Published in Image Processing On Line on 2015–07–20.
Submitted on 2013–02–12, accepted on 2015–05–30.
ISSN 2105–1232 © 2015 IPOL & the authors CC–BY–NC–SA
This article is available online with supplementary materials,
software, datasets and online demo at
https://doi.org/10.5201/ipol.2015.69

# An Analysis of the SURF Method

Edouard Oyallon[1], Julien Rabin[2]

[1] ENS, Département Informatique, France (edouard.oyallon@ens.fr)
[2] University of Caen, GREYC - ENSICAEN, France (julien.rabin@unicaen.fr)

*Communicated by* Yann Gousseau *Demo edited by* Edouard Oyallon

## Abstract

The SURF method (*Speeded Up Robust Features*) is a fast and robust algorithm for local, similarity invariant representation and comparison of images. Similarly to many other local descriptor-based approaches, interest points of a given image are defined as salient features from a scale-invariant representation. Such a multiple-scale analysis is provided by the convolution of the initial image with discrete kernels at several scales (box filters). The second step consists in building orientation invariant descriptors, by using local gradient statistics (intensity and orientation). The main interest of the SURF approach lies in its fast computation of operators using box filters, thus enabling real-time applications such as tracking and object recognition. The SURF framework described in this paper is based on the PhD thesis of H. Bay [ETH Zurich, 2009], and more specifically on the paper co-written by H. Bay, A. Ess, T. Tuytelaars and L. Van Gool [Computer Vision and Image Understanding, 110 (2008), pp. 346–359]. An implementation is proposed and used to illustrate the approach for image matching. A short comparison with a state-of-the-art approach is also presented, the SIFT algorithm of D. Lowe [International Journal of Computer Vision, 60 (2004), pp. 91–110], with which SURF shares a lot in common.

## Source Code

The source code and the online demo are accessible at the IPOL web page of this article[1]. The proposed implementation of the SURF algorithm is written in C++ ISO/ANSI. It performs features extraction from digital images and provides local correspondences for a pair of images. An epipolar geometric consistency checking may additionally be used to discard mismatches when considering two pictures from the same scene. This optional post-processing uses the ORSA algorithm by B. Stival and L. Moisan [International Journal of Computer Vision, 57 (2004), pp. 201–218].

**Keywords:** SURF; SIFT; image comparison; local descriptors; feature detection; feature matching

[1] https://doi.org/10.5201/ipol.2015.69

# 1 Introduction

## 1.1 Context, Motivation and Previous Work

Over the last decade, the most successful algorithms to address various computer vision problems have been based on local, affine-invariant descriptions of images. The targeted applications encompass, but are not limited to, image stitching and registration, image matching and comparison, indexation and classification, depth estimation and 3-D reconstruction. Like many image processing approaches, a popular and efficient methodology is to extract and compare local patches from different images. However, in order to design fast algorithms and obtain compact and locally invariant representations, some selection criteria and normalization procedures are required. A sparse representation of the image is also necessary to avoid extensive patch-wise comparisons that would be computationally expensive. The main challenges are thus to keep most salient features from images (such as corners, blobs or edges) and then to build a local description of these features which is invariant to various perturbations, such as noisy measurements, photometric changes, or geometric transformation.

Such problems have been addressed since the early years of computer vision, resulting in a very prolific literature. Without being exhaustive, one may first mention the famous Stephen-Harris corner detector [9], and the seminal work of Lindeberg on multi-scale feature detection (see e.g. [12]). Secondly, invariant local image description from multi-scale analysis is a more recent topic: SIFT descriptors [15] –from which SURF [2] is largely inspired– are similarity invariant descriptors of an image that are also robust to noise and photometric change. Some algorithms extend this framework to fully affine transformation invariance [18, 28], and dense representation [26].

The main interest of the SURF approach [2] studied in this paper is its fast approximation of the SIFT method. It has been shown to share the same robustness and invariance while being faster to compute.

## 1.2 Outline and Algorithm Overview

The SURF algorithm is in itself based on two consecutive steps (feature detection and description) that are described in Sections 4 and 5. The last step is specific to the application targeted. In this paper, we chose image matching as an illustration (Sections 5.4 and 6).

**Multi-scale analysis** Similarly to many other approaches, such as the SIFT method [15], the detection of features in SURF relies on a scale-space representation, combined with first and second order differential operators. The originality of the SURF algorithm (*Speeded Up Robust Features*) is that these operations are speeded up by the use of box filters techniques (see e.g. [25], [27]) that are described in Section 2. For this reason, we will use the term *box-space* to distinguish it from the usual Gaussian scale-space. While the Gaussian scale space is obtained by convolution of the initial images with Gaussian kernels, the *discrete box-space* is also obtained by convolving the original image with box filters at various scales. A comparison between these two scale-spaces is proposed in Section 3.

**Feature detection** During the detection step, the local maxima in the *box-space* of the "determinant of Hessian" operator are used to select interest point candidates (Section 4). These candidates are then validated if the response is above a given threshold. Both the scale and location of these candidates are then refined using quadratic fitting. Typically, a few hundred interest points are detected in a megapixel image.

**Feature description** The purpose of the next step described in Section 5 is to build a descriptor of the neighborhood of each point of interest that is invariant to view-point changes. Thanks to multi-scale analysis, the selection of these points in the box-space provides scale and translation invariance. To achieve rotation invariance, a dominant orientation is defined by considering the local gradient orientation distribution, estimated from Haar wavelets. Using a spatial localization grid, a 64-dimensional descriptor is then built, based on first order statistics of Haar wavelets coefficients.

**Feature matching** Finally, when considering the image matching task (e.g. for image registration, object detection, or image indexation), the local descriptors from several images are matched. Exhaustive comparison is performed by computing the Euclidean distance between all potential matching pairs. A nearest-neighbor distance-ratio matching criterion is then used to reduce mismatches, combined with an optional RANSAC-based technique [21, 20] for geometric consistency checking.

**Outline** The rest of the paper is structured as follows

- **Section 2** SURF multi-scale representation based on box filters;
- **Section 3** Comparison with linear scale space analysis;
- **Section 4** Interest points detection;
- **Section 5** Invariant descriptor construction and comparison;
- **Section 6** Experimental validation and comparison with other approaches.

### 1.3 Notations

The main notations used throughout this paper are summarized in Table 1.

## 2 Multi-scale Analysis via Box Filters: the Box-Space

The scale invariant analysis is usually achieved by simulating Gaussian zoom-out of the considered image. The corresponding linear scale-space representation is obtained by Gaussian convolutions of the image at several scales [12, 15] (see [7] for a survey of Gaussian Convolution Algorithms). To speed up this time-consuming procedure, the SURF approach [2] approximates the Gaussian kernels and its spatial derivatives by uniform kernels with rectangular (thus separable) support, referred to from now on as *box filters*. Although being a rough approximation, these filters may be evaluated in linear time using the so-called *integral image* technique. By analogy, we will now refer to this approximated scale-space, as the *box-space*. In SURF, the box-space is defined on the triplet coordinates $(x, y, L) \in \Omega \times \mathcal{L}$ where $L \in \mathcal{L} \subset \mathbb{N}$ is the scale parameter that characterizes the size of the box filters involved in differential operators.

In this section, we first recall the definition of the integral image and the convolution property of box filters. The various differential operators involved in SURF feature detection and description are then defined.

### 2.1 Discrete Convolution

The SURF algorithm makes extensive use of discrete convolution with symmetric kernels. In our discrete setting, kernels will be approximated by filters with a finite support. Therefore, the discrete

| Symbol | Description | Definition |
|---|---|---|
| $u$ | image on a continuous domain | $u : \mathbb{R}^2 \mapsto \mathbb{R}$ |
| $\mathbf{u}$ | Discretized image | $\mathbf{u} : (x,y) \in \Omega \subset \mathbb{Z}^2 \mapsto [0,256[\subset \mathbb{R}$ |
| $\mathbf{U}$ | Integral image | Formula (1) |
| $\delta_{(a,b)}$ | Discrete Dirac | $\delta_{(a,b)}(x,y) = \begin{cases} 1 & \text{if } (x,y)=(a,b) \\ 0 & \text{otherwise} \end{cases}$ |
| $*$ | Convolution/Discrete convolution | Section 2.1 |
| $\mathbf{D}_x^L, \mathbf{D}_y^L$ | First order box filters, at scale $L$ | Section 2.3.1, Equation (6) & (7) |
| $\mathbf{D}_{xx}^L, \mathbf{D}_{yy}^L, \mathbf{D}_{xy}^L$ | Second order box filters, at scale $L$ | Section 2.3.2, Equation (10) & (11) |
| $\mathbf{DoH}^L$ | Determinant of Hessian, at scale $L$ | Section 4.1, Equation (24) |
| $\boldsymbol{\Delta}^L$ | Laplacian, at scale $L$ | Section 4.3, Equation (28) |
| $D_x, D_{xy}, \ldots$ | First and second order derivative operator | |
| $DoH_\sigma$ | Scale normalized Determinant of Hessian at scale $\sigma$ | Section 3.3.3, Equation (22) |
| $\Delta_\sigma$ | Scale normalized Laplacian, at scale $\sigma$ | Section 3.3.3, Equation (23) |
| $[\![., .]\!]$ | Discrete interval | $[\![n,m]\!] = \{n, n+1, \ldots, M\}$, $\forall\, n < m \in \mathbb{N}$ |
| $\lfloor \cdot \rceil$ | Round | Closest integer |
| $X^T$ | Matrix transposition | $X_{i,j}^T = X_{j,i}$ |
| $\mathbf{1}_X$ | Indicator function | $\mathbf{1}_X(x) = \begin{cases} 1 & \text{if } x \in X \\ 0 & \text{otherwise} \end{cases}$ |
| $\mathcal{B}_r(X)$ | Ball of radius $r$, centered in $X$ at $\Omega$ | See Section 5 |
| $\Vert\cdot\Vert$ | Norm | See Sections 3.3 , 4.1, 5.3, and 5.4 |
| $\angle\, \cdot$ | Oriented angle with canonical vector $(1,0)^T$, $\angle : \mathbb{R}^2 \mapsto [-\pi, \pi)$ | See Section 5.2 |
| $R_\theta$ | Rotation Matrix in $\mathbb{R}^2$ | See Formula (41) |

Table 1: List of main symbols and operators.

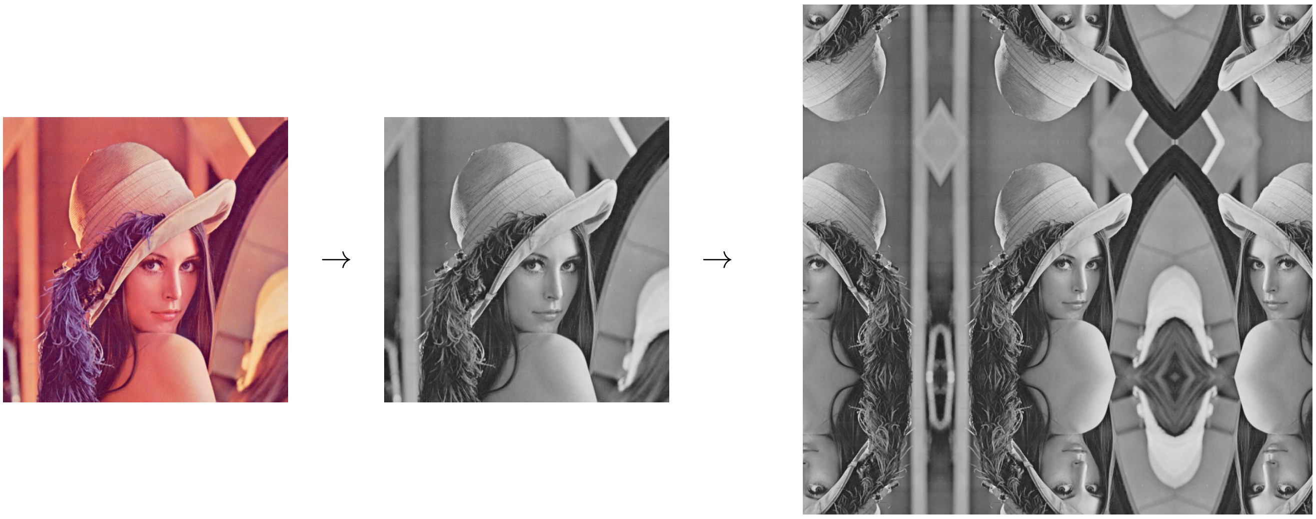


Figure 1: Illustration of preprocessing to create symmetric border condition. (Left) original *lena* image; (Middle) conversion in gray values and image dynamic stretching; (Right) extension by symmetrization.

convolution of a discrete function $\mathbf{f} : \mathbb{Z}^2 \to \mathbb{R}$ with a discrete filter $\mathbf{g} : \Omega \subset \mathbb{Z}^2 \to \mathbb{R}$ with finite support $\Omega$ is defined as

$$\forall\,(x, y) \in \mathbb{Z}^2,\ (\mathbf{f} * \mathbf{g})(x, y) := \sum_{(i,j)\in\Omega} \mathbf{f}(x - i, y - j)\, \mathbf{g}(i, j). \tag{1}$$

**Borders condition** In practice, an image $f$ is known on a finite domain. As a result, boundary conditions must be handled properly in order to lessen visual artifacts at the borders. In our implementation, this difficulty is tackled through the use of symmetric boundary conditions. The symmetrization of the input image is performed with respect to the first and last rows and columns, so that there is no duplication of pixels at the boundaries. More precisely, the original image is symmetrically extended on each side using the largest box filter size (see Figure 1) before computing the integral image.

**Image sub-sampling** Contrary to many other multi-scale approaches that build Gaussian pyramid combined with dyadic sub-sampling to save computation time, the integral image technique detailed hereafter permits fast evaluation of differential operators at different scales without sub-sampling. However, sub-sampling may still be used to speed-up feature detection at larger scales (which is performed in the proposed implementation).

**Image interpolation** Note that, as suggested in [1, 2] to improve the accuracy of the SURF method, the input image may be over-sampled and interpolated using a bilinear scheme. However, in this article we chose not to implement it.

**Image dynamic** The last preprocessing consists in a simple affine stretching of the image dynamic to $[\![0, 255]\!]$.

## 2.2 Integral Image and Box Filters

Let $\mathbf{u}$ be the processed digital image defined over the pixel grid $\Omega = [\![0, N-1]\!] \times [\![0, M-1]\!]$, where $M$ and $N$ are positive integers. In the following, we only consider quantized gray valued images (taking values in the range $[\![0, 255]\!]$), which is the simplest way to achieve robustness to color modifications, such as a white balance correction.

The integral image of $\mathbf{u}$ for $(x, y) \in \Omega$ is

$$\mathbf{U}(x, y) := \sum_{0 \le i \le x} \sum_{0 \le j \le y} \mathbf{u}(i, j). \tag{2}$$

An illustration for $\mathbf{u} = \delta_{(a,b)}$ is shown in Figure 2. We now focus on the convolution of $\mathbf{u}$ with a 2-D uniform function $\mathbf{B}_\Gamma$ over the domain $\Gamma \subset \Omega$:

$$\mathbf{B}_\Gamma(x, y) := \mathbf{1}_\Gamma(x, y) = \begin{cases} 1 & \text{if } (x, y) \in \Gamma \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

When considering any rectangular domain $\Gamma = [\![a, b]\!] \times [\![c, d]\!]$ (meaning that the discrete domain $\Gamma$ is separable in rows and columns coordinates), the convolution of the box filter $\mathbf{B}_\Gamma$ with the discrete image $u$ may be directly expressed from the integral image $\mathbf{U}$

$$\begin{aligned} \forall\, (x, y) \in \Omega, \quad (\mathbf{B}_\Gamma * \mathbf{u})(x, y) \;=\; & \quad \mathbf{U}(x-a, y-c) \quad + \quad \mathbf{U}(x-b-1, y-d-1) \\ & - \; \mathbf{U}(x-a, y-d-1) \; - \; \mathbf{U}(x-b-1, y-c). \end{aligned} \tag{4}$$

As illustrated in Figure 3, and thanks to the previous formula, the pre-computation of an integral image permits to convolve any image with a box filter in *three operations* and *four memory accesses*. We assume from now on that the symmetric border condition is also applied to the integral image, as described in Figure 1.

## 2.3 Differential Operators in SURF

In the SURF framework, differential operators are approximated using box filters at different scales. Without going here into too much detail, the scale of these filters is parametrized by the variable $L \in \mathbb{N}$. More precisely, this parameter relates to the size of the first and second order box filters. For more details on the sampling strategy of $L$, see Section 3.2.

### 2.3.1 First Order Box Filters

The finite difference operator of a discrete image $\mathbf{u}$ at scale $L$ according to the first (respectively second) coordinate is from now on referred to as $\mathbf{D}_x^L$ (resp. $\mathbf{D}_y^L$). It makes use of box filters with size

$$\ell(L) = \lceil 0.8L \rfloor \in \mathbb{N}. \tag{5}$$

Recall that $\lceil \cdot \rfloor$ denotes the rounding operation to the nearest integer. In order to lighten the notations, we do not explicitly write the dependency of $\ell$ towards $L$ in the following.

The first order box filters are defined as convolution filters:

$$\mathbf{D}_x^L\, \mathbf{u} := \left(\mathbf{B}_{[\![-\ell,-1]\!] \times [\![-\ell,\ell]\!]} - \mathbf{B}_{[\![1,\ell]\!] \times [\![-\ell,\ell]\!]}\right) * \mathbf{u}, \tag{6}$$

$$\mathbf{D}_y^L\, \mathbf{u} := \left(\mathbf{B}_{[\![-\ell,\ell]\!] \times [\![-\ell,-1]\!]} - \mathbf{B}_{[\![-\ell,\ell]\!] \times [\![1,\ell]\!]}\right) * \mathbf{u}. \tag{7}$$

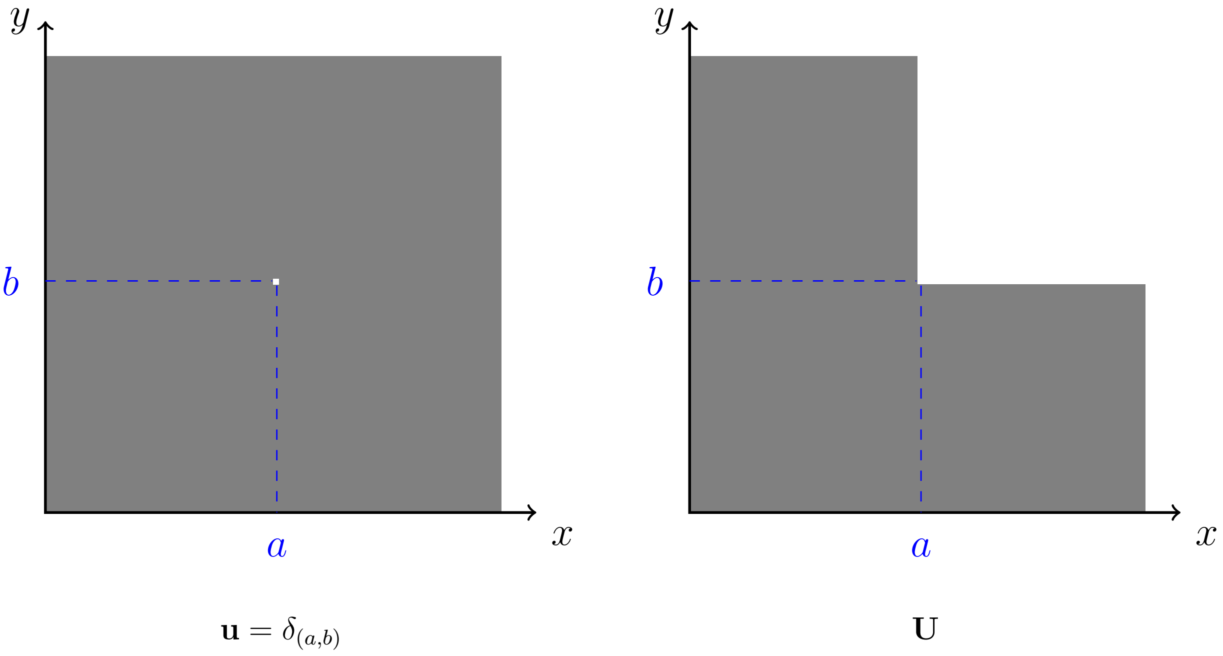


Figure 2: Example of an integral image $\mathbf{U}$ built from a synthetic image $\mathbf{u}$. (Left) An image $\mathbf{u}$ with a single pixel whose value is non zero. (Right) Resulting integral image $\mathbf{U}$.

