École Normale Supérieure de Cachan, France

Anatomy of the SIFT method

A dissertation presented
by

Ives Rey-Otero

in fulfillment of the requirements
for the degree of Doctor of Philosophy
in the subject of Applied Mathematics

Committee in charge

Referees	Coloma Ballester	-	Universitat Pompeu Fabra, ES
	Pablo Musé	-	Universidad de la República, UY
	Joachim Weickert	-	Universität des Saarlandes, DE
Advisors	Jean-Michel Morel	-	ENS de Cachan, FR
	Mauricio Delbracio	-	Duke University, USA
Examiners	Patrick Pérez	-	Technicolor Research, FR
	Frédéric Sur	-	Loria, FR

ENSC-2015 603

September 2015

Version of November 5, 2015 at 12:27.

Abstract of the Dissertation

Ives Rey-Otero

Anatomy of the SIFT method

Under the direction of:
Jean-Michel Morel and Mauricio Delbracio

This dissertation contributes to an in-depth analysis of the SIFT method. SIFT is the most popular and the first efficient image comparison model. SIFT is also the first method to propose a practical scale-space sampling and to put in practice the theoretical scale invariance in scale space. It associates with each image a list of scale invariant (also rotation and translation invariant) features which can be used for comparison with other images. Because after SIFT feature detectors have been used in countless image processing applications, and because of an intimidating number of variants, studying an algorithm that was published more than a decade ago may be surprising. It seems however that not much has been done to really understand this central algorithm and to find out exactly what improvements we can hope for on the matter of reliable image matching methods. Our analysis of the SIFT algorithm is organized as follows. We focus first on the exact computation of the Gaussian scale-space which is at the heart of SIFT as well as most of its competitors. We provide a meticulous dissection of the complex chain of transformations that form the SIFT method and a presentation of every design parameter from the extraction of invariant keypoints to the computation of feature vectors. Using this documented implementation permitting to vary all of its own parameters, we define a rigorous simulation framework to find out if the scale-space features are indeed correctly detected by SIFT, and which sampling parameters influence the stability of extracted keypoints. This analysis is extended to see the influence of other crucial perturbations, such as errors on the amount of blur, aliasing and noise. This analysis demonstrates that, despite the fact that numerous methods claim to outperform the SIFT method, there is in fact limited room for improvement in methods that extract keypoints from a scale-space. The comparison of many detectors proposed in SIFT competitors is the subject of the last part of this thesis. The performance analysis of local feature detectors has been mainly based on the repeatability criterion. We show that this popular criterion is biased toward methods producing redundant (overlapping) descriptors. We therefore propose an amended evaluation metric and use it to revisit a classic benchmark. For the amended repeatability criterion, SIFT is shown to outperform most of its more recent competitors. This last fact corroborates the unabating interest in SIFT and the necessity of a thorough scrutiny of this method.

Contents

1 Introduction **7**

2 Computing an exact Gaussian scale-space **19**
2.1 Introduction . . . 19
2.2 Mathematical Preliminaries . . . 21
2.3 Analysis of three digital Gaussian convolution algorithms . . . 23
2.4 Experiments . . . 30
2.5 Conclusion . . . 30

3 Anatomy of the SIFT Method **37**
3.1 General description . . . 37
3.2 The Gaussian scale-space . . . 38
3.3 Keypoint definition . . . 45
3.4 Keypoint description . . . 53
3.5 Matching . . . 61
3.6 Summary of Parameters . . . 61

4 An analysis of scale-space sampling and keypoints detection in SIFT **65**
4.1 Introduction . . . 65
4.2 The exact implementation of the SIFT method . . . 66
4.3 The theoretical scale invariance . . . 68
4.4 Simulating the digital camera . . . 70
4.5 Empirical analysis of the digital scale-space sampling . . . 71
4.6 Impact of deviations from the perfect camera model . . . 77
4.7 Concluding remarks . . . 80

5 Is repeatability an unbiased criterion for ranking feature detectors? **83**
5.1 Introduction . . . 83
5.2 The repeatability criterion and its bias . . . 85
5.3 Non-redundant repeatability . . . 87
5.4 Spatial coverage of state-of-the-art feature detectors . . . 89
5.5 Experiments . . . 95
5.6 Discussion . . . 103

6 Conclusion **107**

Bibliography **109**

1 Introduction

Motivation

Local stable features are the cornerstone of many image processing and computer vision applications such as image registration [Hartley and Zisserman 2003; Snavely et al. 2006], camera calibration [Grompone von Gioi et al. 2010], image stitching [Haro et al. 2012], 3D reconstruction [Agarwal et al. 2011], object recognition [Grimson and Huttenlocher 1990; Fergus et al. 2003; Bay et al. 2006a; Zhang et al. 2007].

The seminal paper introducing SIFT [Lowe 1999] has sparked an explosion of local keypoints detector/descriptors seeking discrimination and invariance to a specific group of image transformations [Tuytelaars and Mikolajczyk 2008]. SURF [Bay et al. 2006b], Harris and Hessian based detectors [Mikolajczyk et al. 2005], MOPS [Brown et al. 2005], ASIFT [Yu and Morel 2011] and SFOP [Förstner et al. 2009] with methods using binary descriptors such as BRISK [Leutenegger et al. 2011] and ORB [Rublee et al. 2011], are just a few of the successful variants. These add to the numerous non multi-scale detectors such as the Harris-Stephens detector [Harris and Stephens 1988], SUSAN [Smith and Brady 1997], the Förstner detector [Förstner 1994], the morphological corner detector [Alvarez and Morales 1997] and the machine learning based FAST [Rosten and Drummond 2006] and AGAST [Mair et al. 2010].

The importance of feature detectors in countless applications of image processing as well as the intimidating number of variants led us to return where it all started, namely the publication of the SIFT algorithm more than a decade ago. Despite the number of publications, it seems that not much has been done to really understand this central algorithm and to rigorously figure out what improvement we can hope for on the matter of reliable image matching methods. This thesis proposes an in-depth analysis of the SIFT algorithm. It is organized as follows. Chapter 2 focuses on the accurate computation of the Gaussian scale-space which is at the heart of SIFT as well as most of its competitors [Mikolajczyk et al. 2005; Brown et al. 2005]. Chapter 3 offers a meticulous dissection of the complex chain of transformations that form the SIFT method and a presentation of every design parameter from the extraction of invariant keypoint to the computation of feature vectors. In Chapter 4, we use a rigorous image simulation framework to analyze the influence of scale-space sampling on the stability of extracted keypoints on diverse scenarios. This analysis will demonstrate that, despite numerous methods claiming to outperform SIFT, there is in fact limited room for improvement for a more complete or more accurate keypoint detection in scale-space. Nevertheless our study concludes with several significant improvements on scale-space sampling to improve the detection effi-

ciency. The comparison of detectors is the subject of Chapter 5. The evaluation of local feature detectors has been mainly based on the repeatability criterion [Mikolajczyk et al. 2005]. We argue here that this popular criterion is biased towards redundant methods. We propose an amended evaluation metric and use it to revisit a classic benchmark. Again, SIFT is shown to outperform most of its more recent competitors. We close this dissertation with some conclusions and perspectives that are discussed in Chapter 6. This work tries to apply the standards of reproducible research. All algorithms are described in detail (providing a pseudocode) to guarantee a complete description. In particular the IPOL article Anatomy of SIFT [Rey-Otero and Delbracio 2014] (detailed in Chapter 3) permits to explore on line on any image pair all intermediate steps of the SIFT method, to experiment its parameters, and to see their influence on the intermediate steps and on the final result.

The next sections review in more detail the contributions of each chapter.



Figure 1.1: Examples of SIFT applications (excerpt of the IPOL demo archive).

Chapter 2: Computing an exact Gaussian scale-space

SIFT attains scale invariance thanks to the Gaussian scale-space, a multi-scale image representation simulating the family of all possible zoom-outs through increasingly blurred versions of the input image. The continuous Gaussian scale-space of an image $u(\mathbf{x})$ defined for every $\mathbf{x} = (x, y) \in \mathbb{R}^2$ is the function

$$v : (\sigma, \mathbf{x}) \mapsto G_\sigma u(\mathbf{x}),$$

where $G_\sigma u(\mathbf{x})$ denotes the convolution of $u(\mathbf{x})$ with a Gaussian kernel of standard deviation $\sigma > 0$ (the scale), namely

$$G_\sigma u(\mathbf{x}) := \int G_\sigma(\mathbf{x}')u(\mathbf{x} - \mathbf{x}')d\mathbf{x}', \quad \text{with } G_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma^2} e^{-\frac{\|\mathbf{x}\|^2}{2\sigma^2}}.$$

The Gaussian convolution is the cornerstone of several image processing algorithms either as a fast pre-process to increase noise robustness before applying another algorithm, or as the fundamental operator in scale-space theory [Iijima et al. 1974; Lindeberg 1993; Sporring et al. 1997; Witkin 1984]. In SIFT, the Gaussian kernel acts as an approximation of the optical blur introduced in the camera (represented by its point spread function). The Gaussian approximation is convenient because, among other things, it satisfies the semi-group property

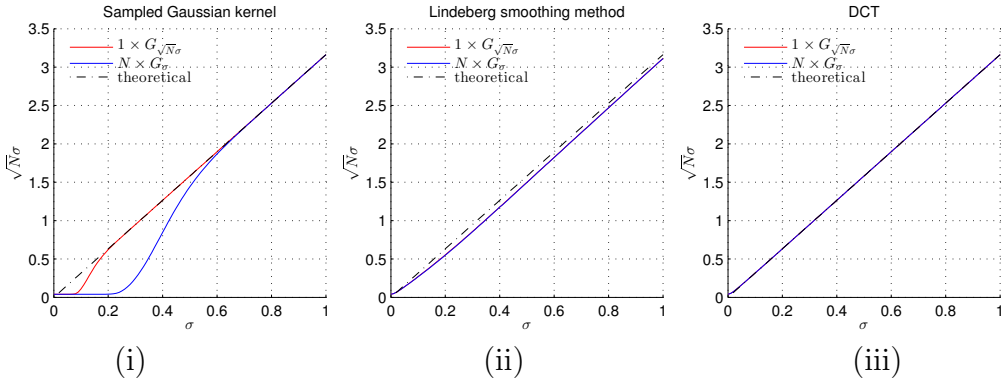
$$G_\sigma G_\gamma u(\mathbf{x}) = G_{\sqrt{\sigma^2+\gamma^2}} u(\mathbf{x}).$$

In particular, this permits to simulate distant snapshots from closer ones. Thus, the scale-space can be seen as a stack of images, each one corresponding to a different zoom factor.

What is the discrete counterpart of this continuous operator? Could it be defined to satisfy the properties of the continuous Gaussian convolution? Numerous algorithms have been proposed for approximating the Gaussian convolution in digital images [Getreuer 2013]. In Chapter 2, we discuss and numerically analyze the precision of three different alternatives for defining a discrete counterpart to the continuous Gaussian operator, namely, the discrete convolution with Gaussian kernel samples, Lindeberg's discrete scale-space smoothing [Lindeberg 1993] (which consists in computing the solution of a spatial discretization of the heat equation), and the Fourier based convolution. We focus on low blur levels, that are crucial for the scale-space accuracy.

We use the semi-group property for measuring the accuracy of each of the analyzed methods. In particular, we test if multiple iterations of the same Gaussian convolution produce the same result as a single convolution with blur level foretold by the semi-group property. An example of such experiment is displayed in Figure 1.2 where the direct convolution and the multiple iterations are performed on the image of a sampled Gaussian function and where a Gaussian function is fitted to the result, to measure the standard deviation of the resulting image.

The conclusions are straightforward. The only method that allows to compute accurately the Gaussian scale-space is the Fourier based convolution. This algorithm computes



(i) (ii) (iii)

Figure 1.2: Using the semi-group property for measuring the accuracy of three Gaussian convolution methods. A Gaussian convolution of parameter $\sqrt{N}\sigma$ is compared to $N = 10$ iterations of a Gaussian convolution of parameter σ for different values of σ. The estimated blur levels for the direct and iterated filters are plotted as a function of σ for: (i) the discrete convolution with Gaussian kernel samples, (ii) Lindeberg's discrete scale-space smoothing and (iii) the Fourier based convolution. The theoretical (expected) value $\sqrt{N}\sigma$ is plotted in black.

exactly the continuous Gaussian convolution at the cost of two DFTs and one operation per pixel:

$$G_\sigma * u(x,y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{(-\lfloor \frac{M}{2} \rfloor + M - 1)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{(-\lfloor \frac{N}{2} \rfloor + N - 1)} \tilde{u}_{m,n} \widehat{G}_\sigma \left(\frac{2\pi m}{M}, \frac{2\pi n}{N} \right) e^{\frac{2i\pi mx}{M}} e^{\frac{2i\pi ny}{N}},$$

where $\widehat{G}_\sigma(\frac{2\pi m}{M}, \frac{2\pi n}{N}) = e^{-\frac{\sigma^2 \pi^2}{2}\left(\left(\frac{2m}{M}\right)^2 + \left(\frac{2n}{N}\right)^2\right)}$ and $\tilde{u}_{m,n}$ are the DFT coefficient of a $M \times N$ image. The discrete convolution with samples from a Gaussian kernel is accurate only if the applied blur level is large enough to avoid aliasing artifacts (i.e., $\sigma > 0.8$). Although Lindeberg's discrete scale-space smoothing method satisfies the semi-group property, it introduces a bias resulting in a lower amount of blur applied.

Evident though they are, these conclusions are vital in the goal of getting a full understanding of the performance and possible limitations of SIFT implementations. We therefore use systematically the conclusions and tools of this study in Chapter 4, to build a perfectly stable scale-space (independent of sampling issues) permitting an unbiased analysis of the SIFT method.

Chapter 3: Anatomy of the SIFT method

The SIFT method is the cornerstone of numerous applications in image processing and computer vision. Because of this ubiquity, SIFT can easily be mistaken with a simple pre-process step that extracts from an image a set of descriptors that are invariant to translations, rotations and zoom-outs. The SIFT algorithm is nevertheless a complex chain of transformations (as illustrated by the summary in Table 1.1).

The Devil is in the details. For each step of the method, multiple implementations are acceptable. Choosing one implementation instead of another impacts in turn the method's performance and invariance properties. Chapter 3 presents a detailed description and

Stage	Description
1.	Compute the Gaussian scale-space in: $\mathbf{u}$ image out:$\mathbf{v}$ scale-space
2.	Compute the Difference of Gaussians (DoG) in: $\mathbf{v}$ scale-space out: $\mathbf{w}$ DoG
3.	Find candidate keypoints (3D discrete extrema of DoG) in: $\mathbf{w}$ DoG out: $\{(x_d, y_d, \sigma_d)\}$ list of discrete extrema (position and scale)
4.	Refine candidate keypoints location with sub-pixel precision in: $\mathbf{w}$ DoG and $\{(x_d, y_d, \sigma_d)\}$ list of discrete extrema out: $\{(x, y, \sigma)\}$ list of interpolated extrema
5.	Filter unstable keypoints due to noise in: $\mathbf{w}$ DoG and $\{(x, y, \sigma)\}$ out: $\{(x, y, \sigma)\}$ list of filtered keypoints
6.	Filter unstable keypoints laying on edges in: $\mathbf{w}$ DoG and $\{(x, y, \sigma)\}$ out: $\{(x, y, \sigma)\}$ list of filtered keypoints
7.	Assign a reference orientation to each keypoint in: $(\partial_m \mathbf{v}, \partial_n \mathbf{v})$ scale-space gradient and $\{(x, y, \sigma)\}$ list of keypoints out: $\{(x, y, \sigma, \theta)\}$ list of oriented keypoints
8.	Build the keypoints descriptor in: $(\partial_m \mathbf{v}, \partial_n \mathbf{v})$ scale-space gradient and $\{(x, y, \sigma, \theta)\}$ list of keypoints out: $\{(x, y, \sigma, \theta, \mathbf{f})\}$ list of described keypoints

Table 1.1: Summary of the SIFT algorithm.

implementation of the SIFT method. From the computation of the Gaussian scale-space of an input image (see Figure 1.3) to the extraction and encoding of keypoint feature vectors (see Figure 1.4), the SIFT algorithm is described in a unequivocal way with pseudocodes. This contributes to a detailed dissection of the method and to a careful presentation of each of its design parameters. As part of a submission to the journal of reproducible research in Image Processing IPOL, this detailed description as well as the source code have been peer-reviewed. A companion online demonstrator allows the reader to use SIFT and individually vary each parameter to analyze its impact on the algorithm results. Since its publication in December 2014, more than 800 online experiments have been recorded in the IPOL archive. Additionally, the Anatomy of SIFT has been used as teaching material at the graduate school of École Normale Supérieure de Cachan in 2013 and 2014.

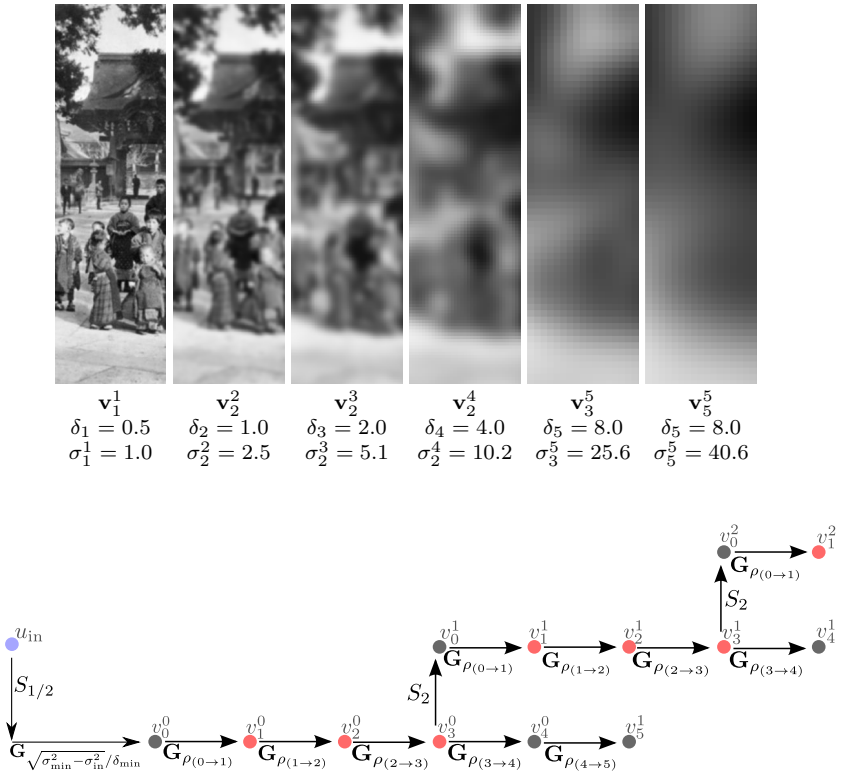


Figure 1.3: The bottom image summarizes the succession of subsamplings and Gaussian convolutions that results in the SIFT scale-space. All images in the scale-space are computed directly or indirectly from the input image (in blue). Each image is characterized by its blur level and its inter-pixel distance.The scale-space is split into octaves: sets of images sharing a common sampling rate. Each octave is composed of three scales (in red) and other three auxiliary scales (in gray).

Chapter 4: An analysis of scale-space sampling and keypoints detection in SIFT

This chapter uses the fully consistent numerical implementation of the scale-space in Chapter 2 and the detailed SIFT parameter analysis of Chapter 3. Its first goal is to assess if



Figure 1.4: The construction of the SIFT descriptor.

indeed the SIFT method successfully detects all the DOG scale-space extrema regardless of sampling issues. This requires the expansion of the method to allow for variations in all of its parameters, to transform it into an exact method. In that way it becomes possible to compare the exact method (more precisely a strongly oversampled scale-space version) to the original one, and to evaluate the completeness and stability of the detected keypoints. The SIFT algorithm has proven to be sufficiently scale invariant to be used in numerous applications. In practice, however, scale invariance may be weakened by various sources of error inherent to the SIFT implementation affecting detections stability and accuracy. The density of the sampling of the Gaussian scale-space and the level of blur in the input image are two of these sources. Chapter 4 presents an empirical analysis of their impact on the extracted keypoints stability.

This empirical analysis relies on a strict image simulation framework. Such framework allows to simulate images that are rigorously consistent with the SIFT camera model. In particular, in this model, the camera point spread function is assumed to be a Gaussian function. Furthermore, it allows to control the camera blur level, aliasing and noise level in the input image, and therefore to measure how invariant is SIFT in a variety of realistic scenarios.

This systematic analysis has both methodological and practical implications, on how to compare feature detectors and on how to improve the SIFT algorithm. We show that increasing the scale-space sampling (both in scale and in space) improves the stability of the detections and the precision of their localization (see Figure 1.5 which reports on the precision of the detections in a series of zoom-outs). We show however, that even with a significantly oversampled scale-space numerical errors prevent from achieving perfect stability. The filtering of unstable keypoints is also explored. Usual strategies to discard unstable detections are shown to be inefficient. Indeed, the ROC (receiver operating characteristic) curve shown in Figure 1.6 illustrates that none of the simple features based on the detector response manage to faithfully separate the stable detections from unstable ones. While aliasing in the input image does not affect the number of detections, it affects stability. For a sufficiently large camera blur $c > 0.6$, the impact of aliasing is shown to be negligible. We also investigate, the effects of a wrong assumption by SIFT on the camera blur as well as the influence of image noise.

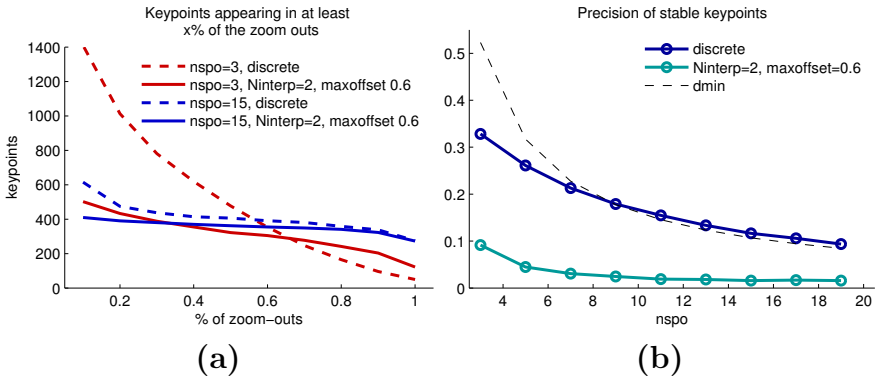


Figure 1.5: Influence of scale-space sampling and extrema refinement on the invariance to zoom-outs. A set of zoomed-out images was simulated and the keypoints extracted. **(a)** The number of keypoints appearing in at least a certain percentage of the simulated images for different scale-space sampling and refinements. The best performance is obtained by significantly upsampling the scale-space and by refining the extrema with the local interpolation. In this case, most of the detected keypoints are present in all the simulated images. On the other hand, the original SIFT sampling leads to low stability even with the extrema refinement step. **(b)** Mean precision of stable keypoints location (appearing in at least 50% of the zoom-outs) plotted as a function of the sampling rate. The local refinement of the extrema position significantly increases the precision of the extrema detection. Also, sampling the scale-space finer than what is proposed in SIFT allows to better localize the extrema.

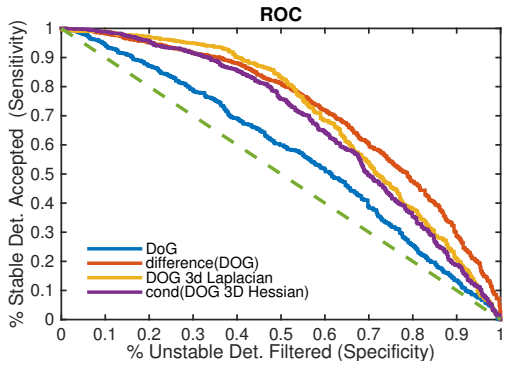


Figure 1.6: Filtering keypoints that are unstable to changes in the scale-space sampling. We consider features computed from the SIFT detector response, namely the difference of Gaussian (DoG). The considered features are the extremum DoG value, the difference between the extremum DoG value and the adjacent samples in the scale-space, the DoG 3D Laplacian at the extremum and finally the condition number of the DoG 3D Hessian at the extremum. The ROC curves illustrate the performance of each feature. A point in a ROC curve indicates the proportion of non-filtered stable keypoints (good detections – sensitivity) as a function of the filtered unstable ones (good removals – specificity) for a particular threshold value. A perfect feature should produce a ROC that is always one. None of the tested features completely allows to separate the unstable from the stable detections. Also, the worst feature for eliminating keypoints unstable to changes in the scale-space sampling is the DoG value.

Chapter 5: Is repeatability an unbiased criterion for ranking feature detectors?

Because so many computer vision applications rely on finding local correspondences between different images, new keypoint detectors and descriptors are constantly being proposed, each one claiming to perform better than the preceding ones. Figure 1.7 shows the detection maps on the `siemens` pattern for some of those methods. Most of the detectors appear to be visually highly redundant. With this symmetric picture as an input, a visual inspection clearly shows that some of the methods are not rotation invariant. This raises the question of how to do a fair comparison between very diverse methods.

Over the last decade, such an evaluation has been mainly based on the repeatability criterion [Mikolajczyk et al. 2005]. The repeatability rate measures the detector's ability to identify the same features (i.e., *repeated* detections) despite variations in the viewing conditions (blur, illumination, rotations, homotheties, homographies, etc). Defined as the ratio between the number of keypoints simultaneously present in all the images of the series (repeated keypoints) over the total number of detections, it can be seen as a measure of the detector's efficiency. Indeed, the repeatability rate incorporates two struggling quality criterion: the number of repeated detections (i.e., potential correspondences) should be maximized while the total number of detections should be minimized since the complexity of the matching grows with the square of the number of detections.

However, because it ignores the keypoints spatial distribution, the repeatability criterion favors redundancy. The following mental experiment illustrates why. Let DET be a generic keypoint detector, and let DET2 be a variant in which each detection is simply counted twice. The number of repeatable keypoints and the total number of detections are both artificially doubled, leaving the repeatability rate unchanged. However, although the number of costly descriptor computations has doubled, no extra benefit can be extracted from the enlarged set of repeated keypoints. The classic repeatability rate fails to report that the benefit over cost ratio of DET2 is half the one of DET. This explains why methods producing correlated detections may misleadingly get better repeatability ratios. Additionally, the reference and widely used code provided by the authors of [Mikolajczyk et al. 2005] does not implement the criterion defined in their article, casting a doubt on the conclusions of all benchmarks that have used this popular code. In Chapter 5, we explain the differences between the criterion published in [Mikolajczyk et al. 2005] and what is actually implemented in the code provided by the authors.

The bias towards redundant detectors motivated the introduction of a variant of the repeatability rate that takes into account the descriptor overlap. To measure the descriptors overlap, each detection $(\mathbf{x}_k, \Sigma_k)$ is assigned an elliptical mask function $f_k(\mathbf{x})$

$$f_k(\mathbf{x}) = Ke^{-\frac{1}{2\zeta^2}(\mathbf{x}-\mathbf{x}_k)^T\Sigma_k^{-1}(\mathbf{x}-\mathbf{x}_k)},$$

if $(\mathbf{x}-\mathbf{x}_k)^T\Sigma_k^{-1}(\mathbf{x}-\mathbf{x}_k) \leq \rho^2$ and 0 elsewhere, with parameters ρ and ζ derived from the descriptor's design.

Denoting $\mathcal{K}$ the set of all detections, the sum of all descriptor masks $\sum_{k\in\mathcal{K}} f_k(\mathbf{x})$ yields a final map showing how much each image pixel contributes to the set of all computed descriptors. Similarly, the maximum taken over the set of all detections $\max_{k\in\mathcal{K}} f_k(\mathbf{x})$, measures the contribution of pixel $\mathbf{x}$ to the best descriptor. The sum of this maximum

over the image domain

$$K_{\mathrm{nr}} := \int_{\Omega} \left(\max_{k \in \mathcal{K}} f_k(\mathbf{x}) \right) d\mathbf{x} \tag{1.1}$$

measures the number of *non-redundant* keypoints. This value can be interpreted as a count of the independent detections. Replacing in the repeatability rate the number of repeated detection by the number of *non-redundant* repeated keypoints yields the *non-redundant* repeatability rate.

We apply this variant to revisit the popular benchmark by Mikolajczyk et al. [2005], comparing twelve classic or recently introduced feature detectors. Figure 1.8 shows the average repeatability and non redundant repeatability of each method on the Oxford sequence plotted as a function of the average number of detections. Experimental evidence shows that the hierarchy of these feature detectors is severely disrupted by the amended comparator. Indeed, the methods that happen to be the most redundant (namely the Hessian and Harris based methods [Mikolajczyk et al. 2005]) are also the methods that perform best according to the classic repeatability (see Figure 1.8 **(a)** Once redundancy is taken into account, the method that gives the highest score while providing numerous keypoints is the SIFT method (see Figure 1.8 **(b)**. We also combine a common descriptor technique to all methods and evaluate their matching performances, which seem to be in agreement with the proposed repeatability criterion.

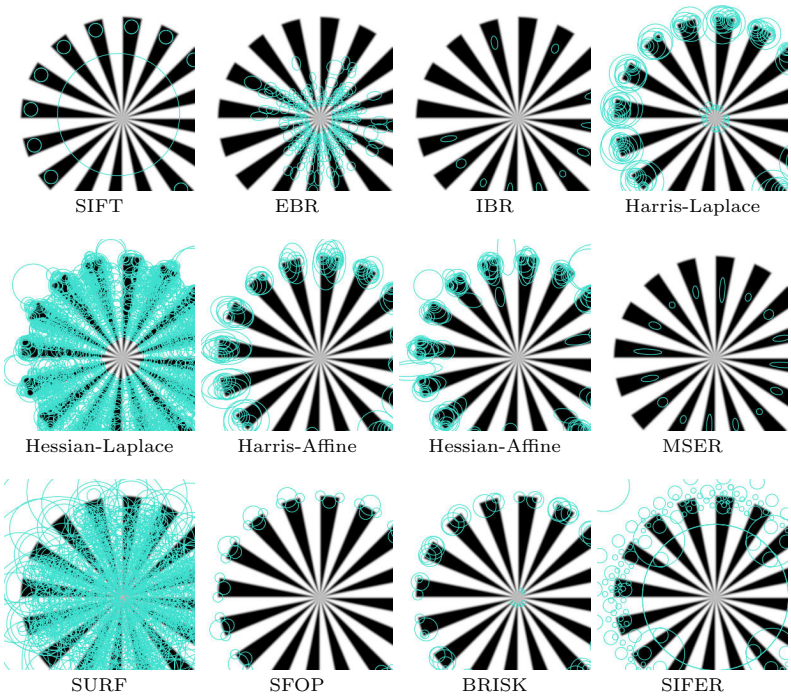


Figure 1.7: Detected keypoints on the `siemens star` test image. Since the publication of SIFT, the image processing community has been buried in an avalanche of feature detectors, all claiming to outperform the competition.

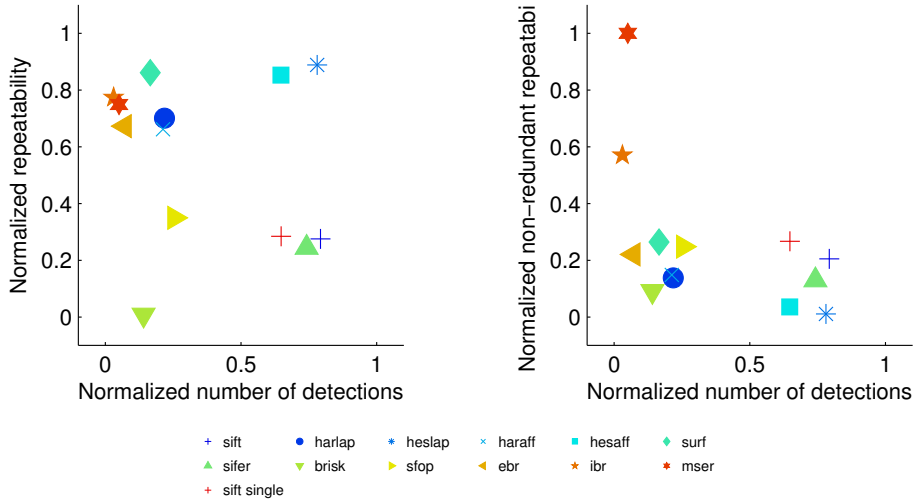


Figure 1.8: Average classic repeatability and non-redundant repeatability as a function of average number of detections, normalized over various sequences. A method performs optimally if it is simultaneously extremal in ordinate and in abscissa, and performs well if it is extremal in at least one of the coordinates. Methods that go up between the classic and the non-redundant repeatability, such as SIFT, are the methods that are less redundant in average.

Summary of contributions

- A review of algorithms used for Gaussian convolution with a focus on Gaussian scale-space computation and a proof that only an exact Fourier based method achieves full consistency with the scale-space requirements.
- A dissection of the SIFT implementation with a peer-reviewed source code and an online demonstrator permitting to vary all parameters and explore their impact on each single intermediate step for the algorithm (scale-space, keypoints, orientation histograms, descriptor, final matching).
- A thorough analysis of the empirical scale invariance using a strict simulation framework.
- The identification of a bias in the most popular performance metric for keypoint detectors. Proposition of an amended criterion and a revision of the benchmark of many state of the art algorithms. In fact, this analysis shows that the SIFT method has not been significantly improved by more recent methods and further justifies the importance of analyzing thoroughly this classic tool.

List of publications

The work in this thesis has led to the publication of the following articles:

- I. Rey-Otero and M. Delbracio. *Computing an exact Gaussian scale-space*. Submitted to Image Processing On Line, April 2014. Under review. Conditionally accepted. `http://www.ipol.im/pub/pre/117/`.

- I. Rey-Otero, J.M. Morel and M. Delbracio. *An Analysis of scale-space sampling in SIFT*. In Image Processing (ICIP), 2014 IEEE International Conference on. pages 4847-4851.
- I. Rey-Otero and M. Delbracio. *Anatomy of the SIFT Method.* Image Processing On Line, 4:370396, 2014. `http://www.ipol.im/pub/art/2014/82/`.
- I. Rey-Otero, M. Delbracio and J.M. Morel. *Comparing feature detectors: A bias in the repeatability criteria*. In Image Processing (ICIP), 2015 IEEE International Conference.
- I. Rey-Otero and M. Delbracio. *Is repeatability an unbiased criterion for ranking feature detectors.* Submitted to SIAM Journal on Imaging Sciences, January 2014, Accepted.
- I. Rey-Otero, J.M. Morel and M. Delbracio. *An analysis of scale-space sampling and keypoints detection in SIFT.* In preparation.

Source codes and demos

Being able to reproduce experiments is a major problem in computer science. This work tries to apply the standards of reproducible research. To that end, all the codes used in the course of this work are documented and distributed freely:

- Chapter 2: `https://github.com/ivreo/gaussian_convolution`
- Chapter 3: `https://github.com/ivreo/sift_anatomy`
- Chapter 4: `https://github.com/ivreo/sift_anatomy_extra`
- Chapter 5: `http://dev.ipol.im/~reyotero/comparing_20140906.tar.gz`

Additionnaly, two of the produced articles were published in the IPOL journal where they can be tested online. This open access journal seeks to mitigate the reproducibility problem by publishing for each article a precise algorithmic description, a reference source code and a demo facility. The companion demo of *Computing an exact Gaussian scale-space* IPOL publication is available at `http://demo.ipol.im/demo/117/` while the companion demo of the *Anatomy of the SIFT Method* IPOL publication can be found at `http://demo.ipol.im/demo/82/`.

2 Computing an exact Gaussian scale-space

Gaussian convolution is one of the most important algorithms in image processing. This chapter focuses on the computation of the Gaussian scale-space, a family of increasingly blurred images, responsible, among other things, for the scale-invariance of the SIFT method. We discuss and numerically analyze the precision of three different alternatives for defining a discrete counterpart to the continuous Gaussian smoothing operator. This study is focused on low blur levels, that are crucial for the scale-space accuracy.

2.1 Introduction

The Gaussian smoothing operator is one of the most popular tools used in digital image processing. This classic operator has been extensively used, either as a fast pre-process to increase noise robustness before applying another algorithm, or as the fundamental operator in scale-space theory [Iijima et al. 1974; Lindeberg 1993; Sporring et al. 1997; Witkin 1984].

Let $u(\mathbf{x})$ be a continuous image defined for every $\mathbf{x} = (x, y) \in \mathbb{R}^2$. The continuous Gaussian smoothing operator is defined as the convolution operator on $\mathbb{R}^2$ with the isotropic Gaussian function of integral equal to one:

$$G_\sigma u(\mathbf{x}) := \int_{\mathbb{R}^2} G_\sigma(\mathbf{x}')u(\mathbf{x} - \mathbf{x}')d\mathbf{x}', \quad \text{with } G_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma^2} e^{-\frac{|\mathbf{x}|^2}{2\sigma^2}},$$

where the Gaussian kernel is parameterized by its standard deviation σ.

This operator is the cornerstone of several image processing algorithms particularly used for building the scale-space, a multi-scale image representation. The rationale is that the Gaussian function is the only kernel that satisfies the following properties [Alvarez et al. 1993; Babaud et al. 1986; Koenderink 1984; Lindeberg 1993; Witkin 1984; Weickert et al. 1999]:

1. **Linearity.** $G_\sigma(\lambda u(\mathbf{x}) + \mu v(\mathbf{x})) = \lambda G_\sigma u(\mathbf{x}) + \mu G_\sigma v(\mathbf{x})$ for any real λ, μ;

2. **Shift invariance.** If $T_{\boldsymbol{\tau}} u(\mathbf{x}) := u(\mathbf{x} - \boldsymbol{\tau})$ denotes the translation of parameter $\boldsymbol{\tau}$, then $G_\sigma(T_{\boldsymbol{\tau}} u)(\mathbf{x}) = T_{\boldsymbol{\tau}}(G_\sigma u)(\mathbf{x})$;

3. **Scale invariance.** If $H_\lambda u(\mathbf{x}) := u(\lambda \mathbf{x})$ denotes an expansion by a factor λ^{-1}, then $G_\sigma(H_\lambda u)(\mathbf{x}) = H_\lambda(G_{\sigma'} u)(\mathbf{x})$ with $\sigma' = \lambda\sigma$;

4. **Rotation invariance.** If $R_\theta u(\mathbf{x}) := u(R_\theta \mathbf{x})$ denotes the rotation of angle $-\theta$, then $G_\sigma(R_\theta)u(\mathbf{x}) = R_\theta(G_\sigma)u(\mathbf{x})$;

5. **Non negativity.** $G_\sigma(\mathbf{x}) \geq 0,\ \forall(\mathbf{x}, \sigma) \in \mathbb{R}^2 \times \mathbb{R}_+$;

6. **Semi-group.** $G_{\sigma_2}(G_{\sigma_1}u)(\mathbf{x}) = G_{\sqrt{\sigma_1^2+\sigma_2^2}}u(\mathbf{x})$.

Additionally, it can be easily checked that if u is continuous and bounded, then $(\sigma, \mathbf{x}) \mapsto G_\sigma u(\mathbf{x})$ is the solution of the heat diffusion equation $\partial v/\partial\sigma = \sigma\Delta v$ with initial condition $v(0, \mathbf{x}) = u(\mathbf{x})$ (see Guichard et al., Chapter 2).

What is the discrete counterpart of this continuous operator? Could it be defined to satisfy the properties of the continuous Gaussian convolution? Despite being central in image processing, the Gaussian convolution is generally crudely approximated by discrete convolutions or even box filters. Numerous algorithms have been proposed for approximating the Gaussian convolution in digital images. We concentrate here on three of the most relevant ones for the accurate computation of the Gaussian scale-space, namely, the Fourier based convolution, the discrete convolution with Gaussian kernel samples and Lindeberg's discrete scale-space smoothing [Lindeberg 1993]. These methods can be described either as approximations of the continuous Gaussian convolution or as linear filters designed to satisfy some of the previously introduced properties expressed in the discrete framework. Other methods, not discussed here, include the use of recursive filters [Young and Van Vliet 1995; Deriche 1993] or the iteration of extended box filters [Gwosdek et al. 2012]. They provide fast and accurate approximations of the Gaussian convolution for large σ values but they crudely approximate the Gaussian kernel for low values of σ (typically $\sigma \leq 1$), making them unsuitable for an accurate computation of the Gaussian scale-space. For a complete survey regarding speed and performance on the Gaussian convolution for large values of σ we refer the reader to [Getreuer 2013].

In this chapter, we propose to use the semi-group property for measuring the accuracy of each of the analyzed methods. In particular, we test if multiple iterations of the same Gaussian convolution produces the same result as a single convolution with blur level foretold by the semi-group property. The conclusions are straightforward. The only method that allows to compute accurately the Gaussian scale-space is the Fourier based convolution. The discrete convolution with samples from a Gaussian kernel is accurate only if the applied blur level is large enough to avoid aliasing artifacts (i.e., $\sigma > 0.8$). Although Lindeberg's smoothing method satisfies the semi-group property, it introduces a bias in the applied amount of blur being significantly lower. Evident though they are, these conclusions may have a strong impact on the conception and performance of algorithms using the Gaussian scale-space.

The rest of the chapter is organized as follows. Section 2.2 introduces the mathematical tools used to justify the Fourier based convolution. Each of the three discussed methods is explained in Section 2.3 where a detailed implementation with a mathematical interpretation is given. In Section 5.5 we present some numerical experiments and we finally conclude in Section 4.7.

2.2 Mathematical Preliminaries

2.2.1 Notations

In the sequel, $u_{k,l} \in \mathbb{R}$ for $k = 0, \ldots, M-1$ and $l = 0, \ldots, N-1$ denote the samples of a digital image of size $M \times N$. By a slight abuse of notation, we will denote this image by $(u_{k,l})$. We denote by $\lfloor \cdot \rfloor$ and $\lceil \cdot \rceil$ the floor and ceiling functions respectively.
The Fourier transform of $f \in L^1(\mathbb{R}^2)$ is the function $\hat{f}$, defined for all $(\xi, \eta) \in \mathbb{R}^2$ by

$$\hat{f}(\xi, \eta) = \int_{\mathbb{R}^2} f(x, y) e^{-i(x\xi + y\eta)} dx dy.$$

The Discrete Fourier Transform (DFT) of $(u_{k,l})$ is defined as the sequence

$$\tilde{u}_{m,n} = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} u_{k,l} e^{-\frac{2i\pi mk}{M}} e^{-\frac{2i\pi nl}{N}}$$

for $m = -\lfloor M/2 \rfloor, \ldots, -\lfloor M/2 \rfloor + M - 1$ and $n = -\lfloor N/2 \rfloor, \ldots, -\lfloor N/2 \rfloor + N - 1$.

The Inverse Discrete Fourier Transform (IDFT) of $(\tilde{u}_{m,n})$ with $m = -\lfloor M/2 \rfloor, \ldots, -\lfloor M/2 \rfloor + M - 1$ and $n = -\lfloor N/2 \rfloor, \ldots, -\lfloor N/2 \rfloor + N - 1$ is defined as the sequence

$$u_{k,l} = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{(-\lfloor \frac{M}{2} \rfloor + M - 1)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{(-\lfloor \frac{N}{2} \rfloor + N - 1)} \tilde{u}_{m,n} e^{\frac{2i\pi mk}{M}} e^{\frac{2i\pi nl}{N}}$$

for $k = 0, \ldots, M-1$ and $l = 0, \ldots, N-1$. The DFT and IDFT are inverse transformations: $\mathsf{IDFT} \circ \mathsf{DFT} = \mathsf{Id}$.

