Gauge-SURF Descriptors

Pablo F. Alcantarilla, Luis M. Bergasa, Andrew Davison

{*pablo.alcantarilla,bergasa*}*@depeca.uah.es*

{*ajd*}*@doc.ic.ac.uk*

Abstract

In this paper, we present a novel family of multiscale local feature descriptors, a theoretically and intuitively well justified variant of SURF which is straightforward to implement but which nevertheless is capable of demonstrably better performance with comparable computational cost. Our family of descriptors, called Gauge-SURF (G-SURF), are based on second-order multiscale gauge derivatives. While the standard derivatives used to build a SURF descriptor are all relative to a single chosen orientation, gauge derivatives are evaluated relative to the gradient direction at every pixel. Like standard SURF descriptors, G-SURF descriptors are fast to compute due to the use of integral images, but have extra matching robustness due to the extra invariance offered by gauge derivatives. We present extensive experimental image matching results on the Mikolajczyk and Schmid dataset which show the clear advantages of our family of descriptors against first-order local derivatives based descriptors such as: SURF, Modified-SURF (M-SURF) and SIFT, in both standard and upright forms. In addition, we also show experimental results on large-scale 3D Structure from Motion (SfM) and visual categorization applications.

Keywords: Gauge coordinates, scale space, feature descriptors, integral

image

1. Introduction

Given two images of the same scene, image matching is the problem of establishing correspondence and is a core component of all sorts of computer vision systems, particularly in classic problems such as Structure from Motion (SfM) [1], visual categorization [2] or object recognition [3]. There has been a wealth of work in particular on matching image keypoints, and the key advances have been in multiscale feature detectors and invariant descriptors which permit robust matching even under significant changes in viewing conditions.

We have studied the use of gauge coordinates [4] for image matching and SfM applications and incorporated them into a Speeded-Up Robust Features (SURF) [5] descriptor framework to produce a family of descriptors of different dimensions which we call Gauge-SURF (G-SURF) descriptors. With gauge coordinates, every pixel in the image is described in such a way that if we have the same 2D local structure, the description of the structure is always the same, even if the image is rotated. This is possible since multiscale gauge derivatives are rotation and translation invariant. In addition, gauge derivatives play a key-role in the formulation of non-linear diffusion processes, as will be explained in Section 3.1. By using gauge derivatives, we can make blurring locally adaptive to the image itself, without affecting image details.

The G-SURF descriptors are very related to non-linear diffusion [6, 7] processes in image processing and computer vision. In the typical Gaussian scale-space [8] framework, details are blurred during evolution (i.e. the convolution of the original image with Gaussian kernels of increasing standard deviation). The advantage of blurring is the removal of noise, but relevant image structures like edges are blurred and drift away from their original locations during evolution. In general, a good solution should be to make the blurring locally adaptive to the image yielding the blurring of noise, while retaining details or edges. Instead of local first-order spatial derivatives, G-SURF descriptors measure per pixel information about image blurring and edge or detail enhancing, resulting in a more discriminative descriptors.

We have obtained notable results in an extensive image matching evaluation using the standard evaluation framework of Mikolajczyk and Schmid [9]. In addition, we have tested our family of descriptors in large-scale 3D SfM datasets [10] and visual categorization experiments [2] with satisfactory results. Our results show that G-SURF descriptors outperform or approximate state of the art methods in accuracy while exhibiting low computational demands making it suitable for real-time applications.

We are interested in robust multiscale feature descriptors, to reliably match two images in real-time for visual odometry [11] and large-scale 3D SfM [10] applications. Image matching here, is in fact a difficult task to solve due to the large motion between frames and the high variability of camera movements. For this purpose, we need desciptors that are fast to compute and at the same time exhibit high performance.

In addition, we have created an open-source library called *OpenGSURF* that contains all the family of G-SURF descriptors and we plan to make it publicly available. This family of descriptors comprises several descriptors of different dimensions based on second-order multiscale gauge derivatives. Depending on the application some descriptors may be preferred instead of others. For example, for real-time applications a low-dimensional descriptor should be preferred instead of a high-dimensional one, whereas for image-matching applications considering severe image transformations one can expect a higher recall by using high-dimensional descriptors. To the best of our knowledge, this is the first open source library that allows the user to choose between different dimensional descriptors. Current open source descriptors libraries [12, 13] just have implementations for the standard SURF and Scale Invariant Feature Transform (SIFT) [14] descriptors' default dimensions (64 and 128 respectively). This can be a limitation and a computational bottleneck for some real-time applications that do not necessarily need those default descriptor dimensions.

The rest of the paper is organized as follows: Related work is described in Section 2. Gauge coordinates are introduced in Section 3 and the importance of gauge derivatives in non-linear diffusion schemes is reviewed in Section 3.1. Then we briefly discuss SURF based descriptors in Section 4. The overall framework of our family of descriptors is explained in Section 5. Finally, we show extensive experimental results in image matching, large-scale 3D SfM and visual categorization applications in Section 6.

2. Related Work

The highly influential SIFT [14] features have been widely used in applications from mobile robotics to object recognition, but are relatively expensive to compute and are not suitable for some applications with real-time demands. Inspired by SIFT, Bay et al. [5] proposed SURF features, which define both a detector and a descriptor. SURF features exhibit better results than previous schemes with respect to repeatability, distinctiviness and robustness, but at the same time can be computed much faster thanks to the use of integral images [15]. Recently, Agrawal et al. [16] proposed some modifications of SURF in both the detection and description steps. They introduced Center Surround Extremas (CenSurE) features and showed that they outperform previous detectors and have better computational characteristics for real-time applications. Their variant of the SURF descriptor, Modified-SURF (M-SURF), efficiently handles the descriptor boundaries problem and uses a more intelligent two-stage Gaussian weighting scheme in contrast to the original implementation which uses a single Gaussian weighting step.

All the mentioned approaches rely on the use of the Gaussian scale-space [8] framework to extract features at different scales. An original image is blurred by convolution with Gaussian kernels of successively large standard deviation to identify features at increasingly large scales. The main drawback of the Gaussian kernel and its set of partial derivatives is that both interesting details and noise are blurred away to the same degree. It seems to be more appropriate in feature description to make blurring locally adaptive to the image data so that noise will be blurred, while at the same time details or edges will remain unaffected. In this way, we can increase distinctiveness when describing an image region at different scale levels. In spirit, non-linear diffusion shares some similarities with the *geometric blur* proposed by Berg and Malik [17], in where the the amount of Gaussian blurring is proportional to the distance from the point of interest.

From their definition, gauge derivatives are local invariants. Matching by local invariants has previously been studied in the literature. In [18], Schmid and Mohr used the family of local invariants known as *local jet* [19] for image matching applications. Their descriptor vector contained 8 invariants up to third order for every point of interest in the image. This work represented a step forward over previous invariant recognition schemes [20]. In [9], Mikolajczyk and Schmid compared the performance of the *local jet* (with invariants up to third order) against other descriptors such as steerable filters [21], image moments [22] or SIFT. In their experiments the local jet exhibits poor performance compared to SIFT. We hypothesize that this poor performance is due to the fixed settings used in the experiments, such as a fixed image patch size and a fixed Gaussian derivative scale. In addition, invariants of high order are more sensitive to geometric and photometric distortions than first-order methods. In [23], the local jet was again used for matching applications, and they showed that even a descriptor vector of dimension 6 can outperfom SIFT for small perspective changes. By a suitable scaling and normalization, the authors obtained invariance to spatial zooming and intensity scaling. Although these results were encouraging, a more detailed comparison with other descriptors would have been desirable. However, this work motivated us to incorporate gauge invariants into the SURF descriptor framework.

Brown et al. [10], proposed a framework for learning discriminative local dense image descriptors from training data. The training data was obtained from large-scale real 3D SfM scenarios, and accurate ground truth correspondences were generated by means of multi-view stereo matching techniques [24, 25] that allow to obtain very accurate correspondences between 3D points. They describe a set of building blocks for building discriminative local descriptors that can be combined together and jointly optimized to minimize the error of a nearest-neighbor classifier. In this paper, we use the evaluation framework of Brown et al. to evaluate the performance of multiscale gauge derivatives under real large-scale 3D SfM scenarios.

3. Gauge Coordinates and Multiscale Gauge Derivatives

Gauge coordinates are a very useful tool in computer vision and image processing. Using gauge coordinates, every pixel in the image is described in such a way that if we have the same 2D local structure, the description of the structure is always the same, even if the image is rotated. This is possible since every pixel in the image is fixed separately in its own local coordinate frame defined by the local structure itself and consisting of the gradient vector $\vec{w}$ and its perpendicular direction $\vec{v}$:

$$\begin{aligned} \vec{w} &= \left(\frac{\partial L}{\partial x}, \frac{\partial L}{\partial y}\right) = \frac{1}{\sqrt{L_x^2+L_y^2}} \cdot (L_x, L_y) \\ \vec{v} &= \left(\frac{\partial L}{\partial y}, -\frac{\partial L}{\partial x}\right) = \frac{1}{\sqrt{L_x^2+L_y^2}} \cdot (L_y, -L_x) \end{aligned} \tag{1}$$

In Equation 1, L denotes the convolution of the image I with a 2D Gaussian kernel $g(x, y, \sigma)$, where σ is the kernel's standard deviation or scale parameter:

$$L(x, y, \sigma) = I(x, y) * g(x, y, \sigma) \tag{2}$$

Derivatives can be taken up to any order and at multiple scales for detecting features of different sizes. Raw image derivatives can only be computed in terms of the Cartesian coordinate frame x and y, so in order to obtain gauge derivatives we need to use directional derivatives with respect to a fixed gradient direction (L_x, L_y). The $\vec{v}$ direction is tangent to the isophotes or lines of constant intensity, whereas $\vec{w}$ points in the direction of the gradient, thus $L_v = 0$ and $L_w = \sqrt{L_x^2 + L_y^2}$. If we take derivatives with respect to first-order gauge coordinates, since these are fixed to the object, irrespective of rotation or translation, we obtain the following interesting results:

1. Every derivative expressed in gauge coordinates is an orthogonal invariant. The first-order derivative $\frac{\partial L}{\partial \vec{w}}$ is the derivative in the gradient direction, and in fact the gradient is an invariant itself.
2. Since $\frac{\partial L}{\partial \vec{v}} = 0$, this implies that there is no change in the luminance if we move tangentially to the constant intensity lines.

