



CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

Local Descriptors for Image Matching and Retrieval



Arun Mukundan

arun.mukundan@gmail.com

Supervisor

Doc. Mgr. Ondrej Chum, Ph.D.

Co-Supervisor

Giorgos Tolias, Ph.D.

July 2020

Available at <http://cmp.felk.cvut.cz/~qqmukund/thesis.pdf>

The author was supported by
ERC CZ project - LL1303,
RCI project CZ.02.1.01/0.0/0.0/16_019/0000765,
SGS15/155/OHK3/2T/13 and
SGS17/185/OHK3/3T/13.

Published by

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University in Prague
Technicka 2, 166 27 Prague 6, Czech Republic
<http://cmp.felk.cvut.cz>
fax: +420 2 2435 7385 , phone: +420 2 2435 7637

Contents

Abstract	5
Abstrakt	7
1 Introduction	9
1.1 Contributions	15
1.2 Publications	16
1.3 Structure of thesis	16
1.4 Authorship	16
2 Related Work	19
2.1 Hand-crafted descriptors	23
2.2 Deep local descriptors	27
3 Datasets	31
3.1 PhotoTourism	32
3.2 HPatches	34
3.3 Others	37
4 Multiple-Kernel Local Descriptors	41
4.1 Preliminaries	41
4.2 Multiple-kernel joint encoding	43
4.3 Whitening	45
4.4 Visualization of kernels	48
4.5 Experiments	55
4.6 Discussion	63
5 Explicitly Spatially Encoded Deep Local Descriptors	65
5.1 Deep local descriptors	65

5.2	A match-kernel perspective	67
5.3	Implementation details	71
5.4	Experiments	75
5.5	Discussion	78
6	Robust Data Whitening	79
6.1	Background on whitening	79
6.2	Background on IRLS	81
6.3	Robust transformation estimation	82
6.4	Joint centering and transformation matrix estimation.	83
6.5	Extension with supervision	85
6.6	Examples on synthetic data	86
6.7	Experiments	88
6.8	Discussion	89
7	Conclusions	91
A	Examples	93
A.1	HardNet	94
A.2	MKDNet	96
A.3	Cartesian	98
A.4	Polar	100

Abstract

This thesis addresses the problem of image matching and retrieval using local descriptors and studies various aspects of their construction. In particular, we focus on the measurements used to describe pixels and their neighborhood and the encoding of their position. We also address post-processing of the derived descriptors, which is learned with or without supervision.

The best performing handcrafted descriptors, such as SIFT and its derivatives, measure patch similarity through modelling discrete spatial distribution of gradient directions. We extend this approach and design a multiple-kernel local descriptor based on efficient match kernels of gradient directions. Gradient positions and directions are encoded by explicitly defined kernels, that combine two parametrizations of gradient position and direction, each parametrization provides robustness to a different type of patch mis-registration: polar parametrization for noise in the patch dominant orientation detection, Cartesian for imprecise location of the feature point. We visually demonstrate the effect and semantic meaning of the parametrization and a post-processing on patch similarity.

The performance of local descriptors was significantly boosted by moving from intensity gradient measurements, which are simple convolutional operators, to deep convolutional network activations. We analyze how the conventional architecture for deep local descriptors, *i.e.* CNN followed by a fully connected layer, provides an encoding that depends on the position of the activations within the patch. We propose to replace the FC layer by explicit spatial encoding as introduced for the handcrafted descriptor, which reduces the number of parameters and makes it independent of the patch resolution. Both descriptors, handcrafted and CNN-based, achieved state-of-the-art performance at the time of publication. Our unsupervised variant is the best performing descriptor constructed without the need of labeled data, while the deep local variant consistently outperformed other deep local descriptors on standard and modern benchmarks.

Finally, we study the problem of correlation with regards to entries of high-dimensional vectors, such as global image descriptors or local descriptors, which leads to a bias in similarity estimation, necessitating a post-processing step, such as data whitening. We analyze robust estimation of the whitening transformation in the presence of outliers.

Inspired by the Iteratively Re-weighted Least Squares (IRLS) approach, we propose an approach that iterates between centering and applying a transformation matrix, a process which is shown to converge for robust class of cost functions. The approach is developed for unsupervised scenarios, and further extend to supervised cases. We demonstrate the robustness of our method to outliers on synthetic 2D data and also show improvements compared to conventional whitening on real data for image retrieval with CNN-based representation. Our robust estimation is not limited to data whitening, but can be used for robust patch rectification too.

Abstrakt

Práce se zabývá problémem párování a vyhledávání obrázků pomocí lokálních deskriptorů a studuje různé aspekty jejich konstrukce. Především se zaměřuje na míry používané pro popis pixelů a jejich okolí a na kódování jejich pozice. Práce se také zabývá post-processingem odvozených deskriptorů, který je učen s učitelem nebo bez učitele.

Nejvýkonnější ručně navržené deskriptory, jako např. SIFT a jeho odvozeniny, měří podobnost oblastí modelováním diskretních prostorových rozdělení směrů gradientů. Tento přístup rozšiřujeme a navrhujeme lokální deskriptory s více jádry, založené na efektivních jádrech pro párování směrů gradientů. Pozice a směry gradient jsou kódovány explicitně definovanými jádry, která kombinují dvě parametrizace směru a pozice gradientu, kde každá parametrizace nabízí robustnost vůči jinému typu chyby při registraci: polární parametrizace vůči šumu v detekci dominantní orientace oblasti, Kartézská vůči nepřesné lokalizaci významného bodu. Vizuálně demonstrujeme efekt a sémantický význam této parametrizace a post-processing podobnosti oblastí.

Výsledky lokálních deskriptorů se výrazně zlepšily posunem od měření gradientů intenzity, tedy jednoduchých konvolučních operátorů, k aktivacím hlubokých konvolučních sítí. Analyzujeme, jak konvenční architektura pro hluboké lokální deskriptory, tj. konvoluční síť ukončená plně propojenou vrstvou, poskytuje kódování, které závisí na pozici aktivací uvnitř dané oblasti. Navrhujeme nahradit plně propojenou vrstvu explicitním prostorovým kódováním, jako je tomu u ručně navržených deskriptorů, což snižuje množství parametrů a činí kódování nezávislým na rozlišení oblasti. Oba deskriptory, ručně navržený i hluboce naučený, dosáhly v době publikace nejlepších výsledků. Naše varianta učená bez učitele je nejvýkonnějším deskriptorem konstruovaným bez potřeby označených dat, zatímco hluboká lokální varianta konzistentně překonávala jiné hluboké lokální deskriptory na standardních i moderních datových sadách.

Na konec se zabýváme problémem korelace položek vysoce-rozměrných vektorů, jako globálních či lokálních deskriptorů, která vede k vychýlení odhadu podobnosti, vyžadujícímu další zpracování jako data whitening. Analyzujeme robustnost odhadu transformace whiteningu v přítomnosti outlierů (vybočujících pozorování). Inspirováni metodou Iteratively Re-weighted Least Squares (IRLS) navrhujeme přístup, který iteruje

mezi centrováním a aplikací transformační matice, proces který konverguje pro robustní třídu ztrátových funkcí. Přístup je vyvinut pro učení bez učitele, a dále rozšířen pro učení s učitelem. Robustnost naší metody vůči outlierům demonstrujeme na syntetických 2D datech a také ukazujeme zlepšení vůči konvenčnímu whiteningu na reálných datech pro vyhledávání obrázků s reprezentací založenou na konvolučních neuronových sítích. Náš robustní odhad není omezen jen na data whitening, ale může být použit také na robustní rektifikaci oblastí.

Introduction

Computer vision is a research field that deals with extraction of high-level information from images or videos, to perform tasks that require interpretation of visual stimuli. Computer vision methods consist of acquisition, processing, analysis and interpretation of such stimuli by construction of models based on physics, geometry, calculus, and statistics. Representations of information contained in such stimuli are very high dimensional. For example, a grayscale image of size 1024×1024 pixels already has a dimensionality of greater than a million. Processing such vast information is expensive, however, progress in sheer computational power has allowed computer vision methods to be applied to a wide variety of tasks. Advances in instance recognition allow modern handheld devices today to offer face recognition as a basic feature [1]. Commercial websites implement services to search for visually similar images across collections numbering in the millions [2], which use standard image retrieval methods. Stores entice the user to virtually try out clothes [3] or place furniture [4], which is a product of the research on augmented reality. Applications that run partly on handheld devices and partly on larger servers allow the user to finely localize themselves based on images of surroundings [5], using complex systems for localization and mapping, and matching across large databases of images. The explosion in the number of publicly available images has enabled reconstruction of entire city scapes, and digitalization of the topography of parts of the real world [6]. Tasks such as assembly line inspection and processing are also widely automated, at the risk of even replacing the current workforce in factories and distribution centres. Computer vision is also used for military purposes such as surveillance and development of autonomous weapons. Critical tasks such as healthcare are currently investing heavily in methods to automate diagnosis, allowing for care at large scales. A critical problem in many of these applications is the need to establish correspondences to match images. In this work, we attempt to address this problem and focus on the tasks of image matching and retrieval.

Many of the computer vision problems require matching of images of the natural world captured on a wide variety of cameras, in various lighting conditions, and capture various scenes. This poses significant challenges to image matching. Objects of interest may be occluded by other irrelevant objects. The object of interest may be obscured by cluttered backgrounds. There may be local deformations, or changes in textures. Illumination conditions may change the appearance of relevant regions. Appearance may also be vastly affected by changes in viewpoint. To mitigate many of these problems, popular approaches describe the image as a collection of local neighborhoods or *local features*. These approaches are successfully used in image matching [7], image retrieval [77], object detection [55], simultaneous localization and mapping [91], 3D reconstruction [37], *etc.*

In particular, many classical approaches to image matching and retrieval [90, 75, 95] are based on finding correspondences of local features between a pair of images. The robustness of the approaches relies on detecting local image structures that are covariant to viewpoint transformations, and representing them in a manner invariant to photometric transformations [99]. A typical procedure of matching images begins with extraction of local features, which consists of two parts, detection and description. Given an image, a *detector* returns a list of regions, which are rectified to canonical forms called *local-patches*. These local-patches are described by compact low-dimensional vectors called *local descriptors*. Tentative correspondences are established using a similarity measure to compare all pairs of descriptors extracted from each image, and may be refined by constraining the geometries of the respective regions to model a valid transformation from one image to the other. Finally, pairs of images which have more inliers than a given threshold are deemed to be matching images depicting the same scene. This pipeline is shown in Figure 1.1.

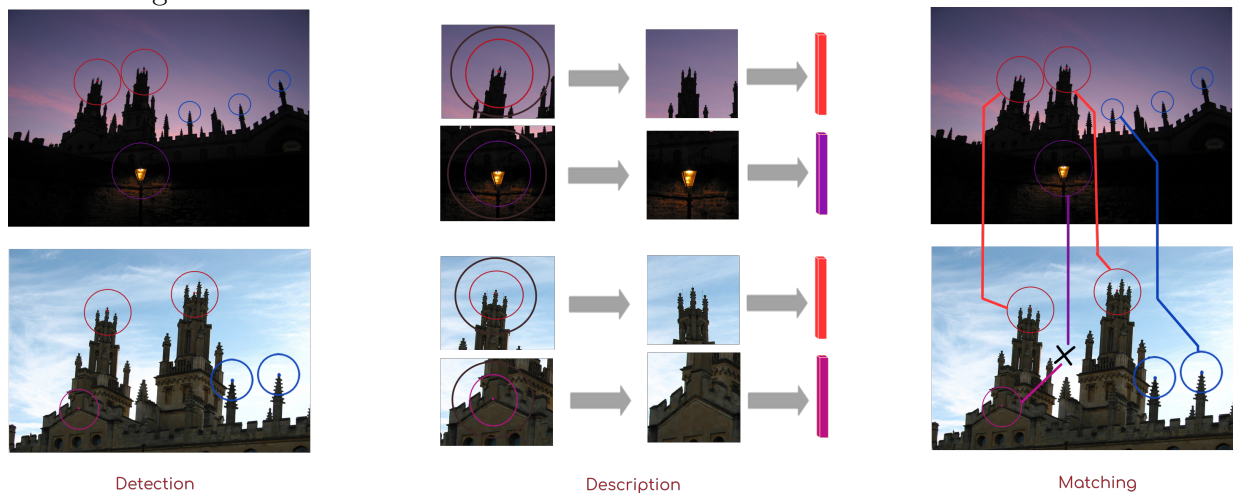


Figure 1.1: Image matching using local descriptors. Detectors detect local pixel neighborhoods, indicated by the circled regions, and return rectified canonical patches called *local-patches*. These patches are compactly described by vectors indicated by the colored bars called *local descriptors* such that similar features have similar descriptors. (Images taken from the Oxford5k dataset.)

Detectors find local pixel neighborhoods that are distinctive and can be repeatably detected. For each detected neighborhood, a rectified canonical form is computed using geometries associated with these neighborhoods, *e.g.* size, shape, orientation, *etc.*. This rectified canonical form is called a *local-patch*. The local-patch has a very useful property, namely, measurements in the coordinate system of the local-patch are invariant with respect to certain classes of image transformations. However, two rectified patches coming from matching local features are far from being identical in general. The difference has two sources, namely appearance change in imaging process and geometric misalignment. The former comes from different light conditions, imaging artifacts, *etc.*. The latter is caused by non-planarity of the surface, the detected feature covering slightly different area of the 3D surface, or incorrect rectification of the patch. These are consequences of either the appearance changes or of insufficient geometric invariance of the detector, *i.e.* affine invariant detector acting on projectively transformed surface.

Descriptors start with the local-patch as input, and compute a vector representation that robustly describes the patch and can be compared with other descriptors using a defined similarity measure. Formally, given a rectified canonical patch, $\mathcal{P} \in \mathbb{R}^{N \times N}$, we seek a description function, $\Psi : \mathbb{R}^{N \times N} \rightarrow \mathbb{R}^D$, and a similarity function, $\text{sim} : \mathbb{R}^D \times \mathbb{R}^D \rightarrow \mathbb{R}$, such that the scalar value $\text{sim}(\Psi(\mathcal{P}), \Psi(\mathcal{Q}))$ for a pair of patches \mathcal{P} and \mathcal{Q} is high if they are corresponding patches, *i.e.* the local features are coming approximately from the same surface of a 3D scene, and low if they are non-corresponding patches. A naive solution could be to simply compare corresponding pixels by vectorizing the patch as the descriptor. However, such a solution fails to account for variations in illumination, geometric misalignment, *etc.*

Many successful methods can be described as extracting appearance information from local pixel neighborhoods that compose the patch, and pooling this information into a compact vector. These approaches are reviewed in Chapter 2. Based on whether the features that describe these neighborhoods are explicitly specified, approaches to the construction of local descriptors are largely divided into two categories, hand-crafted and deep local descriptors. While hand-crafted approaches enjoy the advantages of being easy to interpret and tune, and do not need large amounts of labeled training data, deep local approaches capture rich information and achieve vastly superior performance from leveraging the said data.

Hand-crafted design of local descriptors has been popular for more than two decades with some widely used examples [55, 16, 60, 94, 25, 50]. Such descriptors do not require any training data or supervision. This kind of approach allows to easily inject domain expertise, prior knowledge or even the result of a thorough analysis [29]. Learning methods have been also employed in order to learn parts of the hand-crafted design, *e.g.* the pooling regions [103, 88], from the training data. Prior work on hand-crafted feature descriptors has shown that it is beneficial to explicitly address the geometric misalignment. Some of the approaches handling this are soft assignment of gradients to bins in SIFT and continuous spatial encoding by kernel methods in different [23] or multiple coordinate systems.

In Chapter 4, we introduce a novel hand-crafted approach based on the so called *kernel descriptors* [19, 17, 96]. They provide a quite flexible framework for matching sets, patches in our case, by encoding different properties of the set elements, pixels in our case. In particular, we build upon the hand-crafted kernel descriptor proposed by Bursuc *et al.* [23] that is shown to have good performance, even compared to deep local alternatives. Its few parameters are easily tuned on a validation set, while it is shown to perform well on multiple tasks, as we confirm in our experiments. We combine our descriptor with post-processing that is learned by the data in unsupervised or supervised ways. We show how to reduce the estimation error and significantly improve results even without any supervision.

The hand-crafted nature and simplicity of our descriptor allows to visualize and analyze its parametrization, and understand its advantages and disadvantages. It leads us to propose a simple combination of parametrizations each offering robustness to different types of patch mis-registrations. Interestingly, the same analysis is possible even for the learned post-processing. We observe that its effect on the patch similarity is semantically meaningful. The feasibility of such analysis and visualization is an advantage of our approach, and hand-crafted approaches in general, compared to deep local methods, as it is typically not straightforward to visualize and understand what the latter has learned. Deep local approaches use complex multi-million parameter models and current visualization techniques provide only partial views on their behavior over different visual patterns.

Descriptor post-processing is known to be essential to account for co-occurrences in data. A typical post-processing method is the whitening transformation, which is a linear transformation that performs correlation removal or suppression by mapping the data to a different space such that the covariance matrix of the data in the transformed space is identity. However, it has been shown [97] that an unsupervised approach based on least squares minimization is likely to be affected by outliers: even a single outlier of high magnitude can significantly deviate the solution.

In Chapter 6, we propose an unsupervised way to learn the whitening transformation such that the estimation is robust to outliers. Inspired by the Iteratively Re-weighted Least Squares approach [8], we employ robust M-estimators. We perform minimization of robust cost functions such as ℓ_1 or Cauchy. Our approach iteratively alternates between two minimizations, one to perform the centering of the data and one to perform the whitening. In each step a weighted least squares problem is solved and is shown to minimize the sum of the ℓ_2 norms of the training vectors. We demonstrate the effectiveness of this approach on synthetic 2D data and on real data of convolutional neural network representations for image search. The method is additionally extended to handle supervised cases, where we show further improvements. Finally, our methodology is not limited to data whitening. We provide a discussion on applying it for robust patch rectification of MSER features [58].

Deep local approaches use models based on Convolutional Neural Networks (CNNs). CNNs are powerful in modeling the appearance variance, while weak in modeling the geometric displacement (at least with a single FC layer). Fully Convolutional Networks (FCN) take an image or a patch as input and produce a tensor, where a vector at each spatial location can be seen as measurements over its receptive field. In the case of variable-sized, or non-aligned input, such as images, the response tensor is transformed into

a descriptor typically by some form of global pooling [78, 43, 76], which discards geometric information. In the case of aligned input of fixed size, such as rectified local-patches, the tensor is vectorized and further processed. Vectorization has similar interpretation to vectorizing spatial bins in SIFT [55]. Commonly, the vectorized tensor is processed by a single fully-connected (FC) layer [62, 93], that can be either interpreted as learned affine (linear and bias) transformation of the space, *e.g.* whitening and dimensionality reduction, or as spatially dependent embedding with efficient match kernels (EMK) [19, 23].

In Chapter 5, we propose to model the geometric misalignment by efficient match kernels that explicitly encode the spatial positions of the responses. To encode the spatial information, kernel-based explicit feature maps are used in a similar fashion to Chapter 4. The key contribution of this work is a CNN module that explicitly models the spatial information of a rectified patch. This can be seen as a transition from soft binning, *i.e.* overlapping receptive fields, to continuous efficient match kernels. In contrast to models with an FC layer, with efficient match kernels the number of model parameters does not grow with increased resolution of the input patch, *i.e.* the models for 32×32 patch input has the same number of parameters as the model for 64×64 . The applications of the proposed descriptor go beyond that of local-patches, *e.g.* tasks where encoding spatial position is essential [53, 70].

The success of deep local approaches, is critically dependent on the availability of vast datasets with patch-level annotation. Such datasets are generated using information from the Structure-from-Motion (SfM) pipeline. Adoption of these better descriptors, however, has been limited, with old-school descriptors like SIFT [55] still seeing wide usage. This is attributed to lack of large and diverse datasets, causing inconsistent claims of performance, saturation of scores on prevalent benchmarks [13], and lack of generalization to related tasks. Modern benchmarks propose expanding this to the realm to other tasks such as image matching and patch retrieval, which act as proxy for real-world applications of descriptors. They measure the ability of the descriptor to distinguish between pairs of patches that are in correspondence and otherwise is using metrics such as Receiver Operating Characteristics (ROC) and Precision and Recall (PR). Recent deep local approaches also collect their own training sets using the SfM pipeline, allowing them to supplement the training with supervision beyond positive and negative pairs. Standard benchmarks and metrics for evaluation are described in Chapter 3.

1.1 Contributions

The main contributions of the thesis:

- We propose a multiple-kernel local descriptor based on efficient match kernels from pixel gradients. It combines two parametrizations of gradient position and direction, each parametrization provides robustness to a different type of patch mis-registration: polar parametrization for noise in the patch dominant orientation detection, Cartesian for imprecise location of the feature point. Combined with whitening of the descriptor space, that is learned with or without supervision, the performance is significantly improved. We analyze the effect of the whitening on patch similarity and demonstrate its semantic meaning. Our unsupervised variant is the best performing descriptor constructed without the need of labeled data. Despite the simplicity of the proposed descriptor, it competes well with deep learning approaches on a number of different tasks.
- We propose a kernelized deep local descriptor based on efficient match kernels of neural network activations. Response of each receptive field is encoded together with its spatial location using explicit feature maps. Two location parametrizations, Cartesian and polar, are used to provide robustness to a different types of canonical patch misalignment. Additionally, we analyze how the conventional architecture, *i.e.* a fully connected layer coming after the convolutional part, encodes responses in a spatially variant way. This is replaced by explicit spatial encoding in our descriptor, whose potential applications are not limited to local-patches. We evaluate the descriptor on standard benchmarks. Both versions, encoding 32×32 or 64×64 patches, consistently outperform all other methods on all benchmarks. The number of parameters of the model is independent of the patch resolution.
- We analyze robust estimation of the whitening transformation in the presence of outliers. Inspired by the Iteratively Re-weighted Least Squares approach, we iterate between centering and applying a transformation matrix, a process which is shown to converge to a solution that minimizes the sum of ℓ_2 norms. The approach is developed for unsupervised scenarios, but further extend to supervised cases. We demonstrate the robustness of our method to outliers on synthetic 2D data and also show improvements compared to conventional whitening on real data for image retrieval with CNN-based representation. Finally, our robust estimation is not limited to data whitening, but can be used for robust patch rectification, *e.g.* with MSER features.

1.2 Publications

Chronologically ordered list of author's publications and corresponding bibliography numbers used for the rest of the thesis:

[68] Mukundan, Arun and Toliás, Giorgos and Chum, Ondřej. **Robust Data Whitening as an Iteratively Re-weighted Least Squares Problem** *SCIA*, 2017. (citations: 0)

[67] Mukundan, Arun and Toliás, Giorgos and Chum, Ondřej. **Multiple-kernel local-patch descriptor** *BMVC*, 2017. (citations: 12)

This publication won the Best Science Paper, Honorable mention at BMVC, 2017.

[69] Mukundan, Arun and Toliás, Giorgos and Bursuc, Andrei and Jégou, Hervé and Chum, Ondřej. **Understanding and improving kernel local descriptors** *IJCV*, 2018. (citations: 5)

[66] Mukundan, Arun and Toliás, Giorgos and Chum, Ondřej. **Explicit Spatial Encoding for Deep Local Descriptors** *CVPR*, 2019. (citations: 9)

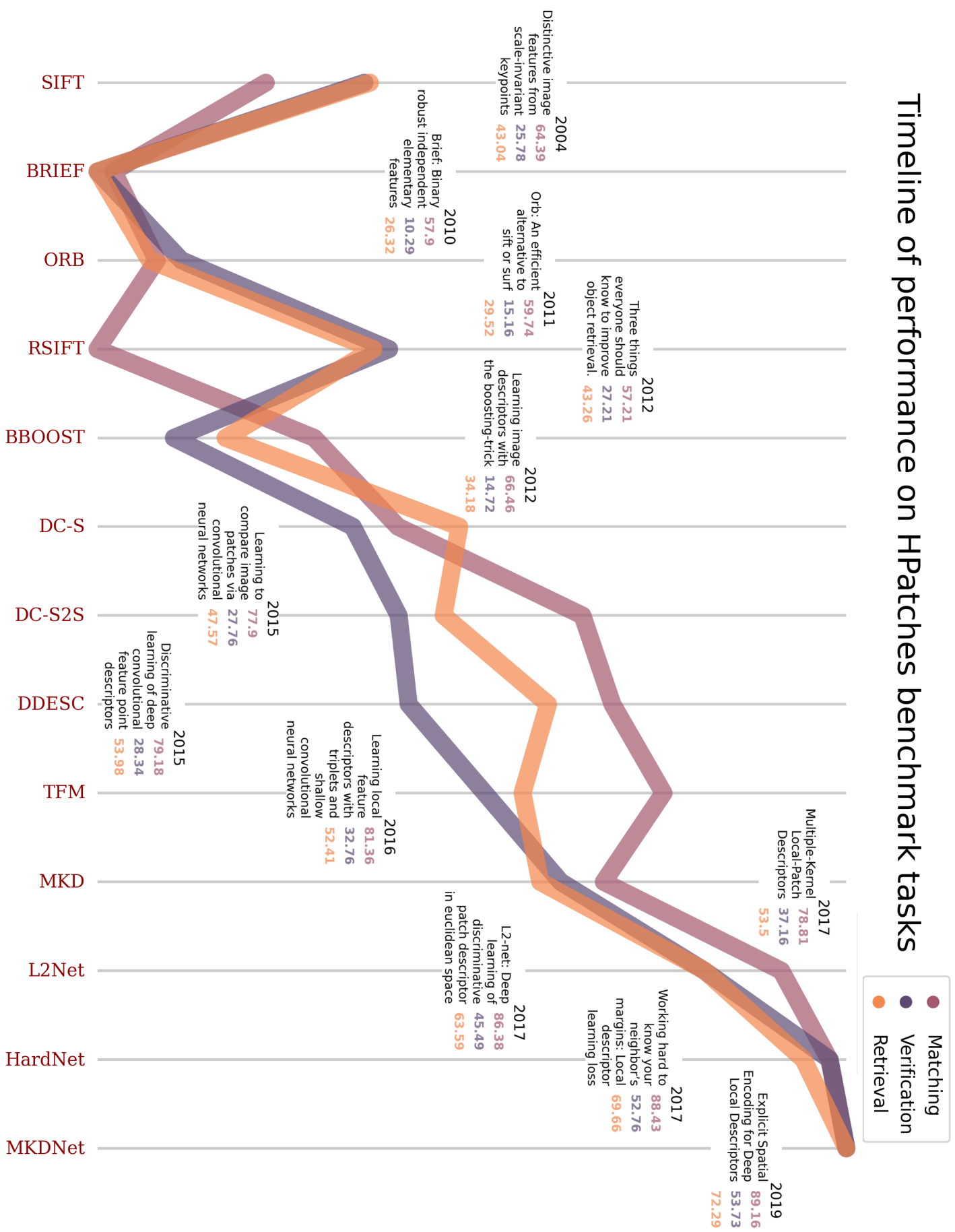
1.3 Structure of thesis

This thesis is organized as follows. Related work and state-of-the-art approaches in related areas are described in Chapter 2. Standard benchmarks and evaluation protocols are defined in Chapter 3. Multiple-kernel local descriptor is explained in Chapter 4. Our state-of-the-art deep local descriptor MKDNet is described in Chapter 5. Our approach for robust data whitening is presented in Chapter 6. Conclusions of this work are presented in Chapter 7.

1.4 Authorship

I hereby certify that the results presented in this thesis were achieved during my own research, in cooperation with my supervisor Ondřej Chum and co-supervisor Giorgos Toliás.

Timeline of performance on HPatches benchmark tasks



Related Work

In this chapter, we briefly review prior work on designing local descriptors. Descriptors often use local-patches extracted by local feature detectors as input. Many popular detectors begin by detecting local features in scale-space. These local features may be centred on corners [36, 92, 79, 80], saddle points [52, 9], blobs [54], *etc.*. These detectors also estimate some rectification parameters, such as orientation or scale, which are used to construct the rectified canonical form, *i.e.* local-patch. As noted in prior work [61], affine transformation is sufficient to locally model image distortions arising from viewpoint changes under two assumptions. The scene surface is assumed to be locally approximated by a plane and perspective effects are assumed to be small on a local scale and are ignored. Measurement regions which are covariant to affine transformations demonstrate superior performance on standard benchmarks [61, 77]. The local-patches (their local coordinate system) can be defined by the shape and scale of an ellipse and a dominant orientation [61] or explicitly constructed as in LAF [57]. Recent work using Convolutional Neural Network (CNN) based detectors demonstrates benefits from refining the geometry of measurement regions using models learned on large training sets [63]. Examples of popular detectors include ORB [80], SIFT [54], MSER [58] and HessianAffine [74], and are illustrated in Figure 2.1 and Figure 2.2.

Descriptors may also be computed without explicitly constructing a local-patch. In such cases, the descriptor must compensate for the parameters of rectification that the detector does not estimate. For example, if the detector does not estimate rotation, then the descriptor must be rotation invariant. Intensity moment invariants were introduced to construct affine and photometric invariant descriptors [100]. Spin images [42] are a general shape representation, and can be used instead of a rectified patch. They can be used to construct intensity-based rotation invariant descriptors [47], by extracting a 2D histogram that depends only on intensity and distance from the centre. In practice however, descriptors that use a rectified canonical form outperform the rest, and have been the popular approach.



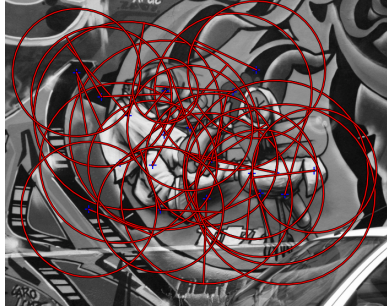




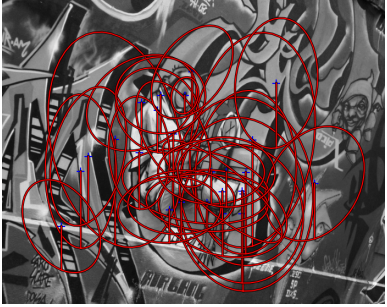
Detector	Image A	Image B
ORB		
SIFT		
MSER		
HessianAffine		

Figure 2.1: Local features detected by four different detectors are shown, namely ORB [80], SIFT [54], MSER [58] and HessianAffine [74]. Image A and Image B are images of the same scene from different viewpoints. A subset of the detections for each case is shown, for illustrative purposes, where the ellipse is representative of the shape of the local feature. While ORB and SIFT rectify using scale and angle of dominant orientation, HessianAffine and MSER use local affine frames for rectification.

In this work, we assume that the input to the description stage is the rectified canonical form called local-patch (Figure 2.2). This allows for comparison of corresponding pixel neighborhoods, as the coordinate system established by the local-patch ensures such measurements are valid. Measurements that capture the appearance of these neighborhoods may be specified explicitly, as in the case of hand-crafted descriptors, or learned implicitly, as in the case of deep local descriptors. These measurements are pooled into a compact vector in a manner that retains spatial information, *e.g.* concatenation, spatial embedding, *etc.*. A post-processing step is common to both hand-crafted and learned descriptors. This post-processing ranges from simple PCA dimensionality reduction, to transformations learned on annotated data. Finally, the resulting vector is ℓ_2 -normalized, allowing for comparison using inner product. This procedure for hand-crafted descriptors is shown in Figure 2.3 and for deep local descriptors in Figure 2.4. In the following sections, we review prior literature on hand-crafted and deep local descriptors.


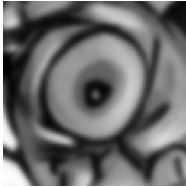


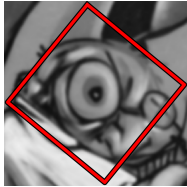





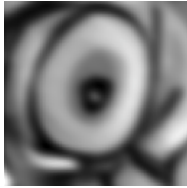

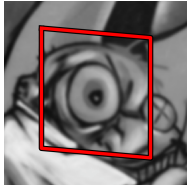

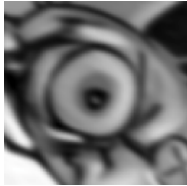

Detector	Image A	Local patch \mathcal{P}	Local patch \mathcal{Q}	Image B
ORB				
SIFT				
HessianAffine				
MSER				

Figure 2.2: Local-patches obtained by rectification of image regions, from local features detected by four different detectors are shown, namely ORB [80], SIFT [54], MSER [58] and HessianAffine [74]. Image A and Image B are images of the same scene from different viewpoints. For illustrative purposes, a specific local feature is chosen, close to the eye of the man in the graffiti. The red boxes in the images delimit the region of the image that forms the local-patch.

2.1 Hand-crafted descriptors

Hand-crafted descriptors have dominated the research landscape and a variety of approaches and methodologies exists. A common, well performing approach is to compute compact representations that capture local pixel neighborhoods and aggregate them while being invariant to desired photometric or geometric transformations [54, 25, 15]. This approach allows for injection of domain expertise and prior knowledge by explicitly specifying the measurements to be considered. Proposed descriptors capture image properties like pixel intensities, color, texture, edges and so forth. Some methods do not use any training data [82, 54], while other methods use labeled or unlabeled data to learn the parts of the model [44, 21, 88].

A generic pipeline for extracting a hand-crafted descriptor can be described as follows: the input is a rectified canonical local-patch, which is typically preprocessed, *e.g.* gray value normalization in view of tolerance to changes in illumination [23], smoothing in view of robust feature maps [44] and so forth. Features are then extracted from pixel neighborhoods of the preprocessed map, such as responses from a filter bank [54], outcomes of binary tests [25] and others. The features are pooled, usually preserving the spatial information using methods such as concatenation [54], spatial embedding [23]. The pooled descriptors are post-processed, commonly by whitening [23] or dimensionality reduction [44], [54], and finally ℓ_2 -normalized. The resulting vector is the local descriptor. This pipeline is illustrated in Figure 2.3. We follow a nomenclature introduced in literature [60] to categorize descriptors into *distribution-based descriptors*, *differential descriptors*, *spatial-frequency methods*, *binary descriptors*, and descriptors based on *local binary patterns*. Additionally, we briefly introduce *kernel descriptors* as they are relevant to our work.

