}_p) = (\det(A))^k H(u, A\mathbf{x}_1, \ldots, A\mathbf{x}_p)$ for any image $u \in \mathcal{L}$, any points $\mathbf{x}_1, \ldots, \mathbf{x}_p$, and any $A \in GL(2,\mathbf{R})^+$. If $k = 0$, we say that H is an affine invariant density.*

Inspired by Theorem 3.3, we give some examples of affine k-covariant densities. We always assume that the image $u : \mathbf{R}^2 \to \mathbf{R}$ is smooth enough so that we can compute its gradient. This is so, for instance, if $u = G_t * u_0$, where $u_0 : \mathbf{R}^2 \to \mathbf{R}$ is a given image in $L^\infty(\mathbf{R}^2)$ (the space of measurable and essentially bounded functions) and G_t is the Gaussian of variance $t > 0$.

Examples.

1. The most basic invariant density is $H_{00}(u, \mathbf{x}) = u(\mathbf{x})$, $\mathbf{x} \in \mathbf{R}^2$.
2. Since $\nabla u_A(\mathbf{x}) = A^t \nabla u(A\mathbf{x})$, we have

$$\langle \mathbf{y}, \nabla u_A(\mathbf{x}) \rangle = \langle \tilde{\mathbf{y}}, \nabla u(\tilde{\mathbf{x}}) \rangle,$$

where $\tilde{\mathbf{x}} = A\mathbf{x}$ and $\tilde{\mathbf{y}} = A\mathbf{y}$, and $\langle \cdot, \cdot \rangle$ denotes the standard scalar product. Thus $H_{01}(u, \mathbf{x}, \mathbf{y}) = \langle \mathbf{y}, \nabla u(\mathbf{x}) \rangle$, $\mathbf{x}, \mathbf{y} \in \mathbf{R}^2$, is an affine invariant density.

3. Observe that

$$\nabla u_A(\mathbf{x}) \wedge \nabla u_A(\mathbf{y}) = \det A \, \nabla u(A\mathbf{x}) \wedge \nabla u(A\mathbf{y}).$$

Thus, we see that $H_{10}(u, \mathbf{x}, \mathbf{y}) = \nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})$ is an affine 1-covariant density.

4. Let

$$J := \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}.$$

The matrix J corresponds to a rotation by an angle of $\frac{\pi}{2}$. Observe that $J^2 = -I$, $JJ^t = I$, and

$$JA^tJ^{-1} = \operatorname{Cof} A^t = \det A \cdot A^{-1} \qquad \text{and} \qquad AJA^t = \det A \, J. \tag{3.1}$$

Notice that we have

$$D^2 u_A(\mathbf{x}) = A^t D^2 u(A\mathbf{x}) A.$$

Then the quantity $H_{20}(u,\mathbf{x},\mathbf{y},\mathbf{z}) := \langle D^2u(\mathbf{x})(J\nabla u(\mathbf{y})), J\nabla u(\mathbf{z})\rangle$ is an affine 2-covariant density, $\mathbf{x},\mathbf{y},\mathbf{z}\in\mathbf{R}^2$. Indeed

$$\begin{aligned}\langle D^2u_A(\mathbf{x})(J\nabla u_A(\mathbf{y})), J\nabla u_A(\mathbf{z})\rangle &= \langle A^tD^2u(A\mathbf{x})A(JA^t\nabla u(A\mathbf{y})), J\nabla u(A\mathbf{z})\rangle\\ &= \langle D^2u(A\mathbf{x})(AJA^t\nabla u(A\mathbf{y})), AJA^t\nabla u(A\mathbf{z})\rangle\\ &= (\det A)^2\langle D^2u(A\mathbf{x})(J\nabla u(A\mathbf{y})), J\nabla u(A\mathbf{z})\rangle.\end{aligned}$$

Notice that $\langle D^2u(\mathbf{x})(J\nabla u(\mathbf{x})), J\nabla u(\mathbf{x})\rangle = |\nabla u(\mathbf{x})|^3\mathrm{curv}(u)(\mathbf{x})$, where $\mathrm{curv}(u)(\mathbf{x})$ denotes the curvature of the level line of u passing by the point $\mathbf{x}$.

5. Combining the above quantities, one can get other affine invariant quantities. For instance, the quantity

$$Q(u)(\mathbf{x},\mathbf{y}) := \frac{\nabla u(\mathbf{x})\wedge\nabla u(\mathbf{y})}{|\langle D^2u(\mathbf{x})(J\nabla u(\mathbf{x})), J\nabla u(\mathbf{x})\rangle|^{1/2}}$$

is affine invariant.

Lemma 3.5. *Let $k\in\mathbf{R}$. Assume that $H(u,\mathbf{x}_1,\dots,\mathbf{x}_p)$ is an affine k-covariant density defined for u in a $GL(2,\mathbf{R})^+$ invariant class of images $\mathcal{L}$. If we integrate H with respect to j of its coordinates, we obtain an affine $(k-j)$-covariant density.*

Proof. (i) Suppose that we integrate its first j coordinates. Then

$$\begin{aligned}\int_{\mathbf{R}^2}\dots\int_{\mathbf{R}^2} H(u_A,\mathbf{x}_1,\dots,\mathbf{x}_p)\,d\mathbf{x}_1\dots\mathbf{x}_j &= (\det A)^k\int_{\mathbf{R}^2}\dots\int_{\mathbf{R}^2} H(u,A\mathbf{x}_1,\dots,A\mathbf{x}_p)\,d\mathbf{x}_1\dots\mathbf{x}_j\\ &= (\det A)^{k-j}\int_{\mathbf{R}^2}\dots\int_{\mathbf{R}^2} H(u,\mathbf{y}_1,\dots,\mathbf{y}_p)\,d\mathbf{y}_1\dots\mathbf{y}_j.\end{aligned}$$

■

Using the above examples combined with Lemmas 3.2 and 3.5, we get examples of affine covariant and invariant quantities.

Proposition 3.6. *Let $u:\mathbf{R}^2\to\mathbf{R}$ be an image which we assume smooth enough and let $n(\mathbf{x})$ denote the unit normal to the level line of u passing by the point $\mathbf{x}$. Let $X_\lambda\in\mathcal{CC}(\{u\ge\lambda\})$, $\lambda\in\mathbf{R}$. Assume, if necessary, that λ is not a critical value, that is, $\nabla u(\mathbf{y})\neq 0$ for any $\mathbf{y}\in\partial X_\lambda$. Let $k\in\mathbf{R}$.*

(i) *For any $\mathbf{x}\in\mathbf{R}^2$, the integrals $\int_{X_\lambda}|\nabla u(\mathbf{x})\wedge\nabla u(\mathbf{y})|^k\,d\mathbf{y}$ and*

$$\int_{\partial X_\lambda}|\nabla u(\mathbf{x})\wedge\nabla u(\mathbf{y})|^k/|\nabla u(\mathbf{y})|\,d\mathcal{H}^1(\mathbf{y})$$

are affine $(k-1)$-covariant quantities.

(ii) *The integrals $\int_{X_\lambda}|\langle\mathbf{y},\nabla u(\mathbf{y})\rangle|^k\,d\mathbf{y}$ and $\int_{\partial X_\lambda}|\langle\mathbf{y},\nabla u(\mathbf{y})\rangle|^k/|\nabla u(\mathbf{y})|\,d\mathcal{H}^1(\mathbf{y})$ are affine -1-covariant.*

(iii) *The quantities*

$$\int_{X_\lambda}|\langle D^2u(\mathbf{y})(J\nabla u(\mathbf{y})), J\nabla u(\mathbf{y})\rangle|^k\,d\mathbf{y},$$

$$\int_{\partial X_\lambda}|\langle D^2u(\mathbf{y})(J\nabla u(\mathbf{y})), J\nabla u(\mathbf{y})\rangle|^k/|\nabla u(\mathbf{y})|\,d\mathcal{H}^1(\mathbf{y})$$

are affine $(2k-1)$-covariant quantities.

Remark 2. Many other covariant quantities can be defined. For instance, for any $\mathbf{x}, \mathbf{e} \in \mathbf{R}^2$, the integral

$$H(u, \mathbf{x}, \mathbf{e}) := \int_0^\infty |\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{x} + s\mathbf{e})|^k \, ds$$

is an affine k-covariant quantity. For any $\mathbf{x}, \mathbf{y} \in \mathbf{R}^2$, the integral

$$\int_0^\infty |\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y} + sJ\nabla u(\mathbf{y}))|^k \, ds$$

is an affine $(k-1)$-covariant quantity. Also the quantities

$$\int_{X_\lambda} |\det D^2 u(\mathbf{y})|^k \, d\mathbf{y} \quad \text{and} \quad \int_{\partial X_\lambda} |\det D^2 u(\mathbf{y})|^k / |\nabla u(\mathbf{y})| \, d\mathcal{H}^1(\mathbf{y})$$

are affine $(2k-1)$-covariant quantities.

The covariant quantities defined in (i) are related to the affine covariant quantities defined in [3, 4]. The covariant (ii) coincides with the invariant for curved edges defined in [51, 52] (see also [33]). The covariant quantities defined in (iii) on the level lines of u contain the particular case $\int_{\partial X_\lambda} |\kappa(\mathbf{x})|^{1/3} \, d\mathcal{H}^1(\mathbf{x})$ which is the affine arc length parameter that played a fundamental role in the development of affine invariant scale space [1, 46, 47, 40]. This quantity has also been used in [9].

By combining the quantities given above we can get other ones. For instance, the quantities

$$\frac{|\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|^m}{\int_{X_\lambda} |\langle D^2 u(\mathbf{y})(J\nabla u(\mathbf{y})), J\nabla u(\mathbf{y})\rangle|^k \, d\mathbf{y}}$$

and

$$\frac{|\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|^m}{\int_{\partial X_\lambda} |\langle D^2 u(\mathbf{y})(J\nabla u(\mathbf{y})), J\nabla u(\mathbf{y})\rangle|^k / |\nabla u(\mathbf{y})| \, d\mathcal{H}^1(\mathbf{y})}$$

are $m - 2k + 1$ affine covariant, $m, k \in \mathbf{R}$. Other examples could be generated.