2.2.2 DFT and DCT interpolations

A convenient continuous image model is to represent images as trigonometric polynomials, or equivalently, periodic band-limited functions. The limited bandwidth of camera lenses motivates this approach. The periodic extension of the signal is arbitrary, as any other signal extension, but it is particularly convenient for Fourier interpolation.

We will say that P is a bi-dimensional trigonometric polynomial of degrees $\lfloor \frac{M}{2} \rfloor$ and $\lfloor \frac{N}{2} \rfloor$, and periodicities a and b, if and only if

$$P(x, y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{(-\lfloor \frac{M}{2} \rfloor + M - 1)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{(-\lfloor \frac{N}{2} \rfloor + N - 1)} a_{m,n} e^{\frac{2i\pi mx}{a}} e^{\frac{2i\pi ny}{b}},$$

where $a_{m,n} \in \mathbb{C}$ for $m = -\lfloor \frac{M}{2} \rfloor, \ldots, -\lfloor \frac{M}{2} \rfloor + M - 1$ and $n = -\lfloor \frac{N}{2} \rfloor, \ldots, -\lfloor \frac{N}{2} \rfloor + N - 1$. The following proposition, characterizes the polynomial coefficients that satisfy an interpolation criterion.

Proposition 1. *(The DFT Interpolation) There exists a unique trigonometric polynomial u of degrees $\lfloor \frac{M}{2} \rfloor$ and $\lfloor \frac{N}{2} \rfloor$, and of periodicities a and b, that satisfies the interpolation condition*

$$u\left(k\frac{a}{M}, l\frac{b}{N}\right) = u_{k,l}, \quad \text{for } k = 0, \ldots, M-1 \text{ and } 0, \ldots, N-1,$$

namely

$$u(x,y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\left(-\lfloor \frac{M}{2} \rfloor + M - 1\right)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{\left(-\lfloor \frac{N}{2} \rfloor + N - 1\right)} \tilde{u}_{m,n} e^{\frac{2i\pi mx}{a}} e^{\frac{2i\pi ny}{b}}$$

where the polynomial coefficients $\tilde{u}_{m,n}$ are computed by the DFT *of* $(u_{k,l})$.

From now on, we will consider without loss of generality that $a = M$ and $b = N$. This can be fulfilled by an appropriate parameterization of $\mathbb{R}^2$.

DCT interpolation

When manipulating images through the DFT interpolation, the digital image is implicitly extended to $\mathbb{Z}^2$ via periodization. This eventually leads to strong discontinuities at image borders. The discrete cosine transform (DCT) interpolation reduces the discontinuities caused by the brutal periodization by first symmetrizing the image.

The DCT interpolation of the digital image $(u_{k,l})$ of size $M \times N$ is equivalent to the DFT interpolation of the symmetrized signal $(\mathring{u}_{k,l})$ of size $2M \times 2N$ where $\mathring{u}_{k,l} = u_{s_M(k),s_N(l)}$ with $s_M(k) = \min(k, 2M - 1 - k)$ for $0 \leq k \leq 2M - 1$ and $s_N(l)$ is defined similarly. The DFT interpolation (Proposition 1) applied to this particular case defines the DCT interpolation as the only trigonometric polynomial $\mathring{u}$ of degrees M and N and of periodicities $2M$ and $2N$ that interpolates exactly the symmetrized image

$$\mathring{u}\,(k,l) = \mathring{u}_{k,l}, \quad \text{for } k = 0, \ldots, 2M-1 \text{ and } l = 0, \ldots, 2N-1$$

namely,

$$\mathring{u}(x,y) = \sum_{m=-M}^{M-1} \sum_{n=-N}^{N-1} \widetilde{\mathring{u}}_{m,n} e^{\frac{i\pi mx}{M}} e^{\frac{i\pi ny}{N}}.$$

Thus, the DCT interpolation can be expressed as

$$\mathring{u}(x,y) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \alpha_m \alpha_n \mathsf{DCT}(u)_{m,n} \cos\left(\frac{\pi(x + 1/2)m}{M}\right) \cos\left(\frac{\pi(y + 1/2)n}{N}\right),$$

with $\alpha_m = 1/2$ when $m = 0$ and $\alpha_m = 1$ elsewhere, and with $(\mathsf{DCT}(u)_{m,n})$ denoting the type-II DCT coefficients of image $(u_{k,l})$ defined for $m = 0, \ldots, M-1$ and $n = 0, \ldots, N-1$ by

$$\mathsf{DCT}(u)_{m,n} = \frac{1}{MN} \sum_{l=0}^{M-1} \sum_{k=0}^{N-1} u_{k,l} \cos\left(\frac{\pi(k + 1/2)m}{M}\right) \cos\left(\frac{\pi(l + 1/2)n}{N}\right).$$

Since $\mathring{u}(k,l) = u_{k,l}$ for $k = 0, \ldots, M-1$ and $l = 0, \ldots, N-1$, the inverse DCT transform IDCT, is computed by

$$\mathsf{IDCT}(u)_{k,l} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \alpha_m \alpha_n u_{m,n} \cos\left(\frac{\pi(k + 1/2)m}{M}\right) \cos\left(\frac{\pi(l + 1/2)n}{N}\right),$$

with $\alpha_m = 1/2$ when $m = 0$ and $\alpha_m = 1$ elsewhere.

2.2.3 The convolution theorem

Let $u_{k,l} \in \mathbb{R}$ for $k = 0, \ldots, M-1$ and $l = 0, \ldots, N-1$ be a real-valued digital image of size $M \times N$, and let $u(x, y)$ be its DFT interpolation,

$$u(x,y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\left(-\lfloor \frac{M}{2} \rfloor + M - 1\right)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{\left(-\lfloor \frac{N}{2} \rfloor + N - 1\right)} \tilde{u}_{m,n} e^{\frac{2i\pi mx}{M}} e^{\frac{2i\pi ny}{N}}$$

where $(\tilde{u}_{m,n}) = \mathsf{DFT}((u_{k,l}))$. The following theorem states that the convolution of the DFT interpolation of a digital image with a linear filter can be computed exactly by properly weighting its DFT coefficients. This result plays an important role in the present framework as the link between the continuous image model and the discrete computations that in practice we are able to compute.

Theorem 1. *The convolution of the trigonometric polynomial*

$$u(x,y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\left(-\lfloor \frac{M}{2} \rfloor + M - 1\right)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{\left(-\lfloor \frac{N}{2} \rfloor + N - 1\right)} \tilde{u}_{m,n} e^{\frac{2i\pi mx}{M}} e^{\frac{2i\pi ny}{N}}$$

with a function $f \in L^1(\mathbb{R}^2)$, is the trigonometric polynomial

$$f * u(x,y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\left(-\lfloor \frac{M}{2} \rfloor + M - 1\right)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{\left(-\lfloor \frac{N}{2} \rfloor + N - 1\right)} \tilde{u}_{m,n} \widehat{f}\left(\frac{2\pi m}{M}, \frac{2\pi n}{N}\right) e^{\frac{2i\pi mx}{M}} e^{\frac{2i\pi ny}{N}}.$$

Proof. Let us consider the pure wave of frequency $\xi \in \mathbb{R}^2$, $g_\xi(\mathbf{x}) = e^{i\xi \cdot \mathbf{x}}$. Then,

$$f * g_\xi(\mathbf{x}) = \int_{\mathbb{R}^2} f(\mathbf{x}') g_\xi(\mathbf{x} - \mathbf{x}') d\mathbf{x}' = e^{i\xi \cdot \mathbf{x}} \int_{\mathbb{R}^2} f(\mathbf{x}') e^{-i\xi \cdot \mathbf{x}'} d\mathbf{x}' = \widehat{f}(\xi) g_\xi(\mathbf{x}),$$

which is a pure wave of the same frequency. The result follows from the linearity of convolution. □

This theorem can be extended to the DCT interpolation by considering the symmetrized $2M \times 2N$ image.

2.3 Analysis of three digital Gaussian convolution algorithms

Several algorithms have been proposed for the Gaussian convolution in digital images. This chapter looks at three of them, the Fourier based convolution, the discrete convolution with samples from the Gaussian kernel and Lindeberg's discrete scale-space smoothing. In what follows we describe each of these algorithms.

2.3.1 DFT convolution

Since only a finite set of the image values are known, it is not possible to directly compute the continuous Gaussian convolution. However, if for example, we accept that the image is band-limited and periodic, then we can fully recover the image values at the continuous domain. This is done by the DFT interpolation presented in Proposition 1. Moreover, if we accept these convenient hypotheses, the continuous Gaussian convolution can be computed exactly in the Fourier domain at the cost of two DFTs and one operation per pixel, as indicated by the convolution theorem (Theorem 1).

Remark. The Fourier transform of the isotropic Gaussian kernel $G_\sigma(x, y)$ with standard deviation σ is $\widehat{G}_\sigma(\xi, \eta) = e^{-\frac{\sigma^2}{2}(\xi^2+\eta^2)}$.
Applying Theorem 1 to the Gaussian kernel, the continuous Gaussian convolution of the DFT interpolation $u(x, y)$ of digital image $(u_{k,l})$ is

$$G_\sigma * u(x, y) = \sum_{m=-\lfloor \frac{M}{2} \rfloor}^{\left(-\lfloor \frac{M}{2} \rfloor + M - 1\right)} \sum_{n=-\lfloor \frac{N}{2} \rfloor}^{\left(-\lfloor \frac{N}{2} \rfloor + N - 1\right)} \tilde{u}_{m,n} \widehat{G}_\sigma \left(\frac{2\pi m}{M}, \frac{2\pi n}{N} \right) e^{\frac{2i\pi mx}{M}} e^{\frac{2i\pi ny}{N}},$$

where $\widehat{G}_\sigma(\frac{2\pi m}{M}, \frac{2\pi n}{N}) = e^{-\frac{\sigma^2\pi^2}{2}\left(\left(\frac{2m}{M}\right)^2 + \left(\frac{2n}{N}\right)^2\right)}$.

A description of the method is presented in Algorithm 1 while Figures 2.1 and 2.2 provide illustrations of the algorithm behavior in the image and Fourier domains.

Since this algorithm implements the continuous Gaussian convolution (assumed an underlying continuous image model), all properties of the continuous Gaussian convolution are verified. However, using the DFT interpolation amounts to implicitly assuming that the digital image originates from sampling a band-limited periodic function below the Nyquist rate. The assumption that the image is well sampled is often unrealistic. Although for natural images, the low-pass filter behavior of digital cameras justifies the assumption of an underlying band-limited function, the frequency band will not be necessarily the same as the one covered by the sampling. The periodic assumption is unnatural. The forced periodization can lead to strong discontinuities at image borders (see Figure 2.1), which contradicts in some extent the band limited-assumption, causing ringing. These artifacts can be reduced by using the DCT variant.

DCT convolution

Since the DCT interpolation of an $M \times N$ image is equivalent to the DFT interpolation of the $2M \times 2N$ mirror symmetrized image, Theorem 1 can be reformulated in terms of DCT sequences. The underlying continuous image model is then a trigonometric polynomial of degrees M and N and periodicities $2M$ and $2N$. The continuous convolution of the DCT interpolation $u(x, y)$ of the digital image $(u_{k,l})$ with a Gaussian kernel of standard deviation σ is

$$G_\sigma * u(x, y) = \sum_{m=-M}^{M-1} \sum_{n=-N}^{N-1} \mathsf{DCT}(u)_{m,n} \widehat{G}_\sigma \left(\frac{\pi m}{M}, \frac{\pi n}{N} \right) e^{\frac{i\pi mx}{M}} e^{\frac{i\pi ny}{N}},$$

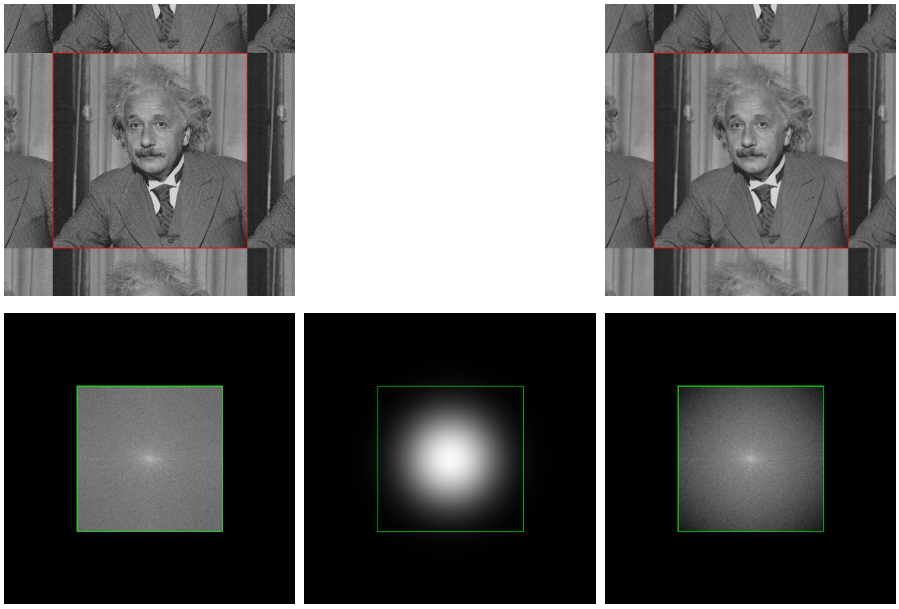


Figure 2.1: Illustrating the continuous Gaussian convolution through DFT interpolation. The first column illustrates the adopted continuous image model. The image is defined on the $\mathbb{R}^2$ plane. It is periodic $u(x+kM, y+lN) = u(x,y)$ (top) and band-limited with $\text{supp}(\widehat{u}) \in [-\pi,\pi]^2$ (bottom). The borders of the digital image are represented by a red box. The green box indicates $[-\pi,\pi]^2$, the domain relative to the sampling. The domain is extended to illustrate that the continuous image is periodic and band-limited. The second column illustrates the Fourier transform of the Gaussian kernel defined on the $\mathbb{R}^2$ plane for $\sigma = 0.8$. The convolution in the spatial domain is equivalent to a multiplication in the Fourier domain. The last column illustrates the periodic trigonometric polynomial relative to this Gaussian convolution (top) and a representation of the Dirac amplitudes in its Fourier transform (bottom). It is a classic convention, also adopted here, that a continuous digital image is displayed by showing a constant value on each pixel, equal to its sampled value at the pixel center. Thanks to the optical blur of the screen and of our vision, the resulting visual is a decent representation of the smooth ideal image.

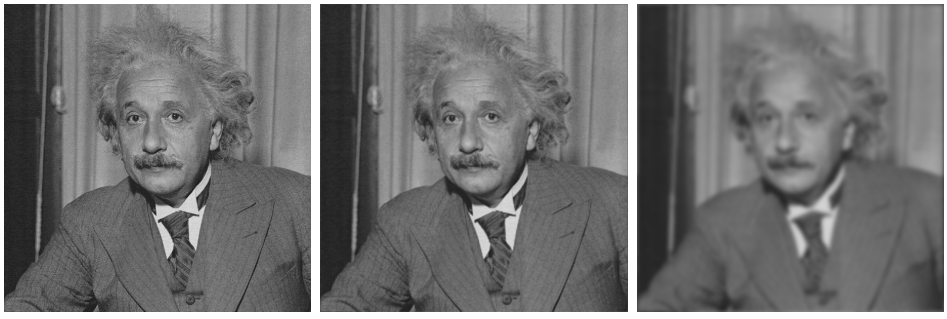
where $\widehat{G}_\sigma(\frac{\pi m}{M}, \frac{\pi n}{N}) = e^{-\frac{\sigma^2\pi^2}{2}\left(\left(\frac{m}{M}\right)^2+\left(\frac{n}{N}\right)^2\right)}$. The complexity of the process is reduced by using the symmetry of the DCT coefficients. The continuous Gaussian convolution involves weighting the $M \times N$ type-II DCT coefficients. The algorithm is detailed in Algorithm 2. Figure 2.3 shows a Gaussian DCT smoothing on a grayscale image. This figure illustrates how the artifacts induced by the implicit DFT periodization are removed with the DCT variant.

2.3.2 Sampled Gaussian kernel convolution.

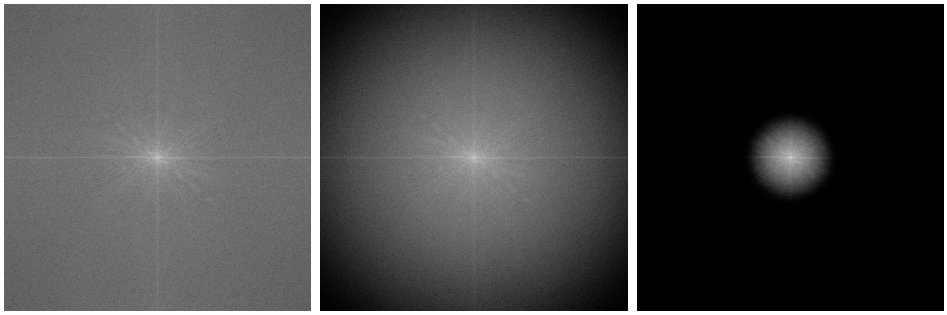
The most common discrete approximation of the Gaussian convolution is obtained by sampling the truncated continuous Gaussian kernel. Although being straightforward to implement, as we will show it does not satisfy the semi-group property for small σ values.

The continuous 1D Gaussian kernel of standard deviation σ is truncated at width $2K\sigma$; typical values of K are 3 or 4 to gather most of the signal's energy. Then, it is sampled to produce the discrete filter $(g_k)_k$ of width $2\lceil K\sigma\rceil + 1$ and normalized to sum to 1,

$$g_k = Se^{-\frac{k^2}{2\sigma^2}}, \quad k = -\lceil K\sigma\rceil, \ldots, \lceil K\sigma\rceil \quad \text{and} \sum g_k = 1.$$



Input Image $\sigma = 1.0$ $\sigma = 3.0$



Fourier spectrum modulus $\sigma = 1.0$ $\sigma = 3.0$

Figure 2.2: Grayscale input image and the results of applying the DFT Gaussian convolution with parameter $\sigma = 1.0, 3.0$. In the bottom row, the respective moduli of the Fourier spectra show the attenuation of high frequencies. The high values in the Fourier spectra along the vertical and horizontal axes are caused by the strong discontinuities when periodizing the image.

The convolution with the sampled Gaussian kernel algorithm [Getreuer 2013] (detailed in Algorithm 3) consists in the computation of the separable 2D discrete convolution

$$v_{k,l} = \sum_{k'=-\lceil K\sigma \rceil}^{\lceil K\sigma \rceil} g_{k'} \sum_{l'=-\lceil K\sigma \rceil}^{\lceil K\sigma \rceil} g_{l'} \; u'_{k-k',l-l'},$$

where u' denotes the extension of u to the $\mathbb{Z}^2$ plane either by (M, N)-periodization or symmetrization followed by $(2M, 2N)$-periodization. Formally

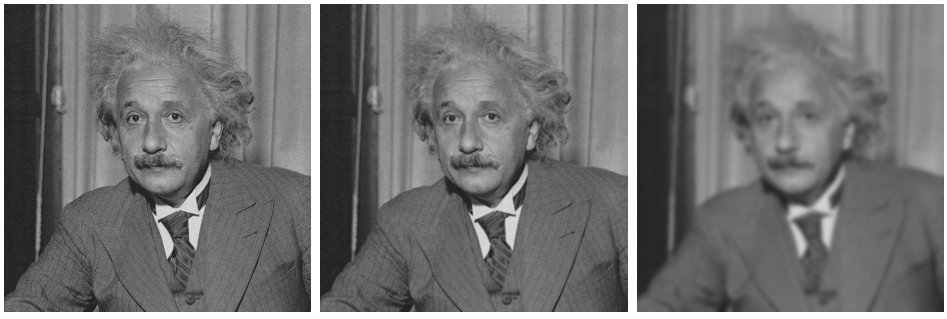
$$u'(k, l) = u(s_M(k), s_N(l)) \quad \text{with } s_M(k) = k \bmod M$$

for periodic extension, or

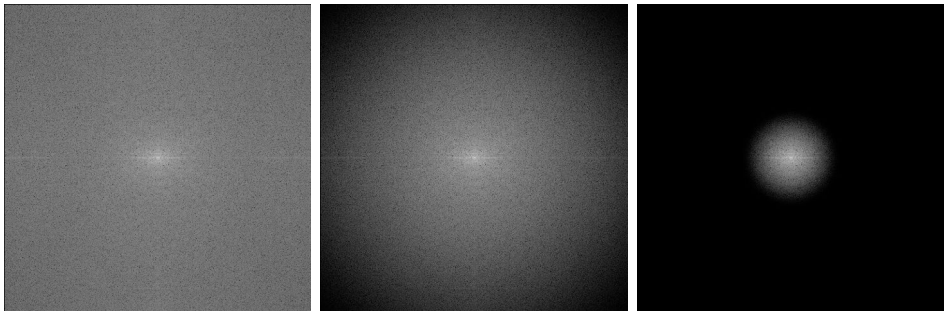
$$s_M(k) = \min(k \bmod 2M, 2M - 1 - (k \bmod 2M))$$

in case of prior symmetrization with respect to $-1/2$. The pseudocode in Algorithm 3 incorporates the symmetric extension of the signal, the modification for the periodic extension is straightforward.

Gaussian kernel aliasing and semi-group property. The Fourier transform of the Gaussian function G_σ, $\widehat{G}_\sigma(\xi, \mu) = e^{-\sigma^2 \frac{\xi^2+\mu^2}{2}}$ has no compact support. Thus, the



Input Image $\sigma = 1.0$ $\sigma = 3.0$



Fourier spectrum modulus $\sigma = 1.0$ $\sigma = 3.0$

Figure 2.3: Grayscale input image and the results of applying the DCT Gaussian convolution with parameter $\sigma = 1.0, 3.0$. In the bottom row, the respective moduli of the Fourier spectra show the attenuation of high frequencies similar to the case of DFT convolution. The main difference is that in the DCT Gaussian smoothing, the implicit symmetrization of the image avoids the strong discontinuities when periodizing.

sampling of the Gaussian kernel never satisfies the band-limited assumption needed by the Nyquist-Shannon sampling theorem (see e.g., Gasquet and Witomski [1999]). Since the value of $\widehat{G}_\sigma(\xi, \mu)$ at the Nyquist frequency is $e^{-\pi^2\sigma^2/2}$, the aliasing is not significant for $\sigma > 1$ [Morel and Yu 2011]. As we will show in the numerical experiments in Section 5.5, the aliasing of the Gaussian kernel contributes to the lack of semi-group property.

Truncation error. The error due to kernel truncation is shown in Figure 2.4. The error is very small for large enough values of K (for instance, the error is less than 10^{-4} for $K \geq 4$). The truncation at $\lceil K\sigma \rceil$ also induces oscillations on the spectrum. If $(v_{k,l})$ and $(v_{k,l}^{\text{trunc}})$ denote the respective outputs of the convolutions of u with the infinite and the truncated versions of the sampled Gaussian kernel, then their DFT coefficients are related by

$$\hat{v}_{m,n}^{\text{trunc}} = \sum_{m'=0}^{M-1} \sum_{n'=0}^{N-1} \hat{v}_{m',n'} D_{\lceil K\sigma \rceil}(2\pi(m-m')/M) D_{\lceil K\sigma \rceil}(2\pi(n-n')/N)$$

for $m = -\lfloor M/2 \rfloor, \ldots, -\lfloor M/2 \rfloor + M - 1$ and $n = -\lfloor N/2 \rfloor, \ldots, -\lfloor N/2 \rfloor + N - 1$ and where D_L denotes the Dirichlet function (also known as the periodic sinc function),

$$D_L(x) = \begin{cases} \frac{\sigma_{\text{in}}(L+1/2)x)}{\sigma_{\text{in}}(x/2)} & \text{if } x \neq 2k\pi, k \in \mathbb{Z} \\ \\ (-1)^{k(L-1)} & \text{if } x = 2k\pi, k \in \mathbb{Z}. \end{cases}$$

The oscillating spectrum due to the convolution with the Dirichlet kernel is noticeable for small values of K, as can be seen in Figure 2.5.

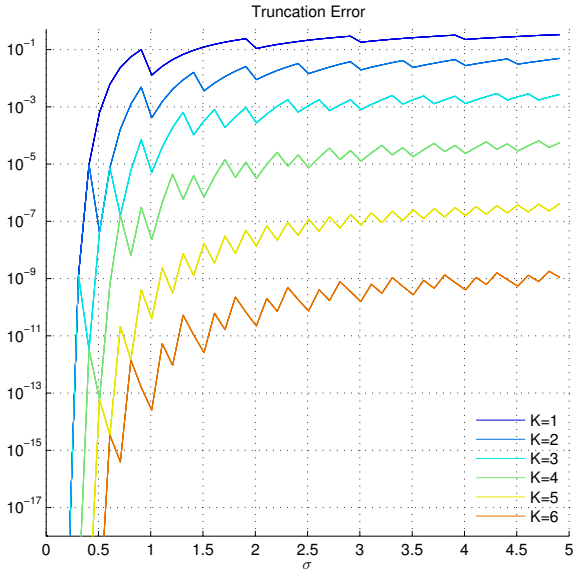


Figure 2.4: Impact of Gaussian kernel truncation. The sampled 1D Gaussian kernel g_k is truncated at $\lceil K\sigma \rceil$ samples. The truncation error is defined as the square root of the total energy loss, $\sqrt{\sum_{|k|>\lceil K\sigma \rceil} g_k^2}$, divided by the square root of the total energy of the Gaussian kernel, $\sqrt{\sum_k g_k^2}$. We have approximated the total energy of the Gaussian kernel by $\sum_{|k|\leq\lceil 50\sigma \rceil} g_k^2$. The truncation error decays rapidly with K and becomes smaller than 10^{-4} for $K \geq 4$.

2.3.3 Lindeberg's discrete scale-space smoothing

Let u be a continuous and bounded signal, then the Gaussian convolution $v : \sigma \mapsto G_\sigma u$ is the solution of the heat equation $\frac{\partial v}{\partial \sigma} = \sigma \Delta v$ with initial condition $v(0, \mathbf{x}) = u(\mathbf{x})$. Equivalently, thanks to the re-parameterization $t = \sigma^2/2$, $v : t \mapsto G_{\sqrt{2t}} u$ is the solution of the equation $\frac{\partial v}{\partial t} = \Delta v$. Lindeberg's smoothing method [Lindeberg 1993] is based on computing the solution of a spatial discretization of the equation $\frac{\partial v}{\partial t} = \Delta v$.

For a one-dimensional sequence $(u_k)_{k\in\mathbb{Z}}$, Lindeberg's smoothing method consists in finding $v_k(t)$ solution of

$$\partial_t v_k(t) = \Delta^{\text{discr}} v_k(t), \quad \text{with} \quad v_k(0) = u_k,$$

where $\Delta^{\text{discr}} v_k(t)$ denotes the 1D Laplacian finite difference scheme $\Delta^{\text{discr}} v_k = v_{k-1} - 2v_k + v_{k+1}$. This solution can be computed via a discrete convolution with the discrete sequence

$$g_n^{\text{Lindeberg}} = e^{-t} I_n(t),$$

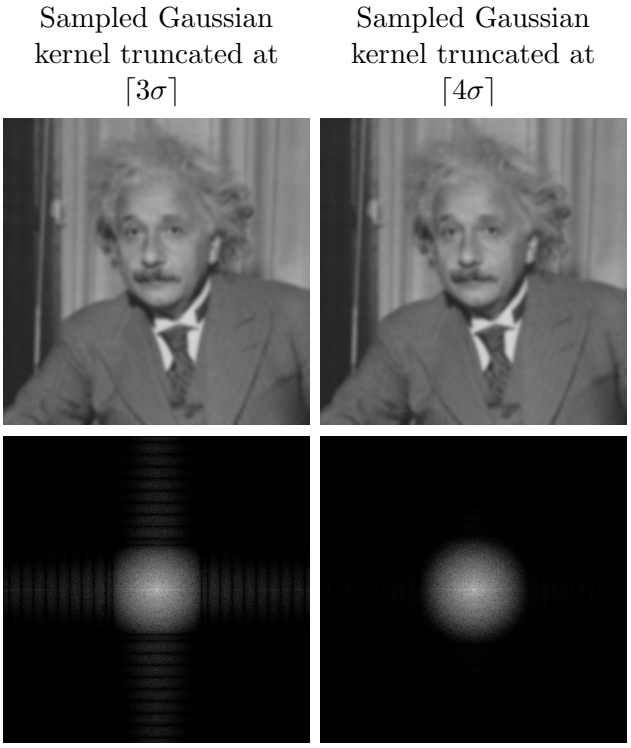


Figure 2.5: Impact of kernel truncation. In the first row, the convolutions of a test image with sampled Gaussian kernel of standard deviation $\sigma = 4$ truncated at $\lceil 3\sigma \rceil$ (left) and $\lceil 4\sigma \rceil$ (right). Second row, the respective image spectra. Notice the oscillations for a truncation at $\lceil 3\sigma \rceil$ of the sampled Gaussian kernel.

where $t = \sigma^2/2$ and I_n denotes the modified Bessel functions.

For a two-dimensional signal $(u_{k,l})_{(k,l)\in\mathbb{Z}^2}$, Lindeberg's smoothing method consists in solving

$$\partial_t v_{k,l}(t) = \Delta_\gamma^{\text{discr}} v_{k,l}(t), \quad \text{with} \quad v_{k,l}(0) = u_{k,l},$$

where $\Delta_\gamma^{\text{discr}}$ denotes the following 2D Laplacian finite difference scheme

$$\Delta_\gamma^{\text{discr}} u = (1-\gamma)\Delta_+ u + \gamma \Delta_\times u,$$

with

$$\begin{aligned}
\Delta_+ u_{k,l} &= u_{k+1,l} + u_{k-1,l} + u_{k,l+1} + u_{k,l-1} - 4u_{k,l}, \\
\Delta_\times u_{k,l} &= 1/2(u_{k+1,l+1} + u_{k+1,l-1} + u_{k-1,l+1} + u_{k-1,l-1}) - 2u_{k,l},
\end{aligned}$$

for $0 \leq \gamma \leq 1/2$. The parameter γ controls the shape of the Laplacian discrete operator. [1] The smoothed image is computed by Euler's method. This explicit time marching scheme consists in applying the following iteration formula

$$\frac{v(p\delta t)_{k,l} - v((p-1)\delta t)_{k,l}}{\delta t} = \Delta_\gamma^{\text{discr}} v((p-1)\delta t)_{k,l}$$

[1] For a thorough analysis of the influence of parameter γ on isotropy, we refer the interested reader to Lindeberg [1993] pp. 127-134.

for $1 \leq p \leq P$ with δt the step size and P the total number of iterations (i.e., $P\delta t = \sigma^2/2$). The stability of Euler's method is guaranteed if the step size satisfies $\delta t < 1/8(1 - \gamma/2\sigma)$. The implementation of Lindeberg's smoothing method is detailed in Algorithm 4.

2.4 Experiments

Let us assume that the Gaussian semi-group property is valid. Then, applying N times a Gaussian filter of parameter σ should produce the same result as filtering only once with a Gaussian function of parameters $\sqrt{N}\sigma$. This allows us to evaluate the validity of the semi-group property for all the described methods.

Indeed, if an image of a Gaussian kernel is filtered by a Gaussian function of a given standard deviation, the filtered signal should be a Gaussian function of a standard deviation given by the semi-group property. Thus, the following experiment was carried out. A sampled Gaussian function of standard deviation σ_{in} was considered as the input signal. It was filtered N times by each of the different Gaussian filters implementations with parameter σ. A Gaussian function was fitted to the filtered image by least squares. The estimated standard deviation was compared to the theoretical expected value $\sqrt{\sigma_{\text{in}}^2 + N\sigma^2}$. The input Gaussian standard deviation was set to $\sigma_{\text{in}} = 1.0$ to avoid aliasing artifacts, and the number of iterations N was set to 10.

The results are shown in Figures 2.6 to 2.9. Each figure shows the estimated blurs, the differences between estimated and theoretical values, and the root-mean-square error between the pixels of the two filtered images.

The experiment demonstrates that the DFT convolution (Figure 2.6) and its DCT variant (Figure 2.7) fully satisfy the semi-group property with machine precision error [2]. Figure 2.8 shows the previous experiment for the sampled Gaussian kernel truncated at $\lceil 5\sigma \rceil$. For low values of σ, the estimated blur level deviates from the theoretical value $\sqrt{N}\sigma$ and the method fails to satisfy the semi-group property. This is due to the aliasing in the sampled Gaussian kernel. The difference with respect to the theoretical values is less than 10^{-3} for $\sigma \geq 0.8$. Applying Lindeberg's method consists in solving a discretized version of the heat equation. The parameter γ which defines the Laplacian finite difference scheme is set here to $\gamma = 1/2$. Lindeberg's smoothing method satisfies the semi-group property (Figure 2.9) but the estimated blur is lower than the theoretical value.

Additionally, direct and iterated convolutions were applied on a test image. For all four methods, the RMSE between the direct and the iterated convolutions [3] is displayed in Figure 2.10, while Figure 2.11 shows the image difference. The DCT and DFT convolution produce the lowest errors. Nevertheless, the sampled Gaussian kernel and Lindeberg's method give similar errors for large values of σ (i.e., $\sigma \geq 0.9$).

2.5 Conclusion

In this chapter we analyzed three of the most commonly used methods for the Gaussian smoothing of digital images. We focused on methods most commonly used for the com-

[2] The algorithms are implemented using single-precision float data type

[3] Root-mean square error. For (x_n) and (y_n) with $n = 1 \ldots N$, $\texttt{rmse}((x_n), (y_n)) = \frac{1}{N}\sqrt{\sum_{n=1}^{N}(x_n - y_n)^2}$.

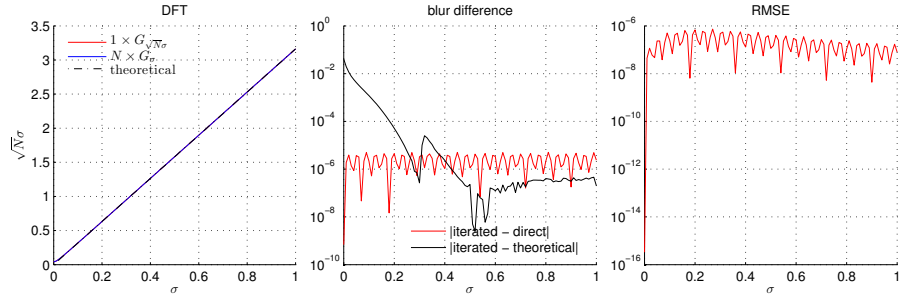


Figure 2.6: DFT convolution. A Gaussian convolution of parameter $\sqrt{N}\sigma$ is compared to $N = 10$ iterations of a Gaussian convolution of parameter σ (denoted $N \times G_\sigma$) for different values of σ. On the left, the estimated blur levels for the direct and iterated filters are plotted as a function of σ. The theoretical value $\sqrt{N}\sigma$ is plotted in black. On the center, the difference between the estimated blur levels for direct and iterated filters as a function of σ is plotted in red. This difference is below 10^{-5} which indicates that the DFT method satisfies the semi-group property. The difference between the estimated blur level in the iterated filtered image and the theoretical blur level as a function of σ is plotted in black. The DFT convolution is accurate since this difference is below 10^{-3} for $\sigma \geq 0.1$ and is below 10^{-6} for $\sigma \geq 0.4$. On the right, the root-mean-square error (RMSE) between the pixel values of both filtered images confirms the DFT consistency regarding the semi-group property.

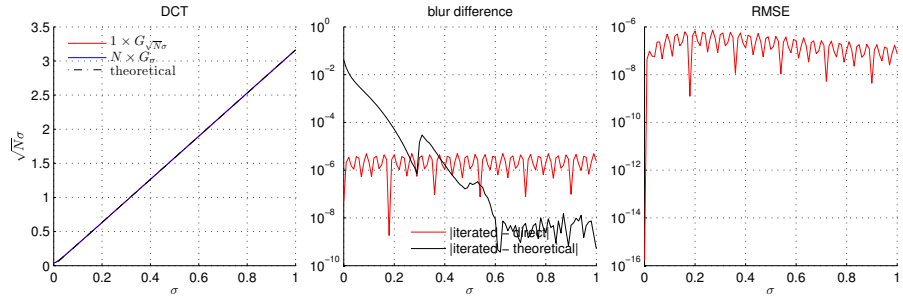


Figure 2.7: The DCT convolution of an image is the DFT convolution after symmetrization of the image. Unsurprisingly, the semi-group property is satisfied by this variant.

putation of the Gaussian scale-space. We have detailed their implementation as well as an analysis of how they differ from the continuous Gaussian convolution.

Computing the Gaussian scale-space with high precision requires an accurate implementation of the Gaussian convolution for low blur levels. With that aim, we focused on the accuracy at low levels of Gaussian blur (i.e., $\sigma \leq 1$).

The DFT and DCT Gaussian convolutions fully satisfy the semi-group property, thus giving an accurate discrete implementation of the continuous Gaussian convolution. The discrete convolution with samples from a Gaussian kernel also satisfies the semi-group property for large applied blur (i.e., $\sigma > 0.8$). However, the aliasing of the sampled kernel for low blur levels makes it unsuitable for accurate computations of the Gaussian scale-pace. Finally, although Lindeberg's smoothing method satisfies the semi-group property, it introduces a bias in the applied amount of blur.

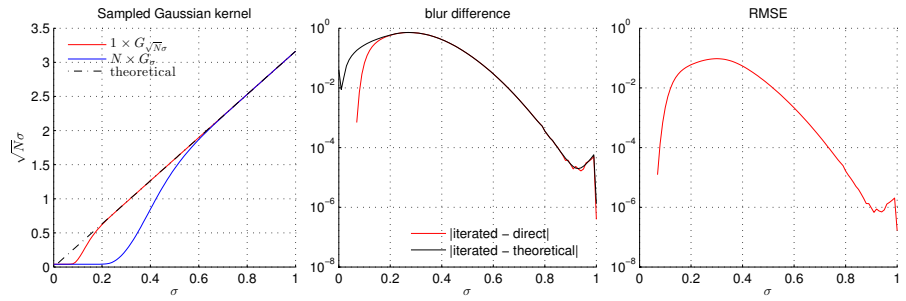


Figure 2.8: Sampled Gaussian kernels truncated at $\lceil 5\sigma \rceil$. A convolution of parameter $\sqrt{N}\sigma$ is compared to $N = 10$ iterations of a filter of parameter σ for the range $0 \leq \sigma \leq 1$. On the left, the estimated blur levels for the direct and iterated filters are plotted for different values of σ. The theoretical value $\sqrt{N}\sigma$ is plotted in black. The center and right plots show the blur level difference and the RMSE of the filtered images respectively. For low values of σ, the estimated blur after N convolutions is lower that the theoretical value. Indeed, in this case, the sampled kernels are aliased and the method does not satisfy the semi-group property. This is confirmed by a blur difference above 10^{-2} for $\sigma \leq 0.6$ (red curve, center plot). The difference with respect to the theoretical values is less than 10^{-3} for $\sigma \geq 0.8$. For very low values of σ (e.g., $\sigma \approx 0.2$), the measured blur is null. This is reasonable since in this case, the sampled kernel is reduced to a sequence with only one nonzero coefficient.

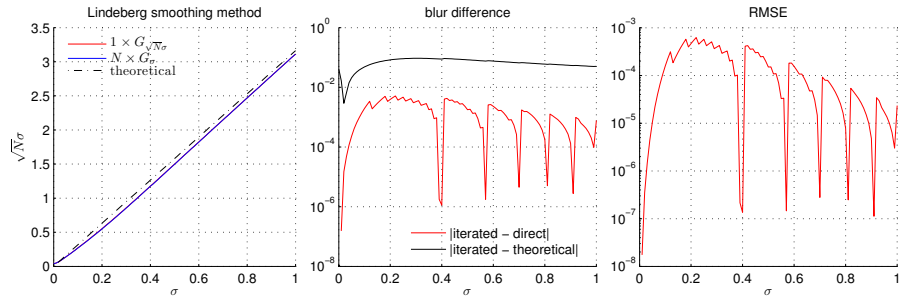


Figure 2.9: Lindeberg's smoothing method. A smoothing of parameter $\sqrt{N}\sigma$ is compared to $N = 10$ iterations of the method with parameter σ (denoted $N \times G_\sigma$) for the range $0 \leq \sigma \leq 1$. The method consists in the resolution of a discretized version of the heat equation. The experiment demonstrates that Lindeberg's smoothing method satisfies the semi-group property. The two measured blurs are almost identical (difference around 10^{-3}, see center plot). However, the estimated blur is lower than the theoretical value of $\sqrt{N}\sigma$.

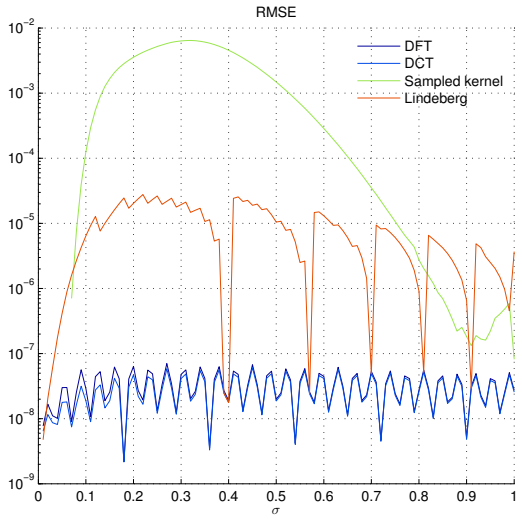


Figure 2.10: Validity of the semi-group property on a natural image (`portrait`) for DFT and DCT convolutions, convolution with sampled Gaussian kernels and Lindeberg's smoothing method. The RMSE between a convolution of parameter $\sqrt{N}\sigma$ and $N = 10$ iterations of a Gaussian filtering of parameter σ is plotted as a function of $0 \leq \sigma \leq 1$. DCT and DFT produce the lowest errors, followed by Lindeberg's method. For $\sigma \geq 0.9$ the RMSE produced with the sampled Gaussian kernel is similar to those produced by Lindeberg's method.