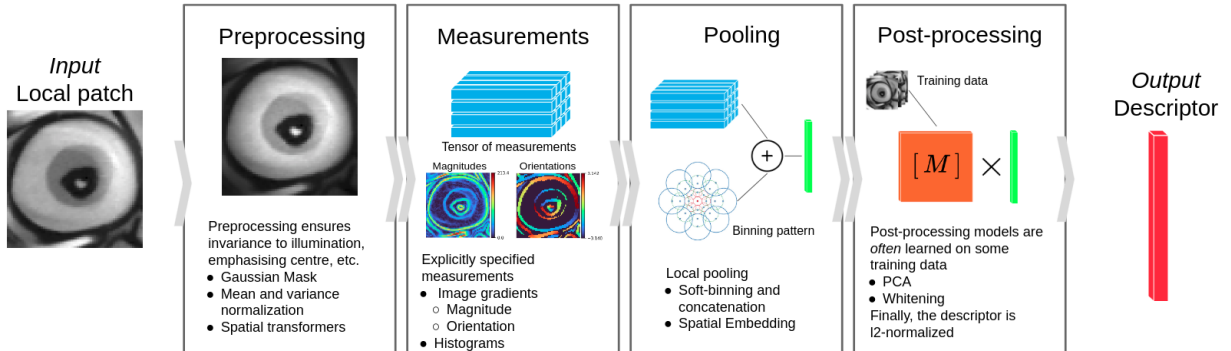


Figure 2.3: A generic pipeline for hand-crafted local descriptor extraction is shown. For each stage, an example is shown and some other approaches are listed. The patch undergoes *preprocessing*, followed by extraction of *measurements* describing local pixel neighbourhoods. Hand-crafted approaches explicitly specify the measurements. These measurements are *pooled*. In some cases, the descriptors are *post-processed*, using models trained on some data. Finally to allow for comparison by inner product, the descriptor is usually ℓ_2 -normalized.

Distribution-based descriptors. Local descriptors based on histograms of features have been studied extensively [54, 15, 22, 46, 82]. Distribution-based descriptors are dominated by the most widely known local descriptors are SIFT [54] and its variants. SIFT has consistently performed well across many tasks as measured by its performance on various benchmarks and tasks including image retrieval [77], stereo matching [60], 3D reconstruction [83], *etc.* and its variants are competitive even on recent benchmarks [41]. The design of SIFT was inspired by an observation about biological vision. It was known that neurons in the human eye respond to gradients at particular orientations and spatial frequencies. Early attempts used these responses as measurements on the local-patch. However, if the location of the gradient on the retina is allowed to shift over a small receptive field rather than being precisely localized, the performance on recognition accuracy improves tremendously [30]. The SIFT descriptor uses this observation by accumulating image gradients into histograms corresponding to several spatial bins and orientations. This allows for the shift of the gradients within the spatial and orientation bins as described above, making the descriptor tolerant to inaccurate detections and minor deformations. It also emphasizes the centre of the patch by weighting the gradients with a Gaussian mask. The final descriptor is computed by concatenating the histograms, and clipping them. Similarity of two given patches is computed as the Euclidean distance between their respective descriptors. It can be shown that the resulting score is a comparison of the gradient at each position in one patch with the gradient at every position in the other patch, *i.e.* as a *match kernel* [19].

Many improvements to SIFT have been proposed, including adapting the feature space to the data distribution using Principal Component Analysis (PCA) [44], improving the similarity measure [11] or countering detection errors that result in misaligned canonical patches [23]. PCA-SIFT [44] proposes to simply use the flattened gradients, which are reduced to their principal components, learned in an unsupervised manner on a diverse collection of patches. A simple and effective improvement of SIFT is brought by RootSIFT [11], which notes that Hellinger measures outperform Euclidean distances when comparing histograms. It is computed by l_1 -normalizing, element-wise square-rooting, and finally l_2 -normalizing the SIFT descriptor. RootSIFT is frequently used as a reference baseline for measuring performance on many tasks including patch verification [13], 3D reconstruction [83] and image retrieval [77].

Some approaches suggested improvements to schemes used for spatial pooling. SIFT uses a Cartesian grid to define spatial bins over which to pool gradients. DAISY [94] proposes radial pooling bins to account for tolerance to rotational misalignments caused by errors in the detection of the local feature. GLOH [60] proposes a log-polar binning pattern, which emphasizes the centre of the patch, and also reduces the dimensionality of the descriptor by using Principal Component Analysis (PCA). Among recent work, DSP-SIFT [29] counters the aliasing effects caused by the binned quantization in SIFT by pooling gradients over multiple scales.

Differential descriptors. A pixel neighborhood can also be described by a set of derivatives upto a given order. The properties of these derivatives, also known as *local*

jets are well studied [45, 33], and were used to describe local pixel neighborhoods. Steerable filters [34] are an efficient method of synthesizing filters of arbitrary orientations from linear combinations of basis filters. Steering derivatives in the direction of the gradient makes them invariant to rotation.

Spatial-frequency methods. Some descriptors are designed to use features from filter-bank responses other than image gradients. Typically, the local-patch is convolved with a set of filters and the responses are pooled and normalized. SURF [15] uses wavelet responses, and relies on integral images for image convolutions, thus significantly reducing computation time. Patches can also be described using wavelet transforms [22], phase, orientation and amplitude features [46], and local gray value invariants, obtained by convolution with Gaussian derivatives [82].

Binary descriptors. One approach to improve efficiency of extraction and storage of local descriptors is to compute binary representations. CARD [10] improves the computational efficiency by using look-up tables to calculate histograms based on the assumption that the principal gradients can be quantized, regardless of the binning pattern. Improvements to efficiency of storage and computation were also introduced by the use of binary descriptors. BRIEF [25] uses randomly defined pairwise intensity comparisons as binary tests, and computes a binary vector of test responses. BRISK [50] improves on BRIEF by first estimating an orientation, and proposed a hand-crafted sampling pattern for the binary tests. ORB [80] was a descriptor proposed by a popular open source organization, that chose to offer the method free of licensing restrictions that SIFT and SURF had imposed. It improves on BRIEF by estimating rotation by ‘steering’ the responses of the binary tests, and decorrelating the resulting responses by training over the PASCAL [31] dataset.

Local binary patterns. Local binary patterns (LBP) are illumination invariant textural primitives that are invariant to shifts in gray value [72, 39]. They are tolerant to illumination changes and can be computed very efficiently. A histogram of the binary patterns computed over a region is used as the feature descriptor. The operator describes each pixel by the relative gray levels of its neighboring pixels. LBP-based approaches perform especially well for classification of textures [71]. Another successful approach focuses on capturing invariants based on pixel attributes like gray value intensity [82]. Some approaches [86] note that images of the same object may not share image properties (colors, textures, edges), and proposes using self-similarity for describing local regions.

Kernel descriptors. Kernel descriptors based on the idea of Efficient Match Kernels (EMK) [19] encode entities inside a patch (such as gradient, color) in a continuous domain, rather than as a histogram and form a flexible way to design descriptors with the desired invariant properties. The kernels and their few parameters are often hand-picked and tuned on a validation set. Kernel descriptors are commonly represented by a finite-dimensional explicit feature maps [101]. Quantized descriptors, such as SIFT, can be also interpreted as

kernel descriptor [23, 18]. Furthermore, the widely used RootSIFT descriptor [11] can be also thought of as an explicit feature map from the original SIFT space to the RootSIFT space, such that the Hellinger kernel is *linearised*, *i.e.* the linear kernel (dot product) in RootSIFT space is equivalent to the Hellinger kernel in the original SIFT space. In this case, the feature mapping is performed by ℓ_1 -normalization and square-rooting, without any expansion in dimensionality. Kernel descriptors have been proposed not only for local patches [23] but also as a global image descriptor [18]. In Chapter 4 we build upon EMK by integrating multiple pixel attributes in the patch descriptor. Unlike EMK which relies on features from random projections that require subsequent learning, we leverage instead explicit feature maps to approximate a kernel behavior directly. These representations can be further improved by minimal learning.

Hand-crafted approaches are still in use today in commercial applications despite the superior performance of deep local descriptors. They offer the advantages of interpretability, efficient computation, semantically useful hyperparameters for tuning, and simple post-processing methods to adapt the descriptors to the distribution of data. Implementations are available through popular and free software [102, 20], and generally do not need special hardware to run tractably. Further, recent benchmarks surprisingly show that they are indeed competitive with the latest deep local methods on tasks such as image matching when the matching pipeline is properly tuned [41].

2.2 Deep local descriptors

While hand-crafted features are designed by explicitly defining the measurements to be used, modern approaches instead learn the features in an end to end manner, by incorporating convolutional neural networks (CNNs) that are trained on large annotated datasets. The proposed architectures mimic those that proved successful on full-image tasks such as classification. They typically consist of a fully convolutional network (FCN) connected to a fully connected (FC) layer [106, 14, 93, 62]. The FCN extracts a feature map consisting of features corresponding to pixel neighbourhoods arranged on a regular Cartesian grid. The FC layer transforms the features based on their position on the grid and aggregates them into a single vector. This vector is ℓ_2 -normalized to obtain the final descriptor. The procedure is depicted in Figure 2.4.

Earliest attempts [32] directly used convolutional features trained on ImageNet [28]. These features were compared with traditional descriptors like SIFT, and showed impressive results, though they were tuned on the task of image classification instead. To optimize the feature descriptor directly for establishing matching pairs, training data with patch-level annotation is required. These are in the form of pairs of corresponding and non-corresponding patches, which are obtained from Structure-from-Motion (SfM) pipeline or through synthetic transformations. Some approaches learned both the features and the similarity measure [106]. Later works [14, 93] used features that could be compared using Euclidean distances, thus allowing the use of traditional pipelines. The architectures

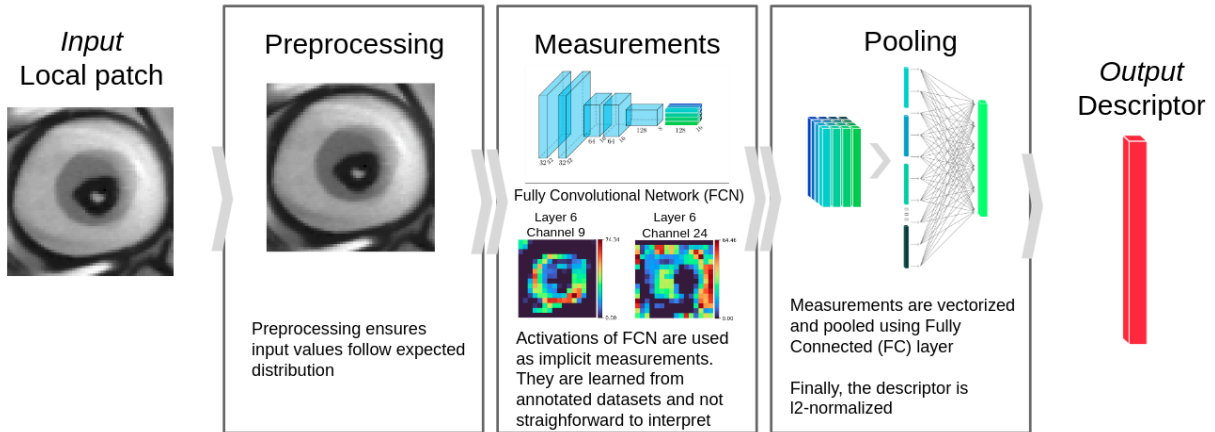


Figure 2.4: A generic pipeline for extraction of deep local descriptors extraction is shown. For each stage, an illustration is presented and some important points are listed. Deep local descriptors are based on Convolutional Neural Networks (CNNs). The patch undergoes *preprocessing* to ensure input values stay in a preferable range. Measurements are implicitly specified by activations of a Fully Convolutional Network (FCN). This corresponds to making measurements on overlapping receptive fields placed on a Cartesian grid. The activations are vectorized and pooled using Fully Connected (FC) layer. Finally to allow for comparison by inner product, the descriptor is usually ℓ_2 -normalized.

proposed vary in the depth of the network [106, 87], the non-linearity layer [14] and pooling layer [66].

Recent work in deep local descriptors use shallow networks with carefully designed training strategies and loss functions. The networks typically use either pairs or triplets of patches as input. Pairs are labeled as positives if they are corresponding patches and negatives if they are non-corresponding patches. Triplets consist of an anchor and a positive and negative example. The losses are designed such that negative pairs are pushed apart such that the distance between them is increased, and the positive pairs are pulled closer together such that their distances are decreased. It is observed that sampling training examples, especially *hard negatives* is critical to good performance. Hard negatives which refers to examples of non-corresponding pairs of patches whose descriptors were similar, *i.e.* the network is likely to incorrectly classify them as positives. Recently, the larger focus is on crafting better loss functions [93, 62, 56] and using better datasets [56, 65]. We describe some of the approaches below.

DeepDESC [87] uses a hinge embedding loss, which penalizes the positive pairs by their Euclidean distance, and the negative pairs by their distance from a margin, ensuring that the loss is positive. The margin determines the ideal distance to every negative sample. Additionally, descriptors are extracted from a randomly chosen set of pairs, and only the ‘hardest’ subset is used to train the network. D2C [106] uses a hinge embedding loss to train both the feature descriptor and a metric layer to compare a pair of patches.

While the metric layer improves performance, the drawback is that the descriptors cannot be used traditional pipelines that assume Euclidean distance as the similarity measure. They augment the training samples by flipping both patches in pairs horizontally and vertically and rotating by 90, 180, 270 degrees. They do not mention using any form of hard negative mining. It is also noted that a multi-scale approach improves performance. A novel 2-channel architecture that effectively describes the patch at two scales, which they call centre and surround, is introduced. This approach shows good performance but comes at the cost of increased computation. TFeat [14] improves on the hinge loss by introducing triplet-based losses that take advantage of in-triplet hard negative mining. A training example is a triplet of patches, consisting of an anchor, a positive and a negative patch. The triplet loss is based on the distances between the positive and negative pairs and a predefined margin. They note that losses are categorized as ranking-based losses and ratio-based losses. Further, an anchor-swap strategy is introduced that defines the negative pair as either query and negative example or positive example and negative example based on whichever yields the hardest negative in the triplet. They provide a thorough comparison of using such losses by measuring performance on standard benchmarks.

L2-Net [93], proposes a loss function composed of multiple terms, and incorporates supervision on intermediate feature maps. Further, they propose a new training regime that makes use of all pairs of patches in a batch. HardNet [62] improves on the training strategy of L2-Net. They discard the use of auxiliary losses and instead define a single loss that maximizes the distance between the farthest positive and closest negative example in the batch. HardNet performs well on multiple benchmarks and tasks [7, 13]. In Chapter 5, we improve on HardNet by interpreting them as efficient match kernels and design a novel local descriptor that explicitly encodes the spatial information, which performs on par with state-of-the-art descriptors with fewer parameters and outperforms on standard benchmarks with the same number of parameters.

DOAP [38] observes that as the descriptors are finally used for matching, it is advantageous to use a learning-to-rank formulation that optimizes local feature descriptors for nearest neighbor matching. They argue that average precision evaluates the performance of retrieval systems under the binary relevance assumption: retrieved results are either “relevant” or “irrelevant” to the query, and formulate a loss that incorporates this measure. Further, they compensate for errors in detection by using spatial transformers to predict and correct geometric noise.

GeoDesc [56] proposes to leverage geometric information from the SfM framework to improve the quality of supervision, though from the results presented, the improvement in performance seems to stem from the introduction of a larger, curated dataset. Similarly, in PhotoSync [65], a sampling technique for generating matching correspondences is used, which proposes to ensure that the training dataset has sufficient variations in viewpoint and scale. Such sampling strategies demonstrate considerable improvement in performance on standard tasks [7].

Deep local approaches leverage the availability of large sets of annotated examples to optimize their parameters. They demonstrate irrefutably superior performance on standard benchmarks. There are many free implementations of frameworks that can be used to

reproduce these methods. They benefit vastly from hardware acceleration. Further, authors usually provide trained models, as the training is usually expensive. However, adoption in real world applications is not as widespread despite the superior performance. This may be due to the difficulty in tuning models, which usually requires many examples to correct aberrant behaviour, or the difference in data distribution between training and real world regimes.

In the next chapter, we describe standard benchmarks and evaluation protocols, based upon which progress in the state of the art is defined.

Datasets

Evaluation of local descriptors requires a sufficiently large dataset that captures variation in appearance, viewpoint, *etc.* that are encountered in practice. Datasets commonly used in literature include the PhotoTourism dataset [103], HPatches dataset [13], Rome patches dataset [73, 51], Generated Matching dataset [32], Oxford matching dataset [60] and WxBs dataset [64]. Datasets may include annotation at patch-level, *i.e.* each patch has an associated label, and patches having the same label are in correspondence, or at image-level *i.e.* images depicting the same scene are known but the corresponding regions are unknown. Datasets that include patch-level annotation [103, 13] usually evaluate descriptors on the task of patch correspondence, while those that include image-level annotation [64, 60], evaluate on the task of image matching or retrieval. Datasets that include patch-level annotation are used not only to evaluate, but also to train descriptors that learn parameters of their model. Such datasets are derived from either Structure-from-Motion (SfM) methods, or by the use of synthetic transformations. Datasets that use SfM pipelines typically use images that are collected from a large public source with weak annotation, *i.e.* returned images may not be relevant. Local features are extracted from these images using detectors, and the extracted features are described using local descriptors. Given the descriptors and the geometry of the features, pairs of images that are in correspondence are identified. The transformations are calculated, from a reference image, to all matching images. Finally, bundle adjustment is performed, and we obtain a set of 3-dimensional points, called a point cloud, and the poses of the camera for each image. For each such point, local-patches are extracted from associated images, and the label and patch is stored. This gives us the correspondences we need for supervision used for training and evaluation local descriptors. Freely available software [84, 85] for this pipeline has facilitated augmenting supervision with the information from the pipeline. Apart from selecting a better training set [65], learning methods [56] may even integrate this information into the loss while training. Early datasets [103] propose use Receiver Operating Characteristic (ROC) curve to measure performance. Modern datasets [13] suggest using Precision-Recall and mean Average Precision (mAP) instead to account for the imbalanced nature of some tasks, where the negatives far outnumber the positives.

These datasets also propose tasks that are proxies for matching and retrieval.

In our evaluations, we use the PhotoTourism dataset and the HPatches dataset, which are described in further detail in Section 3.1 and Section 3.2, and we briefly review other popular datasets in the Section 3.3.

3.1 PhotoTourism

The PhotoTourism dataset was introduced by Winder *et al.* [103] in 2007. They argue that there was an absence of a systematic exploration of the algorithms proposed to design local descriptors. The prevailing performance evaluation, the Oxford matching dataset [60] used natural images which were nearly planar and used camera motions which could be approximated as homographies. Ground truth homographies were then obtained semi-automatically, thus 3D effects were not adequately represented. To account for non-planar effects around interest points, illumination changes and distortions typical of real life scenarios, the PhotoTourism dataset uses correspondences from reconstructed 3D scenes. This introduces 3D appearance variation around each interest point as it is captured in multiple images of a 3D scene where the camera matrices and 3D point correspondences can be accurately recovered. However, this does not guarantee that patches depicting different 3D points do not contain the same object, as shown in figure 3.1. In the following sections, we describe the collection of the dataset and the evaluation metrics used.

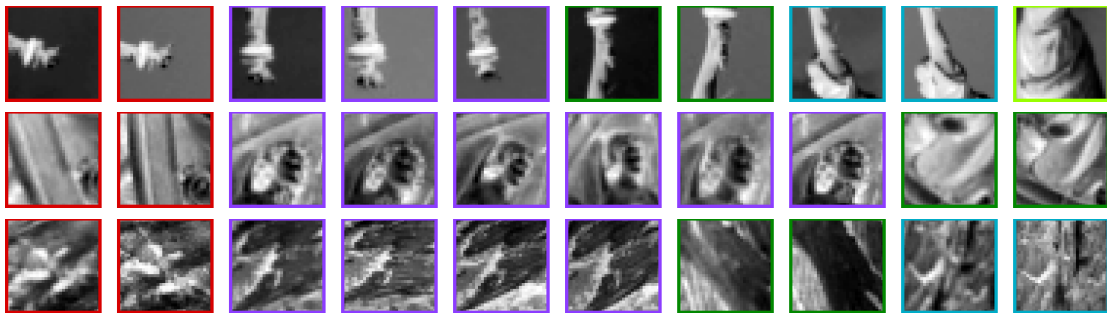


Figure 3.1: Samples of patches from the three PhotoTourism sets, Liberty (top), Notre-Dame (middle), and Yosemite (bottom). The color of the box indicates the 3D point label associated with the patch. It can be observed, from the first examples of the Liberty dataset, that different labels may depict the same object, *i.e.* the torch of the statue of Liberty.

Collection of dataset. The data is taken from SfM reconstructions from Trevi Fountain (Rome), Notre Dame (Paris) and Half Dome (Yosemite). A large set of images are captured from each scene. The 3D point cloud representing the scene is reconstructed using the SfM pipeline. Local features were extracted using the SIFT detector. Local-patches were extracted corresponding to these features, and described using the SIFT descriptor. The descriptors were matched across all images using a symmetry criterion, and robust

estimation of the fundamental matrices using RANSAC. Further, bundle adjustment was applied to reconstruct 3D points. Sets of corresponding local features across different images can be constructed, as each 3D point is associated with multiple local features. Train and test ground truth are obtained by randomly choosing corresponding and non-corresponding pairs from the extracted local-patches.

Evaluation protocol. The test set of the evaluation protocol consists of local-patch pairs from one of the three sets, Liberty, NotreDame and Yosemite, and a label indicating correspondence. Casting the evaluation in the framework of HPatches [13], supervision is defined by a list $T = ((\mathcal{P}_i, \mathcal{Q}_i, y_i), i = 1, \dots, N)$, where $\mathcal{P}_i, \mathcal{Q}_i \in \mathbb{R}^{64 \times 64}$ are a pair of patches and labels $y_i \in \{0, 1\}$ indicate if the pair is in correspondence or not. Normalized local patch descriptors, $\hat{\mathbf{V}}_{\mathcal{P}}$ and $\hat{\mathbf{V}}_{\mathcal{Q}}$, are extracted from the patches. The similarity between each pair is obtained as a confidence score by simple inner product, $s_i(\mathcal{P}_i, \mathcal{Q}_i) = \hat{\mathbf{V}}_{\mathcal{P}_i}^\top \hat{\mathbf{V}}_{\mathcal{Q}_i}$. The scores are sorted by permutation π such that the scores are in decreasing order, *i.e.* $s_{\pi_1} > s_{\pi_2} > \dots > s_{\pi_n}$. The true positive rate is calculated as correctly detected matches as a fraction of all true matches, and the false positive rate as incorrectly detected matches as a fraction of all true non-matches.

$$\text{tpr}_i = \frac{\sum_{k=1}^i y_{\pi_k}}{\sum_{k=1}^N y_{\pi_k}} \quad (3.1)$$

$$\text{fpr}_i = \frac{\sum_{k=1}^i 1 - y_{\pi_k}}{\sum_{k=1}^N 1 - y_{\pi_k}} \quad (3.2)$$

Performance is measured in terms of the percentage of false matches present when 95% of all correct matches are detected. This summary value is defined as the False Positive Rate at 95% of true positive rate (FPR95), *i.e.*

$$\text{FPR95} = \text{fpr}_I, \quad (3.3)$$

where I is the least index for which $\text{tpr}_I > .95$. In the case of learned descriptors, the protocol is to train on one of the three sets and test on the other two. The average over all six combinations is reported.

3.2 HPatches

The HPatches dataset was introduced by Balntas, et al. [13] in 2017, ten years after the introduction of PhotoTourism. HPatches dataset is a large dataset suitable for training and testing modern descriptors, together with strictly defined evaluation protocols in several tasks consisting of matching, retrieval and classification. They observe that results of the time that compared descriptors were inconsistent due to ambiguously defined evaluation protocols. Specifically, they attribute variations in different descriptor evaluations to the

fact that there is no predefined set of regions to match. As a consequence, results depend strongly on the choice of detector (method, implementation, parameters, measurement regions), making the comparison of descriptors unreliable. Further, as the measurement is on an imbalanced dataset (many more negatives than positives), they reject the use of the ROC curve, and instead suggest the Precision-Recall and mean Average Precision as a better choice for metrics. In the following sections, we describe the collection of the dataset and the evaluation metrics used.

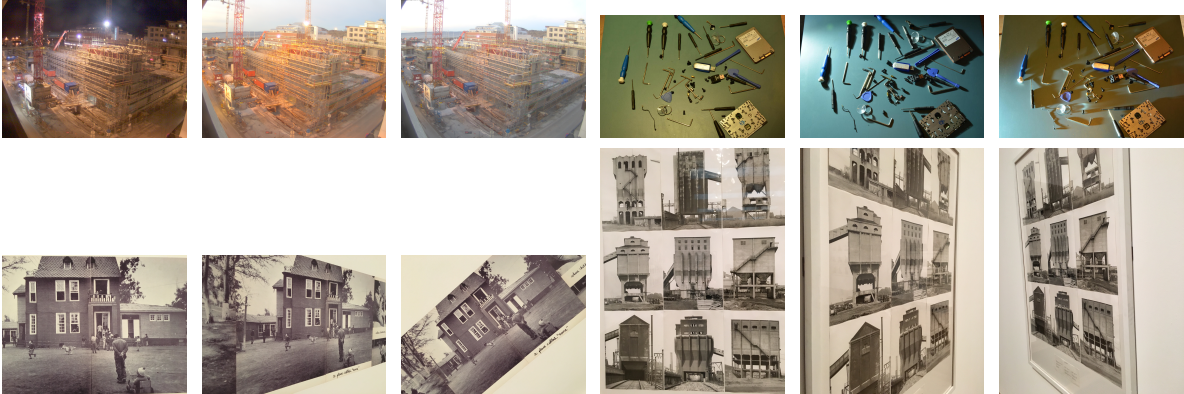


Figure 3.2: Samples of image sequences from HPatches dataset from the sequences ‘i_construction’, ‘i_tools’, ‘v_home’ and ‘v_machines’, respectively showing illumination and viewpoint changes. It is noted that many viewpoint changes show images of planar scenes, while illumination changes do not include any change in viewpoint.

Collection of dataset. Scenes are selected to be representative of different use cases and captured under varying viewpoint, illumination, or temporal conditions, including challenging nuisance factors often encountered in applications. A total of 116 image sequences are collected, with 57 scenes where the main nuisance factors are photometric changes and the remaining 59 sequences having significant geometric deformations due to viewpoint change.

A sequence includes a reference image and 5 target images with varying photometric or geometric changes. The sequences are captured such that the geometric transformations between images can be well approximated by estimating a homography from the reference image to each of the target images. Local features are extracted from the reference image, and target images for each sequence. Scale invariant interest point detectors, *i.e.* DoG, Hessian-Hessian and Harris-Laplace are used to detect features for scales larger than a certain threshold. Near-duplicate regions are discarded based on their intersection-over-union (IoU) overlap and one region per cluster is randomly retained.

A subset of approximately 1,300 regions per image are then randomly chosen. For each sequence, patches are detected in the reference image and projected on the target images using the estimated ground-truth homographies. In order to provide a deeper insight into the effects of synthetic noise, noise in detection is simulated by perturbing the features in 3 stages, EASY, HARD and TOUGH. In each region the dominant orientation angle is

estimated using a histogram of gradient orientations. Rectified canonical local-patches are extracted by normalizing the detected affine region to a circle using bilinear interpolation and extracting a square of 65×65 pixels that circumscribes the circle. Examples of the image sequences and the extracted local-patches are shown in Figures 3.2 and 3.3.

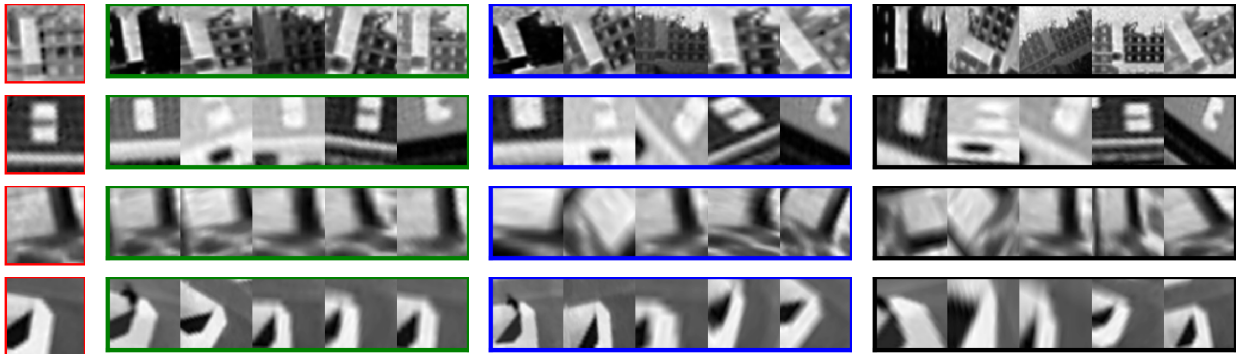


Figure 3.3: Samples of patches from HPatches dataset from the sequences ‘i_castle’, ‘i_tools’, ‘v_home’ and ‘v_machines’, respectively showing illumination and viewpoint changes. Reference patch is shown in red, EASY in green, HARD in blue, TOUGH in black.

Evaluation protocol and metrics. Three tasks, *patch verification*, *image matching*, and *patch retrieval* are defined. The tasks are designed to imitate typical use cases of local descriptors. Patch verification measures the ability of a descriptor to classify whether two patches are coming approximately from the same surface of a 3D scene. Image matching tests to what extent a descriptor can correctly identify correspondences in two images. Finally, patch retrieval tests how well a descriptor can match a query patch to a pool of patches extracted from many images, including many distractors, by returning a ranked list. This is a proxy to local feature based image retrieval.

We describe the protocol that establishes the precision and recall evaluation metric used in HPatches. Given a query patch, we compute a ranked list of patches retrieved based on the similarity score. Let $y = (y_1, \dots, y_N) \in \{-1, 0, +1\}^N$ be labels of this list. The labels indicate if the patch is in correspondence (positive, +1), is not in correspondence (negative, -1) or if the patch is to be ignored (0).

Precision (P_i) and recall (R_i) at rank i are given by ¹

¹Here, $[x]_+ = \max(0, x)$.

$$P_i(y) = \sum_{k=1}^i [y_k]_+ / \sum_{k=1}^i |y_k| \quad , \quad (3.4)$$

$$R_i(y) = \sum_{k=1}^i [y_k]_+ / \sum_{k=1}^N [y_k]_+ \quad . \quad (3.5)$$

Average precision (AP) is defined as

$$AP(y) = \sum_{k:y_k=+1} P_k(y) / \sum_{k=1}^N [y_k]_+ \quad . \quad (3.6)$$

It generalizes over standard definition of PR by allowing entries to be ignored. The protocols for each of the three tasks *patch verification*, *image matching*, *patch retrieval* are defined and are described below.

Patch verification. This task seeks to measure the discriminativeness of descriptors. The task is to determine whether a pair of patches is in correspondence or not. This task is similar to the evaluation scheme used by PhotoTourism [103]. Ground-truth consists of a list of positive and negative patch pairs. The test set can be represented as $T = ((\mathcal{P}_i, \mathcal{Q}'_i, y_i), i = 1, \dots, N)$ where $\mathcal{P}_i, \mathcal{Q}'_i \in \mathbb{R}^{65 \times 65}$ are patches and $y_i = \pm 1$ determines if they are in correspondence or not. Local descriptors are extracted from these patches, and the similarity s_i between a pair of patches $\mathcal{P}_i, \mathcal{Q}_i$ is defined as the inner product of corresponding ℓ_2 -normalized descriptors, $s_i = \hat{\mathbf{V}}_{\mathcal{P}}^T \hat{\mathbf{V}}_{\mathcal{Q}}$. The scores are sorted in decreasing order, $(s_{\pi_1} \geq s_{\pi_2} \geq \dots \geq s_{\pi_n})$ by the permutation π . The performance of the descriptor is measured by the average precision of the ranked patches, *i.e.* $AP(y_{\pi_1}, \dots, y_{\pi_N})$ as defined in equation 3.6. The test set is further divided based on the level of synthetic perturbation into EASY, HARD and TOUGH, and based on whether negatives are taken from the same scene or from different scenes (INTRA, INTER). Each set consists 2×10^5 positives and 10^6 negatives. The summary score is the mean average precision over the six sets. They claim that the imbalanced nature of the test set makes AP a better metric than FPR95.

Image matching. This task is a proxy to the task of matching images using local features. In this task, an image I is defined as a collection of N patches, *i.e.* $I = \{\mathcal{P}_i, i = 1, \dots, N\}$. Given a pair of images $T = (I_0, I_1)$ from the same sequence, the patches $\mathcal{P}_i \in I_0$ and $\mathcal{Q}_i \in I_1$ must be in correspondence. Therefore, given descriptor $\mathbf{V}_{\mathcal{P}}$ from image I_0 , the closest descriptor from image I_1 must be $\mathbf{V}_{\mathcal{Q}}$. The evaluation consists of assigning a label $y_i = 2[\sigma_i == i] - 1$ for each $\mathcal{P}_i \in I_0$, and the corresponding score s_{σ_i} where the index $\sigma_i \in 1, \dots, N$ indicates the best match $\mathcal{Q}_{\sigma_i} \in I_1$. As before, the scores are sorted in decreasing order, $(s_{\pi_1} \geq s_{\pi_2} \geq \dots \geq s_{\pi_n})$ by the permutation π . The performance of the descriptor is measured by the average precision of the ranked patches, *i.e.* $AP(y_{\pi_1}, \dots, y_{\pi_N})$ as defined in equation 3.6. The test set is further divided based on variation by viewpoint

or illumination, and based on the level of synthetic perturbation EASY, HARD and TOUGH. The summary score is the mean average precision over the six sets.

Patch retrieval. This task imitates the image retrieval, except retrieving patches instead of images. The task is to find an ordered list of ‘relevant’ patches given a query patch \mathcal{P}_0 . The ground-truth is in the form of a collection $T = (P_0, (\mathcal{P}_i, y_i), i = 1, \dots, N)$, which consists of labels y_i and patches \mathcal{P}_i . Assuming the query patch \mathcal{P}_0 is taken from image I_0 , all patches $\mathcal{P}_k \in I_0$ are assigned a label $y_k = 0$. This is to account for patches extracted from such ‘*innocuous errors*’ as repeated structures, which are not considered detrimental to the retrieval of a correct image, as they originate in the same image. Patches $\mathcal{P}_{k'} \in I_0$ are labelled such that $y_i = +1$ if \mathcal{P}_i and \mathcal{P}_0 are in correspondence (positives), and $y_i = -1$ if they are not in correspondence (negatives). As in the previous task, given a query patch \mathcal{P}_0 , we compute a ranked list of patches \mathcal{P}_i retrieved based on the similarity score s_i . The scores are sorted in decreasing order, ($s_{\pi_1} \geq s_{\pi_2} \geq \dots \geq s_{\pi_n}$) by the permutation π . The performance is given by $AP(y_{\pi_1}, \dots, y_{\pi_N})$ as defined in equation 3.6, which takes advantage of the provision to ignore entries. The test set consists of 1×10^4 queries and their corresponding patches, and 2×10^4 distractors randomly selected from the entire dataset. It is further divided into EASY, HARD and TOUGH sets based on the level of synthetic perturbation. The summary score is the mean average precision over all queries in all three sets.