By using gauge coordinates, we can obtain a set of invariant derivatives up to any order and scale that can be used efficiently for image description and matching. Of special interest, are the second-order gauge derivatives L_{ww} and L_{vv}:

$$L_{ww} = \frac{L_x^2 L_{xx} + 2 \cdot L_x L_{xy} L_y + L_y^2 L_{yy}}{L_x^2 + L_y^2} \tag{3}$$

$$L_{vv} = \frac{L_y^2 L_{xx} - 2 \cdot L_x L_{xy} L_y + L_x^2 L_{yy}}{L_x^2 + L_y^2} \tag{4}$$

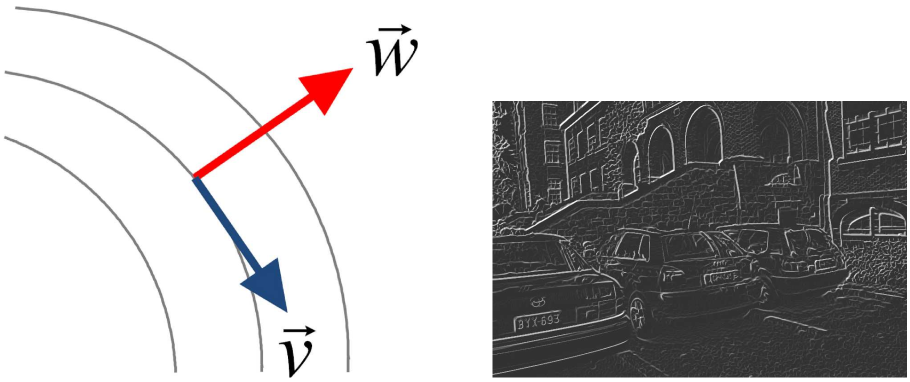
These two gauge derivatives can be obtained as the product of gradients in $\vec{w}$ and $\vec{v}$ directions and the 2×2 second-order derivatives or Hessian matrix.

$$L_{ww} = \frac{1}{L_x^2 + L_y^2} \begin{pmatrix} L_x & L_y \end{pmatrix} \begin{pmatrix} L_{xx} & L_{xy} \\ L_{yx} & L_{yy} \end{pmatrix} \begin{pmatrix} L_x \\ L_y \end{pmatrix} \tag{5}$$

$$L_{vv} = \frac{1}{L_x^2 + L_y^2} \begin{pmatrix} L_y & -L_x \end{pmatrix} \begin{pmatrix} L_{xx} & L_{xy} \\ L_{yx} & L_{yy} \end{pmatrix} \begin{pmatrix} L_y \\ -L_x \end{pmatrix} \tag{6}$$

L_{vv} is often used as a ridge detector. Ridges are elongated regions of approximately constant width and intensity, and at these points the curvature of the isophotes is high. L_{ww} gives information about gradient changes in the gradient direction.

Figure 1(a) illustrates first-order gauge coordinates. Unit vector $\vec{v}$ is always tangential to lines of constant image intensity (isophotes), while unit vector $\vec{w}$ is perpendicular and points in the gradient direction. Figure 1(b) depicts an example of the resulting second-order gauge derivative L_{ww} on one of the images from the Mikolajczyk and Schmid's standard dataset [9].



(a) (b)

Figure 1: (a) Local first-order gauge coordinates. (b) Resulting gauge derivative L_{ww} applied on the first image of the Leuven dataset, at a fixed scale $\sigma = 2$ pixels.

According to [26], where Schmid and Mohr explicitly describe the set of second-order invariants used in the local jet, we can find two main differences between the second-order gauge derivatives L_{ww}, L_{vv} and the local jet. The first difference is that by definition gauge derivatives are normalized with respect to the modulus of the gradient at each pixel. However, this normalization can be also included in the local jet formulation as shown in [23]. The second difference, and the most important one, is that the invariant L_{vv} is not included in the set of second-order derivatives of the local jet. The invariant L_{vv} plays a fundamental role in non-linear diffusion processes [7, 27]. Typically, Equation 4 is used to evolve the image in a way that locally adapts the amount of blurring to differential invariant structure in the image in order to perform edge-preserving smoothing [4].

3.1. Importance of Gauge Derivatives in Non-Linear Diffusion Schemes

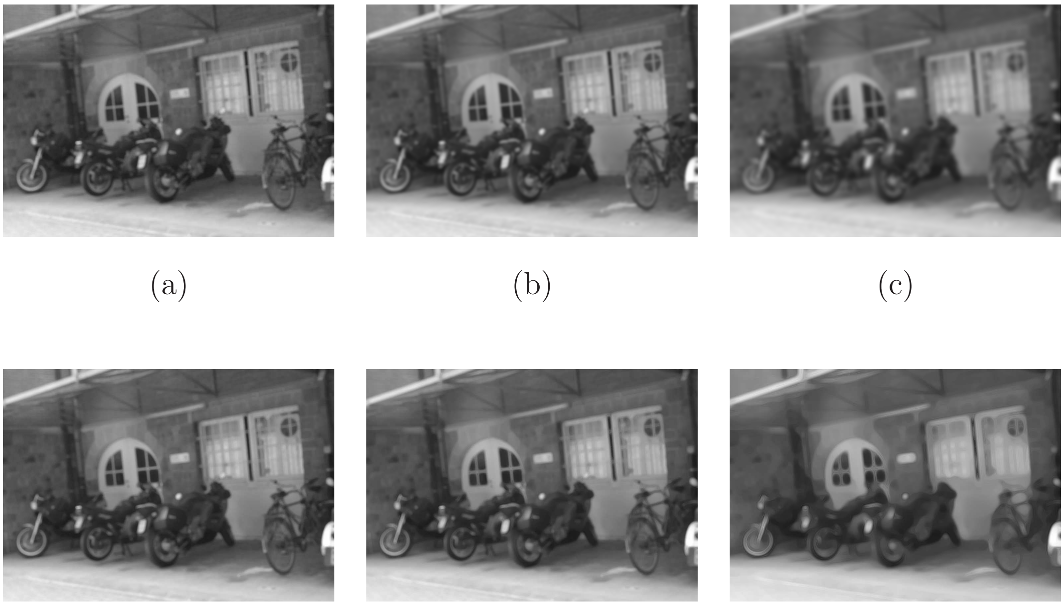
In this section we aim to throw some more light on our decision to use gauge derivatives in a feature descriptor by briefly reviewing non-linear image diffusion, and highlighting the important role of gauge derivatives in these schemes. Koendenrik [28] and Lindeberg [8] showed that the Gaussian kernel and its set of partial derivatives provide the unique set of operators for the construction of linear scale-space under certain conditions. Some examples of algorithms that rely on the Gaussian scale-space framework are SIFT [14] and SURF [5] invariant features.

However, to repeat, details are blurred in Gaussian scale-space during evolution. The advantage of blurring is the removal of noise, but relevant image structures like edges are blurred and drift away from their original locations during evolution. In general, a good solution should be to make the blurring locally adaptive to the image yielding the blurring of noise, while retaining details or edges.

In the early nineties, several Partial Differential Equations (PDEs) were proposed for dealing with the mentioned Gaussian scale-space problem. Some famous examples are the Perona-Malik equation [6] and the Mean Curvature Motion (MCM) [7]. Note that in general, non-linear diffusion approaches perform better than linear diffusion schemes [4, 29]. Recently, Kuijper showed in [29] that the evolution of an image can be expressed as a linear combination of the two different second-order gauge derivatives L_{ww} and L_{vv}. According to this, we can conclude that non-linear approaches steer between blurring L_{ww} and edge regularising L_{vv}. Some examples of practical applications of L_{ww} flow are image impaiting [30]. For L_{vv} flow an example is the cited MCM [7].

Based on this, we can think about a local invariant descriptor that takes into account the information encoded in the two gauge derivatives L_{vv} and L_{ww} while the image evolves according to a scale σ. Notice that in our family of descriptors we just replace the first-order local derivatives L_x and L_y with the gauge derivatives L_{vv} and L_{ww} and do not perform any image evolution through a non-linear scale space. That is, our descriptors will measure information about blurring (L_{ww}) and edge enhancing (L_{vv}) for different scale levels.

Another difference between first-order local derivatives and gauge ones is that gauge derivatives are intrisically weighted with the strength of the gradient L_w. That is, the weighting is intrinsically related to the image structure itself, and no artificial weighting such as Gaussian weighting is needed. This is an important advantage over other descriptors, such as for example SURF, where different Gaussian weighting schemes [16] have been proposed to improve the performance of the original descriptor.