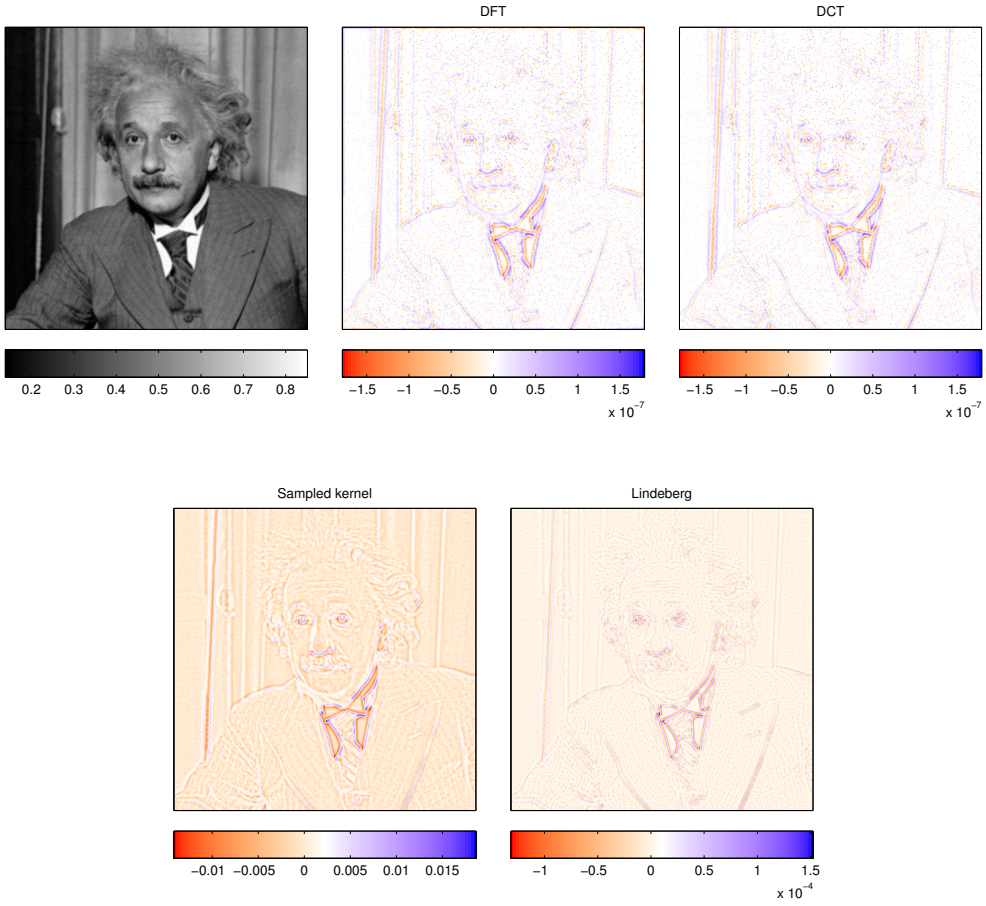


Figure 2.11: Image difference between direct and iterated convolutions for the four studied algorithms applied on the test image `portrait`. For the DFT and DCT convolutions, for sampled Gaussian kernel and Lindeberg's method, the smoothing parameter σ is set to 0.5 for each iterated filtering and to $0.5\sqrt{10}$ for direct filtering. The methods based on Fourier and Lindeberg's method are consistent with the semi-group property. The measured RMSE between direct and iterated convolution are 7.81×10^{-3} (DFT and DCT), 6.29 (sampled kernel) and 5.90×10^{-2} (Lindeberg). The DFT and DCT methods achieve machine precision.

Pseudocodes

Algorithm 1: DFT convolution.

Inputs: - u, input digital image of $M \times N$ pixels.
- σ, standard deviation of the Gaussian kernel.

Output: v, output image of $M \times N$ pixels.

`//Compute the DFT coefficients of` u
$(\widetilde{u}_{m,n}) \leftarrow \mathsf{DFT}(u_{k,l})$

`//Weight the DFT coefficients`
for $-\lfloor \frac{M}{2} \rfloor \leq m \leq -\lfloor \frac{M}{2} \rfloor + M - 1$ **and** $-\lfloor \frac{N}{2} \rfloor \leq n \leq -\lfloor \frac{N}{2} \rfloor + N - 1$ **do**

$$\widetilde{v}_{m,n} \leftarrow \widetilde{u}_{m,n} \widehat{G}_\sigma \left(\frac{2\pi m}{M}, \frac{2\pi n}{N} \right) = \widetilde{u}_{m,n} e^{-\frac{\sigma^2 \pi^2}{2} \left(\left(\frac{2m}{M}\right)^2 + \left(\frac{2n}{N}\right)^2 \right)}$$

`//Compute Inverse discrete Fourier transform of` $\widetilde{v}$
$(v_{k,l}) \leftarrow \mathsf{IDFT}(\widetilde{v}_{m,n})$

return v

Algorithm 2: DCT convolution.

Inputs: - u, input digital image of $M \times N$ pixels.
- σ, standard deviation of the Gaussian kernel.

Output: v, output image of $M \times N$ pixels.

Temporary: -$\mathsf{DCT}(u)$, type-II DCT coefficients of the input image, $M \times N$ real coefficients.
-$\mathsf{DCT}(v)$, type-II DCT coefficients of the output image.

`//Compute the DCT coefficients of` u
$(\mathsf{DCT}(u)_{m,n}) \leftarrow \mathsf{DCT}(u_{k,l})$

`//Weight the DCT coefficients`
for $0 \leq m \leq M - 1$ **and** $0 \leq n \leq N - 1$ **do**

$$\mathsf{DCT}(v)_{m,n} \leftarrow \mathsf{DCT}(u)_{m,n} \widehat{G}_\sigma \left(\frac{\pi m}{M}, \frac{\pi n}{N} \right) = \mathsf{DCT}(u)_{m,n} e^{-\frac{\sigma^2 \pi^2}{2} \left(\left(\frac{m}{M}\right)^2 + \left(\frac{n}{N}\right)^2 \right)}$$

`//Compute Inverse discrete cosine transform of` $\mathsf{DCT}(v)$
$(v_{k,l}) \leftarrow \mathsf{IDCT}(\mathsf{DCT}(v)_{m,n})$

return v

Algorithm 3: Convolution with a sampled Gaussian kernel.

Inputs: u, input digital image of $M \times N$ pixels.
σ, standard deviation of the Gaussian kernel
Output: v, output digital image of $M \times N$ pixels.
Parameter: K, the Gaussian kernel is truncated at $-\lceil K\sigma \rceil$ and $\lceil K\sigma \rceil$.
Temporary: w, $M \times N$ image used to store intermediate computations.

`//Sample the truncated Gaussian kernel.`
for $-\lceil K\sigma \rceil \leq k \leq \lceil K\sigma \rceil$ **do** $g_k = e^{-\frac{k^2}{2\sigma^2}}$

`//Normalize the sequence to sum to 1.`
for $-\lceil K\sigma \rceil \leq k \leq \lceil K\sigma \rceil$ **do** $g_k = g_k / (\sum_{k'} g_{k'})$

`//Convolution on columns`
for $0 \leq m \leq M-1$ **and** $0 \leq n \leq N-1$ **do**
  $w_{m,n} \leftarrow \sum_{k=-\lceil K\sigma \rceil}^{\lceil K\sigma \rceil} g_k u_{s_M(m-k),n}$
  with $s_M(m) = \min(m \bmod 2M, 2M - 1 - m \bmod 2M))$

`//Convolution on lines`
for $0 \leq m \leq M-1$ **and** $0 \leq n \leq N-1$ **do**
  $v_{m,n} \leftarrow \sum_{k=-\lceil K\sigma \rceil}^{\lceil K\sigma \rceil} g_k w_{m,s_N(n-k)}$
  with $s_N(n) = \min(n \bmod 2N, 2N - 1 - n \bmod 2N))$

return v

Algorithm 4: Lindeberg's smoothing method.

Input: u input digital image of $M \times N$ pixels.
Output: v output digital image of $M \times N$ pixels.
Parameters: σ applied blur.
$0 \leq \gamma \leq 1/2$ parameter defining $\Delta_\gamma^{\text{discr}}$ the Laplacian finite difference scheme.
Temporary: P, number of Euler iterations.
δt, Euler step size.
$\Delta_+ v$, $\Delta_\times v$, $\Delta_\gamma^{\text{discr}} v$, auxiliary discrete Laplacians.
`// Euler method setting`
$P \leftarrow \lceil 8(1 - \gamma/2)\sigma^2 \rceil$
$\delta t \leftarrow \frac{\sigma}{P}$
`// Initialization`
$v \leftarrow u$
`// Euler Method`
for $p = 1, .., P$ **do**
  `// Compute discrete Laplacian`
  for $0 \leq k \leq M-1$ **and** $0 \leq l \leq N-1$ **do**
    $\Delta_+ v_{k,l} \leftarrow u_{k+1,l} + u_{k-1,l} + u_{k,l+1} + u_{k,l-1} - 4u_{k,l}$
    $\Delta_\times u_{k,l} \leftarrow \frac{1}{2}(u_{k+1,l+1} + u_{k+1,l-1} + u_{k-1,l+1} + u_{k-1,l-1}) - 2u_{k,l}$
    $\Delta_\gamma^{\text{discr}} v_{k,l} \leftarrow (1-\gamma)\Delta_+ v_{k,l} + \gamma \Delta_\times v_{k,l}$
    note: The half-sample symmetric boundary condition is used.
  `// Euler iteration formula`
  for $0 \leq k \leq M-1$ **and** $0 \leq l \leq N-1$ **do**
    $v_{k,l} \leftarrow v_{k,l} - \delta t \Delta_\gamma^{\text{discr}} v_{k,l}$

note: $\lceil \cdot \rceil$ denotes the ceiling function.

3 Anatomy of the SIFT Method

This chapter presents a detailed description and implementation of the SIFT method. This contributes to a detailed dissection of SIFT's complex chain of transformations and to a careful presentation of each of its design parameters. A companion online demonstration allows the reader to use SIFT and individually set each parameter to analyze its impact on the algorithm results.

3.1 General description

The scale invariant feature transform, SIFT [Lowe 2004], extracts a set of descriptors from an image. The extracted descriptors is invariant to an image *translation, rotation and scaling (zoom-out)*. SIFT descriptors have also proved to be robust to a wide family of image transformations, such as slight changes of viewpoint, noise, blur, contrast changes, scene deformation, while remaining discriminative enough for matching purposes.

The seminal paper introducing SIFT in 1999 [Lowe 1999] has sparked an explosion of competitors. The performance of some of them is examined in Chapter 5.

The SIFT algorithm consists of two successive and independent operations: the detection of interesting points (i.e., keypoints) and the extraction of a descriptor associated with each of them. Since these descriptors are robust, they are usually used for matching pair of images. Object recognition and video stabilization are other popular applications examples. Although the descriptor comparison is not strictly speaking a step of the SIFT method, we have included it in our description for a sake of completeness.

The algorithm principle. SIFT detects a series of keypoints from a multi-scale image representation. This multi-scale representation consists of the Gaussian scale-space introduced in Chapter 2. Each keypoint is a blob-like structure whose center position (x, y) and characteristic scale σ are accurately located. SIFT computes the dominant orientation θ over a region surrounding each one of these keypoints. For each keypoint, the quadruple (x, y, σ, θ) defines the center, size and orientation of a normalized patch where the SIFT descriptor is computed. As a result of this normalization, SIFT keypoint descriptors are in theory invariant to any translation, rotation and scale change. The descriptor encodes the spatial gradient distribution around a keypoint by a 128-dimensional vector. This feature vector is generally used to match keypoints extracted from different images.

The algorithmic chain. In order to attain scale invariance, SIFT is built on the Gaussian scale-space. Here, the Gaussian scale-space simulates the family of all possible zoom-outs through increasingly blurred versions of the input image. The Gaussian convolution acts as an approximation of the optical blur, and the Gaussian kernel approximates the camera's point spread function. Section 3.2 details how the Gaussian scale-space representation is computed in SIFT.

To attain translation, rotation and scale invariance, the extracted keypoints must be related to structures that are unambiguously located, both in scale and position. This excludes image corners and edges since they cannot be precisely localized both in scale and space. Image blobs or more complex local structures characterized by their position and size, are therefore the most suitable structures for SIFT.

Detecting keypoints consists in computing the 3D extrema of a differential operator applied to the scale-space. The differential operator used in the SIFT algorithm is the difference of Gaussians (DoG), presented in Section 3.3.1. The extraction of 3D continuous extrema consists of two steps: first, the DoG representation is scanned for 3D discrete extrema. This gives a first coarse location of the extrema, which are then refined to subpixel precision using a local quadratic model. The extraction of 3D extrema is detailed in Section 3.3.2. As we will see in Chapter 4, there are many phenomena that can lead to the detection of unstable keypoints. Therefore SIFT incorporates a cascade of tests to discard the less reliable ones. Only those that are precisely located and sufficiently contrasted are retained. Section 3.3.3 discuses two different discarding steps: the rejection of 3D extrema with small DoG value and the rejection of keypoint candidates laying on edges.

SIFT invariance to rotation is obtained by assigning to each keypoint a reference orientation. This reference is computed from the gradient orientation over a keypoint neighborhood. This step is detailed in Section 3.4.1. Finally the spatial distribution of the gradient inside an oriented patch is encoded to produce the SIFT keypoint descriptor. The design of the SIFT keypoint descriptor is described in Section 3.4.2. This ends the algorithmic chain defining the SIFT algorithm. Additionally, Section 3.5 illustrates how SIFT descriptors can be used to find local matches between pairs of images. The method presented here is the matching procedure described in the original paper by D. Lowe [Lowe 1999].

This complex chain of transformations is governed by a large number of design parameters. Section 3.6 summarizes all of them and provides an analysis of their respective influence. Chapter 4 will provide a thorough analysis of the parameters affecting the scale-space and the detection of keypoints. Table 3.1 presents the details of the adopted notation while the consecutive steps of the SIFT algorithm are summarized in Table 3.2.

3.2 The Gaussian scale-space

The Gaussian scale-space representation is a family of increasingly blurred images. This blurring process simulates the loss of detail produced when a scene is photographed from farther and farther (i.e., when the zoom-out factor increases). The scale-space, therefore, provides SIFT with scale invariance as it can be interpreted as the simulation of a set of snapshots of a given scene taken at different distances. In what follows we detail the construction of the SIFT scale-space.

u	Images, defined on the continuous domain $(x, y) = \mathbf{x} \in \mathbb{R}^2$
$\mathbf{u}$	Digital images, defined in a rectangular grid $(m, n) \in \{0, \dots, M-1\} \times \{0, \dots, N-1\}$
v	Gaussian scale-space, defined on continuous domain $(\sigma, \mathbf{x}) \in \mathbb{R}^+ \times \mathbb{R}^2$
$\mathbf{v}$	Digital Gaussian scale-space, list of octaves $\mathbf{v} = (\mathbf{v}^o)$, $o = 1, \dots, n_{\text{oct}}$ Each octave $\mathbf{v}^o$ is defined on a discrete grid $(s, m, n) \in \{0, \dots, n_{\text{spo}}+2\} \times \{0, \dots, M_o-1\} \times \{0, \dots, N_o-1\}$
w	Difference of Gaussians (DoG), defined on continuous domain $(\sigma, \mathbf{x}) \in \mathbb{R}^+ \times \mathbb{R}^2$
$\mathbf{w}$	Digital difference of Gaussians (DoG), list of octaves $\mathbf{w} = (\mathbf{w}^o)$, $o = 1, \dots, n_{\text{oct}}$ Each octave $\mathbf{w}^o$ is defined on a discrete grid $(s, m, n) \in \{0, \dots, n_{\text{spo}}+1\} \times \{0, \dots, M_o-1\} \times \{0, \dots, N_o-1\}$
$\boldsymbol{\omega}$	DoG value after 3D extremum subpixel refinement
$\partial_x v$	Scale-space gradient along x ($\partial_y v$ along y), defined on continuous domain $(\sigma, \mathbf{x}) \in \mathbb{R}^+ \times \mathbb{R}^2$
$\partial_m \mathbf{v}$	Digital scale-space gradient along x ($\partial_n \mathbf{v}$ along y), list of octaves ($\partial_m \mathbf{v} = (\partial_m \mathbf{v}^o)$, $o = 1, \dots, n_{\text{oct}}$) Each octave $\partial_m \mathbf{v}^o$ is defined on a discrete grid $(s, m, n) \in \{2, \dots, n_{\text{spo}}\} \times \{1, \dots, M_o-2\} \times \{1, \dots, N_o-2\}$
G_ρ	Continuous Gaussian convolution of standard deviation ρ
$\mathbf{G}_\rho$	Digital Gaussian convolution of standard deviation ρ (see (3.4))
$\mathbf{S}_2$	Subsampling operator by a factor 2, $(\mathbf{S}_2\mathbf{u})(m, n) = \mathbf{u}(2m, 2n)$
$\mathbf{I}_\delta$	Digital bilinear interpolator by a factor $1/\delta$ (see Algorithm 6).

Table 3.1: Summary of the notation used in this chapter.

3.2.1 Gaussian blurring

Consider a *continuous* image $u(\mathbf{x})$ defined for every $\mathbf{x} = (x, y) \in \mathbb{R}^2$. Let us remind that the continuous Gaussian smoothing is defined as the convolution

$$G_\sigma u(\mathbf{x}) := \int_{\mathbb{R}^2} G_\sigma(\mathbf{x}') u(\mathbf{x} - \mathbf{x}') d\mathbf{x}'$$

where $G_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma^2} e^{-\frac{|\mathbf{x}|^2}{2\sigma^2}}$ is the Gaussian kernel parameterized by its standard deviation $\sigma \in \mathbb{R}^+$. The Gaussian smoothing operator satisfies a semi-group relation,

$$G_{\sigma_2}(G_{\sigma_1} u)(\mathbf{x}) = G_{\sqrt{\sigma_1^2+\sigma_2^2}} u(\mathbf{x}). \tag{3.1}$$

We call Gaussian scale-space of u the three-dimensional (3D) function

$$v : (\sigma, \mathbf{x}) \mapsto G_\sigma u(\mathbf{x}). \tag{3.2}$$

We have seen in Chapter 2 that in the case of digital images there is some ambiguity on how to define a discrete counterpart to the continuous Gaussian smoothing operator. In Lowe's original work, the digital Gaussian smoothing is implemented as a discrete convolution with samples of a truncated Gaussian kernel.

Digital Gaussian smoothing. Let $\mathbf{g}_\sigma$ be the one-dimensional digital kernel obtained by sampling a truncated Gaussian function of standard deviation σ,

$$\mathbf{g}_\sigma(k) = K e^{-\frac{k^2}{2\sigma^2}}, \quad -\lceil 4\sigma \rceil \le k \le \lceil 4\sigma \rceil, \quad k \in \mathbb{Z} \tag{3.3}$$

where $\lceil \cdot \rceil$ denotes the ceil function and K is set so that $\sum \mathbf{g}_\sigma(k) = 1$. Let $\mathbf{G}_\sigma$ denote the digital Gaussian convolution of parameter σ and $\mathbf{u}$ be a digital image of size $M \times N$. Its

Stage	Description
1.	Compute the Gaussian scale-space in: $\mathbf{u}$ image out:$\mathbf{v}$ scale-space
2.	Compute the Difference of Gaussians (DoG) in: $\mathbf{v}$ scale-space out: $\mathbf{w}$ DoG
3.	Find candidate keypoints (3D discrete extrema of DoG) in: $\mathbf{w}$ DoG out: $\{(x_d, y_d, \sigma_d)\}$ list of discrete extrema (position and scale)
4.	Refine candidate keypoints location with sub-pixel precision in: $\mathbf{w}$ DoG and $\{(x_d, y_d, \sigma_d)\}$ list of discrete extrema out: $\{(x, y, \sigma)\}$ list of interpolated extrema
5.	Filter unstable keypoints due to noise in: $\mathbf{w}$ DoG and $\{(x, y, \sigma)\}$ out: $\{(x, y, \sigma)\}$ list of filtered keypoints
6.	Filter unstable keypoints laying on edges in: $\mathbf{w}$ DoG and $\{(x, y, \sigma)\}$ out: $\{(x, y, \sigma)\}$ list of filtered keypoints
7.	Assign a reference orientation to each keypoint in: $(\partial_m \mathbf{v}, \partial_n \mathbf{v})$ scale-space gradient and $\{(x, y, \sigma)\}$ list of keypoints out: $\{(x, y, \sigma, \theta)\}$ list of oriented keypoints
8.	Build the keypoints descriptor in: $(\partial_m \mathbf{v}, \partial_n \mathbf{v})$ scale-space gradient and $\{(x, y, \sigma, \theta)\}$ list of keypoints out: $\{(x, y, \sigma, \theta, \mathbf{f})\}$ list of described keypoints

Table 3.2: Summary of the SIFT algorithm.

digital Gaussian smoothing, denoted by $\mathbf{G}_\sigma \mathbf{u}$, is computed via a separable two-dimensional (2D) discrete convolution:

$$\mathbf{G}_\sigma \mathbf{u}(k,l) := \sum_{k'=-\lceil 4\sigma \rceil}^{\lceil 4\sigma \rceil} \mathbf{g}_\sigma(k') \sum_{l'=-\lceil 4\sigma \rceil}^{\lceil 4\sigma \rceil} \mathbf{g}_\sigma(l')\, \bar{\mathbf{u}}(k-k', l-l'), \tag{3.4}$$

where $\bar{\mathbf{u}}$ denotes the extension of $\mathbf{u}$ to $\mathbb{Z}^2$ via symmetrization with respect to $-1/2$, namely,

$$\bar{\mathbf{u}}(k,l) = \mathbf{u}(s_M(k), s_N(l)) \quad \text{with} \quad s_M(k) = \min(k \bmod 2M, 2M - 1 - k \bmod 2M).$$

For the range of values of σ considered in the described algorithm (i.e., $\sigma \geq 0.7$), the digital Gaussian smoothing operator approximately satisfies a semi-group relation with an error below 10^{-4} for pixel intensity values ranging from 0 to 1 (as we have seen in Section 5.5). Applying successively two digital Gaussian smoothings of parameters σ_1 and σ_2 is approximately equal to applying one digital Gaussian smoothing of parameter $\sqrt{\sigma_1^2 + \sigma_2^2}$,

$$\mathbf{G}_{\sigma_2}(\mathbf{G}_{\sigma_1}\mathbf{u}) = \mathbf{G}_{\sqrt{\sigma_1^2+\sigma_2^2}}\mathbf{u}. \tag{3.5}$$

3.2.2 Digital Gaussian scale-space

As previously introduced, the Gaussian scale-space $v : (\mathbf{x}, \sigma) \mapsto G_\sigma u(\mathbf{x})$ is a family of increasingly blurred images, where the scale-space position $(\mathbf{x}, \sigma)$ refers to the pixel $\mathbf{x}$ in the image generated with blur σ. In what follows, we detail how to compute the *digital scale-space*, a discrete counterpart of the continuous Gaussian scale-space.

We will call digital scale-space a family of digital images relative to a discrete set of blur levels and different sampling rates, all of them derived from an input image $\mathbf{u}_{\text{in}}$ with assumed blur level σ_{in}. This family is split into subfamilies of images sharing a common sampling rate. Since in the original SIFT algorithm the sampling rate is iteratively decreased by a factor of two, these subfamilies are called *octaves*.

Let n_{oct} be the total number of octaves in the digital scale-space, $o \in \{1, \dots, n_{\text{oct}}\}$ be the index of each octave, and δ_o its inter-pixel distance. We will adopt as a convention that the input image $\mathbf{u}_{\text{in}}$ inter-pixel distance is $\delta_{\text{in}} = 1$. Thus, an inter-pixel distance $\delta = 0.5$ corresponds to a $2\times$ upsampling of this image while a $2\times$ subsampling results in an inter-pixel distance $\delta = 2$. Let n_{spo} be the number of scales per octave (the default value is $n_{\text{spo}} = 3$). Each octave o contains the images $\mathbf{v}_s^o$ for $s = 1, \dots, n_{\text{spo}}$, each of them with a different blur level σ_s^o. The blur level in the digital scale-space is measured taking as unit length the inter-sample distance in the sampling grid of the input image $\mathbf{u}_{\text{in}}$ (i.e., $\delta_{\text{in}} = 1$). The adopted configuration is illustrated in Figure 3.1.

The digital scale-space also includes three additional images per octave, $\mathbf{v}_0^o, \mathbf{v}_{n_{\text{spo}}+1}^o, \mathbf{v}_{n_{\text{spo}}+2}^o$. The rationale for this will become clear later.

The construction of the digital scale-space begins with the computation of a *seed* image denoted by $\mathbf{v}_0^1$. This image will have a blur level $\sigma_0^1 = \sigma_{\min}$, which is the minimum blur level considered, and inter pixel distance $\delta_0 = \delta_{\min}$. It is computed from $\mathbf{u}_{\text{in}}$ by

$$\mathbf{v}_0^1 = \mathbf{G}_{\frac{1}{\delta_{\min}}\sqrt{\sigma_{\min}^2 - \sigma_{\text{in}}^2}} \mathbf{I}_{\delta_{\min}} \mathbf{u}_{\text{in}}, \tag{3.6}$$

where $\mathbf{I}_{\delta_{\min}}$ is the digital bilinear interpolator by a factor $1/\delta_{\min}$ (see Algorithm 5) and $\mathbf{G}_\sigma$ is the digital Gaussian convolution already defined. The entire digital scale-space is derived

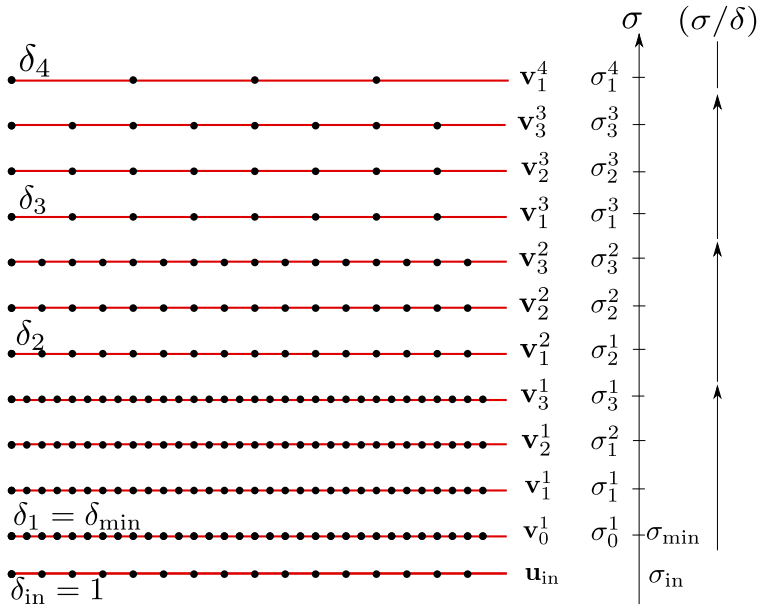


Figure 3.1: Convention adopted for the sampling grid of the digital scale-space $\mathbf{v}$. The blur level is considered with respect to the sampling grid of the input image. The parameters are set to their default value, namely $\sigma_{\text{min}} = 0.8, n_{\text{spo}} = 5, n_{\text{oct}} = 8, \sigma_{\text{in}} = 0.5$.

from this seed image. The default value $\delta_{\min} = 0.5$ implies an initial $2\times$ interpolation. The blur level of the seed image, relative to the input image sampling grid, is set as default to $\sigma_{\min} = 0.8$.

The second and posterior scale-space images $s = 1, \ldots, n_{\text{spo}} + 2$ at each octave o are computed recursively according to

$$\mathbf{v}_s^o = \mathbf{G}_{\rho_{[(s-1)\to s]}} \mathbf{v}_{s-1}^o, \tag{3.7}$$

where

$$\rho_{[(s-1)\to s]} = \frac{\sigma_{\min}}{\delta_{\min}} \sqrt{2^{2s/n_{\text{spo}}} - 2^{2(s-1)/n_{\text{spo}}}}.$$

The first images (i.e., $s = 0$) of the octaves $o = 2, \ldots, n^o$ are computed as

$$\mathbf{v}_0^o = \mathbf{S}_2 \mathbf{v}_{n_{\text{spo}}}^{o-1}, \tag{3.8}$$

where $\mathbf{S}_2$ denotes the subsampling operator by a factor of 2, $(\mathbf{S}_2\mathbf{u})(m,n) = \mathbf{u}(2m, 2n)$. This procedure produces a family of images $(\mathbf{v}_s^o)$, $o = 1, \ldots, n_{\text{oct}}$ and $s = 0, \ldots, n_{\text{spo}} + 2$, having inter-pixel distance

$$\delta_o = \delta_{\min} 2^{o-1} \tag{3.9}$$

and blur level

$$\sigma_s^o = \frac{\delta_o}{\delta_{\min}} \sigma_{\min} 2^{s/n_{\text{spo}}}. \tag{3.10}$$

Consequently, the simulated blurs follow a geometric progression. The scale-space construction process is summarized in Algorithm 5. The digital scale-space construction is thus defined by five parameters:

- the number of octaves n_{oct},
- the number of scales per octave n_{spo},
- the sampling distance $\delta_{\min}$ of the first image of the scale-space $\mathbf{v}_0^1$,
- the blur level $\sigma_{\min}$ of the first image of the scale-space $\mathbf{v}_0^1$, and
- σ_{in} the assumed blur level in the input image $\mathbf{u}^{\text{in}}$.

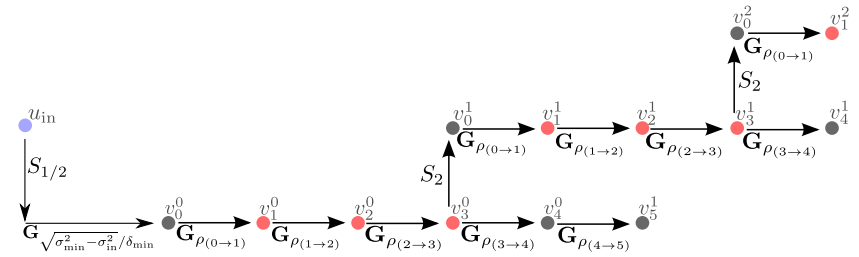
The diagram in Figure 3.2 depicts the digital scale-space architecture in terms of the sampling rates and blur levels. Each point symbolizes a scale-space image $\mathbf{v}_s^o$ having inter-pixel distance δ^o and blur level σ_s^o. The featured configuration is produced from the default parameter values of the Lowe SIFT algorithm: $\sigma_{\min} = 0.8$, $\delta_{\min} = 0.5$, $n_{\text{spo}} = 3$, and $\sigma_{\text{in}} = 0.5$. The number of octaves n_{oct} is upper limited by the number of possible subsamplings. Figure 3.3 shows an example of the digital scale-space images generated with the given configuration.

Algorithm 5: Computation of the digital Gaussian scale-space

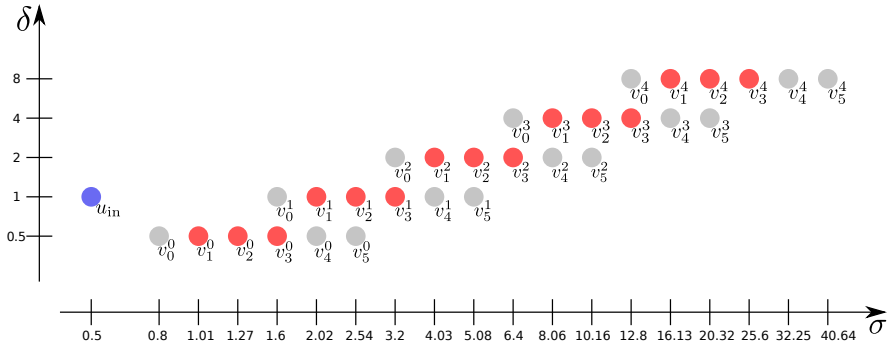
Input: $\mathbf{u}_{\text{in}}$, input digital image of $M \times N$ pixels.
Output: $(\mathbf{v}_s^o)$, digital scale-space, $o = 1, \dots, n_{\text{oct}}$ and $s = 0, \dots, n_{\text{spo}} + 2$.
$\mathbf{v}_s^o$ is a digital image of size $M_o \times N_o$, blur level σ_s^o (eq. (3.10)) and inter-pixel distance $\delta^o = \delta_{\min} 2^{o-1}$, with $M_o = \lfloor \frac{\delta_{\min}}{\delta_o} M \rfloor$ and $N_o = \lfloor \frac{\delta_{\min}}{\delta_o} N \rfloor$. The samples of $\mathbf{v}_s^o$ are denoted by $\mathbf{v}_s^o(m, n)$.
Parameters: - n_{oct}, number of octaves.
- n_{spo}, number of scales per octave.
- $\sigma_{\min}$, blur level in the seed image.
- $\delta_{\min}$, inter-sample distance in the seed image.
- σ_{in}, assumed blur level in the input image.

```
//Compute the first octave
  //Compute the seed image v_0^1
    //1.Interpolate the original image (Bilinear interpolation, see Algo 6)
    u' ← bilinear_interpolation(u_in, δ_min)
    // 2.  Blur the interpolated image (Gaussian blur, see eq (3.4))
    v_0^1 = G_{(1/δ_min)√(σ_min^2 − σ_in^2)} u'
  // Compute the other images in the first octave
  for s = 1, ..., n_spo + 2 do
      v_s^1 = G_{ρ_[(s−1)→s]} v_{s−1}^1

// Compute subsequent octaves
for o = 2, ..., n_oct do
    // Compute the first image in the octave by subsampling
    for m = 0, ..., M_o − 1 and  n = 0, ..., N_o − 1 do
        v_0^o(m, n) ← v_{n_spo}^{o−1}(2m, 2n)
    // Compute the other images in octave o
    for s = 1, ..., n_spo + 2 do
        v_s^o = G_{ρ_[(s−1)→s]} v_{s−1}^o
```



(a) Scale-space construction



(b) Scale-space default configuration

Figure 3.2: (a) The succession of subsamplings and Gaussian convolutions that results in the SIFT scale-space. The first image at each octave $\mathbf{v}_0^o$ is obtained via subsampling, with the exception of the first octave first image $\mathbf{v}_0^0$ that is generated by a bilinear interpolation followed by a Gaussian convolution. (b) An illustration of the digital scale-space in its default configuration. The digital scale-space $\mathbf{v}$ is composed of images $\mathbf{v}_s^o$ for $o = 1, \dots, n_{\text{oct}}$ and $s = 0, \dots, n_{\text{spo}} + 2$. All images are computed directly or indirectly from $\mathbf{u}^{\text{in}}$ (in blue). Each image is characterized by its blur level and its inter-pixel distance, respectively noted by σ and δ. The scale-space is split into octaves: sets of images sharing a common sampling rate. Each octave is composed of n_{spo} scales (in red) and other three auxiliary scales (in gray). The depicted configuration features $n_{\text{oct}} = 5$ octaves and corresponds to the following parameter settings: $n_{\text{spo}} = 3$, $\sigma_{\text{min}} = 0.8$. The assumed blur level of the input image is $\sigma_{\text{in}} = 0.5$.

Algorithm 6: Bilinear interpolation of an image

Input: $\mathbf{u}$, digital image, $M \times N$ pixels. The samples are denoted by $\mathbf{u}(m, n)$.
Output: $\mathbf{u}'$, digital image, $M' \times N'$ pixels with $M' = \lfloor \frac{M}{\delta'} \rfloor$ and $N' = \lfloor \frac{N}{\delta'} \rfloor$.
Parameter: $\delta' < 1$, inter-pixel distance of the output image.
for $m' = 0, \dots, M' - 1$ **and** $n' = 0, \dots, N' - 1$ **do**

$x \leftarrow \delta' m'$ $y \leftarrow \delta' n'$

$$\begin{aligned}\mathbf{u}'(m', n') \leftarrow\ & (x - \lfloor x \rfloor)\,(y - \lfloor y \rfloor)\,\bar{\mathbf{u}}(\lfloor x \rfloor + 1, \lfloor y \rfloor + 1) + (1 + \lfloor x \rfloor - x)\,(y - \lfloor y \rfloor)\,\bar{\mathbf{u}}(\lfloor x \rfloor, \lfloor y \rfloor + 1) \\ & + (x - \lfloor x \rfloor)\,(1 + \lfloor y \rfloor - y)\,\bar{\mathbf{u}}(\lfloor x \rfloor + 1, \lfloor y \rfloor) + (1 + \lfloor x \rfloor - x)\,(1 + \lfloor y \rfloor - y)\,\bar{\mathbf{u}}(\lfloor x \rfloor, \lfloor y \rfloor)\end{aligned}$$

$\bar{\mathbf{u}}$ denotes the extension of $\mathbf{u}$ to $\mathbb{Z}^2$ via symmetrization with respect to -0.5:
$\bar{\mathbf{u}}(k, l) = \mathbf{u}(s_M(k), s_N(l))$ with $s_N(k) = \min(k \bmod 2M, 2M - 1 - k \bmod 2M)$.
note: $\lfloor \cdot \rfloor$ denotes the floor function.



Figure 3.3: Crops of a subset of images extracted from the Gaussian scale-space of an example image. The scale-space parameters are set to $n_{\mathsf{spo}} = 3$, $\sigma_{\mathsf{min}} = 0.8$, and the assumed input image blur level $\sigma_{\mathsf{in}} = 0.5$. Image pixels are represented by a square of side δ_o for better visualization.

3.3 Keypoint definition

Precisely detecting interesting image features is a challenging problem. The keypoint features are defined in SIFT as the extrema of the normalized Laplacian scale-space $\sigma^2 \Delta v$ [Lindeberg 1993]. A Laplacian extremum is unequivocally characterized by its scale-space coordinates $(\sigma, \mathbf{x})$ where $\mathbf{x}$ refers to its center spatial position and σ relates to its size (scale). As will be presented in Section 3.4, the covariance of the extremum $(\sigma, \mathbf{x})$ induces the invariance to translation and scale of its associated descriptor.

Instead of computing the Laplacian of the image scale space, SIFT uses a difference of Gaussians operator (DoG), first introduced by Burt and Adelson [Burt and Adelson 1983] and Crowley and Stern [Crowley and Stern 1984]. Let v be a Gaussian scale-space and $\kappa > 1$. The difference of Gaussians (DoG) of ratio κ is defined by $w : (\sigma, \mathbf{x}) \mapsto v(\kappa\sigma, \mathbf{x}) - v(\sigma, \mathbf{x})$. The DoG operator takes advantage of the link between the Gaussian kernel and the heat equation to approximately compute the normalized Laplacian $\sigma^2 \Delta v$. Indeed, from a set of simulated blurs following a geometric progression of ratio κ, the heat equation is approximated by

$$\sigma \Delta v = \frac{\partial v}{\partial \sigma} \approx \frac{v(\kappa\sigma, \mathbf{x}) - v(\sigma, \mathbf{x})}{\kappa\sigma - \sigma} = \frac{w(\sigma, \mathbf{x})}{(\kappa - 1)\sigma}. \tag{3.11}$$

Thus, we have $w(\sigma, \mathbf{x}) \approx (\kappa - 1)\sigma^2 \Delta v(\sigma, \mathbf{x})$.

The SIFT keypoints of an image are defined as the 3D extrema of the difference of Gaussians (DoG). Since we deal with digital images, the continuous 3D extrema of the DoG cannot be directly computed. Thus, the discrete extrema of the digital DoG are first detected and then their position is refined. The detected points are finally validated to

discard possible unstable and false detections due to noise. Hence, the detection of SIFT keypoints involves the following steps:

1. compute the digital DoG;
2. scan the digital DoG for 3D discrete extrema;
3. refine position and scale of these candidates via a quadratic interpolation;
4. discard unstable candidates such as uncontrasted candidates or candidates laying on edges.

We detail each one of these steps in what follows.

3.3.1 Scale-space analysis: Difference of Gaussians

The digital DoG $\mathbf{w}$ is built from the digital scale-space $\mathbf{v}$. In each octave $o = 1, \dots, n_{\text{oct}}$ and for each image $\mathbf{w}_s^o$ with $s = 0, \dots, n_{\text{spo}} + 1$

$$\mathbf{w}_s^o(m, n) = \mathbf{v}_{s+1}^o(m, n) - \mathbf{v}_s^o(m, n)$$

with $m = 0, \dots, M_o - 1$, $n = 0, \dots, N_o - 1$. The image $\mathbf{w}_s^o$ will be attributed the blur level σ_s^o. This computation is illustrated in Figure 3.4 and summarized in Algorithm 7. See how, in the digital scale-space, the computation of the auxiliary image $\mathbf{v}_{n_{\text{spo}}+2}^o$ is required for computing the DoG approximation $\mathbf{w}_{n_{\text{spo}}+1}^o$. Figure 3.5 illustrates the DoG scale-space $\mathbf{w}$ relative to the previously introduced Gaussian scale-space $\mathbf{v}$. Figure 3.6 shows images of an example of DoG scale-space.

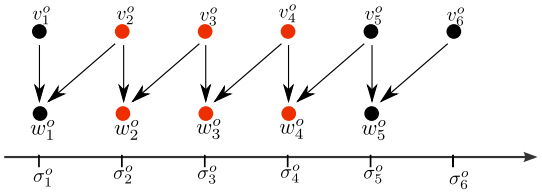


Figure 3.4: The difference of Gaussians operator is computed by subtracting pairs of contiguous scale-space images. The procedure is not centered: the difference between the images at scales $\kappa\sigma$ and σ is attributed a blur level σ.

3.3.2 Extraction of candidate keypoints

Continuous 3D extrema of the digital DoG are calculated in two successive steps. The 3D discrete extrema are first extracted from $(\mathbf{w}_s^o)$ with pixel precision, then their location are refined through interpolation of the digital DoG by using a quadratic model. In what follows, samples $\mathbf{v}_s^o(m, n)$ and $\mathbf{w}_s^o(m, n)$ are noted respectively $\mathbf{v}_{s,m,n}^o$ and $\mathbf{w}_{s,m,n}^o$ for better readability.

Detection of DoG 3D discrete extrema Each sample $\mathbf{w}_{s,m,n}^o$ of the DoG scale-space, with $s = 1, \dots, n_{\text{spo}}$, $o = 1, \dots, n_{\text{oct}}$, $m = 1, \dots, M_o - 2$, $n = 1, \dots, N_o - 2$ (which excludes the image borders and the auxiliary images) is compared to its neighbors to

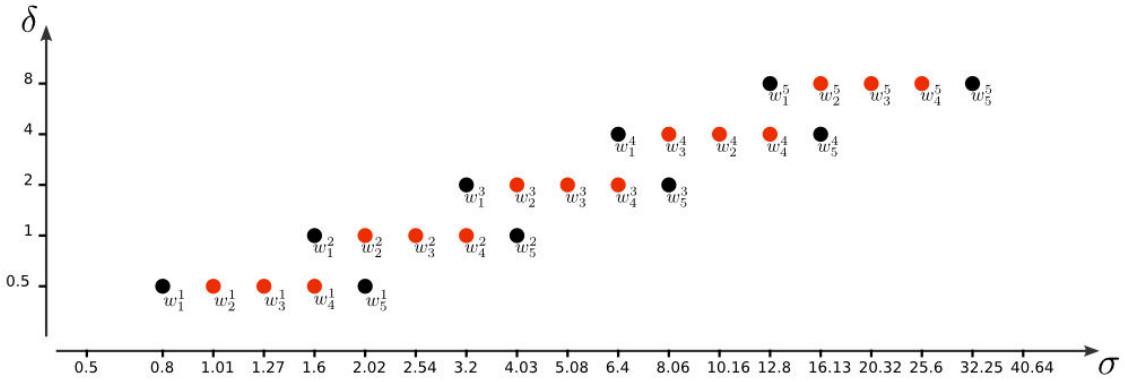


Figure 3.5: The *DoG* scale-space. The difference of Gaussians acts as an approximation of the normalized Laplacian $\sigma^2\Delta$. The difference $\mathbf{w}_s^o = \mathbf{v}_{s+1}^o - \mathbf{v}_s^o$ is relative to the blur level σ_s^o. Each octave contains n_{spo} images plus two auxiliary images (in black).