See Figure 4 for an illustration. Observe that, when $\ell = 1$, these filters correspond to the average symmetric finite difference scheme at pixel scale

$$\mathbf{D}_x^1\,\mathbf{u} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & -1 \end{bmatrix} * \mathbf{u} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} * \mathbf{u}.$$

We will refer to these filters as *first order box filters*, since they can be interpreted as a discrete approximation of first-order derivative operators at a given scale $\ell$. We obtain the following explicit formula for the impulse response of a $\mathbf{D}_y^L$ operator (see Figure 4 for an illustration)

$$\mathbf{D}_y^L\,\delta(x,y) = \begin{cases} 1 & \text{if } (x,y) \in [\![-\ell,\ell]\!] \times [\![-\ell,-1]\!] \\ -1 & \text{if } (x,y) \in [\![-\ell,\ell]\!] \times [\![1,\ell]\!] \\ 0 & \text{otherwise} \end{cases} . \qquad (8)$$

**Computation using integral image** Using formula (4) and the pre-computed integral image $\mathbf{U}$, we can easily evaluate these operators with only seven additions, regardless of the parameter size $\ell$. For instance, considering Formula (4) with $b = -a = \ell$, and $c = -\ell$, $d = -1$ to compute

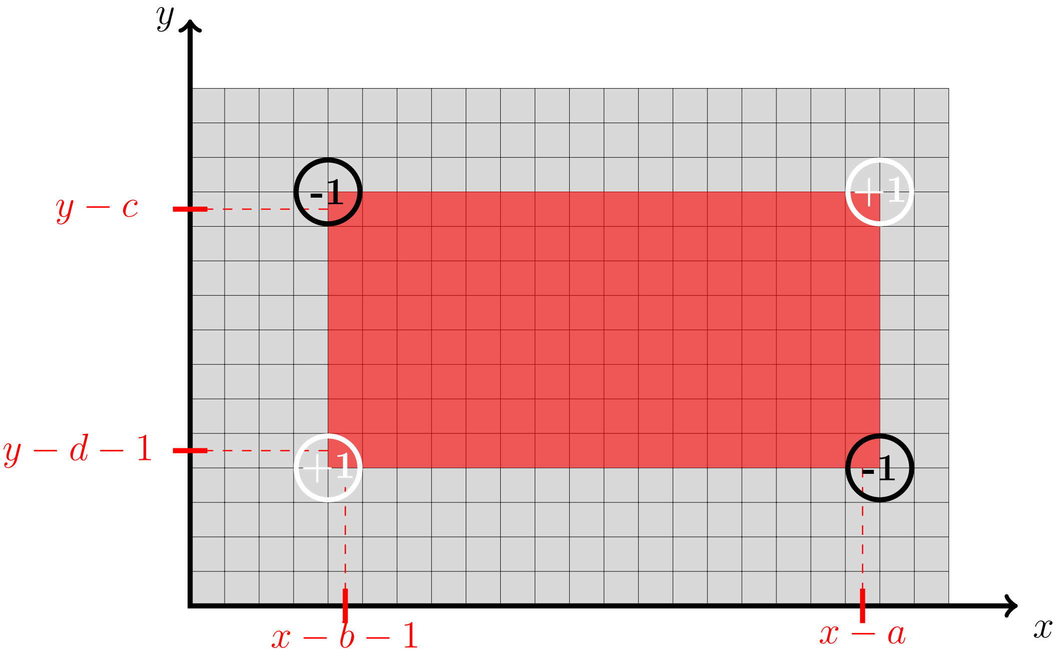


Figure 3: Computation of the discrete convolution with box filter on a rectangular domain using integral image. Only 3 operations and 4 memory accesses are required per pixel, according to Formula (4).

$\mathbf{B}_{[\![-\ell,\ell]\!]\times[\![-\ell,-1]\!]}$, followed by $c=+1$, $d=\ell$ for $\mathbf{B}_{[\![-\ell,\ell]\!]\times[\![1,\ell]\!]}$, we get

$$\begin{aligned}\forall\,(x,y)\in\Omega,\quad \mathbf{D}_y^L\,\mathbf{u}(x,y) \;=\; & \quad\mathbf{U}(x+\ell,y+\ell) && + \;\mathbf{U}(x-\ell-1,y)\\ & -\;\mathbf{U}(x+\ell,y) && - \;\mathbf{U}(x-\ell-1,y+\ell)\\ & -\;\mathbf{U}(x+\ell,y-1) && - \;\mathbf{U}(x-\ell-1,y-\ell-1)\\ & +\;\mathbf{U}(x+\ell,y-\ell-1) && + \;\mathbf{U}(x-\ell-1,y-1).\end{aligned} \tag{9}$$

**Remark** The original authors of SURF [2] simply claim to use Haar wavelets, but without giving any implementation details about the actual form of the kernels. Given this indeterminacy, we adopted anti-symmetric kernels (see Figure 4) because they do not introduce translation shifts.

### 2.3.2 Second Order Box Filters

In SURF, the multi-scale *second order differential operators* are again defined with box filters. These filters are parametrized by a scale variable $L\in\mathbb{N}$, which takes odd values. By analogy with second order difference schemes, the second order operators $\mathbf{D}_{xx}^L$ and $\mathbf{D}_{yy}^L$ at scale $L$ are defined as (see Figure 5 for an illustration)

$$\mathbf{D}_{xx}^L\,\mathbf{u}=(\mathbf{B}_{\Gamma_1}-3\mathbf{B}_{\Gamma_2})*\mathbf{u},\qquad\text{and}\qquad \mathbf{D}_{yy}^L\,\mathbf{u}=(\mathbf{B}_{\Gamma_3}-3\mathbf{B}_{\Gamma_4})*u, \tag{10}$$

where

$$\begin{cases}\Gamma_1=\left[\!\!\left[-\frac{3L-1}{2},\frac{3L-1}{2}\right]\!\!\right]\times[\![-(L-1),(L-1)]\!]\,,\\ \Gamma_2=\left[\!\!\left[-\frac{L-1}{2},\frac{L-1}{2}\right]\!\!\right]\times[\![-(L-1),(L-1)]\!]\subset\Gamma_1\end{cases}.$$

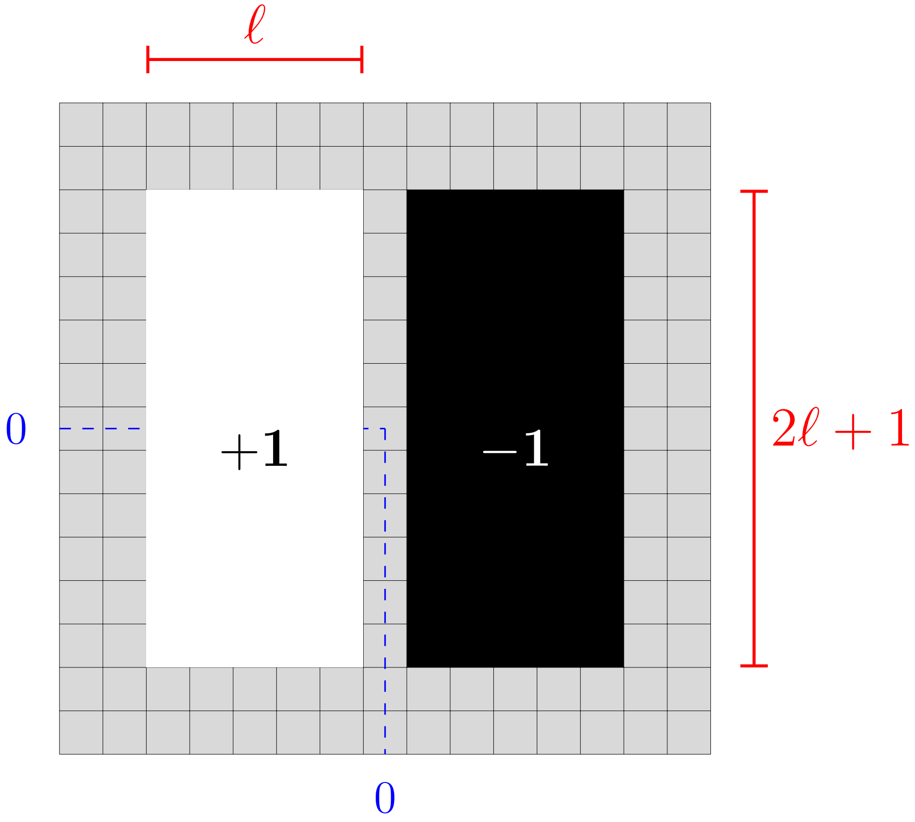


Figure 4: Illustration of the first order box filter. The differential operator $\mathbf{D}_x^L$ (defined in Equation (6)) according to the first coordinate $x$ is shown here at scale $L = 6$, which corresponds to the size parameter $\ell(L) = 5$ using relation (5). This operator, similar to a Haar wavelet, is involved in the construction of local SURF descriptors.

As illustrated in Figure 5, the first domain $\Gamma_1$ is the support of the filter (delimited by white areas), while the domain $\Gamma_2$ corresponds to the central part (shown in black). By permutation of the first and second coordinates, we get

$$\begin{cases} \Gamma_3 = [\![-(L-1),(L-1)]\!] \times [\![-\frac{3L-1}{2}, \frac{3L-1}{2}]\!] \,, \\ \Gamma_4 = [\![-(L-1),(L-1)]\!] \times [\![-\frac{L-1}{2}, \frac{L-1}{2}]\!] \subset \Gamma_3. \end{cases}$$

Likewise, the second order mixed derivative operator $\mathbf{D}_{xy}^L$ is written

$$\mathbf{D}_{xy}^L\, \mathbf{u} = \big(\mathbf{B}_{\Gamma_{++}} + \mathbf{B}_{\Gamma_{--}} - \mathbf{B}_{\Gamma_{-+}} - \mathbf{B}_{\Gamma_{+-}}\big) * \mathbf{u}, \tag{11}$$

where subscripts $++$, $-+$, $--$, $+-$ indicates respectively North-East, North-West, South-West and South-East quadrants

$$\begin{cases} \Gamma_{++} & = [\![1, L]\!] \times [\![1, L]\!] \,, \quad \text{(North-East quadrant)} \\ \Gamma_{-+} & = [\![-L, -1]\!] \times [\![1, L]\!] \,, \quad \text{(North-West quadrant)} \\ \Gamma_{--} & = [\![-L, -1]\!] \times [\![-L, -1]\!] \,, \quad \text{(South-West quadrant)} \\ \Gamma_{+-} & = [\![1, L]\!] \times [\![-L, -1]\!] \,, \quad \text{(South-East quadrant)} \end{cases} .$$

The corresponding filters are respectively shown in Figure 5. We can again compute their explicit formula (impulse responses):

$$\mathbf{D}^L_{xx}\,\delta(x,y) = \begin{cases} -2 & \text{if } (x,y)\in\Gamma_1\cap\Gamma_2 \\ +1 & \text{if } (x,y)\in\Gamma_1\setminus\Gamma_2 \\ 0 & \text{otherwise,} \end{cases} \quad , \quad \mathbf{D}^L_{yy}\,\delta(x,y) = \begin{cases} -2 & \text{if } (x,y)\in\Gamma_3\cap\Gamma_4 \\ +1 & \text{if } (x,y)\in\Gamma_3\setminus\Gamma_4 \\ 0 & \text{otherwise,} \end{cases} \tag{12}$$

and

$$\mathbf{D}^L_{xy}\,\delta(x,y) = \begin{cases} -1 & \text{if } (x,y)\in\Gamma_{+-}\cup\Gamma_{+-} \\ +1 & \text{if } (x,y)\in\Gamma_{++}\cup\Gamma_{--} \\ 0 & \text{otherwise.} \end{cases} \tag{13}$$

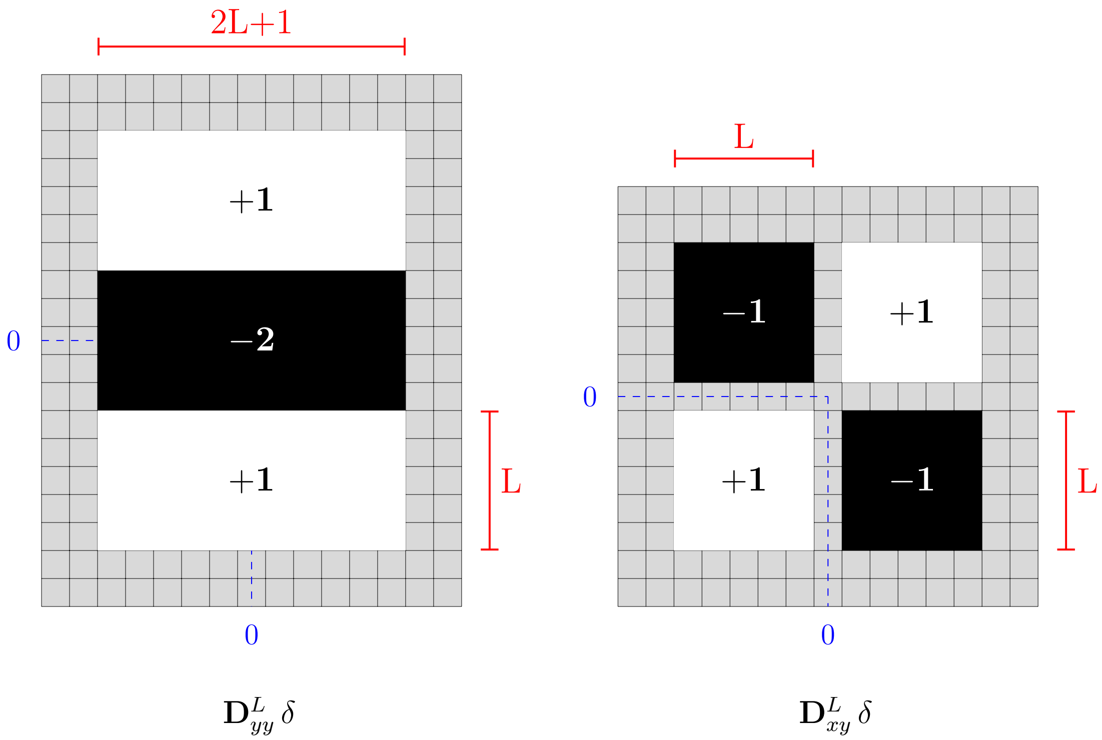


Figure 5: Illustration of second order box filters with scaling parameter $L$ (here $L=5$). (Left) $\mathbf{D}^L_{yy}$ from Equation (10). (Right) $\mathbf{D}^L_{xy}$ from Equation (11).

**Computation using integral image** The convolution with $\mathbf{D}^L_{xx}$ and $\mathbf{D}^L_{yy}$ is easily computed in 8 operations using the integral image (using 2 box filters, one addition and one multiplication), and 15 operations for $\mathbf{D}^L_{xy}$ (4 box filters and 3 additions). For obvious reasons, we do not give the explicit formulas.

Figure 7 shows the responses of second order box filters on a digital image (Figure 6). As we can see, these filters detect either horizontal, vertical or diagonal structures at large scale.

## 3 Comparison with Gaussian Scale-Space Representation

While not being used explicitly in SURF, we take interest here in the approximation of Gaussian kernels by box filters to understand the advantages and the limitations of the SURF approach.

Figure 6: Picture *japan* (image $\mathbf{u}$), at resolution 481 × 321 pixels.

## 3.1 Scale-Space Representation

**Linear scale space** The linear (Gaussian) scale-space representation of a real valued image $u : \mathbb{R}^2 \mapsto \mathbb{R}$ defined on a continuous domain is obtained by a convolution with the Gaussian kernel

$$u_\sigma := G_\sigma * u, \tag{14}$$

where $G_\sigma$ is the centered, isotropic and separable 2-D Gaussian kernel with variance $\sigma^2$

$$\forall (x,y) \in \mathbb{R}^2,\ G_\sigma(x,y) := \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = g_\sigma(x)\, g_\sigma(y) \quad \text{and} \quad g_\sigma(x) = \frac{1}{\sqrt{2\pi}\cdot\sigma} e^{-\frac{x^2}{2\sigma^2}}. \tag{15}$$

The variable $\sigma$ is usually referred to as the *scale parameter*.

**Discrete scale space** In practice, for the processing of a numerical image $\mathbf{u}$, this continuous filter is approximated using regular sampling, truncation and normalization:

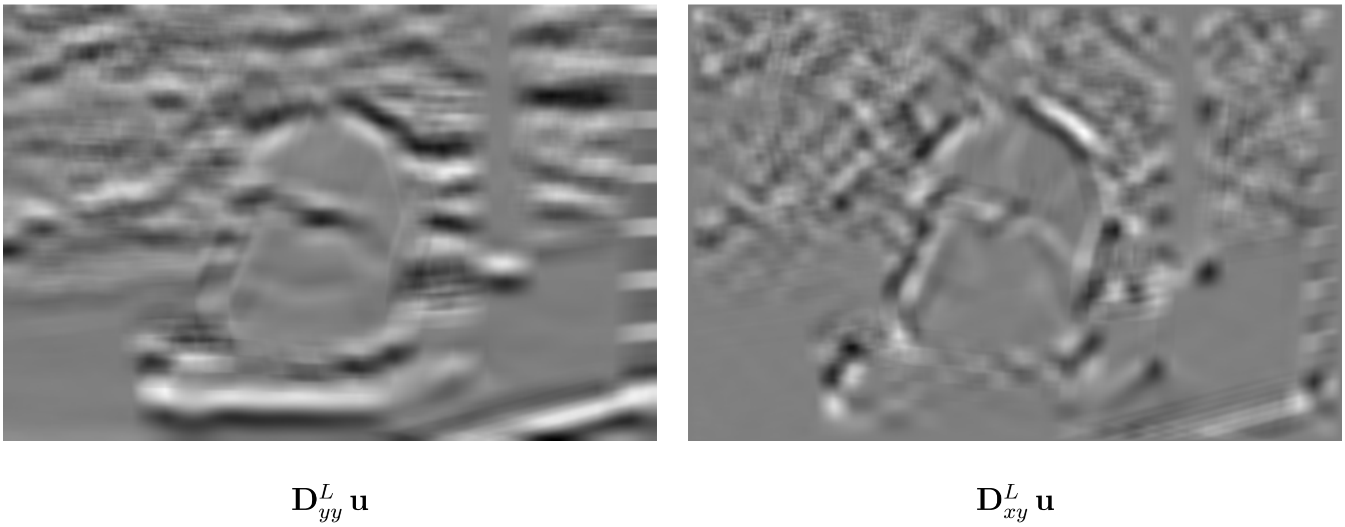
$$\forall i,j \in [\![-K,K]\!] \qquad \mathbf{G}_\sigma(i,j) = \frac{1}{C_K}\, G_\sigma(i,j)\ , \ \text{ where } C_K = \sum_{i,j=-K}^{K} G_\sigma(i,j). \tag{16}$$

The scale variable $\sigma$ is also sampled, generally using a power law, as discussed later in Section 3.2.

**Discrete box space** Making use of the aforementioned box filter technique, such a multi-scale representation can be (very roughly) approximated using a box filter with square domain $\Gamma = [\![-\gamma,\gamma]\!] \times [\![-\gamma,\gamma]\!]$

$$\mathbf{u}_\gamma := \frac{1}{(2\gamma+1)^2}\, \mathbf{B}_\Gamma * \mathbf{u}. \tag{17}$$

The question now is how to set the parameter $\gamma \in \mathbb{N}$ to get the best approximation of Gaussian zoom-out.



$\mathbf{D}^L_{yy}\,\mathbf{u}$ $\mathbf{D}^L_{xy}\,\mathbf{u}$

Figure 7: Illustration of second order box filters. (Left) $\mathbf{D}^L_{yy}\,\mathbf{u}$ and (Right) $\mathbf{D}^L_{xy}\,\mathbf{u}$, using the image *japan* (Figure 6) at scale $L = 17$.