We describe the behavior of a function $H(u)$ with respect to affine illumination changes. We say that $H(u)$ scales as s^α if $H(su + a) = s^\alpha H(u)$ for any $s > 0$, $a \in \mathbf{R}$. We say that $H(u)$ is illumination invariant with respect to affine changes if $H(u)$ scales as s^0.

The quantity $Q(u)$ is affine invariant and scales as $s^{1/2}$. We may combine the quantities described in the examples above and in Proposition 3.6 in order to get affine invariant quantities which scale as s^0. To do this, let us consider an expression of the form

(3.2)
$$\frac{|\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|^m}{\left(\int_{\partial X_\lambda} |\langle D^2 u(\mathbf{y})(J\nabla u(\mathbf{y})), J\nabla u(\mathbf{y})\rangle|^k / |\nabla u(\mathbf{y})| \, d\mathcal{H}^1(\mathbf{y})\right)^q} \left(\int_{\partial X_\lambda} \frac{|\langle \mathbf{y}, \nabla u(\mathbf{y})\rangle|^\gamma}{|\nabla u(\mathbf{y})|} \, d\mathcal{H}^1(\mathbf{y})\right)^p,$$

where $m, k, \gamma, p, q \in \mathbf{R}$. If

$$m - (2k-1)q - p = 0,$$

then the above quantity is affine invariant. If

$$2m - (3k-1)q + (\gamma - 1)p = 0,$$

then the quantity scales as s^0. There are infinitely many solutions of these equations. As examples we can take $m = \frac{1}{2}$, $k = \frac{1}{3}$, $q = \frac{5}{2}$, $\gamma = \frac{1}{4}$, $p = \frac{4}{3}$. Another example is given by $m = \frac{1}{2}$, $k = \frac{1}{2}$, $q = 2$, $\gamma = 1$, $p = \frac{1}{2}$. We notice that there are no nonnull solutions with $p = 0$.

4. Selection of descriptors and their implementation.

Using the principles described in section 3, we select in this section a set of affine invariant quantities, we use them to construct descriptors, and we describe the main details of their implementation. We also describe its corresponding quantized level set (QLS) version. Let us first recall the keypoints for which we will compute our descriptors.

4.1. Keypoints detection.

Since our purpose is to compare our descriptors with SIFT on normalized neighborhoods [31] and with ASIFT, which uses an orbit of images [38], we have to work with two types of keypoints. In the first case we use the Harris affine keypoints [31] as used in [32, 33]. In the second, we use SIFT keypoints [28] as in ASIFT. For a short review of these keypoints, we refer the reader to section 2.3.

In order to compute the Harris affine keypoints and their SIFT descriptor, we use the online binary software provided by Mikolajczyk (available at http://www.robots.ox.ac.uk/∼vgg/research/affine/detectors.html#binaries). We used the updated version of the code from 12-6-2007, under the name Detectors & Descriptors.

In order to compute ASIFT's orbit of images and their SIFT descriptor we use the published ASIFT C++ code [54] (available at http://dx.doi.org/10.5201/ipol.2011.my-asift).

4.2. Descriptors.

Let u be an image defined in the domain Ω, a closed rectangle in $\mathbf{R}^2$. Let $\mathbf{x} \in \Omega$ and let $\mathcal{N}_0$ be a neighborhood of zero. Assume that $\mathcal{N}_\mathbf{x} = \mathbf{x} + \mathcal{N}_0 \subset \Omega$. As above, X_λ denotes a connected component of $\{u \geq \lambda\}$. In what follows $\mathbf{x}$ represents a keypoint and $\mathbf{y}$ represents a point in $\mathcal{N}_\mathbf{x} \cap \partial X_\lambda$, i.e., it lies on the level lines intersecting the neighborhood of $\mathbf{x}$. As usual, $\mathcal{H}^1$ denotes the one-dimensional Hausdorff measure. We assume that ∂X_λ is rectifiable [2].

Using the results of section 3, we consider the following four quantities in our experiments:

- $r1 = \frac{|\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|}{|\langle D^2 u(\mathbf{x})(J\nabla u(\mathbf{x})), J\nabla u(\mathbf{x})\rangle|^{\frac{1}{2}}}$,
- $r2 = \frac{|\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|}{|\langle D^2 u(\mathbf{y})(J\nabla u(\mathbf{y})), J\nabla u(\mathbf{y})\rangle|^{\frac{1}{2}}}$,
- $r3 = \frac{|\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|^{\frac{1}{2}}}{(\int_{\partial X_\lambda} |\langle D^2 u(\bar{\mathbf{y}})(J\nabla u(\bar{\mathbf{y}})), J\nabla u(\bar{\mathbf{y}})\rangle|^{\frac{1}{2}} / |\nabla u(\bar{\mathbf{y}})| \, d\mathcal{H}^1(\bar{\mathbf{y}}))^2} (\int_{\partial X_\lambda} |\bar{\mathbf{y}} \cdot \frac{\nabla u(\bar{\mathbf{y}})}{|\nabla u(\bar{\mathbf{y}})|}| \, d\mathcal{H}^1(\bar{\mathbf{y}}))^{\frac{1}{2}}$,
- $r4 = \frac{|\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})|^{\frac{1}{2}}}{(\int_{\partial X_\lambda} |\langle D^2 u(\bar{\mathbf{y}})(J\nabla u(\bar{\mathbf{y}})), J\nabla u(\bar{\mathbf{y}})\rangle|^{\frac{1}{3}} / |\nabla u(\bar{\mathbf{y}})| \, d\mathcal{H}^1(\bar{\mathbf{y}}))^{\frac{5}{2}}} (\int_{\partial X_\lambda} |\bar{\mathbf{y}} \cdot \frac{\nabla u(\bar{\mathbf{y}})}{|\nabla u(\bar{\mathbf{y}})|}|^{\frac{1}{4}} \, d\mathcal{H}^1(\bar{\mathbf{y}}))^{\frac{4}{3}}$.

Remark 3. Let us mention that in all the above quantities, the numerator contains the expression $H_{10}(u, \mathbf{x}, \mathbf{y}) = \nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})$. Thus, the sinus of the angle formed by the two vectors $\nabla u(\mathbf{x})$ and $\nabla u(\mathbf{y})$ appears in all of them. Since this angle is an essential ingredient of SIFT, in some sense we are adding a further justification to it.

The first and second derivatives used by these quantities are computed using centered differences. The level lines of the image are computed using the tree of shapes [37, 12]. In all the quantities we compute, we set a minimum value $\rho = 10^{-3}$ for any denominator quantity q. That is, if $q < \rho$, then we set the quantity $ri = 0$.

Now, to build up a descriptor with the above quantities we use the same structure as SIFT. Given a keypoint $\mathbf{x}$, we consider a neighborhood $\mathcal{N}_{\mathbf{x}}$ of $\mathbf{x}$ (of size 16×16 in common SIFT and of size 41×41 in SIFT+NN), and we divide it into 4×4 blocks. For each block we compute a weighted histogram of directions quantized in 8 angular bins (measuring angles with respect to the dominant orientation at $\mathbf{x}$). In SIFT, the weights are given by the magnitude of the gradient. This time the weights are given by the descriptors ri, $i = 1, 2, 3, 4$. Each coordinate of $v \in \mathbf{R}^{128}$ contains the weight contributions of all pixels of a given block with a given angular orientation. Finally, in practice, to avoid quantization effects, each orientation occurrence is distributed in several neighboring bins. As we shall verify, using the same arrangement of the descriptor as in SIFT, these quantities permit us to improve the results obtained with SIFT.

As in the introduction, we refer to these descriptors as $\mathcal{AD}$.

Let us mention that the computation times for our descriptors are essentially the same as for SIFT. Indeed, the only difference is that now for each keypoint $\mathbf{x}$ we need to compute quantities like $\nabla u(\mathbf{x}) \wedge \nabla u(\mathbf{y})$, where $\mathbf{y} \in \mathcal{N}_{\mathbf{x}}$. This amounts to a constant additional number of operations per keypoint, which does not increase the complexity with respect to SIFT. To give an example, let us mention the running times corresponding to the computation of our descriptors on the Harris affine keypoints of Figure 3(e). All experiments have been run on a computer with a CPU speed of 2.66 GHz. In our implementation, which is not optimized, SIFT takes 49 seconds, $r1$ and $r2$ take 50 seconds, and $r3$ and $r4$ take 83 seconds. Please note that the time difference between $r1$, $r2$, and SIFT is negligible. The extra time consumed by $r3$ and $r4$ is due to the computation of the level lines of the image (this could be optimized by precomputing and storing them).

4.3. The QLS version of the descriptors. Let us introduce a further variant based on the observation that level sets are affine covariant domains in the sense that, if u is a given image and $A \in GL(2, \mathbf{R})^+$, then $\{u_A \geq \lambda\} = A^{-1}\{u \geq \lambda\}$. Assume that $\mathbf{x}$ is a keypoint of an image u and $\mathcal{N}_{\mathbf{x}}$ is a given neighborhood (be it square or the affine normalized one). We want to organize the descriptor taking into account the level set structure of u in $\mathcal{N}_{\mathbf{x}}$. For that we quantize the image u in $\mathcal{N}_{\mathbf{x}}$. To avoid the effect of illumination changes we first equalize the image u in $\mathcal{N}_{\mathbf{x}}$, call it $H_{\mathbf{x}}(u)$, and then define the (bi)level sets $\mathcal{N}_{\mathbf{x}}^{u,j} := \{\mathbf{y} \in \mathcal{N}_{\mathbf{x}} : j\Delta \leq H_{\mathbf{x}}(u)(\mathbf{y}) < (j+1)\Delta\}$, where Δ is a quantization step and $j = 0, 1, \ldots, \frac{256}{\Delta} - 1$. In practice we take $\Delta = 64$, and we have four level sets corresponding to $j = 0, 1, 2, 3$. The descriptor associated to each ri ($i = 1, 2, 3, 4$) is formed by the concatenation of four vectors $v_j \in \mathbf{R}^{128}$. Each coordinate of v_j receives the weights of the pixels $\mathbf{y} \in \mathcal{N}_{\mathbf{x}}^{u,j}$. If $\Delta = 256$, then we have only a vector $v \in \mathbf{R}^{128}$ with the standard organization of SIFT.