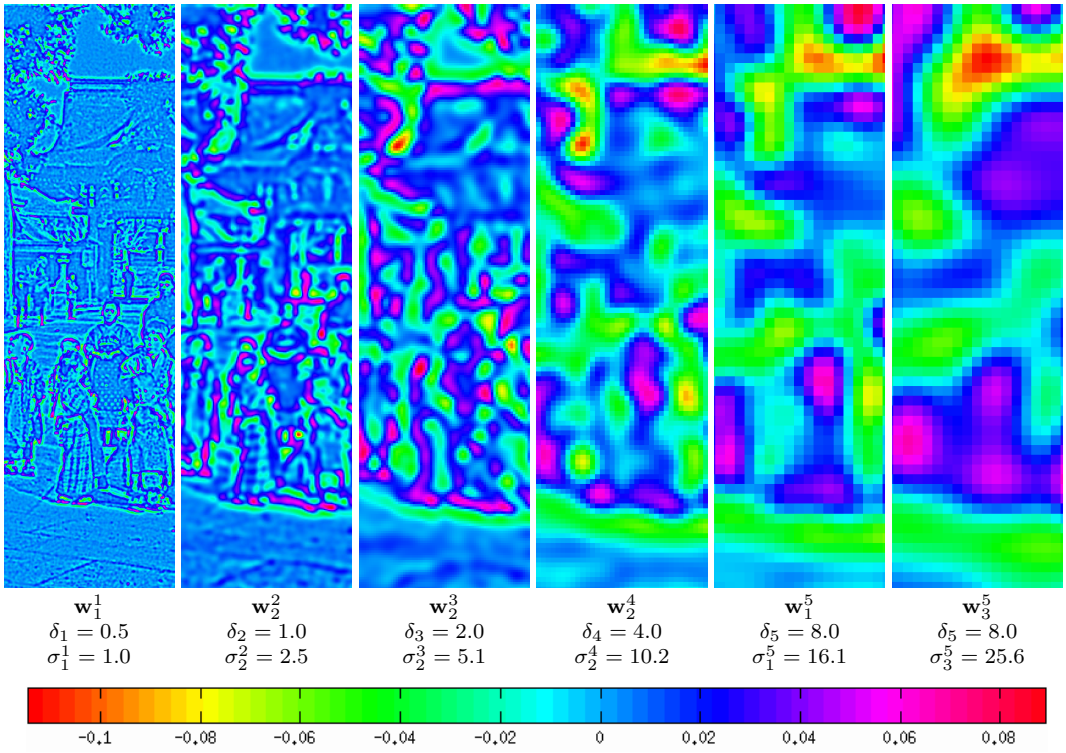


Figure 3.6: Crops of a subset of images extracted from the DoG scale-space of an example image. The DoG operator is an approximation of the normalized Laplacian operator $\sigma^2\Delta v$. The DoG scale-space parameters used in this example are the default: $n_{\text{spo}} = 3$, $\sigma_{\text{min}} = 0.8$, $\sigma_{\text{in}} = 0.5$. Image pixels are represented by a square of side δ_o for better visualization.

detect the 3D discrete maxima and minima (the number of neighbors is $26 = 3 \times 3 \times 3 - 1$). Algorithm 8 summarizes the extraction of 3D extrema from the digital DoG. These comparisons are possible thanks to the auxiliary images $\mathbf{w}_0^o$, $\mathbf{w}_{n_{\text{spo}}+1}^o$ calculated for each octave o. This scanning process is nevertheless a rudimentary way to detect candidate points. It is sensitive to noise, produces unstable detections, and the information it provides regarding the location and scale may be flawed since it is constrained to the sampling grid. To amend these shortcomings, this preliminary step is followed by an interpolation that refines the localization of the extrema and by a cascade of filters that discard unreliable detections.

Keypoint position refinement The location of the discrete extrema is constrained to the sampling grid (defined by the octave o). This coarse localization hinders a rigorous covariance property of the set of keypoints and subsequently is an obstacle to the full scale and translation invariance of the corresponding descriptor. SIFT refines the position and scale of each candidate keypoint using a local interpolation model.

We denote by $\boldsymbol{\omega}_{s,m,n}^o(\boldsymbol{\alpha})$ the quadratic function at sample point (s, m, n) in the octave o, given by

$$\boldsymbol{\omega}_{s,m,n}^o(\boldsymbol{\alpha}) = \mathbf{w}_{s,m,n}^o + \boldsymbol{\alpha}^T \bar{g}_{s,m,n}^o + \frac{1}{2}\boldsymbol{\alpha}^T \bar{H}_{s,m,n}^o \boldsymbol{\alpha}, \tag{3.12}$$

where $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3)^T \in [-1/2, 1/2]^3$; $\bar{g}_{s,m,n}^o$ and $\bar{H}_{s,m,n}^o$ denote respectively the 3D gradient and Hessian at (s, m, n) in the octave o, computed with a finite difference scheme as follows:

$$\bar{g}_{s,m,n}^o = \begin{bmatrix} (\mathbf{w}_{s+1,m,n}^o - \mathbf{w}_{s-1,m,n}^o)/2 \\ (\mathbf{w}_{s,m+1,n}^o - \mathbf{w}_{s,m-1,n}^o)/2 \\ (\mathbf{w}_{s,m,n+1}^o - \mathbf{w}_{s,m,n-1}^o)/2 \end{bmatrix}, \quad \bar{H}_{s,m,n}^o = \begin{bmatrix} h_{ss} & h_{sx} & h_{sy} \\ h_{sx} & h_{xx} & h_{xy} \\ h_{sy} & h_{xy} & h_{yy} \end{bmatrix} \tag{3.13}$$

with

$$\begin{aligned} h_{ss} &= \mathbf{w}_{s+1,m,n}^o + \mathbf{w}_{s-1,m,n}^o - 2\mathbf{w}_{s,m,n}^o, & h_{sx} &= (\mathbf{w}_{s+1,m+1,n}^o - \mathbf{w}_{s+1,m-1,n}^o - \mathbf{w}_{s-1,m+1,n}^o + \mathbf{w}_{s-1,m-1,n}^o)/4, \\ h_{xx} &= \mathbf{w}_{s,m+1,n}^o + \mathbf{w}_{s,m-1,n}^o - 2\mathbf{w}_{s,m,n}^o, & h_{sy} &= (\mathbf{w}_{s+1,m,n+1}^o - \mathbf{w}_{s+1,m,n-1}^o - \mathbf{w}_{s-1,m,n+1}^o + \mathbf{w}_{s-1,m,n-1}^o)/4, \\ h_{yy} &= \mathbf{w}_{s,m,n+1}^o + \mathbf{w}_{s,m,n-1}^o - 2\mathbf{w}_{s,m,n}^o, & h_{xy} &= (\mathbf{w}_{s,m+1,n+1}^o - \mathbf{w}_{s,m+1,n-1}^o - \mathbf{w}_{s,m-1,n+1}^o + \mathbf{w}_{s,m-1,n-1}^o)/4. \end{aligned}$$

This quadratic function is an approximation of the second order Taylor development of the underlying continuous function (where its derivatives are approximated by finite difference schemes).

In order to refine the position of a discrete extremum (s_e, m_e, n_e) at octave o_e SIFT proceeds as follows.

1. Initialize (s, m, n) by the discrete coordinates of the extremum (s_e, m_e, n_e).
2. Compute the continuous extrema of $\boldsymbol{\omega}_{s,m,n}^o$ by solving $\nabla\boldsymbol{\omega}_{s,m,n}^o(\boldsymbol{\alpha}) = 0$ (see Algorithm 11). This yields
$$\boldsymbol{\alpha}^* = -\left(\bar{H}_{s,m,n}^o\right)^{-1} \bar{g}_{s,m,n}^o. \tag{3.14}$$
3. If $\max(|\alpha_1^*|, |\alpha_2^*|, |\alpha_3^*|) \leq 0.5$ (i.e., the extremum of the quadratic function lies in its domain of validity) the extremum is accepted. According to the scale-space

architecture (see (3.10) and (3.9)), the corresponding keypoint coordinates are

$$(\sigma, x, y) = \left(\frac{\delta_{o_e}}{\delta_{\min}} \sigma_{\min} 2^{(\alpha_1^* + s)/n_{\text{spo}}} \, , \, \delta_{o_e}(\alpha_2^* + m) \, , \, \delta_{o_e}(\alpha_3^* + n) \right). \tag{3.15}$$

4. If $\boldsymbol{\alpha}^*$ falls outside the domain of validity, the interpolation is rejected and another one is carried out. Update (s, m, n) to the closest discrete value to $(s, m, n) + \boldsymbol{\alpha}^*$ and repeat from (2).

This process is repeated up to five times or until the interpolation is validated. If after five iterations the result is still not validated, the candidate keypoint is discarded. In practice, the validity domain is defined by $\max(|\alpha_1^*|, |\alpha_2^*|, |\alpha_3^*|) < 0.6$ to avoid possible numerical instabilities due to the fact that the piecewise interpolation model is not continuous. See Algorithm 10 for details.

According to the local interpolation model (3.12), the value of the DoG interpolated extremum is

$$\begin{aligned} \boldsymbol{\omega} := \boldsymbol{\omega}^o_{s,m,n}(\boldsymbol{\alpha}^*) &= \mathbf{w}^o_{s,m,n} + (\boldsymbol{\alpha}^*)^T \bar{g}^o_{s,m,n} + \frac{1}{2}(\boldsymbol{\alpha}^*)^T \bar{H}^o_{s,m,n} \boldsymbol{\alpha}^* \\ &= \mathbf{w}^o_{s,m,n} + \frac{1}{2}(\boldsymbol{\alpha}^*)^T \bar{g}^o_{s,m,n}. \end{aligned} \tag{3.16}$$

This value will be used to discard uncontrasted keypoints.

3.3.3 Filtering unstable keypoints

Discarding low contrasted extrema

Image noise will typically produce several spurious DoG extrema. Such extrema are unstable and are not linked to any particular structure in the image. SIFT attempts to eliminate these false detections by discarding candidate keypoints with a DoG value $\boldsymbol{\omega}$ below a threshold C_{DoG} (see Algorithm 12),

$$\textbf{if } |\boldsymbol{\omega}| < C_{\text{DoG}} \textbf{ then } \text{discard the candidate keypoint.}$$

Since the DoG function approximates $(\kappa - 1)\sigma^2 \Delta v$, where κ is a function of the number of scales per octave n_{spo}, the value of threshold C_{DoG} should depend on n_{spo} (default value $C_{\text{DoG}} = 0.015$ for $n_{\text{spo}} = 3$). The threshold applied in the provided source-code is $\widetilde{C}_{\text{DoG}} = \frac{2^{1/n_{\text{spo}}} - 1}{2^{1/3} - 1} C_{\text{DoG}}$, with C_{DoG} relative to $n_{\text{spo}} = 3$. This guarantees that the applied threshold is independent of the sampling configuration. Before the refinement of the extrema, and to avoid unnecessary computations, a less conservative threshold at 80% of C_{DoG} is applied to the discrete 3D extrema (see Algorithm 9),

$$\textbf{if } |\mathbf{w}^o_{s,m,n}| < 0.8 \times C_{\text{DoG}} \textbf{ then } \text{discard the discrete 3D extremum.}$$

This validation step is investigated in Chapter 4.

Discarding candidate keypoints on edges

Candidate keypoints lying on edges are difficult to precisely locate. This is a direct consequence of the fact that an edge is invariant to translations along its principal axis. Such detections do not help define covariant keypoints and should be discarded. The 2D Hessian of the DoG provides a characterization of those undesirable keypoint candidates. Edges present a large principal curvature orthogonal to the edge and a small one along the edge. In terms of the eigenvalues of the Hessian matrix, the presence of an edge amounts to a big ratio between the largest eigenvalue $\lambda_{\max}$ and the smallest one $\lambda_{\min}$.

The Hessian matrix of the DoG is computed at the nearest grid sample using a finite difference scheme:

$$H^o_{s,m,n} = \begin{bmatrix} h_{xx} & h_{xy} \\ h_{xy} & h_{yy} \end{bmatrix}, \tag{3.17}$$

where

$$h_{xx} = \mathbf{w}^o_{s,m+1,n} + \mathbf{w}^o_{s,m-1,n} - 2\mathbf{w}^o_{s,m,n}, \quad h_{yy} = \mathbf{w}^o_{s,m,n+1} + \mathbf{w}^o_{s,m,n-1} - 2\mathbf{w}^o_{s,m,n},$$
$$h_{xy} = (\mathbf{w}^o_{s,m+1,n+1} - \mathbf{w}^o_{s,m+1,n-1} - \mathbf{w}^o_{s,m-1,n+1} + \mathbf{w}^o_{s,m-1,n-1})/4.$$

The SIFT algorithm discards keypoint candidates whose eigenvalue ratio $r := \lambda_{\max}/\lambda_{\min}$ is less than a certain threshold C_{edge} (the default value is $C_{\text{edge}} = 10$). Since only this ratio is relevant, the eigenvalues computation can be avoided. The ratio of the Hessian matrix determinant and its trace are related to r by

$$\text{edgeness}(H^o_{s,m,n}) = \frac{\text{tr}(H^o_{s,m,n})^2}{\det(H^o_{s,m,n})} = \frac{(\lambda_{\max} + \lambda_{\min})^2}{\lambda_{\max}\lambda_{\min}} = \frac{(r+1)^2}{r}. \tag{3.18}$$

Thus, the filtering of keypoint candidates on edges consists in the following test:

$$\textbf{if } \text{edgeness}(H^o_{s,m,n}) > \frac{(C_{\text{edge}}+1)^2}{C_{\text{edge}}} \textbf{ then } \text{ discard candidate keypoint.}$$

Note that $H^o_{s,m,n}$ is the bottom-right 2×2 sub-matrix of $\bar{H}^o_{s,m,n}$ (3.13) previously computed for the keypoint interpolation. Algorithm 13 summarizes how keypoints on edges are discarded.

3.3.4 Pseudocodes

Algorithm 7: Computation of the difference of Gaussians scale-space (DoG)

Input: $(\mathbf{v}^o_s)$, digital Gaussian scale-space, $o = 1, \dots, n_{\text{oct}}$ and $s = 0, \dots, n_{\text{spo}} + 2$.
Output: $(\mathbf{w}^o_s)$, digital DoG, $o = 1, \dots, n_{\text{oct}}$ and $s = 0, \dots, n_{\text{spo}} + 1$.

for $o = 1, \dots, n_{oct}$ **and** $s = 0, \dots, n_{spo} + 1$ **do**
  for $m = 0, \dots, M_o - 1$ **and** $n = 0, \dots, N_o - 1$ **do**
    $\mathbf{w}^o_s(m,n) = \mathbf{v}^o_{s+1}(m,n) - \mathbf{v}^o_s(m,n)$

Algorithm 8: Scanning for 3D discrete extrema of the DoG scale-space

Input: $(\mathbf{w}_s^o)$, digital DoG, $o = 1, \ldots, n_{\text{oct}}$ and $s = 0, \ldots, n_{\text{spo}} + 1$.
The samples of digital image $\mathbf{w}_s^o$ are denoted by $\mathbf{w}_{s,m,n}^o$.
Output: $\mathcal{L}_{\text{A}} = \{(o, s, m, n)\}$, list of the DoG 3D discrete extrema.

for $o = 1, \ldots, n_{oct}$ **do**
  for $s = 1, \ldots, n_{spo}$, $m = 1, \ldots, M_o - 2$ **and** $n = 1, \ldots, N_o - 2$ **do**
    if *sample* $\mathbf{w}_{s,m,n}^o$ *is larger or smaller than all of its* $3^3 - 1 = 26$ *neighbors* **then**
      Add discrete extremum (o, s, m, n) to $\mathcal{L}_{\text{A}}$

Algorithm 9: Discarding low contrasted candidate keypoints (conservative test)

Inputs: - $(\mathbf{w}_s^o)$, digital DoG, $o = 1, \ldots, n_{\text{oct}}$ and $s = 0, \ldots, n_{\text{spo}} + 1$.
- $\mathcal{L}_{\text{A}} = \{(o, s, m, n)\}$, list of DoG 3D discrete extrema.

Output: $\mathcal{L}_{\text{A'}} = \{(o, s, m, n)\}$, filtered list of DoG 3D discrete extrema.
Parameter: C_{DoG} threshold.

for each *DoG* 3D *discrete extremum* (o, s, m, n) *in* $\mathcal{L}_A$ **do**
  if $|\mathbf{w}_{s,m,n}^o| \geq 0.8 \times C_{DoG}$ **then**
    Add discrete extremum (o, s, m, n) to $\mathcal{L}_{\text{A'}}$

Algorithm 10: Keypoints interpolation

Inputs: - $\mathcal{L}_{\text{A'}} = \{(o, s, m, n)\}$, list of DoG 3D discrete extrema.
- $(\mathbf{w}_s^o)$, digital DoG scale-space, $o = 1, \ldots, n_{\text{oct}}$ and $s = 0, \ldots, n_{\text{spo}} + 1$.

Output: $\mathcal{L}_{\text{B}} = \{(o, s, m, n, x, y, \sigma, \boldsymbol{\omega})\}$, list of candidate keypoints.

for each *DoG* 3D *discrete extremum* (o_e, s_e, m_e, n_e) *in* $\mathcal{L}_A$ **do**
  $(s, m, n) \leftarrow (s_e, m_e, n_e)$ `// initialize interpolation location`
  repeat
    `// Compute the extrema location and value of the local quadratic function` (see Algo 11)
    $(\boldsymbol{\alpha}^*, \boldsymbol{\omega}) \leftarrow$ `quadratic_interpolation`(o_e, s, m, n)
    `// Compute the corresponding absolute coordinates`
    $(\sigma, x, y) = \left(\frac{\delta_{o_e}}{\delta_{\min}} \sigma_{\min} 2^{(\alpha_1^* + s)/n_{\text{spo}}}\,,\, \delta_{o_e}(\alpha_2^* + m)\,,\, \delta_{o_e}(\alpha_3^* + n)\right).$
    `// Update the interpolating position`
    $(s, m, n) \leftarrow ([s + \alpha_1^*], [m + \alpha_2^*], [n + \alpha_3^*])$
  until $\max(|\alpha_1^*|, |\alpha_2^*|, |\alpha_3^*|) < 0.6$ *or after* 5 *unsuccessful tries.*
  if $\max(|\alpha_1^*|, |\alpha_2^*|, |\alpha_3^*|) < 0.6$ **then**
    Add candidate keypoint $(o_e, s, m, n, \sigma, x, y, \boldsymbol{\omega})$ to $\mathcal{L}_{\text{B}}$

note: $[\cdot]$ denotes the round function.

Algorithm 11: Quadratic interpolation on a discrete DoG sample

Inputs: - $(\mathbf{w}_s^o)$, digital DoG scale-space, $o = 1, \ldots, n_{\text{oct}}$ and $s = 0, \ldots, n_{\text{spo}} + 1$.
- (o, s, m, n), coordinates of the DoG 3D discrete extremum.

Outputs: - $\boldsymbol{\alpha}^*$, offset from the center of the interpolated 3D extremum.
- $\boldsymbol{\omega}$, value of the interpolated 3D extremum.

Compute $\bar{g}^o_{s,m,n}$ and $\bar{H}^o_{s,m,n}$ `//DoG 3D gradient and Hessian by eq.`(3.13)

Compute $\boldsymbol{\alpha}^* = -\left(\bar{H}^o_{s,m,n}\right)^{-1} \bar{g}^o_{s,m,n}$

Compute $\boldsymbol{\omega} = \mathbf{w}^o_{s,m,n} - \frac{1}{2}(\bar{g}^o_{s,m,n})^T \left(\bar{H}^o_{s,m,n}\right)^{-1} \bar{g}^o_{s,m,n}$

Algorithm 12: Discarding low contrasted candidate keypoints

Input: $\mathcal{L}_{\text{B}} = \{(o, s, m, n, \sigma, x, y, \boldsymbol{\omega})\}$, list of candidate keypoints.
Output: $\mathcal{L}_{\text{B'}} = \{(o, s, m, n, \sigma, x, y, \boldsymbol{\omega})\}$, reduced list of candidate keypoints.
Parameter: C_{DoG} threshold.

for each *candidate keypoint* $(\sigma, x, y, \boldsymbol{\omega})$ *in* $\mathcal{L}_B$ **do**
  if $|\boldsymbol{\omega}| \geq C_{DoG}$ **then**
    Add candidate keypoint $(\sigma, x, y, \boldsymbol{\omega})$ to $\mathcal{L}_{\text{B'}}$.

Algorithm 13: Discarding candidate keypoints on edges

Inputs: - $(\mathbf{w}_s^o)$, DoG scale-space.
- $\mathcal{L}_{\text{B'}} = \{(o, s, m, n, \sigma, x, y, \boldsymbol{\omega})\}$, list of candidate keypoints.
Output: $\mathcal{L}_{\text{C}} = \{(o, s, m, n, \sigma, x, y, \boldsymbol{\omega})\}$, list of the SIFT keypoints.
Parameter: C_{edge}, threshold over the ratio between first and second Hessian eigenvalues.

for each *candidate keypoint* $(o, s, m, n, \sigma, x, y, \boldsymbol{\omega})$ *in* $\mathcal{L}_{B'}$ **do**
  Compute $H^o_{s,m,n}$ by (3.17) `// 2D Hessian`
  Compute $\frac{\text{tr}(H^o_{s,m,n})^2}{\det(H^o_{s,m,n})}$ `// edgeness`
  if $\frac{tr(H^o_{s,m,n})^2}{det(H^o_{s,m,n})} < \frac{(C_{edge}+1)^2}{C_{edge}}$ **then**
    Add candidate keypoint $(o, s, m, n, \sigma, x, y, \boldsymbol{\omega})$ to $\mathcal{L}_{\text{C}}$.

3.4 Keypoint description

In the literature, rotation invariant descriptors fall into two categories. On the one side, those based on properties of the image that are already *rotation-invariant* and on the other side, descriptors based on a normalization with respect to a reference orientation. The SIFT descriptor is based on a normalization. The local dominant gradient angle is computed and used as a reference orientation. Then, the local gradient distribution is normalized with respect to this reference direction (see Figure 3.7).

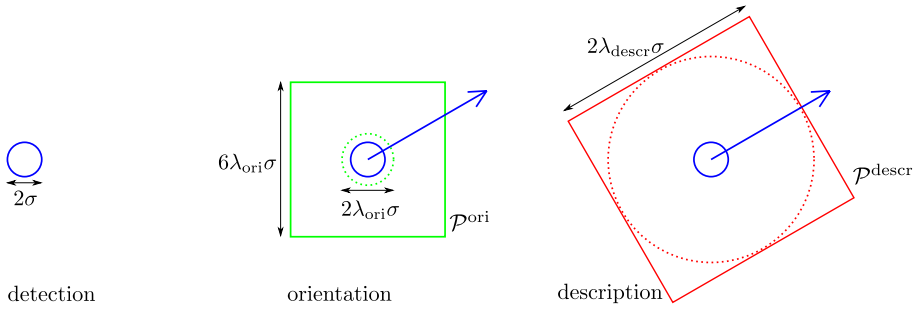


Figure 3.7: The description of a keypoint detected at scale σ (the radius of the blue circle) involves the analysis of the image gradient distribution around the keypoint in two radial Gaussian neighborhoods with different sizes. **The first local analysis** aims at attributing a reference orientation to the keypoint (the blue arrow). It is performed over a Gaussian window of standard deviation $\lambda_{\text{ori}}\sigma$ (the radius of the green circle). The width of the contributing samples patch $\mathcal{P}^{\text{ori}}$ (green square) is $6\lambda_{\text{ori}}\sigma$. The figure shows the default case $\lambda_{\text{ori}} = 1.5$. **The second analysis** aims at building the descriptor. It is performed over a Gaussian window of standard deviation $\lambda_{\text{descr}}\sigma$ (the radius of the red circle) within a square patch $\mathcal{P}^{\text{descr}}$ (the red square) of approximate width $2\lambda_{\text{descr}}\sigma$. The figure features the default settings: $\lambda_{\text{descr}} = 6$, with a Gaussian window of standard deviation 6σ and a patch $\mathcal{P}^{\text{descr}}$ of width 12σ.

The SIFT descriptor is built from the normalized image gradient orientation in the form of quantized histograms. In what follows, we describe how the reference orientation specific to each keypoint is defined and computed.

3.4.1 Keypoint reference orientation

A dominant gradient orientation over a keypoint neighborhood is used as its reference orientation. This allows for orientation normalization and hence rotation-invariance of the resulting descriptor (see Figure 3.7). Computing this reference orientation involves three steps:

A. accumulation of the local distribution of the gradient angle within a normalized patch in an orientation histogram;

B. smoothing of the orientation histogram;

C. extraction of one or more reference orientations from the smoothed histogram.

A. Orientation histogram accumulation. Given a keypoint (x, y, σ), the patch to be analyzed is extracted from the image of the scale-space $\mathbf{v}_s^o$, whose scale σ_s^o is nearest to σ. This normalized patch, denoted by $\mathcal{P}^{\text{ori}}$, is the set of pixels (m, n) of $\mathbf{v}_s^o$ satisfying

$$\max(|\delta_o m - x|, |\delta_o n - y|) \leq 3\lambda_{\text{ori}}\sigma. \tag{3.19}$$

Keypoints whose distance to the image borders is less than $3\lambda_{\text{ori}}\sigma$ are discarded since the patch $\mathcal{P}^{\text{ori}}$ is not totally included in the image. The orientation histogram h from which the dominant orientation is found covers the range $[0, 2\pi]$. It is composed of n_{bins} bins with centers $\theta_k = {}^{2\pi k}/_{n_{\text{bins}}}$. Each pixel (m, n) in $\mathcal{P}^{\text{ori}}$ will contribute to the histogram with a total weight of $c^{\text{ori}}_{m,n}$, which is the product of the gradient norm and a Gaussian weight of standard deviation $\lambda_{\text{ori}}\sigma$ (default value $\lambda_{\text{ori}} = 1.5$) reducing the contribution of distant pixels.

$$c^{\text{ori}}_{m,n} = e^{-\frac{\|(m\delta_o, n\delta_o)-(x,y)\|^2}{2(\lambda_{\text{ori}}\sigma)^2}} \left\| \left(\partial_m \mathbf{v}^o_{s,m,n}, \partial_n \mathbf{v}^o_{s,m,n}\right) \right\|. \tag{3.20}$$

This contribution is assigned to the nearest bin, namely the bin of index

$$b^{\text{ori}}_{m,n} = \left[\frac{n_{\text{bins}}}{2\pi} \left(\text{arctan2} \left(\partial_m \mathbf{v}^o_{s,m,n}, \partial_n \mathbf{v}^o_{s,m,n} \right) \right) \right]. \tag{3.21}$$

where $[\cdot]$ denotes the round function and `arctan2` is the two-argument inverse tangent function[1] with range in $[0, 2\pi]$. The gradient components of the scale-space image $\mathbf{v}^s_o$ are computed through a finite difference scheme

$$\partial_m \mathbf{v}^o_{s,m,n} = \frac{1}{2}\left(\mathbf{v}^o_{s,m+1,n} - \mathbf{v}^o_{s,m-1,n}\right), \qquad \partial_n \mathbf{v}^o_{s,m,n} = \frac{1}{2}\left(\mathbf{v}^o_{s,m,n+1} - \mathbf{v}^o_{s,m,n-1}\right), \tag{3.22}$$

for $m = 1, \ldots, M_o - 2$ and $n = 1, \ldots, N_o - 2$.

B. Smoothing the histogram. After being accumulated, the orientation histogram is smoothed by applying six times a circular convolution with the three-tap box filter $[1, 1, 1]/3$.

C. Extraction of reference orientation(s). The keypoint reference orientations are taken among the local maxima positions of the smoothed histogram. More precisely, the reference orientations are the positions of local maxima larger than t times the global maximum (default value $t = 0.8$). Let $k \in \{1, \ldots, n_{\text{bins}}\}$ be the index of a bin such that $h_k > h_{k^-}$, $h_k > h_{k^+}$, where $k^- = (k-1) \bmod n_{\text{bins}}$ and $k^+ = (k+1) \bmod n_{\text{bins}}$ and such that $h_k \geq t \max(h)$. This bin is centered on orientation $\theta_k = {}^{2\pi(k-1)}/_{n_{\text{bins}}}$. The corresponding keypoint reference orientation θ_{ref} is computed from the maximum position of the quadratic function that interpolates the values h_{k^-}, h_k, h_{k^+},

$$\theta_{\text{ref}} = \theta_k + \frac{\pi}{n_{\text{bins}}} \left(\frac{h_{k^-} - h_{k^+}}{h_{k^-} - 2h_k + h_{k^+}} \right). \tag{3.23}$$

Each one of the extracted reference orientations leads to the computation of one local descriptor of a keypoint neighborhood. The number of descriptors may consequently exceed the number of keypoints. Figure 3.8 illustrates how a reference orientation is attibuted to a keypoint.

[1] The two-argument inverse tangent, unlike the single argument one, determines the appropriate quadrant of the computed angle thanks to the extra information about the signs of the inputs: $\text{arctan2}(x, y) = \arctan({}^x/_y) + \frac{\pi}{2}\text{sign}(y)(1 - \text{sign}(x))$.

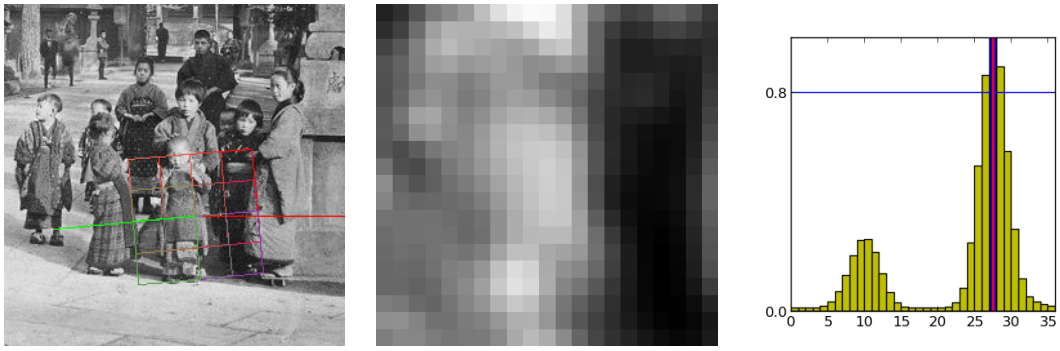


Figure 3.8: Reference orientation attribution. The normalized patch $\mathcal{P}^{\text{ori}}$ (normalized to scale and translation) width is $6\lambda_{\text{ori}}\sigma_{\text{key}}$. The gradient magnitude is weighted by a Gaussian window of standard deviation $\lambda_{\text{ori}}\sigma_{\text{key}}$. The gradient orientations are accumulated into an orientation histogram h which is subsequently smoothed.

3.4.2 Keypoint normalized descriptor

The SIFT descriptor encodes the local spatial distribution of the gradient orientation on a particular neighborhood. SIFT descriptors can be computed anywhere, even densely over the entire image or its scale-space [Liu et al. 2008; Hassner et al. 2012]. In the original SIFT method, however, descriptors are computed for all detected keypoints over its normalized neighborhood, making them invariant to translations, rotations and zoom-outs. Given a detected keypoint, the normalized neighborhood consists in a square patch centered on the keypoint and aligned with the reference orientation.

The descriptor consists in a set of weighted histograms of the gradient orientation computed on different regions of the normalized square patch.

The normalized patch. For each keypoint $(x_{\text{key}}, y_{\text{key}}, \sigma_{\text{key}}, \theta_{\text{key}})$, a normalized patch is isolated from the Gaussian scale-space image relative to the nearest discrete scale (o, s) to scale σ_{key}, namely $\mathbf{v}_s^o$. A sample (m, n) in $\mathbf{v}_s^o$, of coordinates $(x_{m,n}, y_{m,n}) = (m\delta^o, n\delta^o)$ with respect to the sampling grid of the input image, has normalized coordinates $(\hat{x}_{m,n}, \hat{y}_{m,n})$ with respect to the keypoint $(x_{\text{key}}, y_{\text{key}}, \sigma_{\text{key}}, \theta_{\text{key}})$,

$$\begin{aligned}\hat{x}_{m,n} &= \left((m\delta_o - x_{\text{key}})\cos\theta_{\text{key}} + (n\delta_o - y_{\text{key}})\sigma_{\text{in}}\theta_{\text{key}}\right)/\sigma_{\text{key}},\\ \hat{y}_{m,n} &= \left(-(m\delta_o - x_{\text{key}})\sigma_{\text{in}}\theta_{\text{key}} + (n\delta_o - y_{\text{key}})\cos\theta_{\text{key}}\right)/\sigma_{\text{key}}.\end{aligned} \tag{3.24}$$

The normalized patch denoted by $\mathcal{P}^{\text{descr}}$ is the set of samples (m, n) of $\mathbf{v}_s^o$ with normalized coordinates $(\hat{x}_{m,n}, \hat{y}_{m,n})$ satisfying

$$\max(|\hat{x}_{m,n}|, |\hat{y}_{m,n}|) \leq \lambda_{\text{descr}}. \tag{3.25}$$

Keypoints whose distance to the image borders is less than $\sqrt{2}\lambda_{\text{descr}}\sigma$ are discarded to guaranty that the patch $\mathcal{P}^{\text{descr}}$ is included in the image. Note that no image re-sampling is performed. Each sample (m, n) is characterized by the gradient orientation normalized with respect to the keypoint orientation θ_{key},

$$\hat{\theta}_{m,n} = \text{arctan2}\left(\partial_m \mathbf{v}^o_{s,m,n}, \partial_n \mathbf{v}^o_{s,m,n}\right) - \theta_{\text{key}} \bmod 2\pi, \tag{3.26}$$

and its total contribution $c^{\text{descr}}_{m,n}$. The total contribution is the product of the gradient norm and a Gaussian weight (with standard deviation $\lambda_{\text{descr}}\sigma_{\text{key}}$) reducing the contribution of distant pixels,

$$c^{\text{descr}}_{m,n} = e^{-\frac{\|(m\delta^o, n\delta^o)-(x,y)\|^2}{2(\lambda_{\text{descr}}\sigma)^2}} \left\| \left(\partial_m \mathbf{v}^o_{s,m,n}, \partial_n \mathbf{v}^o_{s,m,n}\right) \right\|. \tag{3.27}$$

The array of orientation histograms. The gradient orientation of each pixel in the normalized patch $\mathcal{P}^{\text{descr}}$ is accumulated into an array of $n_{\text{hist}} \times n_{\text{hist}}$ orientation histograms (default value $n_{\text{hist}} = 4$). Each of these histograms, denoted by $h^{i,j}$ for $(i,j) \in \{1,\dots,n_{\text{hist}}\}^2$, has an associated position with respect to the keypoint $(x_{\text{key}}, y_{\text{key}}, \sigma_{\text{key}}, \theta_{\text{key}})$, given by

$$\hat{x}^i = \left(i - \frac{1+n_{\text{hist}}}{2}\right)\frac{2\lambda_{\text{descr}}}{n_{\text{hist}}} \quad , \quad \hat{y}^j = \left(j - \frac{1+n_{\text{hist}}}{2}\right)\frac{2\lambda_{\text{descr}}}{n_{\text{hist}}}.$$

Each histogram $h^{i,j}$ consists of n_{ori} bins $h^{i,j}_k$ with $k \in \{1,\dots,n_{\text{ori}}\}$, centered at $\hat{\theta}^k = 2\pi(k-1)/n_{\text{ori}}$ (default value $n_{\text{ori}} = 8$). Each sample (m,n) in the normalized patch $\mathcal{P}^{\text{descr}}$ contributes to the nearest histograms (up to four histograms). Its total contribution $c^{\text{descr}}_{m,n}$ is split bilinearly over the nearest histograms depending on the distances to each of them (see Figure 3.10). In the same way, the contribution within each histogram is subsequently split linearly between the two nearest bins. This results, for the sample (m,n), in the following updates.

For every $(i,j,k) \in \{1,\dots,n_{\text{hist}}\}^2 \times \{1,\dots,n_{\text{ori}}\}$ such that

$$|\hat{x}^i - \hat{x}_{m,n}| \le \frac{2\lambda_{\text{descr}}}{n_{\text{hist}}}, \quad |\hat{y}^j - \hat{y}_{m,n}| \le \frac{2\lambda_{\text{descr}}}{n_{\text{hist}}} \quad \text{and} \quad |\hat{\theta}^k - \hat{\theta}_{m,n} \bmod 2\pi| \le \frac{2\pi}{n_{\text{ori}}},$$

the histogram $h^{i,j}_k$ is updated by

$$h^{i,j}_k \leftarrow h^{i,j}_k + \left(1 - \frac{n_{\text{hist}}}{2\lambda_{\text{descr}}}\left|\hat{x}^i - \hat{x}_{m,n}\right|\right)\left(1 - \frac{n_{\text{hist}}}{2\lambda_{\text{descr}}}\left|\hat{y}^j - \hat{y}_{m,n}\right|\right)\left(1 - \frac{n_{\text{ori}}}{2\pi}\left|\theta^k - \hat{\theta}_{m,n} \bmod 2\pi\right|\right) c^{\text{descr}}_{m,n}. \tag{3.28}$$



Figure 3.9: SIFT descriptor construction. No explicit re-sampling of the described normalized patch is performed. The normalized patch $\mathcal{P}^{\text{descr}}$ is partitioned into a set of $n_{\text{hist}} \times n_{\text{hist}}$ subpatches (default value $n_{\text{hist}} = 4$). Each sample (m,n) inside $\mathcal{P}^{\text{descr}}$, located at $(m\delta^o, n\delta^o)$, contributes by an amount that is a function of their normalized coordinates $(\hat{x}_{m,n}, \hat{y}_{m,n})$ (see (3.24)). Each sub-patch $\mathcal{P}^{\text{descr}}_{(i,j)}$ is centered at $(\hat{x}_i, \hat{y}_j)$.

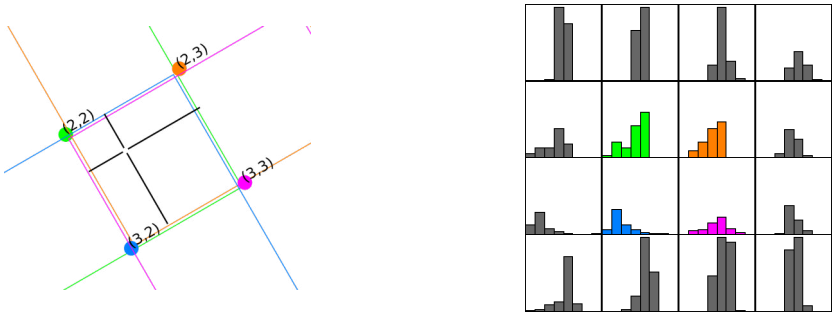


Figure 3.10: Illustration of the spatial contribution of a sample inside the patch $\mathcal{P}^{\text{descr}}$. The sample (m, n) contributes to the weighted histograms $(2, 2)$ (green), $(2, 3)$ (orange), $(3, 2)$ (blue) and $(3, 3)$ (pink); The contribution $c^{\text{descr}}_{m,n}$ is split over four pairs of bins according to (3.28).

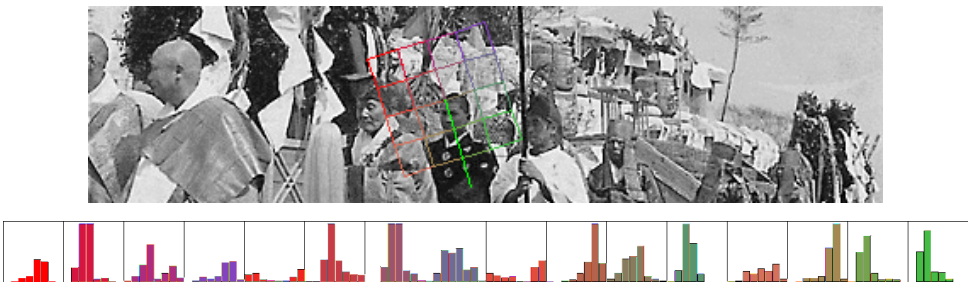


Figure 3.11: The image on top shows the $n_{\text{hist}} \times n_{\text{hist}}$ array sub-patches relative to a keypoint; the corresponding n_{ori} bins histograms are rearranged into a 1D-vector $\vec{v}$ (bottom). This vector is subsequently thresholded and normalized so that the Euclidean norm equals 512 for each descriptor. The dimension of the feature vector in this example is 128, relative to parameter $n_{\text{hist}} = 4$, $n_{\text{ori}} = 8$ (default values).

The SIFT feature vector. The accumulated array of histograms are encoded into a vector feature $\mathbf{f}$ of length $n_{\text{hist}} \times n_{\text{hist}} \times n_{\text{ori}}$, as follows:

$$\mathbf{f}_{(i-1)n_{\text{hist}}n_{\text{ori}}+(j-1)n_{\text{ori}}+k} = h^{i,j}_k,$$

where $i = 1, \ldots, n_{\text{hist}}$, $j = 1, \ldots, n_{\text{hist}}$ and $k = 1, \ldots, n_{\text{ori}}$. The components of the feature vector $\mathbf{f}$ are saturated to a maximum value of 20% of its Euclidean norm, i.e., $\mathbf{f}_k \leftarrow \min(\mathbf{f}_k, 0.2\|\mathbf{f}\|)$. The saturation of the feature vector components seeks to reduce the impact of non-linear illumination changes, such as saturated regions.

The vector is finally renormalized so as to have $\|\mathbf{f}\|_2 = 512$ and quantized to 8 bit integers as follows: $\mathbf{f}_k \leftarrow \min(\lfloor \mathbf{f}_k \rfloor, 255)$. The quantization aims at accelerating the computation of distances between different feature vectors[2]. Figure 3.9 and Figure 3.11 illustrate how a SIFT feature vector is attibuted to an oriented keypoint.

[2] The executable provided by D.Lowe `http://www.cs.ubc.ca/~lowe/keypoints/`, retrieved on September 11th, 2014) uses a different coordinate system (see source code's `README.txt` for details).