**Second moment comparison** One may for instance choose to match the second order moment $\sigma^2$ of the 1D Gaussian $g_\sigma$ and the variance of the corresponding box filter, as suggested by [8]. This leads to the relation

$$\sigma_\gamma^2 = \sum_{i=-\gamma}^{\gamma} \frac{i^2}{2\gamma+1} = \frac{(2\gamma+1)^2-1}{12} = \frac{\gamma(\gamma+1)}{3}\,, \tag{18}$$

where $\sigma_\gamma^2$ is the variance of the centered 1D box filter with width $2\gamma+1$. Thus, for large values of filter size ($\gamma \gg 1$), we get approximately $\sigma_\gamma \approx \frac{\gamma}{\sqrt{3}} \approx 0.58\gamma$. Since $\gamma \in \mathbb{N}$ takes integer values, $\sigma_\gamma$ and $\sigma$ cannot match exactly in general. Moreover, due to the anisotropy of the box filter in 2D, it is impossible to match the covariance matrices.

**SURF scale parameter analogy** Note that box filters are only used to approximate the first and second order of Gaussian derivatives in the SURF algorithm, and not to approximate Gaussian filtering like in [8]. However, when considering the approximation of the second order Gaussian derivative

$$D_{xx}\, G_\sigma(x,y) = D_{xx}\, g_\sigma(x) \times g_\sigma(y) = \frac{1}{\sigma^2}\left(\frac{x^2}{\sigma^2} - 1\right) g_\sigma(x) \times g_\sigma(y)$$

by the second order box filter operator $\mathbf{D}^L_{xx}$ (using either relation (10) or (12)), we can see that the 1D Gaussian filter $g_\sigma(y)$ is approximated by the 1D box filter with parameter $\gamma = \frac{L-1}{2}$ (see Figure 5 for an illustration). The authors of SURF claim that the corresponding Gaussian scale is $\sigma = \frac{1.2}{3}L \approx 0.8\gamma$ for $\gamma \gg 1$, which is close but different to the value given by Formula (18): $\sigma_\gamma \approx 0.58\gamma$.

Other analogies could have been made for scale variables, for instance by considering zero crossing of second order derivative of Gaussians, second moment of Gaussian derivatives, mean-square error minimization, but each one provides different relations. In conclusion, defining a relation between the box parameters ($L$ and $\ell(L)$) and the Gaussian scale variable $\sigma$ seems quite arbitrary.

**Visual comparison** Figure 8 illustrates the difference between the linear scale-space representation obtained by Gaussian filtering (applied to the image in Figure 6) and the *box-space*, that is its approximation by box-filters when using relation (18). While being roughly similar, the approximated scale-space exhibits some strong vertical and horizontal artifacts due to the anisotropy and the high frequencies of the box kernels.

Again, whilst not being used explicitly in SURF, theses artifacts may explain some of the spurious detections of the SURF approach that will be exhibited later on.

### 3.2 Box-Space Sampling

Because of the definition of first and second order box filters, the size parameter $L$ cannot be chosen arbitrarily. Table 2 shows the sampling values and the corresponding variables used to mimic the linear scale space analysis. The following paragraphs give more detailed explanations.

| $o$ | $i$ | $\sigma(L)$ | $L$ | $3L$ | $l(L)$ | $w(L)$ |
|---|---|---|---|---|---|---|
| **1** | 1 | 1.2 | 3 | 9 | 2 | 0.9129 |
|  | **2** | **2.0** | **5** | **15** | **4** | **0.9487** |
|  | **3** | **2.8** | **7** | **21** | **6** | **0.9636** |
|  | 4 | 3.6 | 9 | 27 | 7 | 0.9718 |
| **2** | 1 | 2.0 | 5 | 15 | 4 | 0.9487 |
|  | **2** | **3.6** | **9** | **27** | **7** | **0.9718** |
|  | **3** | **5.2** | **13** | **39** | **10** | **0.9806** |
|  | 4 | 6.8 | 17 | 51 | 14 | 0.9852 |
| **3** | 1 | 3.6 | 9 | 27 | 7 | 0.9718 |
|  | **2** | **6.8** | **17** | **51** | **14** | **0.9852** |
|  | **3** | **10** | **25** | **75** | **20** | **0.9900** |
|  | 4 | 13.2 | 33 | 99 | 26 | 0.9924 |
| **4** | 1 | 6.8 | 17 | 51 | 14 | 0.9852 |
|  | **2** | **13.2** | **33** | **99** | **26** | **0.9924** |
|  | **3** | **19.6** | **49** | **147** | **39** | **0.9949** |
|  | 4 | 26.0 | 65 | 195 | 52 | 0.9962 |

Table 2: Box-space sampling values. The first two columns give the octave and level indexes. The scale $\sigma(L)$ defined in SURF by analogy to the linear scale space is given in the third column, using Equation (20). The size parameter $L = 2^o i + 1$ controls the width $2\ell(L) + 1$ of first order box operators $\mathbf{D}_x^L$ and $\mathbf{D}_y^L$ through the relation $\ell(L) = \lceil 0.8L \rfloor$, and also the width $3L$ of second order box filters $\mathbf{D}_{xx}^L$, $\mathbf{D}_{yy}^L$ and $\mathbf{D}_{xy}^L$. The last column $w(L)$ (Formula (26)) is required later to compute the determinant of Hessian (Section 4.1). The rows highlighted in bold font correspond to the scales values that are finally used to describe the image. Other rows are only required for computation purpose.

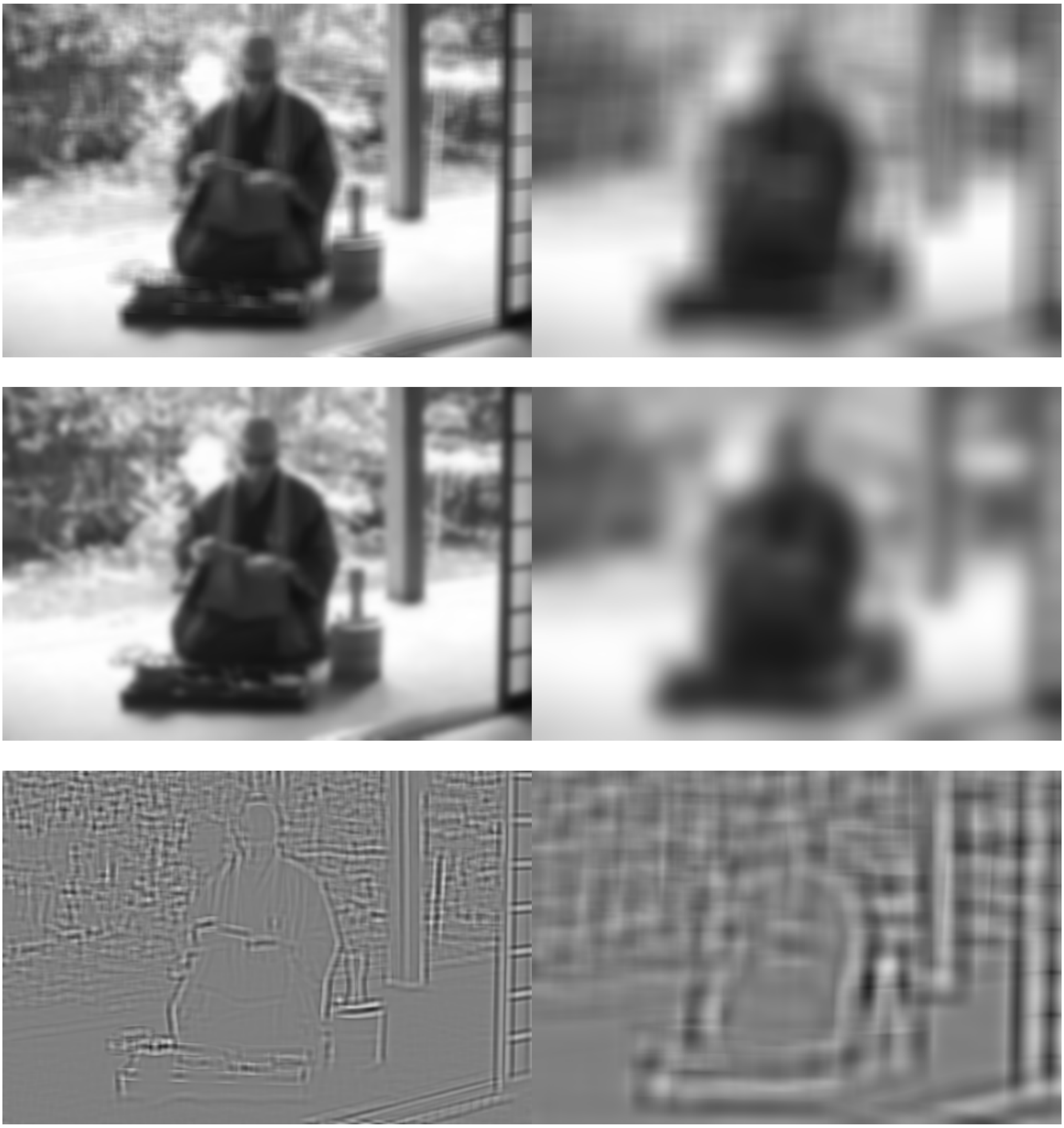


Figure 8: Comparison of the box space and the linear scale space. (Top) Convolution with squared and centered box filters with radii $\gamma = 5$ and $\gamma = 20$ (respectively from left to right). (Middle) Corresponding Gaussian filters with respective scales $\sigma_5 \approx 3.16$ and $\sigma_{20} \approx 11.83$, according to formula (18). (Bottom) Difference between Gaussian and Box filters (using a linear transform for visualization). We can see here that the box space is a rough approximation of the Gaussian scale space, that exhibits some artifacts due to the anisotropy and the high frequencies of the box kernels.

**Octave decomposition** Alike most multi-scale decomposition approaches (see e.g. [14, 16]), the box-space discretization in SURF relies on dyadic sampling of the scale parameter $L$. The box length representation is therefore divided into *octaves* (similarly to SIFT [15, 14]), which are indexed by parameter $o \in \{1, 2, 3, 4\}$, where a new octave is created for every doubling of the kernel size.

Note that, in order to save computation time, the filtered image is generally sub-sampled a factor two at every octave, as done for instance by SIFT [15], so that the complexity remains the same through scales. As pointed out by the authors of SURF [2], the computation time complexity does not depend on scale when using box filters.

However, while not being explicitly stated in the original paper [2], but as done in most implementations we have reviewed (for instance, this approximation is used in [3] but not in [5]), we still choose to use sub-sampling to speed-up the algorithm. More precisely, instead of evaluating the multi-scale operators at each pixel, we use a sampling "step" which depends on the octave level (this sampling is detailed in the next sections). Note that this strategy is consistent with the fact that the number of features is decreasing with respect to scale.

**Scale sampling** Each octave is also divided in several "levels" (indexed here by the parameter $i \in \{1, 2, 3, 4\}$). In the usual discrete scale space analysis, these levels correspond directly to the desired sampling of the scale variable $\sigma$, which parametrizes the discretized Gaussian kernels $\mathbf{G}_\sigma$ (see definition in Equation (16)). In SURF, the relation between scale $L$, octave $o$ and level $i$ variables is

$$L := 2^o\, i + 1\ . \tag{19}$$

These values are summarized in Table 2. Note that because of the non-maxima suppression involved in the feature selection, only intermediate levels are actually used to define interest points and local descriptors ($i \in \{2, 3\}$). The corresponding scales are indicated by rows with bold font in Table 2.

**Scale analogy with linear scale space** As discussed before in Section 3.1, we can define a scale analysis variable by analogy with the linear scale space decomposition. In [2], the scale parameter $\sigma(L)$ associated with octave $o$ and level $i$ is obtained by the following relation

$$\sigma(L) := \frac{1.2}{3}\,(2^o \times i + 1) = 0.4\, L. \tag{20}$$

Since the relation between the scale $\sigma(L)$ of an interest point is linear in the size parameter $L$ of box filters operators, we shall speak indifferently of the former or the latter to indicate the scale.

**Remark** A finer scale-space representation could be obtained (i.e. with sub-pixel values of $L$) using a bilinear interpolation of the image, as suggested in [2]. This is not performed in the proposed implementation.

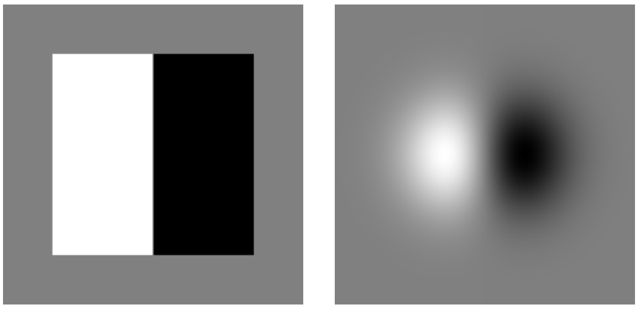
## 3.3 Comparison with Gaussian Derivative Operators

### 3.3.1 First Order Operators

The first order box filters $\mathbf{D}_x^L$ and $\mathbf{D}_y^L$ defined at scale $L$ are approximations of the first derivatives of the Gaussian kernel $G_\sigma$ at the corresponding scale $\sigma(L)$ (see Equation (20)), respectively corresponding to

$$D_x\, G_\sigma(x, y) = -\frac{x}{\sigma^2(L)}\, G_\sigma(x, y) \quad \text{and} \quad D_y\, G_\sigma(x, y).$$

These operators are used for local feature description, detailed in Section 5. Figure 9 compares the first order box filter impulse response with the discretized Gaussian derivative kernel.



$\mathbf{D}_x^L \, \delta$ (Equation (6)) $D_x \, \mathbf{G}_{\sigma(L)}$

Figure 9: Comparison of Gaussian derivative with first order box filter. Illustration of the discrete derivative operator $\mathbf{D}_x^L$ (defined in Section 2.3.1) and discretization of the Gaussian derivative kernel $D_x \, \mathbf{G}_{\sigma(L)}$ when using scale relation $\sigma(L)$ from Equation (20).

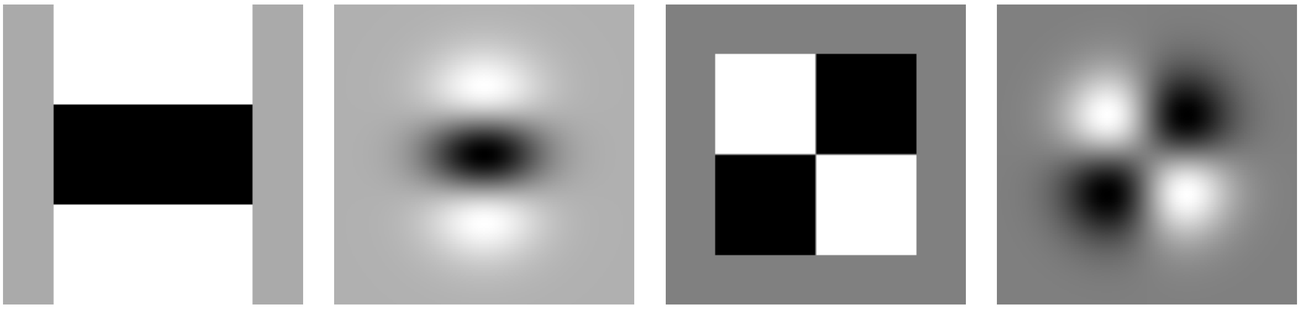


Figure 10: Comparison of second order box filters and second order derivative of Gaussian kernels. (a) operator $\mathbf{D}_{yy}^L$; (b) discretized second order Gaussian derivative $D_y^2 \, \mathbf{G}_\sigma$; (c) operator $\mathbf{D}_{xy}^L$; (d) discretized second order Gaussian derivative $D_{xy} \, \mathbf{G}_\sigma$; For comparison purpose, we used again the scale relation $\sigma(L)$ from Equation (20).

### 3.3.2 The Second Order Operators

Second order differential operators are computed in the scale-space for the detection of interest points [12, 10]. In the linear scale-space representation, this boils down to the convolution with second derivatives of Gaussian kernels

$$D_{xx} \, G_\sigma(x,y) = \frac{1}{\sigma^2}\left(\frac{x^2}{\sigma^2} - 1\right) G_\sigma(x,y), \quad D_{yy} \, G_\sigma, \quad \text{and} \quad D_{xy} \, G_\sigma(x,y) = \frac{xy}{\sigma^4} \, G_\sigma(x,y). \tag{21}$$

In the SURF approach, the convolution with these kernels are approximated by second order box filters, previously introduced respectively as $\mathbf{D}_{xx}^L$, $\mathbf{D}_{yy}^L$ (Equation 10), and $\mathbf{D}_{xy}^L$ (Equation 11). A visual comparison between second order derivatives and their analogous with box filters is shown in Figure 10. These operators are required for the local feature selection step in Section 4.

### 3.3.3 Scale Normalization

According to [13], differential operators have to be normalized when applied in linear scale space in order to achieve scale invariance detection of local features. More precisely, as it can be seen from Equation (21), the amplitude of the continuous second order Gaussian derivative filters decreases with the scale variable $\sigma$ by a factor $\frac{1}{\sigma^2}$.

To balance this effect, second order operators are usually normalized by $\sigma^2$, so that we get for instance

- the scale-normalized determinant of Hessian operator:

$$DoH_\sigma\left(u\right) := \left| \sigma^2 \begin{pmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{pmatrix} \circ u_\sigma \right| = \sigma^4 \left[ D_{xx}\, u_\sigma \cdot D_{yy}\, u_\sigma - \left(D_{xy}\, u_\sigma\right)^2 \right] \; ; \tag{22}$$

- the scale-normalized Laplacian operator:

$$\Delta_\sigma\, u := \sigma^2 \Delta\; u_\sigma = \sigma^2 \Delta\; G_\sigma * u = \sigma^2 (D_{xx} + D_{yy})\, G_\sigma * u = \sigma^2 (D_{xx}\; u_\sigma + D_{yy}\; u_\sigma)\,, \tag{23}$$

  where $\Delta_\sigma\; G_\sigma(x, y) = \sigma^2 (D_{xx} + D_{yy}) \circ G_\sigma(x, y) = \left(\frac{x^2+y^2}{\sigma^2} - 1\right) G_\sigma(x, y)$ is the multi-scale Laplacian of Gaussian. Observe that this operator can be obtained from the trace of the scale-normalized Hessian matrix.

These two operators are widely used in computer vision for feature detection. They are also approximated in SURF, as detailed in the next sections. As a consequence, such a scale-normalization is also required with box filters to achieve similar invariance in SURF. To do so, the authors of SURF proposed that the amplitude of operators $\mathbf{D}^L_{xx}$ (Equation 10), $\mathbf{D}^L_{yy}$ (Equation 10), and $\mathbf{D}^L_{xy}$ (Equation 11) should be reweighted so that the $l^2$ norms of normalized operators become constant over scales.

The quadratic $l^2$ norm of operators are estimated from the squared Frobenius norm of impulse responses

$$\|\mathbf{D}^L_{xx}\|_2^2 := \|\mathbf{D}^L_{xx}\, \delta\|_F^2 = \|\mathbf{D}^L_{yy}\, \delta\|_F^2 = \left(1 + 1 + (-1)^2\right) L(2L - 1) = 6L(2L - 1),$$

so that $\|\mathbf{D}^L_{xx}\|_2^2 \approx 12L^2$ when $L \gg 1$, and

$$\|\mathbf{D}^L_{xy}\|_2^2 := \|\mathbf{D}^L_{xy}\, \delta\|_F^2 = \left(1 + 1 + (-1)^2 + (-1)^2\right) L \times L = 4L^2.$$

This means that box filters responses should be simply divided by the scale parameter $L$ to achieve scale invariance detection.

# 4 Interest Point Detection

In the previous sections, second order operators based on box filters have been introduced. These operators are multi-scale and may be normalized to yield scale invariant response. We will now take interest in their use for multi-scale local feature detection. Once the integral image has been computed, three consecutive steps are performed:

1. **Feature filtering** (Section 4.1) based on a combination of second order box filters;
2. **Feature selection** (Section 4.2) combining non-maxima suppression and thresholding;
3. **Scale-space location refinement** (Section 4.3) using second order interpolation.