3.3 Others

Oxford matching dataset The Oxford matching set [60] was introduced in 2005, to evaluate various local descriptors. They concentrate on specific geometric and photometric transformations of six kinds: rotation, scale change, viewpoint change, image blur, JPEG compression and illumination. For certain transformations, they also use different scene types. One scene type contains structured scenes, that is homogeneous regions with distinctive edge boundaries and the other contains repeated textures of different forms. Figure 3.4 shows examples of scenes from the Oxford matching dataset.

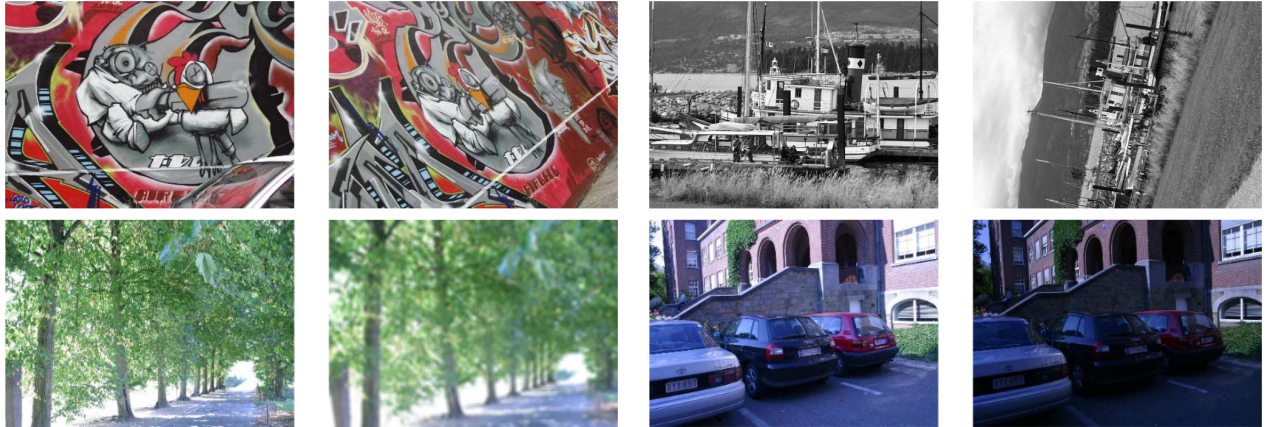


Figure 3.4: Samples of image sequences from Oxford matching dataset.

Evaluation criterion is based on recall. Two regions are matched if the distance between their descriptors is below a threshold t . Each descriptor from the reference image is compared with each descriptor from the transformed one and the number of correct and false matches are noted. The value of t is varied to obtain the curves. The results are presented with recall versus 1-precision. Recall is the number of correctly matched regions with respect to the number of corresponding regions between two images of the same scene, *i.e.* $\text{recall} = \frac{\#\text{correct matches}}{\#\text{correspondences}}$. The number of correct matches and correspondences is determined with the overlap error. Typically, there are very few regions with larger error that are correctly matched and these matches are not used to compute the recall. The number of correspondences (possible correct matches) are determined with the same criterion. The number of false matches relative to the total number of matches is represented by 1-precision. Given recall, 1-precision and the number of corresponding regions, the number of correct matches is $\#\text{correspondences} \cdot \text{recall}$ and the number of false matches by $\#\text{correspondences} \cdot \text{recall} \cdot (1 - \text{precision}) / \text{precision}$.

WxBS dataset WxBS [64] also follows a similar approach, by introducing sequences that capture specific transformations, however, instead of only a single nuisance factor per image pair, they capture complex environments that consist a combination of nuisance factors. For the curious, the ‘x’ in ‘WxBS’ stands for the variety in acquisition conditions of ‘wide baselines’ extending to geometry, illumination, sensor and appearance. They show that state-of-the-art matchers fail on almost all image pairs from the set. Figure 3.5 shows examples of scenes from the WxBS dataset.



Figure 3.5: Samples of image sequences from WxBS dataset.

Evaluation criterion is based on recall. For each image pair, ground-truth is in form of a manually annotated set of correspondences. Recall is computed as a function of a threshold t , as a ratio of correspondences that satisfy a geometry model within the threshold t to the total number of correspondences. Finally, for all pairs of each category, an overall recall per category is defined.

Custom datasets Structure from Motion (SfM) methods have recently utilized large image sets in the form of unordered Internet collections of popular landmarks to reconstruct these landmarks in 3 dimensions, represented as point clouds. Of various methods, incremental SfM is a sequential pipeline, which performs the reconstruction in an iterative manner. COLMAP [84, 85] is an open-source software that implements this pipeline. Specifically, given a large set of images, local features are extracted, followed by matching and geometric verification. A two-view reconstruction is used as a seed and new images are registered incrementally. The pipeline triangulates scene points, filters outliers, and refines the model using bundle adjustment. The final model consists of a 3D point cloud, camera parameters for each image, *etc.*. It also associates with each point a unique label, the images it appears in, and the geometries of the local features in those images. This allows collection of the training sets based on estimated viewpoint changes [65], or inclusion of this information in the loss function [56] while training local descriptors.



Figure 3.6: Sparse model of central Rome using 21K photos produced by COLMAP’s SfM pipeline, taken from the software’s website.



Figure 3.7: Dense models of several landmarks produced by COLMAP’s MVS pipeline, taken from the software’s website.

Multiple-Kernel Local Descriptors

In this chapter, we describe a hand-crafted approach for local descriptors based on *Efficient Match Kernels* (EMK) [19, 17, 96]. EMK provides a flexible framework for matching sets, patches in our case, by encoding different properties of the set elements, pixels in our case. In particular, we build upon the hand-crafted kernel descriptor proposed by Bursuc *et al.* [23]. We extend our descriptor with post-processing that is learned from the data in unsupervised or supervised manner. We visualize and analyze its effect, and attempt to understand its advantages and disadvantages. Based on the analysis, we propose a simple combination of parametrizations each offering robustness to different types of patch mis-registrations. Interestingly, the same analysis is possible even for the learned post-processing. We demonstrate that its effect on the patch similarity is semantically meaningful. Finally, we evaluate the performance of the proposed descriptor on the PhotoTourism and HPatches benchmarks.

4.1 Preliminaries

In this section, we introduce *kernelized descriptors* and *explicit feature maps*. Kernelized descriptors allow us to approximate efficient match kernels. Following the formulation of Bursuc *et al.* [23], we represent a patch \mathcal{P} as a set of pixels $p \in \mathcal{P}$ and compare patches \mathcal{P} and \mathcal{Q} by the match kernel

$$\mathcal{M}(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} k(p, q),$$

where kernel $k : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is the kernel we wish to approximate. Explicit evaluation involves $|\mathcal{P}| \times |\mathcal{Q}|$ comparisons, making it prohibitively expensive. Instead, we use the explicit $\psi : \mathbb{R}^n \rightarrow \mathbb{R}^d$ feature map,

$$\mathcal{M}(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} k(p, q) \quad (4.1)$$

$$\approx \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} \psi(p)^\top \psi(q) \quad (4.2)$$

$$= \sum_{p \in \mathcal{P}} \psi(p)^\top \sum_{q \in \mathcal{Q}} \psi(q). \quad (4.3)$$

Vector $\mathbf{V}(\mathcal{P}) = \sum_{p \in \mathcal{P}} \psi(p)$ is called a *kernelized descriptor* (*KD*). $\mathbf{V}(\mathcal{P})$ is ℓ_2 -normalized by factor $\gamma(\mathcal{P}) = (\mathbf{V}(\mathcal{P})^\top \mathbf{V}(\mathcal{P}))^{1/2}$ to ensure unit self-similarity.

Explicit feature maps. As non-linear kernel for scalars we use the normalized Von Mises probability density function¹, which is used for image [96] and patch [23] representations. It is parametrized by κ controlling the shape of the kernel, where lower κ corresponds to wider kernel, *i.e.* less selective kernel. We use a stationary (shift invariant) kernel that, by definition, depends only on the difference $\Delta_n = p_n - q_n$, *i.e.* $k_{\text{VM}}(p_n, q_n) := k_{\text{VM}}(\Delta_n)$. We approximate this probability density function with Fourier series with N frequencies that produces a feature map $\psi_{\text{VM}} : \mathbb{R} \rightarrow \mathbb{R}^{2N+1}$. It has the property that

$$k_{\text{VM}}(p_n, q_n) \approx \psi_{\text{VM}}(p_n)^\top \psi_{\text{VM}}(q_n). \quad (4.4)$$

In particular we approximate the Fourier series by the sum of the first N terms as

$$k_{\text{VM}}(\Delta_n) \approx \sum_{i=0}^N \gamma_i \cos(i\Delta_n). \quad (4.5)$$

The feature map $\psi_{\text{VM}}(p_n)$ is designed as follows:

$$\psi_{\text{VM}}(p_n) = (\sqrt{\gamma_0}, \sqrt{\gamma_1} \cos(p_n), \dots, \sqrt{\gamma_N} \cos(Np_n), \sqrt{\gamma_1} \sin(p_n), \dots, \sqrt{\gamma_N} \sin(Np_n))^\top. \quad (4.6)$$

¹Also known as the periodic normal distribution

This vector has $2N + 1$ components. It is now easy to show that the inner product of two feature maps is approximating the kernel. That is,

$$\begin{aligned} \psi_{\text{VM}}(p_n)^\top \psi_{\text{VM}}(q_n) &= \gamma_0 + \sum_{i=1}^N \gamma_i (\cos(ip_n) \cos(iq_n) \\ &\quad + \sin(ip_n) \sin(iq_n)) \\ &= \sum_{i=0}^N \gamma_i \cos(i(p_n - q_n)) \\ &\approx k_{\text{VM}}(\Delta_n). \end{aligned} \tag{4.7}$$

The reader is encouraged to read prior work for details on these feature maps [101, 27], which are previously used in various contexts [96, 23].

4.2 Multiple-kernel joint encoding

In this section we consider different patch parametrizations and kernels that result in different patch similarity. We discuss the benefits of each and propose how to combine them. We further learn descriptor transformation with or without supervision and provide useful insight on how patch similarity is affected.

Patch attributes. We consider a pixel p to be associated with coordinates p_x, p_y in Cartesian coordinate system, coordinates p_ρ, p_ϕ in polar coordinate system, pixel gradient magnitude p_m , and pixel gradient angle p_θ . Angles $p_\theta, p_\phi \in [0, 2\pi]$, distance from the center p_ρ is normalized to $[0, 1]$, while coordinates $p_x, p_y \in \{1, 2, \dots, W\}$ for $W \times W$ patches. In order to use feature map ψ_{VM} , attributes p_ρ, p_x , and p_y are linearly mapped to $[0, \pi]$. The gradient angle is expressed *w.r.t.* the patch orientation, *i.e.* p_θ directly, or *w.r.t.* to the position of the pixel. The latter is given as $p_{\bar{\theta}} = p_\theta - p_\phi$.

Patch parametrizations. Composing patch kernel k as a product of kernels over different attributes enables easy design of various patch similarities. Correspondingly, this defines different KD. All attributes $p_x, p_y, p_\rho, p_\theta, p_\phi$, and $p_{\bar{\theta}}$ are matched by the Von Mises kernel, namely, $k_x, k_y, k_\rho, k_\theta, k_\phi$, and $k_{\bar{\theta}}$ parameterized by $\kappa_x, \kappa_y, \kappa_\rho, \kappa_\theta, \kappa_\phi$, and $\kappa_{\bar{\theta}}$, respectively. In a similar manner to SIFT, we apply a Gaussian mask by $p_g = \exp(-p_\rho^2)$ which gives more importance to central pixels.

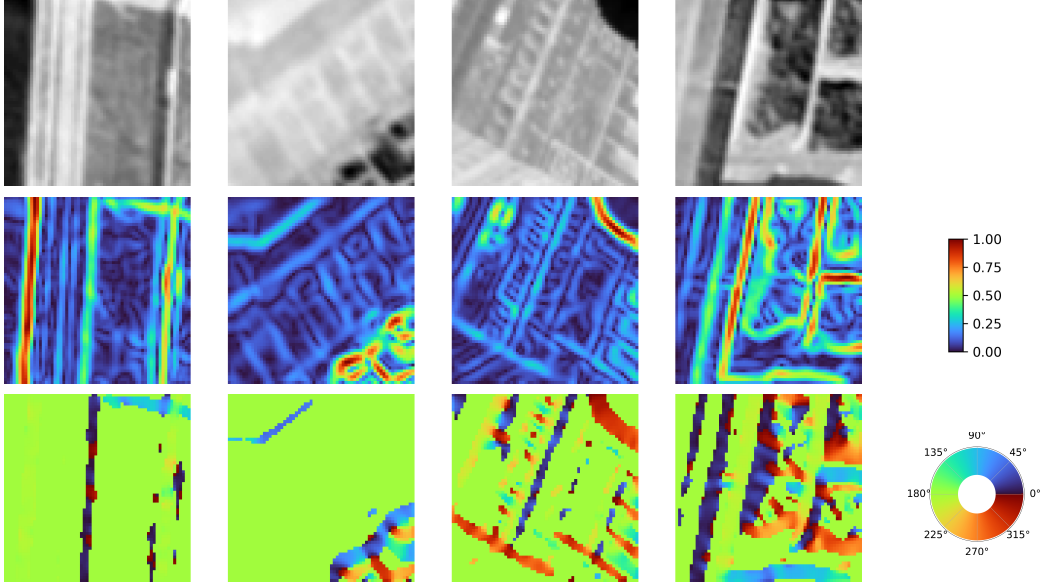


Figure 4.1: Image gradients for some sample patches (top) are shown. The middle row shows the corresponding gradient magnitudes and the bottom row shows gradient orientations (thresholded by mean of magnitudes). The colors are indicated by the colorbar and colorwheel.

In this chapter we focus on the two following match kernels over patches. One in *polar* coordinates $(\phi\rho\tilde{\theta})$ and one in *Cartesian* $(xy\theta)$ coordinates

$$\mathcal{M}_{\phi\rho\tilde{\theta}}(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} p_g q_g \sqrt{p_m} \sqrt{q_m} k_\phi(p_\phi, q_\phi) k_\rho(p_\rho, q_\rho) k_{\tilde{\theta}}(p_{\tilde{\theta}}, q_{\tilde{\theta}}) \quad (4.8)$$

$$\mathcal{M}_{xy\theta}(\mathcal{P}, \mathcal{Q}) = \sum_{p \in \mathcal{P}} \sum_{q \in \mathcal{Q}} p_g q_g \sqrt{p_m} \sqrt{q_m} k_x(p_x, q_x) k_y(p_y, q_y) k_\theta(p_\theta, q_\theta). \quad (4.9)$$

The KD for the two cases are given by

$$\mathbf{V}_{\phi\rho\tilde{\theta}}(\mathcal{P}) = \sum_{p \in \mathcal{P}} p_g \sqrt{p_m} \psi_\phi(p_\phi) \otimes \psi_\rho(p_\rho) \otimes \psi_{\tilde{\theta}}(p_{\tilde{\theta}}) \quad (4.10)$$

$$= \sum_{p \in \mathcal{P}} p_g \sqrt{p_m} \psi_{\phi\rho\tilde{\theta}}(p) \quad (4.11)$$

$$\mathbf{V}_{xy\theta}(\mathcal{P}) = \sum_{p \in \mathcal{P}} p_g \sqrt{p_m} \psi_x(p_x) \otimes \psi_y(p_y) \otimes \psi_\theta(p_\theta) \quad (4.12)$$

$$= \sum_{p \in \mathcal{P}} p_g \sqrt{p_m} \psi_{xy\theta}(p). \quad (4.13)$$

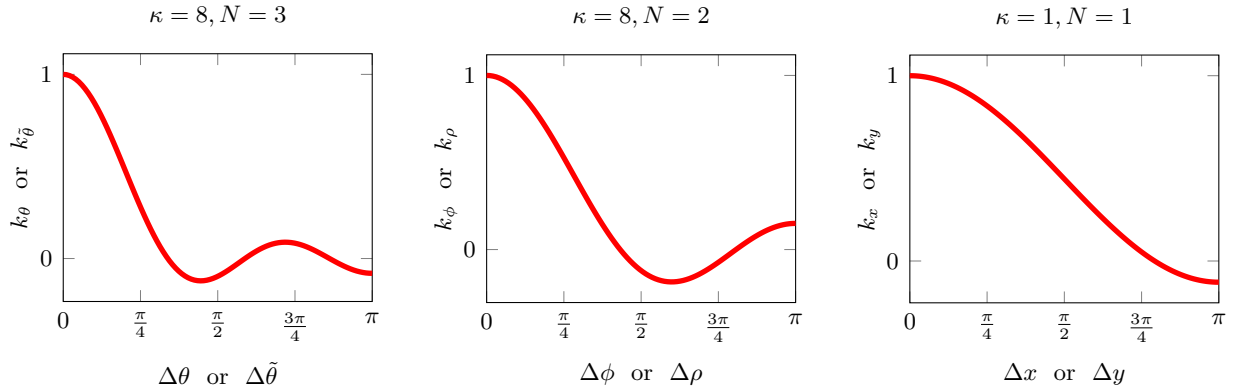


Figure 4.2: Kernel approximations used for pixel attributes. Parameter κ and the number of frequencies N define the final shape. The choice of kernel parameters is guided by [23].

The $\mathbf{V}_{\phi\rho\tilde{\theta}}$ variant is exactly the one proposed by Bursuc *et al.* [23], considered as a baseline for the descriptors introduced in this chapter. Different parametrizations result in different patch similarity, which is analyzed in the following. In Figure 4.2 we present the approximation of kernels used per attribute.

4.3 Whitening

It is known that further descriptor post-processing [75, 12, 23] is beneficial. In particular, KD is further centered and projected as

$$\hat{\mathbf{V}}(\mathcal{P}) = A^\top (\bar{\mathbf{V}}(\mathcal{P}) - \mu), \quad (4.14)$$

where $\mu \in \mathbb{R}^d$ and $A \in \mathbb{R}^{d \times d}$ are the mean vector and the projection matrix. These are commonly learned by PCA [40] or with supervision [75]. The final descriptor is always ℓ_2 -normalized in the end.

We detail different ways to learn the projection matrix A of (4.14) to perform the descriptor post-processing. Let us consider a learning set of patches \mathbb{P} and the corresponding set of descriptors $V_{\mathbb{P}} = \{V(\mathcal{P}), \mathcal{P} \in \mathbb{P}\}$. Let C be the covariance matrix of $V_{\mathbb{P}}$. Vector μ is the mean descriptor vector, and different ways to compute A are as follows.

Supervised whitening. We further assume that supervision is available in the form of pairs of matching patches. This is given by set $\mathbb{M} = \{(\mathcal{P}, \mathcal{Q}) \in \mathbb{P} \times \mathbb{P}, \mathcal{P} \sim \mathcal{Q}\}$, where \sim denotes matching patches. We follow the work of Mikolajczyk and Matas [59] to learn discriminative projections using the available supervision. The discriminative projection is composed of two parts, a whitening part and a rotation part. The whitening part is obtained from the intraclass (matching pairs) covariance matrix $C_{\mathbb{M}}$, while the rotation part is the PCA of the interclass (non-matching pairs) covariance matrix in the whitened

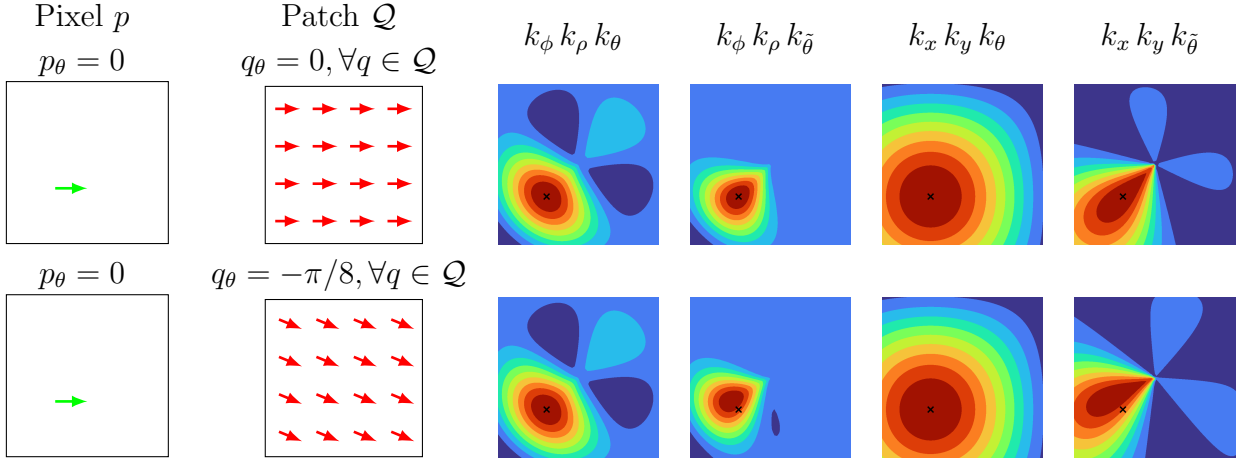


Figure 4.3: Patch maps for different parametrizations and kernels. Different parametrizations are presented, two in polar ($k_\phi k_\rho$) and two in Cartesian coordinates ($k_x k_y$), with absolute (k_θ) or relative ($k_{\bar{\theta}}$) gradient angle for each one. The similarity between each pixel of patch \mathcal{Q} and a single pixel p is shown over patch \mathcal{Q} . All pixels in \mathcal{Q} have the same gradient angle, which is shown in red arrows. The position of pixel p is shown with “ \times ” on the patch maps. We show examples for $\Delta\theta$ equal to 0 (top) and $\pi/8$ (bottom). At the top of each column the kernels that are used (patch similarity) are shown. The similarity is shown in a relative manner and, therefore, the absolute scale is missing. Ten isocontours are sampled uniformly and shown in different color.

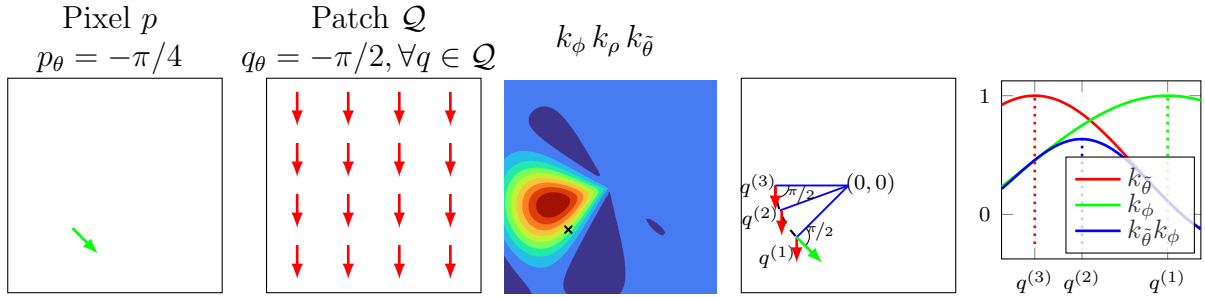


Figure 4.4: Patch map with polar parametrization $k_\phi k_\rho k_{\bar{\theta}}$ for $\Delta\theta = \pi/4$ and the pair of toy pixel and patch on the left. The example explains why the kernel undergoes shifting away from the position of pixel p . The diagram of the 4th column overlays pixel p and 3 pixels of patch \mathcal{Q} with the same distance from the center as p . On the rightmost plot, we illustrate $k_{\bar{\theta}}(p_{\bar{\theta}}, q_{\bar{\theta}})$, $k_\phi(p_\phi, q_\phi)$ for pixels q with $q_\rho = p_\rho$ (on the black dashed circle). $k_{\bar{\theta}}$ is maximized at $q^{(3)}$, k_ϕ at $q^{(1)}$, and their product at $q^{(2)}$.

space. We set the interclass one to be equal to C as this is dominated by non-matching pairs, while the intraclass one is given by

$$C_{\mathbb{M}} = \sum_{(\mathcal{P}, \mathcal{Q}) \in \mathbb{M}} (V(\mathcal{P}) - V(\mathcal{Q})) (V(\mathcal{P}) - V(\mathcal{Q}))^{\top}. \quad (4.15)$$

The projection matrix is now given by

$$A = C_{\mathbb{M}}^{-1/2} \text{eig}(C_{\mathbb{M}}^{-1/2} C C_{\mathbb{M}}^{-1/2}), \quad (4.16)$$

where eig denotes the eigenvectors of a matrix into columns. To reduce the descriptor dimensionality, only eigenvectors corresponding to the largest eigenvalues are used. The same holds for all cases that we perform PCA in the rest of the paper. We refer to this transformation as supervised whitening (W_{S}).

Unsupervised whitening. There is no supervision in this case and the projection is learned via PCA on set $V_{\mathbb{P}}$. In particular, projection matrix is given by

$$A = \text{eig}(C) \text{diag}(\lambda_1^{-1/2}, \dots, \lambda_d^{-1/2})^{\top}, \quad (4.17)$$

where diag denotes a diagonal matrix with the given elements on its diagonal, and λ_i is the i -th eigenvalue of matrix C . This method is called PCA whitening and we denote simply by W [40].

Unsupervised whitening with attenuation. We extend the PCA whitening scheme by introducing parameter t controlling the extent of whitening and the projection matrix becomes

$$A = \text{eig}(C) \text{diag}(\lambda_1^{-t/2}, \dots, \lambda_d^{-t/2})^{\top}, \quad (4.18)$$

where $t \in [0, 1]$, with $t = 1$ corresponding to the standard PCA whitening and $t = 0$ to simple rotation without whitening.

Equivalently, $t = 0$ imposes the covariance matrix to be identity. We call this method attenuated PCA whitening and denote it by W_{UA} .

Unsupervised whitening with shrinkage. The aforementioned process resembles covariance estimation with shrinkage [49, 48]. The sample covariance matrix is known to be a noise estimator, especially when the available samples are not sufficient relatively to the number of dimensions [48]. Ledoit and Wolf [48] propose to replace this by a linear combination of the sample covariance matrix and a structured estimator. Their solution is well conditioned and is shown to reduce the effect of noisy estimation in eigen decomposition. The imposed condition is simply that all variances are the same and all covariances are zero. The shrunk covariance is

$$\tilde{C} = (1 - \beta)C + \beta \mathbf{I}_d, \quad (4.19)$$

where \mathbf{I}_d is the identity matrix and β the shrinking parameter. This process “shrinks” extreme (too large or too small) eigenvalues to intermediate ones. In our experiments we show that a simple tuning of parameter β performs well across different context and datasets. The projection matrix is now

$$A = \text{eig}(C) \text{diag}((\alpha\lambda_1 + \beta)^{-1/2}, \dots, (\alpha\lambda_d + \beta)^{-1/2})^\top, \quad (4.20)$$

where $\alpha = 1 - \beta$. We call this method PCA whitening with shrinkage and denote it by W_{US} . We set parameter β equal to the i -th eigenvalue. A method similar to ours is used in the work of Brown *et al.* [21], but does not allow dimensionality reduction since descriptors are projected back to the original space after the eigenvalue clipping.

Learning affine transformation with Stochastic Gradient Descent (SGD).

We also learn the parameters of affine transform in a manner analogous to deep local descriptors, by optimizing a complex loss function using SGD. We use the same training procedure as HardNet [62], which is explained in further detail in Chapter 5. The results on the PhotoTourism dataset are presented in Table 4.2, and we visualize the learned kernels in Figure 4.8.

4.4 Visualization of kernels

We define pixel similarity $\mathcal{M}(p, q)$ as kernel response between pixels p and q , approximated as $\mathcal{M}(p, q) \approx \psi(p)^\top \psi(q)$. To show a spatial distribution of the influence of pixel p , we define a *patch map* of pixel p (fixed p_x , p_y , and p_θ). The patch map has the same size as the image patches; for each pixel q of the patch, map $\mathcal{M}(p, q)$ is evaluated for some constant value of q_θ .

For example, in Figure 4.3 patch maps for different kernels are shown. The position of p is denoted by \times symbol. Then, $p_\theta = 0$, while $q_\theta = 0$ for all spatial locations of q in the top row and $q_\theta = -\pi/8$ in the bottom row. This example shows the toy patches and their gradient angles in arrows to be more explanatory. The toy patches are directly defined by p_θ , and q_θ . Only p_θ and q_θ are used in later examples, while the toy patches are skipped from the figures.

The example in Figure 4.3 reveals a discontinuity near the center of the patch when pixel similarity is given by $\mathbf{V}_{\phi\tilde{\theta}}$ descriptor. It is caused by the polar coordinate system where a small difference in the position near the origin causes large difference in ϕ and $\tilde{\theta}$. The patch maps reveal weaknesses of kernel descriptors, such the aforementioned discontinuity, but also advantages of each parametrization. It is easy to observe that the kernel parametrized by Cartesian coordinates and absolute angle of the gradient ($\mathbf{V}_{xy\theta}$, third column) is insensitive to small translations, *i.e.* feature point displacement. Moreover, in the bottom row we see that using the relative gradient direction $\tilde{\theta}$ allows to compensate for imprecision caused by small patch rotation, *i.e.* the most similar pixel is not the one at the location of p with different $\tilde{\theta}$, but a rotated pixel with more similar value of $\tilde{\theta}$. This effect is further analyzed in Figure 4.4. The final similarity involves the product of

two kernels that both depend on angle ϕ . They are both maximized at the same point if $\Delta\theta = 0$, otherwise not. The larger $\Delta\theta$ is, the maximum value moves further (in the patch) from p .

We additionally construct patch maps in the case of descriptor post-processing by a linear transformation, *e.g.* descriptor whitening. For a patch of size $W \times W$, the contribution of a pixel pair is given by

$$\hat{\mathcal{M}}(p, q) = (A^\top(\bar{\psi}(p) - \bar{\mu}))^\top (A^\top(\bar{\psi}(q) - \bar{\mu})) \quad (4.21)$$

$$= (\bar{\psi}(p) - \bar{\mu})^\top AA^\top(\bar{\psi}(q) - \bar{\mu}) \quad (4.22)$$

$$= \bar{\psi}(p)^\top AA^\top \bar{\psi}(q) - \bar{\psi}(p)^\top AA^\top \bar{\mu} \\ - \bar{\psi}(q)^\top AA^\top \bar{\mu} + \bar{\mu}^\top AA^\top \bar{\mu}, \quad (4.23)$$

where $\bar{\mu} = \frac{\mu}{W^2}$, and $\bar{\psi}(p) = \frac{\psi}{\gamma}$ where γ accounts for the ℓ_2 -normalization of the whole patch. If A is a rotation matrix then the similarity is affected just by shifting by $\bar{\mu}$. After the transformation, the similarity is no longer shift-invariant. Non-linear post-processing, such as power-law normalization or simple ℓ_2 normalization cannot be visualized, as it acts after the pixel aggregation.

Combining kernel descriptors. We propose to take advantage of both parametrizations $\mathbf{V}_{\phi\rho\tilde{\theta}}$ and $\mathbf{V}_{xy\theta}$, by summing their contribution. This is performed by simple concatenation of the two descriptors. Finally, whitening is jointly learned and dimensionality reduction is performed.

In Figure 4.5 we show patch maps for the individual and combined representation, for different pixels p . Observe how the combined one better behaves around the center. The combined descriptor inherits reasonable behavior around the patch center and insensitivity to position misalignment from the Cartesian parametrization, while insensitivity to dominant orientation misalignment from the polar parametrization, as shown earlier.

When combining the descriptors of different parametrization by concatenation we use both with equal contribution, *i.e.* the final similarity is equal to $k_\phi k_\rho k_{\tilde{\theta}} + k_x k_y k_\theta$. In the case of the raw descriptors this is clearly suboptimal. One would rather regularize by $k_\phi k_\rho k_{\tilde{\theta}} + w k_x k_y k_\theta$ and search for the optimal value of scalar w . We prove that this is not necessary in the case of post-processing by supervised whitening, where the optimal regularization is included in the projection matrix.

We denote a set of descriptors without regularized concatenation by $V_{\mathbb{P}}$ when $w = 1$, while the $V_{\mathbb{P}}^{(w)}$ when $w \neq 1$. It holds that $V_{\mathbb{P}}^{(w)} = \{WV(\mathcal{P}), \mathcal{P} \in \mathbb{P}\}$, where W is a diagonal matrix with ones on the dimensions corresponding to the first descriptors (for $k_\phi k_\rho k_{\tilde{\theta}}$), and has all the rest elements of the diagonal equal to w . The covariance matrix of $V_{\mathbb{P}}$ is C , while of $V_{\mathbb{P}}^{(w)}$ it is $C^{(w)} = WCW^\top$.