As in the introduction, we refer to these descriptors as $\mathcal{AD}$+QLS.

5. Experimental results. Our purpose in this section is to compare our descriptors ri with SIFT on affine normalized neighborhoods [31] and with ASIFT [38]. This is the object of sections 5.5 and 5.6, respectively. We also include in section 5.4 an example of comparison with SIFT on standard neighborhoods. The comparisons will be done using the standard and the QLS versions both for SIFT and for ri.

Let us consider two images $\mathbf{I_l}$ and $\mathbf{I_r}$ defined in the domain Ω. Assume that both contain the image of a planar scene so that there is a homography $\mathcal{H}$ such that $\mathbf{I_r}(\mathbf{x}) = \mathbf{I_l}(\mathcal{H}\mathbf{x})$ for $\mathbf{x} \in \hat{\Omega} \subseteq \Omega$. Since $\mathcal{H}$ can be locally approximated by an affine map, we may assume that $\mathcal{H}$

is an affine transformation. Let us denote by $\mathbf{P_l}$ and $\mathbf{P_r}$ the set of keypoints (see section 4.1) of $\mathbf{I_l}$ and $\mathbf{I_r}$, respectively. Notice that $\mathbf{P_l}$ and $\mathbf{P_r}$ may have different numbers of keypoints, so that there will be keypoints in $\mathbf{I_l}$ without a corresponding one in $\mathbf{I_r}$ and conversely.

Finally, to each keypoint we associate a descriptor which is a vector of 128 coordinates, or of 512 in the case of the QLS versions. The descriptors we consider are SIFT and the descriptors ri, $i = 1, 2, 3, 4$, in their $\mathcal{AD}$ or $\mathcal{AD}$+QLS versions as defined above. Then for each keypoint $p_l \in \mathbf{P_l}$ we look for a matching point $p_r \in \mathbf{P_r}$ using a certain matching strategy. Several of them have been used for performance evaluation [32, 33].

To compare our descriptors with SIFT+NN we follow the experimental protocol proposed in [32, 33]. For that, we first describe the matching strategies (section 5.1), the notion of corresponding regions (section 5.2), and the *precision/recall* curves (section 5.3). The comparison with ASIFT will be done in terms of the significance measure proposed in [38] (see section 5.6).

5.1. Matching strategies. The definition of a match depends on the matching strategy. Assume that to any region A of any of the images $\mathbf{I_l}$ and $\mathbf{I_r}$ we may associate a descriptor D_A, that is, a vector in $\mathbf{R}^N$ for some fixed $N \in \mathbf{N}$. We look into three different matching strategies as proposed in [32]:

1. Threshold based matching (TH): Two regions A of $\mathbf{I_l}$ and B of $\mathbf{I_r}$ are matched if the distance between their descriptors is below a certain threshold. In this strategy, a descriptor can have several matches and several of them can be considered as correct (in the sense that the descriptors are really similar).
2. Nearest neighbor based matching (NN): Two regions A of $\mathbf{I_l}$ and B of $\mathbf{I_r}$ are matched if the descriptor D_B is the nearest neighbor to the descriptor D_A and if the distance $d(D_A, D_B) < threshold$. Please note here that a descriptor can have only one match.
3. Nearest neighbor distance ratio matching (RNN): This strategy, introduced in [28], is similar to NN except that the thresholding is applied to the distance ratio between the first and the second nearest neighbor. With the same example and notation used in NN, let D_C be the second nearest neighbor to D_A; then region A is matched to B if $\frac{||D_A - D_B||}{||D_A - D_C||} < threshold$, where $||\cdot||$ is the Euclidean norm. Note that in this case a descriptor can have only one match.

Note that the NN and RNN matchings are not symmetric concepts with respect to $\mathbf{I_l}$ and $\mathbf{I_r}$. They are computed taking $\mathbf{I_l}$ as a reference image.

5.2. Definition of corresponding regions. In this subsection and subsection 5.3, we are in the context of the comparison of our descriptors with SIFT+NN. Thus we work with Harris affine keypoints with their associated elliptic neighborhood [31]. We also use the terms elliptical region or, simply, region. Assume that we have two keypoints $p_l \in \mathbf{P_l}$ and $p_r \in \mathbf{P_r}$ whose elliptical neighborhoods S_{μ_l} and S_{μ_r} are defined by the shape adaptation matrices μ_l and μ_r, respectively. Let $S_{\mathcal{H}^T \mu_l \mathcal{H}}$ be the image by $\mathcal{H}$ of the elliptical region S_{μ_l}. The two regions S_{μ_l} and S_{μ_r} are said to correspond [32, 33] if the *overlap error* is sufficiently small, that is, if

$$1 - \frac{|S_{\mu_r} \cap S_{\mathcal{H}^T \mu_l \mathcal{H}}|}{|S_{\mu_r} \cup S_{\mathcal{H}^T \mu_l \mathcal{H}}|} < \epsilon. \tag{5.1}$$

Given the homography and the matrices defining the regions, the error is computed numerically by counting the number of pixels in the union and in the intersection of the regions (see [32, 33] for details). In our experiment we choose $\epsilon = 0.5$ as suggested in [32, 33].

5.3. Definition of precision and recall. Assume that we take $\mathbf{I_l}$ as reference image and the matches are computed accordingly. The evaluation criterion used is based on the number of correct matches (true positives (TP)) and the number of false matches (false positives (FP)) obtained for an image pair. Given a matching strategy, we have a correct match of two keypoints when their descriptors satisfy the matching criterion and their elliptical regions correspond according to the overlap criterion (5.1). The other matchings are the false matchings. Both correct and false matchings can be computed since we know the ground truth given by the matrix $\mathcal{H}$. Following [32, 33] we take the overlap error threshold $\epsilon = 0.5$. As argued in those references, there are very few regions that should be matched, have an overlap error greater than 0.5, and pass the matching criterion.

The number of correspondences is counted as the number of possible correct matchings and depends on the matching strategy. Let us explain this. For any region A of $\mathbf{I_l}$, let $N(A)$ be the number of regions of $\mathbf{I_r}$ that correspond to A according to the overlap criterion (5.1). Then, if we use the TH matching strategy, we compute the number of correspondences as $\sum_A N(A)$, while if we use the NN or the RNN matching strategy, we compute it as $\sum_A \min(N(A), 1)$. In all cases the sum is extended to all regions of $\mathbf{I_l}$.

According to the three matching strategies discussed in section 5.1, to match a region A of $\mathbf{I_l}$ to a region B of $\mathbf{I_r}$, a certain distance relation between their descriptors D_A and D_B has to be below a given threshold t. So, given the two images $\mathbf{I_l}$ and $\mathbf{I_r}$ and the threshold t, we compare every descriptor D_l of keypoints in $\mathbf{I_l}$ to every descriptor D_r of keypoints in $\mathbf{I_r}$, and we count the number of TP as well as the number of FP. We repeat this process for different values of t and, in this way, we can study the behavior of the descriptor for different thresholds. Performance of different descriptors is measured using *recall* versus 1-*precision* graphs where *recall* and 1-*precision* are defined as follows:

- $recall = \frac{\#correct\text{-}matches}{\#correspondences}$, where # is read as "the number of."
- 1-$precision = \frac{\#false\text{-}matches}{\#correct\text{-}matches + \#false\text{-}matches} = \frac{FP}{TP+FP}$. Note that the denominator does not take into consideration the *overlap error*. In other words, if the matching algorithm returned k matchings, then $TP + FP = k$ (i.e., $TP + FP$ is independent of the number of correspondences).

Both values depend on t. Please note that *recall* and 1-*precision* are *independent* terms: *recall* is computed with respect to the number of correspondences, and 1-*precision* is computed with respect to the total number of matches returned by the matching algorithm. Now, given *recall* and 1-*precision* measures along with the number of correspondences, we have

$$TP = \#correspondences \times recall,$$

$$FP = \frac{\#correspondences \times recall \times (1\text{-}precision)}{precision}.$$

Remark 4. Note that the *recall* is increasing with t, as is the number of matchings. But this is not the case for 1-*precision*. Thus the 1-*precision*/*recall* curve is not necessarily increasing, although it is often so.

(a) (b) (c)

Figure 2. *The result of a matching between Figures* 3(d) *and* 3(e) *using SIFT keypoints. The descriptors are therefore computed on a square neighborhood. The matching has been done using the RNN matching strategy with a threshold equal to* 0.8. (a) *shows the result of the SIFT descriptor with* 45 *correct matchings and* 33 *false ones.* (b) *shows the result of the* $r1$ *quantity used in a descriptor structure analogous to SIFT with* 56 *correct matchings and* 22 *false ones.* (c) *shows the results of the* $r1$ *quantity used in the descriptor structure based on the level sets as described in section* 4.3*, with* 68 *correct matchings and* 9 *false ones. Images courtesy of K. Mikolajczyk. These images are freely available from* http://www.robots.ox.ac.uk/~vgg/research/affine/index.html.

A perfect descriptor would give a recall equal to 1 for any precision. A horizontal curve in the graph indicates that the recall value is attained with high precision. It also indicates that the detected structures are very similar to each other and the descriptor cannot distinguish between them even when decreasing the precision.