3.4.3 Pseudocodes

Algorithm 14: Computation of the 2D gradient at each image of the scale-space

Input: $(\mathbf{v}_s^o)$, digital Gaussian scale-space, $o = 1, \ldots, n_{\text{oct}}$ and $s = 0, \ldots, n_{\text{spo}} + 2$.
Outputs: - $(\partial_m \mathbf{v}_{s,m,n}^o)$, scale-space gradient along x, $o = 1, \ldots, n_{\text{oct}}$ and $s = 1, \ldots, n_{\text{spo}}$.
- $(\partial_n \mathbf{v}_{s,m,n}^o)$, scale-space gradient along y, $o = 1, \ldots, n_{\text{oct}}$ and $s = 1, \ldots, n_{\text{spo}}$.

for $o = 1, \ldots, n_{oct}$ **and** $s = 1, \ldots, n_{spo}$ **do**
  for $m = 1, \ldots, M_o - 2$ **and** $n = 1, \ldots, N_o - 2$ **do**
    $\partial_m \mathbf{v}_{s,m,n}^o = (\mathbf{v}_{s,m+1,n}^o - \mathbf{v}_{s,m-1,n}^o)/2$
    $\partial_n \mathbf{v}_{s,m,n}^o = (\mathbf{v}_{s,m,n+1}^o - \mathbf{v}_{s,m,n-1}^o)/2$

Algorithm 15: Computing the keypoint reference orientation

Inputs: - $\mathcal{L}_{\mathrm{C}} = \{(o_{\mathrm{key}}, s_{\mathrm{key}}, x_{\mathrm{key}}, y_{\mathrm{key}}, \sigma_{\mathrm{key}}, \boldsymbol{\omega})\}$, list of keypoints.
- $(\partial_m \mathbf{v}^o_{s,m,n})$, scale-space gradient along x, $o = 1, \ldots, n_{\mathrm{oct}}$ and $s = 1, \ldots, n_{\mathrm{spo}}$.
- $(\partial_n \mathbf{v}^o_{s,m,n})$, scale-space gradient along y, $o = 1, \ldots, n_{\mathrm{oct}}$ and $s = 1, \ldots, n_{\mathrm{spo}}$.

Parameters: - λ_{ori}. The patch $\mathcal{P}^{\mathrm{ori}}$ is $6\lambda_{\mathrm{ori}}\sigma$ wide.
The Gaussian window has a standard deviation of $\lambda_{\mathrm{ori}}\sigma$.
- n_{bins}, number of bins in the orientation histogram h.
- t, threshold for secondary reference orientations.

Output: $\mathcal{L}_{\mathrm{D}} = \{(o, s', m', n', x, y, \sigma, \boldsymbol{\omega}, \theta)\}$ list of oriented keypoints.

Temporary: h_k, orientation histogram, $k = 1, \ldots, n_{\mathrm{bins}}$ and with h_k covering $[\frac{2\pi(k-3/2)}{n_{\mathrm{bins}}}; \frac{2\pi(k-1/2)}{n_{\mathrm{bins}}}]$.

for each *keypoint* $(o_{key}, s_{key}, x_{key}, y_{key}, \sigma_{key}, \boldsymbol{\omega})$ *in* $\mathcal{L}_C$ **do**

`// Check if the keypoint is distant enough from the image borders`

if $3\lambda_{ori}\sigma \leq x_{key} \leq h - 3\lambda_{ori}\sigma$ **and** $3\lambda_{ori}\sigma \leq y_{key} \leq w - 3\lambda_{ori}\sigma$ **then**

`// Initialize the orientation histogram` h

for $1 \leq k \leq n_{bins}$ **do** $h_k \leftarrow 0$

`// Accumulate samples from the normalized patch` $\mathcal{P}^{\mathtt{ori}}$ `(eq.(3.19).`

for $m = [(x_{key} - 3\lambda_{ori}\sigma_{key})/\delta_{o_{key}}], \ldots, [(x_{key} + 3\lambda_{ori}\sigma_{key})/\delta_{o_{key}}]$ **do**

for $n = [(y_{key} - 3\lambda_{ori}\sigma_{key})/\delta_{o_{key}}], \ldots, [(y_{key} + 3\lambda_{ori}\sigma_{key})/\delta_{o_{key}}]$ **do**

`// Compute the sample contribution`

$$c^{\mathrm{ori}}_{m,n} = e^{-\frac{\|(m\delta_{o_{\mathrm{key}}}, n\delta_{o_{\mathrm{key}}}) - (x_{\mathrm{key}}, y_{\mathrm{key}})\|^2}{2(\lambda_{\mathrm{ori}}\sigma_{\mathrm{key}})^2}} \left\| \left(\partial_m \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}, \partial_n \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}\right) \right\|$$

`// Compute the corresponding bin index`

$$b^{\mathrm{ori}}_{m,n} = \left[\tfrac{n_{\mathrm{bins}}}{2\pi} \left(\arctan2\left(\partial_m \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}, \partial_n \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}\right) \bmod 2\pi\right)\right]$$

`// Update the histogram`

$$h_{b^{\mathrm{ori}}_{m,n}} \leftarrow h_{b^{\mathrm{ori}}_{m,n}} + c^{\mathrm{ori}}_{m,n}$$

`// Smooth` h

Apply six times a circular convolution with filter $[1, 1, 1]/3$ to h.

`// Extract the reference orientations`

for $1 \leq k \leq n_{bins}$ **do**

if $h_k > h_{k^-}$, $h_k > h_{k^+}$ *and* $h_k \geq t \max(h)$ **then**

`// Compute the reference orientation` $\theta_{\mathtt{key}}$

$$\theta_{\mathrm{key}} = \theta_k + \tfrac{\pi}{n_{\mathrm{bins}}} \left(\tfrac{h_{k^-} - h_{k^+}}{h_{k^-} - 2h_k + h_{k^+}}\right)$$

note: $[\cdot]$ denotes the round function and arctan2 denotes the two-argument inverse tangent.

Algorithm 16: Construction of the keypoint descriptor

Inputs: - $\mathcal{L}_{\mathrm{D}} = \{(o_{\mathrm{key}}, s_{\mathrm{key}}, x_{\mathrm{key}}, y_{\mathrm{key}}, \sigma_{\mathrm{key}}, \theta_{\mathrm{key}})\}$ list of keypoints.
- $(\partial_m \mathbf{v}^o_{s,m,n})$, scale-space gradient along x.
- $(\partial_n \mathbf{v}^o_{s,m,n})$, scale-space gradient along y (see Algorithm 14).

Output: $\mathcal{L}_{\mathrm{E}} = \{(o_{\mathrm{key}}, s_{\mathrm{key}}, x_{\mathrm{key}}, y_{\mathrm{key}}, \sigma_{\mathrm{key}}, \theta_{\mathrm{key}}, \mathbf{f})\}$ list of keypoints with feature vector $\mathbf{f}$.

Parameters: - n_{hist}. The descriptor is an array of $n_{\mathrm{hist}} \times n_{\mathrm{hist}}$ orientation histograms.
- n_{ori}, number of bins in the orientation histograms.
Feature vectors $\mathbf{f}$ have a length of $n_{\mathrm{hist}} \times n_{\mathrm{hist}} \times n_{\mathrm{ori}}$
- λ_{descr}.
The Gaussian window has a standard deviation of $\lambda_{\mathrm{descr}}\sigma_{\mathrm{key}}$.
The patch $\mathcal{P}^{\mathrm{descr}}$ is $2\lambda_{\mathrm{descr}}\frac{n_{\mathrm{hist}}+1}{n_{\mathrm{hist}}}\sigma_{\mathrm{key}}$ wide.

Temporary: $h_k^{i,j}$, array of orientation weighted histograms, $(i,j) \in \{1,\dots,n_{\mathrm{hist}}\}$ and $k \in \{1,\dots,n_{\mathrm{ori}}\}$

for each *keypoint* $(o_{key}, s_{key}, x_{key}, y_{key}, \sigma_{key}, \theta_{key})$ *in* $\mathcal{L}_D$ **do**

`// Check if the keypoint is distant enough from the image borders`
if $\sqrt{2}\lambda_{descr}\sigma \le x_{key} \le h - \sqrt{2}\lambda_{descr}\sigma$ **and** $\sqrt{2}\lambda_{descr}\sigma \le y_{key} \le w - \sqrt{2}\lambda_{descr}\sigma$ **then**

`// Initialize the array of weighted histograms`
for $1 \le i \le n_{hist}$, $1 \le j \le n_{hist}$ **and** $1 \le k \le n_{ori}$ **do** $h_k^{i,j} \leftarrow 0$

`// Accumulate samples of normalized patch` $\mathcal{P}^{\mathtt{descr}}$ `in the array histograms (eq.(3.25))`
for $m = \left[\left(x_{key} - \sqrt{2}\lambda_{descr}\sigma_{key}\frac{n_{hist}+1}{n_{hist}}\right)/\delta_o\right], \dots, \left[\left(x_{key} + \sqrt{2}\lambda_{descr}\sigma_{key}\frac{n_{hist}+1}{n_{hist}}\right)/\delta_o\right]$ **do**
for $n = \left[\left(y_{key} - \sqrt{2}\lambda_{descr}\sigma_{key}\frac{n_{hist}+1}{n_{hist}}\right)/\delta_o\right], \dots, \left[\left(y_{key} + \sqrt{2}\lambda_{descr}\sigma_{key}\frac{n_{hist}+1}{n_{hist}}\right)/\delta_o\right]$ **do**

`// Compute normalized coordinates (eq.(3.24)).`
$\hat{x}_{m,n} = \left((m\delta_{o_{\mathrm{key}}} - x_{\mathrm{key}})\cos\theta_{\mathrm{key}} + (n\delta_{o_{\mathrm{key}}} - y_{\mathrm{key}})\sigma_{\mathrm{in}}\theta_{\mathrm{key}}\right)/\sigma_{\mathrm{key}}$
$\hat{y}_{m,n} = \left(-(m\delta_{o_{\mathrm{key}}} - x_{\mathrm{key}})\sigma_{\mathrm{in}}\theta_{\mathrm{key}} + (n\delta_{o_{\mathrm{key}}} - y_{\mathrm{key}})\cos\theta_{\mathrm{key}}\right)/\sigma_{\mathrm{key}}$

`// Verify if the sample` (m,n) `is inside the normalized patch` $\mathcal{P}^{\mathtt{descr}}$.
if $\max(|\hat{x}_{m,n}|, |\hat{y}_{m,n}|) < \lambda_{descr}\frac{n_{hist}+1}{n_{hist}}$ **then**

`// Compute normalized gradient orientation.`
$\hat{\theta}_{m,n} = \arctan2\left(\partial_m \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}, \partial_n \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}\right) - \theta_{\mathrm{key}} \bmod 2\pi$

`// Compute the total contribution of the sample` (m,n)
$c^{\mathrm{descr}}_{m,n} = e^{-\frac{\|(m\delta^{o_{\mathrm{key}}}, n\delta^{o_{\mathrm{key}}}) - (x_{\mathrm{key}}, y_{\mathrm{key}})\|^2}{2(\lambda_{\mathrm{descr}}\sigma_{\mathrm{key}})^2}} \left\|\left(\partial_m \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}, \partial_n \mathbf{v}^{o_{\mathrm{key}}}_{s_{\mathrm{key}},m,n}\right)\right\|$

`// Update the nearest histograms and the nearest bins (eq.(3.28)).`
for $(i,j) \in \{1,\dots,n_{hist}\}^2$ *such that* $\left|\hat{x}^i - \hat{x}_{m,n}\right| \le \frac{2\lambda_{descr}}{n_{hist}}$ *and* $\left|\hat{y}^j - \hat{y}_{m,n}\right| \le \frac{2\lambda_{descr}}{n_{hist}}$ **do**
for $k \in \{1,\dots,n_{ori}\}$ *such that* $\left|\hat{\theta}^k - \hat{\theta}_{m,n} \bmod 2\pi\right| < \frac{2\pi}{n_{ori}}$ **do**
$h_k^{i,j} \leftarrow h_k^{i,j} +$
$\left(1 - \frac{n_{\mathrm{hist}}}{2\lambda_{\mathrm{descr}}}|\hat{x}_{m,n} - \hat{x}^i|\right)\left(1 - \frac{n_{\mathrm{hist}}}{2\lambda_{\mathrm{descr}}}|\hat{y}_{m,n} - \hat{y}^j|\right)\left(1 - \frac{n_{\mathrm{ori}}}{2\pi}|\hat{\theta}_{m,n} - \hat{\theta}^k \bmod 2\pi|\right) c^{\mathrm{descr}}_{m,n}$

`// Build the feature vector` $\mathbf{f}$ `from the array of weighted histograms.`
for $1 \le i \le n_{hist}$, $1 \le j \le n_{hist}$ **and** $1 \le k \le n_{ori}$ **do**
$\mathbf{f}_{(i-1)n_{\mathrm{hist}}n_{\mathrm{ori}} + (j-1)n_{\mathrm{ori}} + k} = h_k^{i,j}$
for $1 \le l \le n_{hist} \times n_{hist} \times n_{ori}$ **do**
$\mathbf{f}_l \leftarrow \min(\mathbf{f}_l, 0.2\|\mathbf{f}\|)$ `/*normalize and threshold f*/`
compute the l2 norm $\mathbf{f}_l \leftarrow \min(\lfloor 512\mathbf{f}_l/\|\mathbf{f}\|\rfloor, 255)$ `/*quantize to 8 bit integers*/`

Add $(x, y, \sigma, \theta, \mathbf{f})$ to $\mathcal{L}_{\mathrm{E}}$

3.5 Matching

The classical purpose of detecting and describing keypoints is to find matches (pairs of keypoints) between images. In the absence of extra knowledge on the problem, for instance in the form of geometric constraints, a matching procedure generally consists of two steps: the pairing of similar keypoints from respective images and the selection of those that are reliable. In what follows, we present the matching method described in the original article by D. Lowe [Lowe 2004]. This is also the matching method we will use in Chapter 5 for the performance evaluation of various feature detectors. Let $\mathcal{L}^A$ and $\mathcal{L}^B$ be the set of descriptors associated to the keypoints detected in images $\mathbf{u}^A$ and $\mathbf{u}^B$. The matching is done by considering every descriptor associated to the list $\mathcal{L}^A$ and finding one possible match in list $\mathcal{L}^B$. The first descriptor $\mathbf{f}^a \in \mathcal{L}^A$ is paired to the descriptor $\mathbf{f}^b \in \mathcal{L}^B$ that minimizes the Euclidean distance between descriptors,

$$\mathbf{f}^b = \arg\min_{\mathbf{f} \in \mathcal{L}^B} \|\mathbf{f} - \mathbf{f}^a\|_2.$$

Pairing a keypoint with descriptor $\mathbf{f}^a$ requires then to compute distances to all descriptors in $\mathcal{L}^B$. A pair is considered reliable only if its absolute distance is below a certain threshold $C^{\text{match}}_{\text{absolute}}$. Otherwise it is discarded.

To avoid dependence to an absolute distance, the SIFT method uses the second nearest neighbor to define what constitutes a reliable match. SIFT applies an adaptive thresholding $\|\mathbf{f}^a - \mathbf{f}^{b'}\| C^{\text{match}}_{\text{relative}}$, where $\mathbf{f}_{b'}$ is the second nearest neighbor,

$$\mathbf{f}^{b'} = \arg\min_{\mathbf{f} \in \mathcal{L}^B \setminus \{\mathbf{f}^b\}} \|\mathbf{f} - \mathbf{f}^a\|_2.$$

This is detailed in Algorithm 17. The major drawback of using a relative threshold is that it omits detections for keypoints associated to a repeated structure in the image. Indeed, in that case, the distance to the nearest and the second nearest descriptor would be comparable. More sophisticated techniques have been developed to allow robust matching of images with repeated structures [Rabin et al. 2009].

This matching algorithm runs in time $c \cdot N_A \cdot N_B$, where N_A and N_B are the number of keypoints in images $\mathbf{u}^A$ and $\mathbf{u}^B$ respectively, and c is a constant proportional to the time that takes to compare two SIFT features. This is prohibitively slow for images of moderate size, although keypoint matching is highly parallelizable. A better solution is to use more compact descriptors [Tuytelaars and Mikolajczyk 2008] that reduce the cost of distance computation (and thus reduce the value of c). Among the proposed solutions we can find more compact SIFT-like descriptors [Bay et al. 2006b; Ke and Sukthankar 2004] or binary descriptors [Calonder et al. 2010; Rublee et al. 2011; Raginsky and Lazebnik 2009] which take advantage of the fast computation of the Hamming distance between two binary vectors.

3.6 Summary of Parameters

The online demo provided with this publication examines in detail the behavior of each stage of the SIFT algorithm. In what follows, we summarize all the parameters that can be adjusted in the demo.

Algorithm 17: Matching keypoints

Inputs: - $\mathcal{L}^A = \{(x^a, y^a, \sigma^a, \theta^a, \mathbf{f}^a)\}$ keypoints and descriptors relative to image $\mathbf{u}^A$.
- $\mathcal{L}^B = \{(x^b, y^b, \sigma^b, \theta^b, \mathbf{f}^b)\}$ keypoints and descriptors relative to image $\mathbf{u}^B$.

Output: $\mathcal{M} = \{(x^a, y^a, \sigma^a, \theta^a, \mathbf{f}^a), (x^b, y^b, \sigma^b, \theta^b, \mathbf{f}^b)\}$ list of matches with positions.

Parameter: $C^{\text{match}}_{\text{relative}}$ relative threshold

for each *descriptor* $\mathbf{f}^a$ *in* $\mathcal{L}^A$ **do**
- Find $\mathbf{f}^b$ and $\mathbf{f}^{b'}$, nearest and second nearest neighbors of $\mathbf{f}^a$:
- **for each** *descriptor* $\mathbf{f}$ *in* $\mathcal{L}^B$ **do**
  - Compute distance $d(\mathbf{f}^a, \mathbf{f})$
- Select pairs satisfying a relative threshold.
- **if** $d(\mathbf{f}^a, \mathbf{f}^b) < C^{match}_{relative}\, d(\mathbf{f}^a, \mathbf{f}^{b'})$ **then**
  - Add pair $(\mathbf{f}^a, \mathbf{f}^b)$ to $\mathcal{M}$

Digital scale-space configuration and keypoints detection

Parameter	Default value	Description
σ_{min}	0.8	blur level of $\mathbf{v}^1_0$ (seed image)
δ_{min}	0.5	the sampling distance in image $\mathbf{v}^1_0$ (corresponds to a $2\times$ interpolation)
σ_{in}	0.5	assumed blur level in $\mathbf{u}^{\text{in}}$ (input image)
n_{oct}	8	number of octaves (limited by the image size)) $\lfloor \log_2(\min(w,h)/\delta_{\text{min}}/12) + 1 \rfloor$
n_{spo}	3	number of scales per octave
C_{DoG}	0.015	threshold over the DoG response (value relative to $n_{\text{spo}} = 3$)
C_{edge}	10	threshold over the ratio of principal curvatures (edgeness).

Table 3.3: Parameters of the scale-space discretization and detection of SIFT keypoints.

The structure of the digital scale-space is unequivocally characterized by four structural parameters: n_{oct}, n_{spo}, σ_{min}, δ_{min} and by the blur level in the input image σ_{in}. The associated online demo allows the user to change these values. They can be tuned to satisfy specific requirements[3]. For example, increasing the number of scales per octave n_{spo} and the initial interpolation factor δ_{min} increases the number of detections. On the other hand, reducing them results in a faster algorithm.

The image structures that are potentially detected by SIFT have a scale ranging from σ_{min} to $\sigma_{\text{min}} 2^{n_{\text{oct}}}$. Therefore, it may seem natural to choose the lowest possible value of σ_{min} (i.e., $\sigma_{\text{min}} = \sigma_{\text{in}}$). However, depending on the input image sharpness, low scale detections may be the result of aliasing artifacts and should be avoided. Thus, a sound setting of parameter σ_{min} should take into account the image blur level σ_{in} and the possible presence of image aliasing.

The DoG thresholding, controlled by C_{DoG}, was conceived to filter detections due to noise. With that aim in view, C_{DoG} should depend on the input image signal to noise ratio. It is however beyond the scope of this publication to analyze the soundness of such an approach. We will only point out that the reduction of C_{DoG} increases the number of detected keypoints. Recall that the DoG approximates $(2^{1/n_{\text{spo}}} - 1)\sigma^2 \Delta v$, its values depends on the number of scales per octave n_{spo}. The threshold applied in the provided source-code

[3]The number of computed octaves is upper limited to ensure that images in the last octave are at least 12×12 pixels.

is $\widetilde{C}_{\mathrm{DoG}} = \frac{2^{1/n_{\mathrm{spo}}}-1}{2^{1/3}-1} C_{\mathrm{DoG}}$, with C_{DoG} relative to $n_{\mathrm{spo}} = 3$. Section 4.5 in Chapter 4 will investigate how efficient this step is at discarding unstable keypoints. Section 4.5 in the next chapter will investigate how efficient this step is at discarding unstable keypoints.

The threshold C_{edge}, applied to discard keypoints laying on edges, has in practice a negligible impact on the algorithm performance. Indeed, many candidate keypoints laying on edges were previously discarded during the extrema refinement.

Computation of the SIFT descriptor

The provided demo shows the computation of the keypoint reference orientation, and also the construction of the feature vector for any detected keypoint.

Parameter	Default value	Description
n_{bins}	36	number of bins in the gradient orientation histogram
λ_{ori}	1.5	sets how local the analysis of the gradient distribution is: - Gaussian window of standard deviation $\lambda_{\mathrm{ori}}\sigma$ - patch width $6\lambda_{\mathrm{ori}}\sigma$
t	0.80	threshold for considering local maxima in the gradient orientation histogram
n_{hist}	4	number of histograms in the normalized patch is $(n_{\mathrm{hist}} \times n_{\mathrm{hist}})$
n_{ori}	8	number of bins in the descriptor histograms the feature vectors dimension is $n_{\mathrm{hist}} \times n_{\mathrm{hist}} \times n_{\mathrm{ori}}$
λ_{descr}	6	sets how local the descriptor is: - Gaussian window of standard deviation $\lambda_{\mathrm{descr}}\sigma$ - descriptor patch width $2\lambda_{\mathrm{descr}}\sigma$

Table 3.4: Parameters of the computation of the SIFT feature vectors.

The parameter λ_{ori} controls how local the computation of the reference orientation is. Localizing the gradient analysis generally results in an increased number of orientation references. Indeed, the orientation histogram generated from an isotropic structure is almost flat and therefore has many local maxima. Another algorithm design parameter, not included in Table 3.4 because of its insignificant impact, is the level of smoothing applied to the histogram ($N_{\mathrm{conv}} = 6$).

The size of the normalized patch used for computing the SIFT descriptor is governed by λ_{descr}. A larger patch will produce a more discriminative descriptor but will be less robust to scene deformation. The number of histograms $n_{\mathrm{hist}} \times n_{\mathrm{hist}}$ and the number of bins n_{ori} can be set to make the feature vector more compact. These architectural parameters govern the trade off between robustness and discrimination.

Matching of SIFT feature vectors

The SIFT algorithm consists of the detection of image keypoints and their description. The demo provides additionally two naive algorithms to match SIFT features. The first one applies an absolute threshold on the distance to the nearest keypoint feature to define if a match is reliable. The second one applies a relative threshold that depends on the distance to the second nearest keypoint feature.

Increasing the absolute threshold $C^{\mathrm{match}}_{\mathrm{absolute}}$ evidently reduces the number of matches. In a relative threshold scenario, increasing the threshold $C^{\mathrm{match}}_{\mathrm{relative}}$ results in an increased

number of matches. In particular, pairs corresponding to repeated structures in the image will be less likely to be omitted. However, this may lead to an increased number of false matches.

Parameter	Default value	Description
$C^{\text{match}}_{\text{absolute}}$	250 to 300	threshold on the distance to the nearest neighbor
$C^{\text{match}}_{\text{relative}}$	0.6	relative threshold between nearest and second nearest neighbors

Table 3.5: Parameters of the SIFT matching algorithm.

4 An analysis of scale-space sampling and keypoints detection in SIFT

The SIFT algorithm has proven to be sufficiently scale invariant to be used in numerous applications. In practice, however, scale invariance may be weakened by various sources of error inherent to the SIFT implementation affecting the stability and accuracy of keypoint detection. The density of the sampling of the Gaussian scale-space and the level of blur in the input image are two of these sources. This chapter presents an numerical analysis of their impact on the extracted keypoints stability. Such analysis has both methodological and practical implications, on how to compare feature detectors and on how to improve SIFT. We show that even with a significantly oversampled scale-space numerical errors prevent from achieving perfect stability. Usual strategies to filter out unstable detections are shown to be inefficient. We also prove that the effect of the error in the assumption on the initial blur is asymmetric and that the method is strongly degraded in presence of aliasing or without a correct assumption on the camera blur.

4.1 Introduction

For SIFT as well as for its numerous variants, the property of scale invariance is crucial. SIFT was proved to be theoretically scale invariant Morel and Yu [2011]. Indeed, SIFT keypoints are covariant, being the extrema of the image Gaussian scale-space Weickert et al. [1999]; Lindeberg [1993]. In practice, however, the computation of the SIFT keypoints is affected in many ways, which in turn limits the scale invariance.

The literature on SIFT focuses on variants, alternatives and accelerations Brown and Lowe [2007]; Tuytelaars and Mikolajczyk [2008]; Bay et al. [2006b]; Mikolajczyk et al. [2005]; Förstner et al. [2009]; Mainali et al. [2013]; Ancuti and Bekaert [2007]; Pele and Werman [2008]; Rabin et al. [2009]; Ke and Sukthankar [2004]; Calonder et al. [2010]; Rublee et al. [2011]; Tola et al. [2008, 2010]; Vedaldi and Fulkerson [2010]; Leutenegger et al. [2011]; Agrawal et al. [2008]; Winder and Brown [2007]; Winder et al. [2009]; Chen et al. [2010]; Grabner et al. [2006]; Liu et al. [2008]; Moreno et al. [2009]; Brown et al. [2005]; Dickscheid et al. [2011]; Sadek et al. [2012]. The huge amount of citations of SIFT indicates that it has become a standard and a reference in many applications. In contrast, there are almost no articles discussing the SIFT settings and trying to compare SIFT with itself. By this comparison we mean the question of comparing the SIFT invariance

claim with its empirical invariance, and the influence of the SIFT parameters on its own performance. On this strict subject D. Lowe's paper Lowe [2004] remains the principal reference, and it seems that very few of its claims on the parameter choices of the method have undergone a serious scrutiny. The work presented in this chapter intends to fill in the gap for the main claim of the SIFT method, namely its scale invariance, and incidentally on its translation invariance. This is investigated by means of a strict image simulation framework allowing us to control the main image and scale-space sampling parameters: initial blur, scale and space sampling rates and noise level. We show that even in a particularly favorable scenario, many of the detected SIFT keypoints are unstable. We prove that the scale-space sampling has an influence on the scale invariance and that finely sampling the Gaussian scale-space improves the detection of scale-space extrema. We quantify how the empirical invariance is affected by image aliasing and other errors due to wrong assumptions on the input image blur level.

Also, we verify the importance of the quadratic interpolation proposed in SIFT for refining the precision of the localized extrema. This is a fundamental step for the overall algorithm stability by filtering out unstable discrete extrema. On the other hand, we show that the contrast threshold proposed in SIFT is ineffective to remove the unstable detections.

We provide here a thorough and rigorous analysis of the scale-space extrema and their stability. We reach this by separating the mathematical definition of the scale-space from the numerical implementation. We also add an analysis of the difference of Gaussians (DoG) scale-space operator and a discussion on how fine the scale-space should be sampled to fulfill the SIFT invariance claim.

The remainder of this chapter is organized as follows. Section 4.2 details how for the requirements of the present analysis, the implementation of the SIFT method considered here differs from the original. Section 4.3 exposes SIFT theoretical scale invariance. With that aim in view, we explicit the camera model consistent with SIFT. Section 4.4 details how input images are simulated to be rigorously consistent with SIFT camera model. Section 4.5 explores the extraction of SIFT keypoints at each stage of the algorithms focusing on the impact of the scale-space sampling on detections.

Section 4.5 provides an empirical analysis of the scale-space sampling. Section 4.6 looks at the impact of image aliasing and of errors in the estimation of camera blur. We finally close in Section 4.7.

4.2 The exact implementation of the SIFT method

This chapter focuses on the computation of the Gaussian scale-space (Section 3.2), the detection, interpolation and filtering of 3D extrema (Section 3.3).

In this section we fix the adjustments that are required to make it ideally precise. This ideal SIFT will be used in the next sections (in place of the original implementation detailed in Chapter 3) to explore the limits for the SIFT method to detect scale-space extrema.

4.2.1 The Gaussian scale-space and its implementation

The architecture of the Gaussian scale-space requires for the Gaussian convolution to be implemented so it satisfies the semi-group property.

$$G_\sigma G_\gamma u(\mathbf{x}) = G_{\sqrt{\sigma^2+\gamma^2}} u(\mathbf{x}). \tag{4.1}$$

We have seen in Chapter 3 that the Gaussian convolution is implemented in SIFT as a discrete convolution with a sampled truncated Gaussian kernel.

Such implementation satisfies the semi-group property for the SIFT default parameters ($n_{\text{spo}} = 3$), but it fails for larger values of n_{spo}, as the level of blur to be added approaches zero. Indeed, as we demonstrated in Chapter 2, the discrete convolution fails to satisfy the semi-group property for low values of σ (i.e., $\sigma < 0.7$) because of image aliasing when sampling a Gaussian kernel with low standard deviation.

To avoid this undesired phenomenon in our experiments that will consider strong scale oversampling, we replaced the discrete convolution by a Fourier-domain based convolution using the Discrete Cosine Transform (DCT). As we have established in Chapter 2 Section 5.5, the Fourier-based convolution satisfies the semi-group property even for low values of σ. This is why we opted to use the DCT Gaussian filtering. The implementation details are given in Chapter 2, Algorithm 2.

4.2.2 Building an ideal SIFT for parameter exploration

Since our goal was to explore extrema detection, we implemented an ideal SIFT where not only the convolution is exact, but also the extrema filters were turned off. The implementation of SIFT used in the present work differs from the original one (as described in Chapter 3) on two aspects (besides the replacement of the discrete convolution by the Fourier-based one). First, SIFT proposes two filters to discard unreliable keypoints. The first one eliminates poorly contrasted extrema (those with low DoG value) and the second one discards extrema laying on edges (using a threshold on the local Hessian spectrum). These filters were deactivated to gain a full control of all detected extrema and to isolate the impact of each of them in terms of keypoints stability. This choice will be *a posteriori* justified, as we demonstrate in Section 4.5.3 that the DoG contrast threshold is inefficient.

Secondly, we decided to implement the DoG operator in such a way that the same mathematical definition is kept (κ value) regardless of the scale sampling rate (n_{spo} value). SIFT approximates the normalized Laplacian $\sigma^2\Delta$ by the difference of Gaussian operator.

Different DoG definitions lead to different extrema. Consider for instance an image with a Gaussian blob of standard deviation σ_{blob} as input. The normalized Laplacian will have an extremum at the center of the Gaussian blob, and scale $\sigma_{\text{detect}} = \sigma_{\text{blob}}$. On the other hand, the DoG scale-space of parameter κ yields an extremum at scale $\sigma_{\text{detect}} = \sigma_{\text{blob}}/\sqrt{\kappa}$. Consequently, the range of scales simulated in the scale-space is affected by the parameter κ.

For the requirements of the present analysis, and to investigate thoroughly how the operator definition affects extrema extraction, the considered DoG scale-space implementation allows us to set κ and n_{spo} independently.

Implementation details. The input image is oversampled by a factor $1/\delta_{\min}$ to reach the $\delta_{\min}$ sampling rate. This was done by using a cubic B-spline interpolation of order 3. From this interpolated image all images in the scale-space were computed using a

combination of DCT Gaussian convolution and subsampling. For each scale σ simulated in an octave, the algorithm computes two images, the first one corresponding to scale σ and the second one corresponding to scale $\kappa\sigma$ (both being directly computed from the input image). Although we lost the benefit of a low computational cost, this gave us flexibility and allowed us to investigate the influence of the operator definition regardless of the scale-space sampling rate.

4.3 The theoretical scale invariance

In this section we give the correct proof that SIFT is scale invariant and stress the fact that this proof also indicates that knowing exactly the initial camera blur is crucial for the method's consistency.

4.3.1 The camera model

In the SIFT framework, the camera point spread function is modeled by a Gaussian kernel G_c and all digital images are frontal snapshots of an ideal planar object described by the infinite resolution image u_0. In the underlying SIFT invariance model, the camera is allowed to rotate around its optical axis, to take some distance, or to translate while keeping the same optical axis direction. All digital images can therefore be expressed as

$$\mathbf{u} =: \mathbf{S}_1 G_c H \mathcal{T} R u_0, \tag{4.2}$$

where $\mathbf{S}_1$ denotes the sampling operator, H an arbitrary homothety, $\mathcal{T}$ an arbitrary translation and R an arbitrary rotation.

4.3.2 The SIFT method is theoretically invariant to zoom outs

It is not difficult to prove that SIFT is consistent with the camera model. Nevertheless, the proof in Morel and Yu [2011] is inexact, as pointed out in Sadek [2012]. Let $\mathbf{u}_\lambda$ and $\mathbf{u}_\mu$ denote two digital snapshots of the scene u_0. More precisely,

$$\mathbf{u}_\lambda = \mathbf{S}_1 G_c H_\lambda u_0 \quad \text{and} \quad \mathbf{u}_\mu = \mathbf{S}_1 G_c H_\mu u_0. \tag{4.3}$$

Assuming that the images are well sampled, namely that $\mathbf{S}_1$ is invertible by Shannon interpolation, and taking advantage of the semi-group property (4.1), the respective scale-spaces are

$$v_\lambda(\sigma, \mathbf{x}) = G_{\sqrt{\sigma^2 - c^2}} \mathbf{I}_1 \mathbf{S}_1 G_c H_\lambda u_0(\mathbf{x}) = G_\sigma H_\lambda u_0(\mathbf{x}) \tag{4.4}$$

$$v_\mu(\sigma, \mathbf{x}) = G_\sigma H_\mu u_0(\mathbf{x}), \tag{4.5}$$

where $\mathbf{I}_1$ denotes the Shannon interpolation operator. These formulae imply that both scale-spaces only differ by a reparameterization. Indeed, if v_0 denotes the Gaussian scale-space of the infinite resolution image u_0 (i.e., $v_0(\sigma, \mathbf{x}) = G_\sigma u_0(\sigma, \mathbf{x})$) we have

$$v_\lambda(\sigma, \mathbf{x}) = H_\lambda(G_{\lambda\sigma} u_0(\mathbf{x})) = v_0(\lambda\sigma, \lambda\mathbf{x}), \tag{4.6}$$

$$v_\mu(\sigma, \mathbf{x}) = v_0(\mu\sigma, \mu\mathbf{x}), \tag{4.7}$$

thanks to a commutation relation between homothety and convolution.

By a similar argument, the two respective DoG functions are related to the DoG function w_0 derived from u_0. For a ratio $\kappa > 1$ we have

$$\begin{aligned} w_\lambda(\sigma, \mathbf{x}) &= v_\lambda(\kappa\sigma, \mathbf{x}) - v_\lambda(\sigma, \mathbf{x}) && (4.8)\\ &= v_0(\kappa\lambda\sigma, \lambda\mathbf{x}) - v_0(\lambda\sigma, \lambda\mathbf{x}) && (4.9)\\ &= w_0(\lambda\sigma, \lambda\mathbf{x}) && (4.10) \end{aligned}$$

and similarly $w_\mu(\sigma, \mathbf{x}) = w_0(\mu\sigma, \mu\mathbf{x})$.

Consider an extremum point $(\sigma_0, \mathbf{x}_0)$ of the DoG scale-space w_0. Then if $\sigma_0 \geq \max(\lambda c, \mu c)$, this extremum corresponds to extrema $(\sigma_1, \mathbf{x}_1)$ and $(\sigma_2, \mathbf{x}_2)$ in w_λ and w_μ respectively, satisfying $\sigma_0 = \lambda\sigma_1 = \mu\sigma_2$. This equivalence of extrema between the two scale-space guarantees that the SIFT descriptors are identical.

Note that this same relation links the two normalized Laplacian applied on v_λ and v_μ, denoted respectively nL_λ and nL_μ, both related to the normalized Laplacian of v_0 denoted nL_0. We have

$$\begin{aligned} nL_\lambda(\sigma, \mathbf{x}) &= \sigma^2 \Delta v_\lambda(\sigma, \mathbf{x}) && (4.11)\\ &= (\lambda\sigma)^2 \Delta v_0(\lambda\sigma, \lambda\mathbf{x}) && (4.12)\\ &= nL_0(\lambda\sigma, \lambda\mathbf{x}) && (4.13)\\ nL_\mu(\sigma, \mathbf{x}) &= nL_0(\mu\sigma, \mu\mathbf{x}) && (4.14) \end{aligned}$$

Therefore, considering extrema of the normalized Laplacian as keypoints will also lead to SIFT descriptors that are identical.

4.3.3 Knowing the camera blur is crucial for scale invariance

The knowledge of the camera blur is crucial to ensure the theoretical invariance to zoom-outs Sadek [2012]. Indeed, DoG scale-spaces computed with a wrong camera blur have in general unrelated extrema. Starting again from the two digital snapshots $\mathbf{u}_\lambda$ and $\mathbf{u}_\mu$, but assuming a wrong blur c' instead of the correct blur c, the respective Gaussian scale-spaces are:

$$\begin{aligned} v_\lambda(\sigma, \mathbf{x}) &= G_{\sqrt{\sigma^2 - c'^2}} \mathbf{I}_1 \mathbf{S}_1 G_c H_\lambda u_0(\mathbf{x}) && (4.15)\\ &= G_{\sqrt{\sigma^2 - c'^2 + c^2}} H_\lambda u_0(\mathbf{x}) && (4.16)\\ &= v_0(\lambda\sqrt{\sigma^2 - c'^2 + c^2}, \lambda\mathbf{x}) && (4.17) \end{aligned}$$

and

$$v_\mu(\sigma, \mathbf{x}) = v_0(\mu\sqrt{\sigma^2 - c'^2 + c^2}, \mu\mathbf{x}). \quad (4.18)$$

We see that, because of the wrong blur assumption, the scale-space function v_0 is shrunken or dilated along scale. The corresponding DoG scale-spaces are:

$$\begin{aligned} w_\lambda(\sigma, \mathbf{x}) = {} & v_0(\lambda\sqrt{\kappa^2\sigma^2 - c'^2 + c^2}, \lambda\mathbf{x}) \\ & - v_0(\lambda\sqrt{\sigma^2 - c'^2 + c^2}, \lambda\mathbf{x}), \end{aligned} \quad (4.19)$$

$$\begin{aligned} w_\mu(\sigma, \mathbf{x}) = {} & v_0(\mu\sqrt{\kappa^2\sigma^2 - c'^2 + c^2}, \mu\mathbf{x}) \\ & - v_0(\mu\sqrt{\sigma^2 - c'^2 + c^2}, \mu\mathbf{x}). \end{aligned} \quad (4.20)$$



Figure 4.1: Examples of simulated images consistent with SIFT's image camera model. The respective blur levels are $c = 0.5$, $c = 1.0$ and $c = 0.6$.

None of these are linear reparameterizations of the DoG function w_0 anymore. They yield therefore unrelated extrema. Such bias is maximal with detections at finer scales and with large zoom factors.

4.4 Simulating the digital camera

Controlling the image formation process permits us to measure how invariant SIFT is in different scenarios. Such a control will be achieved by simulating images that are consistent with the SIFT camera model. Images at different zoom levels were simulated from a large reference real digital image u_{ref} through Gaussian convolution and subsampling. To simulate a camera having a Gaussian blur level c, a Gaussian convolution of standard deviation cS, with $S > 10$ is first applied to the reference image. The convolved image is then subsampled by a factor S. Assuming that the reference image has an intrinsic Gaussian blur level of $c_{\text{ref}} \ll cS$, the resulting Gaussian blur level is $\sqrt{c^2 + (c_{\text{ref}}/S)^2} \approx c$. We estimated the blur level introduced by a digital reflex camera by fitting a Gaussian function to the estimated camera point-spread-function (following Delbracio et al. [2012]). The obtained Gaussian blur levels varied from $c = 0.35$–0.95, depending on the aperture of the lens (blur level increases with aperture size). Different zoomed-out and translated versions were simulated by adjusting the scale parameter S and by translating the sampling grid. Thanks to the large subsampling factor, the generated images are noiseless. In addition, the images were stored with 32 bit precision to mitigate quantization effects. Figure 4.1 shows some examples of simulated images used in the experiments.

It might be objected that our simulations are highly unrealistic as the images to be compared by SIFT in a real scenario are not perfectly sampled or noiseless. Nevertheless, with an ever growing image resolution, more and more images will be compared by SIFT in large octaves, and therefore after a large subsampling, so that these properties can become realistic in practice. Furthermore, even if applying SIFT to the originals and regardless of initial noise and blur, the images at large scales also become anyway perfect so that the accuracy and repeatability issues under such favorable conditions are relevant.

4.5 Empirical analysis of the digital scale-space sampling

The SIFT method is built on precisely locating the extrema of the DoG scale-space. Ideally, one would like to detect and locate all extrema from the underlying continuous DoG scale-space. However, in practice, we do not have access to the continuous scale-space but to its discrete counterpart. In theory, as $\delta_{\min} \to 0$ and $n_{\text{spo}} \to \infty$ the discrete scale-space better approximates the continuous scale-space therefore allowing to extract reliably all continuous extrema. This section investigates what happens when the sampling rates increase and how sampling affects the successive steps of the rudimentary procedure for detecting 3D scale-space discrete extrema, namely the extraction of discrete extrema, their quadratic interpolation and their filtering based on their DoG response.

To focus on the influence of the scale-space sampling, the study was carried out in the most favorable conditions: noiseless and aliasing-free input images ($c = 1.1$ and $S = 10$). In all experiments we set $\kappa = 2^{1/3}$ to separate the mathematical definition of the DoG analysis operator from the scale-space discretization.

4.5.1 Number of detections

To evaluate how the scale-space sampling rates affects the number of detections we generated different scale-space discretization by varying the parameters $(\delta_{\min}, n_{\text{spo}})$, and extracted the 3D discrete extrema for each one of them.

Figure 4.2 **(a)** shows the number of detected extrema for the different scale-space samplings. At first sight, it seems that some digital scale-space samplings produce many more keypoints than the SIFT default sampling ($\delta_{\min} = 1/2, n_{\text{spo}} = 3$). However, this increase in detections happens for discretizations that are significantly unbalanced in space and in scale. By unbalance we mean that the scale and the space dimensions are sampled with very different sampling rates.

Boundary effect. To do a fair comparison of the different *discrete* detected extrema when changing the scale-space sampling rates, we have to consider that depending on the scale-space sampling, some extrema close to the lower scale boundary are not detected. Indeed, due to the scale discretization there are no detected keypoints with scale below $\sigma_{\min} 2^{1/2 n_{\text{spo}}}$. To compensate for this dead range, which is a function of n_{spo}, we restricted the analysis to a common scale range independent of n_{spo}. This was achieved by discarding all extrema with scale below $\sigma_{\min} 2^{1/3}$. To avoid issues due to the coarse scale discretization, we used the keypoint scale obtained after refinement (3.12). Figure 4.2 **(b)** shows, for all scale-space tested configurations, the number of detections in the common scale region. The number of detected extrema lying in the common region is much more similar for all the scale-space samplings.

Duplicate detections. We will say that detections $(\sigma_0, \mathbf{x}_0)$ and $(\sigma_1, \mathbf{x}_1)$ are the same, if:

$$||\mathbf{x}_0 - \mathbf{x}_1||_\infty \leq \epsilon \quad \text{and} \quad R^{-1} \leq \sigma_1/\sigma_0 \leq R, \tag{4.21}$$

where ϵ and R are the spatial tolerance and scale relative tolerance values respective.