This interest point detection task is summarized in Algorithm 1.

**Algorithm 1** Detection of interest points

```
input: image $\mathbf{u}$
output: listKeyPoints
(Initialization)
$\mathbf{U} \leftarrow$ IntegralImage($\mathbf{u}$)    (Equation (1))
(Step 1: filtering of features)
for $L \in \{3,5,7,9,13,17,25,33,49,65\}$ do    (scale sampling, according to Table 2)
    $\mathbf{DoH}^L(\mathbf{u}) \leftarrow$Determinant_of_Hessian $(\mathbf{U}, L)$    (See Equation (24) and Algorithm 2)
end for
(Step 2: selection and refinement of keypoints)
for $o := 1$ to 4  do    (octave sampling)
    for $i := 2$ to 3 do    (levels sampling for maxima location)
        $L \leftarrow 2^o i + 1$    (Equation (19))
        listKeyPoints $\leftarrow$ listKeyPoints + KeyPoints($o, i, \mathbf{DoH}^L(\mathbf{u})$)    (See Section 4.2 and Algorithm 3)
    end for
end for
return listKeyPoints
```

## 4.1 Feature Filtering

The objective is to detect interesting features in the box-space which are highly discriminant, such as corners, junctions (intersection of edges), blobs, *etc.* Taking inspiration from the previous works of Lindeberg [12, 10, 11, 13] about scale-invariant detection, SURF [1, 2] uses scale-normalized determinant of Hessian to detect saddle points in images. Note that other popular methods are the normalized Laplacian detector (Equation (23)), scale invariant Harris corner detector [18], or affine invariant blob detector [17].

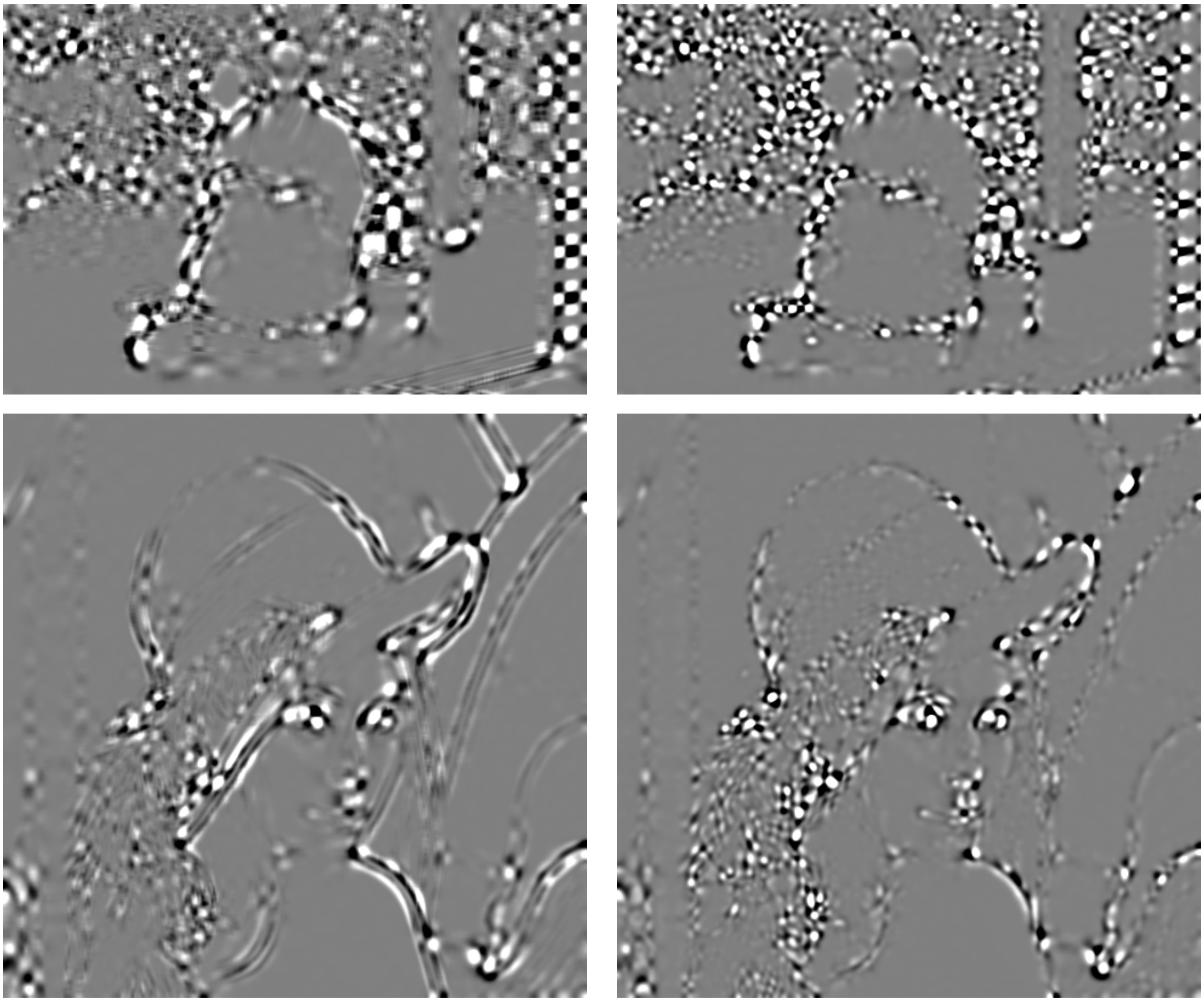
**Determinant of Hessian using box filters** The scale-normalized determinant of Hessian matrix operator proposed in SURF is defined as follows

$$\mathbf{DoH}^L(\mathbf{u}) := \frac{1}{L^4}\left(\mathbf{D}^L_{xx}\,\mathbf{u} \cdot \mathbf{D}^L_{yy}\,\mathbf{u} - \left(w\,\mathbf{D}^L_{xy}\,\mathbf{u}\right)^2\right), \tag{24}$$

using the scaling relation $L = 2^o i + 1$, and a constant weighting factor $w = 0.912$. As mentioned earlier, the normalization factor $\frac{1}{L^4}$ is required to ensure scale invariance. The weighting factor $w$ is used to compensate the numerical approximation of the Hessian determinant by box filters. We will discuss later on these two aspects. An illustration of this multi-scale structure detector is given in Figure 11.

**Computation using integral image** At each point of the box-space, the computation of this operator requires 36 operations: $(8 \times 2 + 15)$ to evaluate box filters, plus 4 multiplications and 1 addition. The method to compute the $\mathbf{DoH}^L(\mathbf{u})$ at a given scale is described in Algorithm 2.

**Sampling** As mentioned before, it is not necessary to evaluate the Hessian determinant for every pixel at large scale. To reduce computation time, the operator response is sampled according to the octave level $o$, such that the number of points tested per octave is $\lceil \frac{M}{2^{o-1}} \rceil \times \lceil \frac{N}{2^{o-1}} \rceil$. In Algorithm 2, this sub-sampling strategy corresponds to a sampling step $p = 2^{o-1}$.



$\mathbf{DoH}^L\,\mathbf{u}$ (Equation (22))

$DoH_\sigma\,\mathbf{u}$ (Equation (23))

Figure 11: Determinant of Hessian Operators applied to the images *japan* (top row) and *lena* (bottom), either using box filters operator $\mathbf{DoH}^L$ at scale $L = 9$ (left column) or Gaussian derivative filter $DoH_\sigma$ at scale $\sigma(L) = 0.4L = 3.6$ (right column).

**Normalization and analogy with continuous determinant of Hessian** Recall that the Hessian determinant based on box filters aims at approximating the $DoH_\sigma$ operator defined in Formula (22). The comparison of their impulse responses is shown in Figure 12. As previously stated, the box filters are not isotropic, which may be source of strong horizontal and vertical artifacts: this can be observed in Figure 11.

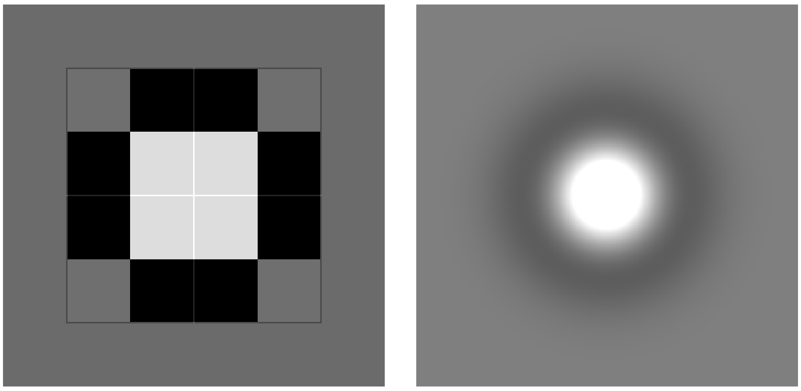
Moreover, recall that to achieve scale invariance, the multi-scale operator response has to be normalized so that the $l^2$ norm remains constant over scales. This is the role of the denominator $L^4$ in (24). Indeed, from Section 3.3.3 we know that $\|\mathbf{D_{xx}}^L\|_2^2 = \|\mathbf{D_{yy}}^L\|_2^2 \approx 12L^2$ and $\|\mathbf{D_{xy}}^L\|_2^2 = 4L^2$ (see Section 3.3.3). On the other hand, we can show that

$$\|D_{xx}G_\sigma\|_2^2 = \frac{3}{16\pi\sigma^6} = 3\|D_{xy}G_\sigma\|_2^2\,. \tag{25}$$

By simple analogy between the two approaches, for instance considering the normalized Gaussian

**Algorithm 2** Feature detection using determinant of Hessian operator

```
input: image $\mathbf{u}$, integral image $\mathbf{U}$, octave $o$, level $i$
output: $\mathbf{DoH}^L(\mathbf{u})$
function Determinant_of_Hessian $(\mathbf{U}, o, i)$
    $L \leftarrow 2^o i + 1$    (Scale variable, Equation (19))
    for $x := 0$ to $M - 1$, step $2^{o-1}$ do    (Loop on columns)
        for $y := 0$ to $N - 1$, step $2^{o-1}$ do    (Loop on rows)
            $\mathbf{DoH}^L(\mathbf{u})(x, y) \leftarrow$ Formula (24) (with (4), (10) and (11))
        end for
    end for
    return $\mathbf{DoH}^L(\mathbf{u})$
end function
```



$\mathbf{DoH}^L\,\delta$ (Equation (24)) $\qquad$ $DoH_\sigma\ \delta$ (Equation (22))

Figure 12: Comparison of the Determinant of Hessian operators, based on box filters and Gaussian filters. (Left) Impulse response of the SURF determinant of Hessian operator $\mathbf{DoH}^L$ (Equation (24)), based on second order box filters at scale $L$; (Right) Discretization of the continuous determinant of Hessian operator $DoH_\sigma$, using scale relation (20). A non-linear mapping has been used for visualization purpose.

derivative, one has

$$\|\sigma^2 D_{xx} G_\sigma\|_2^2 \propto \sigma^4 \times \frac{1}{\sigma^6} = \frac{1}{\sigma^2} \quad \text{and} \quad \|C(L)\mathbf{D}_{xx}^L\|_2^2 \propto C(L)^2 L^2 \propto \frac{1}{L^2} \quad \text{if} \quad C(L) \propto \frac{1}{L^2},$$

so that the scale normalization factor $C(L)$ for second order box filters should be proportional to $\frac{1}{L^2}$.

However, the previous normalization is only true when $L \gg 1$. Indeed, while we have $\frac{\|D_{xx}G_\sigma\|_2^2}{\|D_{xy}G_\sigma\|_2^2} = 3$ at any scale $\sigma$, this is not exactly true with box filters, where:

$$\frac{\|\mathbf{D}_{xx}^L\|_2^2}{\|\mathbf{D}_{xy}^L\|_2^2} = \frac{3(2L-1)}{2L} \approx 3 \text{ when } L \gg 1.$$

To account for this difference in normalization for small scales, while keeping the same (fast) unnormalized box filters, the author of SURF introduced in (24) a weighting factor

$$w(L) = \frac{\|\mathbf{D}_{xx}^L\|_2}{\|\mathbf{D}_{xy}^L\|_2} \cdot \frac{\|D_{xy}G_\sigma\|_2}{\|D_{xx}G_\sigma\|_2} = \sqrt{\frac{2L-1}{2L}}. \tag{26}$$

The numerical values of this parameter are listed in the last column of Table 2. As noticed by the authors of SURF, the variable $w(L)$ does not vary so much across scales. This is the reason why the weighting parameter $w$ in Equation (24) is fixed to $w(3) = 0.9129$.

## 4.2 Feature Selection

A point of interest is a point of the box-space that is approximately *covariant* to any similarity of the initial image $\mathbf{u}$. The term covariant implies here that the surrounding region covariantly changes with the considered class of transformation. To design a descriptor invariant to similarities, the selection of the interest point is jointly performed with the estimation of the similarity transform parameters (location, scale and orientation).

In the SURF methodology, interest points are defined as local maxima of the aforementioned $\mathbf{DoH}^L$ operator applied to the image $\mathbf{u}$. These maxima are detected by considering a $3 \times 3 \times 3$ neighborhood, and performing an exhaustive comparison of every voxel of the discrete box-space with its 26 nearest-neighbors. The corresponding feature selection procedure is described in Algorithm 3.

**Algorithm 3** Selection of features

```
input: $o, i, \mathbf{DoH}^L(\mathbf{u}), t_H$        (Determinant of Hessian response at octave o and level i)
output: listKeyPoints        (List of keypoints in box space with sub-pixel coordinates (x,y,L))
function KeyPoints ($o, i, \mathbf{DoH}^L(\mathbf{u})$)
    $L \leftarrow 2^o i + 1$
    for $x := 0$ to $M - 1$, step $2^{o-1}$ do        (Loop on columns)
        for $y = 0$ to $N - 1$, step $2^{o-1}$ do        (Loop on rows)
            if $\mathbf{DoH}^L(\mathbf{u})(x, y) > t_H$ then        (Thresholding, Equation (27))
                if isMaximum ($\mathbf{DoH}^L(\mathbf{u}), x, y$) then        (Non-maximum suppression)
                    if isRefined ($\mathbf{DoH}^L(\mathbf{u}), x, y, L$) then        (Refinement by interpolation, Section 4.3)
                        addListKeyPoints ($x, y, L$)
                    end if
                end if
            end if
        end for
    end for
    return listKeyPoints
end function
```

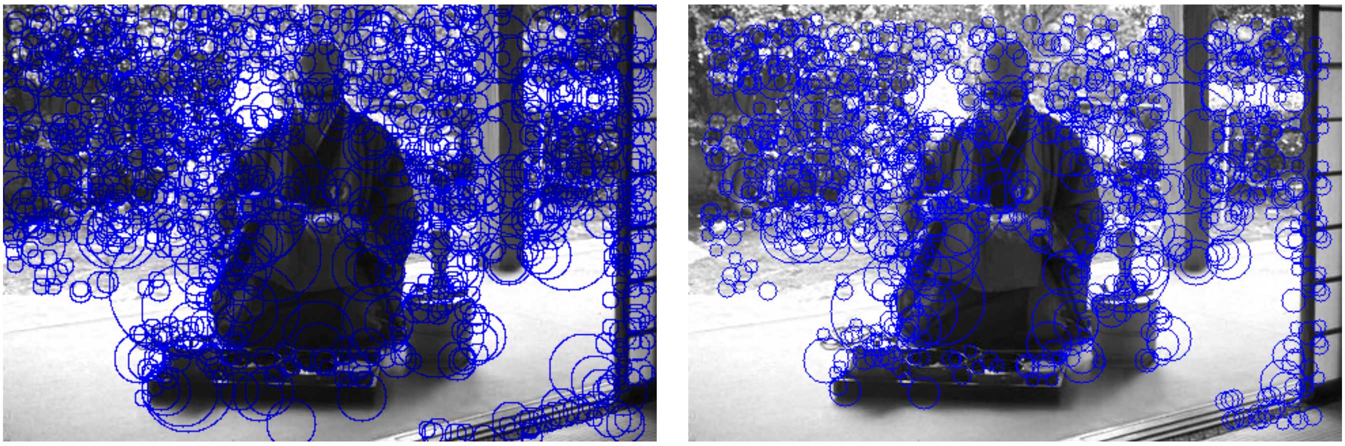
**Remark** A faster method has been proposed in [22] to find the local maxima without exhaustive search, which has been not implemented for the demo.

**Thresholding** Using four octaves and two levels for analysis, eight different scales are therefore analyzed (see Table 2 in Section 3.2). In order to obtain a compact representation of the image -and also to cope with noise perturbation- the algorithm selects the most salient features from this set of local maxima. This is achieved by using a threshold $t_H$ on the response of the $\mathbf{DoH}^L$ operator

$$\mathbf{DoH}^L(\mathbf{u})(x, y) > t_H \,. \tag{27}$$

Note that, since the operator is scale-normalized, the threshold is constant. In the demo, we set $t_H = 10^3$ assuming that the input image $u$ takes values in the interval $[\![0, 255]\!]$. This setting enables us to have a performance similar to the original SURF algorithm [2, 1] (see Section 6 for more details).

Figure 13 shows the set of interest points detected as local box-space maxima of the $\mathbf{DoH}^L$ operator after the thresholding step. For visualization purpose, the radii of the circles is set as 2.5 times the box scale $L$ of the corresponding interest points.



Proposed implementation

OpenCV *2.3.1*

Figure 13: Illustration of interest point detection (without box-space refinement) of the *japan* picture. On the left, the proposed implementation of SURF detects 1184 interest points, while the implementation from OpenCV *2.3.1* selects 919 interest points. The major difference lies in the management of the borders of the image and the threshold value $t_H$ on the $\mathbf{DoH}^L$ operator.

**Comparison with Laplacian operator** As mentioned earlier, one could use the scale normalized Laplacian operator instead of the determinant of Hessian. By analogy to Equation (23), and using the fact that this operator can be defined as the trace of the Hessian matrix, we define the box-Laplacian operator as follows

$$\mathbf{\Delta}^L(\mathbf{u}) := \frac{1}{L^2}\left(\mathbf{D}^L_{xx} + \mathbf{D}^L_{yy}\right)\mathbf{u}\,. \tag{28}$$

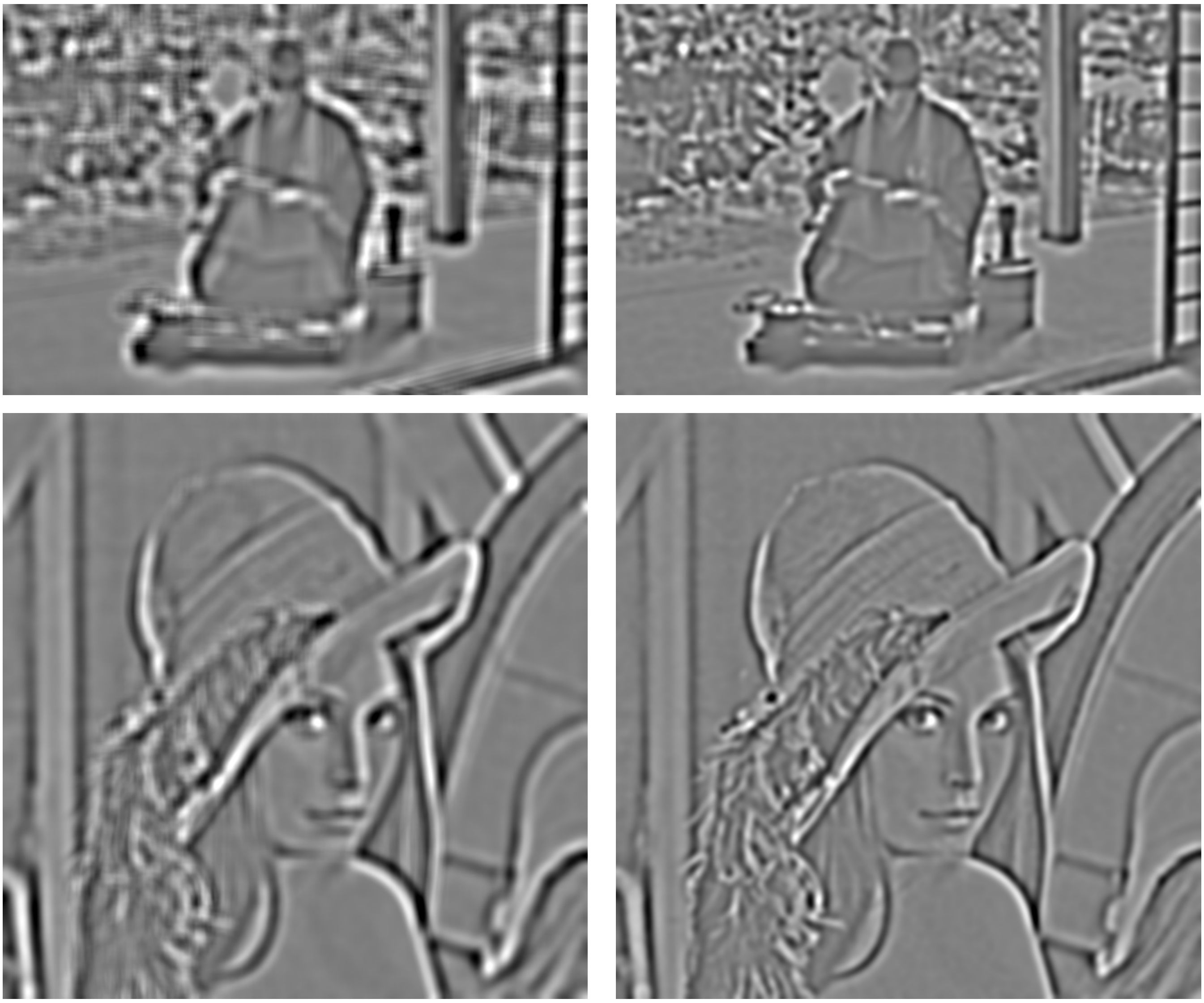
An illustration of this operator is given in Figure 14, along with a comparison with the Laplacian of Gaussian kernel.

