University of California
Los Angeles

Invariant Representations and Learning for Computer Vision

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Andrea Vedaldi

2008

The dissertation of Andrea Vedaldi is approved.

Serge Belongie

Pietro Perona

Luminita Vese

Alan Yuille

Song-Chun Zhu

Stefano Soatto, Committee Chair

University of California, Los Angeles

2008

To Daniela and Romano, my parents

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to thank first my advisor, Professor Stefano Soatto. His deep and broad views on science and his intellectual creativity have been of great inspiration for my research work. Many thanks go to the members of my commitee, Professors Alan Yuille, Song-Chun Zhu, Luminita Vese, Serge Belongie and Pietro Perona, and the former member Professor Adnan Darwiche, for their support and suggestions, both scientific and personal.

The "historical" member of the UCLA Vision Lab, Alessandro Bissacco, Gianfranco Doretto, Hailin Jin, and Paolo Favaro, supported me with their friendship and their professional advice in the first and most difficult years of my doctoral studies. Special thanks go to Alessandro Bissacco, for hosting me through all these years, and to Paolo Favaro, for sharing his original ideas with me and coauthoring some of my best works. Many thanks go to Payam Saisan for inspiring me with his amazing ability to withstand even the most difficult situations.

I would also like to thank the many people I met during my staying at the UCLA Vision Lab: Alessandro Duci, Daniel Cremers, Fabio Cuzzolin, Jae-Young Choi, Siddharth Manay, Daniele Fontanelli, Gregorio Guidi, Byung-Woo Hong, Nicola Moretto, Emmanuel Prados, and René Vidal. They greatly inspired me with their intellectual and scientific abilities. Many thanks go to my co-authors Gregorio Guidi and Andrew Rabinovich for our great collaborations. Thanks go to Professor Judea Pearl: It was both a pleasure and an honor to sit through his wonderful classes. I express my gratitude to Professor Ruggero Frezza, my former advisor, whose support was fundamental for my coming to UCLA for my doctoral studies.

I would like to thank all the younger members of the UCLA Vision Lab: Brian Fulkerson, Eagle Jones, Teresa Ko, Yifei Lou, Jason Meltzer, Taehee Lee, Jeremi Sudol, Michalis Raptis, Kamil Wnuk and Zhao Yi. It was a pleasure to work among so many bright people. Special thanks go to Brian Fulkerson and Eagle Jones for our collaborations.

Finally, I would like to acknowledge the unconditional support of my mother and my father, which made all this possible.

Vita

1979	Born in Verona, Italy
1998–2003	D. Eng., Information Engineering, University of Padova, Italy (maximum honors).
2003–2006	M.S., Computer Science, University of California at Los Angeles, USA (4/4 GPA).
2006	Teaching Assistant, University of California at Los Angeles, USA
2006	UCLA Outstanding Master Award

Publications and Presentations

A. Vedaldi and S. Soatto, "Relaxed Matching Kernels for Object Recognition," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition* (CVPR), 2008.

A. Vedaldi and S. Soatto, "Joint Alignment up to (Lossy) Transformations," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition* (CVPR), 2008.

A.Vedaldi, P. Favaro, and E. Grisan, "Boosting Invariance and Efficiency in Supervised Learning," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2007 (oral presentation).

A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, and S. Belongie, "Objects in Context," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2007 (oral presentation).

E. Jones, A. Vedaldi and S. Soatto, "Inertial structure from motion with autocalibration", in *Proceedings of the ICCV Workshop on Dynamical Vision*, 2007.

A. Vedaldi, G. Guidi, and S. Soatto, "Moving Forward in Structure From Motion," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition* (CVPR), 2007.

A. Vedaldi and S. Soatto, "A Complexity-Distortion Approach to Joint Pattern Alignment," in *Advances in Neural Information Processing Systems* (NIPS) 19, 2007.

A. Vedaldi and S. Soatto, "Local Features, All Grown Up," in *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition* (CVPR), pp. 1753-1760, 2006 (oral presentation).

A.Vedaldi and S. Soatto, "Viewpoint Induced Deformation Statistics and the Design of Viewpoint Invariant Features: Singularities and Occlusions," in *Proceedings of the European Conference on Computer Vision* (ECCV), Vol. 2, p. 374, 2006.

A.Vedaldi and S.Soatto, "Features for recognition: Viewpoint invariance for non-planar scenes," in *Proceedings of the International Conference on Computer Vision* (ICCV), 2005 (oral presentation).

A.Vedaldi, H.Jin, P.Favaro, and S.Soatto, "KALMANSAC: Robust filtering by consensus," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2005.

Presentation: A.Vedaldi, P. Favaro, and E. Grisan, "Boosting Invariance and Efficiency in Supervised Learning," presented at the *International Conference on Computer Vision (ICCV)*, 2007.

Presentation: "Local Features, All Grown Up," presented at the *IEEE Conf. on Computer Vision and Pattern Recognition* (CVPR), 2006.

Presentation: "Features for recognition: Viewpoint invariance for non-planar scenes," presented at the *International Conference on Computer Vision* (ICCV), 2005.

Presentation: "Viewpoint Invariant Features: Why it works, what to do when it does not," presented at Berkeley, UCSD, Padua, Venice, and Verona Computer Science Depratments, 2007.

Abstract of the Dissertation

Invariant Representations and Learning for Computer Vision

by

Andrea Vedaldi
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2008
Professor Stefano Soatto, Chair

In computer vision one is often interested in extracting from images only a few of the many physical parameters that are entangled in the formation of such images. For instance, one may ask whether an image portrays a certain object regardless the illumination and viewpoint conditions. Eliminating the effects of irrelevant factors is therefore an integral part of most vision algorithms. In this manuscript we discuss two ways of doing so: Extracting invariant representations from images and learning with invariance constraints. In particular, in the first part of the thesis we prove the existence of general local viewpoint invariant features, propose a method to improve their distinctiveness by optimizing their support, introduce a dataset and formalism to empirically evaluate invariant representations, and propose joint data alignment to automatically learn canonical representations. In the second part, we modify the AdaBoost learning framework to operate efficiently under invariance constraints, we introduce a general family of positive definite kernels that can be used to trade off invariance and distinctiveness of image representations, and we propose quick shift, a novel and efficient clustering algorithm.