Learning the supervised whitening on $V_{\mathbb{P}}$ as in (4.16) produces projection matrix

$$A = C_{\mathbb{M}}^{-1/2} \text{eig}(C_{\mathbb{M}}^{-1/2} C C_{\mathbb{M}}^{-1/2}), \quad (4.24)$$

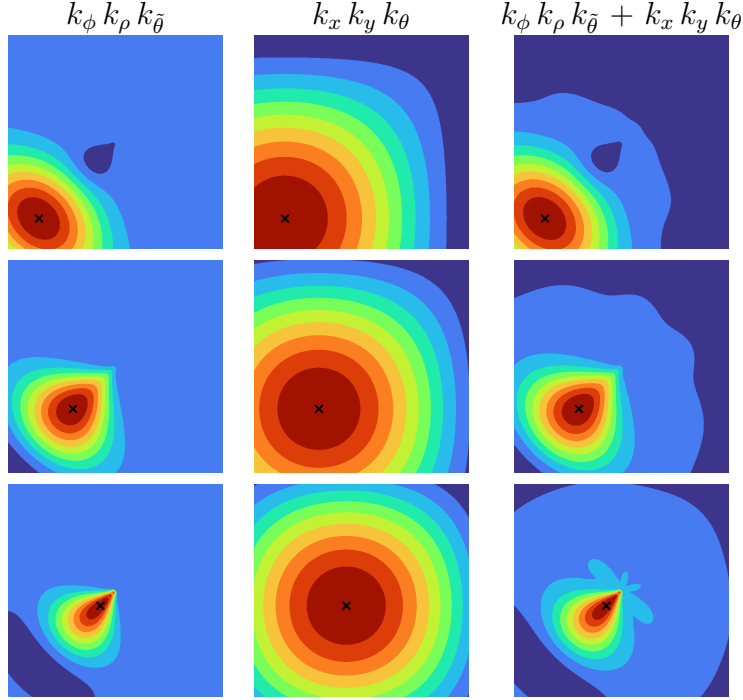


Figure 4.5: Patch maps for different pixels and parametrizations and their concatenation. Two parametrizations in polar and Cartesian coordinates, with relative and absolute gradient angle, respectively are presented. $\Delta\theta$ is fixed to be 0 (individual values of p_θ and q_θ do not matter due to shift invariance) and pixel p is shown with “ \times ”. Note the behaviour around the centre in the concatenated case. Ten isocontours are sampled uniformly and shown in different color.

while learning it on $V_{\mathbb{P}}^{(w)}$ produces projection matrix

$$A^{(w)} = C_{\mathbb{M}}^{(w)-1/2} \text{eig}(C_{\mathbb{M}}^{(w)-1/2} C^{(w)} C_{\mathbb{M}}^{(w)-1/2}). \quad (4.25)$$

Cholesky decomposition of C gives

$$C = U^\top U = LL^\top, \quad (4.26)$$

which leads to the Cholesky decomposition

$$C^{(w)} = WU^\top UW^\top = WLL^\top W^\top. \quad (4.27)$$

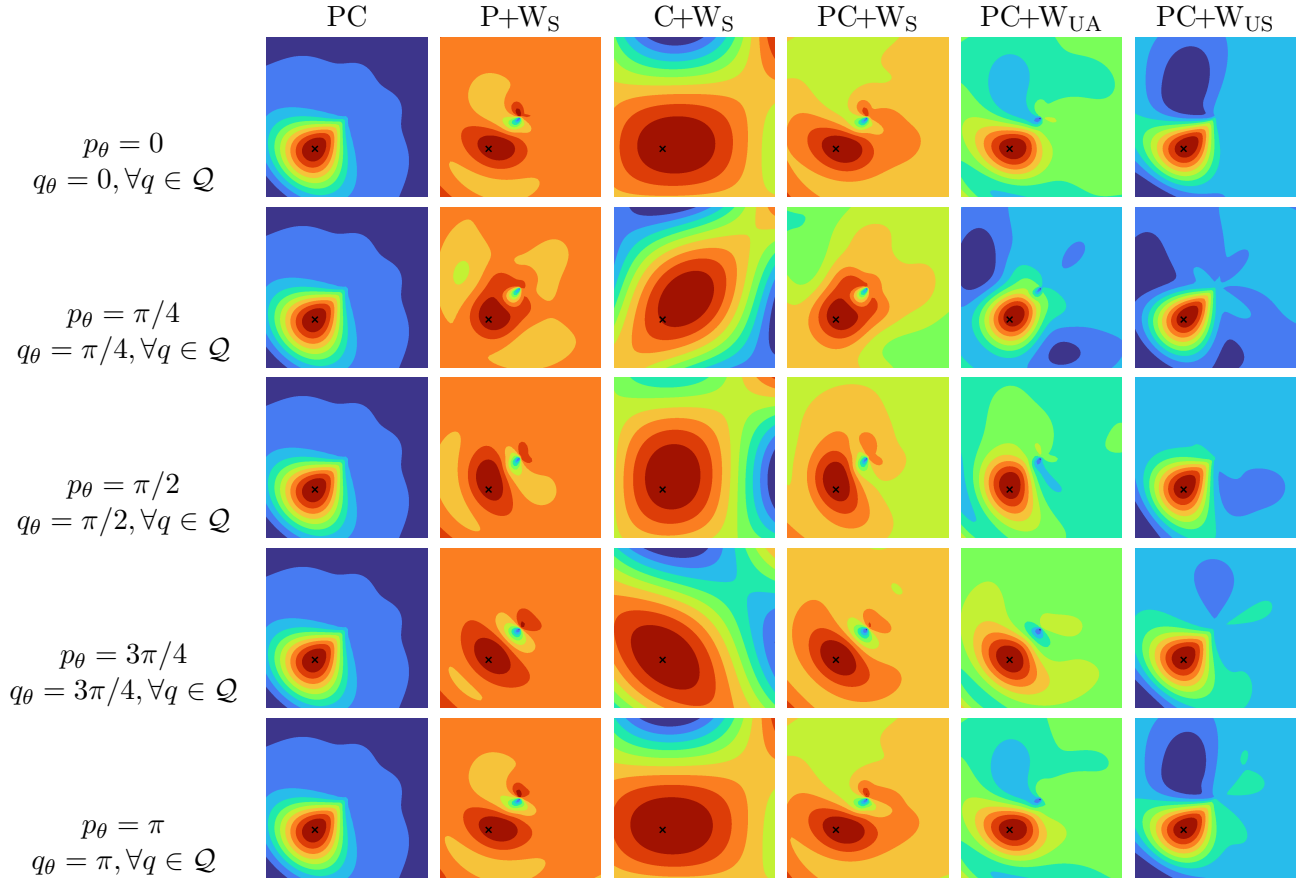


Figure 4.6: Patch maps for different parametrizations, their concatenation, different post-processing methods, and varying p_θ and q_θ , while $\Delta\theta$ is always 0. Pixel p is shown with “ \times ”. P: polar parametrization, C: Cartesian parametrization, W_S : supervised whitening, W_{UA} : unsupervised whitening (attenuation), W_{US} : unsupervised whitening (shrinkage). W_{UA} is shown for $t = 0.7$ and W_{US} for $\beta = \lambda_{40}$. Whitening is learned on Liberty dataset. Observe that the similarity is no more shift invariant after the whitening and how the shape follows the angle of the gradients. Ten isocontours are sampled uniformly and shown in different color.

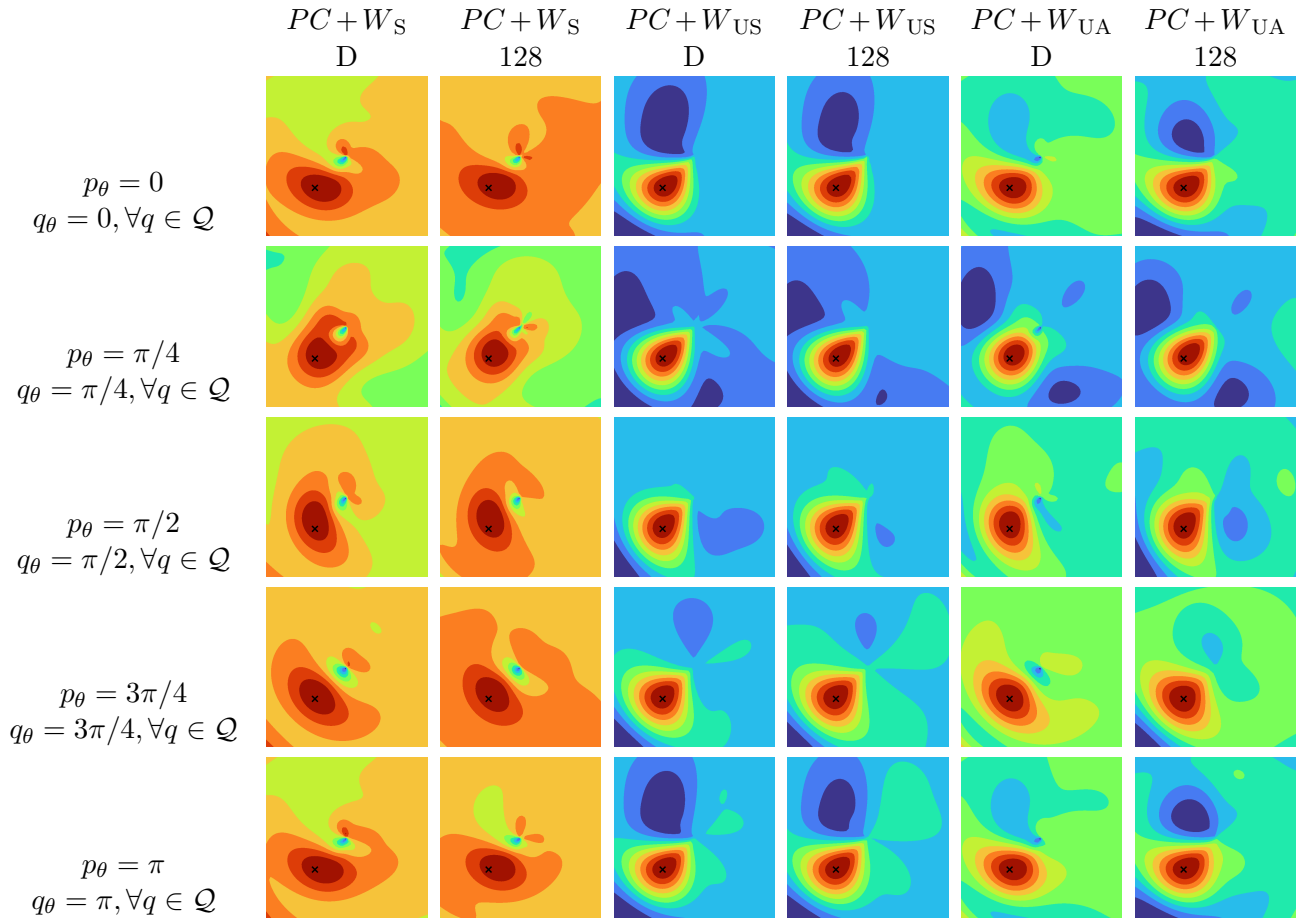


Figure 4.7: Patch maps for concatenated parametrization, different post-processing methods, with and without dimensionality reduction. p_θ and q_θ are varying, while $\Delta\theta$ is always 0. Pixel p is shown with “ \times ”. P: polar parametrization, C: Cartesian parametrization, W_S : supervised whitening. Whitening is learned using W_S : supervised whitening, W_{UA} : unsupervised whitening (attenuation), W_{US} : unsupervised whitening (shrinkage) on Liberty dataset. $\Delta\theta$ is fixed to be 0. Ten isocontours are sampled uniformly and shown in different color.

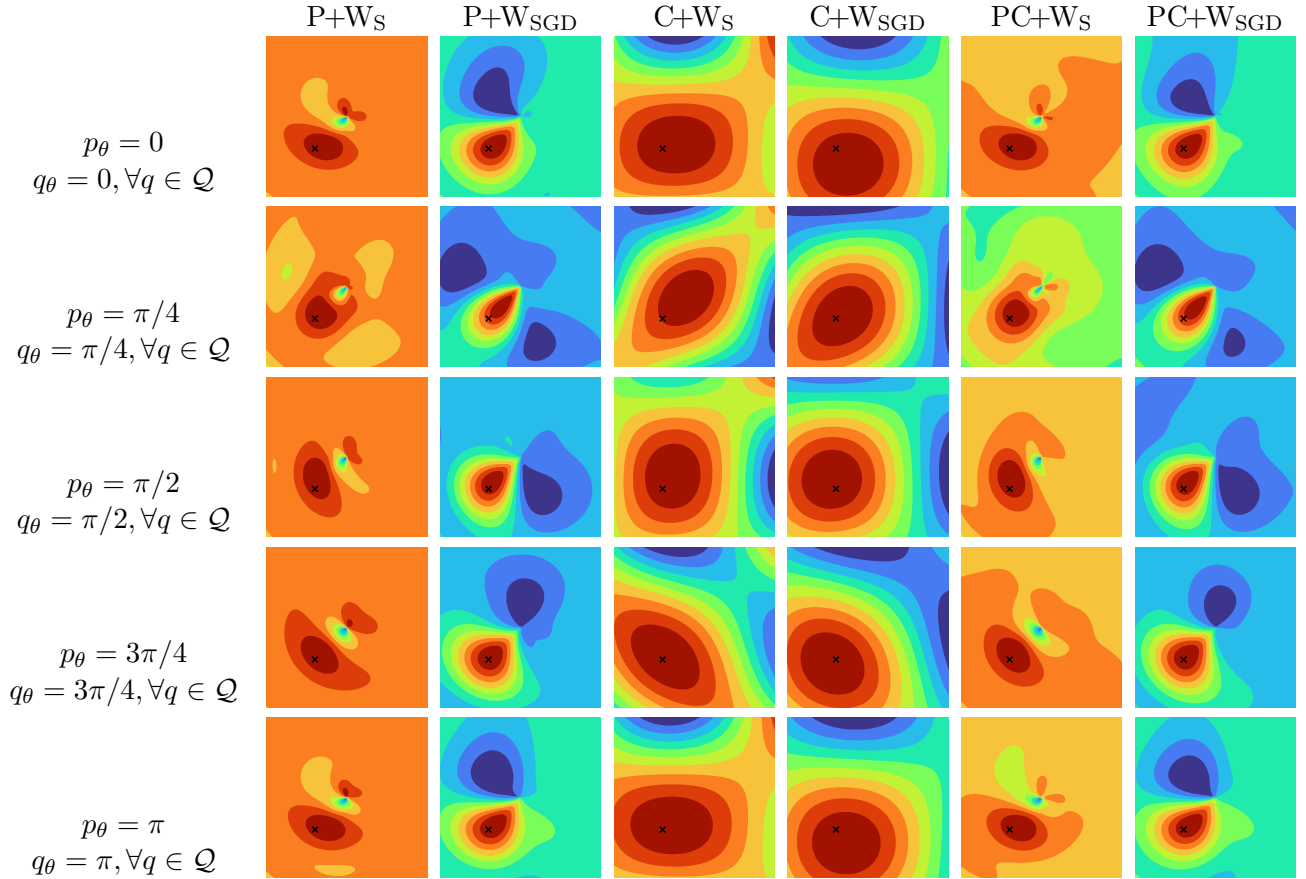


Figure 4.8: Patch maps for different parametrizations, their concatenation, different post-processing methods, and varying p_θ and q_θ , while $\Delta\theta$ is always 0. Pixel p is shown with “x”. P: polar parametrization, C: Cartesian parametrization, W_S: supervised whitening. Whitening is learned using Linear Discriminant Analysis (W_S) or Stochastic Gradient Descent (W_{SGD}) on Liberty dataset. $\Delta\theta$ is fixed to be 0. Ten isocontours are sampled uniformly and shown in different color.

Using (4.27) allows us to rewrite (4.25) as

$$\begin{aligned}
A^{(w)} &= (L^\top W^\top)^{-1} \text{eig}((WU^\top)^{-1}WCW^\top(L^\top W^\top)^{-1}) \\
&= (W^\top)^{-1}(L^\top)^{-1} \text{eig}(U^{\top-1}W^{-1}WCW^\top(W^\top)^{-1}(L^\top)^{-1}) \\
&= (W^\top)^{-1}(L^\top)^{-1} \text{eig}(U^{\top-1}C(L^\top)^{-1}) \\
&= (W^\top)^{-1}(L^\top)^{-1} \text{eig}(C_{\mathbb{M}}^{-1/2}CC_{\mathbb{M}}^{-1/2}) \\
&= (W^\top)^{-1}A.
\end{aligned} \tag{4.28}$$

Whitening descriptor $V(\mathcal{P}) \in V_{\mathbb{P}}$ with matrix A is performed by

$$\hat{V}(\mathcal{P}) = A^\top(V(\mathcal{P}) - \mu), \tag{4.29}$$

while whitening descriptor $V(\mathcal{P})^{(w)} \in V_{\mathbb{P}}^{(w)}$ with matrix $A^{(w)}$ is performed by

$$\begin{aligned}
\hat{V}(\mathcal{P})^{(w)} &= A^{(w)\top}(WV(\mathcal{P}) - W\mu) \\
&= A^\top W^{-1}(WV(\mathcal{P}) - W\mu) \\
&= \hat{V}(\mathcal{P}).
\end{aligned} \tag{4.30}$$

No matter what the regularization parameter is, the descriptor is identical after whitening. We conclude that there is no need to perform such regularized concatenation.

Understanding the whitened patch similarity. We learn the different whitening variants of Section 4.3 and visualize their patch maps in Figure 4.6. All examples shown are for $\Delta\theta = 0$ but gradient angles p_θ and q_θ jointly vary. We initially observe that the similarity is shift invariant only in the first column of patch maps where no whitening is applied. This is expected by definition. Projecting by matrix A does not allow to reconstruct the shift invariant kernels anymore; the similarity does not only depend on $\Delta\theta$, which is 0, but also on p_θ and q_θ .

The patch similarity learned by whitening exhibits an interesting property. The shape of the 2D similarity becomes anisotropic and gets aligned with the orientation of the gradient. Equivalently, it becomes perpendicular to the edge on which the pixel lies. This is a semantically meaningful effect. It prevents over-counting of pixel matching along aligned edges of the two patches. In the case of a blob detector this can provide tolerance to errors in the scale estimation, *i.e.* the similarity remains large towards the direction that the blob edges shift in case of scale estimation error.

We presume that this is learned by pixels with similar gradient angle that co-occur frequently. A similar effect is captured by both the supervised and the unsupervised whitening with covariance shrinkage, it is, though, less evident in the case of W_{US} . Moreover, we see that it is mostly the Cartesian parametrization that allows this kind of deformation.

According to our interpretation, supervised whitening [59, 75] owes its success to covariance estimation that is more noise free. The noise removal comes from supervision,

but we show that standard approaches for well-conditioned and accurate covariance estimation have similar effect on the patch similarity even without supervision. The observation that different parametrizations allow for different types of co-occurrences to be captured is related to other domains too. For instance, CNN-based image retrieval exhibits improvements after whitening [12], but this is very unequal between average and max pooling. However, observing the differences is not as easy as in our case with the visualized patch similarity.

Patch maps are a way to visualize and study the general shape of the underlined similarity function. In a similar manner, we visualize the kernel responses for a particular pair of patches to reflect which are the pixels contributing the most to the patch similarity. This is achieved by assigning strength $\sum_{q \in \mathcal{Q}} \hat{\mathcal{M}}(p, q)$ to pixel p . In cases without whitening, $\mathcal{M}(p, q)$ is used. We present such heat maps in Figure 4.9. Whitening significantly affects the contribution of most pixels. The over-counting phenomenon described in Section 4.4 is also visible; some of the long edges are suppressed.

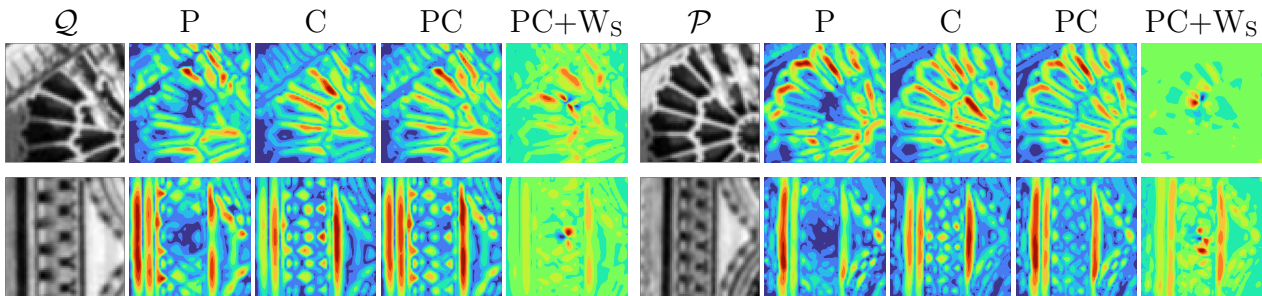


Figure 4.9: Positive patch pairs (patches \mathcal{Q} and \mathcal{P}) and the corresponding heat maps for polar (P), Cartesian (C), combined (PC), and whitened combined (PC+W_S) parametrization. Red (blue) corresponds to maximum (minimum) value. Heat maps on the left side correspond to $\sum_{p \in \mathcal{P}} \mathcal{M}(p, q)$, while the ones on the right side to $\sum_{q \in \mathcal{Q}} \mathcal{M}(p, q)$. In the case of PC+W_S, $\hat{\mathcal{M}}(p, q)$ is used instead of $\mathcal{M}(p, q)$.

4.5 Experiments

As previously introduced in Chapter 3, we evaluate our descriptor on two benchmarks, namely the widely used *PhotoTourism* (PT) dataset [103], and the recently released *HPatches* (HP) dataset [13]. We first show the impact of the shrinkage parameters in unsupervised whitening, and then compare with the baseline method of Bursuc *et al.* [23] on top of which we build our descriptor. We examine the generalization properties of whitening when learned on PT but tested on HP, and finally compare against state-of-the-art descriptors on both benchmarks. In all our experiments with descriptor post-processing the dimensionality is reduced to 128, while the combined descriptor original has 238 dimensions, except for the cases where the input descriptor is already of lower dimension. Our experiments are conducted with a Matlab implementation of the descriptor, which

takes 5.6 ms per patch for extraction on a single CPU on a 3.5GHz desktop machine. A GPU implementation reduces time to 0.1 ms per patch on an Nvidia Titan X.

Datasets and protocols. We use the two publicly available patch datasets described in Chapter 3. The Phototourism dataset contains three sets of patches, namely, Liberty (Li), NotreDame (No) and Yosemite (Yo). Additionally, labels are provided to indicate the 3D point that the patch corresponds to, thereby providing supervision. It has been widely used for training and evaluating local descriptors. Performance is measured by the false positive rate at 95% of recall (FPR95). The protocol is to train on one of the three sets and test on the other two. An average over all six combinations is reported.

The HPatches dataset contains local patches of higher diversity, is more realistic, and during evaluation the performance is measured on three tasks: *verification*, *retrieval*, and *matching*. We follow the standard evaluation protocol [13] and report mean Average Precision (mAP). We follow the common practice and use models learned on Liberty of PT to compare descriptors that have not used HP during learning. We evaluate on all 3 train/test splits and report the average performance. All reported results on HP (our and other descriptors) are produced by our own evaluation by using the provided framework, and descriptors².

Impact of the shrinkage parameter. We evaluate the impact of the shrinkage parameter involved in the unsupervised whitening. It is t for W_{UA} and $\beta = \lambda_i$ for W_{US} . Results are presented in Figure 4.11 for evaluation on PT and HP dataset, while the whitening is learned on the same or different dataset. The performance is stable for a range of values, which makes it easy to tune in a robust way across cases and datasets. In the rest of our experiments we set $t = 0.7$ and $\beta = \lambda_{40}$. In Figure 4.10 we show the eigenvalues used by W , W_{UA} , and W_{US} . The contrast between the larger and smaller eigenvalues is decreased.

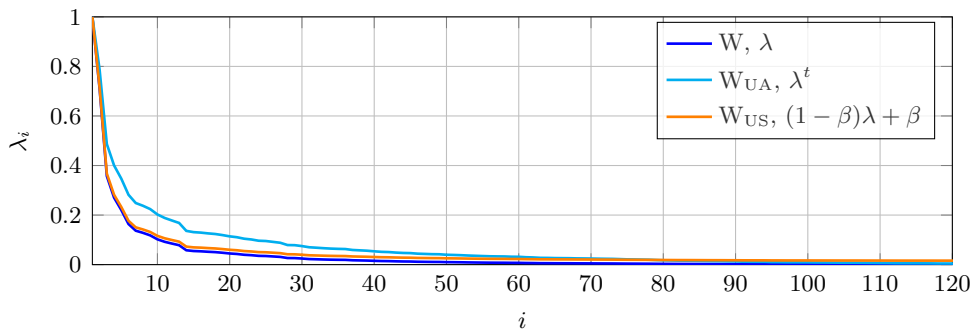


Figure 4.10: Eigenvalues for standard PCA whitening, attenuated whitening ($t = 0.7$) and whitening with shrinkage ($\beta = \lambda_{40}$). Values are normalized so that the maximum eigenvalue is 1. First 120 eigenvalues (out of 238) are shown.

²L2Net and HardNet descriptors were provided by the authors of HardNet [62].

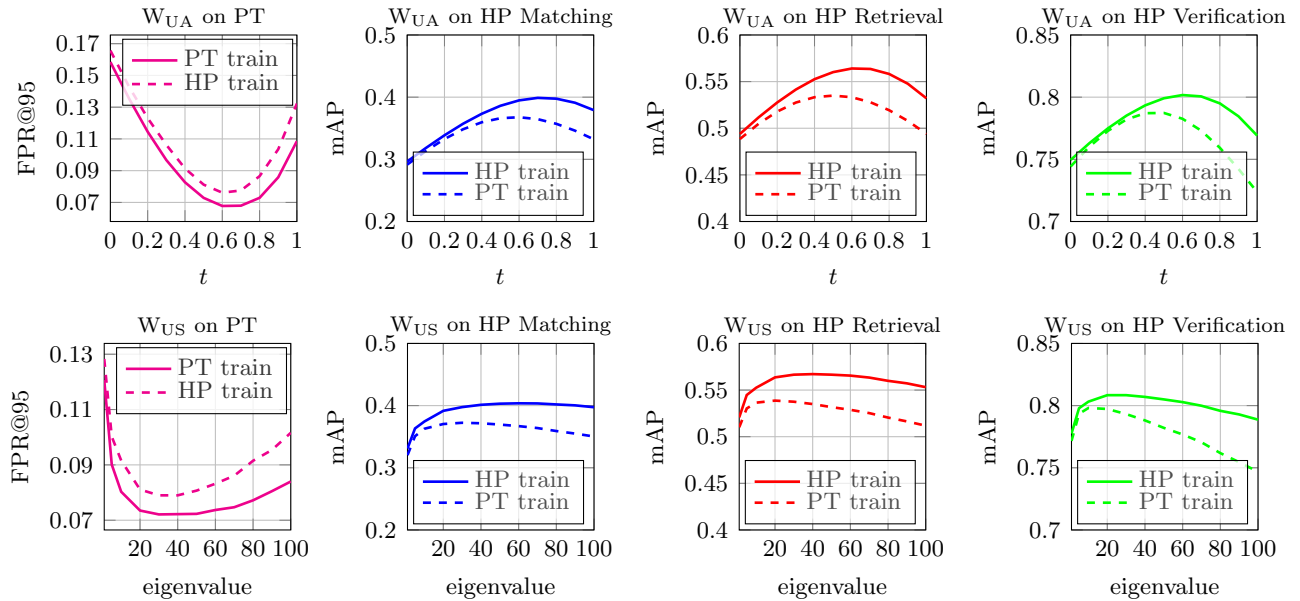


Figure 4.11: Impact of the shrinkage parameter for unsupervised whitening when trained on the same or different dataset. Performance is evaluated on PhotoTourism and HPatches datasets versus shrinkage parameter t for the attenuated whitening W_{UA} (top row), and versus shrinkage parameter $\beta = \lambda_i$ for whitening with shrinkage W_{US} (bottom row).

Comparison with the baseline. We compare the combined descriptor against the different parametrizations when used alone. The experimental evaluation is shown in Table 4.1 for the PT dataset. The baseline is followed by PCA and square-rooting, as originally proposed in [23]. We did not consider the square-rooting variant in our analysis in Section 4.2 because such non-linearity does not allow to visualize the underlined patch similarity. Supervised whitening on top of the combined descriptor performs the best. Unsupervised whitening significantly improves too, while it does not require any labeling of the patches.

Polar parametrization with the relative gradient direction (*polar*) significantly outperforms the Cartesian parametrization with the absolute gradient direction (*cartes*). After the descriptor post-processing (*polar* + W_S vs. *cartes* + W_S), the gap is reduced. The performance of the combined descriptor (*polar* + *cartes*) without descriptor post-processing is worse than the baseline descriptor. That is caused by the fact, that the two descriptors are combined with an equal weight, which is clearly suboptimal. No attempt is made to estimate the mixing parameter explicitly. It is implicitly included in the post-processing stage. Figure 4.13 presents patch similarity histograms for matching and non-matching pairs, showing how their separation is improved by the final descriptor.

We perform an experiment with synthetic patch transformations to test the robustness of different parametrizations. The whole patch is synthetically rotated or translated by

Test			Liberty		Notredame		Yosemite	
Train	D	Mean	No	Yo	Li	Yo	Li	No
<i>polar</i> [23]	175	22.42	24.34	24.34	16.06	16.06	26.85	26.85
<i>cartes</i>	63	35.87	34.06	34.06	34.10	34.10	39.47	39.47
<i>polar</i> + <i>cartes</i>	238	25.37	26.16	26.16	20.04	20.04	29.91	29.91
<i>polar</i> + PCA + SQRT [23]	128	8.30	12.09	13.13	5.16	5.41	7.52	6.49
<i>polar</i> [23] + W_S	128	7.06	8.55	10.48	4.40	3.94	8.86	6.12
<i>cartes</i> + W_S	63	15.13	17.31	20.34	10.90	11.85	16.84	13.55
<i>polar</i> + <i>cartes</i> + W_S	128	5.94	7.46	9.85	3.45	3.55	6.47	4.89
<i>polar</i> + <i>cartes</i> + W_{UA}	128	6.79	10.59	11.17	3.80	4.36	5.58	5.16
<i>polar</i> + <i>cartes</i> + W_{US}	128	7.22	10.61	11.14	4.27	4.46	6.75	6.09

Table 4.1: Performance comparison on PhotoTourism dataset between the baseline approach and our combined descriptor. The benefit of learned whitening (W_S), over the standard PCA followed by square-rooting, as well as the other variants that do additional regularization (W_{UA} , W_{US}) without supervision, is presented. FPR95 is reported for all methods.

Test			Liberty		Notredame		Yosemite	
Train	D	Mean	No	Yo	Li	Yo	Li	No
P + W_{SGD}	128	6.19	8.30	8.80	3.79	3.24	7.35	5.68
C + W_{SGD}	128	13.68	16.32	16.63	10.06	9.42	15.77	13.89
PC + W_{SGD}	128	5.76	7.52	8.42	3.45	2.83	6.80	5.53

Table 4.2: Performance on PhotoTourism dataset when using Stochastic Gradient Descent (SGD) to train the whitening transform. P, C and PC are compared. FPR95 is reported for all methods.

appropriately transforming pixel position and gradient angle in the case of rotation. A fixed amount of rotation/translation is performed for one patch of each pair of the PT dataset and results are presented in Figure 4.12. It is indeed verified that the Cartesian parametrization is more robust to translations, while the polar one to rotations. The joint one finally partially enjoys the benefits of both.

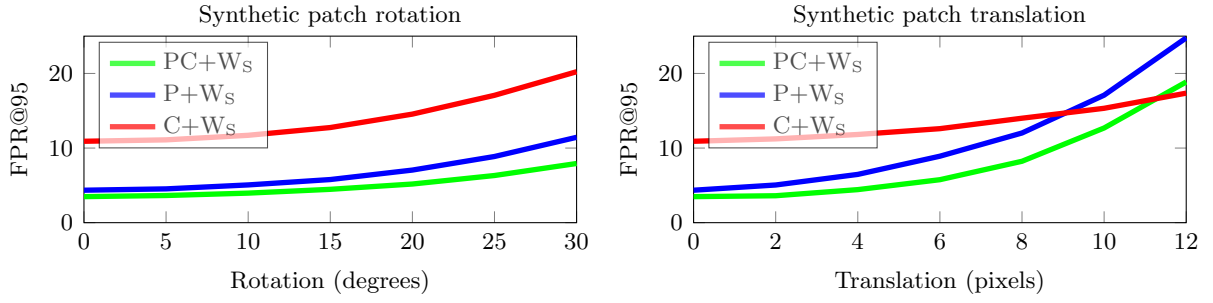


Figure 4.12: Performance on PT (training on Liberty, testing on NotreDame) when one patch of each pair undergoes synthetic rotation or translation.

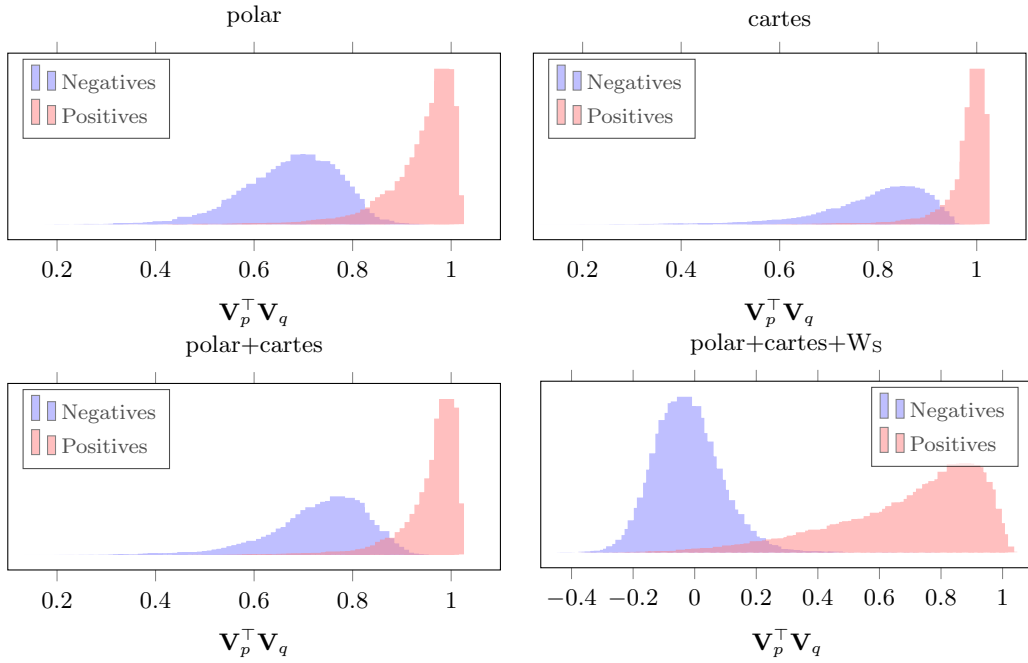


Figure 4.13: Histograms of patch similarity for positive and negative patch pairs. Histograms are constructed from 50K matching and 50K non-matching pairs from NotreDame dataset.

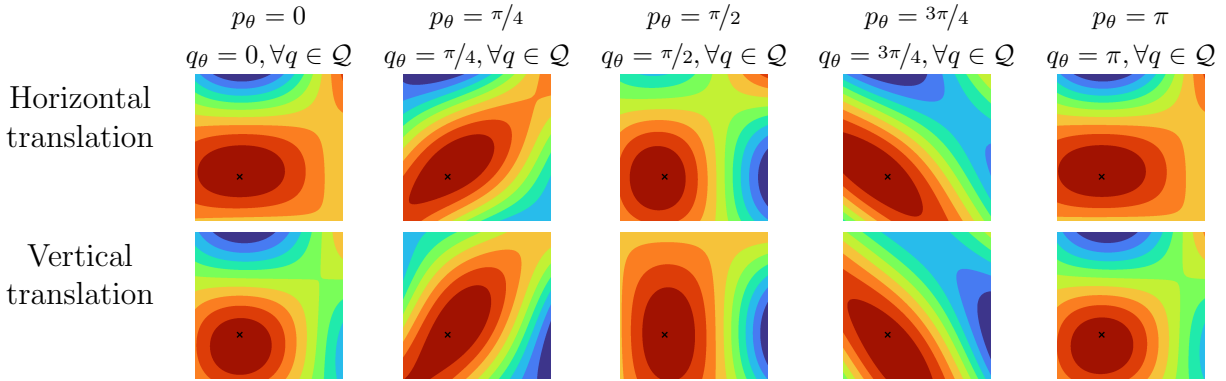


Figure 4.14: Patch maps for Cartesian parametrization with whitening transform learned on synthetically generated correspondences from HPatches dataset that are translated versions of local-patches detected using HessianAffine detector. $\Delta\theta$ is fixed to be 0 and pixel p is shown with “ \times ”. Note the behaviour around the centre in the concatenated case. Ten isocontours are sampled uniformly and shown in different color.