Clearly, there is a compromise between saying that two detections are not the same and allowing some displacement due to numerical errors. Currently, we are not tackling the problem of precision (how accurate a keypoint can be localized) but the problem of

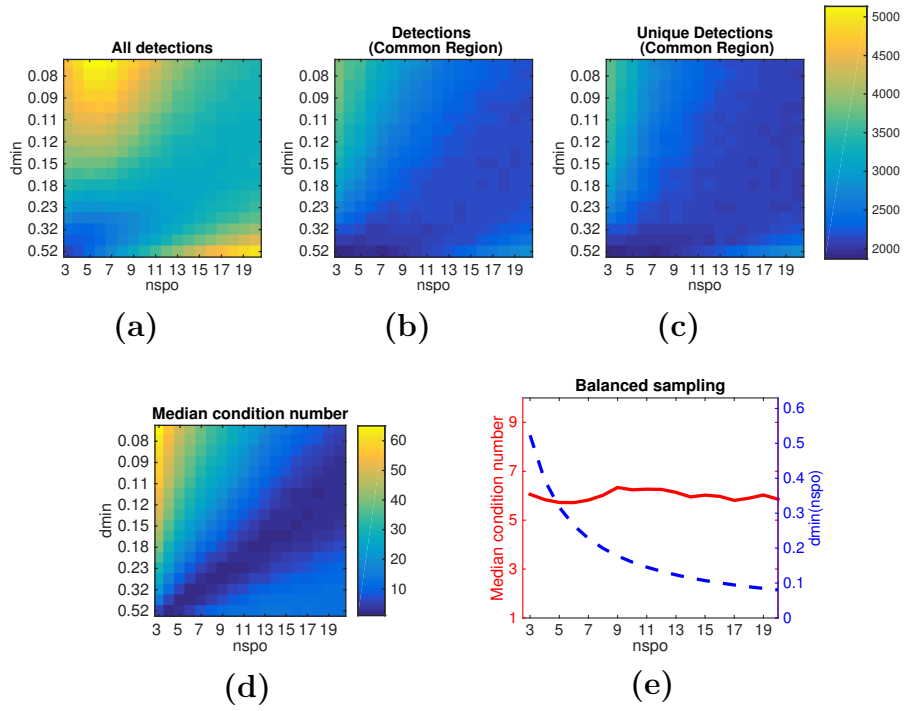


Figure 4.2: Influence of the scale-space sampling rate $(n_{\text{spo}}, \delta_{\text{min}})$ on the number of detected DoG extrema. **(a)** Number of 3D DoG discrete extrema. Unbalanced discretizations can produce twice as many detections as the default scale-space sampling used in SIFT ($n_{\text{spo}} = 3, \delta_{\text{min}} = 1/2$). This gap is reduced after compensating for a boundary effect by discarding 3D discrete extrema with detected scale below $\sigma_{\text{min}} 2^{1/3}$ **(b)**, and after removing duplicate detections **(c)**. Unbalanced discretizations may lead to inaccurate local models for the extrema refinement proposed in SIFT. **(d)** Median of the condition numbers of DoG 3D Hessians used for extrema interpolations. Unbalanced sampling grids (shown in the top-right or bottom-left parts of this graph) produce extrema with significantly poor Hessian condition number. This leads to unstable extrema interpolations. **(e)** Balanced sampling rates (those satisfying (4.22), shown in the dotted blue line) lead to extrema having well conditioned Hessian matrices (red line).

not mixing two different detections. With that aim, it seems reasonable that the tolerance values are set in order to avoid that one detection be mistaken for another. We opted to set tolerance values to $\epsilon = 1.0$ and $R = 2^{1/2}$ independently of the scale-space sampling.

Let $\mathcal{D}$ be the set of detected DoG extrema. We call duplicates of $(\mathbf{x}_0, \sigma_0) \in \mathcal{D}$ the subset of detected extrema $\mathrm{D}(\mathbf{x}_0, \sigma_0) \subset \mathcal{D}$ that satisfy (4.21). Given the set of all detected keypoints $\mathcal{D}$, we say that $\mathcal{U}$ is a representative set of unique detections if

$$\mathcal{U} = \arg\min |U| \quad \text{s.t.} \quad U \subset \mathcal{D} \text{ and } \cup_{(\mathbf{x},\sigma)\in U} D(\mathbf{x}, \sigma) = \mathcal{D},$$

where the number of keypoints in the set U is denoted by $|U|$. Figure 4.2 **(c)** shows the number of unique detections in the common scale region. The number of unique detections is similar to the number of detections (Figure 4.2 **(b)**). This indicates that in general duplicate detections are negligible.

Balance the scale and space DoG sampling.

The SIFT algorithm proposes to refine the position of a discrete extremum using

a quadratic interpolation. Having an unbalance sampling in scale and space may lead to an unreliable interpolation due to the very different discretization. As we presented in Section 2, the refinement of a keypoint is done by solving a linear system (from (3.12)). The sensitiveness to numerical errors can be measured by the linear system's condition number (i.e., the condition number of the Hessian at the extrema to be refined). Figure 4.2 **(d)** shows the median of the condition number for the sets of detected extrema associated with different scale-space samplings. It shows that using a balanced sampling rate improves the overall stability of the extrema interpolation.

By balanced sampling we mean that the distance separating adjacent samples in the scale dimension is similar to the distance separating adjacent samples in space. For a DoG scale-space with parameter κ, the distance between the first two simulated scales is

$$\Delta\sigma = \kappa\sigma_{\min}(2^{1/n_{\text{spo}}} - 1).$$

Thus, to equally sample the Gaussian kernel

$$G(\mathbf{x}, \sigma) = \frac{1}{2\pi\sigma^2} e^{-||\mathbf{x}||^2/2\sigma^2}$$

in scale and space, the spatial inter pixel distance should be

$$\delta_{\min} = \sqrt{2}\Delta\sigma = \sqrt{2}\kappa\sigma_{\min}(2^{1/n_{\text{spo}}} - 1). \tag{4.22}$$

This relation between both sampling rates is plotted in Figure 4.2 **(e)** along with the median condition numbers on this set of balanced sampling rates. The condition number is mostly constant for balanced samplings.

4.5.2 Stability of DoG extrema to scale-space sampling

To evaluate if all 3D discrete extrema are equally stable to an increase of the DoG sampling rate, we simulated a set of increasingly dense balanced scale-spaces. We set the minimal scale-space blur level to $\sigma_{\min} = 1.1$. We simulated increasingly dense scale-space samplings $(n_{\text{spo}}, \delta_{\min})_i$, for $i = 1, \ldots, n$ with $n_{\text{spo}} = 3, \ldots, 19$ and the balanced spatial sampling rate $\delta_{\min} := \delta_{\min}(n_{\text{spo}})$ given by (4.22) ($i = 1$ being the coarsest one and $i = n$ the finest one). Figure 4.3 **(a)** shows that the number of detections is approximately constant for different balanced sampling rates.

Let $\mathcal{D}_i$ for $i = 1 \ldots, n$ be the sets of detected 3D extrema for the discretizations described above. Given a detected extremum $(\mathbf{x}_0, \sigma_0) \in \mathcal{D}_i$, we say the extremum is detected in $\mathcal{D}_j$ if there exists $(\mathbf{x}, \sigma) \in \mathcal{D}_j$ such that they are the same detection according to the precision conditions (4.21). We say that a detected extremum $(\mathbf{x}_0, \sigma_0) \in \mathcal{D}_i$ is new if it was not detected in $\mathcal{D}_{i-1}$. Given the sampling i, the rate of new extrema is computed as the fraction of new detected keypoints and the total number of detections. In the same way, we define the rate of lost extrema as those present in the (coarser) sampling i and not present in the (finer) sampling $i + 1$. Figure 4.3 **(b)** shows the rate of new and lost detections as a function of the sampling rate. The new detection rate decreases with the sampling rate and stabilizes to a minimal rate of 10% of the total number of detections for $n_{\text{spo}} \geq 14$. The same observations applied to the rate of lost extrema.

This surprising result means that despite sampling the scale-space very finely, 3D discrete extrema keep appearing and disappearing when changing the sampling.

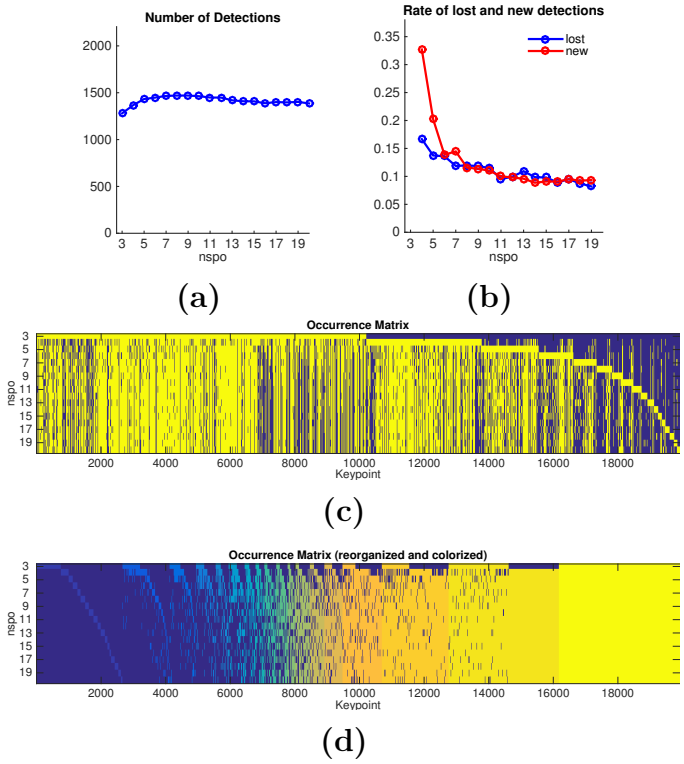


Figure 4.3: Influence of sampling density on stability. A set of increasingly dense and balanced scale-spaces is computed. The scale-space samplings are indexed by the n_{spo} value, and the δ_{min} is given by (4.22). **(a)** The number of detections is roughly constant for different sampling rates. **(b)** The rates of lost extrema (detected in the current sampling but not in the immediately finer sampling) and of new extrema (detected in the current sampling but not in the immediately coarser sampling) decrease with the sampling rate n_{spo} and stabilize around 10% of the total number of detections. **(c)** The occurrence matrix. Each row in this matrix corresponds to one of the simulated samplings (n_{spo}), while each column indicates if a keypoint was detected in that particular sampling. **(b)** For better visualization, the columns are colored and reorganized in increasing order of stability (yellow: always detected, blue: detected only once). Almost 20% of the detections appear no matter the scale-space sampling rate.

To illustrate how discrete extrema appear and disappear as scale-space sampling rates changes, we decided to investigate the stability of each single detected extremum. The set of all unique detected extrema is formed by gathering the extrema detected on all the simulated scale-spaces $\mathcal{D}_{\text{all}} = \cup_{i=1,\dots,n}\mathcal{D}_i$ and then by extracting a unique set of detections $\mathcal{U}_{\text{all}}$. For each detected extremum $(\mathbf{x}, \sigma) \in \mathcal{U}_{\text{all}}$, we checked for its presence in each of the $\mathcal{D}_i$ detection sets. This was done by using the same definition as in (4.21). The results are summarized in the *occurrence* matrix shown in Figure 4.3 **(c)**. Each simulated discretization is indexed by the n_{spo} value. Each entry in this matrix indicates if a keypoint in $\mathcal{U}_{\text{all}}$ (column) was found in the scale-space with a given discretization $i = 1, \dots, n$ (where i is the row index in the matrix).

We define the *stability* of a unique keypoint as the ratio between the number of discretizations it is detected in divided by the total number of discretizations. Figure 4.3 **(d)** shows the normalized *occurence* matrix, where each entry in the occurence matrix is multiplied by the *stability* value (therefore each column has the same color). Also, keypoints

(columns) were reorganized from less to more stable (left to right).

The normalized occurrence matrix confirms that a majority of the keypoints are stable as they appear on at least 80% of the discretizations, and that some keypoints tend to appear and disappear repeatedly as sampling rates increase. It also shows that the proportion of unstable keypoints (e.g., those appearing less than 20%) is low overall but is significantly larger for coarse discretizations than in denser ones.

4.5.3 Can unstable (intermittent) detections be detected?

To increase its overall detection stability, SIFT discards non-contrasted extrema based on their absolute DoG value. However, many other features, computed from the values of the extremum and its neighbors, could be used as well. The DoG value, the Laplacian of the DoG, the DoG Hessian condition number and the minimal absolute value of the difference between the extremum and its adjacent samples are some of them.

To find out if any of these simple features is good at predicting if a discrete extremum is stable (to different sampling rates), we proceeded as follows. Given the set of unique detections $\mathcal{U}_{\text{all}}$ computed by gathering all detections from the different scale-spaces with different sampling rates, we considered two subsets of unique keypoints: one subset of *stable* unique extrema (with occurrence rate above 80%) and one subset of *unstable* unique extrema (occurrence rate below 20%). Figure 4.4 **(a–d)** shows the proportion of extrema in both stable/unstable sets respectively, that have a feature value below a certain threshold. The considered features are: (a) the DoG value, (b) the Laplacian of the DoG, (c) the DoG Hessian condition number and (d) the minimal absolute value of the difference between the extremum and its adjacent samples.

This figure demonstrates that none of these features manages to faithfully separate the stable from the unstable ones. This is confirmed by the ROC curve shown in Figure 4.4 **(e)** (see figure caption for details). Noticeably, the keypoint feature giving the lowest discrimination performance is the DoG value used by SIFT.

4.5.4 The influence of extrema interpolation on stability, precision and invariance

The refinement of the discrete extrema position proposed in SIFT has two main purposes. First, it allows to locate the extrema to subpixel accuracy thanks to a local continuous model of the DoG scale-space. But this refinement procedure also detects and discards unstable discrete extrema.

In this section, we analyze the impact of the refinement procedure. To that aim, we considered an input image and a series of transformations simulating small displacements of the camera. Although the analysis was restricted for a sake of simplicity to the case of translations and scale changes, it could be easily generalized to more complex image transformations such as perspective projections.

We examined the influence of the two main parameters in the refinement procedure (see Section 3.3.2): the maximal number of allowed interpolations N_{interp}, and the maximum offset M_{offset} authorized for the extremum at each refinement iteration.

Our performance measure will be the *stability*, measured by considering the number of keypoints that appear in at least a certain percentage of the simulated image transformations. A perfectly stable keypoint would be one that appears in all the simulated images,

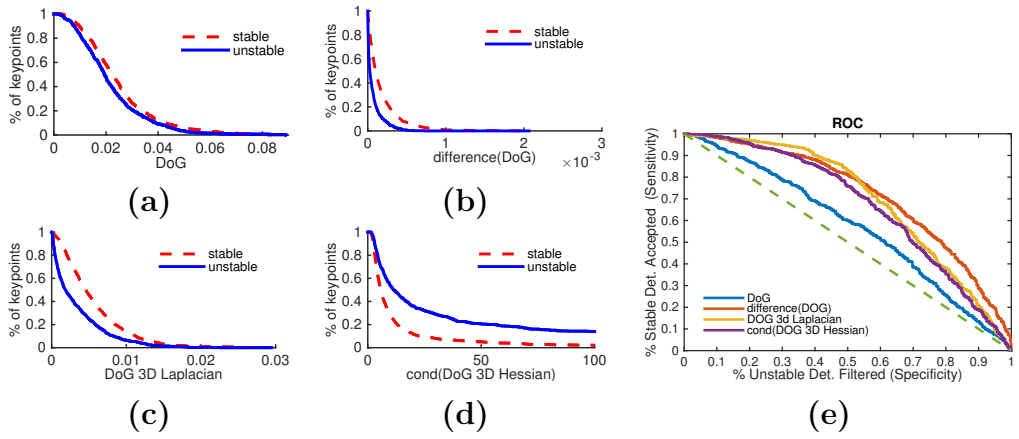


Figure 4.4: Filtering keypoints that are unstable to changes in the scale-space sampling. Increasing thresholds are applied respectively to the set of stable and unstable detections. The considered features are: **(a)** the extremum DoG value, **(b)** the difference of extremum DoG value and the adjacent samples in the scale-space, **(c)** the DoG 3D Laplacian value at the extremum, **(d)** the condition number of the DoG 3D Hessian at the extremum. None of the tested features separates convincingly the unstable from the stable detections. This is confirmed by the ROC curves, illustrating the performance of each feature, shown in **(e)**. A point in a ROC curve indicates the proportion of non-filtered stable keypoints (good detections – sensitivity) as a function of the filtered unstable ones (good removals – specificity) for a particular threshold value. A perfect feature should produce a ROC that is always one. According to this, the worst feature for eliminating keypoints unstable to changes in the scale-space sampling is the DoG value.

while a perfectly unstable keypoint would be one that only appears in one of the images. We also measured the *precision* by computing the average standard deviation of the location of the stable keypoints, where keypoints were considered stable if they appeared in at least 50% of the simulated transformations.

Figure 4.5 **(a,b)** shows the number of unique keypoints that appear in at least a given percentage of the translations for different values of M_{offset}. Each figure corresponds to a given sampling rate ($n_{\text{spo}} = 3$ and 15) and a given maximal number of interpolations ($N_{\text{interp}} = 1, 2, \infty$). Ideally, one would like to have a large number of stable detections, which would correspond to a flat curve. Although the number of detections for the SIFT sampling rate ($n_{\text{spo}} = 3$) is large, it decreases quickly when considering only the more stable ones, present in a large percentage of the simulated transformations. On the other hand, $n_{\text{spo}} = 15$ leads to flatter curves, which implies more stable detections, and demonstrates that increasing the scale-space sampling improves stability. The refinement of the extrema helps to discard the unstable ones.

The fact that the results with $N_{\text{interp}} = 2$ and $N_{\text{interp}} = \infty$ are identical (second and third row of Figure 4.5), implies that there is no extra benefit in allowing more than two iterations. The present analysis indicates that allowing a maximum of two interpolations ($N_{\text{interp}} = 2$) in combination with a maximum displacement of $M_{\text{offset}} = 0.6$ produce the largest number of stable keypoints. This conclusion is independent of the considered n_{spo}. Therefore, for the remainder of the chapter, we consider the refinement step with these two values.

Increasing the scale-space sampling rate in conjunction with extrema interpolation has

a tremendous impact on the detection precision. Figure 4.5 shows for both, discrete and interpolated detections, the mean of the precision of stable keypoints (appearing in at least 50% of translations) as a function of the scale-space sampling rate.

We repeated the same experiment but different camera zoom-outs were simulated. The results are very similar to the pure camera translation case (see Figure 4.6). In general, sampling the scale-space finer than what is proposed in SIFT (e.g., $n_{\text{spo}} > 3$) allows to better localize the DoG extrema. In addition, the local refinement of the extrema position increases the extrema precision.

4.5.5 Influence of κ

The DoG scale-space is formed by computing the difference of Gaussians operator at scales $\kappa\sigma$ and σ. To analyze the influence of the DoG parameter κ, we computed the extrema of different DoG scale-spaces produced with $\kappa = 2^{1/30}, 2^{1/29}, \ldots, 2^{1/2}$. In order to minimize sampling related instability, the scale-spaces were sampled at $n_{\text{spo}} = 15$ and the respective $\delta_{\min}$.

The number of detected extrema is more or less constant for different values of κ (Figure 4.7 **(a)**) Depending on the κ value, the same structure is detected at a different scale. As pointed out in Section 4.2.2, a Gaussian blob of standard deviation σ produces an extrema of the DoG at scale $\sigma/\sqrt{\kappa}$. Thus, we have normalized the detections scale by $\sigma_{\text{normalized}} = \sigma\sqrt{\kappa}$. To compare the keypoints detected with different κ values, we also restricted the analysis to those lying on the common scale range, that is, $\sigma_{\min}\sqrt{2^{1/2}} \leq \sigma \leq 2\sigma_{\min}\sqrt{2^{1/30}}$.

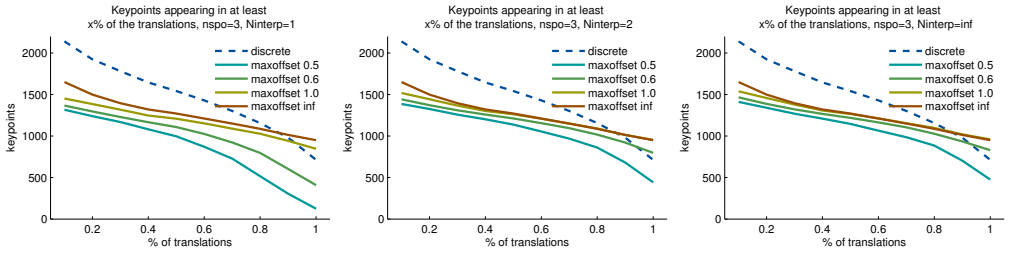
We proceeded similarly as before by gathering all the detections from the different DoG scale-spaces and computed a set of unique detections. Then, we proceeded to create the occurrence matrix. The occurrence matrix in Figure 4.7 **(b)** shows that the different κ's lead for the most part to identical detections. Almost half the keypoints are detected in every DoG scale-space and a large percentage of the keypoints is detected in most simulated scale-spaces.

4.6 Impact of deviations from the perfect camera model

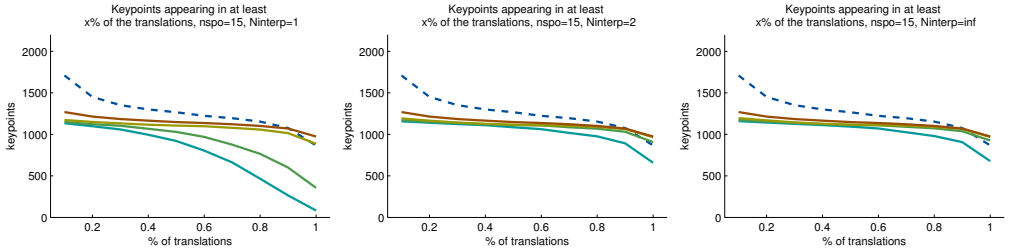
In order to achieve perfect invariance, SIFT formally requires that the image is acquired in perfect conditions. This means that the input image should be noiseless, well-sampled (according to the Nyquist-Shannon sampling theorem) and with an *a priori* known level of Gaussian blur c. These ideal conditions justify the construction of the image scale-space. In this section, we evaluate what happens when there are deviations from these ideal requirements.

4.6.1 Image aliasing

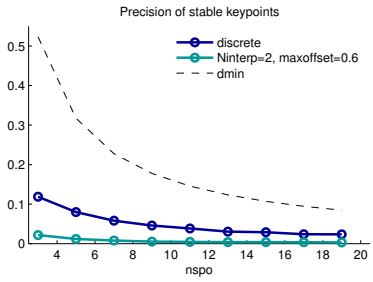
Let us assume that the input image was generated with a camera having a Gaussian point-spread-function of standard deviation c. If c is low (i.e., $c \leq 0.7$) the acquired image will be subject to aliasing artifacts. We shall assume first that this camera blur c is known beforehand, so that the SIFT method can be applied consistently.



(a) $n_{\text{spo}} = 3$



(b) $n_{\text{spo}} = 15$



(c)

Figure 4.5: Influence of extrema refinement parameters M_{offset} and N_{interp} on the detection stability/-precision. A set of translated images was simulated and the keypoints extracted. Each curve shows the number of keypoints appearing in a least a certain percentage of the simulated image translations for different values of $M_{\text{offset}} = 0.5, 0.6, 1.0, \infty$. The plots in the first, second and third row were generated considering a maximum number of interpolations $N_{\text{interp}} = 1, 2$ and ∞ respectively. The left block of plots **(a)** was generated by sampling the scale-space with $n_{\text{spo}} = 3$ (and the corresponding δ_{min}), while the right block **(b)** was generated using $n_{\text{spo}} = 15$. Allowing two iterations ($N_{\text{interp}} = 2$) and a maximal offset of $M_{\text{offset}} = 0.6$ gives the best performance in terms of stability of detected keypoints. Allowing for more interpolations attempts did not increase the performance, as can be seen by comparing the third row to the second row. **(c)** shows the influence of the extrema refinement on the precision of the stable set of keypoints (appearing in at least 50% of the simulated images). In this pure translation scenario, it appears that the precision of the detected extrema significantly increases when using extrema interpolation and when sampling finely the scale-space (e.g., $n_{\text{spo}} > 3$).

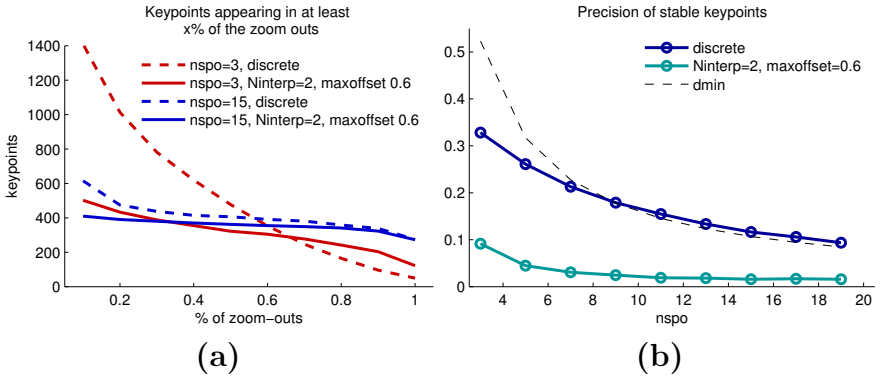


Figure 4.6: Influence of scale-space sampling and extrema refinement on the invariance to zoom-outs. A set of zoomed-out images was simulated and the keypoints extracted. **(a)** The number of keypoints appearing in at least a certain percentage of the simulated images for different scale-space sampling and refinements. The best performance is obtained by significantly oversampling the scale-space, with $n_{\text{spo}} = 15$, and by refining the extrema with the local interpolation. In this case, most of the detected keypoints are present in all the simulated images. On the other hand, the original SIFT sampling $n_{\text{spo}} = 3$ leads to low stability even with the extrema refinement step. **(b)** Mean precision of stable keypoints location (appearing in at least 50% of the zoom-outs) plotted as a function of the sampling rate n_{spo}. The local refinement of the extrema position significantly increases the precision of the extrema detection. Also, using a finer grid than the one proposed in SIFT (e.g., $n_{\text{spo}} > 3$) allows to better localize the extrema.

To evaluate the SIFT performance in this aliasing situation, we simulated random translations of the digital camera. Then, we computed the extrema of the DoG scale-spaces generated with each translated image and compared the extrema. All scale-space consisted of one octave computed with $n_{\text{spo}} = 15$, $\sigma_{\text{min}} = 1.1$ and the interpolation parameters were set to $N_{\text{interp}} = 2$ and $M_{\text{offset}} = 0.6$.

Figure 4.8 **(a)** shows the average number of keypoints detected as a function of the camera blur c. The number of detections is independent of the camera blur. Indeed, a sharper shot does not increase the number of keypoints.

In Figure 4.8 **(b)** we show the number of unique keypoints that appear in at least a certain percentage of the translated images. Keypoints detected from well sampled images (e.g., $c > 0.6$) are stable to translation (the curves are almost flat) while those from severely undersampled images ($c \approx 0.3$) are very sensitive to the position of the sampling grid, as expected.

4.6.2 Unknown input image blur level

A more realistic scenario is the case where the level of blur of the input image c is unknown. SIFT requires this value to create the scale-space starting at a known level of image blur σ_{min}. A wrong assumption of the input camera blur affects the range of simulated scales simulated in the Gaussian scale-space.

To demonstrate to what extent the wrong knowledge of the input camera blur produce unrelated keypoints, we compared the keypoints extracted assuming an image blur of $c = 0.7$ from a set of images having actual random blur c_{real} uniformly picked from $[c - \Delta c, c + \Delta c]$.

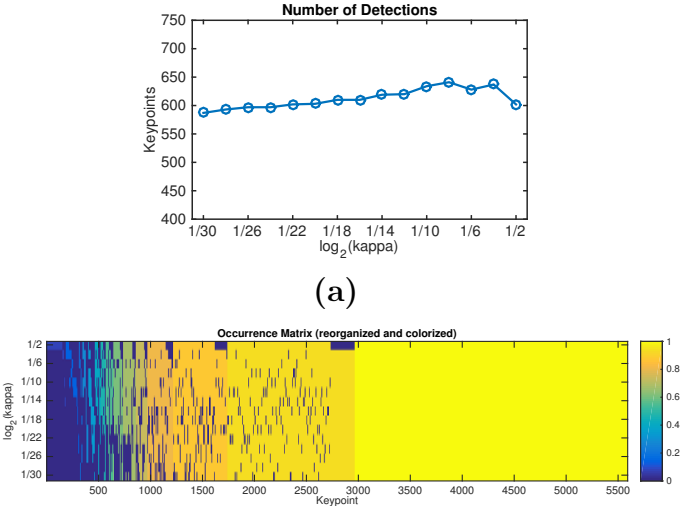


Figure 4.7: Influence of the DoG parameter κ. The number of detected keypoints is roughly constant for different values of κ **(a)** . The occurrence matrix for the set of unique normalized keypoints detected in the different DoG scale-spaces **(b)**. A large majority of the keypoints are detected in most simulated scale-spaces when changing the value of κ.

Figure 4.9 shows the number of unique keypoints that appear in at least a certain percentage of the simulated images. This was evaluated for different ranges of uncertainty (i.e., $\Delta c = 0.05 - 0.4$). The larger the range of uncertainty Δc, the more unrelated the extrema are (the curve decreases very fast, indicating the presence of many unique keypoints appearing in only a few of the simulated images).

4.6.3 Image noise

The digital image acquisition is always affected by noise that undermines the performance of SIFT. To evaluate the impact of image noise we simulated different image acquisition, by adding random white Gaussian noise to the input image. Then, we proceeded to compute the keypoints that are detected in a certain percentage of the simulated images. Figure 4.10 is self-explanatory and demonstrates the strong impact of noise level on keypoint stability. Noise has a strong impact on the stability of the detected keypoints.

4.7 Concluding remarks

We presented a systematic analysis of the main steps involved in the detection of keypoints in the SIFT algorithm. One of the main conclusions is that the original parameter choice in SIFT is not sufficient to ensure a theoretical and practical scale (and even translation) invariance, which was the main claim of the SIFT method. In addition, we show that the SIFT invariance claim is strongly affected if the assumption on the level of blur in the input image is wrong.

Specifically, we showed that increasing the scale-space sampling from $n_{\text{spo}} = 3$ to $n_{\text{spo}} = 15$ (and respectively the space sampling rate δ_{min}) improves the stability of the

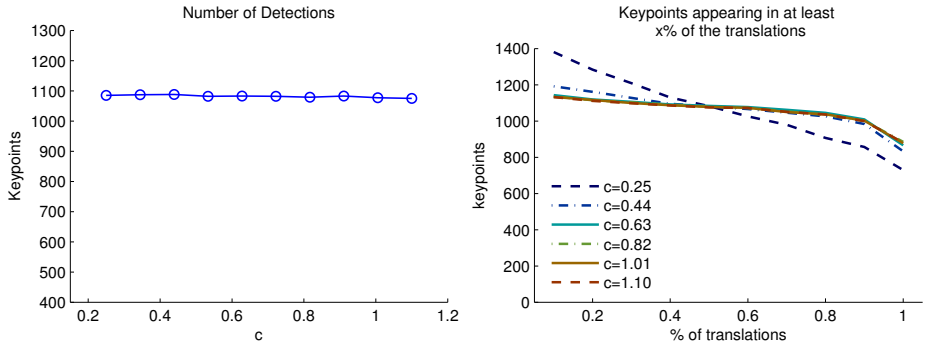


Figure 4.8: Impact of image aliasing. For various camera blurs, $0.25 \leq c \leq 1.1$, a set of translated images were simulated and the DoG keypoints extracted ($n_{\text{spo}} = 15$, $\sigma_{\text{min}} = 1.1$). Aliasing does not affect the number of detections **(a)**. In **(b)** we show the number of unique keypoints appearing in at least a certain percentage of the simulated translations. Detections are less stable for severely aliased images ($c = 0.25$), while for $c > 0.6$, the impact of aliasing is negligible.

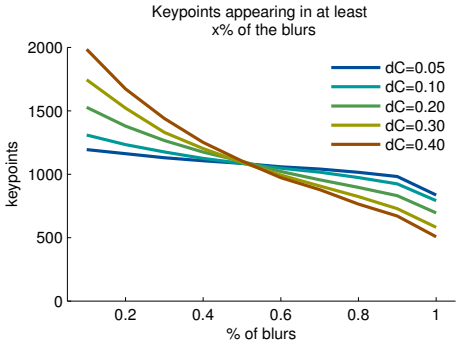


Figure 4.9: The impact of a wrong assumption on the camera blur. Comparison of the keypoints extracted assuming $c = 0.7$ when the real camera blur was picked randomly in $[c - \Delta c, c + \Delta c]$. The number of keypoints that appear in at least a certain percentage of the simulated images is plotted for different levels of uncertainty on camera blur ($\Delta c = 0.05 - 0.4$).

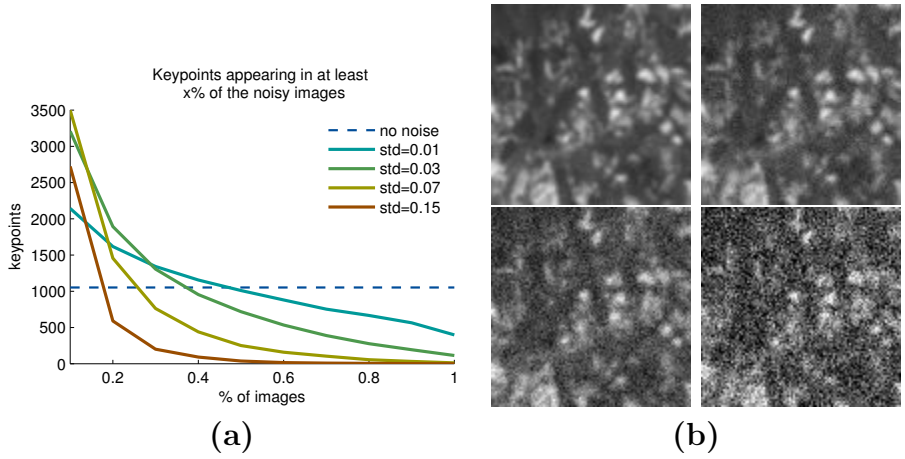


Figure 4.10: Impact of image noise. **(a)** The number of unique keypoints that appear in at least a certain proportion of the simulated images is plotted for different levels of image noise. Noise has a significant impact on the DoG extrema detection. **(b)** Crops of the input images simulated with $c = 0.8$ and added Gaussian white noise of standard deviation $\sigma_{\text{noise}} = 0.01, 0.03, 0.07$ and 0.15.

detected keypoints. This implies that if a series of image transformations (e.g., translations, zoom-outs) are applied to an image, the keypoints detected in one of them will be detected with high probability in all the others. This stability property is fundamental for fulfilling the scale invariance claim. The extrema refinement was shown to improve both the precision and the stability of the detected keypoints. We showed that the largest number of stable keypoints is achieved with parameters $M_{\text{offset}} = 0.6$ and $N_{\text{interp}} = 2$ (while SIFT recommends $N_{\text{interp}} = 5$). We also demonstrated that the DoG threshold fails to filter out unstable keypoints, and that the different definitions of the DoG scale-space (parameter κ) lead for the most part to identical detections up to a normalization of the scale. Finally, we showed how the presence of aliasing and noise in the acquired image deteriorate detections stability.

5 Is repeatability an unbiased criterion for ranking feature detectors?

Most computer vision applications rely on algorithms finding local correspondences between different images. Because of the importance of the problem, new keypoint detectors and descriptors are constantly being proposed, each one claiming to perform better than the preceding ones. This raises the question of a fair comparison between very diverse methods. This evaluation has been mainly based on a repeatability criterion of the keypoints under a series of image perturbations (blur, illumination, rotations, homotheties, homographies, etc). In this chapter, we argue that the classic repeatability criterion is biased favoring algorithms producing redundant overlapped detections. To overcome this bias, we propose a variant of the repeatability rate taking into account the descriptors overlap. We apply this variant to revisit the popular benchmark by Mikolajczyk et al. [Mikolajczyk et al. 2005], comparing several classic and recently introduced feature detectors. Experimental evidence shows that the hierarchy of these feature detectors is severely disrupted by the amended comparator.

5.1 Introduction

Local stable features are the cornerstone of many image processing and computer vision applications such as image registration [Hartley and Zisserman 2003; Snavely et al. 2006], camera calibration [Grompone von Gioi et al. 2010], image stitching [Haro et al. 2012], 3D reconstruction [Agarwal et al. 2011], object recognition [Grimson and Huttenlocher 1990; Fergus et al. 2003; Bay et al. 2006a; Zhang et al. 2007] or visual tracking [Reid 1979; Zhou et al. 2009]. The introduction of the SIFT method by David Lowe in 1999 [Lowe 1999, 2004] sparked an explosion of local keypoints detector/descriptors seeking discrimination and invariance to a specific group of image transformations [Tuytelaars and Mikolajczyk 2008].

Ideally, one would like to detect keypoints that are stable to image noise, illumination changes, and geometric transforms such as scale changes, affinities, homographies, perspective changes, or non-rigid deformations. Complementarily, the detected features should provide information as diverse as possible. Detections should, for example, be well distributed throughout the entire image extracting information from all image regions and from boundary features of all kinds (e.g., textures, corners, blobs). Hence, there is a vari-

ety of detectors/descriptors built on different principles and having different requirements. While the SIFT method and its similar competitors [Bay et al. 2006b; Mikolajczyk et al. 2005; Mainali et al. 2013] detect blob like structure in a multi-scale image decomposition, other approaches [Brown et al. 2005; Mikolajczyk et al. 2005; Rosten and Drummond 2006; Förstner et al. 2009; Rosten et al. 2010; Leutenegger et al. 2011] explicitly detect corners or junctions at different scales. As opposed to interest point detectors, interest region detectors [Tuytelaars and Van Gool 1999, 2000; Kadir et al. 2004; Cao et al. 2005] extract the invariant salient regions of an image based on its topographic map. To fairly compare the very different feature detectors it is fundamental to have a rigorous evaluation protocol.

The repeatability rate measures the detector's ability to identify the same features (i.e., *repeated* detections) despite variations in the viewing conditions. Defined as the ratio between the number of keypoints simultaneously present in all the images of the series (repeated keypoints) over the total number of detections, it can be seen as a measure of the detector's efficiency. Indeed, the repeatability rate incorporates two struggling quality criteria: the number of repeated detections (i.e., potential correspondences) should be maximized while the total number of detections should be minimized since the complexity of the matching grows with the square of the number of detections.

Interest point detectors can also be indirectly evaluated through a particular application. In [Mikolajczyk and Schmid 2005], the authors propose to evaluate detector-descriptor combinations in an image matching/recognition scenario. Although this approach can lead to very practical observations, the conclusions about the keypoints stability is intertwined with the descriptor's discrimination ability.

In this chapter, we show that the repeatability criterion suffers from a systematic bias: it favors redundant and overlapped detections. This has serious consequences, as evenly distributed and independent detections are crucial in image matching applications. The concentration of many keypoints in a few image regions is generally not helpful, no matter how robust and repeatable they may be. To better measure the detectors redundancy, we introduce a modified repeatability criterion. We consider the area actually covered by the descriptor and we evaluate the *descriptor overlap* as a measure of redundancy.

The remainder of this chapter is organized as follows. Section 5.2 describes the repeatability criterion, discusses its variants, and illustrates how algorithms producing redundant detections may have a good performance according to this traditional quality measure. In Section 5.3 we introduce a correction to the repeatability criterion that overcomes this bias, by accounting for the descriptor overlap. Section 5.4 reviews twelve state-of-the-art feature detectors, and details the region involved in the feature extraction for each of the analyzed methods. This extracted region will be the key ingredient for the proposed overlap measure. Comparative performance tables and maps gathered in Section 5.5 show that the hierarchy of detectors is drastically altered by the new repeatability criterion. This result is further confirmed by analyzing the detection/matching performance using the same normalized descriptor for all the detectors. Conclusions are finally summarized in Section 5.6.

5.2 The repeatability criterion and its bias

5.2.1 Definition of the repeatability criterion

Consider a pair of images $u_a(\mathbf{x})$, $u_b(\mathbf{x})$ defined for $\mathbf{x} \in \Omega \subset \mathbb{R}^2$ and related by a planar homography H, that is, $u_b = u_a \circ H$. The detector repeatability rate for the pair (u_a, u_b) is defined as the ratio between the number of detections simultaneously present in both images, i.e., repeated detections, and the total number of detections in the region covered by both images.

In the repeatability framework, a detection generally consists of an elliptical region, denoted $R(\mathbf{x}, \Sigma)$, parametrized by its center $\mathbf{x}$ and a 2×2 positive-definite matrix Σ,

$$R(\mathbf{x}, \Sigma) = \left\{ \mathbf{x}' \in \Omega \mid (\mathbf{x}' - \mathbf{x})^T \Sigma^{-1} (\mathbf{x}' - \mathbf{x}) \leq 1 \right\}.$$

A pair of detections (elliptical regions $R(\mathbf{x}_a, \Sigma_a)$ and $R(\mathbf{x}_b, \Sigma_b)$) from images $u_a(\mathbf{x})$ and $u_b(\mathbf{x})$ will be considered repeated if

$$e_{\text{overlap}} = 1 - \frac{|R(\mathbf{x}_a, \Sigma_a) \cap R(\mathbf{x}_{ba}, \Sigma_{ba})|}{|R(\mathbf{x}_a, \Sigma_a) \cup R(\mathbf{x}_{ba}, \Sigma_{ba})|} \leq \epsilon_{\text{overlap}}, \tag{5.1}$$

where e_{overlap} is the overlap error, $\mathbf{x}_{ba} = H\mathbf{x}_b$, $\Sigma_{ba} = A\Sigma_b A^T$ represents the reprojection of the ellipse $R(\mathbf{x}_b, \Sigma_b)$ from image u_b into the image u_a and A is the local affine approximation of the homography H.

The union and intersection of the detected regions are examined on the reference image $u_a(\mathbf{x})$ by projecting the detection on the image u_b into the image u_a. The union covers an area denoted by $|R(\mathbf{x}_a, \Sigma_a) \cup R(\mathbf{x}_{ba}, \Sigma_{ba})|$ while $|R(\mathbf{x}_a, \Sigma_a) \cap R(\mathbf{x}_{ba}, \Sigma_{ba})|$ denotes the area of their intersection. The parameter $\epsilon_{\text{overlap}}$ is the maximum overlap error tolerated. In most published benchmarks it is set to $\epsilon_{\text{overlap}} = 0.40$ [Mikolajczyk and Schmid 2004; Mikolajczyk et al. 2005; Mainali et al. 2013].

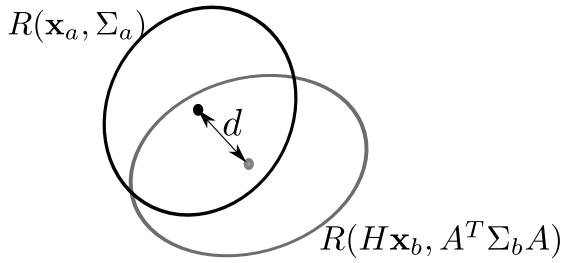


Figure 5.1: Illustration of the repeatability criterion. Detection $R(\mathbf{x}_b, \Sigma_b)$ on image u_b is reprojected on the reference image u_a. If the overlap error is lower than $\epsilon_{\text{overlap}}$ (see (5.1)), the detections are considered repeated.