Finally, the appendix discusses applications to the problem of structure from motion. We introduce KALMANSAC, a very robust filtering method for on-

line structure from motion, and we discuss theoretical aspects of structure from forward motion.

CHAPTER 1

Introduction

Many tasks in computer vision require extracting from images only a few categorical properties, such as the presence or absence of a certain object ("is there a refrigerator?") or the category of a scene ("is this a kitchen?"). Even when we are interested in estimating more detailed parameters, such as the outline and pose of an object and its relationship with the context, most of the physical factors that are entangled in the formation of images are irrelevant. We may not care about the exact color, texture, or shape of an object. We also may not be interested in recovering the viewpoint from which it is seen. Finally, the illumination conditions are likely to be unimportant to us. Nevertheless, an image depends crucially on all such parameters and it is simply not possible to ignore them in analysis.

The larger part of this thesis explores methods to explicitly factor out the effects of the irrelevant imaging parameters in the design of vision algorithms. Chapters 2 through 5 study how and to which extent this can be obtained by pre-processing images and extracting invariant representations (see also Section 1.1). In Chapter 2 we prove the existence of generic viewpoint invariant local features and study their properties. In Chapter 3 we address the limitations arising from the locality of viewpoint invariant representations and we propose a method to lift them. In Chapter 4 we study how to evaluate and learn from data invariant representations, based on accurate ground truth. In Chapter 5 we propose a complexity-based approach for the construction of canonical invariant features.

Chapter 6 departs from the pre-processing based methods and focuses on how to exploit directly the invariant structure of the visual data during analysis (refer also to Section 1.2). In particular, the chapter proposes a modification of the popular AdaBoost algorithm that incorporates invariance at the level of the weak learners, resulting in a powerful and efficient learning algorithm. In Chapter 7 we study a family of positive definite kernels useful to trade off invariance and discriminative power of image representations derived from the popular bag-of-features model. In Chapter 8 we study a clustering algorithms that extend the generality and improve the efficiency of the popular mean shift paradigm.

The appendices include two contributions dedicated to applications in the field of of Structure From Motion. In Appendix A we propose a robust filtering

scheme suitable for on-line structure from motion which combines the popular RANSAC algorithm and the Kalman filter. Appendix B deals with the problem of singularities in structure from forward motion. Finally, in Appendix C we describe our open source implementation of the popular SIFT feature detector and descriptor that we made available to the community.

The following sections serve as a brief introduction to the concept of invariance and illustrate how the material of the manuscript fits in this context.

1.1 Invariant representations

One of the main topics of this manuscript is image representations for computer vision. The goal of a representation is to simplify images to the end of solving a given vision problem, such as recognizing the presence of a certain object or identifying selected points of a 3-D scene. In this sense, it may seem that the best representation is the one that removes the effects of all the irrelevant physical parameters from an image, returning a quantity which directly encodes the solution of the problem. For instance, the ideal representation for recognizing the presence of an object would be a single bit, which is in one-to-one correspondence with the answer. Thus, why do we distinguish representations from solutions?

The reason is that representations are general: they are preprocessing procedures which help obtaining, but do not provide directly, the solution to different problems (such as recognizing different objects). An important example are the so called "invariant features". An *invariant feature* $\phi(x)$ is a function of a data point $x \in \mathcal{X}$ such that, if t is a transformation in some family $\mathcal{T} \subset \mathcal{X}^{\mathcal{X}}$, then $\phi(tx) = \phi(x)$. Applied to images, an invariant feature is just an image statistic (i.e., a function of the image pixels) that is invariant to a well defined set of image transformations, such as deformations of the image domain or rescalings of its range. In our context, invariant features are useful when the image variability induced by some of the nuisance parameters can be captured in terms of transformations of the images themselves. The most important case is viewpoint invariance, extensively studied in Chapter 2.

Of course, it is not enough for a feature to be invariant (otherwise any constant function ϕ would do!). In fact, the feature should also embody the information useful to answer a question of interest. We call such a feature *sufficient* or, when referring to a classification or categorization problem, *sufficiently discriminative*. This concept can be formulated in at least two related ways, which are illustrated next by an example. Consider the problem of finding the category $c \in \{1, \dots, C\}$ of the object portrayed by an image x.

Functional sufficiency. The object category $c = c(x)$ is given as a function

of the image x. The feature ϕ is *sufficient* if, whenever two images x and y portray objects of different categories $c(x) \neq c(y)$, the representations $\phi(x) \neq \phi(y)$ are different as well.

Statistical sufficiency. The object category c is obtained by solving a *statistical inference problem*. The image x is "generated" by the object category c and a nuisance parameter ν through a model specified by the log likelihood function $l(x; c, \nu)$. The nuisance ν includes irrelevant details, such as the viewing conditions and clutter, and can be eliminated[1] by considering the reduced model [Rem84] $\bar{l}(x, c) = \sup_\nu l(x; c, \nu)$. The feature ϕ is *sufficient* if, whenever the function $\bar{l}(x, \cdot) - \bar{l}(y, \cdot)$ is not constant for two images x and y, the representations $\phi(x) \neq \phi(y)$ are different as well.

While sufficiency is a desirable property for a feature, sometimes we prefer to trade it off for other useful properties. For instance, constructing invariant features requires assumptions about the image formation that are usually not satisfied globally. In fact, in Chapter 2 we discuss *local* invariant features which are extracted from image regions or *patches*. Local features, considered in isolation, are usually not sufficient to solve a complex problem such as recognizing an object category, and a combination of them must be considered (e.g., a bag-of-features (Chapter 7)). An important example are the so called "viewpoint invariant local features", which characterize image patches independently of the viewpoint from which they are seen. In Chapter 2 we show that such features are sufficient to discriminate local portions of the scene based on the albedo (texture).