We perform an experiment with synthetically generated local patches to observe the effect of whitening when learned on specific types of perturbations. Local features are detected using the HessianAffine detector for a large number of images from the HPatches dataset. For each feature, 5 corresponding local patches are extracted by translating the centres of the features in either horizontal or vertical direction. These local patches are used to randomly generate a large set of positives (matching pairs), and used to learn the supervised whitening transform for the Cartesian variant. The resulting kernels are presented in Figure 4.14. It is observed that the kernels are accordingly aligned to different axes, to account for the respective direction of translation.

Generalization of whitening. We learn the whitening on PT or HP (supervised and unsupervised) and evaluate the performance on HP. We present results in Table 4.3. Whitening always improves the performance of the raw descriptor. The unsupervised variant is superior when learning it on an independent dataset. It generalizes better, implying over-fitting of the supervised one (recall the observations of Figure ??). Learning on HP (the corresponding training part per split) with supervision significantly helps. Note that PT contains only patches detected by DoG, while HP uses a combination of detectors.

Name	Train	Sup.	R	M	V
<i>polar + cartes</i>	N/A	N/A	45.23	29.68	77.78
<i>polar + cartes</i> + W_{UA}	PT	No	52.78	36.46	77.31
<i>polar + cartes</i> + W_{US}	PT	No	53.50	37.16	78.81
<i>polar + cartes</i> + W_S	PT	Yes	49.66	32.58	75.82
<i>polar + cartes</i> + W_{UA}	HP	No	56.36	39.88	80.06
<i>polar + cartes</i> + W_{US}	HP	No	56.71	40.13	80.70
<i>polar + cartes</i> + W_S	HP	Yes	61.79	44.40	83.50

Table 4.3: Generalization of different whitening approaches. Mean Average Precision (mAP) for 3 tasks of HP, namely Retrieval (R), Matching (M), and Verification (V). The whitening is learned on PT or HP. We denote supervised by *Sup*.

Comparison with the State of the Art. We compare the performance of the proposed descriptor with previously published results on PhotoTourism dataset. Results are shown in Table 4.4. Our method obtains the best performance among the unsupervised/hand-crafted approaches by a large margin. Overall, it comes right after the two very recent CNN-based descriptors, namely L2Net [93] and HardNet [62]. The advantage of our approach is the low cost of the learning. It takes less than a minute; about 45 seconds to extract descriptors of Liberty and about 10 seconds to compute the projection matrix. CNN-based competitors require several hours or days of training.

The comparison on the HPatches dataset is reported in Figure 4.15. We use the provided descriptors and framework to evaluate all approaches by ourselves. For the descriptors that require learning, the model that is learned on Liberty-PT is used. Our unsupervised descriptor is the top performing hand-crafted variant by a large margin. Overall, it is always outperformed by HardNet, L2Net, while on verification is it additionally outperformed by DDesc and TF-M. Verification is closer to the learning task (loss) involved in the learning of these CNN-based methods.

Finally, we learn supervised whitening W_S for all other descriptors, post-process them, and present results in Figure 4.16. The projection matrix is learned on HP, in particular the training part of each split. Supervised whitening W_S consistently boosts the performance of all descriptors, while this comes at a minimal extra cost compared to the initial training of a CNN descriptor. Our descriptor comes 3rd at 2 out of 3 tasks. Note that it uses the whitening learned on HP (similarly to all other descriptors of this comparison), but does not use the PT dataset at all. All CNN-based descriptors train their parameters on Liberty-PT which is costly, while the overall learning of our descriptor is again in the order of a single minute.

Supervised			Unsupervised		
Name	D	FPR@95	Name	D	FPR@95
Brown <i>et al.</i> [21]	2936	15.36	RootSIFT	128	26.14
Trzcinski <i>et al.</i> [98]	128	17.08	RootSIFT + PCA + SQRT [23]	80	17.51
Simonyan <i>et al.</i> [88]	7377	10.38	<i>polar</i> + PCA + SQRT [23]	128	8.30
DCS2S [106]	512	9.67	<i>polar</i> + <i>cartes</i> + W_{UA} (our)	128	6.79
DDESC [87]	128	9.85	<i>polar</i> + <i>cartes</i> + W_{US} (our)	128	7.21
Matchnet [35]	4096	7.75			
TFM [14]	128	6.47			
L2Net+ [93]	128	2.22			
HardNet+ [62]	128	1.51			
<i>polar</i> + <i>cartes</i> + W_S (our)	128	5.98			

Table 4.4: Performance comparison with the state of the art on PhotoTourism dataset. We report FPR@95 averaged over 6 dataset combinations for supervised (left) and unsupervised (right) approaches. The whitening for our descriptor is learned on the corresponding training part of PT for each combination.

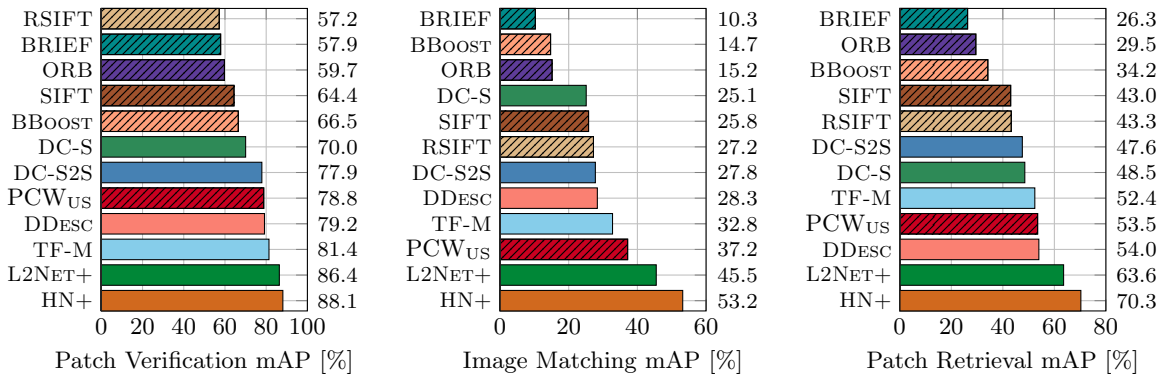


Figure 4.15: Performance comparison on HP benchmark. The learning, whenever applicable, is performed on Liberty of PT dataset. Descriptors that do not require any supervision in the form of labeled patches, *i.e.* hand-crafted or unsupervised, are shown in striped bars. Our descriptor is denoted by PCW_{US} (P=*polar*, C=*cartes*).

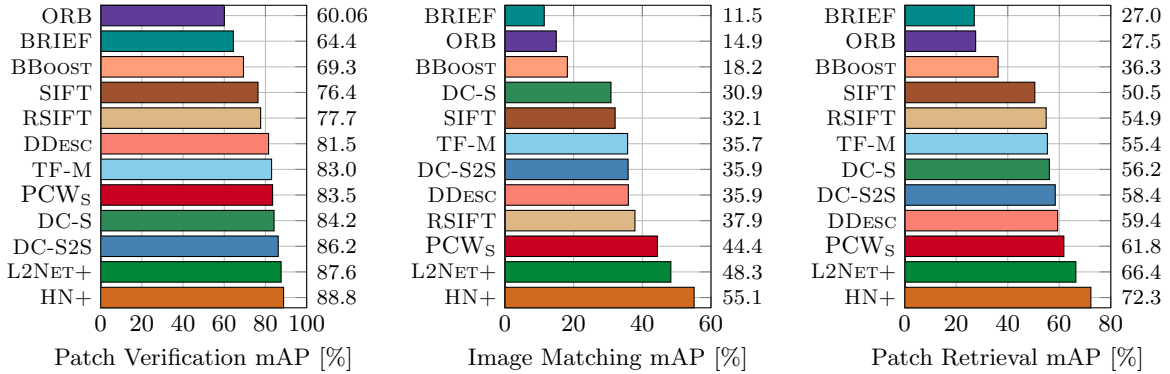


Figure 4.16: Performance comparison on HP benchmark when post-processing all descriptors with supervised whitening W_S which is learned on HP. The initial learning of the descriptor, whenever applicable, is performed on Liberty of PT dataset. Our descriptor uses the whitening learned on HP and does not use the PT dataset at all. Our descriptor is denoted by PCW_S ($P=polar$, $C=cartes$).

4.6 Discussion

In this chapter, we have proposed a multiple-kernel local-patch descriptor based on efficient match kernels from pixel gradients. We investigated descriptor whitening as post-processing in supervised and unsupervised settings. We have shown that tolerance to different types of mis-registration can be achieved by combining two parametrizations of gradient position and direction. Polar parametrization accounts for noise in the patch dominant orientation detection while Cartesian parametrization accounts for imprecise location of the feature point. We have shown how the learned post-processing finds the optimal mixing ratio when combining the two parametrizations. We visualized the kernels that were learned and showed that its effect on patch similarity is semantically meaningful. We showed that it is beneficial to learn the whitening in a robust manner in the unsupervised case. Interestingly, the unsupervised variant generalizes better and is the best performing unsupervised hand-crafted descriptor so far. Despite the simplicity of the proposed descriptor, it competes well with deep learning approaches on a number of different tasks. The lessons learned from analyzing the similarity after whitening are applied for further improvements of local-patch descriptors in the succeeding chapter.

Explicitly Spatially Encoded Deep Local Descriptors

In this chapter, we extend the approach using multiple-kernel efficient match kernels to deep local descriptors. The descriptors previously introduced encoded simple gradients, which describe a very small neighbourhood. We propose to encode the appearance of larger neighbourhoods using CNNs, which are powerful in modeling the appearance variance, while weak in modeling the geometric displacement (at least with a single FC layer). We perform pooling of these features using efficient match kernels that explicitly encode the spatial positions of the responses. This can be seen as a transition from soft binning, *i.e.* overlapping receptive fields, to continuous efficient match kernels. In contrast to models with an FC layer, with efficient match kernels the number of model parameters does not grow with increased resolution of the input patch, *i.e.* the models for 32×32 patch input has the same number of parameters as the model for 64×64 . The proposed method consistently outperform other approaches at both resolutions, as we show by evaluation on the PhotoTourism and HPatches benchmarks. The applications of the proposed descriptor go beyond that of local-patches, *e.g.* tasks where encoding spatial position is essential [53, 70].

5.1 Deep local descriptors

Conventional architectures for deep local descriptors consist of a sequence of fully convolutional layers and a final FC layer. We denote the descriptor extraction process by function $\psi : \mathbb{R}^{N \times N} \rightarrow \mathbb{R}^D$, where N is the size of the input patch and D the dimensionality of the final descriptor. Descriptor for patch $\mathcal{P} \in \mathbb{R}^{N \times N}$ is given by $\psi(\mathcal{P}) \in \mathbb{R}^D$ or equivalently $\boldsymbol{\psi}_{\mathcal{P}}$ to simplify the notation.

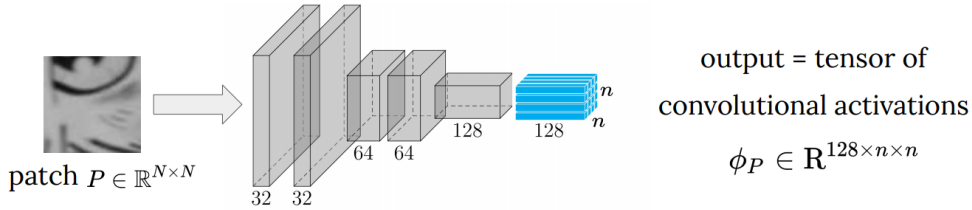


Figure 5.1: Architecture of the fully convolutional network (FCN) used in the proposed descriptor is shown. The input consists of a patch \mathcal{P} of size $N \times N$. The gray boxes represent the intermediate feature maps, which correspond to successively larger receptive fields. The final activations $\phi_{\mathcal{P}}$ (shown in blue) are the measurements which are encoded along with their position in the proposed descriptor.

We denote the convolutional part of the network, *i.e.* a *Fully Convolutional Network* (FCN), by function $\phi : \mathbb{R}^{N \times N} \rightarrow \mathbb{R}^{n \times n \times d}$. Size n of the resulting feature map is related to input size N and the architecture of the network. Feature map $\phi(\mathcal{P})$, equivalently denoted by $\phi_{\mathcal{P}}$, is a 3D tensor of activations, which we also view as a 2D grid of d -dimensional vectors. We call these vectors *convolutional descriptors* and use $\phi_{\mathcal{P}}^p$ to denote the vector with coordinates $p = (i, j)$ on the $n \times n$ grid, *i.e.* $p \in [n]^2$ ¹. Each convolutional descriptor corresponds to a region of the input patch \mathcal{P} that is equal to the receptive field size of the feature map. The 3D tensor, $\phi_{\mathcal{P}}$, for our architecture is illustrated in Figure 5.1.

The standard practice is to vectorize 3D tensor $\phi_{\mathcal{P}}$ and feed it to an FC layer with parameters that consist of matrix $W \in \mathbb{R}^{D \times (n \times n \times d)}$ and bias $\mathbf{w} \in \mathbb{R}^D$. The final descriptor is constructed as

$$\psi_{\mathcal{P}} = W \text{vec}(\phi_{\mathcal{P}}) + \mathbf{w}, \quad (5.1)$$

where vec denotes tensor vectorization. A local descriptor is typically ℓ_2 -normalized, which is equivalently achieved by introducing a normalization factor $\gamma_{\mathcal{P}} = 1/\sqrt{\psi_{\mathcal{P}}^{\top} \psi_{\mathcal{P}}}$ producing descriptor $\hat{\psi}_{\mathcal{P}} = \gamma_{\mathcal{P}} \psi_{\mathcal{P}}$.

Similarity (or distance) between patches \mathcal{P} and \mathcal{Q} is estimated with inner product (or Euclidean distance) $\hat{\psi}_{\mathcal{P}}^{\top} \hat{\psi}_{\mathcal{Q}}$. The ℓ_2 -normalized descriptor is always used to compare patches, but we often use $\psi_{\mathcal{P}}$ (and not $\hat{\psi}_{\mathcal{P}}$) simply to specify which descriptor variant is used. Several deep local descriptors in the recent literature, namely L2Net [93], HardNet [62], and GeoDesc [56] follow such an architecture and can be formulated in the same way. The architecture described is powerful in modeling the appearance variance, while weak in modeling the geometric displacement. In the following sections, we use cast deep local descriptors in the framework of match kernels. This allows us to model the geometric misalignment by efficient match kernels that explicitly encode the spatial positions of the responses.

¹ $[i] = \{1 \dots i\}$ and $[i]^2 = [i] \times [i]$

5.2 A match-kernel perspective

We provide an alternative, but equivalent, construction of deep local descriptors. We consider matrix W as a concatenation of n^2 matrices, *i.e.*

$$W = \begin{pmatrix} W_{(1,1)}^\top \\ \vdots \\ W_{(i,j)}^\top \\ \vdots \\ W_{(n,n)}^\top \end{pmatrix}^\top, \quad (5.2)$$

where $W_p \in \mathbb{R}^{D \times d}$. Descriptor in (5.1) can be now written as

$$\psi_p = \sum_{p \in [n]^2} W_p \phi_p^p + \mathbf{w}', \quad (5.3)$$

where $\mathbf{w}' = \mathbf{w}/n^2$. Moreover, patch similarity becomes

$$\begin{aligned} \hat{\psi}_p^\top \hat{\psi}_q &\propto \sum_{p,q \in [n]^2} (W_p \phi_p^p + \mathbf{w}')^\top (W_q \phi_q^q + \mathbf{w}') \\ &= \sum_{p,q \in [n]^2} g_{fc}(\phi_p^p, p)^\top g_{fc}(\phi_q^q, q), \end{aligned} \quad (5.4)$$

where $g_{fc} : \mathbb{R}^d \times [n]^2 \rightarrow \mathbb{R}^D$ is a function that encodes a convolutional descriptor in a translation variant way, depending on its position in the $n \times n$ grid. The match kernel formulation in (5.4) interprets deep local descriptor similarity as similarity accumulation for all pairs of positions on the $n \times n$ grid. It reveals that matching between convolutional descriptors in ϕ_a and ϕ_b is performed in a translation variant way. The *encoding function* g in the case of conventional deep local descriptors is

$$g_{fc}(\mathbf{v}, p) = W_p \mathbf{v} + \mathbf{w}', \quad (5.5)$$

where matrix W_p and \mathbf{w}' come from the parameters of the FC layer. Figure 5.2 illustrates this match-kernel perspective. We propose a new encoding function g , not restricted to standard CNN architecture (layers), that explicitly encodes position p on the 2D grid.

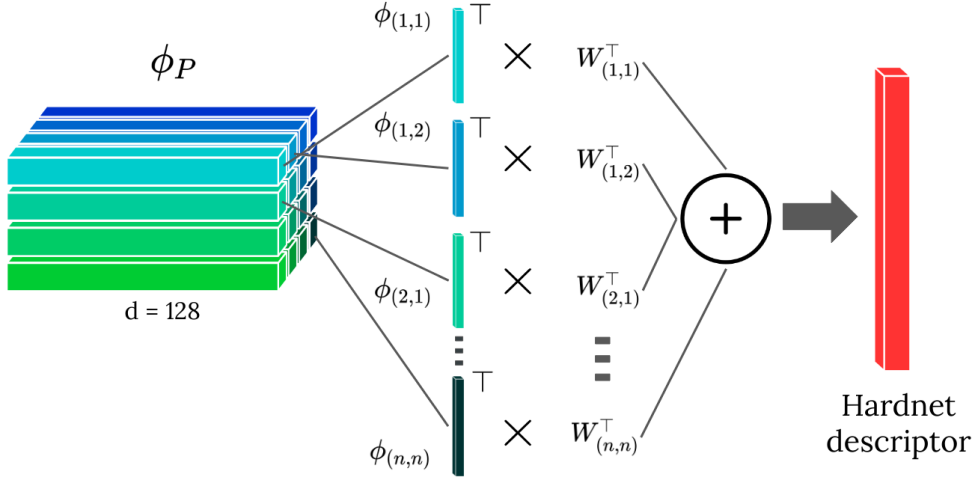


Figure 5.2: A match-kernel perspective of typical deep local descriptors such as HardNet [62] is illustrated. ϕ_P is a tensor of convolutional activations, and W is the matrix representing the weights of the Fully Connected (FC) layer. The matrix W is interpreted as a concatenation of n^2 matrices, *i.e.* $W_p \forall p \in [n]^2$. The feature map at each position p is transformed by W_p and aggregated.

Position encoding. We use explicit feature maps [101], described in Section 4.2 to encode the position. Let $f : \mathbb{R} \rightarrow \mathbb{R}^{2s+1}$ be a feature map, where s is a design choice defining the dimensionality of the embedding. Such a feature map defines a shift invariant kernel $K : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ with kernel signature k , so that $K(\alpha, \beta) = k(\alpha - \beta)$

$$f(\alpha)^\top f(\beta) = K(\alpha, \beta) = k(\alpha - \beta). \tag{5.6}$$

The kernel K (or the feature map f) is constructed to approximate the Von Mises kernel [96].

We propose encoding function $g_{xy} : \mathbb{R}^d \times [n]^2 \rightarrow \mathbb{R}^{D(2s+1)^2}$ given by

$$g_{xy}(\phi_P^p, p) = \phi_P^p \otimes f(x_p) \otimes f(y_p), \tag{5.7}$$

where \otimes is the Kronecker product and x_p and y_p provide the coordinates of position p in a Cartesian coordinate system ². It is a joint encoding of the convolutional descriptor and the explicit representation of its position. Similarity of two such encodings is given by

$$g_{xy}(\phi_P^p, p)^\top g_{xy}(\phi_Q^q, q) = \phi_P^p \phi_Q^q \cdot k(x_p - x_q) \cdot k(y_p - y_q). \tag{5.8}$$

It is equivalent to the product of descriptor similarity and similarity of positions on the Cartesian grid.

²For $p = (i, j)$, $x_p = i$ and $y_p = j$.

Following the paradigm of descriptor whitening of hand-crafted descriptors [69, 13], we propose the final local descriptor

$$\boldsymbol{\psi}_{\mathcal{P}}^{xy} = \sum_{p \in [n]^2} w_p M_{xy} g_{xy}(\boldsymbol{\phi}_{\mathcal{P}}^p, p) + \mathbf{m}_{xy} \quad (5.9)$$

$$= M_{xy} \left(\sum_{p \in [n]^2} w_p g_{xy}(\boldsymbol{\phi}_{\mathcal{P}}^p, p) \right) + n^2 \mathbf{m}_{xy}, \quad (5.10)$$

where $M_{xy} \in \mathbb{R}^{D \times d(2s+1)^2}$ and $\mathbf{m}_{xy} \in \mathbb{R}^D$ are parameters to be learned during training, while $w_p = \exp(-\rho_p^2)$ is a weight giving importance according to the distance ρ_p from the center of the patch. Note that in contrast to (5.3) the same matrix, *i.e.* M_{xy} , is used for all convolutional descriptors. As a result the number of required parameters is reduced and multiplication by M_{xy} can be efficiently performed after the summation (5.9). In analogy to the encoding of position in a Cartesian coordinate system, we additionally propose the encoding *w.r.t.* a polar coordinate system³ by

$$g_{\rho\theta}(\boldsymbol{\phi}_{\mathcal{P}}^p, p) = \boldsymbol{\phi}_{\mathcal{P}}^p \otimes f(\rho_p) \otimes f(\theta_p), \quad (5.11)$$

and the corresponding descriptor

$$\boldsymbol{\psi}_{\mathcal{P}}^{\rho\theta} = \sum_{p \in [n]^2} w_p M_{\rho\theta} g_{\rho\theta}(\boldsymbol{\phi}_{\mathcal{P}}^p, p) + \mathbf{m}_{\rho\theta}. \quad (5.12)$$

Different parameterizations, *i.e.* using different coordinate system, provide tolerance to different kinds of misalignment between patches. Cartesian offers tolerance to translation misalignment, while polar offers tolerance to rotation and scale misalignment. To benefit from both types of tolerance, we further use the combined encoding that uses the two coordinate systems and is produced by concatenation of the previous encoding. It is defined as function $g_c : \mathbb{R}^d \times \mathbb{R}^d \times [n]^2 \rightarrow \mathbb{R}^{2D(2s+1)^2}$ given by

$$g_c(\boldsymbol{\phi}_{\mathcal{P}}^p, \tilde{\boldsymbol{\phi}}_{\mathcal{P}}^p, p) = \left((\boldsymbol{\phi}_{\mathcal{P}}^p \otimes f(x_p) \otimes f(y_p))^\top, (\tilde{\boldsymbol{\phi}}_{\mathcal{P}}^p \otimes f(\rho_p) \otimes f(\theta_p))^\top \right)^\top \quad (5.13)$$

where $\tilde{\boldsymbol{\phi}}$ is used to show that the two encodings do not need to rely on the same FCN $\boldsymbol{\phi}$. Subscript c refers to the combined coordinate system, but we skip $xy\rho\theta$ to simplify the notation. The final descriptor proposed is

$$\star \boldsymbol{\psi}_{\mathcal{P}}^c = \sum_{p \in [n]^2} w_p M_c g_c(\boldsymbol{\phi}_{\mathcal{P}}^p, \tilde{\boldsymbol{\phi}}_{\mathcal{P}}^p, p) + \mathbf{m}_c \quad (5.14)$$

where $M_c \in \mathbb{R}^{D \times 2d(2s+1)^2}$, and left superscript \star is used to denote that a separate FCN is used for each encoding, correspondingly coordinate system.

³For $p = (i, j)$, $\rho_p = \sqrt{(i-c)^2 + (j-c)^2}$ and $\theta_p = \tan^{-1} \frac{j-c}{i-c}$, where $c = (n+1)/2$.

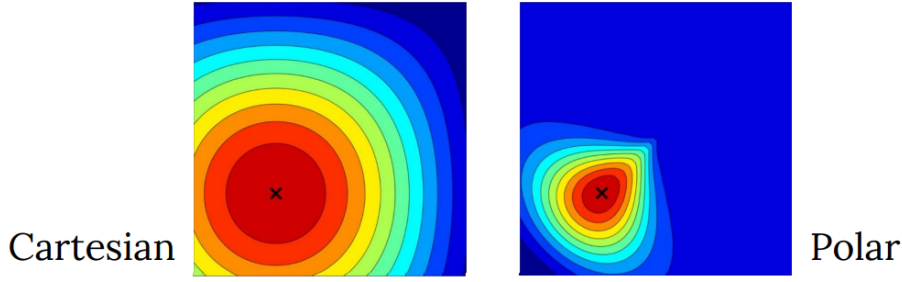


Figure 5.3: Patch maps for different parametrizations and kernels. Two parametrizations in polar and Cartesian coordinates are presented. The position p of feature map is shown with “ \times ” on the patch maps. Observe that the Cartesian kernel is tolerant to translation misalignment, while the polar kernel is tolerant to rotational misalignment. Ten isocontours are sampled uniformly and shown in different color. Red indicates maximum similarity and blue indicates minimum similarity.

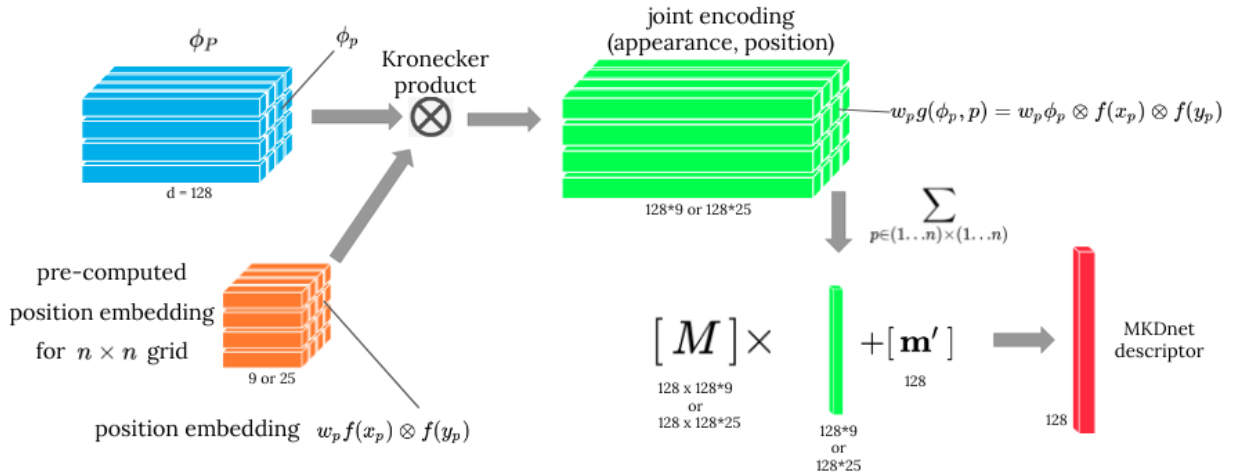


Figure 5.4: The match-kernel perspective of our proposed deep local descriptor is shown. ϕ_P is a tensor of convolutional activations, and $w_p f(x_p) \otimes f(y_p)$ is the matrix representing the pre-computed position embedding. The Kronecker product of the activations and the embedding is shown as $w_p g(\phi_p, p)$. Similarity is decomposed into influence of appearance and position $w_p g(\phi_p, p)^\top w_q g(\phi_q, q) = w_p w_q \phi_p^\top \phi_q \cdot k(x_p - x_q) \cdot k(y_p - y_q)$. The feature map at each position p is transformed by the same whitening transform (\mathbf{M}, \mathbf{m}') and aggregated to give the descriptor $\psi(\mathcal{P})$. Comparison of a pair of patches therefore is equivalent to comparison of transformed feature maps at each position, allowing a match-kernel interpretation.

5.3 Implementation details

In this section, we provide implementation details that concern the efficiency of the aggregation, describe the different architectures and their required number of parameters, and finally discuss the training procedure.

Efficient aggregation. We describe the implementation details for variant $\psi_{\mathcal{P}}^{xy}$, but these hold for other variants too in the same way. Vectors $w_p f(x_p) \otimes f(y_p) \in \mathbb{R}^{(2s+1)^2}$ that encode positions $p \in [n]^2$ are fixed for the 2D grid of size $n \times n$. Thus, we pre-compute and store them in matrix $F \in \mathbb{R}^{n^2 \times (2s+1)^2}$. We reshape 3D tensor ϕ_a into matrix $\Phi \in \mathbb{R}^{n^2 \times d}$. Given these two matrices and due to the linearity of matrix to vector multiplication we can re-write the descriptor as

$$\begin{aligned} \psi_{\mathcal{P}}^{xy} &= \sum_{p \in [n]^2} w_p M_{xy} g_{xy}(\phi_{\mathcal{P}}^p, p) + \mathbf{m}_{xy}, \\ &= M_{xy} \left(\sum_{p \in [n]^2} w_p g_{xy}(\phi_{\mathcal{P}}^p, p) \right) + n^2 \mathbf{m}_{xy}, \end{aligned} \quad (5.15)$$

$$\begin{aligned} &= M_{xy} \left(\sum_{p \in [n]^2} w_p \phi_{\mathcal{P}}^p \otimes f(x_p) \otimes f(y_p) \right) + n^2 \mathbf{m}_{xy}, \\ &= M_{xy} \text{vec}(\Phi^{\top} F) + n^2 \mathbf{m}_{xy}. \end{aligned} \quad (5.16)$$

Multiplication $\Phi^{\top} F$ makes the computation memory efficient because it avoids explicit storing of the Kronecker product for each p . To evaluate (5.15), the memory requirements are $n^2 d (2s+1)^2$ numbers, while to evaluate (5.16), only $n^2 (d + (2s+1)^2)$ numbers are allocated. Using setup $d = 128$ and $s = 2$ in our experiments, the memory requirements are reduced by a factor of 20.9.

Architecture. We use the HardNet+ [62] architecture for the convolution part, since HardNet+ achieves state-of-the-art performance on all benchmarks. We also use it a baseline to compare with.

The statistics of the convolutional part ϕ are described in Table 5.1 (left). Each convolutional layer is followed by batch normalization and ReLU, while no bias is used. Table 5.1 (right) provides the total number of parameters for HardNet+ and our networks, namely, plain polar or Cartesian encoding with different dimensionality of the explicit feature maps ($s = 1$ and $s = 2$ frequencies used), and the joint encoding with a common (ϕ) or separate (ϕ and $\tilde{\phi}$) convolutional part. Note that for the joint encoding with separate convolutional parts and $s = 2$ frequencies, the proposed network needs roughly the same number of parameters as HardNet+ with input patch of size 32×32 pixels ($N = 32$). In all other settings of the proposed architecture, the number of parameters is significantly

reduced. Importantly, the number of parameters for larger patch sizes (such as 64×64), that provide better performance, the number of parameters stays fixed for the proposed architecture. For Hardnet+, the number of parameters of the FC layer increases by a factor of 4 for 64×64 input patches.

			Convolutional part ϕ	
	Conv. layer	Param. matrix shape	# Parameters	
	1	[1, 32, 3, 3]	288	
	2	[32, 32, 3, 3]	9,216	
	3	[32, 64, 3, 3]	18,432	
	4	[64, 64, 3, 3]	36,864	
	5	[64, 128, 3, 3]	73,728	
	6	[128, 128, 3, 3]	147,456	
	Total		285,984	

HardNet	$N = \{32, 64\}$		ψ^{xy}	
	$N = 32$	$N = 64$	$N = \{32, 64\}$	
			$s=1$	$s=2$
ϕ	285,984	285,984	ϕ	285,984 285,984
FC	1,048,576	4,194,304	M, \mathbf{m}	147,584 409,728
Total	1,334,560	4,480,288	Total	433,568 695,712

ψ^c	$N = \{32, 64\}$		$^*\psi^c$	
	$s=1$	$s=2$	$N = \{32, 64\}$	
			$s=1$	$s=2$
ϕ	285,984	285,984	ϕ	285,984 285,984
$\tilde{\phi}$	285,984	285,984	$\tilde{\phi}$	285,984 285,984
M, \mathbf{m}	295,040	819,328	M, \mathbf{m}	295,040 819,328
Total	581,024	1,105,312	Total	867,008 1,391,296

Table 5.1: Number of parameters for different models. The convolutional part ϕ has identical architecture for all models. Cases where both ϕ and $\tilde{\phi}$ appear use a separate convolutional part for the Cartesian and the polar descriptor. These specifications correspond to $d = 128$, and $D = 128$. The resulting n is equal to 8 and 16 for N equal to 32 and 64, respectively. Descriptor $\psi^{\rho\theta}$ has identical requirements as descriptor ψ^{xy} . The parameter requirements of our descriptor remain unchanged for different patch size N .

Training. We would like to highlight the contribution of the explicit spatial encoding and to provide direct comparison to the current state-of-the-art descriptor construction. To avoid changing many things at the same time, we follow exactly the same training procedure as HardNet+, which we briefly review below.

The network is trained with the triplet loss defined as

$$\ell(\hat{\psi}_{an}, \hat{\psi}_{pos}, \hat{\psi}_{neg}) = [1 - \|\hat{\psi}_{an} - \hat{\psi}_{pos}\| + \|\hat{\psi}_{an} - \hat{\psi}_{neg}\|]_+, \quad (5.17)$$

acting on a triplet formed by an anchor, a positive (matching to the anchor), and a negative (non-matching to the anchor) descriptor. A batch of size 1024 patches is constructed from 512 pairs of anchor-positive descriptors. Regarding a particular pair in the batch, the positive descriptors of all other pairs are considered as candidate negatives. Finally, the one with the smallest Euclidean distance to the anchor within the batch is chosen as a hard negative to form a triplet.

We use Stochastic Gradient Descent (SGD) to perform the training. The total training set consists of 2 million anchor-positive pairs and the training lasts 10 epochs. The learning rate is set to 10, and linearly decays to zero withing 10 epochs. Momentum is equal to 0.9 and weight decay to 10^{-4} . Random orthogonal initialization is used for the weights of the network [81]. The method is implemented in the PyTorch framework.