Let Ω be the region covered by both images u_a and u_b. Since the number of repeated detections is upper bounded by the minimal number of detections in Ω (under the assumption that there are no multiple matches), the repeatability rate is defined as

$$\text{rep} = \frac{\text{number of repeated detections}}{\min\left(|\mathcal{K}_a|_\Omega, |\mathcal{K}_b|_\Omega\right)} \tag{5.2}$$

where $|\mathcal{K}_a|_\Omega$ and $|\mathcal{K}_b|_\Omega$ denote the respective numbers of detections inside Ω.

5.2.2 Illustration and alternative definitions

To discuss and illustrate the repeatability criterion, let us consider the particular case of a pair of detections $R(\mathbf{x}_a, \Sigma_a)$ and $R(\mathbf{x}_b, \Sigma_b)$ whose re-projections on the reference image are two disks, both of radius r and with centers separated by a distance d (Figure 5.1). Such a pair will be considered repeated if $d/r \leq f(\epsilon_{\text{overlap}})$, where f is a monotone function easily derived from (5.1). Figure 5.2 **(a)** shows the maximum distance d under which both detections will be considered repeated as a function of the radius r.

As pointed out in [Mikolajczyk et al. 2005], detectors providing larger regions have a better chance of yielding good overlap scores, boosting as a result their repeatability scores. This also means that one can artificially increase the repeatability score of any detector by increasing the scale associated with its detections. The authors of [Mikolajczyk et al. 2005] proposed to avoid this objection by normalizing the detected region size before computing the overlapped error. The two detected elliptical regions $R(\mathbf{x}_a, \Sigma_a)$ and $R(\mathbf{x}_b, \Sigma_b)$ in (5.1) are replaced respectively by the elliptical regions $R(\mathbf{x}_a, \kappa^2/r_a R_a \Sigma_a)$ and $R(\mathbf{x}_b, \kappa^2/r_b R_b \Sigma_b)$, where r_a and R_a (respectively r_b and R_b) are the radii of the elliptical region $R(\mathbf{x}_a, \Sigma_a)$ (respectively $R(\mathbf{x}_b, \Sigma_b)$) and $\kappa = 30$ is its radii geometric mean after normalization.

The idea of such normalization was to prevent boosting a detector's performance by enlarging its associated ellipse. Yet, such a criterion is not scale-invariant, meaning that it may be over or under permissive depending on the detection size. For example, the maximal distance separating repeated detections of equal size does not take into account the scale (e.g., the radius of the circle in our special case illustration, see Figure 5.2 **(b)**). In consequence, with $\epsilon_{\text{overlap}}$ set to its standard value ($\epsilon_{\text{overlap}} = 40\%$), two circular detections of radius 1px and centers separated by 12px can still be regarded as repeated, although their respective descriptors may not even overlap!

Surprisingly, the code provided by the authors of [Mikolajczyk et al. 2005][1] does not implement any of the criteria defined in their article. The code introduces a third definition by incorporating an additional criterion on the maximum distance separating two repeated keypoints that depends on the scale by

$$|\mathbf{x}_a - H\mathbf{x}_b| \leq 4\sqrt{r_a R_a}.$$

This criterion is illustrated in Figure 5.2 **(c)** for the same study case of two circular detections of equal size. This third criterion is not scale invariant either. Thus in this paper we shall stick to the first definition, which is scale invariant. With the non-redundant repeatability criterion to be introduced in the next section, it will become pointless to try "boosting" a detector's scale. Indeed such attempts will result in decreased matching performance. The detection's characterizing scale will be the spatial extent of the descriptor ultimately computed, which is the real practical scale associated with each detector.

5.2.3 Repeatability favors redundant detectors

The following mental experiment sheds light on how the repeatability favors redundancy. Let DET be a generic keypoint detector, and let DET2 be a variant in which each detection is simply counted twice. The number of repeatable keypoints and the total number of detections are both artificially doubled, leaving the repeatability rate unchanged. However, although the number of costly descriptor computations has doubled, no extra benefit can

[1] `http://www.robots.ox.ac.uk/~vgg/research/affine/` retrieved on August 5th, 2014

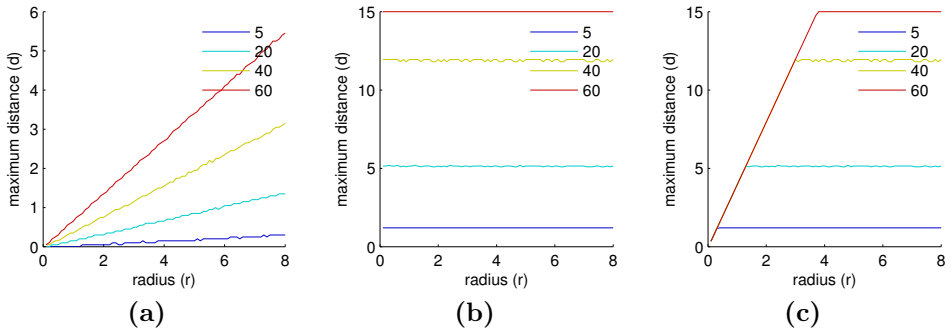


Figure 5.2: Illustrating three different definitions of the repeatability criteria. Consider a pair of detections whose re-projections on the reference image are two disks of radius r with their centers separated a distance d. The maximal tolerated separation distance d_{max} between repeated detections is plotted as a function of the radius r for four values of the parameter $\epsilon_{\text{overlap}}$ (5%, 20%, 40% and 60%). **(a)** original definition given by (5.1), **(b)** with ellipses normalization $\kappa = 30$, **(c)** definition implemented in the code provided by the authors of [Mikolajczyk et al. 2005]. Only the first definition is scale invariant.

be extracted from the enlarged set of repeated keypoints. The classic repeatability rate fails to report that the benefit over cost ratio of DET2 is half the one of DET. This explains why methods producing correlated detections may misleadingly get better repeatability ratios.

A popular attempt for mitigating this drawback is to compare detectors at a fixed number of detections [Li et al. 2011; Mainali et al. 2013]. This would not, however, solve the problem for two reasons. Firstly, by a similar reasoning as before, one can imagine a detector that repeats its best detection N times (N being the "fixed" number of detections) while discarding the rest. Such a detector would achieve optimal repeatability, despite being useless. But most importantly, given a detector, selecting the N best detections via a parameter (e.g., a threshold) is not generally an easy task. For example, in SIFT, the most popular way of adjusting the number of detected keypoints is by thresholding the analysis operator (Difference of Gaussians) to retain only the most salient features. However, it is well known that this does not necessarily lead to a good selection in terms of stability [Rey-Otero et al. 2014]. To improve the selection, Li et al. [Li et al. 2011] proposed a supervised regression process to learn how to rank SIFT keypoints. Although, this scheme produces good results it requires supervised learning.

For these reasons, we believe that a fair comparison should prefer the genuinely independent detections. The metric introduced in the following section is a first attempt in this direction.

5.3 Non-redundant repeatability

Besides the repeatability measure, which ignores the keypoints spatial distribution, other specific metrics have been proposed. Some examine the spatial distribution of the descriptors and others evaluate how well they describe the image. The ratio between the convex hull of the detected features and the total image surface is used in [Dickscheid and Förstner 2009] as a coverage measure. The harmonic mean of the detections positions is used in [Ehsan et al. 2011, 2013] as a measure of concentration. In [Dickscheid et al.

2011], the authors propose to measure the completeness of the detected features, namely the ability to preserve the information contained in an image by the detected features. The *information content* metric proposed in [Schmid et al. 2000] quantifies the distinctiveness of a detected feature with respect to the whole set of detections. Non disctinctive features are indeed harmful, as they can match to other many and therefore confuse the matching. Being complementary to it, these metrics are generally used in combination with the repeatability rate. Nevertheless, since the purpose of the repeatability is to report on the benefit/cost ratio of a given detector, it should also, by itself, report on the description redundancy. We shall see that the descriptors redundancy can be naturally incorporated in the repeatability criterion.

5.3.1 Non-redundant detections

To evaluate the redundancy of a set of detections $k \in \mathcal{K}$, each detection $(\mathbf{x}_k, \Sigma_k)$ can be assigned, in accordance with the descriptor associated canonically with the keypoint for each method, a mask function $f_k(\mathbf{x})$ consisting of a truncated elliptical Gaussian

$$f_k(\mathbf{x}) = Ke^{-\frac{1}{2\zeta^2}(\mathbf{x}-\mathbf{x}_k)^T\Sigma_k^{-1}(\mathbf{x}-\mathbf{x}_k)},$$

if $(\mathbf{x}-\mathbf{x}_k)^T\Sigma_k^{-1}(\mathbf{x}-\mathbf{x}_k) \leq \rho^2$ and 0 elsewhere. Each mask is normalized so that its integral over the image domain is equal to 1. The values ρ and ζ control the extent of the detected feature, as it can be derived from the descriptor's design. They will be fixed here for each detector by referring to the original paper where it was introduced (section 5.4). Indeed most detectors proposals come up with a descriptor or at least with a characterization of the region where this descriptor should be computed.

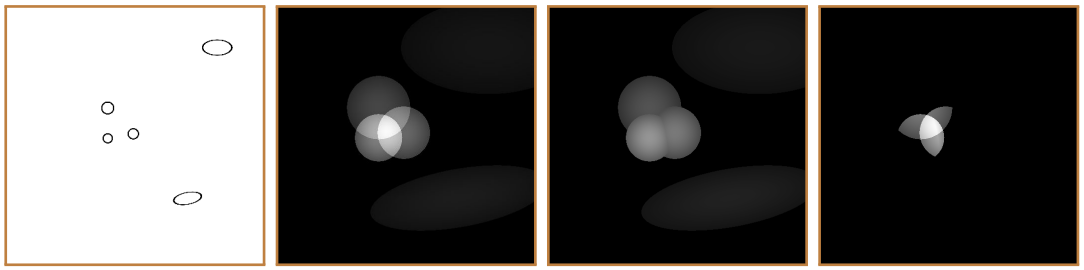


Figure 5.3: The mask functions formalizing the keypoint description on a toy example consisting of several Gaussian blobs **(a)**. The sum over all detections $\sum_{k\in\mathcal{K}} f_k(\mathbf{x})$ maps the contribution of each image pixel to different descriptors **(b)**. The max over all detections masks $\max_{k\in\mathcal{K}} f_k(\mathbf{x})$ maps the pixel contributions to the best available descriptor **(c)**. Their difference maps the detection redundancy **(d)**.

The sum of all descriptor masks $\sum_{k\in\mathcal{K}} f_k(\mathbf{x})$ yields a final map showing how much each image pixel contributes to the set of all computed descriptors. Note that one pixel may contribute to several descriptors (as in the example shown in Figure 5.3). Similarly, the maximum taken over all detections $\max_{k\in\mathcal{K}} f_k(\mathbf{x})$ measures the contribution of pixel $\mathbf{x}$ to the best descriptor. Thanks to the mask normalization, the number of keypoints

$K := card\,(\mathcal{K})$ is given by

$$K = \int_\Omega \left(\sum_{k \in \mathcal{K}} f_k(\mathbf{x}) \right) d\mathbf{x}, \tag{5.3}$$

where Ω denotes the image domain. On the other hand,

$$K_{\text{nr}} := \int_\Omega \left(\max_{k \in \mathcal{K}} f_k(\mathbf{x}) \right) d\mathbf{x} \tag{5.4}$$

measures the number of *non-redundant* keypoints. This value can be interpreted as a count of the independent detections.

To gain some intuition and see why this measurement is quite natural, let us examine four illustrative cases. Assume that there are only two detected keypoints so that $K = 2$. If the two detections

1. completely overlap, then $K_{\text{nr}} = 1$.
2. If they share the same center but have different sizes, then $1 < K_{\text{nr}} < K = 2$. But if their sizes are significantly different, then $K_{\text{nr}} \approx 2$, which makes sense. Indeed, one of them describes a fine detail and the other one a detail at a larger scale. Their information contents are roughly independent.
3. If both keypoints are very close to each other then again $1 < K_{\text{nr}} < K = 2$ and the above remark on scales still applies.
4. If the descriptors do not overlap at all then $K_{\text{nr}} = K = 2$.

The propensity of a given algorithm to extract overlapped and redundant detections can therefore be measured by computing the *non-redundant detection ratio*:

$$\text{nr-ratio} := K_{\text{nr}}/K. \tag{5.5}$$

5.3.2 Non-redundant Repeatability

The above definitions entail a straightforward modification of the repeatability criterion (5.2). Let $\mathcal{K}_r$ be the set of repeatable keypoints (satisfying (5.1)) between two snapshots, and Ω the area simultaneously covered by both images. We define the *non-redundant repeatability rate* by

$$\text{nr-rep} := \frac{\int_\Omega \max_{k \in \mathcal{K}_r} f_k(\mathbf{x}) d\mathbf{x}}{\min\left(|\mathcal{K}_a|_\Omega, |\mathcal{K}_b|_\Omega\right)} \tag{5.6}$$

where $|\mathcal{K}_a|_\Omega$ and $|\mathcal{K}_b|_\Omega$ denote the respective numbers of detections inside Ω. The number of repeated detections in (5.2) is replaced in (5.6) by the number of non-redundant detections.

5.4 Spatial coverage of state-of-the-art feature detectors

In this section we review the twelve state-of-the-art feature detectors that will be compared using the non-redundant repeatability criteria. Our goal is to specify the region of the

descriptor associated with each detector. It is classically objected that the descriptors associated with a detector may influence its matching performance. Hence the detector performance should be evaluated independently of its associated descriptor, and conversely. Fortunately, most papers introducing a detector also specify the area of interest around each detector as a circular or elliptical region. This is the region on which the final descriptor will be computed, regardless of its description technique. This information about the descriptor's region can be taken from the original papers. It is independent of the ultimate choice of a description technique, which may indeed vary strongly. In our discussion of each detector, we shall nevertheless also associate a fixed type of descriptor to each method, so as to be able to compare matching performance on an equal footing. This comparison is performed at the end of Section 5.5.

Some of the detectors considered here were also compared in the original benchmark by Mikolajczyk et al. [Mikolajczyk et al. 2005], namely, the Harris-Laplace and Hessian-Laplace [Mikolajczyk et al. 2005], Harris-Affine and Hessian-Affine [Mikolajczyk et al. 2005], EBR [Tuytelaars and Van Gool 1999], IBR [Tuytelaars and Van Gool 2000] and MSER [Matas et al. 2004]. We also included here for completeness methods published since: SIFT [Lowe 1999, 2004], SURF [Bay et al. 2006b], SFOP [Förstner et al. 2009], BRISK [Leutenegger et al. 2011] and SIFER [Mainali et al. 2013]. Table 5.1 summarizes the algorithms invariance properties. For details, we refer the reader to the original methods publications and to the survey by Tuytelaars and Mikolajczyk [Tuytelaars and Mikolajczyk 2008].

Furthermore, we shall show detection maps on pattern images as well as on several natural photographs to illustrate the behavior of each algorithm.

Most keypoint detection methods share the use of the *Gaussian scale-space* $u(\mathbf{x}, \sigma)$ defined by

$$u(\mathbf{x}, \sigma) := (G_\sigma * u)(\mathbf{x}), \text{ with } G_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma} e^{-\frac{\|\mathbf{x}\|^2}{2\sigma^2}},$$

where σ and $\mathbf{x}$ are respectively called the scale and space variables.

	detects	feature	rotation	zoom	homothety	affine
SIFT	$(\mathbf{x}, \sigma)$	blob	yes	yes	no	no
EBR	parallelograms	corners	yes	no	yes	limited
IBR	$(\mathbf{x}, \Sigma)$	blob	yes	no	yes	yes
Hessian-Laplace	$(\mathbf{x}, \sigma)$	blob	yes	yes	no	no
Hessian-Affine	$(\mathbf{x}, \Sigma)$	blob	yes	yes	no	limited
Harris-Laplace	$(\mathbf{x}, \sigma)$	corner	yes	yes	no	no
Harris-Affine	$(\mathbf{x}, \Sigma)$	corner	yes	yes	no	limited
MSER	regions	contrasted level lines	yes	no	no	yes
SURF	$(\mathbf{x}, \sigma)$	blob	limited	yes	no	no
SFOP	$(\mathbf{x}, \sigma)$	junction, circles	yes	no	yes	no
BRISK	$(\mathbf{x}, \sigma)$	corners	yes	yes	no	no
SIFER	$(\mathbf{x}, \sigma)$	blob	no	no	yes	limited

Table 5.1: Summary of algorithms' invariance properties. A zoom is the combination of a homothety and a Gaussian smoothing modeling the camera's point spread function. The considered detectors detect elliptical regions $(\mathbf{x}, \Sigma)$, circular regions $(\mathbf{x}, \sigma)$, regions or parallelograms.

SIFT (scale invariant feature transform) [Lowe 1999, 2004] is probably the most popular local image comparison method. As we have seen in Chapter 3, the SIFT keypoints are the stable interpolated 3D extrema of difference of Gaussian scale-space. We also remind that the description of a keypoint consists of a feature vector assembled from the gradient distribution over an oriented patch surrounding the detected keypoint. For a detection at scale σ, the described patch covers a circular area of radius $\rho\sigma = 6\sqrt{2}\sigma$ weighted by a Gaussian mask of standard deviation $\zeta\sigma = 6\sigma$ [2]. The described patch is oriented along a dominant orientation of the gradient distribution. SIFT considers multiple dominant orientations. This means that one keypoint may be described by various feature vectors, each corresponding to one of the dominant orientations. We shall also consider a variant of SIFT that only takes one feature vector per detection, the one corresponding to the dominant orientation. We shall call it SIFT-single (SIFT-S).

EBR (edge based regions) [Tuytelaars and Van Gool 1999] is an affine-invariant region detector. This method is not based on a scale-space image representation but on explicitly searching the image for structures of various sizes. Starting from a Harris corner point, EBR localizes the two nearby edges and analyzes their curvature to assign to each segment a characteristic direction and length. EBR returns the parallelogram bounded by the two edge segments. The parallelogram regions can be mapped into elliptical shapes having the same first and second moments. The EBR descriptor consists of a set of invariant moments computed over the elliptical region. For the sake of comparison, we will rely on the matching experiments on an affine normalized SIFT feature vector computed over the same elliptical region. In contrast with the SIFT method, the normalized patch is not weighted by a Gaussian mask.

IBR (intensity based regions) [Tuytelaars and Van Gool 2000] is an affine-invariant method which detects elliptical shapes of various sizes centered on specific gray level extrema. This method is not based on the Gaussian scale-space. By detecting abrupt changes in the intensity profiles along a set of rays originating from a gray value extremum, IBR extracts contrasted regions of various sizes and associates to them elliptical shapes. Similarly to EBR, invariant moments are computed over the detected region to build the feature vector. For a sake of homogeneity in our matching comparisons we shall instead use a SIFT descriptor computed on the affine normalized patch, without applying a Gaussian weighting mask.

Harris-Laplace and Hessian-Laplace detectors [Mikolajczyk et al. 2005]. Unlike SIFT, these methods use two multi-scale representations instead of one. The first one is used to determine the keypoint location and the second one is used to select its characteristic scale. In the case of the Hessian-Laplace method, the first multi-scale representation is the 2D Hessian determinant while the second one is the normalized Laplacian, both computed on the Gaussian pyramid [Lindeberg 1993]. The 2D Hessian determinant extremum gives the keypoint location $\mathbf{x}$. Then, the extremum of the scale-space Laplacian $\Delta u(\mathbf{x}, \sigma)$ with respect to σ gives the keypoint scale. The detector goes back and forth between both multi-scale representations to iteratively refine $\mathbf{x}$ and σ. The Harris-Laplace method proceeds almost identically. Only the Harris operator [Harris and Stephens 1988] is used

[2] In the original SIFT algorithm the area covered by the descriptor is a square patch of size $12\sigma \times 12\sigma$. However, to uniformize all the algorithms since some of them do not give a reference keypoint orientation, we opted to replace the patch by the smallest disk containing it, which therefore covers a slightly larger area.

in place of the 2D Hessian to extract the keypoint location $\mathbf{x}$. The Harris-Laplace features are predominantly corners while the Hessian-Laplace mostly detects blobs. Unlike in the SIFT method, the extrema are not interpolated to subpixel precision. Once extracted, each keypoint is locally described, using the SIFT or the GLOH descriptor [Mikolajczyk et al. 2005; Mikolajczyk and Schmid 2005]. Consequently, for a detection at scale σ, the described patch covers a circular area of radius $\rho\sigma = 6\sqrt{2}\sigma$ weighted by a Gaussian mask of standard deviation $\zeta\sigma = 6\sigma$.

Harris-Affine and Hessian-Affine detectors [Mikolajczyk et al. 2005] are affine extensions of the Harris-Laplace and Hessian-Laplace detectors. Instead of detecting keypoints, both methods detect elliptical regions. Compared to the Harris-Laplace and Hessian-Laplace methods, the affine variants contain an additional step in which the second-moment matrix is used to estimate an elliptical shape around each keypoint[3]. These elliptical shapes are used to normalize the local neighborhood by an affine transformation before its description (using the SIFT or the GLOH descriptor). The SIFT descriptor is adopted in the present study. If σ denotes the geometric mean of the ellipse radii, then the described patch covers a circular area in the affine-normalized neighborhood of radius $\rho\sigma = 6\sqrt{2}\sigma$ weighted by a Gaussian mask of standard deviation $\zeta\sigma = 6\sigma$.

MSER (maximally stable extremal regions) [Matas et al. 2004] is an affine-invariant method which extracts regions that are connected components of image upper level sets. By examining how the area of the image upper-level sets evolves with respect to an image intensity threshold, MSER measures the region stability. The MSERs are the regions that achieve a local maximum of the (non-positive) derivative of the region area with respect to its level. MSER proposes to compute feature descriptors at different scales of the detected region size (1.5, 2 and 3 times the convex hull of the detected region). In addition, MSER regions can be easily mapped into elliptical shapes and then used to compute an affine descriptor of the detected region. In the present framework, for each of the detected regions a SIFT feature vector on an affine normalized patch of twice the size of the detected region was computed.

SURF (speeded-up robust features) [Bay et al. 2006b] can be regarded as a fast alternative to SIFT. SURF keypoints are the 3D extrema of a multi-scale image representation that approximates the 2D Hessian determinant computed on each scale of the Gaussian scale-space. The Gaussian convolution is approximated using box filters computed via integral images. SURF descriptors are computed over a Gaussian window centered at the keypoint, and encode the gradient distribution around the keypoint using 2D Haar wavelets. The described patch for a detection at scale σ covers a circular area of radius $\rho\sigma = 10\sqrt{2}\sigma$ weighted by a Gaussian mask of standard deviation $\zeta\sigma = 3.3\sigma$. Note that the described areas used in SIFT and SURF are slightly different. A SURF descriptor patch is larger but uses a more concentrated Gaussian mask.

SFOP (scale-invariant feature operator) [Förstner et al. 2009] is a versatile multi-scale keypoint detector that explicitly models and detects corners, junctions and circular features. SFOP is built on the Förstner feature operator [Förstner 1994] for detecting junctions and on the spiral model [Bigün 1990] for unifying different feature types into a

[3] The elliptical shape is estimated via an iterative procedure. Unreliable detections with degenerated second-moment matrices are also discarded in the process.

common mathematical formulation. For detecting keypoints at different scales, the input image is decomposed into a series of images using a Gaussian pyramid. Each image is then scanned for various feature types, namely, circular structures of various sizes and junctions of different orientations. At each pixel, the algorithm takes a surrounding patch and evaluates its consistency to the feature model. Although SFOP only concerns keypoint detection, the authors recommend combining the SFOP detector with SIFT's descriptor. Consequently, the described patch for a detection at scale σ also covers a circular area of radius $\rho\sigma = 6\sqrt{2}\sigma$ weighted by a Gaussian mask of standard deviation $\zeta\sigma = 6\sigma$.

BRISK (binary robust invariant scalable keypoints) [Leutenegger et al. 2011] focuses on speed and efficiency. The BRISK detector is a multi-scale adaptation of FAST and its optimized version AGAST [Rosten and Drummond 2006; Mair et al. 2010] corner detectors. The AGAST corner detector is first applied separately to each scale of a Gaussian pyramid decomposition to rapidly identify potential regions of interests. For each pixel in such regions, a corner score quantifying the detection confidence is computed (see [Mair et al. 2010] for details). Based on the AGAST corner score, BRISK performs a 3D non-maxima suppression and a series of quadratic interpolations to extract the BRISK keypoints $(\mathbf{x}, s)$, being $(\mathbf{x})$ the 2D position and s the feature size. The BRISK descriptor is a binary string resulting from brightness differences computed around the keypoint.

In the current analysis, we calibrated the size of the detections s provided by the BRISK binary to make it comparable to the other methods. We empirically found that the image of a 2D Gaussian function of standard deviation σ produces a SIFT detection of scale σ while it produces a BRISK feature of size $s = 4\sigma$. In consequence, for a BRISK detection of size s, the described patch in the present study covers a circular area of radius $\rho s = \frac{3}{2}\sqrt{2}s$ weighted by a Gaussian mask of standard deviation $\zeta s = \frac{3}{2}s$.

SIFER (scale-invariant feature detector with error resilience) [Mainali et al. 2013]. The recently introduced SIFER algorithm tightly follows SIFT, but computes a different multi-scale image representation. Instead of smoothing the image with a set of Gaussian filters and computing its Laplacian, SIFER convolves the image with a bank of cosine modulated Gaussian kernels (see Figure 5.4).

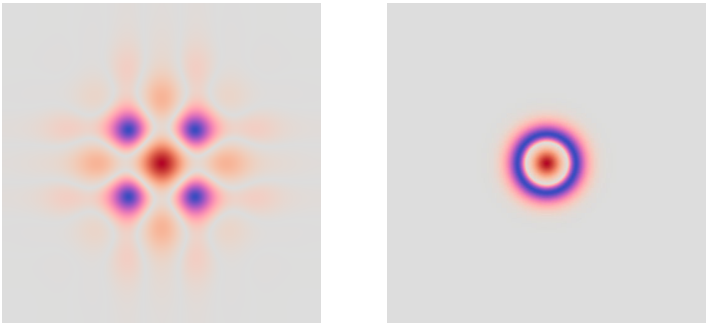
$$\mathrm{cmg}_\sigma(x, y) = \left(2\pi\sigma^2 \left(\cos\left(\frac{cx}{\sigma}\right) + \cos\left(\frac{cy}{\sigma}\right)\right) G_\sigma\right). \tag{5.7}$$

The 3D extrema of the resulting multi-scale representation are the SIFER keypoints. The method is homothety invariant. Unlike SIFT, however, SIFER is not zoom-out invariant. Indeed, its kernel does not commute with a Gaussian camera blur. The authors claim that, despite loosing rotation invariance, the approach increases the detection precision in both scale and space thanks to the better localization of the modulated cosine filters. The descriptor computed at each extracted keypoint is identical to the SIFT descriptor. Therefore, the described patch considered in the present study covers a circular area of radius $\rho\sigma = 6\sqrt{2}\sigma$ weighted by a Gaussian mask of standard deviation $\zeta\sigma = 6\sigma$.

Table 5.2 summarizes the values of ρ and ζ for each method.

5.4.1 Detection maps

Different detectors extract different kinds of features, in different amounts and with different spatial distributions. To visually inspect the algorithms general behavior, figures



$\mathrm{cmg}_\sigma(x, y)$ $\sigma^2 \Delta G_\sigma(x, y)$

Figure 5.4: SIFER (left) and SIFT (right) filter kernels. The SIFER kernel, a Gaussian modulated along the two axes by cosine functions is not rotation invariant, while the difference of Gaussians used in SIFT is.

	ρ	ζ
SIFT	$6\sqrt{2}$	6
Hessian-Laplace	$6\sqrt{2}$	6
Hessian-Affine	$6\sqrt{2}$	6
Harris-Laplace	$6\sqrt{2}$	6
Harris-Affine	$6\sqrt{2}$	6
MSER	2	-

	ρ	ζ
EBR	1	-
IBR	1	-
SURF	$10\sqrt{2}$	3.3
SFOP	$6\sqrt{2}$	6
BRISK	$3/2\sqrt{2}$	$3/2$
SIFER	$6\sqrt{2}$	6

Table 5.2: Summary of the parameters controlling the spatial coverage of a detection for each evaluated method. The parameter ρ controls the size of the patch encoded in the descriptor. For methods that apply a Gaussian weighting window to the described patch, the parameter ζ controls the standard deviation of the Gaussian function.

5.5 and 5.6 show the detection maps for the twelve compared methods on the `siemens` pattern and on the `bike` image from the Oxford dataset [Mikolajczyk et al. 2005].

The detection number varies from one method to the other, and also from one sequence to the next. MSER generally detects fewer features than the rest while SIFT and the Harris and Hessian based methods detect many more.

The rotation invariance of the methods is easily tested by examining the detections on the `siemens star` test image shown in Figure 5.5. Unsurprisingly, SIFT and SFOP are rotation invariant while SIFER is not. More surprisingly, the Hessian and Harris based methods are not rotation invariant. Although the Hessian determinant and the Laplacian of the Gaussian smoothing are isotropic, the methods fail to maintain the theoretical invariance properties due to the discretization of the differential operators.

Several feature detectors generate multiple detections from a single local feature. This is clearly the case for Harris-Affine, Hessian-Affine and, to a lesser extent, for BRISK. In general, with the exception of SIFT, SFOP and MSER, all the detectors appear to be visually highly redundant.

In some cases, while detections are numerous, they cluster on a reduced part of the scene. This is observed for instance with SIFER, (see e.g., Figure 5.6). This seems to imply that the information contained in the descriptors computed from SIFER keypoints is both redundant and incomplete.

5.5 Experiments

Using the proposed non-redundant repeatability criterion, we examined the performance of the described feature detectors on the Oxford dataset [Mikolajczyk et al. 2005][4]. The Oxford dataset contains eight sequences of six images each designed to help assess the stability of the detections with respect to habitual image perturbations, namely, rotation and scale changes, viewpoint changes, camera blur, illuminations changes and JPEG compression artefacts. The eight sequences are shown in Figure 5.7. The original and publicly available binaries of all but one methods were used[5]. No reference implementation of SIFER was available, we therefore relied on our own implementation rigorously following the published description [Mainali et al. 2013]. The parameters of each method were set to their default values. All scripts and codes are available for download [6].

The performance evaluation of a detector is two-dimensional. On the one hand, a detector should produce as many detections as possible, while on the other, it should keep to a minimum the number of non-repeatable detections. In other words, the best detector is the one that has simultaneously the largest repeatability ratio and the largest number of detections.

As we showed in the previous section, a quick visual examination of the detection maps already reveals that some methods are more redundant than others. For example, it is

[4]Dataset available at `http://www.robots.ox.ac.uk/~vgg/research/affine/`

[5]Methods binaries `http://www.robots.ox.ac.uk/~vgg/research/affine/`, `http://docs.opencv.org/doc/tutorials/features2d/feature_detection/feature_detection.html`, `http://www.vision.ee.ethz.ch/~surf/` and `http://www.cs.ubc.ca/~lowe/keypoints/` `http://www.ipb.uni-bonn.de/sfop/`

[6]In particular a documented and optimized version of the repeatability criteria [Mikolajczyk et al. 2005] along with the two variants discussed in Section 5.2 are available for download at `http://dev.ipol.im/~reyotero/comparing_20140906.tar.gz`.

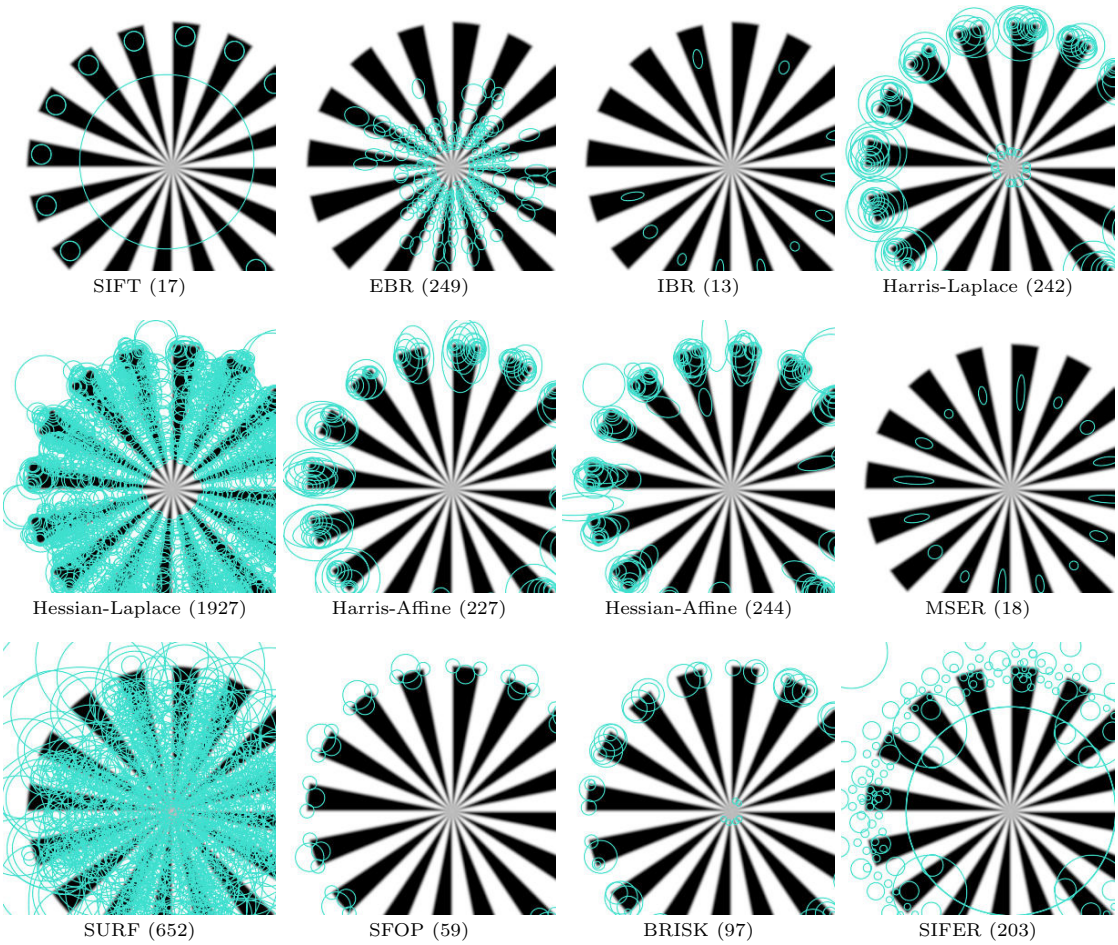


Figure 5.5: Keypoints map on `siemens star` test image. For a better readability of the figure, the descriptor ellipses are reduced to one sixth of their real size. Thus, when two ellipses overlap, their associated descriptors are in strong overlap. This is particularly conspicuous for the Hessian and Harris detectors. The total number of detected keypoints by each method is shown in brackets. SIFT and SFOP seem to be the only (experimentally) rotationally invariant methods. The elliptical shapes deduced from the MSER regions have different sizes in each rotated triangle. By design, SIFT detects blob like structures and SFOP additional features, such as corners and edges.

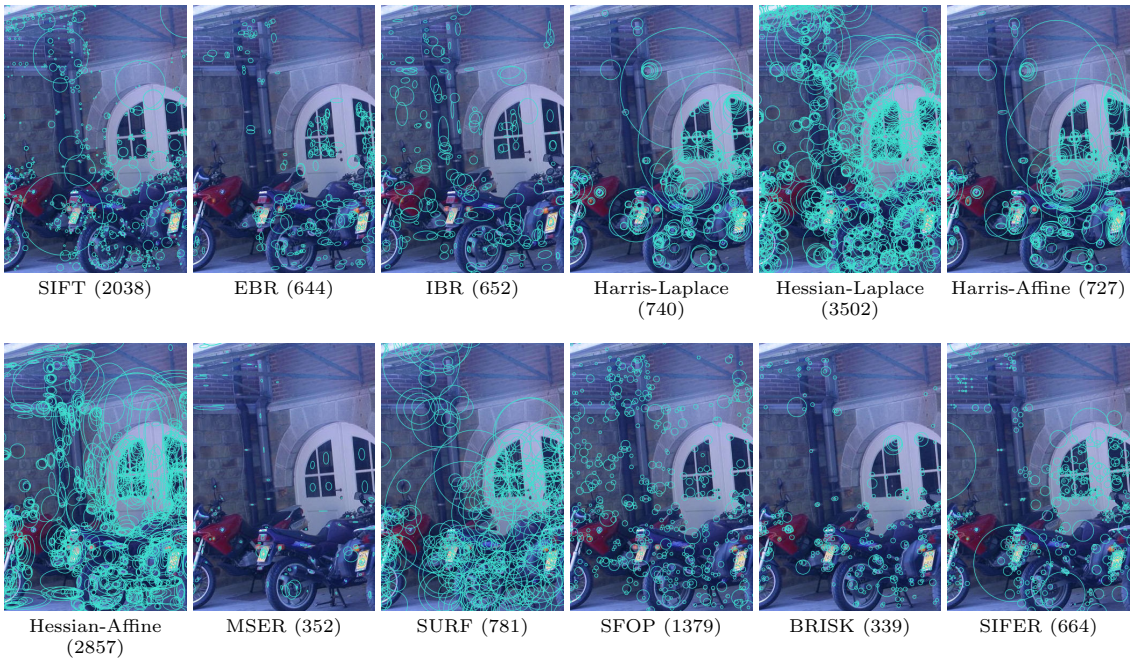


Figure 5.6: Keypoints map on an image from the `bikes` sequence. For a better readability of the figure, we reduced six times the descriptors ellipses with respect to their real size. This also means that when two ellipses overlap, their associated descriptors are in strong overlap. The total number of detected keypoints by each method is shown in brackets. The number of detections significantly varies with the algorithm. Hessian based methods and SIFT produce many more detections than the rest. All methods, with the exception of IBR and EBR, detect features at very different scales. In particular, SIFT and SFOP detect very small structures. Most algorithms detect the same structure several times, producing significantly overlapped detections. The SIFER detections are disturbingly concentrated on clusters not necessarily overlapped. Yet the proposed non-redundant repeatability metric will not penalize such behavior. For the Harris and Hessian based methods, note how corners generate trails of detections of increasing size.

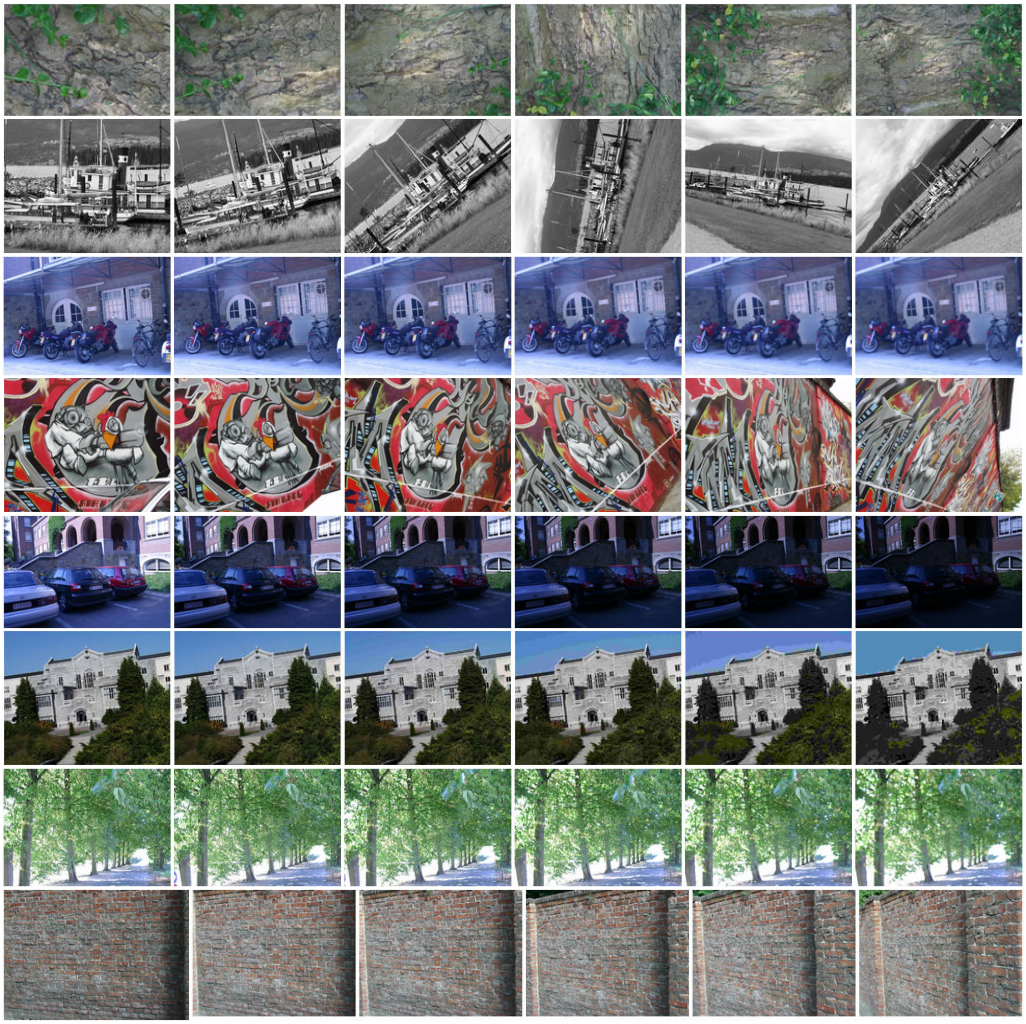


Figure 5.7: The Oxford dataset. Different series of images simulating different image transformations. From top to bottom: `bark` and `boat` (scale changes and rotations), `bikes` (camera blur), `graf` (viewpoint changes), `leuven` (illumination changes), `trees` camera blur, `ubc` (JPEG compression), `wall` (viewpoint changes) image series.

clear from Figure 5.5 (`siemens star`) that SIFER, SURF and the Hessian based methods produce highly redundant detections. The non-redundancy ratio shown in Table 5.3 **(a)** for the eight Oxford sequences helps rank the methods in terms of redundancy. With non-redundant ratios lower than 7% on all eight sequences, the Hessian based detectors are the most redundant methods. On the other end of the spectrum, the least redundant method is MSER having an average non-redundant ratio of 51%. SIFT and its SIFT-single variant come second, with non-redundant ratios ranging from 20% to 36%. Since the number of detections of SIFT and Hessian-Laplace are comparable (Table 5.3 **(b)**), the cost of extracting and matching descriptors is similar for both methods. Notwithstanding this fact, SIFT produces well-spread detections while the Hessian-Laplace are redundant and overlapped. Under such circumstances, we expect that taking into account the descriptors overlap will change significantly the hierarchy given by the repeatability rates.