(a) (b) (c)

(d) (e) (f)

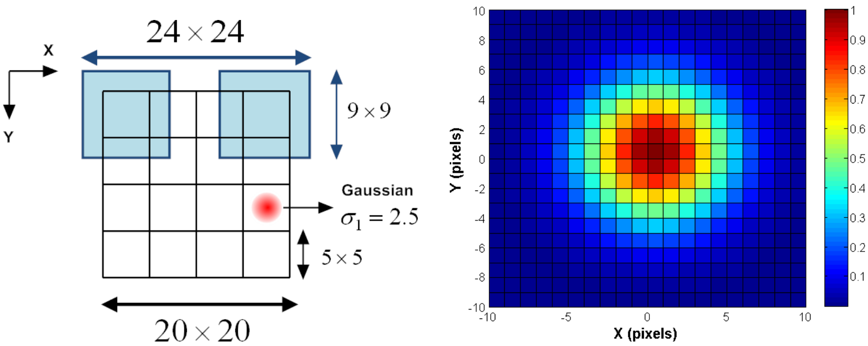
Figure 2: Gaussian scale-space versus Non-Linear diffusion schemes. The first row depicts the evolution of the sixth image from the Mikolajczyk and Schmid's Bikes dataset considering a Gaussian scale space of increasing σ in pixels. (a) $\sigma = 2$ (b) $\sigma = 4$ (c) $\sigma = 8$. The second row depicts the evolution of the same reference image but considering the MCM non-linear diffusion flow. (d) $\sigma = 2$ (e) $\sigma = 4$ (f) $\sigma = 8$. Notice how with non-linear diffusion schemes, details are enhanced and noise is removed, whereas for the Gaussian scale-space, details and noise are blurred in the same degree.

4. SURF Based Descriptors

Agrawal et al. proposed in [16] the Modified Upright-SURF descriptor (MU-SURF) which is a variant of the original U-SURF descriptor. MU-SURF handles descriptor boundary effects and uses a more robust and intelligent two-stage Gaussian weighting scheme. For a detected feature at scale s, Haar wavelet responses L_x and L_y of size $2s$ are computed over a $24s \times 24s$ region. This region is divided into $9s \times 9s$ subregions with an overlap of $2s$. The Haar wavelet responses in each subregion are weighted with a Gaussian ($\sigma_1 = 2.5s$) centered on the subregion center and summed into a descriptor vector $d_v = (\sum L_x, \sum L_y, \sum |L_x|, \sum |L_y|)$. Then, each subregion vector is weighted using a Gaussian ($\sigma_2 = 1.5s$) defined over a mask of 4×4 and centered on the interest keypoint. Finally, the descriptor vector of length 64 is normalized into a unit vector to achieve invariance to contrast. Figure 3(a) depicts the involved regions and subregions in the MU-SURF descriptor building process.

The main difference between the MU-SURF and U-SURF descriptor is that the size of the region is reduced to $20s \times 20s$ divided into $5s \times 5s$ subregions without any overlap between subregions. In addition, Haar wavelet responses in each subregion are weighted by a Gaussian ($\sigma = 3.3s$) centered at the interest keypoint. This is a very small standard deviation considering that the square grid size is $20s \times 20s$. Figure 3(b) depicts a normalized 2D Gaussian kernel considering a standard deviation $\sigma = 3.3$. Notice how this weighting scheme smoothes completely the contribution of points far from the point of interest. Therefore, only points within a distance of ± 5 pixels have a significant influence in the whole descriptor.

The upright version of SURF-based descriptors (U-SURF) is faster to compute and usually exhibits higher performance (compared to their corresponding rotation invariant version, SURF) in applications where invariance to rotation is not necessary. Some examples of these applications are 3D reconstruction [5] or face recognition [31]. Although the MU-SURF descriptor is not invariant to rotation, it can be easily adapted for this purpose by interpolating Haar wavelet responses according to a dominant orientation in the same way as is done in the orginal SURF descriptor.



(a) (b)

Figure 3: (a) MU-SURF descriptor building process. All sizes are relative to the scale of the feature. (b) The single Gaussian weighting scheme proposed in the original SURF descriptor. Normalized 2D gaussian kernel values considering a Gaussian kernel of standard deviation $\sigma = 3.3$ centered at the interest keypoint. Best viewed in color.

5. Gauge-SURF Descriptors

Our family of G-SURF descriptors is based on the original SURF descriptor. However, instead of using the local first-order derivatives L_x and L_y, we replace these two derivatives by the second-order gauge derivatives L_{ww} and

L_{vv}. For computing multiscale gauge derivatives, we always need to compute the derivatives first in the Cartesian coordinate frame (x, y), and then fix the gradient direction (L_x, L_y) for every pixel. After these computations, we can obtain invariant gauge derivatives up to any order and scale with respect to the new gauge coordinate frame $(\vec{w}, \vec{v})$.

From the definition of gauge coordinates in Equation 1, it can be observed that these coordinates are not defined at pixel locations where $\sqrt{L_x^2 + L_y^2} = 0$, i.e. at saddle points and extrema of the image. In practice this is not a problem as ter Haar Romeny states in [4], since we have a small number of such points, and according to Morse theory [32] we can get rid of such singularities by infinitesimally small local changes in the intensity landscape. What we do in practice is to not sum the contributions of these points into the final descriptor vector.

Now, we will describe the building process of a GU-SURF descriptor of dimension 64. For a detected feature at scale s, we compute first and second-order Haar wavelet responses $L_x, L_y, L_{xx}, L_{xy}, L_{yy}$ over a $20s \times 20s$ region. We call L_x the Haar wavelet response in the horizontal direction and L_y the response in the vertical direction. The descriptor window is divided into 4×4 regular subregions without any overlap. Within each of these subregions Haar wavelets of size $2s$ are computed for 25 regularly distributed sample points. Once we have fixed the gauge coordinate frame for each of the pixels, we compute the gauge invariants $|L_{ww}|$ and $|L_{vv}|$. Each subregion yields a four-dimensional descriptor vector $d_v = (\sum L_{ww}, \sum L_{vv}, \sum |L_{ww}|, \sum |L_{vv}|)$. Finally, the total length of the unitary descriptor vector is 64.

Figure 4 depicts an example of the GU-SURF descriptor building process.

For simplicity reasons, we only show one gauge coordinate frame for each of the 4×4 subregions. Note that if we want to compute a descriptor which is invariant to rotation, we do not need to interpolate the value of the invariants L_{ww} and L_{vv} according to a dominant orientation as in SURF or M-SURF. Due to the rotation invariance of gauge derivatives, we only have to rotate the square grid.

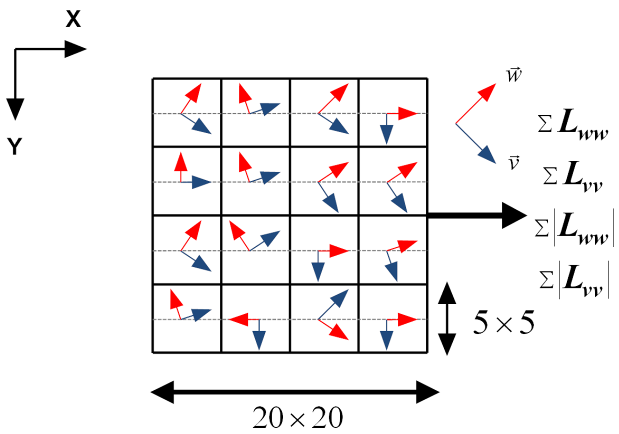


Figure 4: GU-SURF descriptor building process. Note that for the rotationally-invariant version of the descriptor we just have to rotate the square grid.

In the same way as proposed in SURF, we use box-filters to approximate first and second-order Gaussian derivatives. These box-filters are constructed through the use of integral images [15], which allows the approximation of Gaussian derivatives with low computational demands.

In Section 5.1, we describe the rest of descriptors of the G-SURF family included in the *OpenGSURF* library and the notation of the descriptors we will use throughout the rest of the paper.

5.1. Descriptors Notation

Similar to [5], we can modify the number of divisions of the square grid and the size of each subregion in Figure 4 to obtain descriptors of different dimensions. The descriptor size has a major impact on the matching speed and recall rates. We also tested the extended version of the descriptors [5]. Due to space limitations, we will not evaluate this version of the descriptors in this paper. However, this option is included in the OpenGSURF library. As shown in [5], the overall effect of the extended descriptor is minimal.

Now, we will describe the notation for the set of descriptors we use throughout the rest of the paper, with the number of dimensions of the descriptors in parenthesis. For the SURF-based descriptors the default dimension is 64, whereas for SIFT the default dimension is 128.

- **SURF (64)**: Original SURF implementation as described in [33] that uses a single Gaussian weighting scheme of a standard deviation $\sigma = 3.3s$ centered at the interest keypoint and a square grid of $20s \times 20s$.

- **M-SURF (64)**: Modified-SURF descriptor as described in [16]. This descriptor uses a square grid of $24s \times 24s$ considering an overlap of Haar wavelets responses and two Gaussian weighting steps.

- **G-SURF (64)**: Gauge-SURF descriptor, that uses second-order multiscale gauge derivatives and a square grid of $20s \times 20s$ without any additional Gaussian weighting step.

- **MG-SURF (64)**: Modified Gauge-SURF descriptor, that uses the same scheme as the M-SURF but replacing first-order local derivatives (L_x, L_y) for second-order gauge ones (L_{ww}, L_{vv}).

- **NG-SURF (64)**: No Gaussian Weighting-SURF descriptor. This descriptor is exactly the same as the original SURF descriptor, with the difference that no Gaussian weighting step is applied. In this way, we can perform a fair comparison between gauge derivatives and first-order local derivatives based descriptors without any additional weighting scheme.

- **SIFT (128)**: The SIFT descriptor as described in [14]. This descriptor has a dimension of 128.