Visualization of kernels. We construct encodings $g(\mathbf{v}, p)$, before aggregation, for our descriptors and for the conventional case and construct a similarity map to analyze the impact of the position encoding. We present such visualization in Figure 5.5. We pick a position p and compute similarity

$$M(q|p, g_{fc}) = g_{fc}(\phi_{\mathcal{P}}^p, p)^\top g_{fc}(\phi_{\mathcal{Q}}^q, q), \forall q \in [n]^2,$$

for the conventional case, and

$$M(q|p, g_c) = (M_c g_c(\phi_{\mathcal{P}}^p, p) + \mathbf{m}_c/n^2)^\top (M_c g_c(\phi_{\mathcal{Q}}^q, q) + \mathbf{m}_c/n^2), \forall q \in [n]^2,$$

for ours in the case of the combined descriptor, where $\mathcal{P} = \mathcal{Q}$. We observe how all architectures, including the conventional one, result in large similarity values near p .

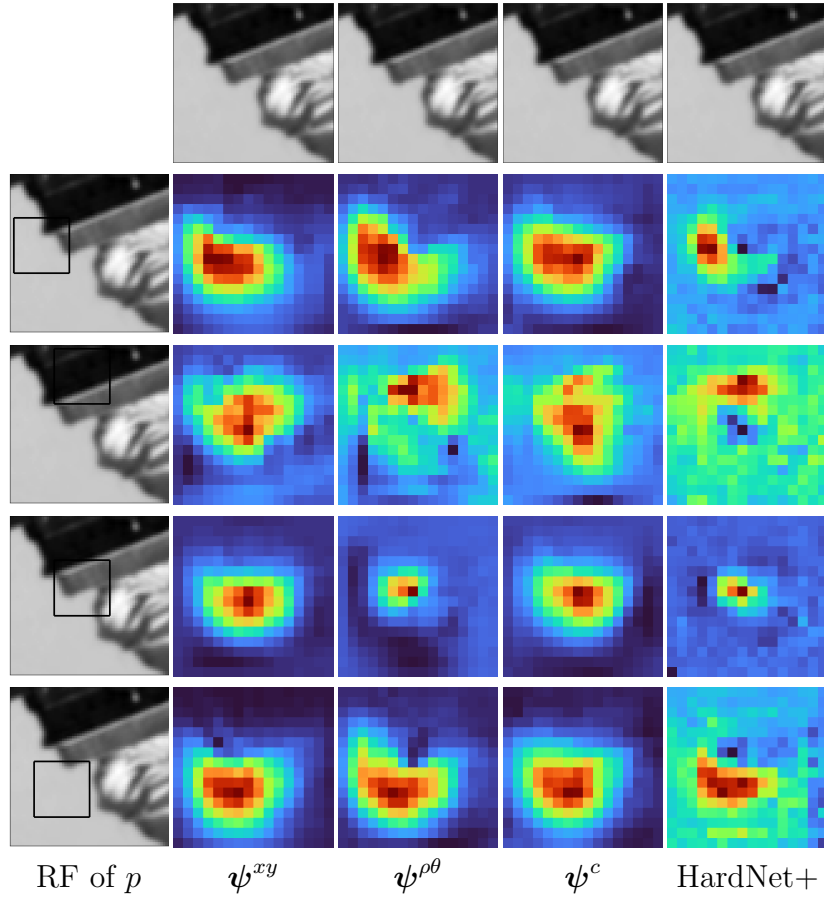


Figure 5.5: Visualization of similarity between a position p of the $n \times n$ grid (rows) on a patch and the whole patch itself for different methods (columns). Heat-maps are normalized to $[0, 1]$ with red corresponding to the maximum similarity. Red box is used to depict the receptive field (RF) of p .

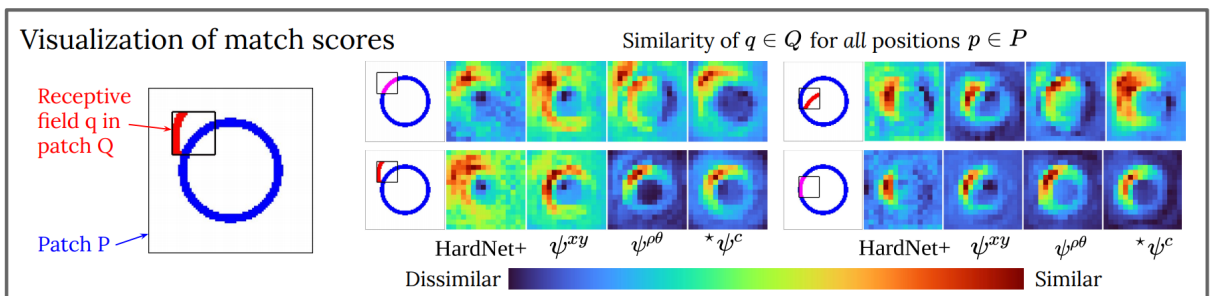


Figure 5.6: The visualization of similarity in the case of a synthetic patch is shown, to illustrate the effect of appearance and spatial dependence. Similarity is measured between a position p of the $n \times n$ grid (rows) on a patch and the whole patch itself for different methods (columns). Heat-maps are normalized to $[0, 1]$ with red corresponding to the maximum similarity. Red is used to depict the appearance of q , and its position in the patch depicts the spatial position encoded.

5.4 Experiments

We first describe the datasets used in our experiments and the evaluation protocols used, then present qualitative results showing the impact of the training on patch similarity, and finally present the results achieved by different variants of our descriptor and show a comparison with the state of the art.

Datasets and protocols. We use the two publicly available patch datasets described in Chapter 3, namely *PhotoTourism* (PT) [103] and *HPatches* (HP) [13]. We use the former for both training and evaluation, while the latter only for evaluation when training on PT to show the generalization ability of the descriptor.

The PT dataset consists of following 3 separate sets, Liberty, Notredame and Yosemite. Each consists of local features detected with the Difference-of-Gaussians (DoG) detector and verified through an SfM pipeline. Each set comprises about half a million 64×64 patches, associated with a discrete label which is the outcome of SfM verification. The test set consists of 100k pairs of patches corresponding to the same (positive) 3D point, and an equal number corresponding to different (negative) 3D points. The metric used to measure performance is the *false positive rate at 95% of recall* (FPR@95). Models are trained on one set and tested on the other two, and the mean of 6 scores is reported.

The HP dataset contains patches of higher diversity and is more realistic. Evaluation is performed on three different tasks, namely verification, retrieval, and matching. Despite the fact that we do not train on HP, we evaluate on all 3 train/test splits and report the average performance to allow future comparisons. We follow the common practice and train our descriptor on Liberty of PT to evaluate on HP.

We repeat each experiment three times, with different random seeds to initialize the parameters, and report mean and standard deviation of the 3 runs. We followed this policy for all variants and datasets.

Recently, larger and more diverse datasets [65, 56] have been introduced to improve local descriptor training. These are shown to improve the performance of state-of-the-art descriptors even by simply replacing the training dataset. We have not included them in our experiments but expect the impact to be similar on our descriptor too.

We train and evaluate different variants of the proposed descriptor. If not otherwise stated, we use input patches of size equal to 32×32 , which is the standard practice for deep local descriptors. We further examine the case of 64×64 input patches. We always set $d = 128$ and $D = 128$. The dimensionality of the feature maps is controlled by s which we set equal to 1 or 2 in our experiments.

Reproducing HardNet+. Our implementation, training procedure, and training hyper-parameters are based on HardNet+⁴. We reproduce its training and report our own results, proving that our benefit is not an outcome of implementation details. We report both the achieved performance in the original publication and our reproduced ones in all the comparisons.

⁴<https://github.com/DagnyT/hardnet>

Baselines for ablation study. We train and test the following two baselines to see the impact of the position encoding. First, we train a descriptor that encodes convolutional descriptors in $\phi_{\mathcal{P}}$ in a translation invariant way, *i.e.* no position encoding at all. It is implemented by spatial sum pooling on $\phi_{\mathcal{P}}$ and given by

$$\psi_{\mathcal{P}}^{\text{sum}} = \sum_{p \in [n]^2} \psi_{\mathcal{P}}^p. \quad (5.18)$$

The dimensionality of ψ_a^{sum} is equal to d and not D in this case. However, $d = D = 128$, making this descriptor directly comparable to all others.

Second, we train a descriptor that encodes the spatial information simply by concatenation, *i.e.* vectorization of $\phi_{\mathcal{P}}$, which does not provide any tolerance to position misalignments. It is given by

$$\psi_{\mathcal{P}}^{\text{cat}} = \text{vec } \psi_{\mathcal{P}} \quad (5.19)$$

Impact of position encoding. We compare our descriptor with HardNet+ on PT and show results in Table 5.2. Conceptually it is a comparison between the conventional architecture that uses an FC layer to “feed” the convolutional descriptors to, and our kernel-based approach to explicitly encode the spatial information. Our descriptors (with $s = 2$) slightly outperforms HardNet+ while it has roughly the same number of parameters. Even the variant with fewer parameters ($s = 1$) performs similarly.

A more thorough comparison, examining the impact of the explicit spatial encoding, is performed on HP and presented in Figure 5.7. Firstly, we evaluate ψ^{SUM} as part of an ablation study. It is translation invariant that totally discards the spatial information. It does not require additional parameters other than the ones for FCN ϕ . It has significantly lower performance compared to all the other descriptors. We additionally tried including multiplication by matrix M_{sum} in (5.18) and did not notice performance improvements. Descriptor ψ^{CAT} is another case not requiring additional parameters. It is translation variant in a “rigid” way, whose tolerance to translation misalignment is restricted to the amount that the large receptive field offers. Despite the very large dimensionality, it is not a top performer. Even our light-weight variant with as few as 127k additional parameters (excluding ϕ) recovers most of the performance loss due to lack of spatial information, *i.e. w.r.t.* ψ^{SUM} . This result suggests that the common choice of an FC layer for deep local descriptors might be over-parametrized. It is not the best performing either. Our variant ψ^{c2} is consistently the top performing one on all tasks.

Test			Liberty		Notredame		Yosemite	
Train	Params	Mean	No	Yo	Li	Yo	Li	No
HardNet+ †	1,334,560	1.51	1.49	2.51	0.53	0.78	1.96	1.84
HardNet+	1,334,560	1.43 ± 0.02	1.25 ± 0.03	2.35 ± 0.03	0.48 ± 0.01	0.74 ± 0.02	2.15 ± 0.01	1.61 ± 0.10
* ψ^{c1}	867,008	1.53 ± 0.03	1.27 ± 0.03	2.31 ± 0.08	0.48 ± 0.02	0.82 ± 0.05	2.58 ± 0.08	1.72 ± 0.09
* ψ^{c2}	1,391,296	1.36 ± 0.01	1.14 ± 0.03	2.16 ± 0.10	0.42 ± 0.01	0.73 ± 0.02	2.18 ± 0.07	1.51 ± 0.12

Table 5.2: Performance comparison of the proposed descriptors with the state-of-the-art descriptor HardNet+ on the PhotoTourism dataset. Performance is measured via FPR@95. We repeat each experiment/training 3 times and report mean performance and standard deviation. Patch size is $N = 32$. †: Reported in the original work.

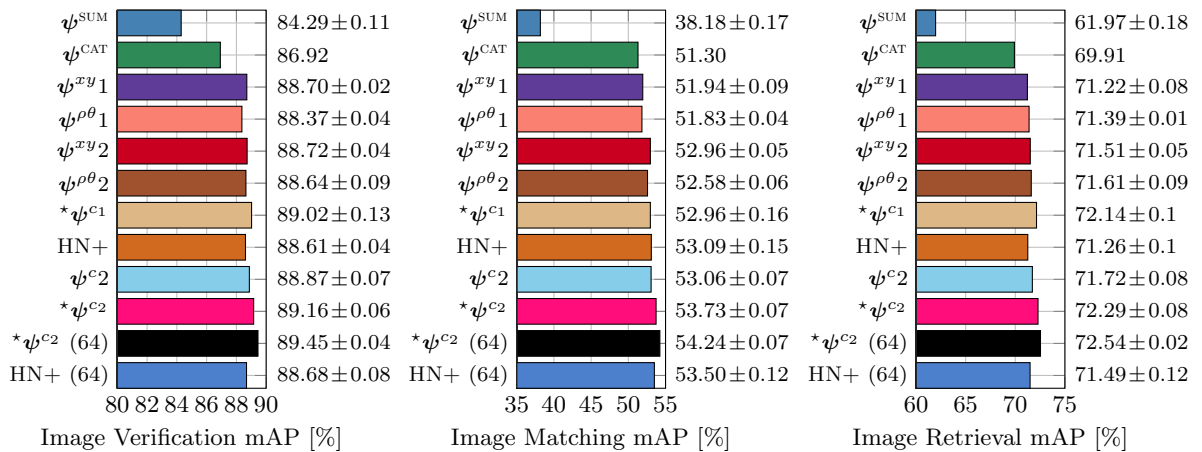


Figure 5.7: Performance comparison on the HPatches benchmark. The training is performed on the Liberty set of PhotoTourism dataset for all descriptors and with identical setup. Performance is measured via mean Average Precision (mAP). We repeat each experiment/training 3 times and report mean performance and standard deviation (with the exception of ψ^{CAT} that due to very high dimensionality was trained only once). All descriptors have 128 dimensions, with the exception of ψ^{CAT} which has 8192. The methods are sorted *w.r.t.* the required number of parameters (top is the least demanding, *i.e.* less parameters). All methods are trained and tested with patch size $N = 32$ unless when (64) is reported.

Comparison with the state of the art. We finally present a comparison to the state of the art on HP in Figure 5.8. The comparison includes a set of hand-crafted and learned local descriptors, namely RSIFT [11], SIFT [54], BRIEF [25], BBoost [98], ORB [80],

MKD [69], DeepCompare [106], DDesc [87], TFeat [14], L2Net [93] and HardNet [62]. The proposed descriptor achieves the best performance with a 128D descriptor on all 3 tasks consistently.

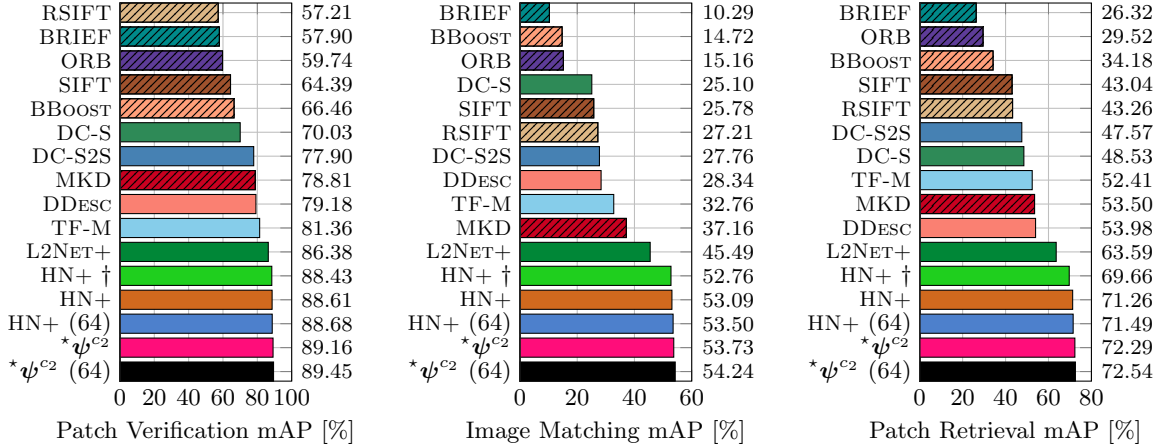


Figure 5.8: Performance comparison with the state of the art on the HPatches benchmark. The learning for learned descriptors is performed on the Liberty set of PhotoTourism dataset. Hand-crafted descriptors are shown with striped bars. Performance is measured via mean Average Precision (mAP). The performance of our descriptor is the mean of 3 repetitions of each experiment/training. All methods are trained and tested with patch size $N = 32$ unless when (64) is reported. †: Reported in the original work.

5.5 Discussion

In this chapter, we proposed a kernelized deep local descriptor based on efficient match kernels of neural network activations. We showed that the FC layer can be interpreted as encoding spatial relations between the activations of the CNN, and cast conventional convolutional local descriptors in the framework efficient match kernels. We build upon the descriptor proposed in Chapter 4 by moving from intensity gradient measurements to complex multilayer convolutional network activations. We showed that replacing the FC layer by explicit spatial encoding as introduced for the hand-crafted descriptor reduces the number of parameters and makes it independent of the patch resolution. We visualized the similarity maps for conventional and proposed architectures, and showed the effect of appearance and spatial dependence. The applications of our approach goes beyond that of local-patches, *e.g.* tasks where spatial encoding is essential.

Robust Data Whitening

In this chapter, we propose a robust estimator of the whitening transformation in the presence of outliers. Local descriptors are high-dimensional vectors, and their entries are often correlated, which leads to a bias in similarity estimation. To remove the correlation, a linear transformation, called whitening, is commonly used. We have described the whitening procedure used for the local descriptors introduced in Chapter 4 and Chapter 5. However, in the presence of outliers, this estimation is inaccurate, leading to a suboptimal solution. We propose a novel algorithm, inspired by the Iteratively Re-weighted Least Squares (IRLS) approach, that iterates between centering and applying a transformation matrix, a process which is shown to converge for robust class of cost functions [8]. The approach is developed for unsupervised scenarios, and extended to supervised cases. We demonstrate the robustness of our method to outliers on synthetic 2D data and also show improvements compared to conventional whitening on real data for image retrieval with CNN-based representation. Finally, our robust estimation is not limited to data whitening, but can be used for robust patch rectification, *e.g.* with MSER features. In the following sections, we first briefly review the background of data whitening and then give a geometric interpretation, which forms our motivation for the proposed approach.

6.1 Background on whitening

A whitening transformation is a linear transformation that transforms a vector of random variables with a known covariance matrix into a set of new variables whose covariance is the identity matrix. The transformation is called “whitening” because it changes the input vector into a white noise vector.

We consider the case where this transformation is applied on a set of zero centered vectors $\mathcal{X} = \{\mathbf{e}[x]_1, \dots, \mathbf{e}[x]_i, \dots, \mathbf{e}[x]_N\}$, with $\mathbf{e}[x]_i \in \mathbb{R}^d$, where $\Sigma = \sum_i \mathbf{e}[x]_i \mathbf{e}[x]_i^\top$. The whitening transformation P is given by

$$P^\top P = \Sigma^{-1}. \quad (6.1)$$

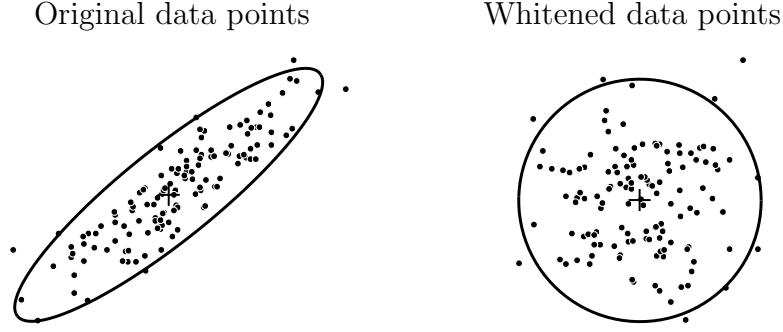


Figure 6.1: Left: Points in 2D and their covariance shown with an ellipse. Right: The corresponding whitened 2D point set.

In Figure 6.1 we show a toy example of 2D points and their whitened counterpart.

Assumption. In the following text, we assume that the points of \mathcal{X} do *not* lie in a linear subspace of dimensionality $d' < d$. If this is the case, a solution is to first identify the d' -dimensional subspace and perform the proposed algorithms on this subspace. The direct consequence of the assumption is that the sample covariance matrix Σ is full rank, in particular $\det(\Sigma) > 0$.

It is clear from (6.1) that the whitening transformation is given up to an arbitrary rotation $R \in \mathbb{R}^{d \times d}$, with $R^\top R = I$. The transformation matrix P of the whitening is thus given by

$$P = R\Sigma^{-1/2}. \quad (6.2)$$

Geometric interpretation. Assuming zero-mean points, the whitening transform P in equation 6.2 minimizes the sum of squared ℓ_2 norms among all linear transforms T with $\det(T) = \det(\Sigma)^{-1/2}$.

$$C_{\ell_2}(P) = \sum_i \|Pe[x]_i\|^2 \quad (6.3)$$

$$= \sum_i \text{tr}(\mathbf{e}[x]_i^\top P^\top Pe[x]_i) \quad (6.4)$$

$$= \sum_i \text{tr}((\mathbf{e}[x]_i \mathbf{e}[x]_i^\top) P^\top P) \quad (6.5)$$

$$= \text{tr} \left(\left(\sum_i \mathbf{e}[x]_i \mathbf{e}[x]_i^\top \right) P^\top P \right) \quad (6.6)$$

$$= \text{tr}(\Sigma P^\top P) \quad (6.7)$$

$$= \sum_{j=1}^d \lambda_j, \quad (6.8)$$

where λ_i are the eigenvalues of $\Sigma P^\top P$ and $\|\cdot\|$ is denoting ℓ_2 norm.

Upon imposing the condition $\det(T) = \det(\Sigma)^{-1/2}$, we get that $\det(\Sigma P^\top P) = \prod_{j=1}^d \lambda_j$ is constant with respect to P . It follows from the arithmetic and geometric mean inequality, that the sum in (6.3) is minimized when $\lambda_i = \lambda_j, \forall i = j$. Equality of all eigenvalues allows us to show that

$$\Sigma P^\top P = I \quad (6.9)$$

$$P^\top P = \Sigma^{-1} \quad (6.10)$$

$$P = R\Sigma^{-1/2} \quad (6.11)$$

which is exactly the solution in (6.2) that also minimizes (6.3). The need for the existence of Σ^{-1} justifies the stated full rank assumption.

6.2 Background on IRLS

In the context of distance minimization the IRLS method minimizes the cost function

$$C_h(\mathbf{e}[\theta]) = \sum_{i=1}^N h \circ f(\mathbf{e}[\theta], \mathbf{e}[x]_i), \quad (6.12)$$

where f is a distance function that is defined on some domain, h is a function that makes the cost less sensitive to outliers, and $\mathbf{e}[x]_i \in \mathcal{X}$. Some examples of robust h functions are ℓ_1 , Huber, pseudo-Huber, *etc.* as described in prior literature [8]. For instance, assume the case of the geometric median of the points in \mathcal{X} . Setting $f(\boldsymbol{\mu}, \mathbf{e}[x]_i) = \|\boldsymbol{\mu} - \mathbf{e}[x]_i\|$ and $h(z) = z$, we get the cost (6.12) as the sum of ℓ_2 norms. The minimum of this cost is attained when $\boldsymbol{\mu}$ is equal to the geometric median.

It is shown [8] that a solution for $\operatorname{argmin}_{\mathbf{e}[\theta]} C_h(\mathbf{e}[\theta])$ may be found by solving a sequence of weighted least squares problems. Given some initial estimate $\mathbf{e}[\theta]^0$, the parameters $\mathbf{e}[\theta]$ are iteratively estimated

$$\mathbf{e}[\theta]^{t+1} = \operatorname{argmin}_{\mathbf{e}[\theta]} \sum_{i=1}^N w(\mathbf{e}[\theta]^t, \mathbf{e}[x]_i) f(\mathbf{e}[\theta], \mathbf{e}[x]_i)^2, \quad (6.13)$$

where for brevity $w(\mathbf{e}[\theta]^t, \mathbf{e}[x]_i)$ is denoted w_i^t in the following. Provided $h(\sqrt{z})$ is differentiable at all points and concave, for certain values of w_i^t and conditions on f this solution minimizes $C_h(\mathbf{e}[\theta])$. In some cases, it may even be possible to find a simple and analytic solution.

Given that the iterative procedure indeed converges to a minimum cost of (6.12), we get the following condition on the weights:

$$\nabla_{\mathbf{e}[\theta]}(h \circ f(\mathbf{e}[\theta], \mathbf{e}[x]_i)) = 0, \quad (6.14)$$

$$\nabla_{\mathbf{e}[\theta]}(w_i^t f(\mathbf{e}[\theta], \mathbf{e}[x]_i)^2) = 0. \quad (6.15)$$

This results in the following weights

$$w_i^t = \frac{h'(f(\mathbf{e}[\theta]^t, \mathbf{e}[x]_i))}{2f(\mathbf{e}[\theta]^t, \mathbf{e}[x]_i)}. \quad (6.16)$$

6.3 Robust transformation estimation

From the observation in Section 6.2, we know that there is a closed-form solution to the problem of finding a linear transformation P so that $\sum_i \|P\mathbf{e}_i\|^2$ is minimized subject to a fixed determinant $\det(P)$. The idea of the robust whitening is to use this least squares minimizer in a framework similar to the iterative re-weighted least squares to minimize a robust cost.

In contrast to the conventional whitening and the minimization of (6.3), we now propose the estimation of a whitening transform (transformation matrix P) in a way that is robust to outliers. We assume zero mean points and seek the whitening transformation that minimizes the robust cost function of (6.12). We set $f(P, \mathbf{e}[x]_i) = \|P\mathbf{e}[x]_i\|$ and use the ℓ_1 cost function $h(z) = z$. Other robust cost functions can be used, too¹.

We seek to minimize the sum of ℓ_2 norms in the whitened space

$$C_{\ell_1}(P) = \sum_{i=1}^N f(P, \mathbf{e}[x]_i) = \sum_{i=1}^N \|P\mathbf{e}[x]_i\|. \quad (6.17)$$

The corresponding iteratively re-weighted least squares solution is given by

$$P^{t+1} = \operatorname{argmin}_P \sum_{i=1}^N w_i^t \|P\mathbf{e}[y]_i^t\|^2, \quad (6.18)$$

where $\mathbf{e}[y]_i^t = P^t \mathbf{e}[y]_i^{t-1}$ and $\mathbf{e}[y]_i^0 = \mathbf{e}[x]_i$. This means that each time transformation P^t is estimated and applied to whiten the data points. In the following iteration, the estimation is performed on data points in the whitened space. The effective transformation at iteration t with respect to the initial points $\mathbf{e}[x]_i$ is given by

$$\hat{P}^t = \prod_{i=1}^t P^i. \quad (6.19)$$

¹We also use Cauchy cost in our experiments. It is defined as $h(z) = b^2 \log(1 + z^2/b^2)$.

Along the lines of proof (6.3) we find a closed form solution that minimizes (6.17) as

$$\sum_i w_i^t \|P\mathbf{e}[y]_i^t\|^2 \quad (6.20)$$

$$= \text{tr} \left(\left(\sum_i w_i^t \mathbf{e}[y]_i^t \mathbf{e}[y]_i^{t\top} \right) P^\top P \right) \quad (6.21)$$

$$= \text{tr} \left(\tilde{\Sigma} P^\top P \right) \quad (6.22)$$

$$(6.23)$$

where $\tilde{\Sigma} = \sum_i w_i^t \mathbf{e}[y]_i^t \mathbf{e}[y]_i^{t\top}$ is a *weighted covariance*. Therefore, P is given, up to a rotation, as

$$P = R\tilde{\Sigma}^{-1/2}. \quad (6.24)$$

6.4 Joint centering and transformation matrix estimation.

In this section we describe the proposed approach for data whitening. We propose to jointly estimate a robust mean $\boldsymbol{\mu}$ and a robust transformation matrix P by alternating between the two previously described procedures: estimating the geometric median and estimating the robust transformation. In other words, in each iteration, we first find $\boldsymbol{\mu}$ keeping P fixed and then find P keeping $\boldsymbol{\mu}$ fixed. In this way the assumption for centered points when finding P is satisfied. Given that each iteration of the method outlined above reduces the cost, and that the cost must be non-negative, we are assured convergence to a local minimum. We propose to minimize cost

$$C_{\ell_1}(P, \boldsymbol{\mu}) = \sum_{i=1}^N \|P(\mathbf{e}[x]_i - \boldsymbol{\mu})\|. \quad (6.25)$$

In order to reformulate this as an IRLS problem, we use $h(z) = z$, and $f(P, \boldsymbol{\mu}, \mathbf{e}[x]_i) = \|P(\mathbf{e}[x]_i - \boldsymbol{\mu})\|$. Now, at iteration t the minimization is performed on points $\mathbf{e}[y]_i^t = \hat{P}^t(\mathbf{e}[x]_i - \hat{\boldsymbol{\mu}}^t)$ and the conditions for convergence with respect to $\boldsymbol{\mu}$ (skipping t and notation for effective parameters for brevity) are

$$\nabla_{\boldsymbol{\mu}}(h \circ f) = \nabla_{\boldsymbol{\mu}} \|P(\mathbf{e}[x]_i - \boldsymbol{\mu})\| \quad (6.26)$$

$$= \nabla_{\boldsymbol{\mu}} \sqrt{(\mathbf{e}[y]_i - \boldsymbol{\mu})^\top P^\top P (\mathbf{e}[y]_i - \boldsymbol{\mu})} \quad (6.27)$$

$$= \frac{1}{2\|P(\mathbf{e}[y]_i - \boldsymbol{\mu})\|} \cdot \nabla_{\boldsymbol{\mu}} M \quad (6.28)$$

$$(6.29)$$

$$\nabla_{\boldsymbol{\mu}}(w_i \cdot f^2) = w_i \cdot \nabla_{\boldsymbol{\mu}} M \quad (6.30)$$

$$(6.31)$$

where we have $M = (\mathbf{e}[y]_i - \boldsymbol{\mu})^\top P^\top P (\mathbf{e}[y]_i - \boldsymbol{\mu})$. This gives the expression for the weight

$$w_i^t = \frac{1}{2\|\hat{P}^t(\mathbf{e}[x]_i - \hat{\boldsymbol{\mu}}^t)\|}. \quad (6.32)$$

A similar derivation gives us the weights for the iteration step of P . Therefore in each iteration, we find the solutions to the following weighted least squares problems,

$$\boldsymbol{\mu}^{t+1} = \operatorname{argmin}_{\boldsymbol{\mu}} \sum_{i=1}^N w_i(P^t, \boldsymbol{\mu}^t) \|P^t(\mathbf{e}[y]_i - \boldsymbol{\mu})\|^2, \quad (6.33)$$

$$P^{t+1} = \operatorname{argmin}_P \sum_{i=1}^N w_i(P^t, \boldsymbol{\mu}^{t+1}) \|P(\mathbf{e}[y]_i^t - \boldsymbol{\mu}^{t+1})\|^2. \quad (6.34)$$

The effective centering and transformation matrix at iteration t are given by

$$\hat{\boldsymbol{\mu}}^t = \sum_{i=1}^t \left(\prod_{j=1}^{i-1} P_j^{-1} \right) \boldsymbol{\mu}^i, \quad \hat{P}^t = \prod_{i=1}^t P^i. \quad (6.35)$$

The whole procedure is summarized in Algorithm 1, where `chol` is used to denote the Cholesky decomposition.

Algorithm 1 Robust Whitening

```

1: procedure ROBUST WHITENING( $\mathcal{X}$ )
2:    $\mathbf{e}[z]_0 \leftarrow \mathcal{X}$ 
3:    $\boldsymbol{\mu}_0 \leftarrow$  Initialize centre to mean of  $\mathbf{e}[z]_0$ 
4:    $P_0 \leftarrow$  Initialize transform to identity matrix
5:   for  $t \leq \text{niter}$  do
6:      $\boldsymbol{\mu}^t \leftarrow \frac{1}{N} \sum_{i=1}^N w_i (P^{t-1}, \boldsymbol{\mu}^{t-1}) \mathbf{e}[z]_i^{t-1}$ 
7:      $\tilde{\Sigma}^t \leftarrow \sum_{i=1}^N w_i (P^{t-1}, \boldsymbol{\mu}^t) (\mathbf{e}[z]_i^{t-1} - \boldsymbol{\mu}^t) (\mathbf{e}[z]_i^{t-1} - \boldsymbol{\mu}^t)^\top$ 
8:      $P^t \leftarrow \frac{\text{chol}(\tilde{\Sigma}^t)}{\det(\text{chol}(\tilde{\Sigma}^t))^{1/d}}$ 
9:      $\mathbf{e}[z]^t \leftarrow P^t (\mathbf{e}[z]^{t-1} - \boldsymbol{\mu}^t)$ 
10:     $\hat{\boldsymbol{\mu}}^t \leftarrow \sum_{i=1}^t \left( \prod_{j=1}^{i-1} P^{j-1} \right) \boldsymbol{\mu}^i$ 
11:     $\hat{P}^t \leftarrow \prod_{i=0}^t P^i$ 
12:  end for
13:  return  $\hat{\boldsymbol{\mu}}^t, \hat{P}^t$ 
14: end procedure

```

6.5 Extension with supervision

We firstly review the work of Cai *et al.* [24] who perform supervised descriptor whitening and then present our extension for robust supervised whitening.

Background on linear discriminant projections [24]. The linear discriminant projections (LDP) are learned via supervision of pairs of similar and dissimilar descriptors. A pair (i, j) is similar if $(i, j) \in \mathcal{S}$ while dissimilar if $(i, j) \in \mathcal{D}$. The projections are learned in two parts. Firstly, the whitening part is obtained as the square-root of the intra-class covariance matrix $C_S^{-1/2}$, where

$$C_S = \sum_{(i,j) \in \mathcal{S}} (x_i - x_j)(x_i - x_j)^\top. \quad (6.36)$$

Then, the rotation part is given by the PCA of the inter-class covariance matrix which is computed in the space of the whitened descriptors. It is computed as $\text{eig}(C_S^{-1/2} C_D C_S^{-1/2})$, where

$$C_D = \sum_{(i,j) \in \mathcal{D}} (x_i - x_j)(x_i - x_j)^\top. \quad (6.37)$$

The final whitening is performed by $P_{SD}^\top(x - m)$, where m is the mean descriptor and $P_{SD} = C_S^{-1/2} \cdot \text{eig}(C_S^{-1/2} C_D C_S^{-1/2})$.

It is noted [24] that, if the number of descriptors is large compared to the number of classes (two in this case), then $C_D \approx C_{S \cup D}$ since $|\mathcal{S}| \ll |\mathcal{D}|$. This is the approach we follow.

Algorithm 2 Supervised Robust Whitening

```

1: procedure SUPERVISED ROBUST WHITENING( $\mathcal{X}, \mathcal{S}$ )
2:    $\mathcal{X}_{\mathcal{S}} = \{d : d = x_i - x_j, x_i \in \mathcal{X}, x_j \in \mathcal{X}, (i, j) \in \mathcal{S}\}$ 
3:    $\mathcal{X}_{\mathcal{S}} = \{\mathcal{X}_{\mathcal{S}} \cup -\mathcal{X}_{\mathcal{S}}\}$ 
4:    $\mu_1, P_1 \leftarrow \text{Robust Whitening}(\mathcal{X}_{\mathcal{S}})$ 
5:    $\mu \leftarrow \text{Geometric Median}(\mathcal{X})$ 
6:    $\bar{\mathcal{X}} \leftarrow \mathcal{X} - \mu$ 
7:    $\mu_2, P_2 \leftarrow \text{Robust Whitening}(P_1 \bar{\mathcal{X}})$ 
8:    $R_2 \leftarrow \text{eig}((P_2^\top P_2)^{-1})$ 
9:    $\hat{\mu} \leftarrow \mu + \mu_2$ 
10:   $\hat{P} \leftarrow P_1 R_2$ 
11:  return  $\hat{P}, \hat{\mu}$ 
12: end procedure

```

Robust linear discriminant projections. The proposed method uses the provided supervision in a robust manner by employing the method introduced in Section 6.4. The whitening is estimated in a robust manner by Algorithm 1 on the intra-class covariance. In this manner, small weights are assigned to pairs of descriptors that are found to be outliers. Then, the mean and covariance are estimated in a robust manner in the whitened space. The whole procedure is summarized in Algorithm 2. Mean μ_1 is zero due to the including the pairs in a symmetric manner.