A alternative criterion to sufficiency is to look for features that are *maximally discriminative* [Ber85], in the sense that $\phi(x) = \phi(y)$ exactly when y can be obtained from x by a transformation $t \in \mathcal{T}$. In this case, the feature eliminates only the effects of the transformations $\mathcal{T}$, but nothing else. This criterion is useful in two regards: (i) differently from sufficiency, it is not bound to a specific problem, and (ii) if the transformations $t \in \mathcal{T}$ are irrelevant for solving the problem $c(x)$,[2] then a maximally discriminative feature $\phi(x)$ is sufficient for that problem.

Example 1 (Invariant, sufficient, and maximally discriminative features). Let $c(x) = \chi_{\{x_1 > 0\}}$ be the problem of deciding whether the first coordinate x_1 of a point $x \in \mathcal{X} = \mathbb{R}^3$ is greater than zero. Let $tx = (x_1, x_2 + t, x_3)$ be a one parameter family of translations $t \in \mathcal{T} = \mathbb{R}$. Then:

- $\phi(x) = x$ is sufficient but not invariant.

[1] In practice, eliminating the nuisance parameter in a way which is statistically satisfactory can be harder [KS70, Bas77]. Here we adopt this method only because it is the quickest route to illustrate the concept of sufficiency.

[2] I.e. if $c(x) = c(tx)$ for all $t \in \mathcal{T}$.

- $\phi(x) = x_1$ is both sufficient and invariant, but not maximally discriminative.
- $\phi(x) = (x_1, x_3)$ is sufficient, maximally discriminative, and invariant.

Notice that the maximally discriminative features is also sufficient for the problem $c(x) = \chi_{\{x_3>0\}}$.

1.1.1 Canonization

Invariant and maximally discriminative features are completely characterized by the following:

Lemma 1. *Let $\mathcal{T}$ be a group of transformations*[3] *acting on the set $\mathcal{X}$ and consider the* orbit $\mathcal{T}x = \{tx : t \in \mathcal{T}\}$. *Then*

- *$\phi(x) = \mathcal{T}x$ defines an invariant maximally discriminative feature.*
- *Any other invariant maximally discriminative feature $\phi'(x)$ is equal to $\phi(x)$ up to a bijection.*

Proof. This is a well known result from algebra. A rigorous and elegant proof, based on the concept of universal constructions, can be found in the classic [MB99]. Intuitively, $\mathcal{T}x$ is an equivalence class of the equivalence relation $x \equiv y \Leftrightarrow \{\exists t : y = tx\}$. By definition of maximally discriminative feature, the condition $\phi(x) = \phi(y)$ defines the same equivalence relation. Hence the functions $\mathcal{T}x$ and $\phi(x)$ associate unique labels to the same classes and are equal up to a bijection. □

While sound, adopting $\mathcal{T}x$ as a feature is in most cases impractical. For instance, if $\mathcal{T}$ are image rotations and x is an image patch, then $\mathcal{T}x$ is the set of all rotated versions of x. Fortunately, another conceptual construction shows that a far more manageable feature can be used in place of $\mathcal{T}x$.

Lemma 2 (Canonization)**.** *Let $\mathcal{T}$ be a group of transformations acting on $\mathcal{X}$. Let a* canonization $\mathcal{E} \subset \mathcal{X}$ *be a set that contains exactly one element for each equivalence class $\mathcal{T}x$. Then the function $e(x)$ which maps x to the unique element of the intersection $\mathcal{T}x \cap \mathcal{E}$ defines an invariant maximally discriminative feature.*

Proof. If $S = \mathcal{T}x = \mathcal{T}y$, then $e(x) = e(y)$ because $\mathcal{E}$ contains only one element of the subset S by hypothesis. Moreover, if $e(x) = e(y)$, then $\mathcal{T}x$ and $\mathcal{T}y$ are equivalence classes that share a point and are therefore identical. □

[3] *A set G with an operation $\circ$ is a group if: (associativity) $g_1 \circ (g_2 \circ g_3) = (g_1 \circ g_2) \circ g_3$, (identity) there exists $e \in G$ such that $e \circ g = g \circ e = g$, and (inverse) if $g \neq e$, then there exists g^{-1} such that $g^{-1} \circ g = g \circ g^{-1} = e$. Here g_1, g_2, g_3, g are generic elements of G.*

Example 2 (Continuing Example 1). The maximally discriminative feature $\phi(x) = (x_1, x_3)$ is equal to $\mathcal{T}x = \{x_1\} \times \mathbb{R} \times \{x_3\}$ up to the bijection $(x_1, x_3) \mapsto \{x_1\} \times \mathbb{R} \times \{x_3\}$, as predicted by Lemma 1. Choosing $\mathcal{E} = \{(x_1, 0, x_2)\}$ yields by canonization (Lemma 2) the invariant maximally discriminative feature $\phi(x) = (x_1, 0, x_3)$. Similarly, $\mathcal{E} = \{(x_1, \sin(x_1) + \sqrt{2}x_2, x_2)\}$ yields by canonization the feature $\phi(x) = (x_1, \sin(x_1) + \sqrt{2}x_2, x_2)$.

To summarize, there exists a simple conceptual construction of invariant and maximally discriminative features:

1. Partition the data space $\mathcal{X}$ into equivalence classes, based on the transformation group $\mathcal{T}$.
2. For each equivalence class $\mathcal{T}x$, choose a canonical element $e(\mathcal{T}x) \in \mathcal{T}x$, yielding a canonical set $\mathcal{E}$.
3. Let the feature $\phi(x)$ be the projection $e(\mathcal{T}x)$ of x on the canonical set $\mathcal{E}$.

1.1.2 Viewpoint invariant local features

Chapter 2 studies how the ideas introduced in the previous section apply to the construction of local image features that are invariant to viewpoint change. A famous result [BWR92] seems to suggest that no such construction is possible if the 3-D shape of the scene or the change of viewpoint are generic. Thus it may be surprising that we are able to prove the existence of exactly such generic invariants. This is not a contradiction. The result from [BWR92] disregards any photometric information attached to images. We show that, if the photometric information is exploited, this construction is possible.