5.4. An example of comparison with SIFT on standard neighborhoods. Although our main comparisons will be done with SIFT+NN [31, 33] and with ASIFT [38], which are SIFT's best performing versions, for the sake of illustration let us show an image to compare SIFT with the descriptors in $\mathcal{AD}$ and $\mathcal{AD}$+QLS. In both cases, we use a square neighborhood. We have chosen the descriptor based on $r1$, although we could have chosen any of the ri. In Figure 2 we show the result of a matching between Figures 3(d) and 3(e) (taken from [32]; see section 5.5). It can be seen that although the quantity $r1$ already contributes to improving the matching result, when combined with the quantization on the level sets we get an even more robust descriptor (having fewer false matchings). In this specific example the number of correct matchings increased from 45 for SIFT to 68 for $\mathcal{AD}$+QLS, whereas the number of false matchings dropped from 33 (SIFT) to 9 ($\mathcal{AD}$+QLS). This behavior is common to all images of the dataset below. Quantitative comparisons will be done in subsequent sections.

5.5. Comparing to SIFT with normalized neighborhoods. In this section we compare our descriptors with SIFT on affine normalized neighborhoods around the Harris affine keypoints [31, 33].

Image dataset. The images we use in Figures 2 and 3 are taken from [33]. They can be downloaded from http://www.robots.ox.ac.uk/~vgg/research/affine/index.html. The images in Figure 9 are also in the public domain and can be downloaded from the IPOL website,

Table 1

The number of Harris affine (HA) keypoints found for every image used in the experiments.

Image	# of HA keypoints
3(a)	5027
3(b)	4120
3(c)	3930
3(d)	2325
3(e)	2675
3(f)	2504
3(g)	2822
3(h)	1748
3(i)	1481
3(j)	1730
3(k)	1334
3(l)	1039

Table 2

The #correspondences between every image pair used in the experiments and for all matching strategies. Note that, according to its definition in section 5.3, the #correspondences is the same for the matching strategies NN and RNN and is much higher for TH.

Image pair	*#correspondences*		
	TH	NN	RNN
3(a)→3(b)	46414	2821	2821
3(a)→3(c)	23631	1633	1633
3(d)→3(e)	13016	990	990
3(d)→3(f)	5520	493	493
3(g)→3(h)	10189	739	739
3(g)→3(i)	3840	288	288
3(j)→3(k)	12012	1161	1161
3(j)→3(l)	9686	929	929

http://www.ipol.im/pub/algo/my_affine_sift/. Without loss of generality, and to reduce the running time of the experiments (mostly due to the generation of Figures 4 to 8 using the MATLAB code provided in the above website, whose execution time depends on the number of keypoints), we downscale the images by a factor of 2.

We chose four sets of images, three of them containing a reference image and two simulated affine distortions. The fourth set contains the reference image and two simulated illumination changes. The first set contains different views of a textured scene; again we compare the frontal view, Figure 3(a), with a 50° and a 70° tilt viewpoint change, shown in Figures 3(b) and 3(c). The second set contains different views of a structured scene; we compare the frontal view, Figure 3(d), with a 50° and a 70° tilt viewpoint change, shown in Figures 3(e) and 3(f), respectively. The third set contains simulations of rotations and zooms; we compare Figure 3(g) with Figure 3(h) and then with Figure 3(i). Finally, the fourth set contains a reference image and two simulations of illumination changes; we compare Figure 3(j) with Figure 3(k) and then with Figure 3(l). More information about the set of images and their acquisition can be found in [33].

The number of Harris affine keypoints of each image is shown in Table 1. The number of *correspondences* between each image pair compared is shown in Table 2. We have used an

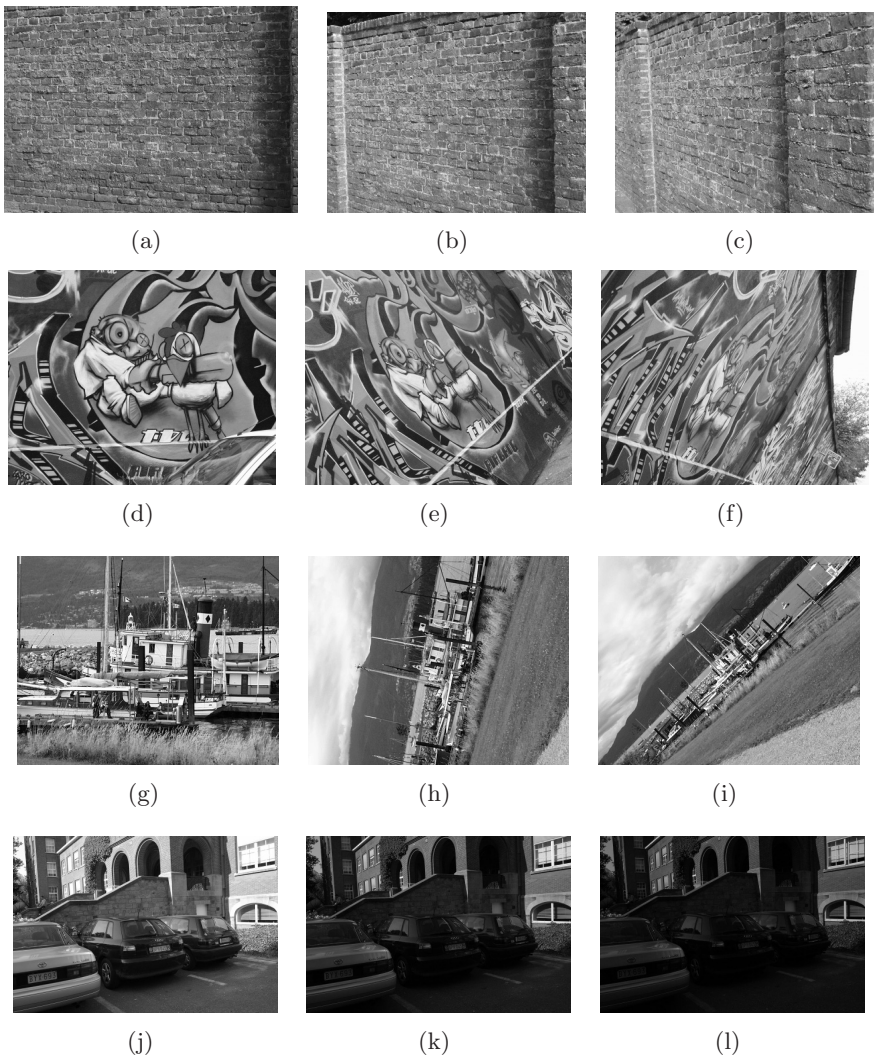


Figure 3. *The images used to compare $\mathcal{AD}$ and $\mathcal{AD}$+QLS to SIFT+NN. A textured scene where* (a) *is a front view,* (b) *an approximately* 50° *change in viewpoint, and* (c) *an approximately* 70° *change in viewpoint. A structured scene where* (d) *is a front view,* (e) *an approximately* 50° *change in viewpoint, and* (f) *an approximately* 70° *change in viewpoint.* (g)–(i) *increasing rotation and zoom factors.* (j)–(l) *increasing illumination change. Images courtesy of K. Mikolajczyk. These images are freely available from* http://www.robots.ox.ac.uk/~vgg/research/affine/index.html.

overlap error parameter $\epsilon = 0.5$ and the three matching strategies {RNN, NN, TH}. Note that, according to its definition in section 5.3, the number of correspondences is the same for the matching strategies NN and RNN and is much higher for TH.

Experiments. First we compare our descriptors $\mathcal{AD}$ with SIFT+NN using in both cases the affine normalized neighborhood. We display the 1-*precision*/*recall* curves for the four images in Figure 3 for all matching strategies.

In a second experiment we compare $\mathcal{AD}$, $\mathcal{AD}$+QLS, and SIFT+NN (using again the normalized neighborhood). The purpose is to compare the performance added by the QLS

strategy. Since all quantities ri behave similarly under the different matching strategies, we just show the results obtained with $r1$ and RNN.

In any case, for each pair of images and for each *matching strategy* we compute the number of correspondences, the number of TP, and the total number of returned matches (TP+FP), varying the threshold t. We used the same code used in [32], which can be found at http://www.robots.ox.ac.uk/∼vgg/research/affine/desc_evaluation.html#code.

Let us show the figures corresponding to the comparison of $\mathcal{AD}$ with SIFT+NN. Figure 4 shows the 1-*precision*/*recall* curves corresponding to the matching of images Figure 3(a) with Figure 3(b) (left column) and of Figure 3(a) with Figure 3(c) (right column). In each column we show the results corresponding to the three different matching strategies {TH, NN, RNN}. Figure 5 shows the results corresponding to the matching of Figure 3(d) with Figure 3(e) and of Figure 3(d) with Figure 3(f). Figure 6 shows the results corresponding to the matching of Figures 3(g) and 3(h), and Figures 3(g) and 3(i). Figure 7 shows the result of matching Figures 3(j) and 3(k), and Figures 3(j) and 3(l).

In general, our descriptors perform better than SIFT; in particular, in Figure 5(c) there is an improvement by a factor of 2. Notice that Figure 5(d) (which corresponds to a large change of viewpoint) shows that when SIFT could retrieve only 5% of TP at a low precision, we retrieved up to 15% of TP. Let us note that, in most cases, the ranking of the descriptors does not change between one matching strategy and another.

In order to illustrate how the curves displayed in the above figures are generated, we show Table 3. To generate these curves, we need the number of TP and TP+FP for each experiment. Table 3 contains these numbers for one of the experiments of Figure 5, namely the matching of Figure 3(d) to Figure 3(e) using the RNN *matching strategy*.