For all the descriptors mentioned above, we denote the *upright* version of the descriptors (not invariant to rotation) by adding the prefix U to the name of the descriptor. For example, GU-SURF is the upright version of the G-SURF descriptor. By modifying the number of divisions of the square grid and the size of each of the subregions, we can obtain descriptors of different dimensions. Now, we will describe the number of divisions of the square grid and the size of each subregion for each of the descriptor sizes we evaluate in this paper. The first number in parenthesis indicates the dimension of the descriptor with the new square grid and subregion size.

- **(36)**: Square grid of size $18s \times 18s$ yielding 3×3 subregions each of size $6s \times 6s$.

- **(144)**: Square grid of size $24s \times 24s$ yielding 6×6 subregions each of size $4s \times 4s$.

6. Results and Discussion

In this section, we present extensive experimental image matching results obtained on the standard evaluation set of Mikolajczyk and Schmid [9], large-scale 3D SfM applications [10] and visual categorization experiments [2]. In addition, we introduce a new dataset named *Iguazu* that consist of a series of six images with the addition of increasing random Gaussian noise levels with respect to the first image of the dataset. In some research areas such medical imaging, RADAR or astronomy, images are usually corrupted by different types of random noise. Therefore, we think that the evaluation of local descriptors in these kind of datasets is of interest.

Our family of G-SURF descriptors implementation is based on the OpenSURF library[1]. The source code of our library is attached as supplementary paper material. OpenSURF is an open source C++ based library with detailed documentation and a reference paper [12]. To our knowledge, this library is widely used in the computer vision and robotics community and exhibits good perfomance, while having speed similar to the original SURF library which is only available as a binary. Currently, OpenSURF uses by default the M-SURF descriptor, since perfomance is much higher than when using the single weighting Gaussian scheme. We think that OpenSURF is a good open source library for perfoming an evaluation and comparison of a set of descriptors that are all based on the same source code framework.

We also show comparison results with respect to SIFT descriptor, using Vedaldi's implementation [13]. In all SIFT experiments we used the default

[1] Available from http://code.google.com/p/opensurf1/

magnification factor $m = 3.0$, i.e. each spatial bin of the histogram has support of size $m \cdot \sigma$ where σ is the scale of the point of interest. This parameter has an important effect in descriptor performance. See [34] for more details.

We have compared G-SURF descriptors to SURF, M-SURF, NG-SURF (all based on OpenSURF implementation) and SIFT (based on Vedaldi's implementation), in both standard and upright forms. Agrawal et al. [16] claim that M-SURF's performance is similar to the original SURF library, although their implementation is much faster than the original one. Like Agrawal et al., we also noticed that the standard single Gaussian weighting scheme as proposed in the original SURF algorithm [5] gives poor results. However, we also include in our comparison the standard SURF method based on the OpenSURF implementations, since this single Gaussian scheme is still used in practically all of the open source libraries that include the SURF algorithm, such as OpenCV or dlib C++ [2]. In addition, in Section 6.2 we also show some comparison results with respect to the OpenCV SURF implementation, since this library has become a de facto standard for fast-to-compute descriptors.

The rest of the experimental results and discussion section is organized as follows: In Section 6.1 we show extensive image matching experiments based on the standard evaluation framework of Mikolajczyk and Schmid [9], with the addition of a new dataset for evaluating descriptor performance under different image noise settings. Then, in Section 6.3 we evaluate the perfor-

[2] Available from http://dclib.sourceforge.net/

mance of G-SURF descriptors in large-scale 3D SfM scenarios. In Section 6.4 we show some results on visual categorization applications, and finally in Section 6.5 we describe some implementation details and timing evaluation results.

6.1. Image Matching Experiments

We tested our descriptors using the image sequences and testing software provided by Mikolajczyk [3]. We used OpenSURF's Fast Hessian to extract the keypoints in every image and then compute the descriptors, setting the number of octaves and number of intervals to 4 and 2 respectively.

The standard dataset includes several image sets (each sequence generally contains 6 images) with different geometric and photometric transformations such as image blur, lighting, viewpoint, scale changes, zoom, rotation and JPEG compression. In addition, the ground truth homographies are also available for every image transformation with respect to the first image of every sequence. We show results on eight sequences of the dataset. Table 1 gives information about the datasets and the image pairs we evaluated for each of the selected sequences. We also provide the number of keypoints detected for each image and the Hessian threshold value to permit reproduction of our results.

Descriptors are evaluated by means of *recall versus 1 - precision* graphs as proposed in [9]. This criterion is based on the number of correct matches

[3] Available from http://www.robots.ox.ac.uk/˜vgg/research/affine/

and the number of false matches obtained for an image pair:

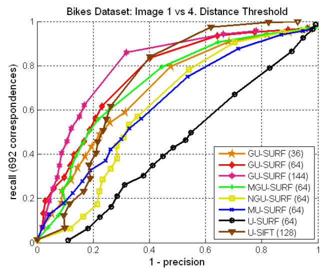
$$recall = \frac{\#correct\ matches}{\#correspondences}$$
$$(7)$$
$$1 - precision = \frac{\#false\ matches}{\#all\ matches}$$

The number of correct matches and correspondences is determined by the overlap error. Two regions (A, B) are deemed to correspond if the overlap error ϵ_0, defined as the error in the image area covered by the regions, is sufficiently small, as shown in Equation 8:

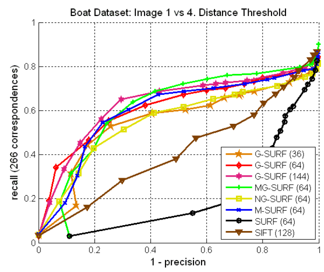
$$\epsilon_0 < 1 - \frac{A \cap H^T \cdot B \cdot H}{A \cup H^T \cdot B \cdot H} \tag{8}$$

In [9] there were shown some examples of the error in relative point location and recall considering different overlap errors. They found that for overlap errors smaller than 20% one can obtain the maximum number of correct matches. In addition, they showed that recall decreases with increasing overlap errors. Larger overlap errors result in a large number of correspondences and general low recall. Based on this, we decided to use an overlap error threshold of $\epsilon_0 < 20\%$, since we think this overlap error is reasonable for SfM applications, where you are only interested in very accurate matches. Furthermore, as in [35] we also impose that the error in relative point location for two corresponding regions has to be less than 2.5 pixels: $\|x_a - H \cdot x_b\| < 2.5$, where H is the homography between the images. Due to space limitations, we only show results on similarity threshold based matching, since this technique is better suited for representing the distribution of the descriptor in its feature space [9].

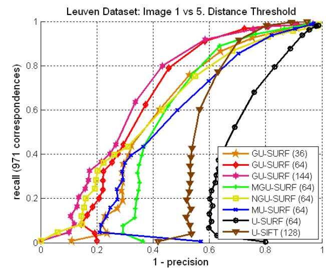
Figure 5 depicts *recall versus 1-precision* graphs for the selected pairs of images. This figure suggests the following conclusions:



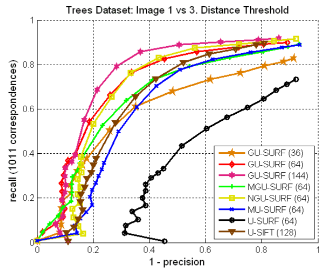
(a)



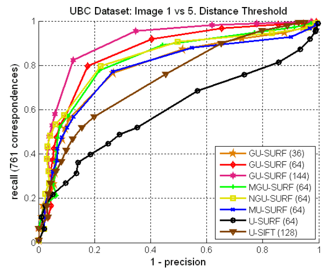
(b)



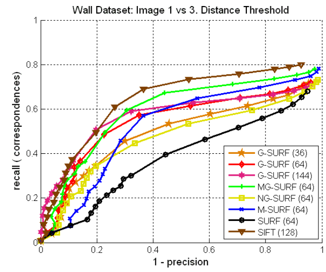
(c)



(d)



(e)



(f)

Figure 5: Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (a) Bikes 1 vs 4. (b) Boat 1 vs 4. (c) Leuven 1 vs 5. (d) Trees 1 vs 3. (e) UBC 1 vs 5. (f) Wall 1 vs 3. Best viewed in color.

Dataset	Image Change	Image N	# Keypoints Image 1	# Keypoints Image N	Hessian Threshold
Bikes	Blur	4	2275	1538	0.0001
Bikes	Blur	5	2275	1210	0.0001
Boat	Zoom+Rotation	4	2676	1659	0.0001
Graffiti	Viewpoint	2	1229	1349	0.001
Leuven	Illumination	5	2705	2009	0.00001
Trees	Blur	3	3975	4072	0.0001
UBC	JPEG Compression	5	2106	2171	0.0001
Van Gogh	Rotation	10	864	782	0.00005
Van Gogh	Rotation	18	864	855	0.00005
Wall	Viewpoint	3	3974	3344	0.0001
Iguazu	Gaussian Noise	3	1603	2820	0.0001
Iguazu	Gaussian Noise	4	1603	3281	0.0001
Iguazu	Gaussian Noise	5	1603	3581	0.0001

Table 1: Sequences and image pairs used for image matching experiments: Image change, image number, keypoints number and Hessian threshold value.

- In general, among the upright evaluation of the descriptors, GU-SURF descriptors perform much better than their competitors, especially for high precision values, with sometimes more than 20% improvement in recall for the same level of precision with respect to MU-SURF (64) and U-SIFT (128) (e.g. Leuven, Bikes and Trees datasets), and even much more improvement with respect to U-SURF (64). GU-SURF (144) was the descriptor that normally achieved the highest recall for all the experiments, followed close by GU-SURF (64). GU-SURF (36) also exhibits good performance, on occasions even better than higher dimensional descriptors such as U-SIFT (128) or MU-SURF (64).