A limitation of viewpoint invariant features is the fact that they are, by nature, local. This is due to the fact that assumptions required for invariance, such as the necessity of avoiding occlusions (Chapter 2), are usually not satisfied globally. Unfortunately, the locality of such representations diminishes their discriminative power; in Chapter 3 we propose a technique to optimize the support of the local features, and increase their distinctiveness.

In Chapter 4 we move to the problem of evaluating empirically local features. This is a difficult task for several reasons. First, evaluation requires accurate ground truth which is difficult to obtain from physical measurements alone. Second, even if ground truth is available, one must define what is expected from the various components of a feature (detector, descriptor, similarity metric) when the working assumptions are not satisfied. This process is essential to rationally assess the robustness of each component to deviation from the idealized assumptions.

In Chapter 5 we explore the technique of joint data alignment as a means of learning canonical representations from data. Canonization is a fundamental technique for the construction of invariant features (Section 1.2). It amounts to mapping images to standard configurations, removing in the process the effect of the nuisance transformations. While the design of such canonical configurations is hard to do by hand, joint alignment can be used to do so automatically.

1.2 Invariant learning

While invariance can be incorporated in preprocessing, this comes with a number of possible disadvantages. For instance, common invariant representations are local, and hence capture only partially the information conveyed by an image. Therefore we are interested in analysis methods that can leverage on invariance properties directly.

Consider for instance the problem of constructing a function $c(x)$ that detects the presence or absence of a face in an image x. Learning is the process of finding such a function c in a large collection $\mathcal{C}$ of analogous functions, based on a limited number of empirical observations of images that do and do not portray a face. The difficulty is that the limited examples are often insufficient to assess the general performance of c, potentially yielding a poor choice of the function. This problem, known as "over-fitting", can be alleviated by constraining the choice with additional information about the structure of the data. Invariance is a natural candidate, as it directly translates into a set of constraints on c. For instance, if x portrays a face, and if y is an image obtained from x by perturbing the viewpoint, then we obtain the constraint $c(x) = c(y)$. In general, however, enforcing such constraints is computationally quite demanding, if not unfeasible. In Chapter 6 we propose a technique that enables incorporating invariance in the popular boosting framework for classification and regression. Inspired by vicinal risk and tangent distance, we introduce invariance at the level of the weak classifiers and we show that this yields an efficient and powerful learning algorithm.

There is yet an alternative approach for exploiting invariance in learning. Instead of building up invariance starting from the raw data, one can start from a representation that is invariant to a large set of transformations and then gradually reduce invariance until the right trade-off with discriminative power is achieved. Notable examples are recent extensions to the popular bag-of-features representation for object recognition. A bag-of-feature representation is an orderless collection of local invariant features, as the one discussed in Chapter 2. It is fully invariant to arbitrary permutations of the features, which provides, among other things, insensitivity to viewpoint change. Recently new representations

have been proposed that extend bag of features to incorporate a certain degree of spatial information, with the result of diminishing invariance and increasing distinctiveness. Such modifications have been formulated as positive definite kernels, that can be readily used in support vector machines and other powerful classifiers. In Chapter 7 we propose a general framework that subsumes and extends these particular cases. The framework highlights interesting properties shared by such methods and enables the systematic construction of new ones.

Finally, Chapter 8 proposes an extension to the popular mean-shift clustering algorithm, which seeks the local modes of the data distribution in kernel space. Similarly to the popular k-means algorithm, clustering can be used to devise invariant or insensitive representations, such as the visual words used in the popular bag-of-features model.

1.3 Applications

The appendix includes applications in the field of Structure From Motion (SFM). The goal of SFM is to reconstruct the 3-D structure of a set of points and the motion of the camera from the projections of those points on the moving camera plane. Extracting such points from images can be done by the technique discussed in Chapter 2.

In Appendix A we study the problem of on-line structure from motion. Here the goal is to progressively update the estimate as more images from a video sequence are received. While this is usually formulated as a filtering problem, the high rate of outliers, caused by the unreliability of point extraction and tracking, requires adopting robust filtering techniques. The chapter introduces a new filter, based on RANSAC and Kalman filtering, which can handle unprecedented rates of outliers.

Finally, in Appendix B we address the problem of structure from forward motion. This is a particularly important case, as often the line of sight and motion direction coincide. Unfortunately, estimation in this particular case is subject to a large number of singularities, which are the focus of the chapter.

CHAPTER 2

Existence of General Viewpoint Invariant Local Features

Visual classification plays a key role in a number of applications and has received considerable attention in the community during the last decade. The fundamental question is easy to state, albeit harder to formalize: when do two or more images "belong to the same class"? A class reflects some commonality among scenes being portrayed [FFP04, FHI00, LSP04, FPZ03]. Classes that contain only one element are often called "objects," in which case the only variability in the images is due to extrinsic factors – the imaging process – but there is no intrinsic variability in the scene. Extrinsic factors include illumination, viewpoint, and so-called clutter, or more generally visibility effects. Classification in this case corresponds to recognition of a particular scene (object) in two or more images. In this chapter we restrict ourselves to object recognition. While this is considerably simpler than classification in the presence of intrinsic variability, there are some fundamental questions yet unanswered: What is the "best" representation for recognition? Is it possible to construct features that are viewpoint-invariant for scenes with arbitrary (non-planar) shape? If so, are these discriminative?

The non-existence of general-case view invariants [BWR92] has often been used to motivate local descriptors, for instance affine invariants. The results of [BWR92], however, pertain to collections of points in 3-D space with no photometric signature associated to them. Since we measure image intensity, we show that *viewpoint invariance can be achieved for scenes with arbitrary (continuous) shape, regardless of their albedo,* under suitable conditions which we outline in Section 2.2. While this result by no means undermines the importance of affine descriptors, we believe it is important to state it precisely and prove it for the record, which we do in Theorem 1. The flip-side of general-case viewpoint invariants is that *they necessarily sacrifice shape information,* and therefore discrimination has to occur based solely on the photometric signature (Section 2.3.1). This result is straightforward to prove (Prop. 3), but since nobody has done so before, we believe it is important. It also justifies the use of "bags of features" in handling viewpoint variations [DS04]. Finally, we show that if viewpoint is factored out as part of the matching process, rather than in the representation, then *shape information is retained,* and can be used for discrimination (Prop. 4).