In order to show the relative performance of $\mathcal{AD}$ and $\mathcal{AD}$+QLS, we show in Figure 8 the 1-precision/recall curves corresponding to all images in Figure 3 for $\mathcal{AD}$, $\mathcal{AD}$+QLS, and SIFT+NN. Again, our descriptors are based on $r1$ and the RNN matching strategy. As it can be seen, in some cases, especially when there are big affine changes, the $\mathcal{AD}$+QLS version provides improvements over $\mathcal{AD}$.

5.6. Comparing to ASIFT. In this experiment we compare our descriptors $\mathcal{AD}$ with SIFT in the context of ASIFT [38] (available at http://dx.doi.org/10.5201/ipol.2011.my-asift). Thus we match $\mathbf{I}_l$ with $\mathbf{I}_r$ using the orbit of images generated by ASIFT for both of them. To test the robustness of SIFT and our descriptors, we compare them on different orbit sizes. We start with the suggested orbit size (ASIFT default) made out of 7 tilts and generating 61 images. Then we reduce the orbit to 5 tilts (27 simulations) and 3 tilts (10 simulations). Throughout these experiments we use the $r1$ quantity. The other quantities behave similarly. Both SIFT and our descriptors are computed on the same set of SIFT keypoints (see section 2.3). For both we use the RNN matching strategy. Thus, for both of them the conditions are equal, except that SIFT is used for ASIFT and we use $\mathcal{AD}$ based on $r1$.

The tests are performed on five image pairs, shown in Figure 9 and listed in Table 4. We compare the number of matches obtained with each method and also the significance of those matches. The significance measure is computed as the logarithm of the NFA (number of false alarms), a quantity obtained during the computation of the affine map relating both images. The affine map is computed using an a contrario optimized version of RANSAC, as in [35]. We use the same implementation as in ASIFT.

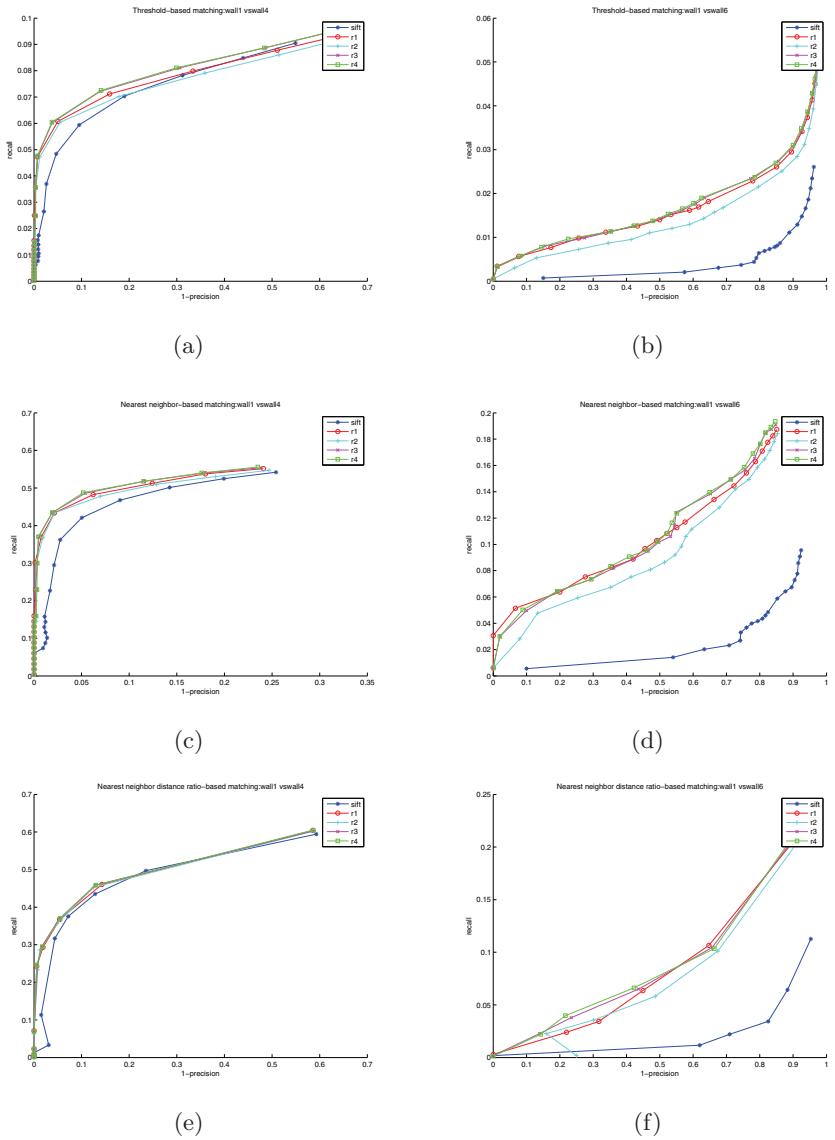


Figure 4. *Comparison between SIFT+NN and our descriptors $\mathcal{AD}$ computed on an affine normalized neighborhood. "1-precision versus recall" graphs showing the results of matching Figure* 3(a) *with* 3(b) *and Figure* 3(a) *with* 3(c)*, in the first and second columns, respectively. The left column shows the matching of Figure* 3(a) *with* 3(b): (a) *using the TH matching strategy,* (c) *using the NN matching strategy, and* (e) *using the RNN matching strategy. The right column shows the matching of Figure* 3(a) *with* 3(c): (b) *using the TH matching strategy,* (d) *using the NN matching strategy, and* (f) *using the RNN matching strategy.*

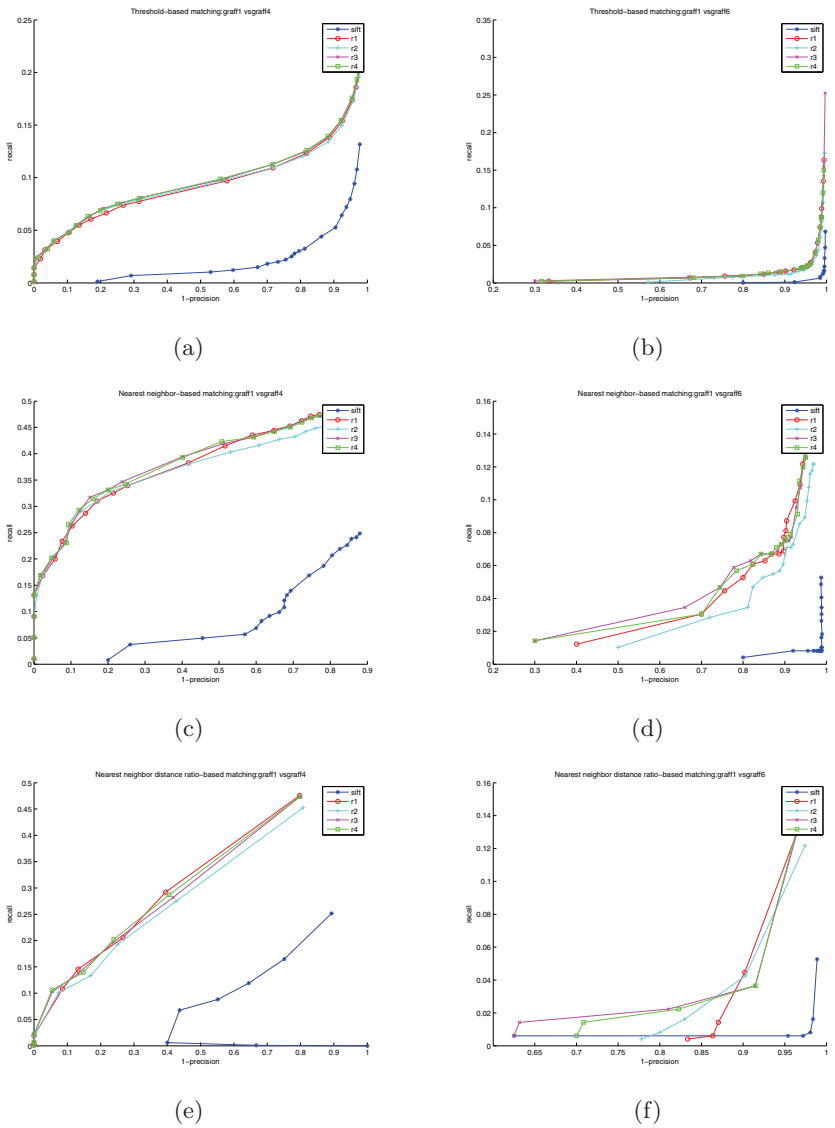


Figure 5. *Comparison between SIFT+NN and our descriptors $\mathcal{AD}$ computed on an affine normalized neighborhood. "1-precision versus recall" graphs showing the results of matching Figure* 3(d) *with* 3(e) *and Figure* 3(d) *with* 3(f)*, in the first and second columns, respectively. The left column shows the matching of Figure* 3(d) *with* 3(e): (a) *using the TH matching strategy,* (c) *using the NN matching strategy, and* (e) *using the RNN matching strategy. The right column shows the matching of Figure* 3(d) *with* 3(f): (b) *using the TH matching strategy,* (d) *using the NN matching strategy, and* (f) *using the RNN matching strategy.*