- In the upright evaluation of the descriptors, one can obtain higher recall rates by means of descriptors that do not have any kind of Gaussian weighting or subregions overlap. For example, we can observe this effect between NGU-SURF (64) and U-SURF (64), where the only difference between both descriptors is the Gaussian weighting step. Furthermore, we can see that between GU-SURF (64) and MGU-SURF (64), GU-SURF (64) obtained higher recall values than when using the modified version of the descriptors.

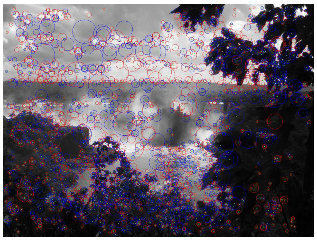
- With respect to the rotation invariant version of the descriptors, in these cases the modified descriptor version plays a more important role. The use of two Gaussian weighting steps and subregions overlap yields a more robust descriptor with respect to large geometric deformations and non-planar rotations. In addition, the Gaussian weighting helps in reducing possible computation errors when interpolating Haar wavelets responses according to a dominant orientation. This interpolation of the responses is not necessary in the case of gauge derivatives, since by definition they are rotation invariant. We can observe that MG-SURF (64) obtained slightly better results compared to M-SURF (64) and SIFT (128) for the Boat dataset (Zoom+Rotation). For the Wall dataset (changes in viewpoint), SIFT (128) was the descriptor that obtained the best results, and MG-SURF (64) obtained better results compared to M-SURF (64), especially for high precision values.

- When comparing gauge-based descriptors and first-order local derivatives descriptors, we can observe that gauge-based descriptors always obtained higher recall values, both in the standard and upright form of the descriptors. We can observe this behaviour between G-SURF (64) versus NG-SURF (64), and MG-SURF (64) versus M-SURF (64) and also depending on the upright version of the descriptors. One of the reasons why gauge derivatives obtained better performance is because they are intrinsically weighted by the strength of the gradient L_w per pixel, and thus the resulting decriptor exhibits a higher discriminative power.

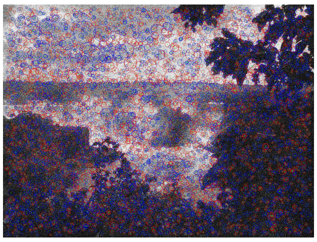
- In all the sequences the worst results were obtained by OpenSURF's SURF implementation, which uses the single Gaussian weighting scheme that gives poor results.

6.1.1. Evaluation under image noise transformations

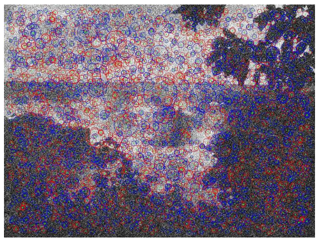
In this section, we evaluate the performance of the descriptors under image noise transformations. For this purpose, we created a new dataset named *Iguazu*. This dataset consists of 6 images, and the image transformation in this case is the progressive addition of random Gaussian noise. For each pixel of the transformed images, we add random Gaussian noise with increasing variance considering grey scale value images. The noise variances for each of the images are the following: Image 2 ± 2.55, Image 3 ± 12.75, Image 4 ± 15.00, Image 5 ± 51.0 and Image 6 ± 102.00, considering that the grey value of each pixel in the image ranges from 0 to 255. This new dataset is available as supplementary paper material. Noisy images are very common in fields such as biomedical imaging [4] and other research areas such as Synthetic Aperture RADAR imaging (SAR) [36]. We think that for these applications, a descriptor which is robust to different noise settings is very desirable. Figure 6 depicts three images of the Iguazu dataset for image random noise transformations, and the *recall versus 1-precision* for three image pairs of the sequence.



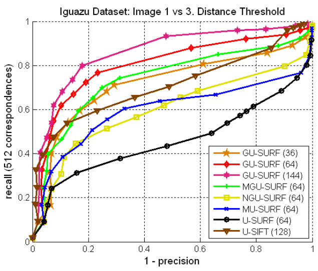
(a)



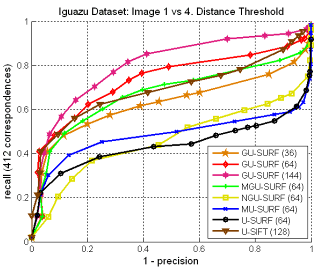
(b)



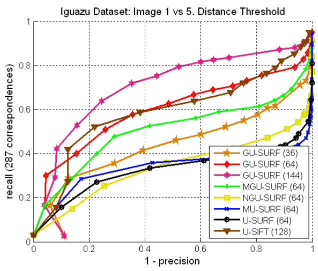
(c)



(d)



(e)



(f)

Figure 6: In the first row (a,b,c), we show some images from the Iguazu dataset, with incremetally increasing random Gaussian noise values per image. Notice that when severe random noise is added to the image, the number of detected blobs increases, mainly at small scales. The detected keypoints are shown in red or blue depending on the sign of the Laplacian. (a) Iguazu 1 (b) Iguazu 3 (c) Iguazu 5. In the second row (d,e,f), Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (d) Iguazu 1 vs 3 (e) Iguazu 1 vs 4 (f) Iguazu 1 vs 5. Best viewed in color.

According to the graphs, we can observe that for this dataset, the difference between gauge-derivatives and first-order local derivatives based descriptors is much more significant than in the previous image transformations evaluation. The best results were obtained again with the GU-SURF (144) descriptor. In this experiment, U-SIFT (128) obtained also good results, with higher recall values than MU-SURF (64), U-SURF (64) and NGU-SURF (64). Notice that in these experiments, GU-SURF (36) obtained better results for the three image pairs than MU-SURF (64), U-SURF (64) and NGU-SURF (64). This is remarkable, due to the low dimension of the descriptor, and this clearly demonstrates the disciminative properties of gauge derivatives against first-order ones. The main reason why G-SURF descriptors exhibit good performance against image noise settings and higher recall rates compared to first-order local derivatives methods is because G-SURF descriptors measure information about the amount of blurring (L_{ww}) and details or edge enhancing (L_{vv}) in the image at different scale levels.

6.1.2. Evaluation under pure rotation sequences

One of the nicest properties of gauge derivatives is their invariance against rotation. In this section, we compare G-SURF descriptors against first-order local derivatives descriptors, to highlight the rotation invariance properties of gauge derivatives. For this purpose, we decided to use the Van Gogh sequence that consists of pure rotation image transformations. This sequence and the ground truth homographies relating the images can be downloaded from Mykolajczyk's older webpage[4]. In order to show the performance of

[4]http://lear.inrialpes.fr/people/mikolajczyk/Database/rotation.html

G-SURF descriptor under pure rotation transformation, we evaluated two image pairs from the Van Gogh sequence. Figure 7 depicts the reference image and the other two images that are related by a pure rotation of 90° and 180° with respect to the reference image.

(a) Image 2 (b) Image 10 (c) Image 18

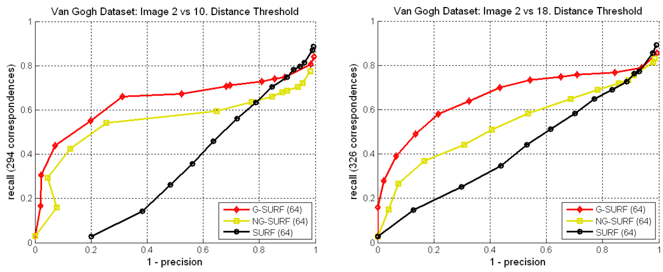
Figure 7: Van Gogh rotation dataset. Images 2 and 10 are related by a pure rotation of 90°, whereas Images 2 and 18 are related by a pure rotation of 180°.

Figure 8 depicts the *recall versus 1-precision* for the selected image pairs from the Van Gogh dataset. In this experiment, we compared only G-SURF (64) versus NG-SURF (64) and SURF (64). According to the results, we can observe that for some points in the graphs, by using G-SURF (64) there is an improvement in recall of about the 20% with respect to NG-SURF (64) and approximately double 40%, with respect to SURF (64) for the same precision values. These results highlight the effect of the nice rotation invariance property of gauge-derivatives in the matching capabilities of the descriptors.

6.2. Comparison to OpenCV

In this section, we also compare our G-SURF descriptors with the latest OpenCV[5] implementation of the SURF descriptor. According to [37],

[5] Available from http://sourceforge.net/projects/opencvlibrary/

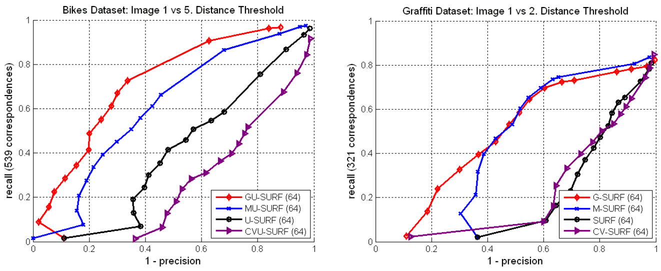


(a) (b)

Figure 8: Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (a) Van Gogh 2 vs 10 (b) Van Gogh 2 vs 18. Best viewed in color.

OpenCV's SURF implementation has become a de facto standard for fast-to-compute descriptors. However as we will show in our results, the descriptor performance is poor and much lower compared to OpenSURF's default M-SURF descriptor. This low performance is because the SURF implementation in OpenCV uses also the single Gaussian weighting scheme as proposed in the original SURF paper [5].