This may contribute to the discussion following [SM71] in the psycho-physical community. On illumination invariants, [CBJ00] showed that even for Lambertian scenes they do not exist. While they used a point light source model, diffuse illumination is perhaps a more germane assumption for cloudy days or indoor scenes, due to inter-reflections [LZ93]. Our results can be easily extended to incorporate invariance to such a first-order model of Lambertian reflection in diffuse illumination.

For the benefit of the reader that is unappreciative of theory alone, we illustrate our results with simple experiments that show that even a naive 3-D viewpoint invariant can support matching whereas current affine descriptors fail (Section 2.4). Of course, existing descriptors only fail at singularities, so our work serves to validate existing methods where appropriate, and to complement them where their applicability is limited. The point of this section is *not* to advocate use of our detector/descriptor as a replacement of existing ones. It only serves to illustrate the theory, and to point out that some of the restrictions imposed on existing methods may be unnecessary.

2.1 Generalized correspondence

The simplest instance of our problem can be stated as follows: *When do two (or more) images portray (portions of) the same scene?* Naturally, in order to answer the question we need to specify what is an image, what is a scene, and how the two are related. We will make this precise later; for the purpose of this introduction we just use a formal notation for the *image* I and the *scene* ξ. An image I is obtained from a scene ξ via a certain function(al) h, that also depends on certain *nuisances* ν of the image formation process, namely viewpoint, illumination, and visibility effects. With this notation we say that two images are in *correspondence*[1] if there exists a scene that generates them

$$I_1 \leftrightarrow I_2 \;\Leftrightarrow\; \exists\, \xi \mid \begin{cases} I_1 = h(\xi, \nu_1) \\ I_2 = h(\xi, \nu_2) \end{cases} \tag{2.1}$$

for some nuisances ν_1, ν_2. *Matching*, or deciding whether two or more images are in correspondence, is equivalent to ascertaining the existence of a scene ξ that generates them all, for some nuisances $\nu_i, i = 1, 2, \ldots$. These (viewpoint, illumination, occlusions, cast shadows) could be *estimated explicitly* as part of the matching procedure, akin to "recognition by reconstruction," or they could be *factored out* in the *representation*, as in "recognition using features." But what

[1]Note that there is no locality implied in this definition, so correspondence here should not be confused with point-correspondence.

is a *feature*? And why is it useful for matching? We will address these questions in Section 2.2.

In the definition of correspondence the "=" sign may seem a bit strong, and it could certainly be relaxed by allowing a probabilistic notion of correspondence. However, even with such a strong requirement, it is trivial to show that any two images can be put in correspondence, making this notion of correspondence meaningless in lack of additional assumptions. Probabilistic assumptions (i.e. priors) require endowing shape and reflectance with probability measures, not an easy feat. Therefore, we choose to make *physical assumptions* that allow us to give a meaningful answer to the correspondence problem. This problem naturally relates to wide-baseline matching [PZ98, FTV03, FTV03, DZZ04, KRS04].

2.2 Existence of general viewpoint invariant local features

In Section 1.1 we defined an image feature $\phi(I)$ as any function(al) of the image I. Of all features, we are interested in those that facilitate establishing correspondence between two images I_1, I_2, or equivalently recognition of the scene ξ (Section 2.1). This requires handling the nuisance ν: A feature ϕ is *invariant to the nuisance* ν if its value does not depend on the nuisance: $\phi(I) = \phi \circ h(\xi, \nu) = \phi \circ h(\xi, \mu) \; \forall \; \nu, \mu$. Naturally, in order to solve the generalized correspondence problem 2.1, we also need the feature to be discriminative for the scene ξ (this notion will be made precise later).

In Section 1.1 the concept of invariance was defined in term of transformations of the image, such as deformations of its domain or rescalings of its range. Unfortunately, not all variations of the nuisance parameter ν can be captured in term of image transformations. In fact, predicting the scene appearance resulting from different viewing conditions ν requires, in general, the knowledge of the scene ξ, and the latter cannot be inferred from an image I alone. However, under suitable conditions the effects of the nuisance factors *can* be captured in such a way. For instance, if the camera is limited to rotate around its axis, then the effect of a variation of viewpoint is captured by a rotation of the image.

The first part of the chapter establishes conditions for which general changes of viewpoint can be captured in term of a family of image transformations, thus enabling the construction of invariant features based on the principles illustrated in Section 1.1. In particular, we will see that, if the scene is Lambertian and if the viewpoint change does not activate occlusions, then the image transforms according to an homeomorphism (Sections 2.2.1 and 2.2.2). This fact, together with the results of Section 1.1, essentially proves the existence of features invariant to generic viewpoint changes, for arbitrary 3-D shapes of the underlying scene ξ

(Section 2.2.3). These features are also *albedo-discriminative*, in the sense that they discriminate between objects with different albedo (texture).

The main issue with this simple construction is the fact that the non-occlusion requirement is too strong for such features to be useful in practice. In fact, not only the surface S must not occlude itself as the viewpoint varies, but it must also be fully visible inside the image domain. The latter requirement is especially stringent, because S lumps together all the objects in the scene. The necessity of avoiding occlusions is the main reason why we move our attention to *local features*, which are computed on subsets Ω of the image domain. We show that our construction extends to general viewpoint invariant *local* features, under the condition that the domain Ω is extracted in a special way (Prop. 1). Namely, the selection of Ω must be *co-variant* with the image transformations. Therefore Section 2.2.4 is dedicated to prove the existence of a co-variant detector, i.e. of a mechanism that, given the image I alone, is able to extract a countable (in practice, finite) number of co-variant regions Ω. This finally leads us to the main result of the chapter, i.e. the existence of generic viewpoint invariant local features (Thm. 1).