6.6 Examples on synthetic data

We compare the proposed and the conventional whitening approaches on synthetic 2D data in order to demonstrate the robustness of our method to outliers. We sample a set of 2D points from a normal distribution, which is shown in Figure 6.2 (a) and then add an outlier and show the result in Figure 6.2 (b). In the absence of outliers, both methods provide a similar estimation as shown in Figure 6.3. It is also shown how the iterative approach reduces the cost at each iteration. With the presence of an outlier, the estimation of the conventional approach is largely affected, while the robust method gives a much better estimation, as shown in Figure 6.3. Using the Cauchy cost function the estimated covariance is very close to that of the ground truth. The weights assigned to each point with the robust approach are visualized in Figure 6.2 and show how the outlier is discarded in the final estimation. Finally, in Figure 6.4, we compare the conventional way with our approach for outlier of increasing distance.

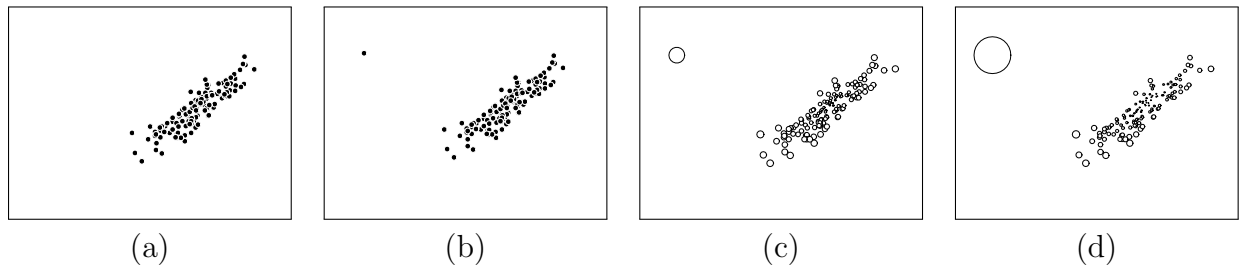


Figure 6.2: (a) Set of 2D points drawn from a Gaussian distribution with zero mean. (b) Same set as (a) with an additional point (outlier) placed at a distance equal to 2 times the maximum distance from the center of the initial set. (c) Visualization of the weights assigned in the set of (b) with the robust whitening which uses the ℓ_1 cost function. Note that the size of the circles is inversely proportional to the weight. (d) Same as (c), but using the Cauchy cost.

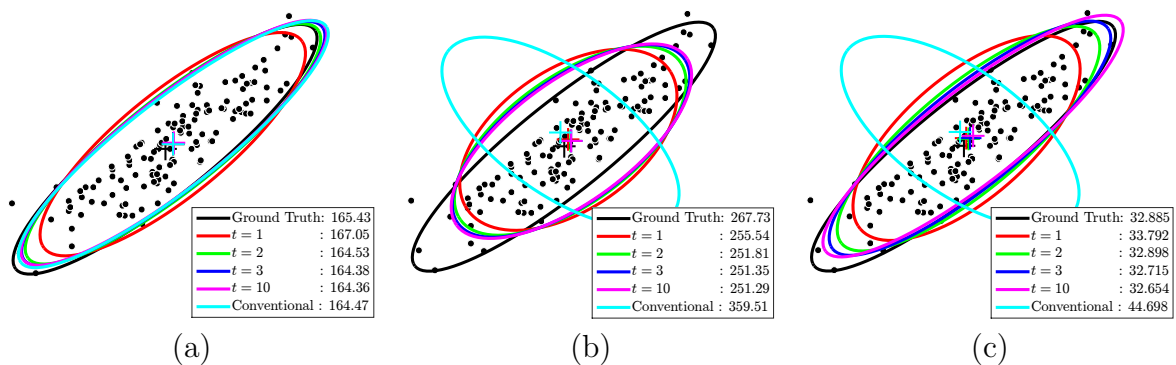


Figure 6.3: Visualization of the covariance (ellipse) and center (cross) of the estimated whitening transformation at iteration t and the conventional estimate. The example is performed using the set of 2D points of Figure 6.2. The ground truth distribution that created the data points is shown in black. The conventional estimate is shown in cyan. We show the effective estimate of the t^{th} iteration. The two approaches are compared without an outlier in (a) or with an outlier using ℓ_1 in (b) or Cauchy cost function in (c). The outlier is placed at a distance equal to 10 times the maximum inlier distance. The outlier is not plotted to keep the scale of the figure reasonable. The ℓ_1 (or Cauchy) cost is shown in the legend.

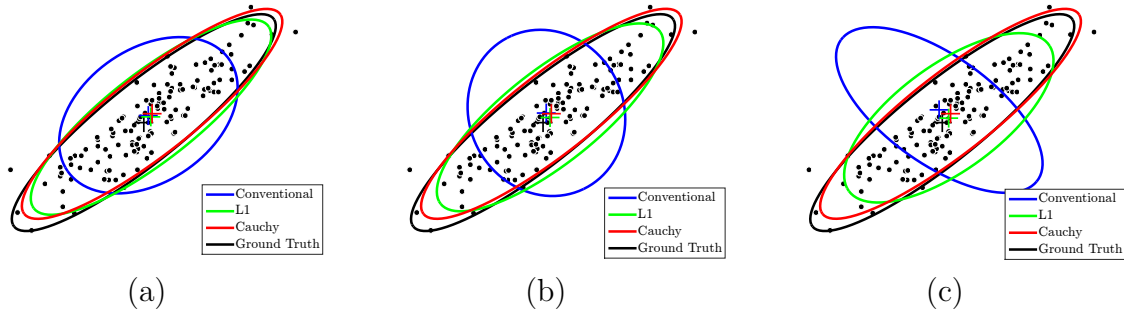


Figure 6.4: Visualization of the covariance (ellipse) and center (cross) of the estimated whitening transformation using the conventional approach and ours. The example is performed using the set of 2D points of Figure 6.2. The two approaches are compared for the case of an outlier placed at distance equal to 3 (a), 5 (b) and 10 (c) times the maximum inlier distance. The outlier is not shown to keep the resolution high.

6.7 Experiments

In this section, the robust whitening is applied to real-application data. In particular, we apply the whitening transformation on global image descriptors, *i.e.* a single vector representation per image, derived from CNN. We test on SPOC [12] descriptors, which are CNN-based image descriptors constructed via sum pooling of network activations in the internal convolutional layers. We evaluate on 3 popular retrieval benchmarks, namely Oxford5k, Paris6k and Holidays (the upright version), and use around 25k training images to learn the whitening. We use VGG network [89] to extract the descriptors and, in contrast to the work of Babenko and Lempitsky [12], we do not ℓ_2 -normalize the input vectors. The final ranking is obtained using Euclidean distance between the query and the database vectors. Evaluation is performed by measuring mean Average Precision (mAP). As in the case of conventional whitening, the dimension reduction is performed by preserving those dimensions that have the highest variance. This is done by finding an eigenvalue decomposition of the estimated covariance and ordering the eigenvectors according to decreasing eigenvalue.

There are many approaches performing robust PCA [26, 104, 105] by assuming that the data matrix can be decomposed into the sum of a low rank matrix and a sparse matrix corresponding to the outliers. We employ the robust PCA (RPCA) method by Candès *et al.* [26] to perform a comparison. The low rank matrix is recovered and PCA whitening is learned on this.

We present results in Table 6.1, where the robust approach offers a consistent improvement over the conventional PCA whitening [12]. Especially in the case where the whitening is learned on few training vectors, the improvement is larger as outliers will heavily influence the conventional whitening, as shown in Figure 6.5. Our approach is also better than RPCA whitening for large dimensionalities. It seems that RPCA

underestimates the rank of the matrix and does not offer any further improvements for large dimensions.

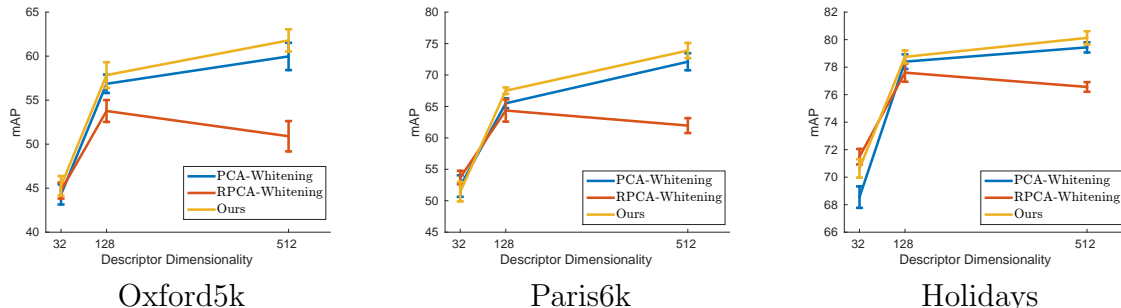


Figure 6.5: Retrieval performance comparison using mAP on 3 common benchmarks. Comparison of the conventional PCA whitening, RPCA whitening and our approach for descriptors of varying dimensionality. The training set contains a small subset of 512 vectors randomly selected. The experiment is performed 10 times and mean performance is reported while standard deviation is shown on the curves. Descriptors extracted using VGG.

Dataset		Oxford5k			Paris6k			Holidays		
Method	S	32D	128D	512D	32D	128D	512D	32D	128D	512D
Raw		-	-	51.4	-	-	61.6	-	-	78.8
PCA whitening		44.7	56.6	66.7	53.4	67.0	77.1	69.6	78.4	80.6
RPCA whitening		44.0	52.4	55.6	55.9	61.1	65.1	70.5	75.8	77.4
Ours		45.8	58.5	67.7	50.0	68.3	78.4	70.7	78.8	81.8
LDP	×	39.4	59.9	68.8	56.1	70.2	76.6	67.5	77.7	80.8
Ours	×	49.9	62.3	70.3	57.6	72.0	78.0	69.0	78.6	82.1

Table 6.1: Retrieval performance comparison using mAP on 3 common benchmarks. Comparison of retrieval using the initial sum-pooled CNN activations, post-processing using the baselines and our methods for unsupervised and supervised whitening. Results for descriptors of varying dimensionality. The full training set is used. Descriptors extracted using VGG. S: indicates the use of supervision.

6.8 Discussion

In this chapter, we cast the problem of data whitening as minimization of robust cost functions. We showed how geometric interpretation is equivalent to a minimization of the sum of squared ℓ_2 norms over a set of linear transforms. We briefly introduced the framework of Iteratively Re-weighted Least Squares (IRLS), and showed that it can be used to minimize a large family of cost functions. We proposed a novel method to iteratively estimate a whitening transformation that is robust to the presence of outliers. We proposed a natural extension to the supervised case.

The applicability of the proposed method goes beyond robust whitening. Consider, for example, the task of affine-invariant descriptors of local features, such as MSERs [58]. A common approach is to transform the detected feature into a canonical frame prior to computing a robust descriptor based on the gradient map of the normalized patch (SIFT [55]). To remove the effect of an affine transformation, a centre of gravity and centered second-order moment (covariance matrix) are used. It can be shown that both the centre of gravity and the covariance matrix are affine-covariants, *i.e.* if the input point set is transformed by an affine transformation A , they transform with the same transformation A .

The proposed method searches μ and P by minimization over all possible affine transformations with a fixed determinant. In turn, μ is fully affine covariant and P is affine covariant up to an unknown scale (and rotation, $P^\top P$ cancels the rotation). To the best of our knowledge, this type of robust-to-outliers covariants have not been used.

Conclusions

We studied construction of local descriptors for image matching and retrieval and proposed novel solutions. We proposed two architectures and evaluated their performance on standard and modern benchmarks. Both achieved state-of-the-art results when introduced. We also address the robust estimation of whitening transforms in the presence of outliers, and proposed a novel iterative algorithm for unsupervised and supervised cases.

In Chapter 4, we proposed a multiple-kernel local descriptor combining two parametrizations of gradient position and direction. We have performed descriptor whitening and have shown that its effect on patch similarity is semantically meaningful. We show that learning the whitening in a supervised or unsupervised way boosts the performance. We show that it is beneficial to learn the whitening in a robust manner in the unsupervised case. Interestingly, the unsupervised variant generalizes better and is the best performing performing unsupervised hand-crafted descriptor so far. As opposed to deep local descriptors, we note that computation and training is much faster. The lessons learned from analyzing the similarity after whitening were applied for further improvements of local descriptors in the succeeding work.

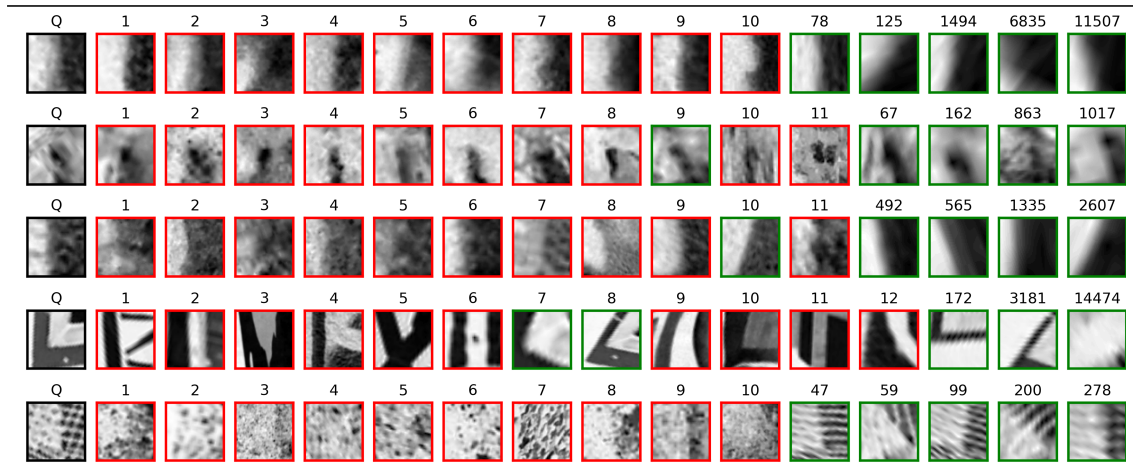
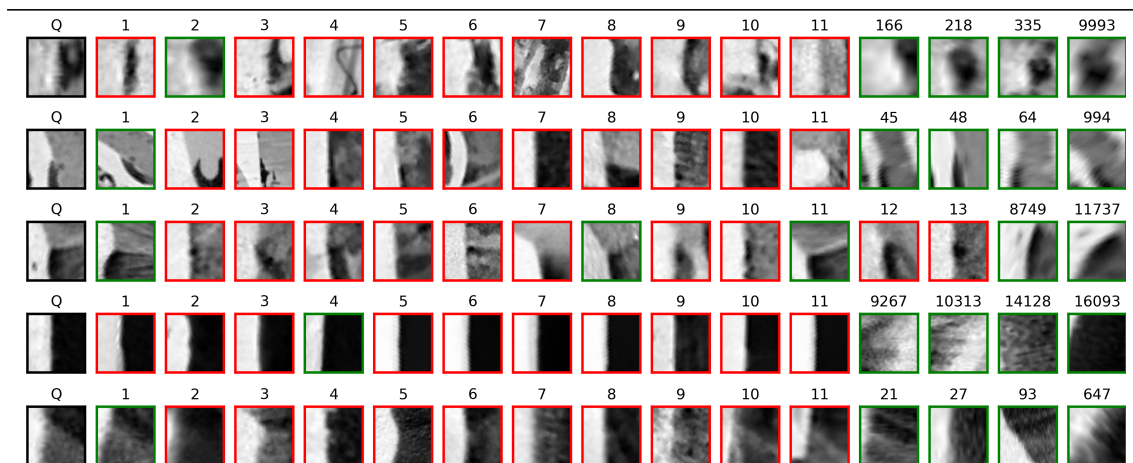
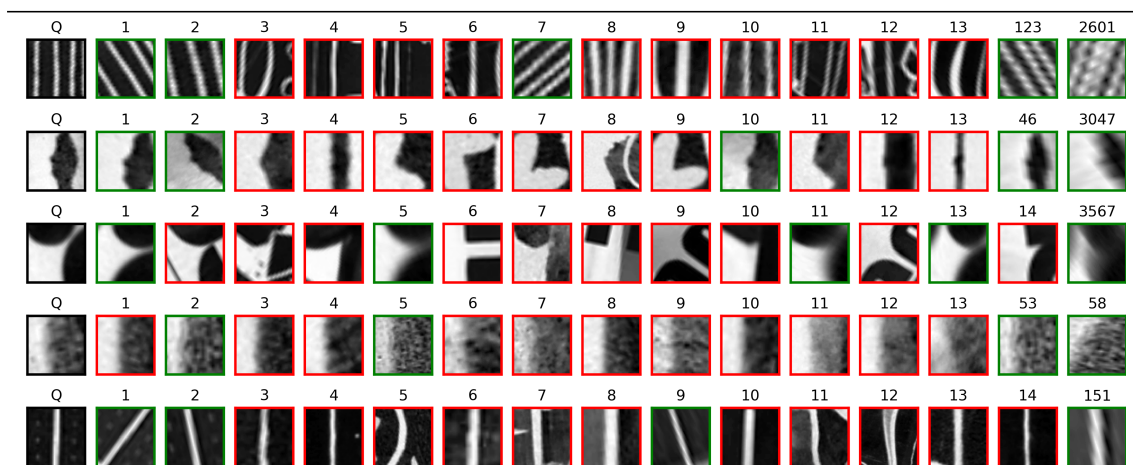
In Chapter 5, we extend our approach to deep local descriptors and improve upon conventional architectures, by interpreting conventional convolutional local descriptors as efficient match kernels. We reveal that they learn spatially variant encoding through the last FC layer. We proposed a novel local descriptor that uses measurements of fully convolutional networks and explicitly encodes the spatial information. It achieves the same performance as state-of-the-art descriptors with fewer parameters and consistently outperforms on all standard patch benchmarks with the same number of parameters. The number of parameters of our model is independent of the size of the input patch. This approach is not limited to local descriptors, but can be extended to all use cases where we have to spatially encode measurements on an established coordinate system, as a local pooling method.

In Chapter 6, we cast the problem of data whitening as minimization of robust cost functions. We proposed algorithms to iteratively estimate a whitening transformation that is robust to the presence of outliers, for both supervised and unsupervised case. With the use of synthetic data, we show that our estimation is almost unaffected even with extreme cases of outliers, while it also offers improvements when whitening CNN descriptors for image retrieval. Our approach is not limited to estimation of whitening transforms, but can be extended to other use cases, *e.g.* robust patch rectification.

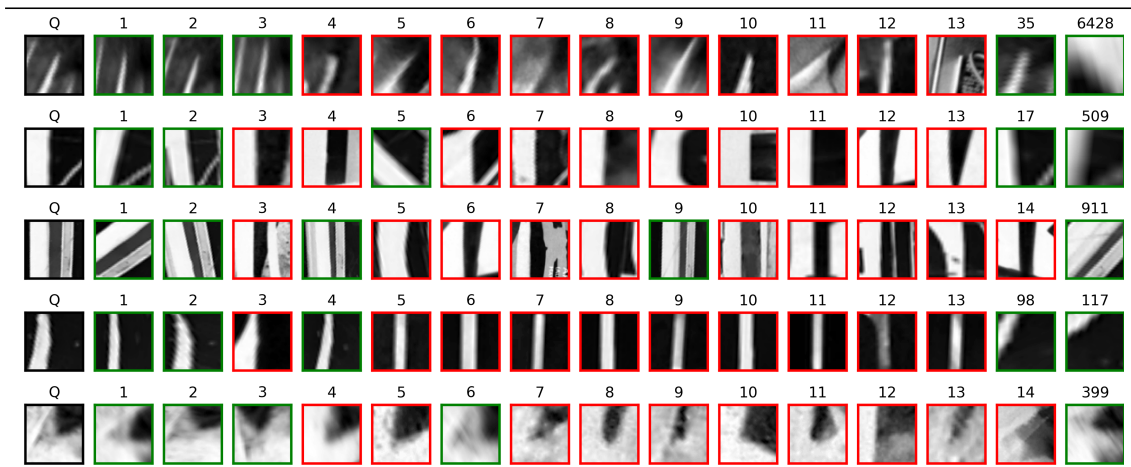
Examples

We investigate the strengths and weaknesses of each architecture by identifying cases where one *succeeds* and the other *fails*, on the task of *patch retrieval*. We use the test set provided by the HPatches benchmark, specifically, the ‘hard’ subset of ‘view’ split. It consists 10K queries from the reference image of each scene, which are matched against 20K distractors, and the 5 correct correspondences in the other images from the scene. We show a subset of retrievals for cases where the descriptor gets n correct retrievals within the first 5 retrievals, for $n \in \{0, \dots, 5\}$. We hope to qualitatively gauge what kind of behaviour each architecture exhibits.

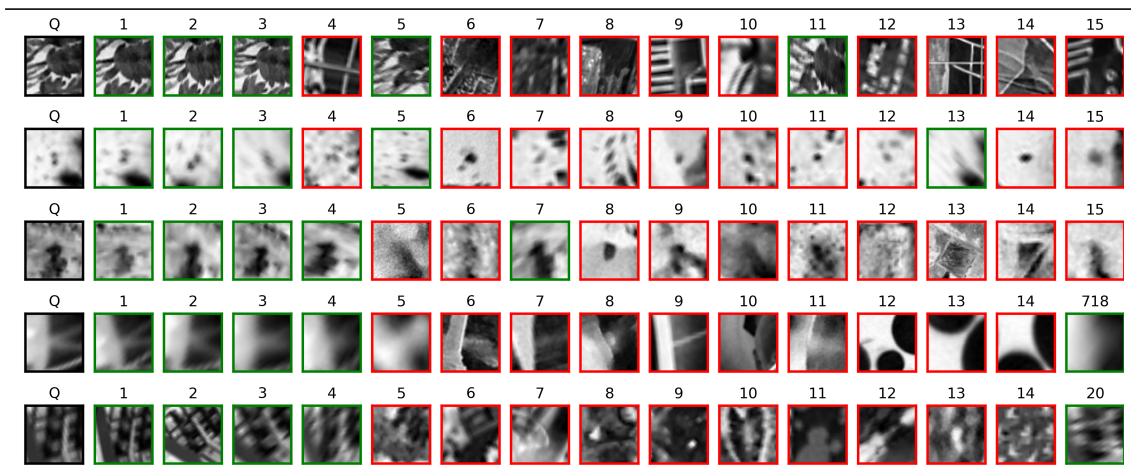
A.1 HardNet

HardNet: $n = 0$ HardNet: $n = 1$ HardNet: $n = 2$ 

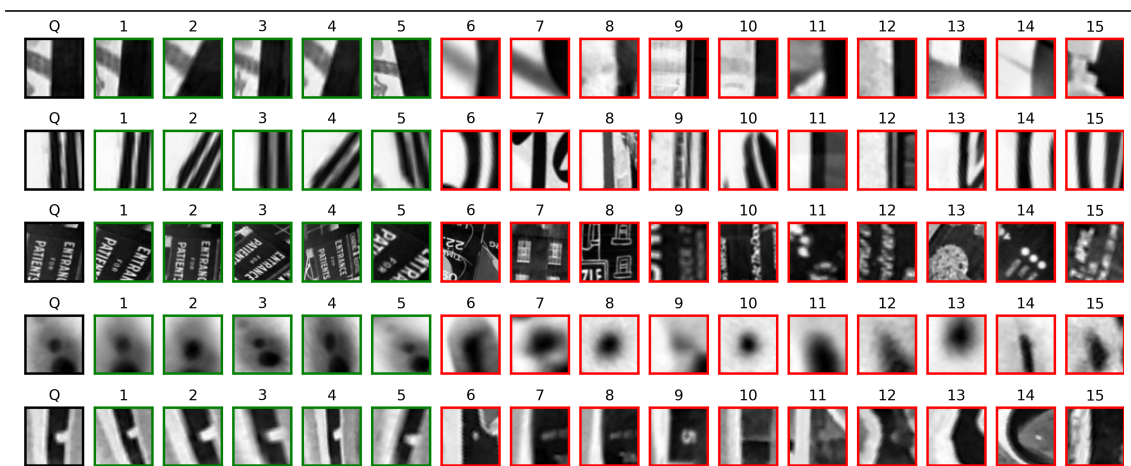
HardNet: $n = 3$



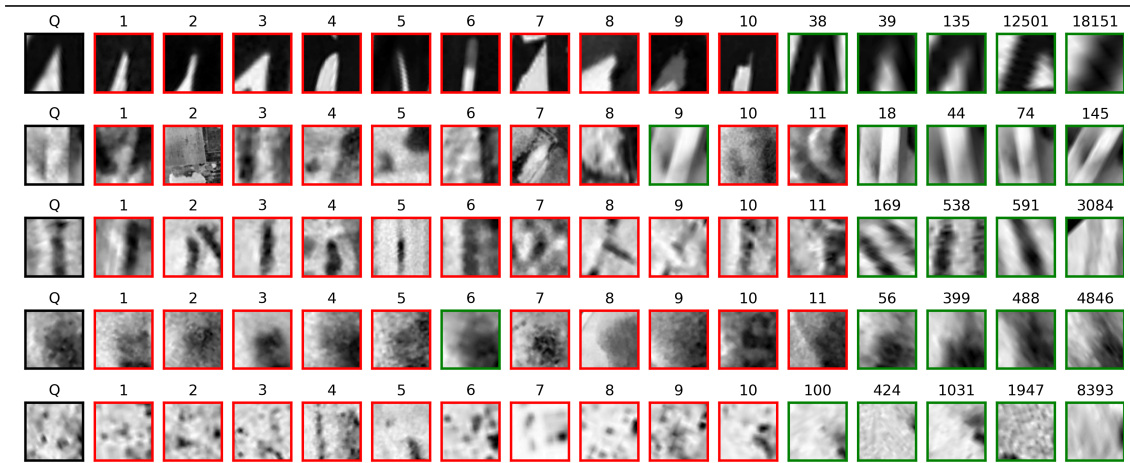
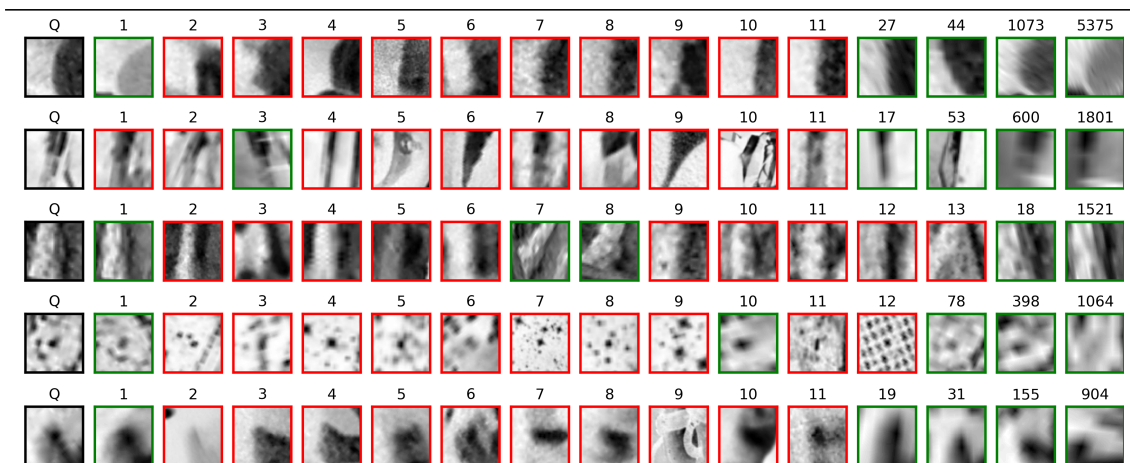
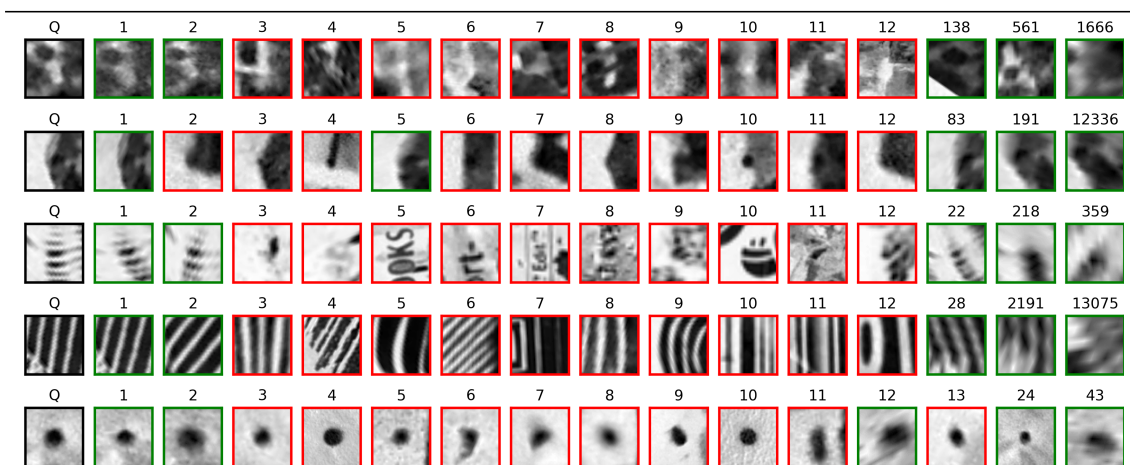
HardNet: $n = 4$