We can tell from Figures 11 and 14 that the advantage of the determinant of Hessian over the Laplacian is that it does not respond to edges. In practice, this means that SURF does not require a supplementary geometric criterion to discard interest points located on edges (like for instance in [15]). Indeed, edge features are not discriminant for local image comparison. However, the Laplacian is still used in SURF for local feature description, as explained in the next section.

## 4.3 Scale-Space Location Refinement

**Taylor expansion** For each local maximum of the $\mathbf{DoH}^L$ operator, the localization of the corresponding interest point $X_0$ with coordinates $X_0 : (x_0, y_0, L_0)$ in the box-space may be refined using a quadric fitting. Indeed, considering a $\mathcal{C}^2$ function $f : \mathbb{R}^3 \mapsto \mathbb{R}$, we have the following second order Taylor expansion

$$X = X_0 + \xi, \quad f(X) = f(X_0) + \xi^T \cdot Df(X_0) + \frac{1}{2}\xi^T \cdot D^2 f(X_0) \cdot \xi + O(\|\xi\|^3), \tag{29}$$



$\mathbf{\Delta}^L$ (Equation (28)) $\Delta_\sigma$ (Equation (23))

Figure 14: Trace of Hessian Operators applied to the images *japan* from Figure 6 (top row) and *lena* Figure 1 (bottom), using $\mathbf{\Delta}^L$ from Equation (28) at scale $L = 9$ (left column) and $\Delta_\sigma$ from Equation (23) at scale $\sigma(L) = 0.4L = 3.6$ (right column). In comparison with the determinant of Hessian (see Figure 11), the Laplacian operator yields strong response on edges, thus often requiring a supplementary contour detector to discard interest points located on edges (see for instance [15]).

where $Df$ is the gradient of $f$, and $D^2 f$ is the Hessian matrix. Optimizing $\xi$ to maximize the quadratic expression, we get the following condition

$$0 = Df(X_0) + D^2 f(X_0) \cdot \xi.$$

If the interest point $X_0$ is not degenerate, the Hessian matrix is full rank so that we can solve the corresponding linear system.

As advocated by [1, 2] and formerly proposed in [4], we can transpose this relation to the response of the determinant of Hessian operator $\mathbf{DoH}^L(\mathbf{u})$ in the box space, so that

$$X = X_0 + \xi, \quad \text{where} \quad \xi = \begin{pmatrix} \xi_x & \xi_y & \xi_L \end{pmatrix}^T = -H_0^{-1} d_0. \tag{30}$$

In the previous expression, $L$ is the scale variable, and $d_0$ and $H_0$ are the discrete gradient and the discrete Hessian of $\mathbf{DoH}^L(\mathbf{u})$ at point $X_0 = (x_0, y_0, L_0)$, respectively denoted as follows

$$d_0 = \begin{pmatrix} d_x \\ d_y \\ d_L \end{pmatrix} \qquad H_0 = \begin{pmatrix} H_{xx} & H_{xy} & H_{xL} \\ H_{xy} & H_{yy} & H_{yL} \\ H_{xL} & H_{yL} & H_{LL} \end{pmatrix}.$$

The numerical evaluation of the components of the gradient vector $d_0$ and the symmetric Hessian matrix $H_0$ of $\mathbf{DoH}^{L_0}(\mathbf{u})$ at scale $L_0$ and location $(x_0, y_0)$ is obtained from finite difference schemes in the box space, with a $3 \times 3 \times 3$ centered neighborhood.

**Finite difference scheme for gradient and Hessian matrix estimation** Taking into account that a sub-sampling is performed with a step parameter $p = 2^{o-1}$ which is scale dependent, expressions are quite straightforward for the derivative according to spatial coordinates, with $\forall\, X_0 \in \Omega \times \mathcal{L}$

$$d_x(X_0) = \frac{1}{2p}\big(\mathbf{DoH}^{L_0}(\mathbf{u})(x_0 + p, y_0) - \mathbf{DoH}^{L_0}(\mathbf{u}))(x_0 - p, y_0)\big), \tag{31}$$

$$H_{xx}(X_0) = \frac{1}{p^2}\big(\mathbf{DoH}^{L_0}(\mathbf{u})(x_0 + p, y_0) + \mathbf{DoH}^{L_0}(\mathbf{u})(x_0 - p, y_0) - 2\,\mathbf{DoH}^{L_0}(\mathbf{u})(x_0, y_0)\big), \tag{32}$$

$$\begin{aligned} H_{xy}(X_0) =& \frac{1}{4p^2}\Big[\ \mathbf{DoH}^{L_0}(\mathbf{u})(x_0 + p, y_0 + p) + \mathbf{DoH}^{L_0}(\mathbf{u})(x_0 - p, y_0 - p) \\ & - \mathbf{DoH}^{L_0}(\mathbf{u})(x_0 - p, y_0 + p) - \mathbf{DoH}^{L_0}(\mathbf{u})(x_0 + p, y_0 - p)\ \Big]. \end{aligned} \tag{33}$$

Special care has to be given for the processing of the scale coordinate, since this time the sampling is done (see Equation 19). Using the chain rule, and $\frac{\partial L}{\partial i} = 2^o = 2p$, one thus has

$$d_L(X_0) = \frac{\partial i}{\partial L} d_i(X_0) = \frac{1}{4p}\big(\mathbf{DoH}^{L_0+2p}(\mathbf{u})(x_0, y_0) - \mathbf{DoH}^{L_0-2p}(\mathbf{u}))(x_0, y_0)\big), \tag{34}$$

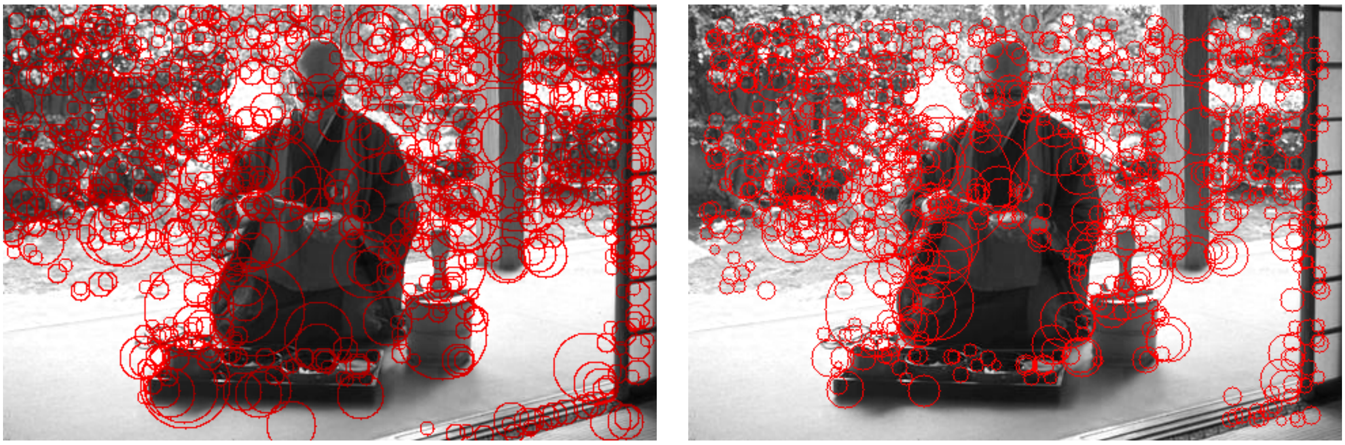
$$\begin{aligned} H_{xL}(X_0) =& \frac{1}{8p^2}\Big[\ \mathbf{DoH}^{L_0+2p}(\mathbf{u})(x_0 + p, y_0) + \mathbf{DoH}^{L_0-2p}(\mathbf{u})(x_0 - p, y_0) \\ & - \mathbf{DoH}^{L_0+2p}(\mathbf{u})(x_0 - p, y_0) - \mathbf{DoH}^{L_0-2p}(\mathbf{u})(x_0 + p, y_0)\ \Big], \end{aligned} \tag{35}$$

$$H_{LL}(X_0) = \frac{1}{4p^2}\big(\mathbf{DoH}^{L_0+2p}(\mathbf{u})(x_0, y_0) + \mathbf{DoH}^{L_0-2p}(\mathbf{u}))(x_0, y_0) - 2\,\mathbf{DoH}^{L_0}(\mathbf{u})(x_0, y_0)\big). \tag{36}$$

**Refinement rejection** It is possible that the refined point $X = X_0 + \xi$ does not belong to the neighborhood of $X$, i.e. $\max(|\xi_x|, |\xi_y|, \frac{1}{2}|\xi_L|) > p$. This happens if the Hessian matrix is ill conditioned. In practice, those interest points are not reliable for matching and are thus discarded, as proposed in [4]. The location refinement procedure is described in Algorithm 4 and is illustrated in Figure 15.

**Algorithm 4** Box-space location refinement

```
input: $X_0 : (x_0, y_0, L_0)$ and $\mathbf{DoH}^L(\mathbf{u})$    (Refinement in box-space of detected point $X_0$ at scale $L_0 = 2^o i + 1$)
output: True/False & $X : (x, y, L)$    (Is the interest point kept ? If so, return the position of the refined detected point.)
function isRefined ($\mathbf{DoH}^L(\mathbf{u}), X_0$)
    $p \leftarrow 2^{o-1}$    (Step parameter for finite different scheme)
    $H_0 \leftarrow$ Formulas (32), (33), (35) and (36).    (Hessian matrix)
    $d_0 \leftarrow$ Formulas (31) and (34).    (Gradient vector)
    $\xi \leftarrow -{H_0}^{-1} \cdot d_0$    (Maximum refinement, see Equation (30))
    if $\max(|\xi_x|, |\xi_y|, \frac{1}{2}|\xi_L|) < p$ then    (Check precision improvement)
        $(x, y, L) \leftarrow (x_0, y_0, L_0) + \xi$    (Refinement using 2nd order Taylor expansion, see Equation (29))
        return True, $X : (x, y, L)$
    else
        return False
    end if
end function
```



Proposed implementation

OpenCV *2.3.1*

Figure 15: Selected features after scale-space location refinement. (Left) the proposed implementation of SURF retains 946 interest points (among 1184); (Right) the SURF implementation from OpenCV *2.3.1* selects 874 interest points from 919.

# 5 Interest Point Description

From the previous stage, we have obtained a set of $P$ interest points in the box-space

$$\Big\{ X_k : (x_k, y_k, L_k) \in [0, M-1] \times [0, N-1] \times [0, 65] \Big\}_{k=1,\dots,P}$$

that corresponds to the most salient features of the input image. Observe that, because of the refinement step, scale-space coordinates are continuous. In order to compare and find correspondences between interest points from different images, a local descriptor of their neighborhood is encoded for each of them. The goal is to obtain a geometric invariant representation, which is also robust to various perturbations such as noise, illumination or contrast change.

The scale space analysis already offers scale and translation invariance. To achieve similarity invariance, one also needs rotational invariance. One way to achieve this consists in extracting for each interest point a dominant orientation following the procedure detailed in Section 5.2. Then, SURF features are built from the normalized gradient distribution in the vicinity of interest points (Section 5.3). Finally, a method to compare SURF descriptors and find local correspondences between images is exposed in Section 5.4.

## 5.1 Scale of an Interest Point

In the previous section, for the sake of simplicity, we have referred to $L$ as the scale parameter of the box-space. Let us now recall that the analogous relation between the box-filter size parameter $L$ and the equivalent scale variable $\sigma$ from linear scale space is given by Equation (20). Thus, for a given detected feature point $X_k$, one has the corresponding scale variable

$$\sigma_k = \lfloor 0.4\, L_k \rceil. \tag{37}$$

This scale variable is used thoroughly in the following sections to define various scale normalizations involved in SURF descriptors. Note that the rounding operator $\lfloor \cdot \rceil$ is only required for numerical convenience.

## 5.2 Dominant Orientation of an Interest Point

Alike the SIFT method, the main orientations of SURF keypoints are computed from the local distribution of the gradient orientation.

**Weighted gradient computation** For each interest point $X_k$, we first consider the neighborhood $\mathcal{B}_{6\,\sigma_k}(x_k, y_k)$ defined as the disk of radius $6\sigma_k$ with center $(x_k, y_k)$. The computation of the gradient at this scale $\sigma_k$, and in this neighborhood $\mathcal{B}_{6\,\sigma_k}(x_k, y_k)$ is obtained by convolution with first order box filters (see the corresponding definition of $\mathbf{D}_x^{L_k}$ (Equation (6)) and $\mathbf{D}_y^{L_k}$ (Equation (7))). To reduce the impact of remote pixels, the gradient samples are weighted according to their distance from the interest point, using a discrete Gaussian kernel (Equation (16)), with a standard deviation equal to $2\sigma_k$. The weighted gradient at point $(x, y)$ then writes ($x$ and $y$ being integer coordinates of the pixel grid $\Omega$)

$$\forall (x,y) \in \Omega \cap \mathcal{B}_{6\sigma_k}(x_k, y_k), \qquad \phi_k(x,y) := \begin{pmatrix} \mathbf{D}_x^{L_k} \\ \\ \mathbf{D}_y^{L_k} \end{pmatrix} \circ \mathbf{u}(x,y) \cdot \mathbf{G}_1\left( \frac{x - x_k}{2\sigma_k}, \frac{y - y_k}{2\sigma_k} \right). \tag{38}$$

**Orientation score function** Unlike the SIFT approach in which a histogram is built from gradient samples to estimate the dominant orientation [15], SURF computes the following score vector $\Phi$ according to the orientation $\theta$

$$\Phi_k(\theta) = \sum_{(x,y) \in \mathcal{B}_{6\sigma_k}(x_k, y_k)} \phi_k(x,y) \times \mathbf{1}_{[\theta - \frac{\pi}{6}, \theta + \frac{\pi}{6}]} \left( \angle\, \phi_k(x,y) \right). \tag{39}$$

This vector sums all the weighted gradients in the considered neighborhood which have approximately the same orientation $\theta$ (with a fixed tolerance of $\pm\frac{\pi}{6}$). Here, $\angle\,\phi \in [-\pi, \pi)$ denotes the angle between vector $\phi \in \mathbb{R}^2$ and canonical vector $(1,0)^T$ [2]. The orientation score function is defined as the $l^2$ norm of this aggregated vector $\|\Phi_k(\theta)\|$.

[2] This corresponds to the C++ standard math library function atan2.

**Dominant orientation** Finally, the orientation of the interest point $X_k$ is defined as the global maximum of the orientation score function

$$\theta_k = \angle\,\Phi_k(\theta^\star) \qquad \text{where} \qquad \theta^\star \in \operatorname*{argmax}_{\theta\in\Theta} \|\Phi_k(\theta)\|. \tag{40}$$

In the proposed demo, we evaluate the orientation score function for 40 different angles, uniformly sampled over the unit circle: $\Theta = \left\{\frac{k\pi}{20}, k = 0, 1, \ldots, 39\right\}$.

An example of dominant orientation estimation is given in Figure 16, in which the estimated orientations are represented by segments. Algorithm 5 summarizes the computation of the main orientation of an interest point.

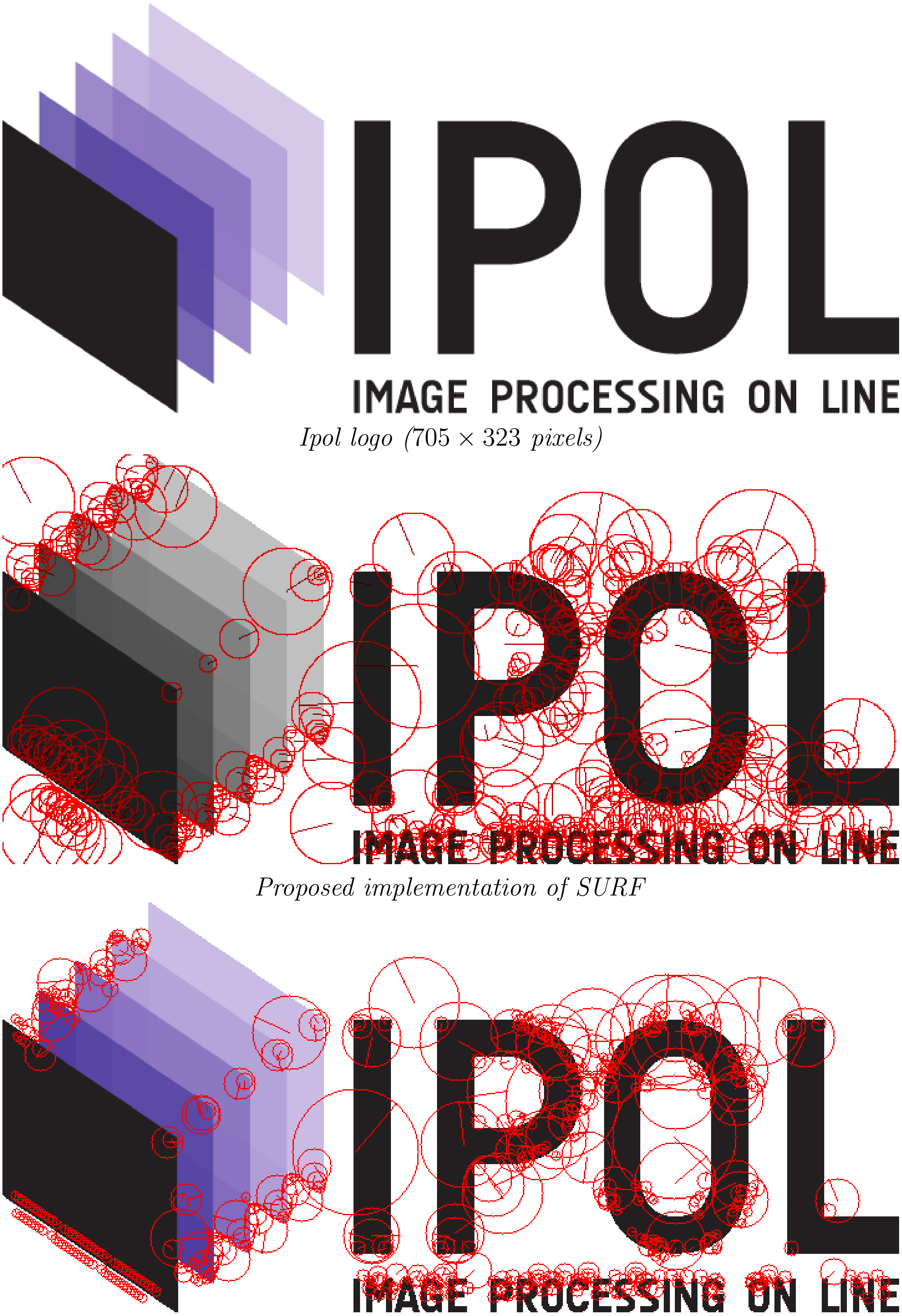
**Algorithm 5** Computation of the orientation

```
input: $\mathbf{u}, X : (x, y, L)$
output: $\theta$    (Main orientation of the interest point neighborhood)
function Orientation $(\mathbf{u}, x, y, L)$
    $\sigma \leftarrow \lceil 0.4L \rfloor$    (Scale variable definition from Equation (37))
    for $i := -6$ to 6 do    (Span the keypoint neighborhood)
        for $j := -6$ to 6 do
            if $i^2 + j^2 \leq 36$ then    (Check that $(x + i\sigma, y + j\sigma) \in \mathcal{B}_{6\sigma}(x, y)$)
                $\phi_{(i,j)} = \left(\mathbf{D}_x^L, \mathbf{D}_y^L\right)^T \circ \mathbf{u}(x+i\sigma, y+j\sigma) \times \mathbf{G}_1\left(\frac{i}{2}, \frac{j}{2}\right)$    (Using Equation (43), Equation (6) and (7))
            end if
        end for
    end for
    for $k := 0$ to 39 do    (Span the discrete set of tested angular sector $\Theta$)
        $\theta_k \leftarrow \frac{k\pi}{20}$    ($\Theta = \{\theta_k\}_k$)
        $\Phi(\theta_k) \leftarrow$ Formula (39)    (Compute orientation score function for angular sector $\theta_k$)
    end for
    $\theta^\star = \operatorname*{argmax}_{\theta\in\Theta} \|\Psi(\theta)\|$ according to Formula (40)
    return $\theta = \angle\,\Phi(\theta^\star)$
end function
```

**Sampling strategies** Once again, as previously done for interest point detection, a sub-sampling strategy is used to speed up the SURF descriptor construction. Indeed, the weighted gradient operator is not computed in practice for all the pixels from the neighborhood, which would be computationally prohibitive for large scales. Instead, pixel locations are sampled with a step parameter equal to the scale of the interest point $\sigma_k$, as explicitly detailed in Algorithm 5. Another advantage of this strategy is that the number of gradient samples used to estimate the main orientation is consistent throughout the scales (approximately 100 samples in our implementation).