The classic repeatability and the non-redundant repeatability rates as well as the number of detections for the eight Oxford sequences are provided in Table 5.3. Also, in Figure 5.8 the average repeatability rates for all the compared detectors are plotted as a function of the number of detections. Note that in general, the number of repeated points oscillates around 40% of the total number of detections. This is a much lower rate than usually achieved with the more permissive definition of the repeatability criterion, see Section 5.2.

As previously said, the repeatability score must be compared alongside the number of detections to have a complete performance evaluation of detectors. The methods that provide in general the largest number of detections are SIFT, SIFER and the Hessian based methods. MSER, EBR and IBR produce significantly less detections. The methods that are the most redundant happen to be also the methods that perform well according to the classic repeatability criteria (see Table 5.3 **(d)**). Indeed, the Hessian based methods are among the methods with largest repeatability while providing numerous detections. Note that SFOP is outperformed by the Harris based methods in all eight sequences, while providing a similar number of detections.

These conclusions are drastically altered when the redundancy of detections is taken into account. According to the non-redundant repeatability shown in Table 5.3 **(d)**, the hardly redundant SIFT method achieves one of the top three best scores while providing in general one of the largest number of detections. The Hessian based methods and SIFER, while achieving detection numbers comparable to those of SIFT, perform poorly according to the non-redundant repeatability. Despite having fewer detections, the non-redundant repeatability of SURF is lower than the one of SIFT in five sequences out of eight. Unlike what was concluded with the classic criterion, SFOP outperforms the Harris based methods in seven out of eight sequences. In fact, SFOP performs generally well. In all sequences, SFOP is one of the three best algorithms according to the non-redundant repeatability while it performed poorly for the traditional repeatability. On average, MSER and IBR produce the best non-redundant repeatability scores. Nevertheless, with up to ten times more detections, SIFT should be preferred to MSER except for severe changes of viewpoint (see Figure 5.8). In principle, MSER is not blur invariant. Yet, it performs surprisingly well on the sequence `bikes`, containing well contrasted large geometric features. MSER may benefit here from its low number of detections.

To summarize the relative performance of each method on the entire Oxford data set we proceeded as follows. First, the number of detections, the repeatability and non-redundant repeatability rates on each sequence were rescaled to cover the interval $[0, 1]$.

Then, we computed the mean of the rescaled detectors performance over the eight sequences. Figure 5.9 shows the relative repeatability and non-redundant repeatability scores as a function of the number of the normalized number detections. In this map a method performs optimally if it is simultaneously extremal in ordinate and in abscissa, and performs well if it is extremal in at least one of the coordinates. Thus, the normalized benchmark reveals that the ranking of detectors is severely disrupted when considering the detectors redundancy. While for example Harris and Hessian based methods, SURF and EBR significantly reduce their performance (going down in the plot), MSER and BRISK improve their relative position to the others. When the redundancy is not taken into account the method producing the most detections and with the highest repeatability is Hessian Laplace, while when considering the non-redundant variant it is SIFT.

	bark *scale*	bikes *blur*	boat *scale*	graf *viewp*	leuven *illum*	trees *blur*	ubc *jpeg*	wall *viewp*	*mean*
SIFT	**0.26**	**0.27**	**0.26**	**0.29**	**0.35**	**0.20**	**0.24**	**0.23**	*0.24*
SIFT-S	**0.31**	**0.32**	**0.31**	**0.34**	**0.41**	**0.24**	**0.29**	**0.27**	*0.29*
EBR	0.23	0.18	0.10	0.11	0.15	**0.23**	0.1	0.08	*0.12*
IBR	0.20	0.24	0.21	0.21	**0.30**	0.20	**0.21**	**0.28**	*0.22*
HARLAP	0.11	0.10	0.06	0.06	0.12	0.04	0.08	0.07	*0.07*
HESLAP	0.04	0.04	0.03	0.03	0.05	0.03	0.04	0.03	*0.03*
HARAFF	0.12	0.10	0.07	0.07	0.13	0.05	0.08	0.08	*0.07*
HESAFF	0.04	0.05	0.05	0.04	0.07	0.03	0.04	0.04	*0.04*
MSER	**0.61**	**0.55**	**0.51**	**0.55**	**0.58**	**0.48**	**0.55**	**0.48**	*0.51*
SURF	0.16	0.16	0.11	0.11	0.16	0.10	0.13	0.13	*0.12*
SFOP	0.17	0.24	**0.21**	0.26	0.25	0.17	0.19	0.18	*0.20*
BRISK	0.26	**0.28**	0.14	**0.27**	0.26	0.10	0.17	0.13	*0.15*
SIFER	**0.31**	0.23	0.18	0.21	0.22	0.16	0.18	0.19	*0.19*

(a) Average *non-redundant ratio* $\mathrm{nr} := K_{\mathrm{nr}}/K$.

	bark *scale*	bikes *blur*	boat *scale*	graf *viewp*	leuven *illum*	trees *blur*	ubc *jpeg*	wall *viewp*	*mean*
SIFT	**1022**	**1035**	**3803**	**1907**	**1737**	**9143**	**5296**	**8678**	*4077*
SIFT-S	**848**	871	**3226**	**1642**	1474	7507	**4273**	**7255**	*3387*
EBR	75	366	665	577	458	535	756	2012	*681*
IBR	132	573	281	294	238	1141	563	453	*459*
HARLAP	118	541	1439	1121	568	4420	1540	1963	*1465*
HESLAP	**815**	**2936**	**2795**	**3165**	**2233**	**8202**	**3594**	**4914**	*3582*
HARAFF	120	533	1392	1103	556	4397	1501	1932	*1442*
HESAFF	**807**	**2470**	2217	**2180**	**1539**	**7876**	3146	4798	*3129*
MSER	85	195	592	280	276	1839	716	1373	*670*
SURF	183	547	948	913	608	3000	1194	1564	*1120*
SFOP	476	**1041**	826	530	1014	3293	1859	2243	*1410*
BRISK	119	194	1150	374	521	3012	1409	2413	*1150*
SIFER	159	730	**4321**	1571	**2591**	**8818**	**6610**	**8535**	*4167*

(b) Average number of detections in the common area.

	bark *scale*	bikes *blur*	boat *scale*	graf *viewp*	leuven *illum*	trees *blur*	ubc *jpeg*	wall *viewp*	*mean*
SIFT	23.4	44.3	17.6	11.8	42.5	6.7	29.1	8.0	*22.9*
SIFT-S	23.3	44.6	18.1	11.9	43.5	7.1	30.0	8.3	*23.4*
EBR	7.5	**66.6**	**53.5**	**38.6**	55.1	16.0	51.4	**38.8**	*40.9*
IBR	37.2	51.9	46.4	**50.6**	58.1	**33.4**	45.6	36.1	*44.9*
HARLAP	**52.5**	52.4	40.2	21.3	50.2	**23.2**	**73.6**	29.9	*42.9*
HESLAP	**57.9**	**69.5**	**50.0**	22.4	**70.1**	**33.1**	**73.8**	**36.4**	*51.7*
HARAFF	48.6	50.0	36.7	26.9	47.5	20.2	**71.8**	27.9	*41.2*
HESAFF	**54.7**	**66.8**	46.8	**30.7**	**65.9**	**28.4**	**72.7**	35.8	*50.2*
MSER	32.9	52.2	42.4	**55.6**	**72.8**	18.0	44.8	**40.4**	*44.9*
SURF	**63.6**	**72.6**	**48.2**	19.4	**64.6**	29.5	70.9	**36.7**	*50.7*
SFOP	29.7	31.8	25.9	13.7	42.6	8.4	36.2	18.8	*25.9*
BRISK	2.4	9.9	4.0	4.3	18.2	5.4	16.6	5.8	*8.3*
SIFER	1.4	49.9	7.4	1.5	37.5	9.1	50.9	10.0	*20.9*

(c) Average repeatability.

	bark *scale*	bikes *blur*	boat *scale*	graf *viewp*	leuven *illum*	trees *blur*	ubc *jpeg*	wall *viewp*	*mean*
SIFT	7.3	**15.2**	6.6	4.5	**19.2**	3.2	10.6	3.7	*8.8*
SIFT-S	8.8	**18.1**	**7.8**	5.3	**22.6**	3.9	**13.1**	4.4	*10.5*
EBR	5.3	**15.4**	6.6	**9.2**	10.3	**6.8**	7.4	5.4	*8.3*
IBR	**19.8**	15.2	**15.4**	**17.5**	**26.2**	**11.9**	**13.7**	**17.7**	*17.2*
HARLAP	**11.1**	9.1	4.1	2.6	10.1	3.1	6.8	4.9	*6.5*
HESLAP	3.8	3.7	2.4	1.2	4.6	2.1	3.5	2.6	*3.0*
HARAFF	11.1	9.4	4.0	4.2	10.4	3.0	7.2	5.3	*6.8*
HESAFF	4.1	4.6	2.8	2.8	6.4	2.2	4.1	3.0	*3.7*
MSER	**27.2**	**35.9**	**24.0**	**32.8**	**49.8**	**13.1**	**29.9**	**25.4**	*29.8*
SURF	**14.7**	13.3	7.1	3.7	14.1	**5.6**	10.2	**8.0**	*9.6*
SFOP	10.0	11.5	**10.3**	**6.2**	16.7	4.2	10.7	**6.1**	*9.5*
BRISK	2.3	7.2	2.7	3.4	11.8	3.5	7.7	3.9	*5.3*
SIFER	1.2	14.7	3.4	1.2	12.3	3.7	**11.2**	3.8	*6.4*

(d) Average non-redundant repeatability.

Table 5.3: Detectors comparison regarding repeatability and non-redundant repeatability rates on the eight sequences of the Oxford dataset. The algorithm with best number is colored in **red** and the next three in **bordeaux**. Each table focuses on a single metric: the (non-redundant) repeatability or the number of detections. A fair comparison should consider both metrics simultaneously (see Figure 5.8).

Matching scenario. We also explored the algorithms performance on a matching scenario. For that purpose, we adopted the same protocol as in [Mikolajczyk et al. 2005]. Each detector is combined with a SIFT descriptor. Around each detection, a patch is extracted to compute the dominant orientation and a SIFT feature vector. The width of the extracted patch is computed as the mean of the detected ellipse radii multiplied by the method's parameter ρ, as described for each method in Section 5.4. For all the SIFT feature vectors in one image we found the most similar feature vector on the other image (in terms of the Euclidean distance). If the distance to the most similar one is less than

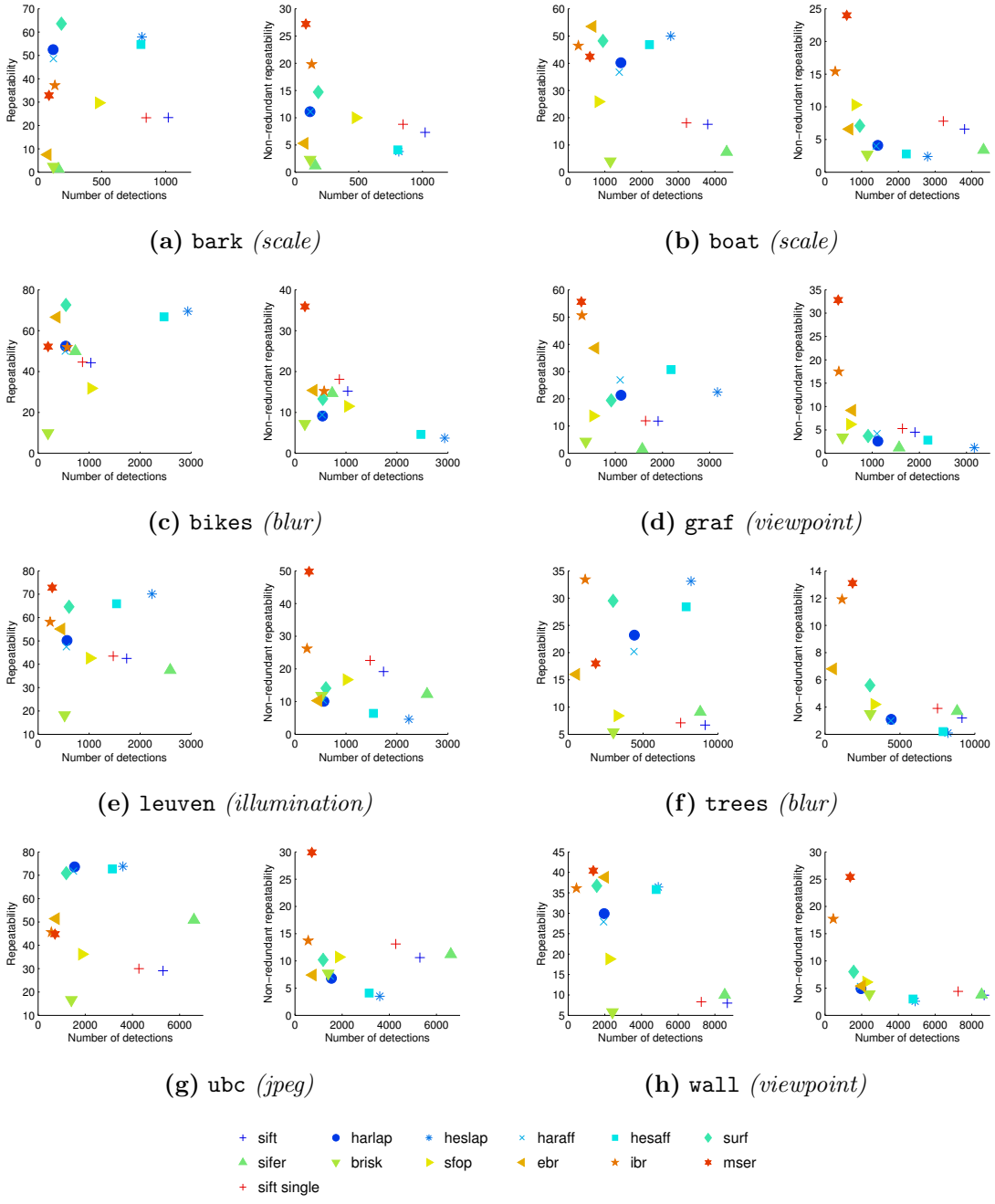


Figure 5.8: The average of the repeatability and non-redundant repeatability on each Oxford sequence is plotted as a function of the average number of keypoints detected. The performance evaluation of a detector is two-dimensional. On the one hand, a detector should detect as many keypoints as possible (abscissa). On the other, the detections should be as repeatable as possible (ordinate). Good detectors are on the top-right region of this plot. To compare a single detector performance the reader might follow the relative ordinate position of a particular detector in a particular scene in the traditional repeatability (left) and the non-redundant repeatability plots (right). For instance, MSER and SIFT algorithms always go up from the traditional to the non-redundant repeatability plots. This means that MSER and SIFT detections are less redundant than the average.

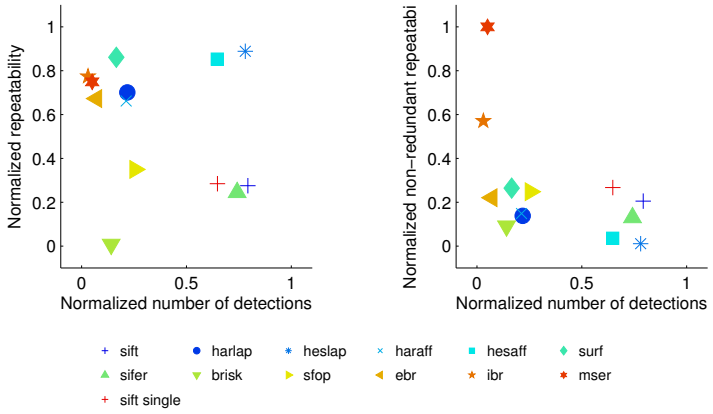


Figure 5.9: Qualitative visualization of the methods repeatability performance. Average over eight sequences. For each sequence, the number of detections, the matching rates and the non-redundant matching rates are scaled to the full range $[0, 1]$ and averaged into a single map. Once normalized, the mean values of each method over the eight sequences are computed. On the left, the *normalized* repeatability is plotted as a function of the *normalized* number of detections. On the right, the *normalized* non-redundant repeatability is plotted as a function of the *normalized* number of detections. The same conclusions observed in each of the eight Oxford sequences apply in this qualitative contest.

60% of the distance to the second nearest feature, then the pair of detections is considered as a match (as proposed in [Lowe 2004]) Table 5.4 **(a)** gives the number of detections in the common area. Table 5.4 **(b)** shows the average total number of matches while Table 5.4 **(c)** presents the number of correct matches, namely those that are consistent with the ground truth. Like in the repeatability criterion, one match is considered correct if the overlap error between the two matched keypoints (elliptical regions) is inferior to 40%. Table 5.4 **(d)** gives the number of non-redundant correct matches.

Due in part to their large number of detections, the Hessian based methods achieve in general the largest number of correct matches. In particular, in the `ubc` sequence, the Hessian-Laplace and Hessian-Affine provide almost twice more correct matches than SIFT on average. However, this apparent advantage of the Hessian based methods fades away once the detection redundancy is taken into account, as revealed by the number of non-redundant correct matches.

SIFT and its single orientation variant achieve the largest number of non-redundant correct matches in most sequences. Although SIFER produces on average the maximum number of non-redundant correct matches on the whole data set, it performs poorly on two sequences (`graf` and `bark`). In Figure 5.10, the average ratios of correct matches for the 13 compared detectors are plotted as a function of the number of detections. Figure 5.11 summarizes the methods matching performance relatively to each other. For that purpose, the number of detections, the ratio of correct matches and the ratio of non-redundant correct matches were rescaled, and the mean values over the eight sequences of the rescaled ratios are plotted as a function of the normalized number of detected keypoints.

Similarly to what we have observed on the repeatability ratio, the normalized matching benchmark reveals that the ranking of detectors is significantly disrupted when considering the detectors redundancy. Indeed, when the redundancy is not taken into account, the Hessian Laplace detector is the one producing more detections and more number of

	bark scale	bikes blur	boat scale	graf viewp	leuven illum	trees blur	ubc jpeg	wall viewp	mean
SIFT	**1022**	**1035**	**3803**	**1907**	**1737**	**9143**	**5296**	**8678**	*4077*
SIFT S	**848**	871	**3226**	**1642**	1474	7507	**4273**	**7255**	*3387*
EBR	75	366	665	577	458	535	756	2012	*681*
IBR	132	573	281	294	238	1141	563	453	*459*
HARLAP	118	541	1439	1121	568	4420	1549	1963	*1465*
HESLAP	**815**	**2936**	**2795**	**3165**	**2233**	**8202**	**3594**	**4914**	*3582*
HARAFF	120	533	1392	1103	556	4397	1501	1932	*1442*
HESAFF	**807**	**2470**	2217	**2180**	**1539**	**7876**	3146	4798	*3129*
MSER	85	195	592	280	276	1839	716	1373	*670*
SURF	183	547	948	913	608	3000	1194	1564	*1120*
SFOP	476	**1041**	826	530	1014	3293	1859	2243	*1410*
BRISK	119	194	1150	374	521	3017	1409	2413	*1150*
SIFER	159	730	**4321**	1571	**2591**	**8818**	**6610**	**8535**	*4167*

(a) Average number of detections in the common area.

	bark scale	bikes blur	boat scale	graf viewp	leuven illum	trees blur	ubc jpeg	wall viewp	mean
SIFT	**328**	272	**567**	**171**	**518**	335	890	**1190**	*534*
SIFT-S	**388**	**322**	**638**	**199**	**585**	370	**1040**	**732**	*534*
EBR	5	60	20	14	52	15	186	0	*44*
IBR	9	70	13	13	30	49	112	20	*40*
HARLAP	21	203	244	53	154	339	943	210	*271*
HESLAP	**168**	**1125**	**378**	**125**	**654**	**705**	**2023**	**573**	*719*
HARAFF	9	156	125	49	123	226	840	202	*216*
HESAFF	50	**857**	148	**68**	400	**508**	**1636**	567	*529*
MSER	7	67	38	12	109	61	194	155	*80*
SURF	47	311	178	54	233	**410**	741	261	*280*
SFOP	**133**	310	218	64	357	186	588	384	*280*
BRISK	5	29	68	20	115	126	345	160	*108*
SIFER	9	**313**	**384**	55	**873**	**553**	**2330**	**1694**	*776*

(b) Total number of matches.

	bark scale	bikes blur	boat scale	graf viewp	leuven illum	trees blur	ubc jpeg	wall viewp	mean
SIFT	**107**	241	**365**	**105**	**420**	162	759	**303**	*308*
SIFT-S	**133**	286	**413**	**124**	**475**	180	894	128	*329*
EBR	0	57	13	7	44	7	174	0	*38*
IBR	2	65	8	6	27	30	102	14	*32*
HARLAP	19	189	**220**	48	141	258	**928**	174	*247*
HESLAP	**138**	**1048**	**327**	**102**	**610**	**537**	**1942**	**456**	*645*
HARAFF	8	143	103	42	109	172	820	159	*194*
HESAFF	41	**782**	123	**50**	366	**372**	**1585**	**443**	*470*
MSER	5	66	32	8	106	51	190	134	*74*
SURF	41	**294**	157	45	211	**312**	695	228	*248*
SFOP	**76**	249	190	48	296	116	532	242	*217*
BRISK	2	14	28	8	57	48	177	51	*48*
SIFER	0	**286**	136	10	**704**	**263**	**2196**	**504**	*512*

(c) Number of correct matches.

	bark scale	bikes blur	boat scale	graf viewp	leuven illum	trees blur	ubc jpeg	wall viewp	mean
SIFT	**47**	**107**	**173**	**53**	**234**	**91.0**	**344**	**181**	*154*
SIFT-S	**52**	**119**	**190**	**60**	**265**	**101.2**	**387**	70	*156*
EBR	0	7	2	1	5	2	8	0	*3*
IBR	0	10	2	1	7	7	9	12	*6*
HARLAP	7	38	37	11	39	59	89	48	*41*
HESLAP	**19**	80	39	14	75	**78**	93	64	*58*
HARAFF	3	36	29	12	37	51	90	51	*39*
HESAFF	12	84	29	**14**	70	70	99	72	*56*
MSER	2	43	26	6	81	39	129	**106**	*54*
SURF	11	48	29	11	49	64	72	69	*44*
SFOP	**31**	**97**	**70**	**22**	**130**	63	**161**	**96**	*84*
BRISK	1	8	20	6	39	32	81	39	*28*
SIFER	0	**91**	**74**	7	**253**	**110**	**537**	**211**	*160*

(d) Number of non-redundant correct matches.

Table 5.4: The matching performance of the compared detectors on the eight sequences of the Oxford dataset. Average values are rounded to the nearest integer. In **red** the algorithm with the largest number in the column. The other top three are in **bordeaux**. The best algorithm is the one that produces the largest number of correct (non redundant) matches, provided it does not make too many detections. This is a bi-dimensional criterion that is not fully represented in a single table. Another comparison will consider both components simultaneously (Figure 5.10).

correct matches per detection. If instead we consider the redundancy, SIFT is the method producing more detections and more non-redundant correct matches per detection.

Interestingly, computing a single orientation for each keypoint improves the performance of the SIFT method. Indeed, this lowers the computational cost of descriptor computations, increases the non-redundant repeatability and maintains the number of non-redundant correct matches.

5.6 Discussion

In this paper, we have shown that the classic repeatability criterion is biased towards favoring algorithms producing redundant overlapped detections. This bias motivated the introduction of a variant of the repeatability rate taking into account the descriptor overlap. To illustrate the new repeatability criterion, the performance of several state-of-the-art methods was examined. Experimental evidence showed that, once the descriptors overlap is taken into account, the traditional hierarchy of several popular methods is severely disrupted. Thus, the detections and associated descriptions generated by some methods are highly correlated. Such redundant parasite detections are arguably caused by scale-space

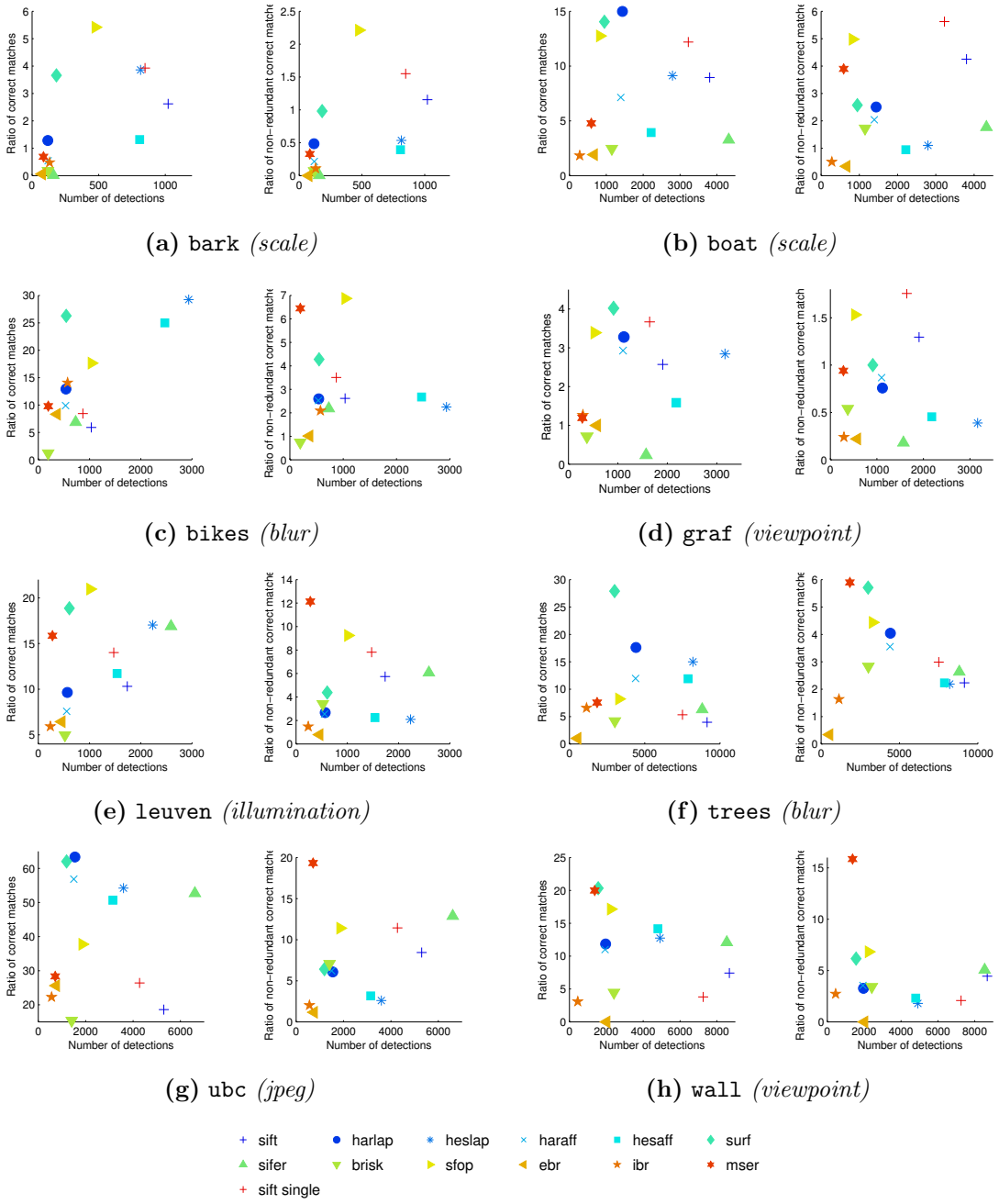


Figure 5.10: Ratio of correct matches (left) and non-redundant correct matches (right) i.e., the number of matches over number of detections in the area covered by both images. Again, to compare a single detector matching performance the reader might follow the relative ordinate position of a particular detector in a particular scene. Generally, MSER, SIFT and SFOP algorithms go up once the redundancy of matches is taken into account. On the other side, Hessian based methods and EBR/IBR always go down once the matches redundancy is taken into account.

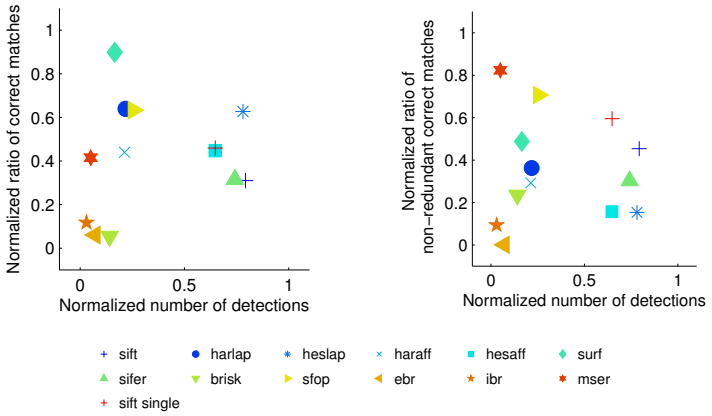


Figure 5.11: Qualitative visualization of the methods relative matching performances. Average over eight sequences. For each sequence, the number of detections, the matching rates and the non-redundant matching rates are scaled to the full range $[0, 1]$ and averaged into a single map. In a matching scenario taking into account the redundancy of matches, SIFT outperforms Hessian based methods.

sampling issues (as in the case of Hessian and Harris based methods) or the method's design. For example, the SIFER's kernel generates clusters of scale space extrema for each blob. The proposed repeatability criterion seems in agreement with the redundancies observed on patterns and on natural images. It also agrees with the detectors matching performance when combined with a common descriptor technique. Experimental evidence reveals that the SIFT and SFOP methods perform best overall as they offer the best balance between a large number of detections and a strong non-redundant repeatability, while MSER performs best for strong affine distortions with fewer detections. The amended metric aims at giving a general yet realistic assessment of keypoint detectors. The revisited benchmark along with detection maps on simple patterns seems to invalidate the performance gains reported over the last decade.

6 Conclusion

This dissertation contributes to an in-depth analysis of the SIFT method. We started this analysis by reviewing three algorithms implementing the Gaussian convolution with a focus on the accurate computation of the Gaussian scale-space, which gives the SIFT method its invariance properties. We proved that only an exact Fourier based method achieves full consistency with the scale-space requirements. The conclusion of this meticulous analysis, as evident as they may be, have a strong impact on the conception and performance not only of the SIFT method but also of any other algorithms using the Gaussian scale-space.

Feature detectors are complex chains of transformations. One can regret that the level of detail in most of the work published since D. Lowe's seminal paper is rarely sufficient to permit their complete unambiguous implementation. We provided here a meticulous description of the SIFT method. Following the standards of reproducible research, this description was published along with a peer-reviewed source code and a online demonstrator permitting to vary all parameters and explore their impact on each single intermediate step for the algorithm. This essential research tool also turned out to be a very useful teaching resource. With this in mind, we are currently preparing a MOOC (massive open online course) on feature detectors that will be based on this dissection.

Although the SIFT method has proven to be sufficiently scale invariant to be used in numerous applications, the invariance claims as well as the parameter choices of the method have not undergone a serious scrutiny. In this dissertation, we performed a numerical analysis of the SIFT method that aimed at assessing the invariance claims of SIFT and checking that the SIFT method successfully detects all of the DoG extrema. The research methodology developed to that end consists of defining a strict image simulation framework and expanding the SIFT algorithm to make it an exact and fully customizable method. We examined the influence of the level of blur in the image and that of the scale-space sampling. Among the practical conclusions of this analysis are that oversampling the scale-space improves the stability and the precision of the detection but this is not enough to achieve perfect stability. We also conclude that using the detector response to discard unstable detections is not an efficient strategy. But most importantly, this analysis has demonstrated that this research methodology is essential for the understanding of a feature detector as well as for the design of new ones. Indeed, this methodology can be extended to design purpose built feature detectors (and descriptors), whether they are intended for fast computation, affine invariance, noise robustness, or any kind of image distortion. This will not only lead to better feature detectors (for a given problem) but it will also undoubtedly provide an explanation for why they perform better.

The number of proposed feature detectors keeps increasing. Sadly, instead of producing purpose-built methods that perform well on a specific problem, this very active field of

research focuses on an illusory quest for the best all-around method. This quest explains the popularity of the repeatability criterion, a general performance metric for keypoint detectors. We identified that this criterion leads to a bias towards redundant methods and proposed an amended criterion that takes into account the overlap of detections. The amended criterion was used to revise a popular benchmark of feature detectors. We demonstrated that the SIFT method has not been significantly improved by more recent methods. The numerous methods claiming to outperform the SIFT method yield in fact similar or poorer performance.

Bibliography

S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S.M Seitz, and R. Szeliski. Building Rome in a day. *Commun. ACM*, 54(10):105–112, 2011.

M. Agrawal, K. Konolige, and M.R. Blas. CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching. In *ECCV*. 2008.

Luis Alvarez and Freya Morales. Affine morphological multiscale analysis of corners and multiple junctions. *International Journal of Computer Vision*, 25(2):95–107, 1997. .

Luis Alvarez, Frédéric Guichard, Pierre-Louis Lions, and Jean Michel Morel. Axioms and fundamental equations of image processing. *Archive for Rational Mechanics and Analysis*, 123:199–257, 1993. .

C. Ancuti and P. Bekaert. SIFT-CCH: Increasing the SIFT distinctness by color co-occurrence histograms. In *ISPA*, 2007.

J Babaud, A P Witkin, M Baudin, and R O Duda. Uniqueness of the Gaussian kernel for scale-space filtering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8:26–33, January 1986. . URL `http://dl.acm.org/citation.cfm?id=11295.11298`.

H. Bay, B. Fasel, and L. Van Gool. Interactive museum guide: Fast and robust recognition of museum objects. In *IWMV*, 2006a.

H. Bay, T. Tuytelaars, and L. van Gool. SURF: Speeded Up Robust Features. In *European Conference on Computer Vision*, 2006b. .

Josef Bigün. A structure feature for some image processing applications based on spiral functions. *Computer Vision, Graphics, and Image Processing*, 51(2):166–194, 1990.

M. Brown and D. Lowe. Automatic panoramic image stitching using invariant features. *IJCV*, 74(1):59–73, 2007.

M. Brown, R. Szeliski, and S. Winder. Multi-image matching using multi-scale oriented patches. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005. .

P J Burt and E H Adelson. The Laplacian pyramid as a compact image code. *Communications, IEEE Transactions on*, 31(4):532–540, 1983. .

M. Calonder, V. Lepetit, C. Strecha, and P. Fua. BRIEF: Binary Robust Independent Elementary Features. In *European Conference on Computer Vision (ECCV)*, pages 778–792. Springer, 2010. .

F. Cao, P. Musé, and F. Sur. Extracting meaningful curves from images. *J. Math. Imaging Vision*, 22(2-3):159–181, 2005.

J. Chen, S. Shan, C. He, G. Zhao, M. Pietikainen, X. Chen, and W. Gao. WLD: A robust local image descriptor. *PAMI*, 32(9):1705–1720, 2010.

J.L. Crowley and R.M. Stern. Fast computation of the difference of low-pass transform. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):212–222, 1984. .

M. Delbracio, P. Musé, and A. Almansa. Non-parametric sub-pixel local point spread function estimation. *IPOL*, 2012. .

Rachid Deriche. Recursively implementating the Gaussian and its derivatives. Research report, INRIA, 1993.

T. Dickscheid and W. Förstner. Evaluating the suitability of feature detectors for automatic image orientation systems. In *Computer Vision Systems*, pages 305–314. Springer, 2009.

T. Dickscheid, F. Schindler, and W. Förstner. Coding images with local features. *IJCV*, 94(2):154–174, 2011.

S. Ehsan, N. Kanwal, A.F. Clark, and K.D. McDonald-Maier. Measuring the coverage of interest point detectors. In *Image Analysis and Recognition*, pages 253–261. 2011.

S. Ehsan, A.F. Clark, and K.D. McDonald-Maier. Rapid online analysis of local feature detectors and their complementarity. *Sensors*, 13(8):10876–10907, 2013.

R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.

W. Förstner. A framework for low level feature extraction. In *Computer VisionECCV'94*, pages 383–394. Springer, 1994. .

W. Förstner, T. Dickscheid, and F. Schindler. Detecting interpretable and accurate scale-invariant keypoints. In *IEEE International Conference on Computer Vision*, 2009. .

C. Gasquet and P. Witomski. *Fourier Analysis and Applications: Filtering, Numerical Computation, Wavelets*. Texts in Applied Mathematics. Springer, 1999. ISBN 9780387984858.

P Getreuer. A survey of Gaussian convolution algorithms. *Image Processing On Line*, 2013:286–310, 2013. .

M. Grabner, H. Grabner, and H. Bischof. Fast approximated SIFT. In *ACCV*. 2006.

W. Grimson and D. Huttenlocher. *Object recognition by computer: the role of geometric constraints*. MIT Press, 1990.

R Grompone von Gioi, P. Monasse, J-M. Morel, and Z. Tang. Towards high-precision lens distortion correction. In *ICIP*, 2010.

F. Guichard, J.-M. Morel, and R Ryan. *Contrast invariant image analysis and PDE's*.

Pascal Gwosdek, Sven Grewenig, Andrés Bruhn, and Joachim Weickert. Theoretical foundations of gaussian convolution by extended box filtering. In *Scale Space and Variational Methods in Computer Vision*, pages 447–458. Springer, 2012. .

G. Haro, A. Buades, and J-M. Morel. Photographing paintings by image fusion. *SIAM J. Imaging Sci.*, 5(3):1055–1087, 2012.

C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50, 1988. .

R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

T. Hassner, V. Mayzels, and L. Zelnik-Manor. On SIFTs and their scales. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1522–1528, 2012. .

T. Iijima, H. Genchi, and K. Mori. A theory of character recognition by pattern matching method. In *Learning systems and intelligent robots*, pages 437–450. Springer, 1974. .

T. Kadir, A. Zisserman, and M. Brady. An affine invariant salient region detector. In *ECCV*. 2004.

Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE Computer Vision and Pattern Recognition*, volume 2. IEEE, 2004.

J. Koenderink. The structure of images. *Biological Cybernetics*, 50:363–370, 1984. ISSN 0340-1200. URL `http://dx.doi.org/10.1007/BF00336961`.

S. Leutenegger, M. Chli, and R.Y. Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. In *ICCV*, 2011.

B. Li, R. Xiao, Z. Li, R. Cai, B.-L. Lu, and L. Zhang. Rank-SIFT: Learning to rank repeatable local interest points. In *CVPR*, 2011.

T. Lindeberg. *Scale-space theory in computer vision*. Springer, 1993. .

C. Liu, J. Yuen, A. Torralba, J. Sivic, and W.T. Freeman. SIFT Flow: Dense correspondence across different scenes. In *European Conference on Computer Vision*. 2008. .

D. Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision*, test, 1999. .

D. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004. ISSN 0920-5691. .

P. Mainali, G. Lafruit, Q. Yang, B. Geelen, L. Van Gool, and R. Lauwereins. SIFER: Scale-Invariant Feature Detector with Error Resilience. *IJCV*, 104(2):172–197, 2013.

Elmar Mair, Gregory D Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Computer Vision–ECCV 2010*, pages 183–196. Springer, 2010. .

J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image Vision Comp.*, 22(10):761–767, 2004.

K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004. ISSN 0920-5691. .

K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *PAMI*, 27(10):1615–1630, 2005.

K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1-2):43–72, 2005. ISSN 0920-5691. .

J.-M. Morel and G. Yu. Is SIFT scale invariant? *Inverse Problems and Imaging*, 5(1): 115–136, 2011.

P. Moreno, A. Bernardino, and J. Santos-Victor. Improving the SIFT descriptor with smooth derivative filters. *Pattern Recognition Lett.*, 30(1):18–26, 2009.

O. Pele and M. Werman. A linear time histogram metric for improved sift matching. In *European Conference on Computer Vision*. 2008.

J. Rabin, J. Delon, and Y. Gousseau. A statistical approach to the matching of local features. *SIAM journal on imaging science*, 2(3):931–958, 2009. .

Maxim Raginsky and Svetlana Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *NIPS*, volume 22, pages 1509–1517, 2009.

D.B. Reid. An algorithm for tracking multiple targets. *Trans. Autom. Control*, 24(6): 843–854, 1979. .

I. Rey-Otero, J-M. Morel, and M. Delbracio. An Analysis of scale-space sampling in SIFT. In *ICIP*, 2014.

Ives Rey-Otero and Mauricio Delbracio. Anatomy of the SIFT Method. *Image Processing On Line*, 4:370–396, 2014. .

E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, 2006. .

Edward Rosten, Reid Porter, and Tom Drummond. Faster and better: A machine learning approach to corner detection. *Trans. Pattern Anal. Mach. Intell.*, 32(1):105–119, 2010.

E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. ORB: An efficient alternative to SIFT or SURF. In *ICCV*, pages 2564–2571. IEEE, 2011. .

R. Sadek. *Some problems on temporally consistent video editing and object recognition*. PhD thesis, Universitat Pompeu Fabra, 2012.

R. Sadek, C. Constantinopoulos, E. Meinhardt, C. Ballester, and V. Caselles. On affine invariant descriptors related to SIFT. *SIAM*, 5(2):652–687, 2012.

C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *IJCV*, 37(2):151–172, 2000.

S.M. Smith and J.M. Brady. SUSAN. A new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, 1997. .

N. Snavely, S.M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3D. *ACM T. Graphics*, 25(3):835–846, 2006.

J Sporring, M Nielsen, LMJ Florack, and P Johansen. *Gaussian Scale-Space Theory, volume 8 of Computational Imaging and Vision Series*. Kluwer Academic Publishers, Dordrecht, 1997.

E. Tola, V. Lepetit, and P. Fua. A fast local descriptor for dense matching. In *CVPR*, 2008.

E. Tola, V. Lepetit, and P. Fua. DAISY: An efficient dense descriptor applied to wide-baseline stereo. *PAMI*, 32(5):815–830, 2010.

T. Tuytelaars and K. Mikolajczyk. Local invariant feature detectors: A survey. *Foundations and Trends® in Computer Graphics and Vision*, 3(3):177–280, 2008. .

T. Tuytelaars and L. Van Gool. Content-based image retrieval based on local affinely invariant regions. In *VISUAL*, 1999.

T. Tuytelaars and L. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *BMVC*, 2000.

A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. In *Proc. ACM Int. Conf. Multimed.*, 2010.

J. Weickert, S. Ishikawa, and A. Imiya. Linear scale-space has first been proposed in Japan. *Journal of Mathematical Imaging and Vision*, 10(3):237–252, 1999. .

S. Winder and M. Brown. Learning local image descriptors. In *CVPR*, 2007.

S. Winder, G. Hua, and M. Brown. Picking the best DAISY. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

A Witkin. *Scale-space filtering: A new approach to multi-scale description*, volume 9, pages 150–153. Institute of Electrical and Electronics Engineers, 1984. . URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1172729`.

I T Young and Lucas J Van Vliet. Recursive implementation of the Gaussian filter. *Signal processing*, 44(2):139–151, 1995. .

G. Yu and J.-M. Morel. ASIFT: An Algorithm for Fully Affine Invariant Comparison. *IPOL*, 2011, 2011. .

J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *Int. J. of Comput. Vision*, 73(2):213–238, 2007.

H. Zhou, Y. Yuan, and C. Shi. Object tracking using SIFT features and mean shift. *Comp. Vis. Image Und.*, 113(3):345–352, 2009.