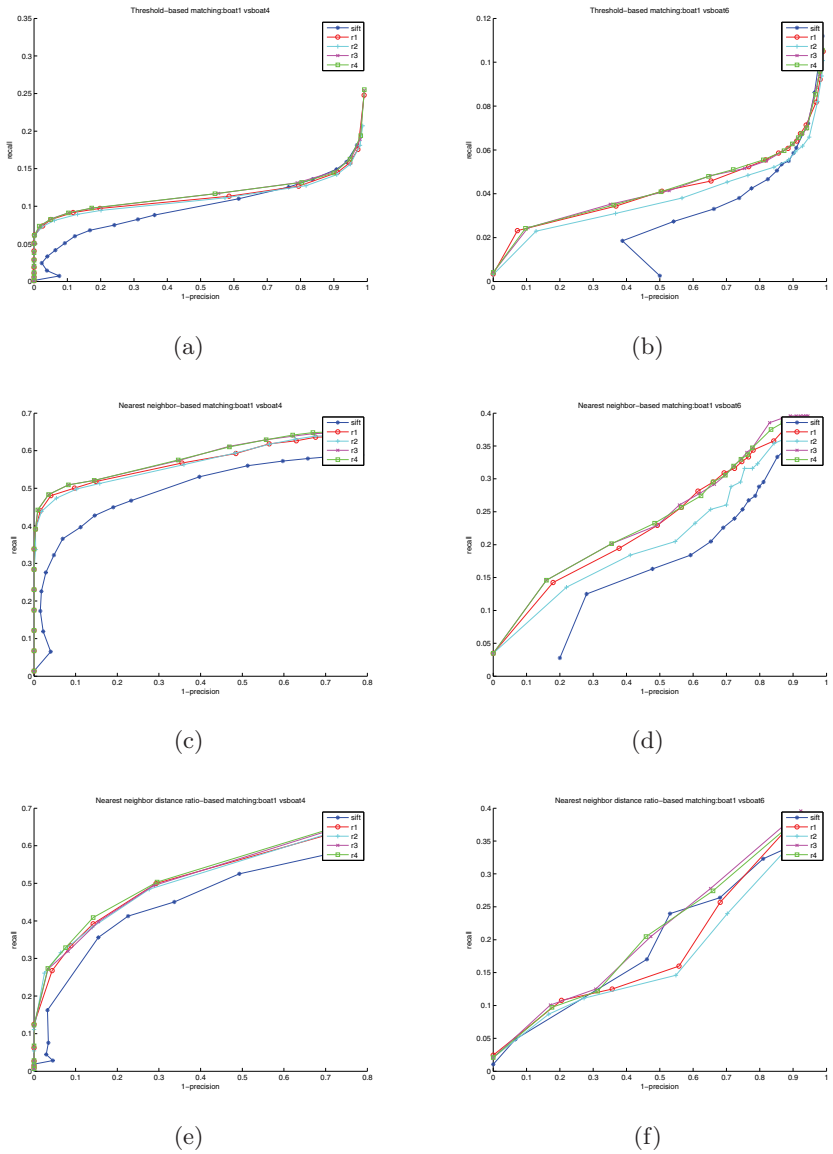


Figure 6. *Comparison between $SIFT{+}NN$ and our descriptors $\mathcal{AD}$ computed on an affine normalized neighborhood. "1-precision versus recall" graphs showing the results of matching Figure* 3(g) *with* 3(h) *and Figure* 3(g) *with* 3(i)*, in the first and second columns, respectively. The left column shows the matching of Figure* 3(g) *with* 3(h): (a) *using the TH matching strategy,* (c) *using the NN matching strategy, and* (e) *using the RNN matching strategy. The right column shows the matching of Figure* 3(g) *with* 3(i): (b) *using the TH matching strategy,* (d) *using the NN matching strategy, and* (f) *using the RNN matching strategy.*

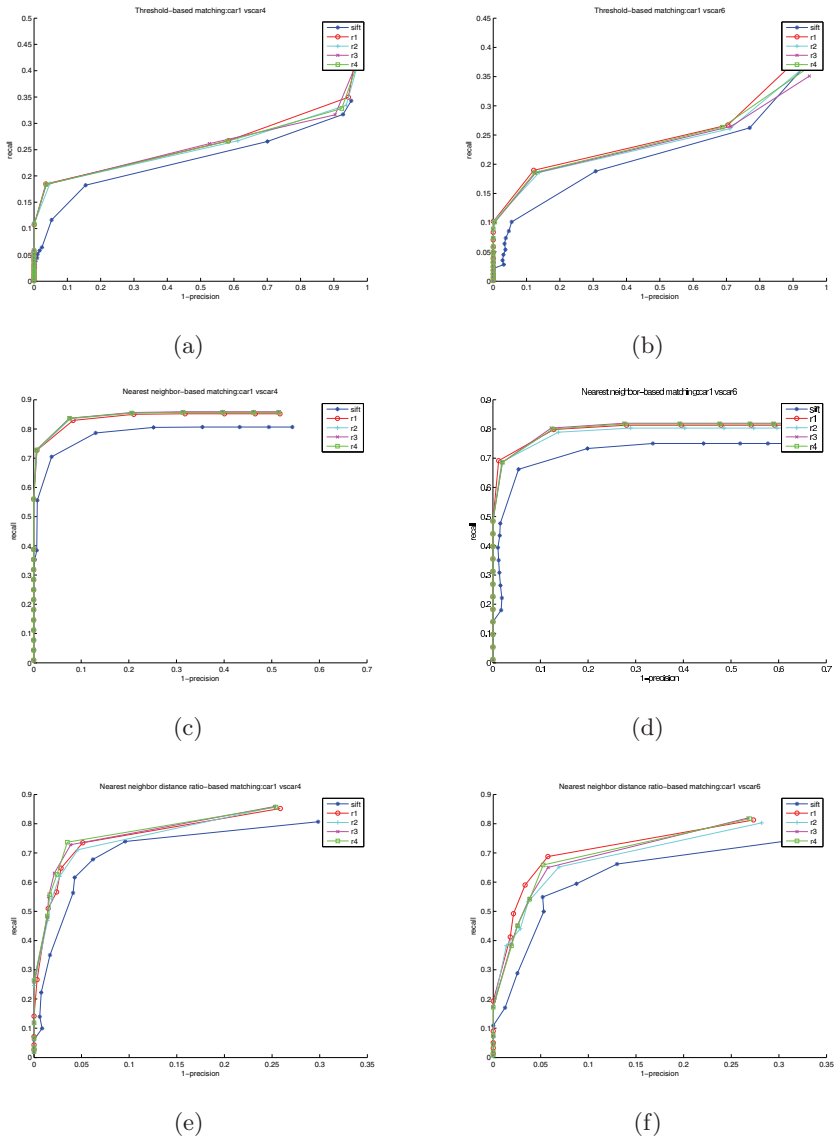


Figure 7. *Comparison between SIFT+NN and our descriptors $\mathcal{AD}$ computed on an affine normalized neighborhood. "1-precision versus recall" graphs showing the results of matching Figure 3(j) with 3(k) and Figure 3(j) with 3(l), in the first and second columns, respectively. The left column shows the matching of Figure 3(j) with 3(k):* (a) *using the TH matching strategy,* (c) *using the NN matching strategy, and* (e) *using the RNN matching strategy. The right column shows the matching of Figure 3(j) with 3(l):* (b) *using the TH matching strategy,* (d) *using the NN matching strategy, and* (f) *using the RNN matching strategy.*

Table 3

Results of an experiment where the input images are Figures 3(d) and 3(e), comparing our 4 $\mathcal{AD}$ descriptors to SIFT, using in both cases an affine normalized neighborhood.

Sample experiment Input: Figures 3(d) and 3(e) Chosen matching strategy: Nearest neighbor distance ratio matching (RNN) Other chosen values: *overlap error* = 0.5 (yielding *#correspondences* = 990)											
Threshold t =		0.3704	0.4167	0.4762	0.5556	0.6667	0.8333	0.8696	0.9091	0.9524	1.0000
SIFT	TP	0	0	0	1	6	67	87	118	163	249
	TP + FP	0	0	1	3	10	119	194	332	655	2325
$r1$	TP	1	2	2	5	15	102	140	189	271	460
	TP + FP	1	2	2	5	15	109	162	248	474	2325
$r2$	TP	1	2	2	3	12	93	130	180	274	464
	TP + FP	1	2	2	3	12	106	152	237	461	2325
$r3$	TP	0	2	2	5	19	98	134	183	276	464
	TP + FP	0	2	2	5	19	109	155	236	480	2325
$r4$	TP	0	2	2	5	20	96	129	183	270	465
	TP + FP	0	2	2	5	20	108	152	243	465	2325

In Table 4 we show the results of the comparison between ASIFT and the descriptor $\mathcal{AD}$ based on $r1$ computed on a square neighborhood. We show the results corresponding to three orbit sizes. We can see the improvement due to the use of the new descriptor both in the number of matchings and in their significance measure. This improvement is clearly visible for a reduced orbit; when using $r1$ we get a matching in four of five images, but only in two of them when using SIFT.

The performance of $\mathcal{AD}$+QLS versus ASIFT is similar to the performance of $\mathcal{AD}$ versus ASIFT, and therefore we omitted the table. Let us mention only that when using $\mathcal{AD}$+QLS we get a matching for the five image pairs in the case of a reduced orbit with 10 simulated images. On the other hand, the reason for the similar behavior of $\mathcal{AD}$ versus $\mathcal{AD}$+QLS in the context of this comparison may be that the information brought by the orbit is sufficient to cancel the benefits gained by the QLS strategy. This is in contrast to the behavior of QLS in the context of SIFT+NN.