### 5.3 SURF Descriptors

A SURF descriptor is a $16 \times 4$ vector, representing normalized gradient statistics (mean and absolute mean values) extracted from a spatial grid $\mathcal{R}$ divided into 4-by-4 regions. These subregions are referred to as $\mathcal{R} = \{\mathcal{R}_{i,j}\}_{1\leq i,j\leq 4}$. For a given oriented interest point $X_k : (x_k, y_k, L_k, \theta_k)$, as illustrated in Figure 17, the corresponding square grid is centered at $(x_k, y_k)$, aligned according to $\theta_k$ and scaled to have a normalized width of $20\sigma_k$, using relation (37).



*Ipol logo* (705 × 323 *pixels)*

*Proposed implementation of SURF*

*SURF algorithm from OpenCV 2.3.1*

Figure 16: Illustration of interest points detection with dominant orientation. The radius of the circle indicates the scale of the interest point and the segment shows the selected orientation based on Algorithm 5. (Middle) 456 points are detected with the proposed implementation of SURF. (Bottom) 478 interest point are selected with the SURF implementation of OpenCV *2.3.1*.

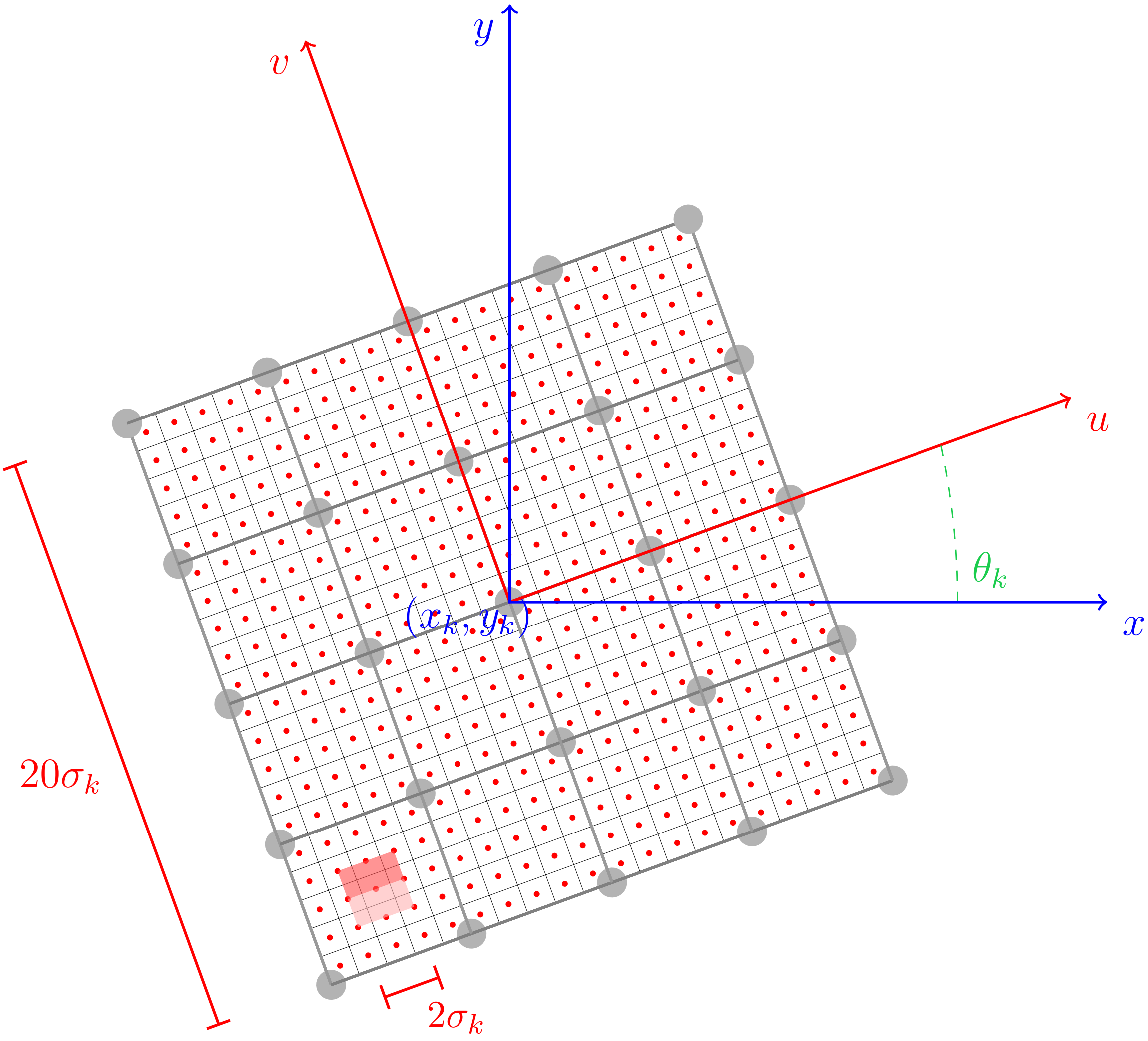


Figure 17: Illustration of the scaled and orientated grid $\mathcal{R}$. The grid $\mathcal{R}$, divided into 16 subregions, is used to build the SURF descriptor in the neighborhood of an interest point $X_k : (x_k, y_k, L_k)$ with orientation $\theta_k$.

**Scale normalized sampling** A subregion $\mathcal{R}_{i,j}$ is a square with side length equal to $5\sigma_k$. Gradient responses are regularly sampled, using a step equal to $\sigma_k$. Hence, 25 gradients are consistently used in each subregion to build the SURF descriptor for any interest point.

**Change of coordinates** The coordinates of gradient samples according to the descriptor grid are denoted by $(u, v)$ whereas the coordinates on the pixel grid $\Omega$ are denoted $(x, y)$. The similarity transform between the two coordinate systems is then

$$S_k : \begin{pmatrix} u \\ v \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \end{pmatrix} = \sigma_k \cdot R_{\theta_k} \begin{pmatrix} u \\ v \end{pmatrix} + \begin{pmatrix} x_k \\ y_k \end{pmatrix} , \qquad \text{where } R_\alpha = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}. \tag{41}$$

To span the grid $\mathcal{R}$, one must take $u, v \in [\![-10, 9]\!] + \frac{1}{2}$ (where the term $\frac{1}{2}$ is used to ensure symmetry). Since the coordinates $(x, y)$ do not correspond exactly to pixel values, rounding is required (nearest neighbor interpolation)

$$(\mathbf{x}, \mathbf{y}) = \lfloor S_k(u, v) \rceil = (\lfloor x \rceil, \lfloor y \rceil) \in \Omega. \tag{42}$$

Note also that the coordinates of $(u, v)$ implicitly depend on the subregion coordinates $\mathcal{R}_{i,j}$ (see Algorithm 6 for more details).

**Gradient normalization** The gradient components are computed by convolution with first order box filters $\mathbf{D}_x^{L_k}$ and $\mathbf{D}_y^{L_k}$ (see Formulas (6) and (7)). To avoid the use of interpolation due to the orientation of the descriptor grid $\mathcal{R}$, the gradients responses are first computed on the regular grid $\Omega$, and then rotated using rotation matrix $R_{-\theta_k}$ to cancel the descriptor orientation.

In addition, as previously done for the dominant orientation selection, the gradient responses are weighted according to the distance of the pixel from the interest point. This processing aims at reducing the importance of distant pixels which are more sensitive to orientation or scale perturbation. In SURF [2], the weights are defined by a discrete Gaussian kernel (Equation (16)), with a standard deviation equal to $3.3\,\sigma_k$.

Eventually, the weighted gradient at point $(u, v)$ reads

$$\forall\, (u, v) \in \mathcal{R}, \qquad \begin{pmatrix} d_x(u, v) \\ d_y(u, v) \end{pmatrix} := R_{-\theta_k} \begin{pmatrix} \mathbf{D}_x^{L_k} \\ \mathbf{D}_y^{L_k} \end{pmatrix} \mathbf{u}\,(\mathbf{x}, \mathbf{y}) \times \mathbf{G}_1 \left( \frac{u}{3.3}, \frac{v}{3.3} \right) . \tag{43}$$

**Gradient statistics** Similarly to SIFT, the SURF descriptor aggregates local gradient information from subregions. However, whereas SIFT builds small histograms of gradient orientations (weighted by gradient magnitude), SURF computes first order statistics on vertical and horizontal gradient responses. The authors of SURF [2] claim that using the sum and the sum of absolute values yields the best compromise between compactness and efficiency. The resulting statistical vector describing subregion $\mathcal{R}_{i,j}$ corresponds to

$$\forall\, (i, j) \in [\![1, 4]\!]^2 , \qquad \mu_k(i, j) = \begin{pmatrix} \sum_{(u,v)\in\mathcal{R}_{i,j}} \; d_x(u, v) \\ \sum_{(u,v)\in\mathcal{R}_{i,j}} \; d_y(u, v) \\ \sum_{(u,v)\in\mathcal{R}_{i,j}} \; |\, d_x(u, v) \,| \\ \sum_{(u,v)\in\mathcal{R}_{i,j}} \; |\, d_y(u, v) \,| \end{pmatrix} . \tag{44}$$

The SURF descriptor associated to the interest point $X_k : (x_k, y_k, L_k, \theta_k)$ is then simply obtained by concatenating the 16 vectors $\mu_k(i,j)$ computed for every subregion

$$\mu_k = \Big(\mu_k(i,j)\Big)_{1 \le i,j \le 4}. \tag{45}$$

**Descriptor normalization** The previous geometric normalization guaranties invariance to similarity transforms. In practice, the use of local gradient information combined with a localization grid $\mathcal{R}$ yields also robustness to small changes of viewpoint. In order to gain invariance to linear contrast changes, the SURF descriptor is normalized to a unit vector, using Euclidean norm

$$\textsc{surf}(X_k) = \frac{\mu_k}{\|\mu_k\|_2}. \tag{46}$$

Besides, because of this $l^2$ normalization, it is unnecessary to use scale-normalized first order box filters.

The construction of such descriptor is summarized in Algorithm 6.

**Sign of the Laplacian** Eventually, for each interest point $X_k$, in addition to scale-space coordinates $(x_k, y_k, L_k)$ and orientation $\theta_k$, the local sign of the Laplacian is also stored. The approximate Laplacian operator response is obtained from $\mathbf{\Delta}^{L_k}\mathbf{u}(x_k, y_k)$, using Formula (28). The goal is to accelerate the comparison of of SURF interest points during the matching step (see Section (5.4) for more details). The sign of the Laplacian response provides some information about the local contrast of the interest point. Since this operator is the trace of the readily available Hessian matrix, this operator only requires one additional operation.

**Algorithm 6** Construction of SURF descriptor

```
input: Input image $\mathbf{u}$, keypoint $X : (x, y, L)$
output: SURF Descriptor, orientation of keypoint and sign of Laplacian
function buildDescriptor $(\mathbf{u}, x, y, L)$
    $\theta \leftarrow$ Orientation $(\mathbf{u}, x, y, L)$    (See Algorithm 5)
    for $i := 1$ to $4$ do    (Span the 4-by-4 subregions $\mathcal{R}_{i,j}$ of the oriented grid)
        for $j := 1$ to $4$ do
            for $u := -9.5$ to $9.5$ step $1$ do    (Span the subregions $\mathcal{R}_{i,j}$ coordinates $(u, v)$)
                for $v := -9.5$ to $9.5$ step $1$ do
                    $(x, y) = S(u, v)$    (Change of coordinates, see Formula (41))
                    $(\mathbf{x}, \mathbf{y}) = \lfloor (x, y) \rceil$    (Nearest neighbor pixel, see Formula (42))
                    $\mathbf{D}_x^{L_k}\mathbf{u}(\mathbf{x}, \mathbf{y})$, $\mathbf{D}_y^{L_k}\mathbf{u}(\mathbf{x}, \mathbf{y})$    (Gradient response)
                    $(d_x(u, v), d_y(u, v)) \leftarrow$ Formula (43)    (Gradient correction)
                end for
            end for
            $\mu(i, j) \leftarrow$ Formula (44)    (Compute subregion statistical descriptor $\in \mathbb{R}^4$)
        end for
    end for
    $\mu \leftarrow$ concatenate $(\{\mu_k(i, j)\}_{1 \le i,j \le 4})$    ($16 \times 4$ dimensional vector (Equation (45)))
    $\textsc{surf}(X) \leftarrow$ normalize $(\mu)$    ($l^2$ normalization (Equation (46)))
    $\mathbf{\Delta}^L(\mathbf{u})(x, y) \leftarrow$ Formula (28)    (Laplacian at scale $L$)
    return $\textsc{surf}(X)$, $\theta$, Sign$(\mathbf{\Delta}^L(\mathbf{u})(x, y))$
end function
```

## 5.4 Feature Matching

From the previous steps, a pair of images $(\mathbf{u}, \mathbf{v})$ to be matched is represented by two sets of interest points ($\{X_k\}_k$ and $\{Y_l\}_l$) with their corresponding SURF descriptors ($\{\textsc{surf}(X_k)\}_k$ and $\{\textsc{surf}(Y_l)\}_l$). Recall that a SURF descriptor $X \in [-1, 1]^{64}$ is a 64-dimensional vector with unitary Euclidean norm: $\|X\| = 1$. The matching step is simply performed here as an exhaustive comparison of these vectors using Euclidean distance, combined with a Nearest-Neighbor Distance Ratio (NN-DR) thresholding technique.

**Feature comparison** For any query descriptor $X_k$ from the first image $\mathbf{u}$, we compute the Euclidean distance $d_{k,l}^2 = \|X_k - Y_l\|_2^2 = 2(1 - X_k^T \cdot Y_l)$ with all the candidate features $Y_l$ from the second image $\mathbf{v}$. Since around a thousand of SURF features are approximately extracted from a 1 megapixel image, millions of scalar products are therefore computed.

To speed up the matching procedure, we first compare the signs of the Laplacian of SURF features as advocated by the authors of SURF. If the signs of two descriptors are different, they are very unlikely similar: the distance is therefore not computed and the potential match is discarded.

Besides, note that many other methods exist in the literature to accelerate the comparison of vectors, either for exact or approximate Nearest-Neighbor search, based on data structure (such as Kd-tree, Vector quantization, etc.). We did not use such techniques in the proposed source code.

**Matching criterion** Subsequently, one needs to extract the most significant correspondences from within typically several millions of putative ones. In other words, taking only the similarity distances $d_{k,l}$ into account, one has to separate the correct matches from the spurious ones. A classic paradigm is the *nearest neighbor matching*, which consists in testing only correspondences between each query descriptor $X_k$ with its most similar candidate, denoted by $Y_{l_1}$ where

$$l_1 \in \underset{l}{argmin} \;\; d_{k,l} \;.$$

The Nearest Neighbor Distance Threshold (NN-DT) is a straightforward technique to validate such a putative correspondence based on a fixed threshold on the similarity measure. Inescapably, such a matching criterion implies a compromise between the number of false positives and false negatives.

As done in SURF [2], we use instead the NN-DR matching criterion, a simple statistical method proposed by Lowe [15] to discard mismatches, which has been shown (see e.g. [23]) to outperform NN-DT. This method considers the ratio between the first and second nearest neighbors to measure the quality of a correspondence between the query and the most similar candidate. Thus, we first compute the closest $Y_{l_1}$ and second closest $Y_{l_2}$ candidates from the query feature $X_k$, such that

$$l_2 \in \underset{l, \text{ s.t. } d_{k,l} \geq d_{k,l_1}}{argmin} \quad d_{k,l}.$$

and then compare the corresponding ratio of distances to a fixed threshold $t$, so that

$$\text{if} \quad \frac{d_{k,l_1}}{d_{k,l_2}} \leq t \qquad \Leftrightarrow \qquad \text{correspondence} \quad (X_k, Y_{l_1}) \quad \text{is validated.}$$

Following [15], in our experiments, the threshold was set to $t = 0.8$.

**Geometric consistency** Finding local correspondences between two images is rarely an end in itself. In practice, one wants for instance to find the geometric deformation between the two images (image registration) or decide if they share a similar object (object detection and recognition). Moreover, the previous matching criterion is generally not sufficient to discard most mismatches. Thus, in the demonstration, we decided to incorporate an optional and additional matching criterion based on the epipolar geometry consistency. To do so, we use the ORSA algorithm [21], which combines the RANSAC [6] algorithm with hypothesis testing.

Without going into details (see for instance [20]) this supplementary criterion consists in finding the most significant geometric relation between the two images to remove correspondences that are not consistent with it. More specifically, we consider the two images as two different views of the same scene (stereo-vision). The epipolar geometry enables to account for the perspective deformation between the coordinates of corresponding interest points in the two images. Random subsets of correspondences are iteratively drawn and used to estimate the related fundamental matrix. Given a tolerance error, the matches that are consistent with the geometry are counted to select the best geometric transformation between the two images.

This supplementary criterion is also given in the proposed source code and is used in the following section for visualization purpose

# 6 Experiments

In this section, we illustrate the performance of the SURF algorithm for image matching. We first evaluate the validity of the proposed implementation by checking the SURF properties: invariance to similarity transform and robustness to small viewpoint changes. Secondly, some examples of image pair matching are shown to illustrate the behavior (both success and failure) of the SURF method, along with a comparison with some other competitive methods.

## 6.1 Performance Evaluation

An executable program has been released by the authors of SURF [3]. Unfortunately, we were only able to use it partially. So, in order to evaluate and validate our implementation of the SURF algorithm, some of the benchmarks proposed in [2] have been reproduced.

In these benchmarks, performance curves are obtained from the experimental protocol from [19] which provides several image sequences with geometric distortion (along with the ground-truth) and the MATLAB code to compute scores. Two different types of performance measures are evaluated to assess the quality of the feature detection and description: the *repeatability score* and the *precision-recall rate.*

**Repeatability** The repeatability score represents the proportion of nearly identical features detected in two different views of the same scene.