In Section 2.3.1 we show a limitation of generic viewpoint invariant features: Because such features must absorb the image deformations induced by a viewpoint change, the information about the underlying shape is lost. So, while features are albedo-discriminative, they are *not* shape-discriminative. We also discuss the discriminative power of such feature in relation to the concept of maximal discriminability introduced in Section 1.1.

2.2.1 Image formation: Lambertian scenes

While global correspondence can be computed for scenes with complex reflectance under suitable assumptions, local correspondence cannot be established in the strict sense defined by (2.1) unless the scene is Lambertian, and even then, it is necessary to make assumptions on illumination to guarantee uniqueness [CBJ00]. In particular, one can easily verify that if the illumination is assumed to be constant (ambient, or "diffuse") then local correspondence can be established. We therefore adopt such assumptions and relegate all non-Lambertian effects as "disturbances."

We can now make the formal notation of Section 2.1 more precise: We represent an image as function $I : \Lambda \subset \mathbb{R}^2 \longrightarrow \mathbb{R}_+;\ x \mapsto I(x)$ defined on a compact domain Λ. A Lambertian scene is represented by a collection of (piecewise smooth) surfaces, as detailed by the following definition.

Definition 1 (Image Formation Model)**.** A *scene* (S, ρ) is a piecewise smooth surface $S \subset \mathbb{R}^3$ (not necessarily simply connected) together with a map $\rho : S \rightarrow$

$\mathbb{R}_+$ called *albedo*. A *viewpoint* $g \in SE(3)$ is a rigid motion that maps points $X \in \mathbb{R}^3$ in the inertial reference frame to the corresponding points gX in the camera reference frame. The *camera projection* π is the function that maps points $X \in \mathbb{R}^3$ (in the camera reference frame) on the image plane $\Lambda \subset \mathbb{R}^2$ by

$$\pi(X) \triangleq \frac{1}{X_3} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix}.$$

The *inverse camera projection* $\pi^{-1}(x)$ is the map that back-projects pixels $x \in \Lambda$ to the nearest surface point X, that is to the points X which verify

- $X \in S$,
- $\pi(gX) = x \ \wedge \ e_3^\top gX \geq 0$ (positive depth),
- $\forall \tilde{X} \in S : x = \pi(g\tilde{X}) \ \wedge e_3^\top g\tilde{X} \geq 0 \ \Rightarrow \ e_3^\top gX \leq e_3^\top g\tilde{X}$ (first intersection).

Occasionally, to emphasize the dependency of the inverse projection on the scene and viewpoint, we write $\pi^{-1}(x) = \pi^{-1}(x; S, g)$. The *image formation model* $I = h(S, \rho, g)$ maps the scene (S, ρ) and the viewpoint g onto an image $I(x)$ according to

$$I(x) = h(S, \rho, g) = \rho(\pi^{-1}(x; S, g)). \tag{2.2}$$

With the notation of Section 2.1, the scene is $\xi = (S, \rho)$ and the nuisance is limited to the motion $\nu = g$.

Remark 1 (Extension to ambient illumination). In the following discussion we will concentrate mainly on the model (2.2) for notational simplicity, but the results can easily be extended to the case of ambient illumination. In this case, to first approximation[2] equation (2.2) becomes

$$I(x) = \alpha\rho(\pi^{-1}(x; S, g)) + \beta \tag{2.3}$$

for an affine transformation $(\alpha, \beta) \in \mathbb{R}^2_+$ of the range of the image. With respect to the model (2.2), the nuisance here comprises both viewpoint and illumination, that is $\nu = (g, \alpha, \beta)$.

2.2.2 The deformation induced by a viewpoint change

As one can easily realize, it is impossible to construct non-trivial statistics that are invariant to occlusions. For this reason viewpoint invariant features make the assumption that no occlusion occurs as the viewpoint changes. Although this

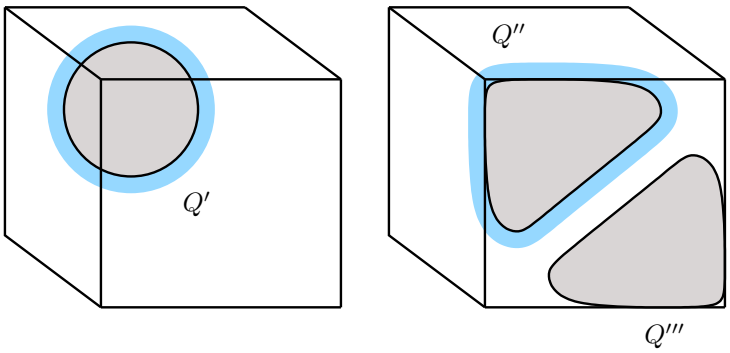


Figure 2.1: **Visible patches.** The patch Q' of the surface of the cube is visible as it projects injectively on the image plane. It is also away from boundaries as there exists an open neighborhood (in blue) that contains the closure of Q' and is itself visible. The patch Q'' is again visible and away from occluding boundaries. Note that the fact that $\partial Q''$ intersects the singularities of the cube (its edges) is inconsequential. Finally, Q''' is visible but *not* away from boundaries. In fact there exists no open neighborhood containing $Q''' \subset S$ which is visible.

is very unlikely for the scene as a whole, it happens frequently enough for local patches.

Definition 2. We say that the surface patch $Q \subset S$ is *visible* from the viewpoint $g \in SE(3)$ if, and only if, the map $\pi(gX)$, $X \in Q$ admits a left inverse and this is the function $\pi^{-1}(x; S, g)$, $x \in \Omega$ (Fig. 2.1.)

Equivalently the patch Q is visible from g if, and only if, we have

$$\pi^{-1}(\pi(gX); S, g) = X, \qquad X \in Q.$$

Sometimes we need the patch to be visible and away from occluding boundaries. This notion is captured by the next definition.

Definition 3. We say that the surface patch Q is visible and *away from (occluding) boundaries* if there exists an open neighborhood of the closure of Q which is itself visible (Fig. 2.1).