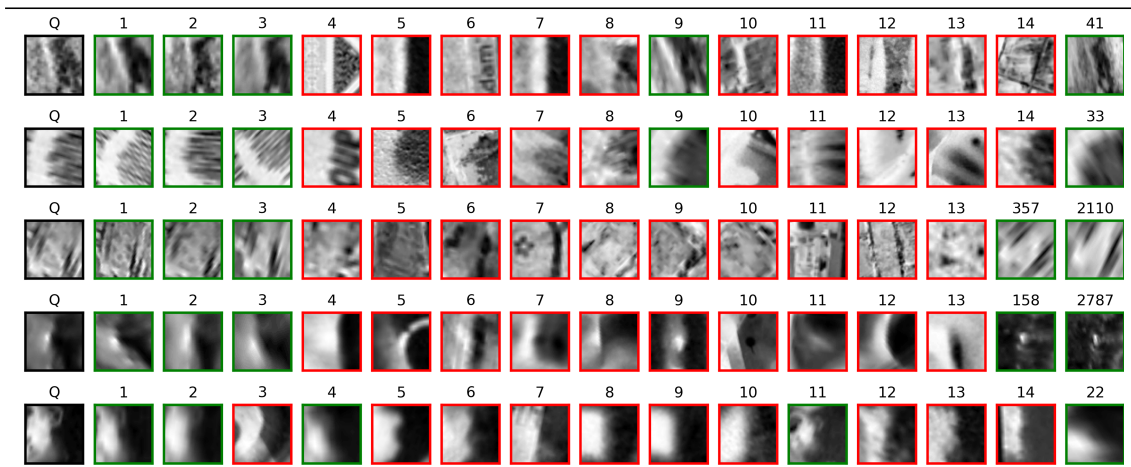
HardNet: $n = 5$



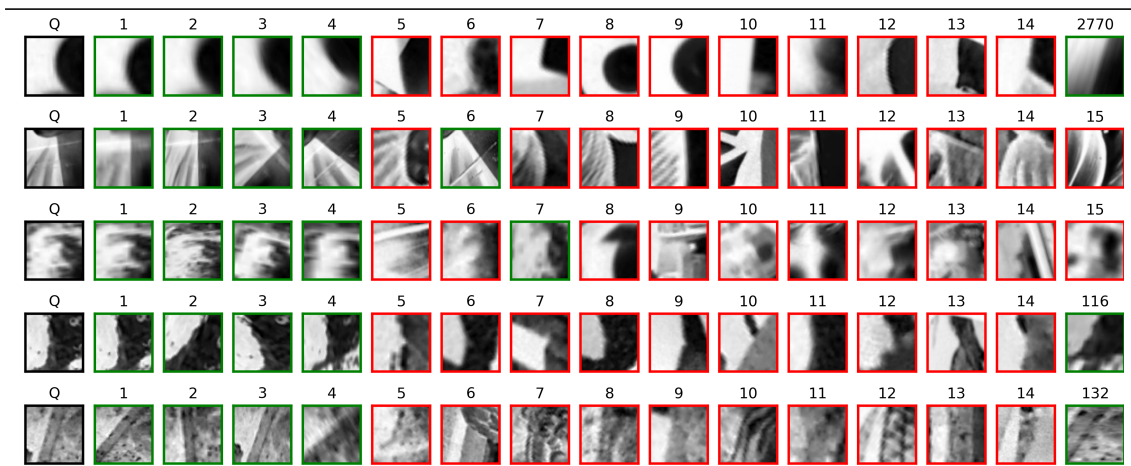
A.2 MKDNet

MKDNet: $n = 0$ MKDNet: $n = 1$ MKDNet: $n = 2$ 

MKDNet: $n = 3$



MKDNet: $n = 4$

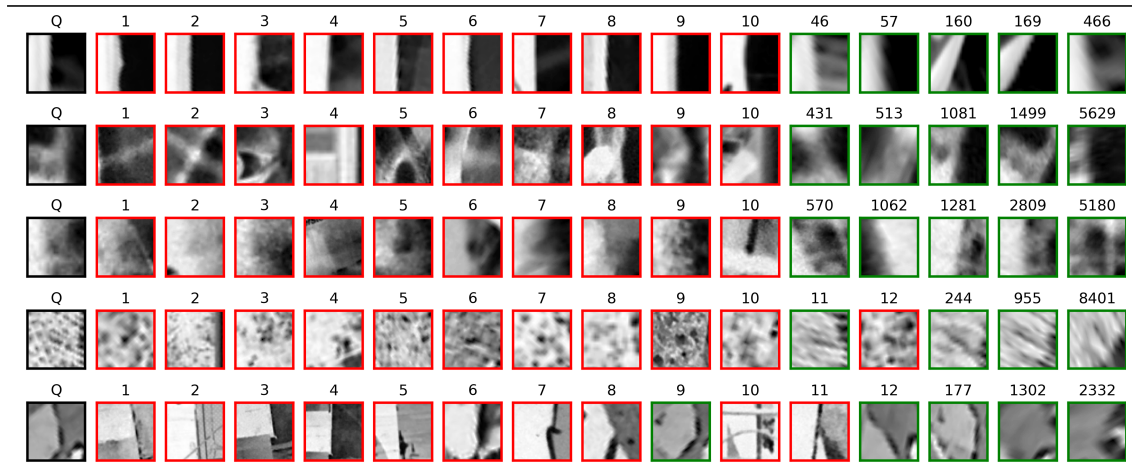


MKDNet: $n = 5$

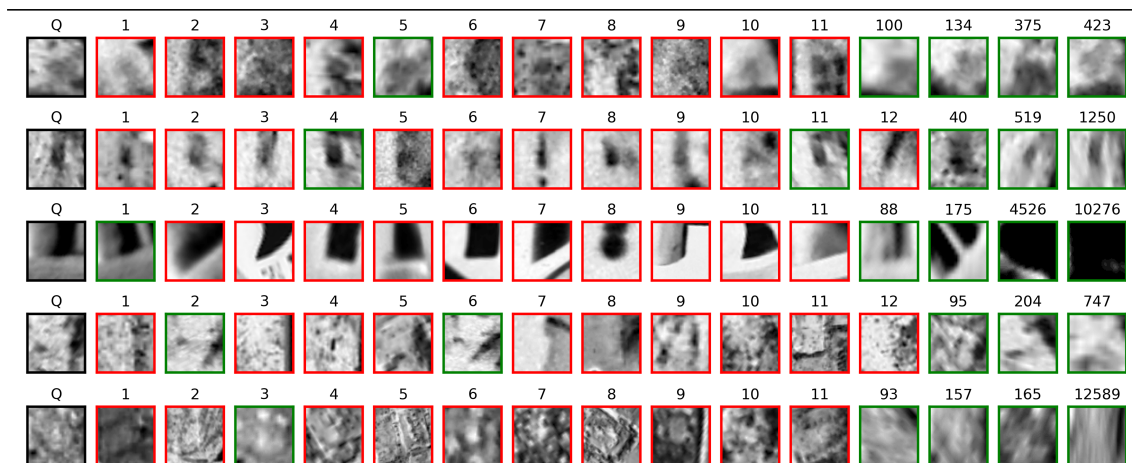


A.3 Cartesian

Cartesian: $n = 0$



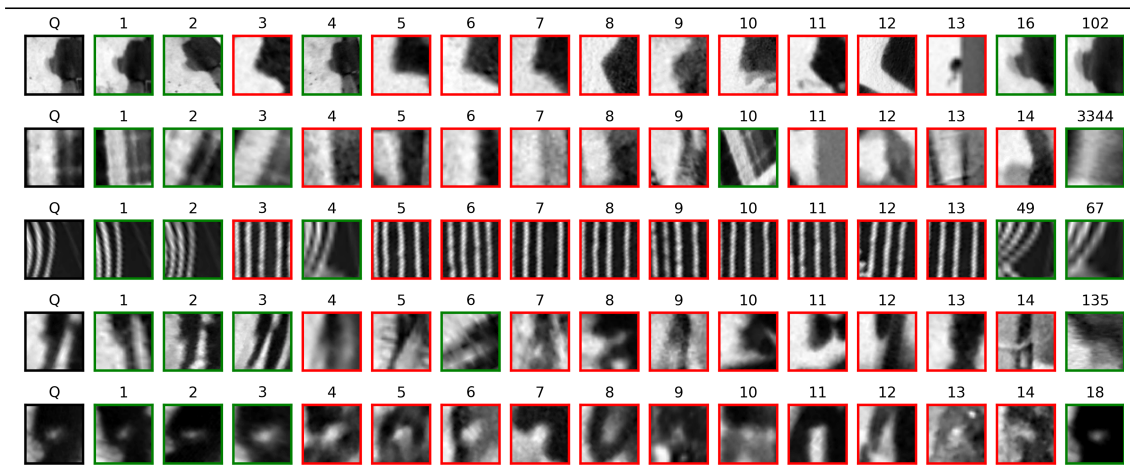
Cartesian: $n = 1$



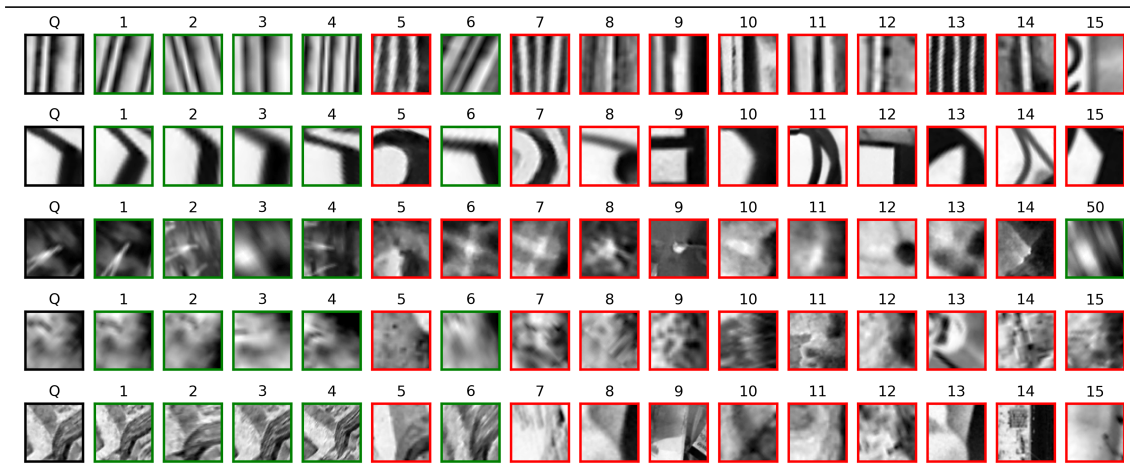
Cartesian: $n = 2$



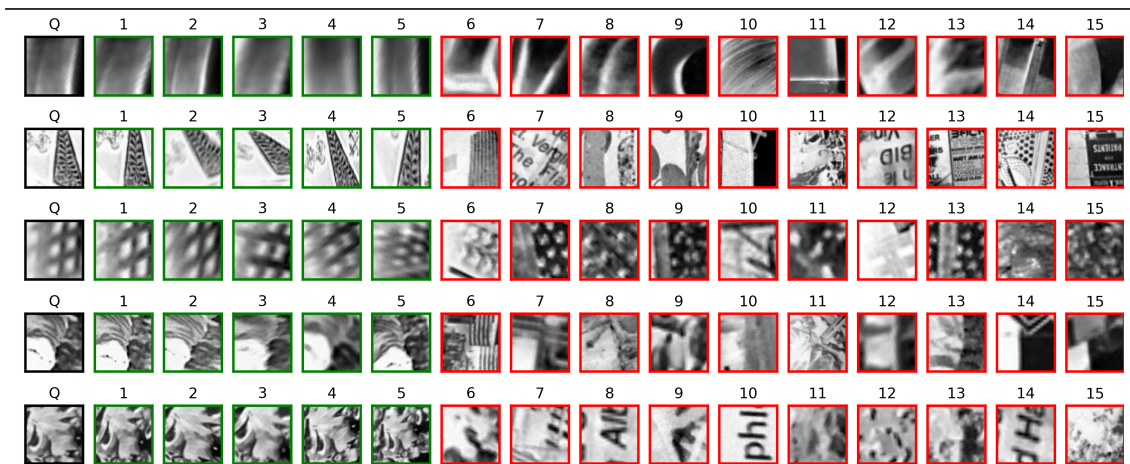
Cartesian: $n = 3$



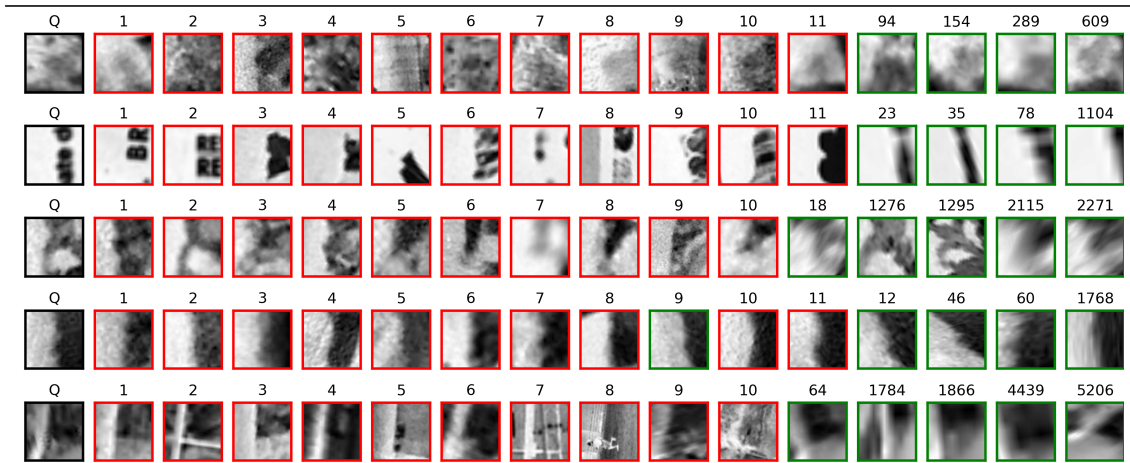
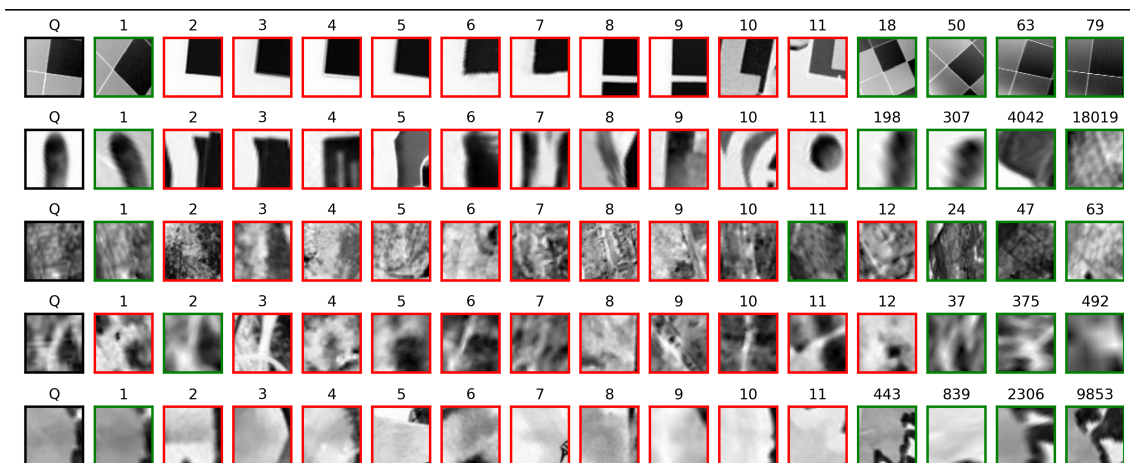
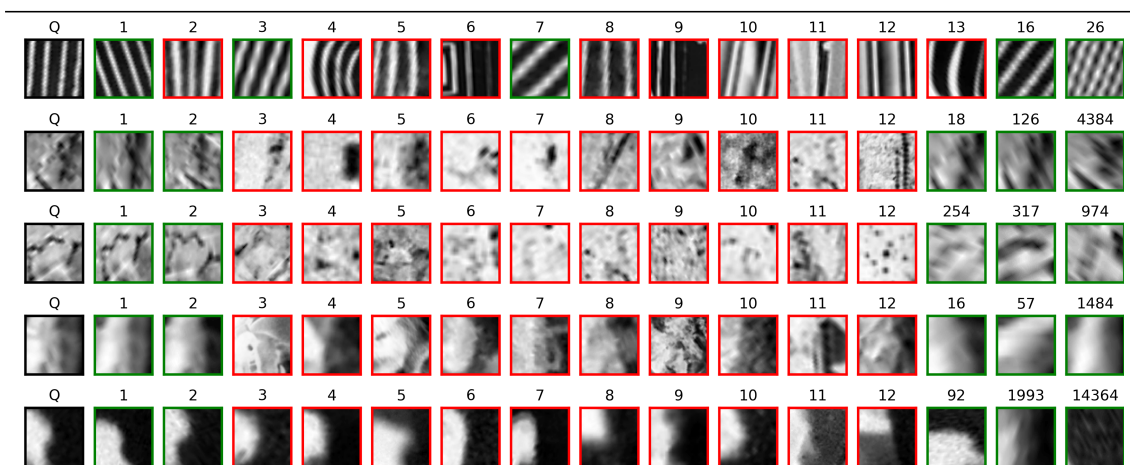
Cartesian: $n = 4$



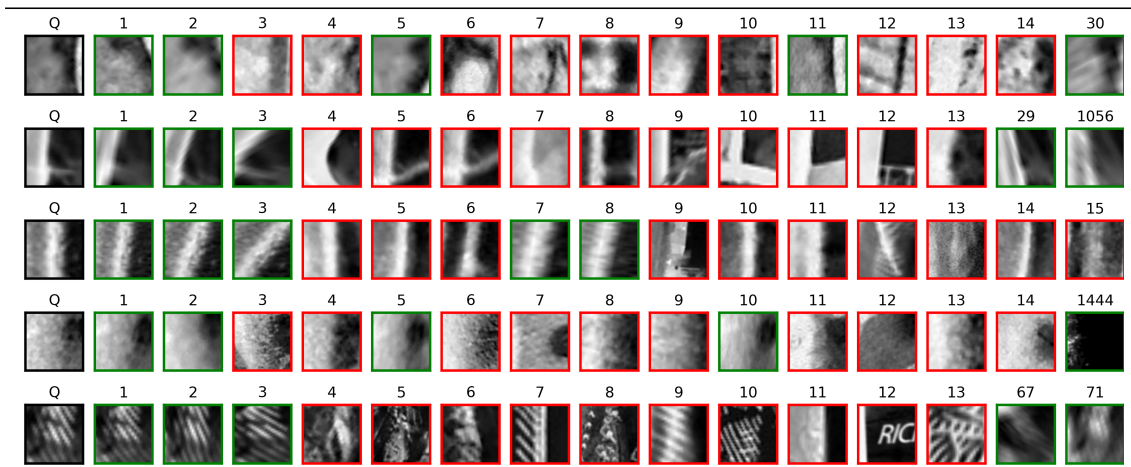
Cartesian: $n = 5$



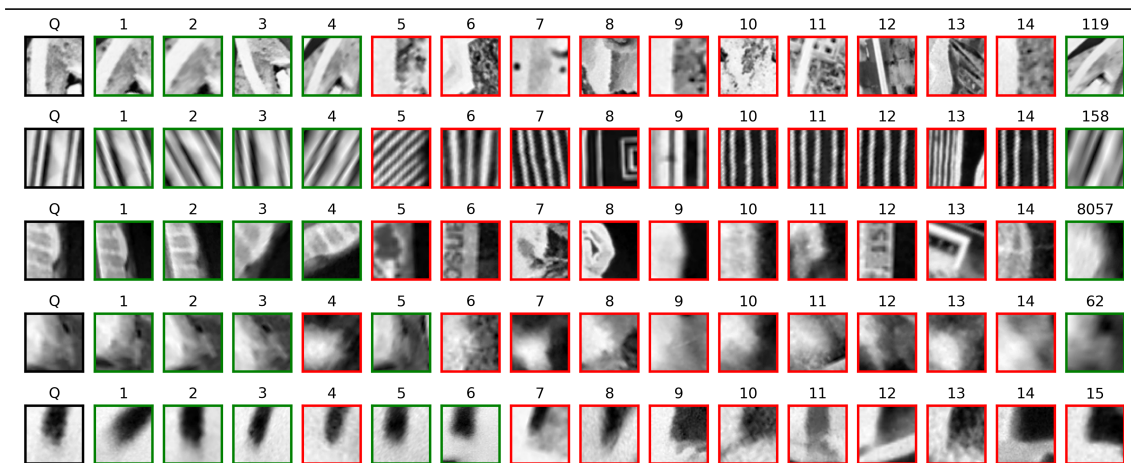
A.4 Polar

Polar: $n = 0$ Polar: $n = 1$ Polar: $n = 2$ 

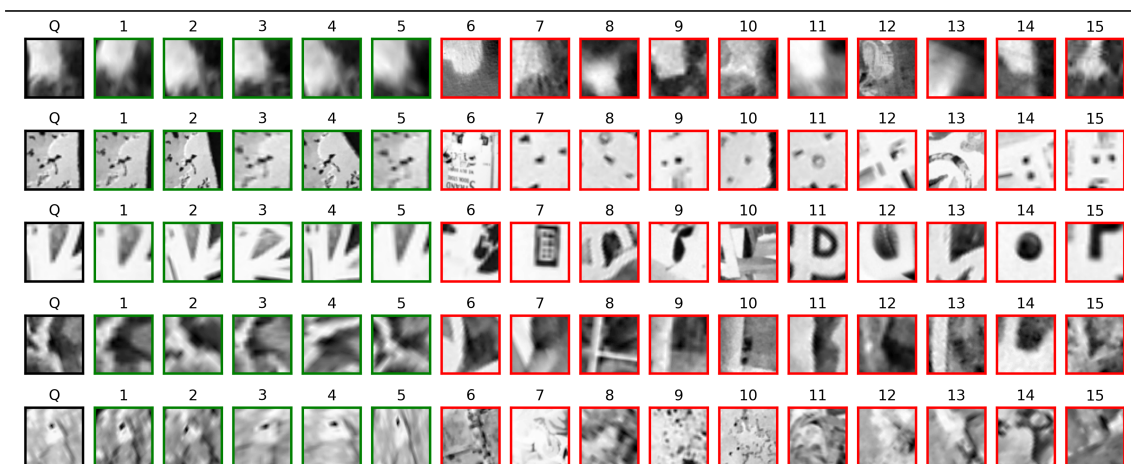
Polar: $n = 3$



Polar: $n = 4$



Polar: $n = 5$



Bibliography

- [1] URL: <https://support.apple.com/en-us/HT208108>.
- [2] URL: <https://images.google.com>.
- [3] URL: <https://zeekit.me>.
- [4] URL: <https://www.magicplan.app/en/>.
- [5] URL: <https://maps.google.com>.
- [6] URL: <https://www.scape.io/>.
- [7] In: 2019. URL: <https://vision.uvic.ca/image-matching-challenge/leaderboard/>.
- [8] Khurrum Aftab and Richard Hartley. ‘Convergence of iteratively re-weighted least squares to robust M-estimators’. In: *2015 IEEE Winter Conference on Applications of Computer Vision*. 2015.
- [9] Javier Aldana-Iuit et al. ‘In the saddle: chasing fast and repeatable features’. In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. 2016, p. 8.
- [10] Mitsuru Ambai and Yuichi Yoshida. ‘CARD: Compact and real-time descriptors’. In: *2011 International Conference on Computer Vision*. 2011, p. 111.
- [11] Relja Arandjelović and Andrew Zisserman. ‘Three things everyone should know to improve object retrieval’. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, p. 1061.
- [12] Artem Babenko and Victor Lempitsky. ‘Aggregating local deep features for image retrieval’. In: *Proceedings of the IEEE international conference on computer vision*. 2015, p. 363.
- [13] Vassileios Balntas et al. ‘HPatches: A benchmark and evaluation of handcrafted and learned local descriptors’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, p. 105.

- [14] Vassileios Balntas et al. ‘Learning local feature descriptors with triplets and shallow convolutional neural networks.’ In: *Bmvc*. 2016, p. 91.
- [15] Herbert Bay, Tinne Tuytelaars and Luc Van Gool. ‘Surf: Speeded up robust features’. In: *European conference on computer vision*. 2006, p. 15469.
- [16] Herbert Bay et al. ‘Speeded-up robust features (SURF)’. In: *Computer vision and image understanding* 110.3 (2008), pp. 346–359.
- [17] Liefeng Bo, Xiaofeng Ren and Dieter Fox. ‘Depth kernel descriptors for object recognition’. In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2011, p. 325.
- [18] Liefeng Bo, Xiaofeng Ren and Dieter Fox. ‘Kernel descriptors for visual recognition’. In: *Advances in neural information processing systems*. 2010, pp. 244–252.
- [19] Liefeng Bo and Cristian Sminchisescu. ‘Efficient match kernel between sets of features for visual recognition’. In: *Advances in neural information processing systems*. 2009, p. 267.
- [20] Gary R Bradski. ‘The OpenCV Library’. In: *Dr. Dobb’s Journal of Software Tools* (2000).
- [21] Matthew Brown, Gang Hua and Simon Winder. ‘Discriminative learning of local image descriptors’. In: *IEEE transactions on pattern analysis and machine intelligence* (2010), p. 403.
- [22] Matthew Brown, Richard Szeliski and Simon Winder. ‘Multi-image matching using multi-scale oriented patches’. In: *CVPR*. 2005, p. 527.
- [23] Andrei Bursuc, Giorgos Tolias and Hervé Jégou. ‘Kernel local descriptors with implicit rotation matching’. In: *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*. ACM. 2015, p. 15.
- [24] Hongping Cai, Krystian Mikolajczyk and Jiri Matas. ‘Learning linear discriminant projections for dimensionality reduction of image descriptors’. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2010), p. 107.
- [25] Michael Calonder et al. ‘Brief: Binary robust independent elementary features’. In: *European conference on computer vision*. 2010, p. 3040.
- [26] Emmanuel J Candès et al. ‘Robust principal component analysis?’ In: *Journal of the ACM* (2011), p. 4578.
- [27] Ondřej Chum. ‘Low dimensional explicit feature maps’. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, p. 8.
- [28] Jia Deng et al. ‘Imagenet: A large-scale hierarchical image database’. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.
- [29] Jingming Dong and Stefano Soatto. ‘Domain-size pooling in local descriptors: DSP-SIFT’. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, p. 93.

- [30] Shimon Edelman, Nathan Intrator and Tomaso Poggio. ‘Complex cells and object recognition’. In: (1997).
- [31] Mark Everingham et al. ‘The pascal visual object classes (voc) challenge’. In: *International journal of computer vision* (2010), p. 6603.
- [32] Philipp Fischer, Alexey Dosovitskiy and Thomas Brox. ‘Descriptor matching with convolutional neural networks: a comparison to sift’. In: *arXiv preprint arXiv:1405.5769* (2014), p. 186.
- [33] Luc Florack et al. ‘The Gaussian scale-space paradigm and the multiscale local jet’. In: *International Journal of Computer Vision* (1996).
- [34] William T. Freeman and Edward H Adelson. ‘The design and use of steerable filters’. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* (1991), p. 3910.
- [35] Xufeng Han et al. ‘Matchnet: Unifying feature and metric learning for patch-based matching’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, p. 402.
- [36] Christopher G Harris, Mike Stephens et al. ‘A combined corner and edge detector.’ In: *Alvey vision conference*. 1988, p. 16532.
- [37] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.
- [38] Kun He, Yan Lu and Stan Sclaroff. ‘Local descriptors optimized for average precision’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, p. 24.
- [39] Marko Heikkilä, Matti Pietikäinen and Cordelia Schmid. ‘Description of interest regions with local binary patterns’. In: *Pattern recognition* (2009), p. 1181.
- [40] Hervé Jégou and Ondřej Chum. ‘Negative evidences and co-occurrences in image retrieval: The benefit of PCA and whitening’. In: *European conference on computer vision*. 2012, p. 306.
- [41] Yuhe Jin et al. ‘Image Matching across Wide Baselines: From Paper to Practice’. In: *arXiv preprint arXiv:2003.01587* (2020).
- [42] Andrew E. Johnson and Martial Hebert. ‘Using spin images for efficient object recognition in cluttered 3D scenes’. In: *IEEE Transactions on pattern analysis and machine intelligence* 21.5 (1999), pp. 433–449.
- [43] Yannis Kalantidis, Clayton Mellina and Simon Osindero. ‘Cross-dimensional weighting for aggregated deep convolutional features’. In: *European conference on computer vision*. Springer. 2016, pp. 685–701.
- [44] Yan Ke, Rahul Sukthankar et al. ‘PCA-SIFT: A more distinctive representation for local image descriptors’. In: *CVPR* (2004), p. 4231.
- [45] Jan J Koenderink and Andrea J van Doorn. ‘Representation of local geometry in the visual system’. In: *Biological cybernetics* (1987).

- [46] Iasonas Kokkinos and Alan Yuille. ‘Scale invariance without scale selection’. In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. 2008, p. 88.
- [47] Svetlana Lazebnik, Cordelia Schmid and Jean Ponce. ‘A sparse texture representation using affine-invariant regions’. In: *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings*. Vol. 2. IEEE. 2003, pp. II–II.
- [48] Olivier Ledoit and Michael Wolf. ‘A well-conditioned estimator for large-dimensional covariance matrices’. In: *Journal of multivariate analysis* (2004), p. 1724.
- [49] Olivier Ledoit and Michael Wolf. ‘Honey, I shrunk the sample covariance matrix’. In: *The Journal of Portfolio Management* (2004), p. 908.
- [50] Stefan Leutenegger, Margarita Chli and Roland Siegwart. ‘BRISK: Binary robust invariant scalable keypoints’. In: *2011 IEEE international conference on computer vision (ICCV)*. 2011, p. 2817.
- [51] Yunpeng Li, Noah Snavely and Dan Huttenlocher. ‘Location Recognition using Prioritized Feature Matching’. In: *European Conference on Computer Vision (ECCV)*. 2010, p. 353.
- [52] Tony Lindeberg. ‘Scale-space theory: A basic tool for analyzing structures at different scales’. In: *Journal of applied statistics* (1994), p. 1640.
- [53] Rosanne Liu et al. ‘An intriguing failing of convolutional neural networks and the coordconv solution’. In: *Advances in Neural Information Processing Systems*. 2018, p. 63.
- [54] David G Lowe. ‘Distinctive image features from scale-invariant keypoints’. In: *International journal of computer vision* (2004), p. 52482.
- [55] David G Lowe. ‘Object recognition from local scale-invariant features’. In: *Proceedings of the seventh IEEE international conference on computer vision*. 1999.
- [56] Zixin Luo et al. ‘Geodesc: Learning local descriptors by integrating geometry constraints’. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, p. 17.
- [57] Jiri Matas, T Obdrzalek and Ondřej Chum. ‘Local affine frames for wide-baseline stereo’. In:
- [58] Jiri Matas et al. ‘Robust wide-baseline stereo from maximally stable extremal regions’. In: *Image and vision computing*. 2004.
- [59] Krystian Mikolajczyk and Jiri Matas. ‘Improving descriptors for fast tree matching by optimal linear projection’. In: *2007 IEEE 11TH INTERNATIONAL CONFERENCE ON COMPUTER VISION, VOLS 1-6*. 2007, p. 122.
- [60] Krystian Mikolajczyk and Cordelia Schmid. ‘A performance evaluation of local descriptors’. In: *IEEE transactions on pattern analysis and machine intelligence* (2005), p. 8857.

- [61] Krystian Mikolajczyk et al. ‘A comparison of affine region detectors’. In: *International journal of computer vision* (2005).
- [62] Anastasiia Mishchuk et al. ‘Working hard to know your neighbor’s margins: Local descriptor learning loss’. In: *Advances in Neural Information Processing Systems*. 2017, p. 79.
- [63] Dmytro Mishkin, Filip Radenović and Jiri Matas. ‘Repeatability is not enough: Learning affine regions via discriminability’. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 284–300.
- [64] Dmytro Mishkin et al. ‘Wxbs: Wide baseline stereo generalizations’. In: *arXiv preprint arXiv:1504.06603* (2015), p. 26.
- [65] Rahul Mitra et al. ‘A large dataset for improving patch matching’. In: *arXiv preprint arXiv:1801.01466* (2018), p. 8.
- [66] Arun Mukundan, Giorgos Toliás and Ondřej Chum. ‘Explicit Spatial Encoding for Deep Local Descriptors’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, p. 1.
- [67] Arun Mukundan, Giorgos Toliás and Ondřej Chum. ‘Multiple-kernel local-patch descriptor’. In: *British Machine Vision Conference* (2017), p. 6.
- [68] Arun Mukundan, Giorgos Toliás and Ondřej Chum. ‘Robust Data Whitening as an Iteratively Re-weighted Least Squares Problem’. In: *Scandinavian Conference on Image Analysis*. Springer. 2017, p. 0.
- [69] Arun Mukundan et al. ‘Understanding and improving kernel local descriptors’. In: *International Journal of Computer Vision* (2018), p. 1.
- [70] David Novotny et al. ‘Semi-convolutional operators for instance segmentation’. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, p. 18.
- [71] Timo Ojala, Matti Pietikäinen and David Harwood. ‘A comparative study of texture measures with classification based on featured distributions’. In: *Pattern recognition* (1996).
- [72] Timo Ojala, Matti Pietikäinen and Topi Mäenpää. ‘Multiresolution gray-scale and rotation invariant texture classification with local binary patterns’. In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* (2002), p. 13020.
- [73] Mattis Paulin et al. ‘Local Convolutional Features with Unsupervised Training for Image Retrieval’. In: *International Conference on Computer Vision (ICCV)*. 2015, p. 127.
- [74] Michal Perd’och, Ondřej Chum and Jiri Matas. ‘Efficient representation of local geometry for large scale object retrieval’. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2009, pp. 9–16.
- [75] Filip Radenović, Giorgos Toliás and Ondřej Chum. ‘CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples’. In: *European conference on computer vision*. Springer. 2016, p. 278.

- [76] Filip Radenović, Giorgos Tolias and Ondřej Chum. ‘Fine-tuning CNN image retrieval with no human annotation’. In: *IEEE transactions on pattern analysis and machine intelligence* 41.7 (2018), pp. 1655–1668.
- [77] Filip Radenović et al. ‘Revisiting oxford and paris: Large-scale image retrieval benchmarking’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018.
- [78] Ali S Razavian et al. ‘Visual instance retrieval with deep convolutional networks’. In: *ITE Transactions on Media Technology and Applications* 4.3 (2016), pp. 251–258.
- [79] Edward Rosten, Reid Porter and Tom Drummond. ‘Faster and better: A machine learning approach to corner detection’. In: *IEEE transactions on pattern analysis and machine intelligence* (2008), p. 1616.
- [80] Ethan Rublee et al. ‘ORB: An efficient alternative to SIFT or SURF.’ In: *ICCV*. 2011, p. 5270.
- [81] Andrew M Saxe, James L McClelland and Surya Ganguli. ‘Exact solutions to the nonlinear dynamics of learning in deep linear neural networks’. In: *arXiv preprint arXiv:1312.6120* (2013), p. 616.
- [82] Cordelia Schmid and Roger Mohr. ‘Local grayvalue invariants for image retrieval’. In: *IEEE transactions on pattern analysis and machine intelligence* (1997), p. 2107.
- [83] Johannes L Schonberger et al. ‘Comparative evaluation of hand-crafted and learned local features’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, p. 75.
- [84] Johannes Lutz Schönberger and Jan-Michael Frahm. ‘Structure-from-Motion Revisited’. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, p. 495.
- [85] Johannes Lutz Schönberger et al. ‘Pixelwise View Selection for Unstructured Multi-View Stereo’. In: *European Conference on Computer Vision (ECCV)*. 2016, p. 206.
- [86] Eli Shechtman and Michal Irani. ‘Matching Local Self-Similarities across Images and Videos.’ In: *CVPR*. 2007, p. 1048.
- [87] Edgar Simo-Serra et al. ‘Discriminative learning of deep convolutional feature point descriptors’. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, p. 360.
- [88] Karen Simonyan, Andrea Vedaldi and Andrew Zisserman. ‘Learning local feature descriptors using convex optimisation’. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2014), p. 267.
- [89] Karen Simonyan and Andrew Zisserman. ‘Very deep convolutional networks for large-scale image recognition’. In: 2014, p. 26535.
- [90] Josef Sivic and Andrew Zisserman. ‘Video Google: A text retrieval approach to object matching in videos’. In: 2003.

- [91] Randall C Smith and Peter Cheeseman. ‘On the representation and estimation of spatial uncertainty’. In: *The international journal of Robotics Research* (1986).
- [92] Stephen M Smith and J Michael Brady. ‘SUSAN—a new approach to low level image processing’. In: *International journal of computer vision* (1997), p. 4800.
- [93] Yurun Tian, Bin Fan and Fuchao Wu. ‘L2-net: Deep learning of discriminative patch descriptor in euclidean space’. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, p. 90.
- [94] Engin Tola, Vincent Lepetit and Pascal Fua. ‘Daisy: An efficient dense descriptor applied to wide-baseline stereo’. In: *IEEE transactions on pattern analysis and machine intelligence* (2009), p. 1278.
- [95] Giorgos Toliás, Yannis Avrithis and Hervé Jégou. ‘To aggregate or not to aggregate: Selective match kernels for image search’. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2013, pp. 1401–1408.
- [96] Giorgos Toliás et al. ‘Rotation and translation covariant match kernels for image retrieval’. In: *Computer Vision and Image Understanding* (2015), p. 11.
- [97] Fernando De la Torre and Michael J Black. ‘Robust principal component analysis for computer vision’. In: *International Conference on Computer Vision (ICCV)*. 2001.
- [98] Tomasz Trzcinski et al. ‘Learning image descriptors with the boosting-trick’. In: *Advances in neural information processing systems*. 2012, p. 85.
- [99] Tinne Tuytelaars, Krystian Mikolajczyk et al. ‘Local invariant feature detectors: a survey’. In: *Foundations and trends® in computer graphics and vision* (2008), p. 1843.
- [100] Luc Van Gool, Theo Moons and Dorin Ungureanu. ‘Affine/photometric invariants for planar intensity patterns’. In: *European Conference on Computer Vision*. Springer. 1996, pp. 642–651.
- [101] Andrea Vedaldi and Andrew Zisserman. ‘Efficient additive kernels via explicit feature maps’. In: *IEEE transactions on pattern analysis and machine intelligence* (2012), p. 1057.
- [102] Pauli Virtanen et al. ‘SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python’. In: *Nature Methods* (2020).
- [103] Simon AJ Winder and Matthew Brown. ‘Learning local image descriptors’. In: *2007 IEEE Conference on Computer Vision and Pattern Recognition*. 2007, p. 411.
- [104] John Wright et al. ‘Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization’. In: *Advances in neural information processing systems*. 2009, p. 1387.
- [105] Huan Xu, Constantine Caramanis and Sujay Sanghavi. ‘Robust PCA via outlier pursuit’. In: *Advances in neural information processing systems*. 2010, p. 524.

- [106] Sergey Zagoruyko and Nikos Komodakis. ‘Learning to compare image patches via convolutional neural networks’. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, p. 688.