Let $u$ be the image of interest, and $u'$ be the same reference scene from a different viewpoint, so that the transform $T$ between the two view is known (in practice, a homography combined with a mask). For each point $X_k$ selected in $u$, an ideal detector should find the same interest point $X'_k$ in $u'$, such that $X_k = T(X'_k)$. The repeatability test consists in measuring how often this relation is verified. In practice, due to numerical approximations (blur, noise, quantization, errors, *etc*) that result in $u \approx T(u')$, we have to test for each $X_k$ if there is some detected point $X'_i$ in $u'$ which is close enough. To this end, and following [19], we consider the support region $\mathcal{R}_k$ of each keypoint $X_k$, here defined as a disk $\mathcal{B}_s(x_k, y_k)$ centered on the point location and which radius $s$ is fixed (30 pixels in [19]). Two keypoints $X_k$ and $X'_i$ are considered similar if the overlap error between the two registered regions is small enough. We denote $\mathcal{R}'_i$ the support region of the interest point $X'_i$ back-projected in $u$, which is an elliptic region obtained from the geometric transform $T$ of the disk

[3] available at the following address: `http://www.vision.ee.ethz.ch/~surf/`.

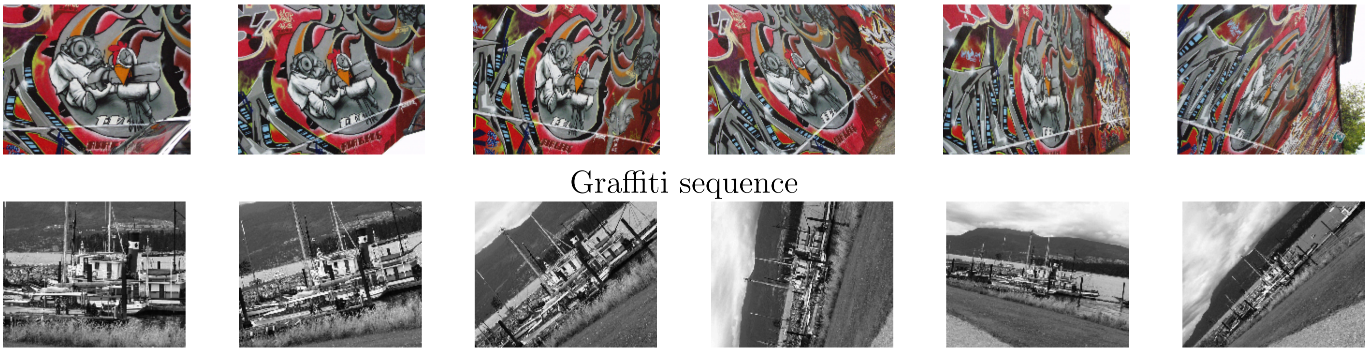
$\mathcal{B}_{s'_i}(x'_i, y'_i)$. As advocated in [19], the radius $s'_i$ verifies $\frac{s'_i}{\sigma'_i} = \frac{s}{\sigma_k}$ to take into account the difference of scale between the two interest-points. The overlap score is then defined as

$$\textit{overlap\_score}\ (X_k, X'_i) = \frac{\mathcal{R}_k \cap \mathcal{R}'_i}{\mathcal{R}_k \cup \mathcal{R}'_i}.$$

A minimum of 50 % overlap score is required to consider that a keypoint $X_k$ in $u$ is repeated in $u'$. Simply speaking, the repeatability score is the proportion of keypoints in $u$ that have a similar keypoint in $u'$. For more details, please refer to [19, Section 4.1].

We have used two image sequences displayed in Figure 18: the ‘Graffiti’ database, where the same (planar) scene is captured under different view angles (from 0° to 60°), and the ‘Boat’ sequence, with rotations and zoom-out (up to a scale change factor of 3). The repeatability score of the SURF interest point detector is hence evaluated depending on the view angle and the scale factor. The corresponding projective transform (homography) between the pictures is known, and used to define correct local correspondences for every pair of images. Note that it is impossible to achieve 100% repeatability score with scale change, since features which are detected at the smallest scale in a given image cannot be detected in the zoom-out image.

Results are displayed in Figure 19 and are very similar to the original SURF (see Figures 16 and 17 in [2]). Observe that for very small perspective or rotation distortions, the repeatability score is very high, while it progressively decreases for larger distortions.



Graffiti sequence

Boat sequence

Figure 18: Database from [19] for performance evaluation. (Top) The ‘graffiti’ image sequence. (Bottom) The ‘boat’ image sequence.

**Precision-Recall** As described in Section 5.4, one uses in practice the NN-DR (Nearest Neighbor Distance Ratio) criterion to select correct matches from all the putative correspondences between the most similar SURF descriptors from a pair of images. This criterion relies on a threshold $t \in [0, 1]$ (which is fixed to 0.8 the rest of this experimental section), such that every tentative correspondence is validated when fixing $t = 1$. For a given value of $t$ in $[0, 1]$, the *precision* and *recall* rates are respectively defined as the proportion of correct matches among the set of selected matches (after nearest neighbor selection), and the proportion of correct matches among the total number of correct matches (i.e. including non-nearest neighbor matches). This boils down to the following rates

$$\text{recall}(t) = \frac{\#\text{correct NN matches}(t)}{\#\text{correct matches}(1)} \in [0, 1], \quad \text{precision}(t) = \frac{\#\text{correct NN matches}(t)}{\#\text{NN matches}(t)} \in [0, 1]. \tag{47}$$

The larger $t$, the better the recall rate. On the other hand, the precision is generally equal to one for small values of $t$ and roughly decreases for larger values of $t$. The ‘recall vs. precision’ curve (a.k.a.

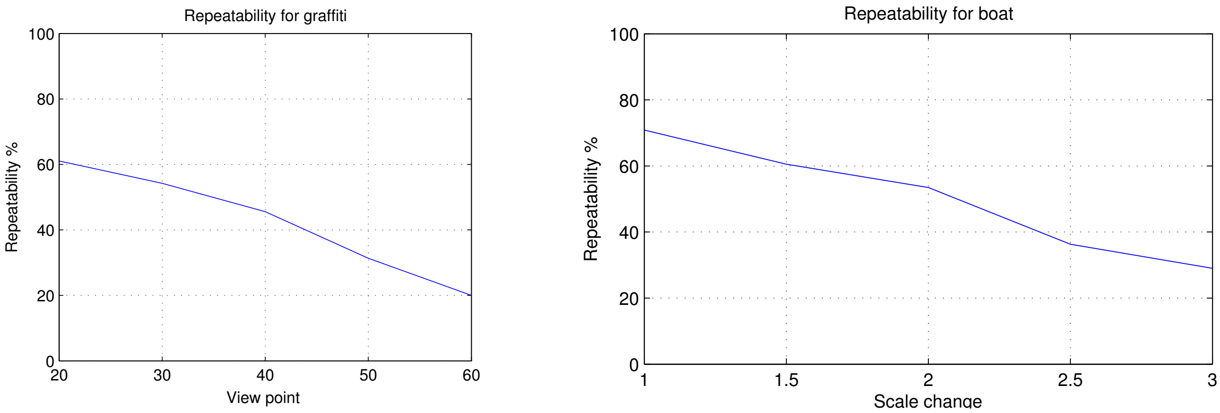


Figure 19: Evaluation of repeatability detection on the datasets displayed in Figure 18. (Left) using the 'graffiti' image database, and (Right) using the 'boat' image database.

ROC curve) is hence built to illustrate this trade-off between recall and precision rate when tuning the parameter $t$.

The precision-recall curve for the 'Boat' database is displayed in Figure 20. More precisely, we used the images #1 and #3 of the sequence displayed in Figure 18. Let us observe that the recall rate is in practice strictly lower than 1 since we only consider nearest neighbor matches. When $t = 1$, the recall rate indicates the proportion of matches that are correct, which is here around 50%. In comparison, using random assignment between $N = 10^3$ interest points would give in average a recall rate of approximately $\frac{1}{N} = 0.1\%$.

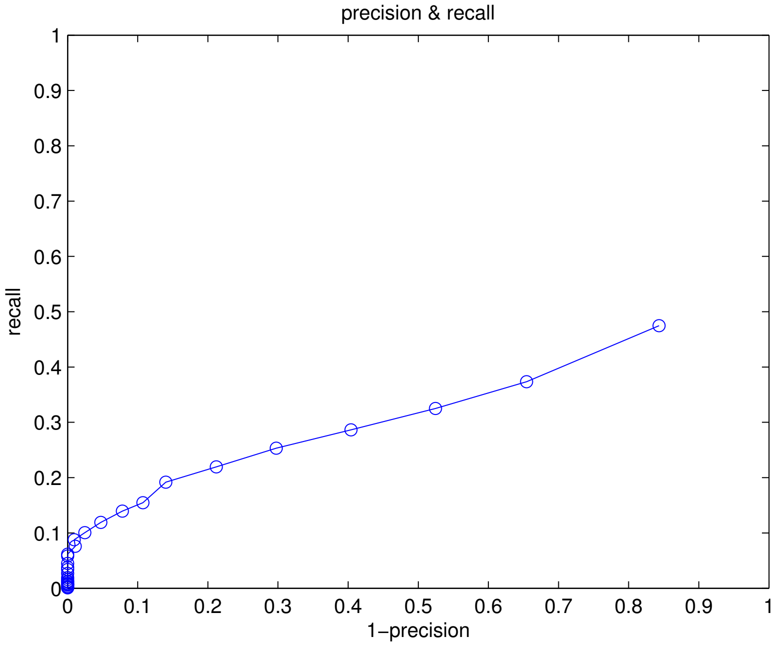


Figure 20: Precision-recall curve for the 'Boat' image pair (using image #1 and #3).

## 6.2 Comparison with SIFT Approach

As previously mentioned, the SURF method shares a lot of common features with the SIFT approach [15]. For a complete, in-depth analysis, we refer the reader to [24]. Some of the theoretical aspects of multi-scale analysis have been already discussed in this Section 3. Let us recall that SIFT detects features using multi-scale Gaussian derivatives and then builds similarity invariant descriptors based on local histograms of gradient's orientation. On the other hand, SURF uses fast second order box filters for interest point detection and then computes first order gradient statistics to encode local information.

In this section, we propose a small visual comparison of the matching results of these two methods. The SIFT correspondences are obtained using the executable program of David Lowe [4]. In order to visually illustrate the suitability of the two approaches, a geometric consistency post-processing is used (Optimized Random Sampling Algorithm from [21]) to discard a large amount of incompatible correspondences. Results are displayed in Figures 21 and 22. The first image is stacked above the second one, and the color lines indicates the correspondences between interest points. We can see that both methods provide matching results that are located in similar regions (contrasted textures, corners and blobs). However, it is noticeable that the SIFT method gives more dense correspondences. In the meantime, it should be mentioned that SURF is faster to run, since the computation time complexity is smaller for feature detection.

## 6.3 Failure Cases

We discuss in this paragraph some practical limitations of the SURF algorithm.

### 6.3.1 Feature Detectability

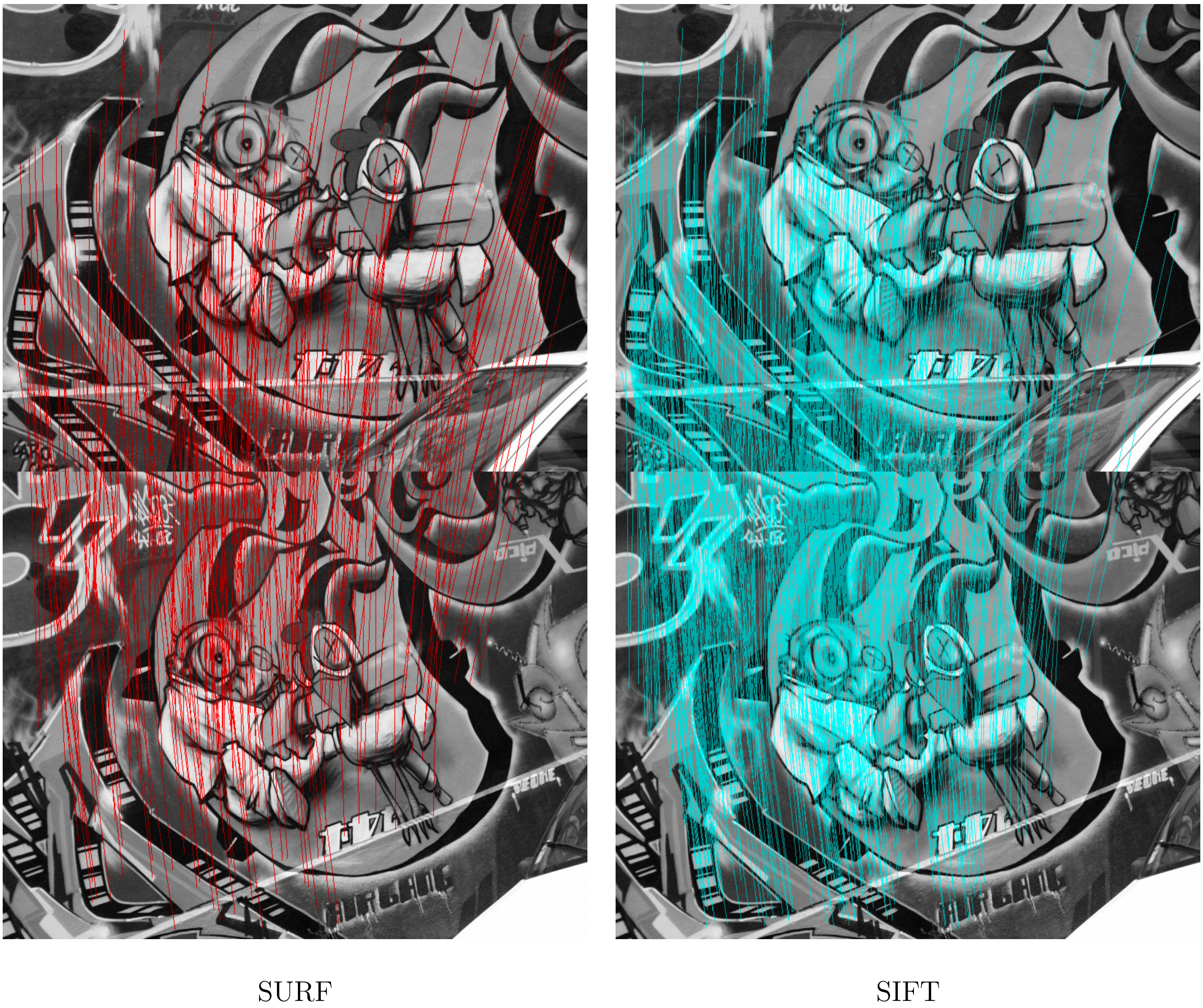
As it can be seen in the previous matching examples, the SURF detector yields a lot of interest points in textured regions or contrasted geometrical structures, such as corners, junctions, blobs, etc. However, interest points are hardly selected in low contrasted areas (see Figure 23). A similar problem occurs with the SIFT approach.

Let us recall that interest point candidates are defined as local maxima of the determinant of Hessian operator $\mathbf{DoH}^L$ (see Section 4, Equation (24)). In order to discard candidates that are due to noise fluctuation, a threshold $t_H$ is used (see Formula (27)). Unfortunately, the determinant of Hessian response depends on the local contrast, so that the thresholding also discards low contrasted features. An interesting work related to this issue consists in investigating the use of locally adapted thresholds.

### 6.3.2 Robustness to Non-Affine Contrast Changes

The SURF descriptors are based on local gradient statistics, which yield invariance to any affine contrast change, such as white balance correction. In Figure 24, we illustrate the impact of a non linear contrast change. In this example, a different gamma correction ($\gamma = \frac{1}{2}$ and $\gamma = 4$) is applied to input images before using the SURF algorithm. This result is to be compared with Figure 21 (i.e. without gamma correction). While the contrast change has little impact on feature detection, the number of matches is greatly reduced (by almost a factor of 6 in this example).

[4] available at `http://www.cs.ubc.ca/~lowe/keypoints/`.



SURF

SIFT

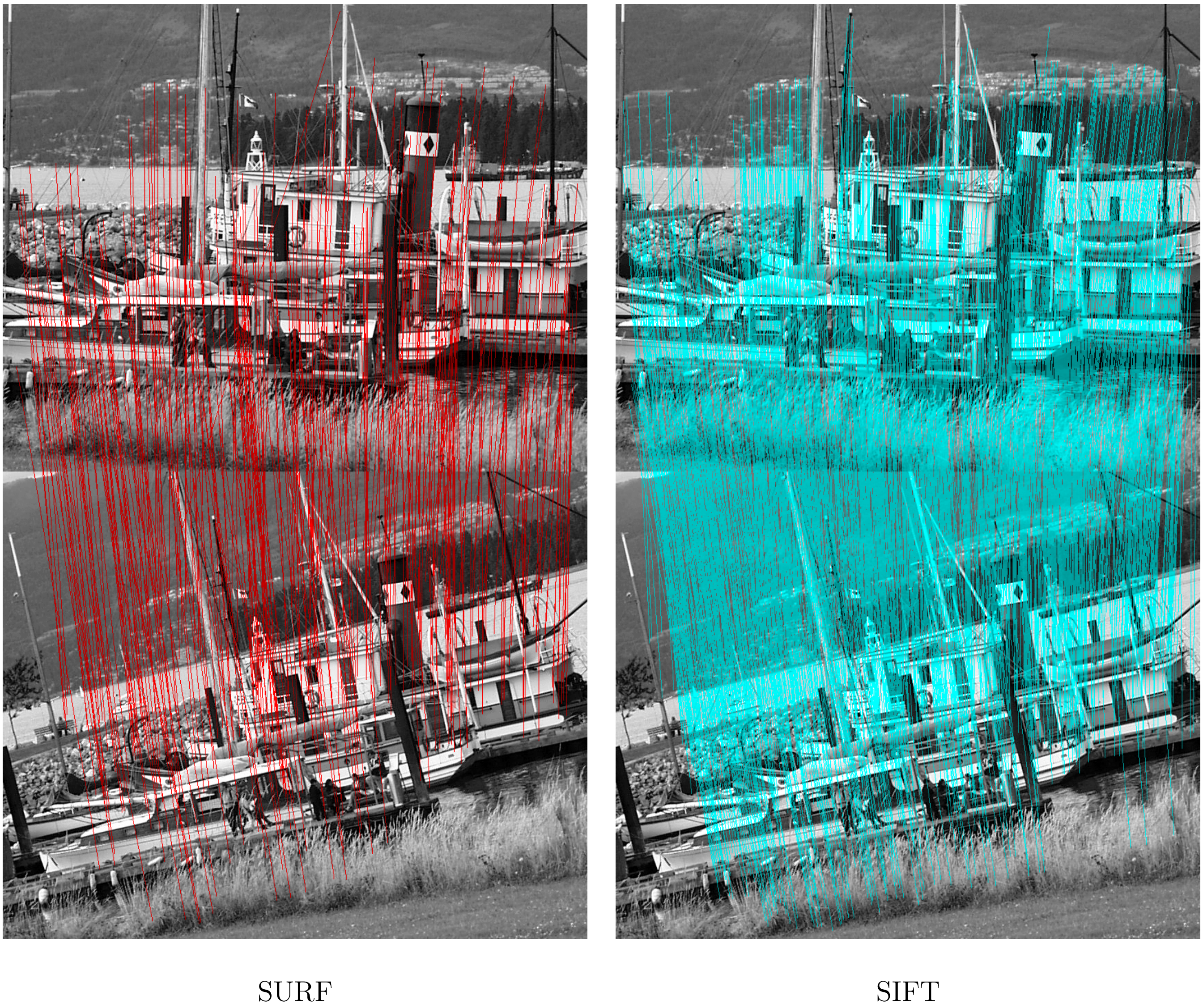
Figure 21: Comparison of SIFT and SURF matching on 'graffiti' image pair. (Left) 215 correspondences with the proposed SURF implementation. (Right) 668 correspondences with the SIFT algorithm.