6. Conclusions. Using a classical result on algebraic invariants of the unimodular group, we have generated in this paper some geometric affine invariant quantities that we used to construct distinctive descriptors which are robust with respect to affine transforms caused by the camera change of position, although full camera affine invariance cannot be guaranteed due to the domain problem (that is, the difficulty of finding an affine covariant domain) and the camera blur. To alleviate the domain problem, based on the observation that the level sets of an image are geometric affine covariant, we have also considered a quantized level set version of the descriptors. We have embedded these quantities into the algorithmic structure of SIFT. We have compared them with both SIFT with an affine normalized neighborhood (SIFT+NN), and with ASIFT, which addresses both the domain and camera blur problems by cleverly sampling the orbit of affine transformations of the images. When comparing with SIFT+NN, we have used the same affine normalized neighborhood; when comparing with ASIFT we have used the same orbit of images. In this way we inherit the same camera affine

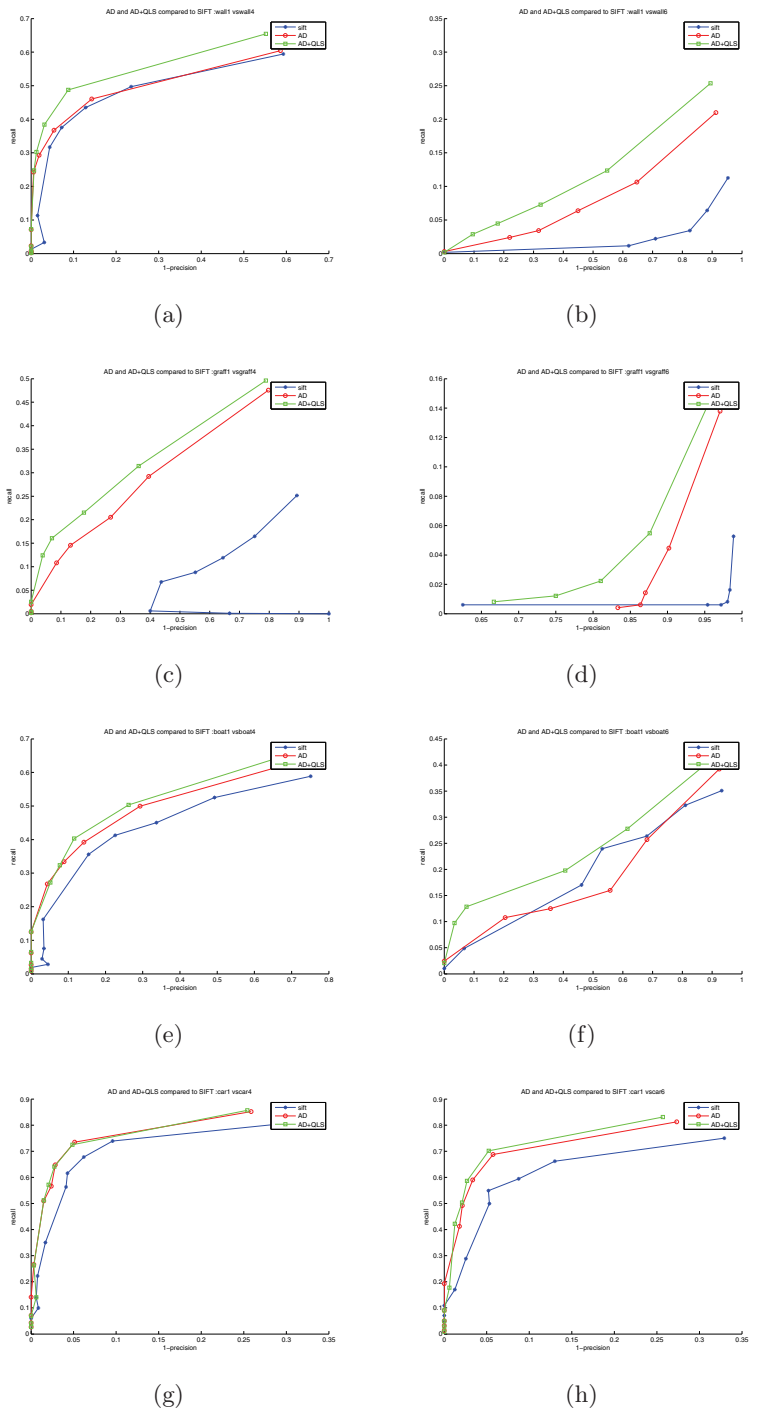
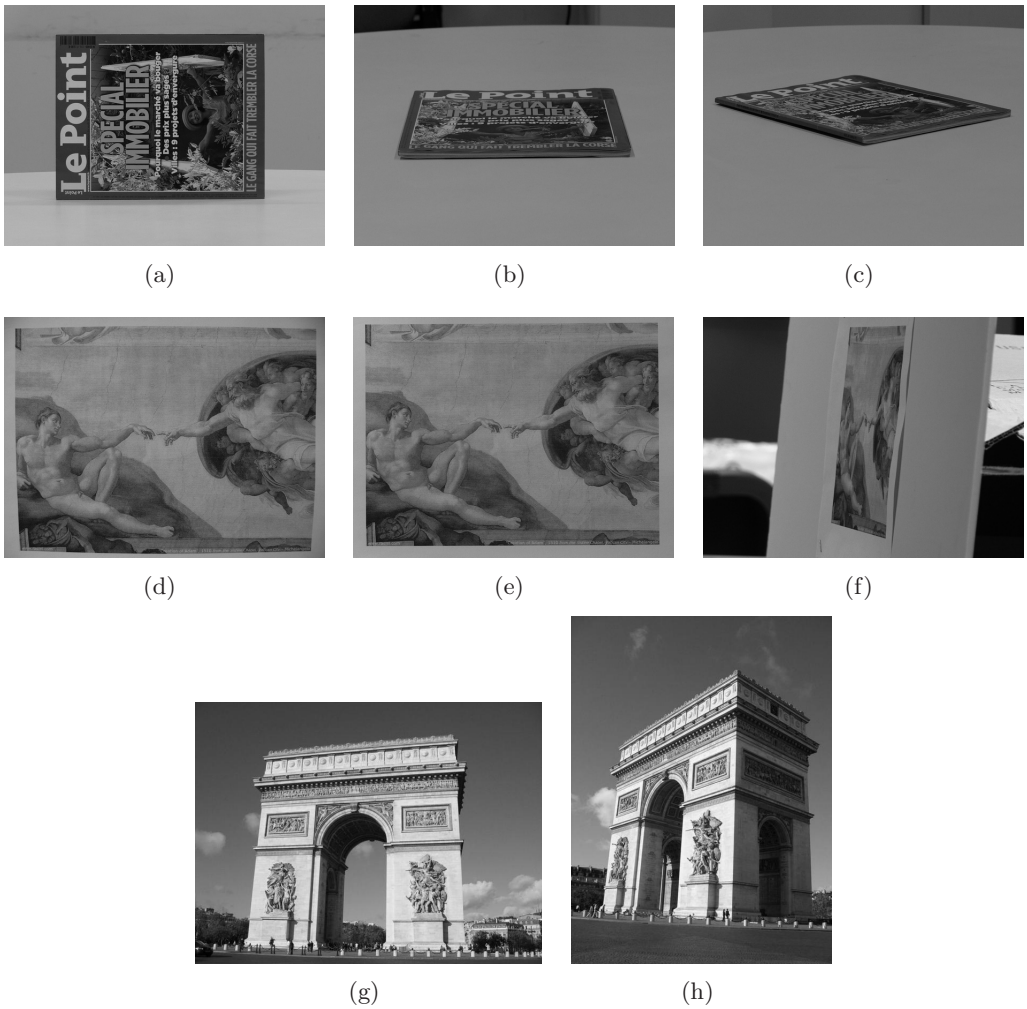


Figure 8. *Comparison between $SIFT{+}NN$ and the descriptors $\mathcal{AD}$ and $\mathcal{AD}{+}QLS$ based on* r1. *"1-precision versus recall" graphs showing the results of matching using the RNN matching strategy.* (a) *and* (b) *show the result of matching Figure* 3(a) *with* 3(b) *and* 3(c)*, respectively.* (c) *and* (d) *show the result of matching Figure* 3(d) *with* 3(e) *and* 3(f)*, respectively.* (e) *and* (f) *show the result of matching Figure* 3(g) *with* 3(h) *and* 3(i)*, respectively.* (g) *and* (h) *show the result of matching Figure* 3(j) *with* 3(k) *and* 3(l)*, respectively.*



(a) (b) (c)

(d) (e) (f)

(g) (h)

Figure 9. *Image pairs used as an input in the experiments comparing our descriptors to SIFT in the context of ASIFT.* (a) *frontal view of a magazine;* (b) *a transition tilt of* 4 *with a* 90° *rotation applied to the magazine;* (c) *a transition tilt of* 4 *with a* 50° *rotation applied to the magazine;* (d) *an image of a painting with no optical zoom;* (e) *image of the same painting with* 10× *optical zoom;* (f) *an* 80° *viewpoint change to the painting image with* 10× *optical zoom; and finally,* (g) *and* (h) *illustrate a case of two images taken from different viewpoints of a very big three-dimensional object. These images are freely available from the IPOL website,* http://www.ipol.im/pub/algo/my_affine_sift/.

Table 4

Table comparing the result of matching an image pair using SIFT computed on an orbit of images and the $\mathcal{AD}$ based on the $r1$ descriptor on the same orbit. The results obtained using $\mathcal{AD}$+QLS are similar to those obtained using $\mathcal{AD}$, except that now we obtain a matching for the 5 image pairs in the case of a reduced orbit with 10 simulated images.

Image pair	Descriptor	Number of tilts					
		7 (61 simulations)		5 (27 simulations)		3 (10 simulations)	
		Matches	log(NFA)	Matches	log(NFA)	Matches	log(NFA)
9(a)→9(b)	SIFT	234	-322.685	140	-181.15	14	-4.33
	$r1$	309	-418.38	182	-247.03	24	-16.7
9(a)→9(c)	SIFT	98	-125.84	68	-87.72	no match	0
	$r1$	145	-192.489	101	-129.25	12	-2.45
9(g)→9(h)	SIFT	173	-185.27	113	-120.57	65	-79.05
	$r1$	318	-373.025	349	-417.36	181	-226.47
9(d)→9(f)	SIFT	53	-54.14	29	-26.2	no match	0
	$r1$	67	-71.95	63	-57.1	no match	0
9(e)→9(f)	SIFT	62	-65.23	42	-48.08	no match	0
	$r1$	97	-96.31	76	-69.49	14	-3.34

invariance of these methods. In both cases, our comparison shows that the proposed descriptors behave more robustly than SIFT with respect to affine deformations induced by camera change of position. In particular, we are still able to match images given a small orbit of affine transformations, while SIFT is not.