Figure 9 depicts *recall versus 1-precision* graphs for two image pairs from the Bikes and Graffiti datasets. In this experiment, we compare G-SURF (64) with respect to M-SURF (64), SURF (64) and CV-SURF (64) both in the upright and standard forms of the descriptors. We denote by CV-SURF the OpenCV implementation of the SURF descriptor using the single weighting scheme as described in Section 4. According to the results, we can see that the OpenCV implementation gives poor results, comparable to SURF (64) in OpenSURF's implementation, since both algorithms use the mentioned single Gaussian weighting scheme. We can appreciate a huge difference in recall with respect to G-SURF (64) and M-SURF (64).



(a) (b)

Figure 9: Image matching experiments: Recall versus 1-precision graphs, Similarity threshold based matching. (a) Bikes 1 vs 5 (b) Graffiti 1 vs 2. Best viewed in color.

6.3. Application to 3D Structure from Motion

In this section, we evaluate the performance of G-SURF based descriptors in large-scale 3D SfM applications. In particular, we use the learning local image descriptors dataset from [10]. In the mentioned work, Brown et al. proposed a framework for learning dense local image descriptors from training data using 3D correspondences from large-scale SfM datasets. For generating ground truth image correspondences between real interest points, the authors used multi-view stereo matching techniques [24, 25] that allow very accurate correspondences between 3D points to obtained.

The available dataset consists of several scale and orientation normalized 64×64 image patches centered around detected Harris corners or Difference of Gaussian (DoG) [14] features. Those patches were extracted from real 3D points of large-scale SfM scenarios. In our evaluation, we used 40,000 patch pairs centered on detected Harris corners from which 50% are match pairs and the other 50% are considered non-match pairs. We attach the set of matches/non-matches image patches used for the evaluation as supplementary material of the paper. In the evaluation framework of Brown et al., two patches are considered to be a match if the detected interest points are within 5 pixels in position, 0.25 octaves in scale and $\pi/8$ radians in angle. Figure 10 depicts some of the pre-defined match, non-match pairs from the Liberty dataset.

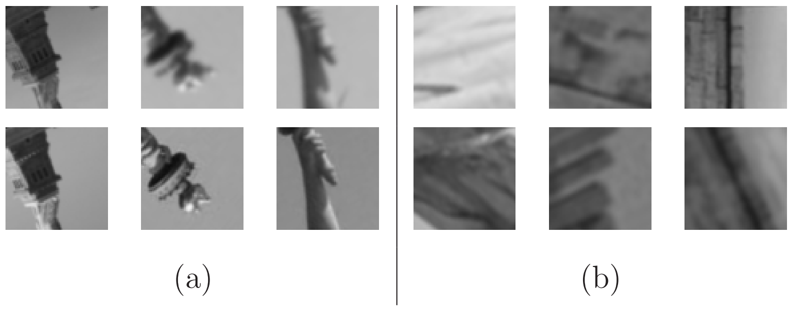


Figure 10: Some of the predefined match, non-match pairs from the Liberty dataset. Each row shows 3 pairs of image patches and the two image patches in each pair are shown in the same column. (a) Match pairs. (b) Non-match pairs.

We performed an evaluation of the upright version of the descriptors U-SURF (64), MU-SURF (64), GU-SURF (64), MGU-SURF (64), NGU-SURF (64) and U-SIFT (128) for both the Liberty and Notre Dame datasets. We chose a scale of 2.5 pixels to make sure that no Haar wavelet responses were computed outside the bounds of the image patch. For all the image pairs in the evaluation set, we computed the distance between descriptors and by means of sweeping a threshold on the descriptor distance we were able to generate ROC curves. Figure 11 depicts the ROC curves for the Liberty dataset, whereas Figure 12 depicts the ROC curves for the Notre Dame dataset.

In addition, in Table 2 we also show results in terms of the 95% error rate which is the percentage of incorrect matches obtained when the 95% of the

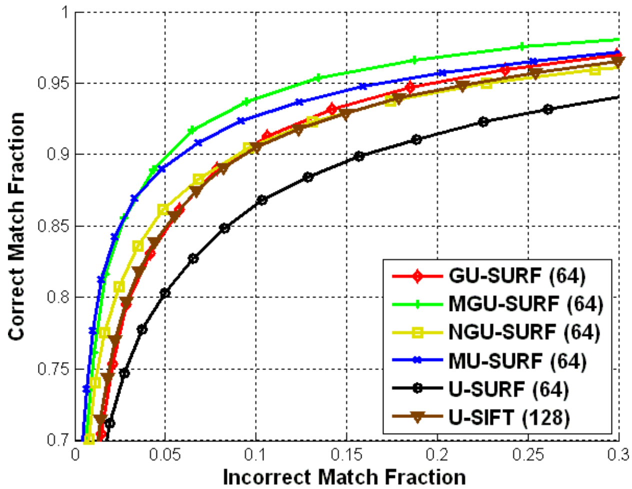


Figure 11: ROC curves for local image descriptors. Liberty dataset. Best viewed in color.

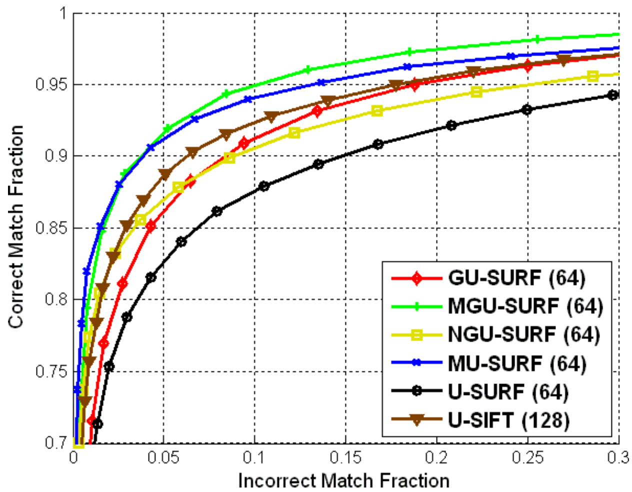


Figure 12: ROC curves for local image descriptors. Notre Dame dataset. Best viewed in color.

true matches are found.

Descriptor	**Liberty**	**Notre Dame**
GU-SURF (64)	19.78	18.95
MGU-SURF (64)	**12.55**	**10.19**
NGU-SURF (64)	22.95	25.22
MU-SURF (64)	16.88	13.17
U-SURF (64)	36.49	34.18
U-SIFT (128)	21.92	17.75

Table 2: Local image descriptors results. 95% error rates, with the number of descriptor dimension in parenthesis.

According to the results, we can observe that the lowest incorrect match fraction rate for the 95% recognition rates was obtained by the MGU-SURF (64) descriptor. This descriptor uses the same square grid configuration, two Gaussian weighting steps and subregions overlap as proposed in [16] for the MU-SURF descriptor. In typical large-scale 3D SfM scenarios, there exist non-planar transformations and illumination changes resulting from viewing a truly 3D scene [10]. In addition, second-order derivatives are more sensitive to perspective or affine changes than first-order ones. Therefore, on those scenarios where the affine changes or changes on perspective are significant, the two-steps Gaussian weighting and subregions overlap seem to have a good effect on the descriptor performance. This is the reason why in this evaluation we obtained better results for MGU-SURF (64) and MU-SURF (64) against GU-SURF (64) and NGU-SURF (64), which do not use any kind of subregion overlap or Gaussian weighting steps. U-SIFT (128) also obtained good results, always better than NGU-SURF (64) and very similar results compared to GU-SURF (64), slightly better for the Notre Dame dataset. U-SIFT (128) also uses biliner interpolation between the bins of the descriptor histogram [14]. When comparing gauge-derivatives based descriptors and first-order local derivatives ones, without any subregion overlap nor any Gaussian weighting step, we can observe that GU-SURF (64) obtained much better results than NGU-SURF (64). As expected, the worst results were obtained for the U-SURF (64) descriptor, since in this descriptor configuration the single Gaussian weighting step smoothes to a very high degree the descriptor information, yielding lower recognition rates.

Besides, in the OpenGSURF library, the user can choose between the SIFT-style clipping normalization or unit vector normalization of the descriptor. This normalization can a have a big impact on the matching performance of the descriptors, as demonstrated in [38, 10], where one can obtain lower error rates by using the SIFT-style clipping normalization. However, in order to avoid the influence of this normalization style in our results, we just show results using the standard unit vector normalization, except for the SIFT descriptor, in which we use its default SIFT-style clipping normalization.

6.4. Application to Visual Categorization Problems

In this experiment, we show that G-SURF based descriptors can be used efficiently in typical visual image categorization or object recognition problems. Bay et al. have shown in previous work [39, 33, 5] that SURF-based descriptors can be used efficiently in this kind of applications. Nowadays, SURF or SIFT invariant descriptors are of common use in typical visual categorization or object recognition schemes [2]. In a similar way to [40], we performed our tests using the Caltech faces, airplanes and camels dataset [6]. Firstly, we resized all the images to a 640×480 resolution and selected 25% of all the images (randomly distributed among the three categories) for training. The rest of the images were used for test evaluation.

Even though this is a simple visual categorization problem, we want to evaluate if G-SURF based descriptors can exhibit higher recognition rates than traditional first-order spatial derivatives based approaches due to the extra invariance offered by using gauge derivatives. Figure 13 depicts three image pairs of the different categories that we used in our evaluation. In particular, we can expect higher confusion between the faces and camels categories. This is because in some images of the camels dataset we can observe some human faces as shown for example in Figure 13(f), and also that camel and human faces share some degree of similarity.