### 6.3.3 Robustness to Large Perspective Distortion

In order to comply with stereo-vision applications, invariance to similarity transformations is not sufficient. As a matter of fact, such applications involve more general perspective deformations such as affine transformations. We therefore investigate now if SURF descriptors are indeed robust to small affine perturbations.

It is obviously the case when considering the two first images of the *Graffiti* sequence where the angle is about $\alpha = 20°$ between viewpoints (Figure 21). But, when considering wide baseline stereo (respectively $\alpha = 30°$, $40°$ and $50°$), as shown in Figure 25, we obtain fewer matches (resp. 75, 19 and 8). Moreover, almost all correspondences are incorrect when the viewpoint angle exceeds $\alpha = 40°$. We know from the repeatability experiment (Figure 19) that this limitation is not due to the interest point detector, but rather to the way SURF descriptors are built, using a similarity transform to define the local grid $\mathcal{R}$ (see Equation (41)).

The same limitation occurs with the SIFT approach, as illustrated in Figure 26. In this exper-



SURF

SIFT

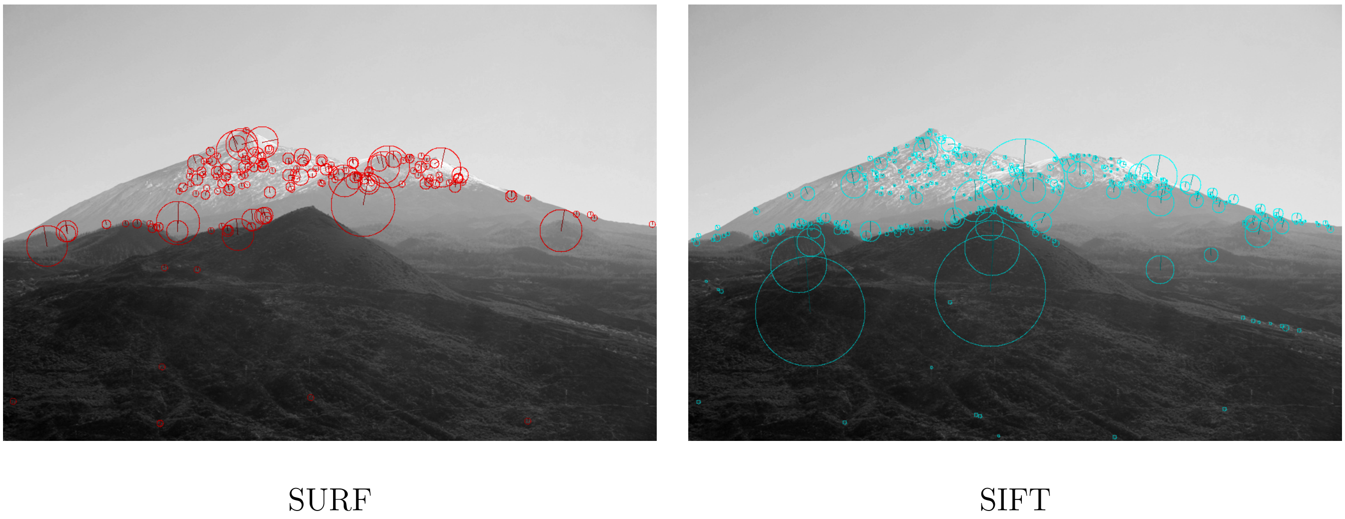
Figure 22: Comparison of SIFT and SURF matching on ‘boat’ image pair. (Left) 289 correspondences with the proposed SURF implementation. (Right) 1433 correspondences with the SIFT algorithm.

iment, it is interesting to observe that most of the correspondences found by SURF and SIFT are located in the less distorted area of the second image. In comparison, the ASIFT technique [28] -which is designed to tackle this kind of problem- manages to find 915 correspondences.

# 7 Conclusion

In this article, we have analyzed in detail the SURF [2] algorithm for local, compact and invariant representation of natural images. This method achieves a performance comparable to other state-of-the-art algorithms in image matching, such as SIFT [15], while being very fast. Its use is therefore of great interest for computer vision.

The numerical performance of the proposed implementation of SURF is consistent with that reported in [2], but also with other implementations, such as [3].



SURF

SIFT

Figure 23: Detection of SIFT and SURF features on a low contrasted image. Most selected interest points are located on the most contrasted region of the image. The lower part of the image, which has less contrasted regions, does not bear any SURF descriptor.

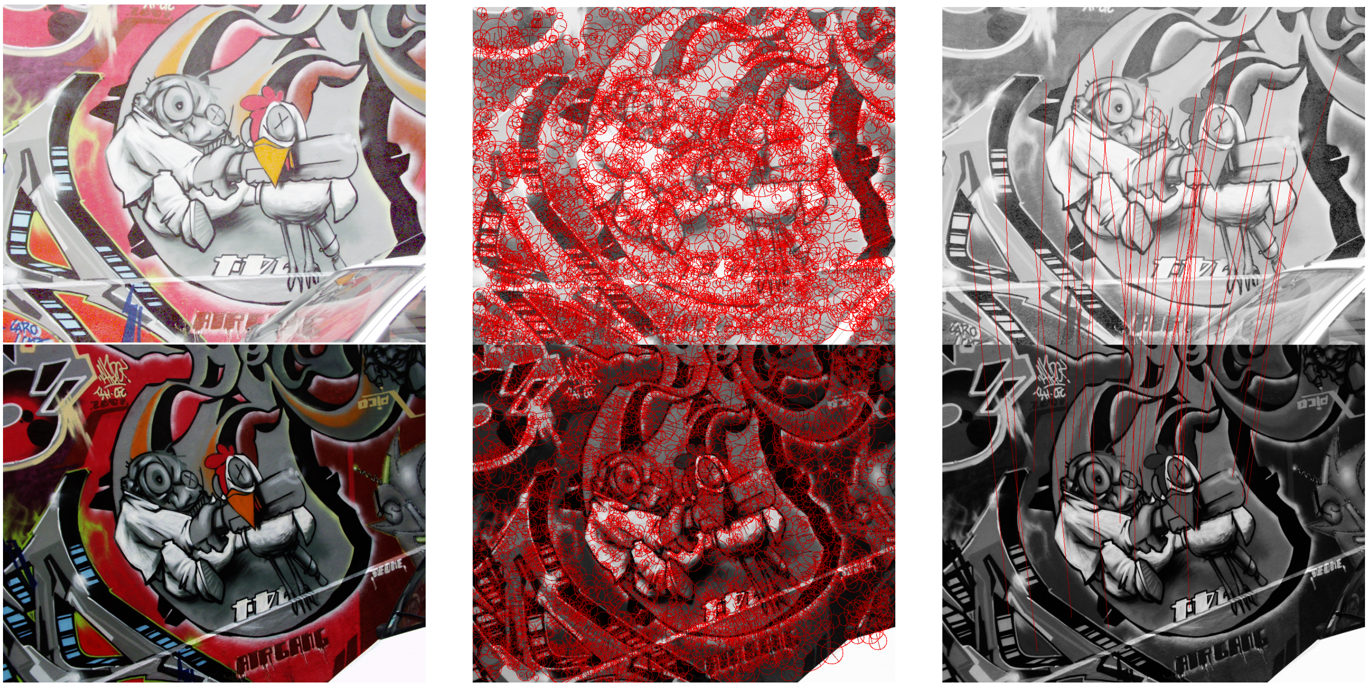


Figure 24: Detection and matching of SURF features on ‘graffiti’ sequence with gamma correction. (Left) The first two images from the dataset are pre-processed with gamma correction ($\gamma = \frac{1}{2}$ for the first image, and $\gamma = 4$ for the second). (Middle) 4114 and 3798 interest points are detected with the proposed SURF implementation. (Right) Only 37 correspondences are found after the matching and the geometric consistency steps.

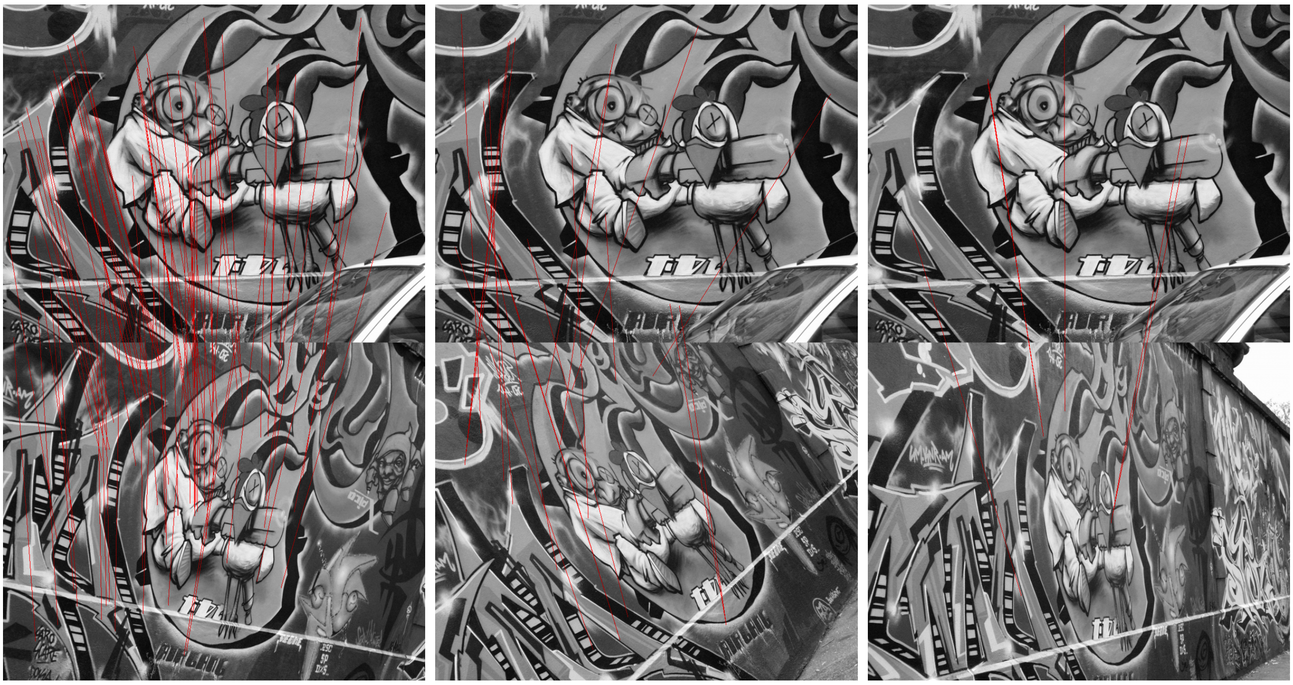


Figure 25: SURF matching on ‘graffiti’ image pair with wide baseline. (Left) 75 (correct) correspondences are found when the angle is about $\alpha = 30°$ between the views. (Middle) 19 (incorrect) correspondences for $\alpha = 40°$. (Right) 8 (incorrect) correspondences for $\alpha = 50°$.

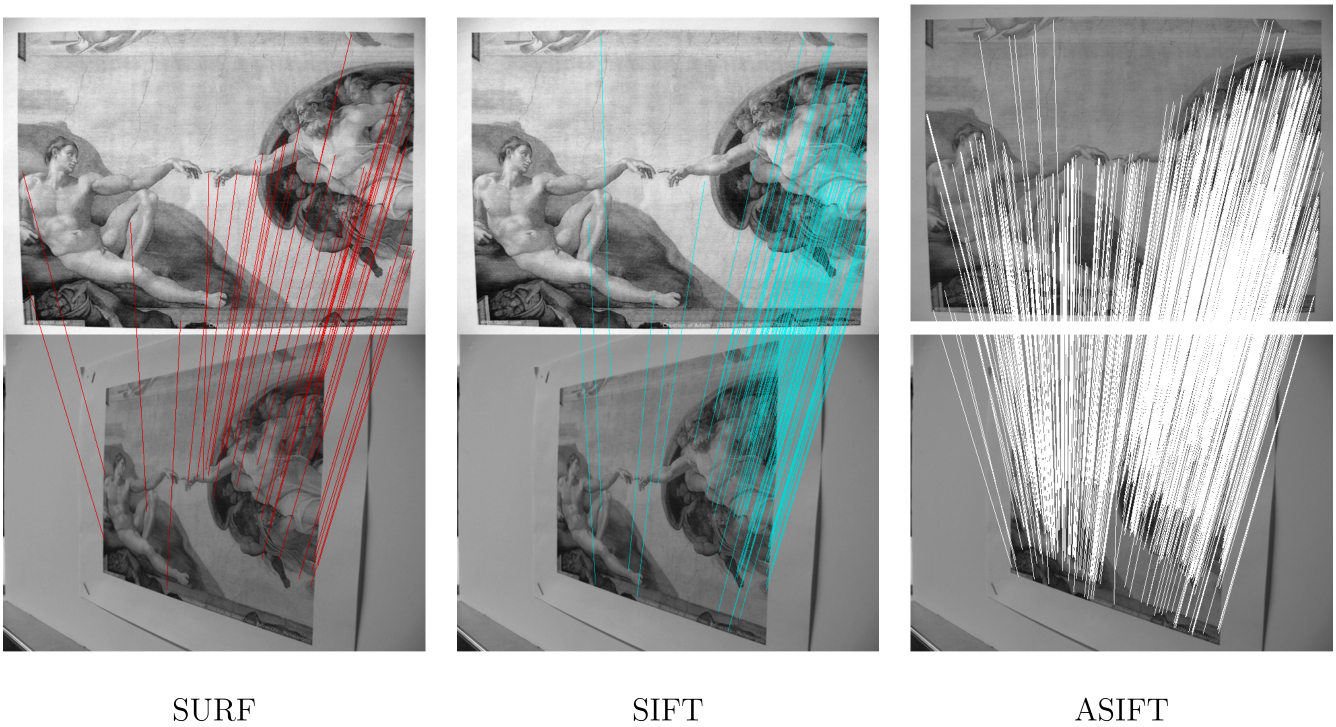


Figure 26: Matching SURF and SIFT features for wide baseline image pair, in comparison with ASIFT approach. When using similarity invariant methods (SURF and SIFT), most selected correspondences are located on the less distorted region of the image, while ASIFT [28] -an affine invariant generalization of SIFT- manages to detect much more correspondences.

## Image Credits

, from the K. Mikolajczyk standard database

Berkeley Segmentation Dataset[5]

provided by the authors

Test image used in [6]

Standard test image

## Acknowledgments

This work was supported by the European Research Council (Advanced Grant "Twelve Labours"). The authors would like also to thank Loïc Simon and anonymous reviewers for their valuable comments and their suggested improvements.

## References

[1] H. Bay, *From wide-baseline point and line correspondences to 3D*, ETH Zurich, 2009. http://dx.doi.org/10.3929/ethz-a-005212689.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, *Speeded-up robust features (SURF)*, Computer Vision and Image Understanding, 110 (2008), pp. 346–359. http://dx.doi.org/10.1016/j.cviu.2007.09.014.

[3] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*, O'Reilly Media, 2008. ISBN:978-0-596-51613-0.

[4] M. Brown and D.G. Lowe, *Invariant features from interest point groups*, in British Machine Vision Conference, Cardiff, Wales, vol. 21, 2002, pp. 656–665. http://dx.doi.org/10.5244/C.16.23.

[5] C. Evans, *Notes on the OpenSURF library*, University of Bristol, Tech. Rep. CSTR-09-001, January, (2009).

[6] M.A. Fischler and R.C. Bolles, *Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography*, Communications ACM, 24 (1981), pp. 381–395. http://dx.doi.org/10.1145/358669.358692.

[7] P. Getreuer, *A Survey of Gaussian Convolution Algorithms*, Image Processing On Line, 3 (2013), pp. 286–310. http://dx.doi.org/10.5201/ipol.2013.87.

[8] P. Gwosdek, S. Grewenig, A. Bruhn, and J. Weickert, *Theoretical foundations of Gaussian convolution by extended box filtering*, Scale Space and Variational Methods in Computer Vision, (2012), pp. 447–458. http://dx.doi.org/10.1007/978-3-642-24785-9_38.

[5] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/segbench/

[6] http://www.ipol.im/pub/art/2011/my-asift/

[9] C. Harris and M. Stephens, *A combined corner and edge detector*, in Alvey vision conference, vol. 15, Manchester, UK, 1988, p. 50.

[10] T. Lindeberg, *Scale-space for discrete signals*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 12 (1990), pp. 234–254. http://dx.doi.org/10.1109/34.49051.

[11] ——, *Scale-space behaviour of local extrema and blobs*, Journal of Mathematical Imaging and Vision, 1 (1992), pp. 65–99. http://dx.doi.org/10.1007/BF00135225.

[12] ——, *Scale-Space Theory in Computer Vision*, Springer, 1993. http://dx.doi.org/10.1007/978-1-4757-6465-9.

[13] ——, *Feature detection with automatic scale selection*, International Journal of Computer Vision, 30 (1998), pp. 79–116. http://dx.doi.org/10.1023/A:1008045108935.

[14] D.G. Lowe, *Object recognition from local scale-invariant features*, in Proceedings of the International Conference on Computer Vision, vol. 2 of ICCV '99, Washington, DC, USA, 1999, IEEE Computer Society, pp. 1150–1157. http://dx.doi.org/10.1109/ICCV.1999.790410.

[15] ——, *Distinctive image features from scale-invariant keypoints*, International Journal of Computer Vision, 60 (2004), pp. 91–110. http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94.

[16] S. Mallat, *A wavelet tour of signal processing*, Academic press, 1999. ISBN:0123743702, 9780123743701.

[17] K. Mikolajczyk and C. Schmid, *An affine invariant interest point detector*, in Computer Vision—ECCV 2002, Springer, 2002, pp. 128–142. http://dx.doi.org/10.1007/3-540-47969-4_9.

[18] ——, *Scale & affine invariant interest point detectors*, International Journal of Computer Vision, 60 (2004), pp. 63–86. http://dx.doi.org/10.1023/B:VISI.0000027790.02288.f2.

[19] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L.V. Gool, *A comparison of affine region detectors*, International Journal of Computer Vision, 65 (2005), pp. 43–72. http://dx.doi.org/10.1007/s11263-005-3848-x.

[20] L. Moisan, P. Moulon, and P. Monasse, *Automatic Homographic Registration of a Pair of Images, with A Contrario Elimination of Outliers*, Image Processing On Line, 2 (2012), pp. 56–73. http://dx.doi.org/10.5201/ipol.2012.mmm-oh.

[21] L. Moisan and B. Stival, *A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix*, International Journal of Computer Vision, 57 (2004), pp. 201–218. http://dx.doi.org/10.1023/B:VISI.0000013094.38752.54.

[22] A. Neubeck and L. Van Gool, *Efficient non-maximum suppression*, in 18th International Conference on Pattern Recognition, vol. 3, IEEE, 2006, pp. 850–855. http://dx.doi.org/10.1109/ICPR.2006.479.

[23] J. Rabin, J. Delon, and Y. Gousseau, *A contrario matching of SIFT-like descriptors*, in 19th International Conference on Pattern Recognition, IEEE, 2008, pp. 1–4. http://dx.doi.org/10.1109/ICPR.2008.4761371.

[24] I. Rey Otero and M. Delbracio, *Anatomy of the SIFT Method*, Image Processing On Line, 4 (2014), pp. 370–396. http://dx.doi.org/10.5201/ipol.2014.82.

[25] P.Y. Simard, L. Bottou, P. Haffner, and Y. Le Cun, *Boxlets: a fast convolution algorithm for signal processing and neural networks*, Advances in Neural Information Processing Systems, (1999), pp. 571–577. ISBN:0-262-11245-0.

[26] E. Tola, V. Lepetit, and P. Fua, *Daisy: An efficient dense descriptor applied to wide-baseline stereo*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 32 (2010), pp. 815–830. http://dx.doi.org/10.1109/TPAMI.2009.77.

[27] P. Viola and M. Jones, *Rapid object detection using a boosted cascade of simple features*, in Proceedings of the IEEE Computer Society Conference, vol. 1, 2001, pp. I–511–I–518 vol.1. http://dx.doi.org/10.1109/CVPR.2001.990517.

[28] G. Yu and J.-M. Morel, *ASIFT: an algorithm for fully affine invariant comparison*, Image Processing On Line, 2011 (2011). http://dx.doi.org/10.5201/ipol.2011.my-asift.