A summary of the symbols is provided in Fig. 2.3. The appearance of the scene patch Q changes as we change the viewpoint. If the patch is visible from all viewpoints, its appearance transforms according to an homeomorphism.

[2]More precisely, the radiance at $p \in S$ is given by $R(p) \doteq \rho(p) \int_{V_p} \langle \nu_p, \lambda \rangle dA(\lambda)$ where ν_p is the normal and V_p the visibility cone at p and dA is the area form on the light source [LZ93].

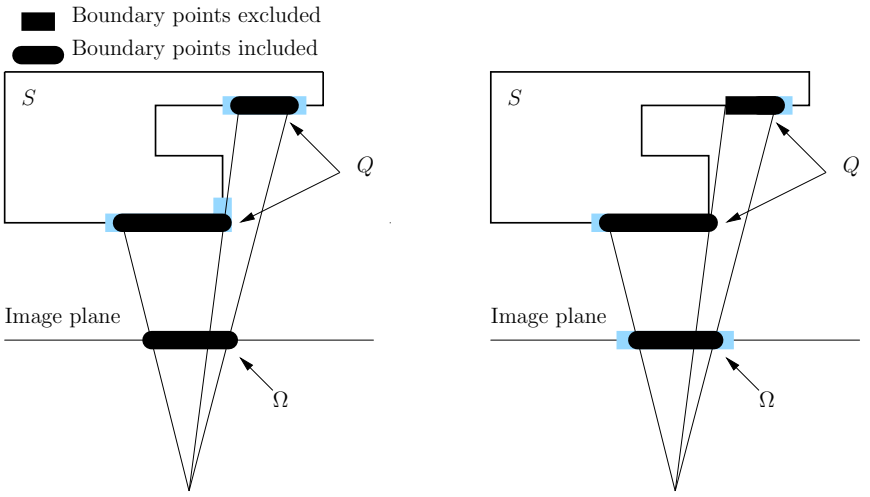


Figure 2.2: **Details of the proofs.** The black segments represent respectively a surface patch $Q \subset S$ and the corresponding image region $\Omega = \pi(gQ)$. Note that Q needs *not* be connected. Left: in Def. 3 the region is is "not away" from an occluding boundary if no open neighborhoods of the closure of the Q is visible. Right: the definition would fail if the open neighbor is taken on the image plane as opposed to the surface. In Prop. 3 we make the assumption that Q is compact. Right: if Q is not compact, the projection π can be continuous on Q and invertible, but the inverse $\pi^{-1}(\cdot; S, g)$ can be discontinuous.

$\Lambda \subset \mathbb{R}^2$	Image domain.
$\Omega, \Omega_t, t = 1, 2$	Regions (subsets) of the image domain Λ.
$S \subset \mathbb{R}^3$	Surface.
$Q \subset S$	Patch (subset) of a surface.

Figure 2.3: **Notation.**

Lemma 3 (Viewpoint induced deformation)**.** *Let I_1 and I_2 be two images obtained from the same scene (S, ρ) and possibly different viewpoints g_1 and g_2. If the compact surface patch $Q \subset S$ is visible from both viewpoints g_1 and g_2, projecting on the image regions $\Omega_1 = \pi(g_1 Q)$ and $\Omega_2 = \pi(g_2 Q)$ respectively, then the image patches $I_1(x)$, $x \in \Omega_1$ and $I_2(x)$, $x \in \Omega_2$ are homeomorphic, in the sense that there exists some homeomorphism (or "warp") $w : \Omega_1 \to \Omega_2$ such that*

$$I_1(x) = (I_2 \circ w)(x), \quad x \in \Omega_1.$$

Note that $\Omega_2 = w(\Omega_1)$.

Proof. Recall that an homeomorphism w is a continuous and invertible mapping $\Omega_1 \to \Omega_2$ with continuous inverse. Since Q is visible from g_t, $t = 1, 2$, then the maps $Q \to \pi(g_t Q)$, $X \mapsto \pi(g_t X)$ are continuous[3] and invertible. As, by hypothesis, Q is compact, these conditions are sufficient to conclude that the inverse maps $\pi^{-1}(x; S, g_t)$, $t = 1, 2$ are continuous as well.[4] Thus both $\pi(g_t X)$, $t = 1, 2$ are homeomorphisms.

We complete the proof by noting that

$$w(x) = \pi \circ g_2 \circ \pi^{-1}(x; S, g_1), \qquad x \in \Omega_1,$$

that g_2 is an homeomorphism $Q \to g_2 Q$, $X \mapsto g_2(X)$ and that compositions of homeomorphisms are homeomorphisms. □

Remark 2. We choose to work with scene patches Q rather than directly with image patches Ω because this simplifies the formalism and makes certain pathological cases straightforward (see Fig. 2.2.)

2.2.3 Viewpoint invariant descriptors

Definition 4 (Local descriptor)**.** Consider a compact image region $\Omega \subset \Lambda$. A *local descriptor* is a map ϕ from an image patch $I|_\Omega \triangleq \{I : \Omega \subset \Lambda \to \mathbb{R}^+; x \mapsto I(x)\}$ to some space $\mathcal{F}$. Occasionally we use the shorthand notation $\phi(I; \Omega)$ for $\phi(I|_\Omega)$.

Often a local descriptor is called local feature descriptor or *feature descriptor*, or simply descriptor, and the space $\mathcal{F}$ is called a *feature space*. The following proposition shows that there exist non-trivial viewpoint invariant local descriptors for regions of the scene that are visible from a set of viewpoints.