In order to perform an evaluation of the different local descriptors, we used our own implementation of the visual bag of keypoints method described in [2]. This implementation has been successfully tested before in an occupant monitoring system based on visual categorization [41]. Basically, we used the standard Fast-Hessian detector to detect features of interest at different scale levels, and then we computed different local descriptors. In this experiment, we only show a comparison between 64 dimensional descriptors in their upright form (U-SURF, MU-SURF, GU-SURF, NGU-SURF). Once the descriptors are extracted, the visual vocabulary is constructed by means of the standard *k-means* clustering scheme [42]. This clustering al-

[6]http://www.vision.caltech.edu/html-files/archive.html

Figure 13: Three pairs of images from the Caltech dataset. (a,d) Faces. (b,e) Airplanes. (c,f) Camels. Notice the possible confusion between the faces and camels categories.

gorithm proceeds by iterated assignments of keypoints descriptors to their closest cluster centers and recomputation of the cluster centers. The selection of the number of clusters and the initialization of the centers are of great importance in the performance of the algorithm. Finally, the visual categorization is done by using a simple Näive Bayes classifier [43]. In order to reduce the influence of the clustering method on the final results, we decided to use a small number of clusters $k = 20$ and performed a random initialization of the cluster centers. To avoid cluster initialization problems, the clusters were randomly initialized ten times in each of the experiments, reporting categorization results just for the cluster initialization that obtained minimum compactness measure.

Tables 3, 4, 5, 6 show information about the performance of each of the different descriptors in the test evaluation. Similar to [2], we used three performance measures to evaluate the performance in visual categorization: the confussion matrix, the overall error rate and the mean ranks. For more information about the meaning of these performance measures, we recommend the reader to check the experiments section in [2].

True Classes	**Faces**	**Airplanes**	**Camels**
Faces	82.6531	0.8714	19.0000
Airplanes	1.3605	91.5033	12.0000
Camels	15.9864	7.6252	69.0000
Mean Ranks	1.1973	1.1154	1.3100
Overall Error Rate	0.1352		

Table 3: Confusion matrix, mean ranks and overall error rate for U-SURF (64).

True Classes	**Faces**	**Airplanes**	**Camels**
Faces	79.2517	0.3267	25.5000
Airplanes	0.6802	93.6819	7.0000
Camels	20.0680	5.9912	67.5000
Mean Ranks	1.2142	**1.0824**	1.3250
Overall Error Rate	0.1303		

Table 4: Confusion matrix, mean ranks and overall error rate for MU-SURF (64).

With respect to the confussion matrix, we can observe that GU-SURF (64) descriptor obtained higher recognition rates for the faces (85.3741%) and camels (72.0000%) categories. However, the MU-SURF (64) descriptor

True Classes	**Faces**	**Airplanes**	**Camels**
Faces	**85.3741**	0.2178	22.5000
Airplanes	0.3401	91.8301	5.5000
Camels	14.2857	7.9520	**72.0000**
Mean Ranks	**1.1564**	1.1132	**1.2800**
Overall Error Rate		**0.1232**	

Table 5: Confusion matrix, mean ranks and overall error rate for GU-SURF (64).

True Classes	**Faces**	**Airplanes**	**Camels**
Faces	80.6122	0.3267	20.0000
Airplanes	1.36054	93.3551	10.0000
Camels	18.0272	6.31808	70.0000
Mean Ranks	1.2074	1.0882	1.3
Overall Error Rate		0.1260	

Table 6: Confusion matrix, mean ranks and overall error rate for NGU-SURF (64).

obtained a higher recognition rate for the airplanes (93.68%) dataset. In the same way, GU-SURF (64) obtained the lowest mean ranks for the faces (1.1564) and camels (1.2800) datasets and MU-SURF (64) obtained the lowest one for the airplanes dataset (1.0824). Regarding the overall error rate, GU-SURF (64) was the descriptor that achieved the lowest error (0.1232). There is a reduction in the overall error rate of 8.88% with respect to U-SURF (64), 5.45% with respect to MU-SURF (64) and 2.22% with respect to NGU-SURF (64). Even though the experimental evaluation was a simple visual categorization problem, we can conclude that G-SURF based descriptors can be used efficiently in these visual recognition schemes. In addition, G-SURF descriptors can also obtain lower error rates and higher recognition rates than traditional approaches that are based only on first-order local derivatives.

6.5. Implementation Details and Timing Evaluation

In this section, we describe some implementation details of G-SURF descriptors and perform a timing evaluation. One of the criticisms about using second-order derivatives in the context of local descriptors, is the higher computational cost that sometimes is not accompanied by a better performance. In this section, we show that by means of using gauge derivatives we can obtain much better performance than first-order based methods with comparable computational cost. Table 7 shows timing results for descriptor computation and also the number of the most important operations in the process of building the upright SURF based descriptors. All timing results were obtained on an Intel i7 2.8GHz computer.

In Table 7, the number of integral image areas means the number of

Case	U-SURF	MU-SURF	MGU-SURF	GU-SURF	GU-SURF	GU-SURF
Dimension	**64**	**64**	**64**	**36**	**64**	**144**
# First-Order Wavelets	800	2592	2592	648	800	1152
# Second-Order Wavelets	0	0	3888	972	1200	1728
# Gaussian Weights	800	2608	0	0	0	0
Square area	20 × 20	24 × 24	24 × 24	18 × 18	20 × 20	24 × 24
# Integral Image Areas	1600	5184	15552	3888	4800	6912
Time (ms)	0.03	0.16	0.30	0.06	0.07	0.10

Table 7: Descriptor Building Process: Number of operations, square area and average computation time per descriptor keypoint.

areas that we have to obtain in order to compute the descriptor. Based on OpenSURF's implementation details [12], one can estimate first-order Haar wavelets L_x, L_y with just the difference of two areas of the integral image for each of the first-order wavelets. For each of the second-order Haar wavelets L_{xx}, L_{yy} it is necessary to compute two areas of the integral image and sum these areas in a proper way. Finally, the most consuming Haar wavelet is L_{xy}, since it requires the computation of 4 areas of the integral image. For example, for the U-SURF (64) case, the total number of areas of the integral image that we need to compute is: $(4 \times 4) \cdot (5 \times 5) \cdot (2 + 2) = 1600$. Due to the extra-padding of $2s$, the MU-SURF (64) case yields: $(4 \times 4) \cdot (9 \times 9) \cdot (2 + 2) = 5184$. On the other hand, the GU-SURF (64) case yields: $(4 \times 4) \cdot (5 \times 5) \cdot (2 + 2 + 2 + 2 + 4) = 4800$. However, the core observation is that for the GU-SURF (64) descriptor one can obtain substantial speed-up for those points in the rectangular grid where the gradient is equal to zero. For those cases we do not need to compute the second-order wavelets, since gauge coordinates are not defined for these points. This corresponds to regions of the images of equal value, and therefore these regions are non-Morse.

Using the same settings as described in Table 1, we can show the fraction of non-Morse points among all the points where Haar wavelets were evaluated. For example, for the following images the ratio is: Leuven Image 1 (17.96%), Bikes Image 1 (17.73%) and Iguazu Image 1 (32.43%). Another computational advantage of the G-SURF descriptor is that it is not necessary to interpolate the Haar wavelet responses with respect to a dominant orientation, since gauge derivatives are rotation invariant.

As explained above, the number of operations for U-SURF (64) is the smallest, yielding a small computation time per descriptor, but the performance is the worst compared to the other SURF-based cases. NGU-SURF (64) descriptor has similar computation times to the U-SURF descriptor, with the advantage that no Gaussian weighting operations are necessary and exhibiting much better performance. The modified version of the descriptors introduces more computations in the descriptor building process, since the square area is $24s \times 24s$. This yields higher computation times per descriptor. In particular, for the MGU-SURF (64) descriptor, the number of integral image areas is the highest (15552), and also the associated computation time per descriptor (0.30 ms). However, this descriptor only offers small advantages in performance against GU-SURF (36), GU-SURF (64) and GU-SURF (144) when we have sequences with strong changes in viewpoints and non-planar rotations (e.g. Wall, Graffiti, Liberty and Notre Dame datasets). In addition, GU-SURF (36), GU-SURF (64) and GU-SURF (144) are faster to compute than MU-SURF (64) and also exhibit much better performance. For the U-SIFT (128) descriptor, we obtained an average computation time per keypoint of 0.42 ms. Besides, for any SIFT-based descriptor one needs to compute the Gaussian scale space since the gradients are precomputed for all levels of the pyramid [14]. Pre-computing the scale space is a highly consuming task in contrast to the fast integral image computation. We obtained a computation time of 186 ms for the SIFT scale space generation, whereas for the SURF integral image we obtained 2.62 ms. For the CVU-SURF case, we obtained an average computation time per keypoint of 0.05 ms.

According to these results, it is clear that image matching using the G-SURF descriptors can be accomplished in real-time, with high matching performance. For example, we think that GU-SURF (36) and GU-SURF (64) are of special interest to be used efficiently in real-time SfM and SLAM applications due to excellent matching performance and computational efficiency.

7. Conclusions

We have presented a new family of multiscale local descriptors, a novel high performance SURF-inspired set of descriptors based on gauge coordinates which are easy to implement but are theoretically and intuitively highly appealing. Image matching quality is considerably improved relative to standard SURF and other state of the art techniques, especially for those scenarios where the image transformation is small in terms of change in viewpoint or the image transformation is related to blur, rotation, changes in lighting, JPEG compression or random Gaussian noise. Our upright descriptors GU-SURF (64) and GU-SURF (36) are highly suited to SfM and SLAM applications due to excellent matching performance and computational efficiency. Furthermore, the rotation invariant form of the descriptors is not necessary in applications where the camera only rotates around its vertical axis, which is the typical case of visual odometry [11, 44] or SLAM [45] applications. We also showed successful results of our familiy of descriptors in large-scale 3D SfM applications and visual categorization problems.