REFERENCES

[1] L. Alvarez, F. Guichard, P.L. Lions, and J.M. Morel, *Axioms and fundamental equations of image processing*, Arch. Rational Mech. Anal., 123 (1993), pp. 199–257.

[2] L. Ambrosio, N. Fusco, and D. Pallara, *Functions of Bounded Variation and Free Discontinuity Problems*, The Clarendon Press, Oxford University Press, New York, 2000.

[3] C. Ballester, *Affine Invariant Segmentation by Variational Method*, Ph.D. thesis, Department of Mathematics and Computer Science, University of Balearic Islands, Palma, Mallorca, 1995.

[4] C. Ballester, V. Caselles, and M. González, *Affine invariant segmentation by variational method*, SIAM J. Appl. Math., 56 (1996), pp. 294–325.

[5] H. Bay, T. Tuytelaars, and L. Van Gool, *SURF: Speeded up robust features*, in Proceedings of the 9th European Conference on Computer Vision (ECCV 2006), 2006, pp. 404–417.

[6] A. Bosch, A. Zisserman, and X. Munoz, *Image classification using random forests and ferns*, in Proceedings of the 11th International IEEE Conference on Computer Vision, Rio de Janeiro, Brazil, 2007.

[7] A. Bosch, A. Zisserman, and X. Muñoz, *Scene classification using a hybrid generative/discriminative approach*, IEEE Trans. Pattern Anal. Mach. Intell., 30 (2008), pp. 712–727.

[8] M. Brown and D.G. Lowe, *Invariant features from interest point groups*, in Proceedings of the British Machine Vision Conference, Cardiff, Wales, 2002, pp. 656–665.

[9] A.M. Bruckstein, R.J. Holt, A.N. Netravali, and T.J. Richardson, *Invariant signatures for planar shape recognition under partial occlusion*, CVGIP: Image Underst., 58 (1993), pp. 49–65.

[10] F. Cao, J.-L. Lisani, J.-M. Morel, P. Musé, and F. Sur, *A Theory of Shape Identification*, Springer-Verlag, New York, 2008.

[11] V. Caselles and P. Monasse, *Grain filters*, J. Math. Imaging Vision, 17 (2002), pp. 249–270.

[12] V. Caselles and P. Monasse, *Geometric Description of Topographic Maps and Applications to Image Processing*, Lecture Notes in Math. 1984, Springer-Verlag, Berlin, Heidelberg, 2010.

[13] N. Dalai and B. Triggs, *Histogram of oriented gradients for human detection*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 2005, pp. 886–893.

[14] J. Gårding and T. Lindeberg, *Direct computation of shape cues using scale-adapted spatial derivative operators*, Int. J. Comput. Vision, 17 (1996), pp. 163–191.

[15] C. Harris and M. Stephens, *A combined corner and edge detector*, in Proceedings of the Fourth Alvey Vision Conference, Vol. 15, Manchester, UK, 1988, pp. 147–151.

[16] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, UK, 2003.

[17] A.E. Johnson and M. Hebert, *Using spin images for efficient object recognition in cluttered 3D scenes*, IEEE Trans. Pattern Anal. Mach. Intell., 21 (1999), pp. 433–449.

[18] Y. Ke and R. Sukthankar, *PCA-SIFT: A more distinctive representation for local image descriptors*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, 2004, pp. 506–513.

[19] R. Kimmel, C. Zhang, A.M. Bronstein, and M.M. Bronstein, *Are MSER features really interesting?*, IEEE Trans. Pattern Anal. Mach. Intell., 33 (2011), pp. 2316–2320.

[20] S. Lazebnik, C. Schmid, and J. Ponce, *Semi-local affine parts for object recognition*, in Proceedings of the British Machine Vision Conference, Vol. 2, 2004, pp. 959–968.

[21] V. Lepetit and P. Fua, *Keypoint recognition using randomized trees*, IEEE Trans. Pattern Anal. Mach. Intell., 28 (2006), pp. 1465–1479.

[22] V. Lepetit, P. Lagger, and P. Fua, *Randomized trees for real-time keypoint recognition*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 2, 2005, pp. 775–781.

[23] T. Lindeberg, *Feature detection with automatic scale selection*, Int. J. Comput. Vision, 30 (1998), pp. 79–116.

[24] T. Lindeberg and J. Gårding, *Shape-adapted smoothing in estimation of 3-D shape cues from affine deformations of local 2-D brightness structure*, Image Vision Comput., 15 (1997), pp. 415–434.

[25] J.L. Lisani, L. Moisan, P. Monasse, and J.M. Morel, *On the theory of planar shape*, Multiscale Model. Simul., 1 (2003), pp. 1–24.

[26] C. Liu, J. Yuen, A. Torralba, J. Sivic, and W. Freeman, *SIFT flow: Dense correspondence across different scenes*, in Proceedings of the 10th European Conference on Computer Vision (ECCV 2008), 2008, pp. 28–42.

[27] D. G. Lowe, *Object recognition from local scale-invariant features*, in Proceedings of the International Conference on Computer Vision, Vol. 2, IEEE Computer Society, Washington, DC, 1999, pp. 1150–1157.

[28] D. G. Lowe, *Distinctive image features from scale-invariant keypoints*, Int. J. Comput. Vision, 60 (2004), pp. 91–110.

[29] J. Matas, O. Chum, M. Urban, and T. Pajdla, *Robust wide-baseline stereo from maximally stable extremal regions*, Image Vision Comput., 22 (2004), pp. 761–767.

[30] E. Meinhardt, *Morphological and Statistical Techniques for the Analysis of 3D Images*, Ph.D. thesis, Department of Information and Communication Technologies, University of Pompeu Fabra, Barcelona, Spain, 2010.

[31] K. Mikolajczyk and C. Schmid, *Scale & affine invariant interest point detectors*, Int. J. Comput. Vision, 60 (2004), pp. 63–86.

[32] K. Mikolajczyk and C. Schmid, *A performance evaluation of local descriptors*, IEEE Trans. Pattern Anal. Mach. Intell., 27 (2005), pp. 1615–1630.

[33] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool, *A comparison of affine region detectors*, Int. J. Comput. Vision, 65 (2005), pp. 43–72.

[34] F. Mindru, T. Moons, and L. Van Gool, *Recognizing color patterns irrespective of viewpoint and illumination*, in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 1999.

[35] L. Moisan and B. Stival, *A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix*, Int. J. Comput. Vision, 57 (2004), pp. 201–218.

[36] P. Monasse, *Contrast invariant registration of images*, in Proceedings of the 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 6, 1999, pp. 3221–3224.
[37] P. Monasse and F. Guichard, *Fast computation of a contrast-invariant image representation*, IEEE Trans. Image Process., 9 (2000), pp. 860–872.
[38] J.-M. Morel and G. Yu, *ASIFT: A new framework for fully affine invariant image comparison*, SIAM J. Imaging Sci., 2 (2009), pp. 438–469.
[39] J.-M. Morel and G. Yu, *Is SIFT scale invariant?*, Inverse Probl. Imaging, 5 (2011), pp. 115–136.
[40] P.J. Olver, G. Sapiro, and A. Tannenbaum, *Differential invariant signatures and flows in computer vision: A symmetry group approach*, in Geometry Driven Diffusion in Computer Vision, B.M. ter Haar Romeny, ed., Kluwer Academic, Dordrecht, The Netherlands, 1994, pp. 255–306.
[41] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua, *Fast keypoint recognition using random ferns*, IEEE Trans. Pattern Anal. Mach. Intell., 32 (2010), pp. 448–461.
[42] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, *Object retrieval with large vocabularies and fast spatial matching*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.
[43] D. Pritchard and W. Heidrich, *Cloth motion capture*, Comput. Graphics Forum, 22 (2003), pp. 263–271.
[44] E. Rosten and T. Drummond, *Machine learning for high-speed corner detection*, in Proceedings of the 9th European Conference on Computer Vision (ECCV 2006), 2006, pp. 430–443.
[45] P. Salembier and J. Serra, *Flat zones filtering, connected operators, and filters by reconstruction*, IEEE Trans. Image Process., 4 (1995), pp. 1153–1160.
[46] G. Sapiro and A. Tannenbaum, *Affine invariant scale-space*, Int. J. Comput. Vision, 11 (1993), pp. 25–44.
[47] G. Sapiro and A. Tannenbaum, *On affine plane curve evolution*, J. Funct. Anal., 119 (1994), pp. 79–120.
[48] M.S. Sarfraz and O. Hellwich, *An efficient front-end facial pose estimation system for face recognition*, Pattern Recognition Image Anal., 18 (2008), pp. 434–441.
[49] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.
[50] J. Sivic and A. Zisserman, *Video Google: A text retrieval approach to object matching in videos*, in Proceedings of the Ninth IEEE International Conference on Computer Vision, 2003, pp. 1470–1477.
[51] T. Tuytelaars and L. Van Gool, *Content-based image retrieval based on local affinely invariant regions*, in Visual Information and Information Systems, Lecture Notes in Comput. Sci. 1614, Springer-Verlag, Berlin, Heidelberg, 1999, pp. 493–500.
[52] L.J. Van Gool, T. Tuytelaars, and A. Turina, *Local features for image retrieval*, in State-of-the-Art in Content-Based Image and Video Retrieval (Dagstuhl Seminar, 1999), Kluwer, Deventer, The Netherlands, 2001, pp. 21–41.
[53] H. Weyl, *The Classical Groups: Their Invariants and Representations*, 15th ed., Princeton University Press, Princeton, NJ, 1997.
[54] G. Yu and J.-M. Morel, *ASIFT: An Algorithm for Fully Affine Invariant Comparison*, Image Processing On Line, 2011, http://www.ipol.im/pub/algo/my_affine_sift/.