Another important conclusion that we showed in this paper, is that descriptors based on gauge-derivatives can exhibit much higher performance than first-order local derivatives based descriptors. This is possible, due to the extra invariance offered by gauge-derivatives and also our G-SURF descriptors have comparable computational cost with respect to other approaches.

As future work we are interested in testing the usefulness of G-SURF descriptors for more challenging object recognition tasks (e.g. The PASCAL Visual Object Classes Challenge). In addition, we also plan to incorporate our descriptors into real-time SfM applications and evaluate them in loop closure detection problems such as in [46]. Future work will aim at optimising the code for additional speed up and also we will exploit the use of gauge coordinates in the detection of features in non-linear scale spaces. Moreover, we would like to introduce our gauge-based descriptors on a DAISY-like framework [47] for performance evaluation on different computer vision applications.

According to the obtained results and other successful approaches such as *geometric blur*, we hope that in the near future we can break with the standard scale-space paradigm in computer vision algorithms. In the standard scale-space paradigm the true location of a boundary at a coarse scale is not directly available in the coarse scale image. The reason for this is simply because Gaussian blurring does not respect the natural boundaries of objects. We believe that introducing new invariant features that fully exploit non-linear diffusion scale spaces (both in detection and local description of features) can represent step forward improvements on traditional image matching and object recognition applications.

References

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, R. Szeliski, Building Rome in a Day, in: Intl. Conf. on Computer Vision (ICCV), 2009.

[2] G. Csurka, C. Bray, C. Dance, L. Fan, Visual categorization with bags of keypoints, in: In Workshop on Statistical Learning in Computer Vision, ECCV, 2004, pp. 1–22.

[3] D. Lowe, Object recognition from local scale-invariant features, in: Intl. Conf. on Computer Vision (ICCV), Corfu, Greece, 1999, pp. 1150–1157.

[4] B. M. ter Haar Romeny, Front-End Vision and Multi-Scale Image Analysis. Multi-Scale Computer Vision Theory and Applications, written in Mathematica, Kluwer Academic Publishers, 2003.

[5] H. Bay, A. Ess, T. Tuytelaars, L. V. Gool, SURF: Speeded up robust features, Computer Vision and Image Understanding 110 (3) (2008) 346–359.

[6] P. Perona, J. Malik, Scale-space and edge detection using annisotropic diffusion, IEEE Trans. Pattern Anal. Machine Intell. 12 (7) (1990) 1651–1686.

[7] L. Álvarez, P. Lions, J. Morel, Image selective smoothing and edge detection by nonlinear diffusion, SIAM Journal on Numerical Analysis (SINUM) 29 (1992) 845–866.

[8] T. Lindeberg, Feature detection with automatic scale selection, Intl. J. of Computer Vision 30 (2) (1998) 77–116.

[9] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Trans. Pattern Anal. Machine Intell. 27 (10) (2005) 1615–1630.

[10] M. Brown, H. Gang, S. Winder, Discriminative learning of local image descriptors, IEEE Trans. Pattern Anal. Machine Intell. 33 (1) (2011) 43–57.

[11] D. Nistér, O. Naroditsky, J. Bergen, Visual Odometry, in: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2004.

[12] C. Evans, Notes on the OpenSURF library, Tech. Rep. CSTR-09-001, University of Bristol (January 2009).
URL http://www.cs.bris.ac.uk/Publications/Papers/2000970.pdf

[13] A. Vedaldi, B. Fulkerson, VLFeat: An open and portable library of computer vision algorithms, http://www.vlfeat.org/ (2008).

[14] D. Lowe, Distinctive image features from scale-invariant keypoints, Intl. J. of Computer Vision 60 (2) (2004) 91–110.

[15] P. Viola, M. J. Jones, Robust real-time face detection, Intl. J. of Computer Vision 57 (2) (2004) 137–154.

[16] M. Agrawal, K. Konolige, M. R. Blas, CenSurE: Center Surround Extremas for realtime feature detection and matching, in: Eur. Conf. on Computer Vision (ECCV), 2008.

[17] A. C. Berg, J. Malik, Geometric blur for template matching, in: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Hawaii, USA, 2001, pp. 607–614.

[18] C. Schmid, R. Mohr, Local grayvalue invariants for image retrieval, IEEE Trans. Pattern Anal. Machine Intell. 19 (5) (1997) 530–535.

[19] L. M. J. Florack, B. M. ter Haar Romeny, J. J. Koenderink, M. A. Viergever, Cartesian differential invariants in scale-space, Journal of Mathematical Imaging and Vision 3 (1993) 327–348.

[20] C. Rothwell, A. Zisserman, D. Forsyth, J. Mundy, Canonical frames for planar object recognition, in: Eur. Conf. on Computer Vision (ECCV), 1992, pp. 757–772.

[21] W. Freeman, E. Adelson, The design and use of steerable filters, IEEE Trans. Pattern Anal. Machine Intell. 13 (9) (1991) 891–906.

[22] L. V. Gool, T. Moons, D. Ungureanu, Affine/photometric invariants for planar intensity patterns, in: Eur. Conf. on Computer Vision (ECCV), 1996, pp. 642–651.

[23] B. Platel, E. Balmachnova, L. Florack, B. ter Haar Romeny, Top-Points as interest points for image matching, in: Eur. Conf. on Computer Vision (ECCV), 2006.

[24] M. Goesele, B. Curless, S. Seitz, Multi-view stereo revisited, in: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), New York, USA, 2006, pp. 2402–2409.

[25] M. Goesele, N. Snavely, B. Curless, H. Hoppe, S. Seitz, Multi-view stereo for community photo collections, in: Intl. Conf. on Computer Vision (ICCV), Rio de Janeiro, Brasil, 2007, pp. 14–20.

[26] C. Schmid, R. Mohr, Matching by local invariants, Tech. rep., INRIA (Aug. 1995).

[27] L. Álvarez, F. Guichard, P. Lions, J. M. Morel, Axioms and fundamental equations of image processing, Arch. for Rational Mechanics 123 (3) (1993) 199–257.

[28] J. Koenderink, The structure of images, Biological Cybernetics 50 (1984) 363–370.

[29] A. Kuijper, Geometrical PDEs based on second-order derivatives of gauge coordinates in image processing, Image and Vision Computing 27 (8) (2009) 1023–1034.

[30] V. Caselles, J.-M. Morel, C. Sbert, An axiomatic approach to image interpolation, IEEE Trans. on Image Processing.

[31] P. Dreuw, P. Steingrube, H. Hanselmann, H. Ney, SURF-Face: Face Recognition under Viewpoint Consistency Constraints, in: British Machine Vision Conf. (BMVC), 2009.

[32] J. Damon, Local Morse theory for solutions to the heat equation and Gaussian blurring, Journal of Differential Equations 115 (2) (1995) 368–401.

[33] H. Bay, T. Tuytelaars, L. V. Gool, SURF: Speeded up robust features, in: Eur. Conf. on Computer Vision (ECCV), 2006.

[34] A. Vedaldi, An open implementation of the SIFT detector and descriptor, Tech. Rep. 070012, UCLA CSD (2007).

[35] K. Mikolajczyk, C. Schmid, Scale and affine invariant interest point detectors, Intl. J. of Computer Vision 60 (2004) 63–86.

[36] R. Liu, Y. Wang, SAR image matching based on speeded up robust feature, in: WRI Global Congress on Intelligent Systems, 2009.

[37] M. Calonder, V. Lepetit, P. Fua, BRIEF: Binary Robust Independent Elementary Features, in: Eur. Conf. on Computer Vision (ECCV), 2010.

[38] G. Hua, M. Brown, S. Winder, Discriminant embedding for local image descriptors, in: Intl. Conf. on Computer Vision (ICCV), Rio de Janeiro, Brazil, 2007.

[39] H. Bay, B. Fasel, L. V. Gool, Interactive Museum Guide: Fast and Robust Recognition of Museum Objects, in: Proceedings of the first international workshop on mobile vision, 2006.

[40] R. Fergus, P. Perona, A. Zisserman, Object class recognition by unsupervised scale-invariant learning, in: IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2003, pp. 264–271.

[41] J. Yebes, P. Alcantarilla, L. Bergasa, Occupant monitoring system for traffic control based on visual categorization, in: IEEE Intelligent Vehicles Symposium (IV), 2011.

[42] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2007.

[43] D. Lewis, Naive (bayes) at forty: The independence assumption in information retrieval, in: Eur. Conf. on Machine Learning (ECML), 1998, pp. 4–15.

[44] M. Kaess, K. Ni, F. Dellaert, Flow separation for fast and robust stereo odometry, in: IEEE Intl. Conf. on Robotics and Automation (ICRA), Kobe, Japan, 2009.

[45] A. J. Davison, I. D. Reid, N. D. Molton, O. Stasse, MonoSLAM: Real-time single camera SLAM, IEEE Trans. Pattern Anal. Machine Intell. 29 (6).

[46] A. Angeli, D. Filliat, S. Doncieux, J. A. Meyer, Fast and Incremental Method for Loop-Closure Detection using Bags of Visual Words, IEEE Trans. Robotics 24 (2008) 1027–1037.

[47] E. Tola, V. Lepetit, P. Fua, DAISY: An efficient dense descriptor applied to wide-baseline stereo, IEEE Trans. Pattern Anal. Machine Intell. 32 (5) (2010) 815–830.