Proposition 1 (Existence of viewpoint invariant descriptors)**.** *There exists a local descriptor $\phi(I; \Omega)$ with the following two properties:*

- ***Invariance.*** *Given any compact surface patch $Q \subset S$ visible from viewpoint g_1 and g_2, then $\phi(I_1; \Omega_1) = \phi(I_2; \Omega_2)$ where $\Omega_t = \pi(g_t Q)$, $t = 1, 2$.*
- ***Albedo-discriminativeness.*** *Consider two scenes (S_t, ρ_t), $t = 1, 2$ and two compact patches $Q_t \subset S_t$, $t = 1, 2$ visible from viewpoints g_t, $t = 1, 2$. Then $\phi(I_1; \Omega_1) = \phi(I_2; \Omega_2)$ (same descriptors), if, and only if, there exists an homeomorphisms $w : Q_1 \to Q_2$ such that $\rho_1|_{Q_1} = \rho_2|_{Q_2} \circ w$ (same albedos).*

[3] Take Q with the subset topology induced by $\mathbb{R}^3$ and note that, as Q is visible, no singular point of π (that is points of zero depth) is comprised in Q.

[4] In general a continuous and invertible function does not have necessarily continuous inverse, but this is the case if the domain is compact.

Figure 2.4: **Non-triviality of the local descriptor.** The image patch on the left cannot be mapped by an homeomorphisms to the image patch on the right. Although this is obvious, the statement can be proved formally by noting that, for example, the homotopy group of the gray areas differ (because the gray areas have a different number of "holes"). Note also that both patches are delimited by an extremal region and, as such, constitutes a valid example for the co-variant regions of Thm. 1.

Proof. Denote $I_t|_{\Omega_t}$ the image patches $I_t(x)$, $x \in \Omega_t$, $t = 1, 2$ and let ϕ be the orbit of $I_t|_{\Omega_t}(x)$ under the action of the homeomorphism:

$$\phi(I_t; \Omega_t) = \{I_t|_{\Omega_t} \circ w^{-1} : w \in \text{homeomorphism on } \Omega_t\}.$$

By Lemma 3 the patches $I_t(x)$, $x \in \Omega_t$ for $t = 1, 2$ are related by homeomorphisms. As such, they both generate the same equivalence class $\phi(I_1; \Omega_1) = \phi(I_2; \Omega_2)$, which proves the invariance of the descriptor.

In order to prove that the descriptor $\phi(I; \Omega)$ is albedo-discriminative we note that the albedo ρ_t can be identified with its projection $I_t|_{\Omega_t}$ up to an homeomorphism $\pi^{-1} : \Omega_t \to S_t$, as

$$I_t(x) = \rho(\pi^{-1}(x; S, g_t)), \qquad x \in \Omega_t.$$

Then we observe that, by definition, $\phi(I_1; \Omega_1) = \phi(I_2; \Omega_2)$ if, and only if, there exists an homeomorphism $w : \Omega_1 \to \Omega_2$ such that $I_1|_{\Omega_1} = I_2|_{\Omega_2} \circ w$. □

Remark 3 (Non-triviality). The descriptor of Prop. 1 is albedo-discriminative, but this alone does not imply that it is not trivial. In order to prove this, we have to check that there exist albedos which are *not* equivalent up to homeomorphism. This is of course very easy to check and is left to Fig. 2.4.

Although the descriptor $\phi(I; \Omega)$ constructed in Prop. 1 is viewpoint invariant, it is unpractical because its values are infinite collections of patches. However, a more efficient descriptor can be constructed by finding a *canonical* way to select a representative of the equivalence class.

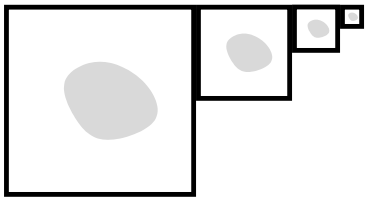


Figure 2.5: **Invariance to scale and finiteness.** Any scale invariant region detector cannot be intrinsically finite. To see this, consider an object on which the detector selects a region Ω. By replicating the object at decreasing scales, we can fit a countable number of such regions into the image. Note that this example applies to any scale invariant detector and not only to the one discussed here.

Corollary 1 (Invariance by canonization)**.** *There exists an invariant and albedo-discriminative local descriptor in the sense of Prop. 1 which has the same "complexity" as the patch $I|_\Omega$. This is obtained by mapping the equivalence class $\phi(I;\Omega)$ onto a canonical patch $I_e|_{\Omega_e}$.*

2.2.4 Co-varaint detection

In order to compute the local feature $\phi(I;\Omega)$ as specified in Def. 4, we need to specify its domain Ω. To satisfy the hypothesis of Prop. 1, we need to select regions Ω_t on each image $t = 1, 2$ in such a way that they are projections of the same portion of the scene regardless of the viewpoint. This process is called *co-variant detection*. The domain is often referred to as a "co-variant region," in the sense that it varies with the map w.[5] The pair of co-variant detector and invariant descriptor, also referred to as detector/descriptor, is constitutes an invariant feature. In order for the latter to be of any use, detection has to be performed based on the images alone, with no information on the scene. In this section we illustrate two possible ways to proceed.

We allow the detector to selects multiple regions from each image because (1) the image might contain multiple useful regions (what is "useful" will become clear later) and (2) there is no way to distinguish between them *a priori*. At the same time, however, we should limit the detector to select a *finite* number of regions as it does not make computational sense to extract an infinite number of local features. Unfortunately, as illustrated by Fig. 2.5, a detector cannot

[5]In fact, it transforms with the inverse w^{-1}, so a more appropriate nomenclature would be "contra-variant."

be at the same time scale invariant and finite if the resolution of the image is infinite. In the practical case this problem never arises as only numerical images are available and details below the size of the pixel cannot be resolved. Thus, in order to keep the formulation simple and clean, we relax the notion of finiteness to the one of countability.[6] This does not trivialize the concept of detector as the number of possible regions is not countable.

Definition 5 (Detector)**.** A *detector* is a function d that maps an image $I : \Lambda \to \mathbb{R}$ to a countable collection of image regions: $d(I) = \{\Omega^{(1)}, ..., \Omega^{(k)}, ...\}$ with $\Omega^{(k)} \subset \Lambda$.

For our purposes, the detector must satisfy two additional properties, known as "co-variance" and "repeatability". Simply stated, this means that, if the region Ω_1 is selected in the first image I_1 and if it deforms according to a warp w due to a viewpoint change, then the region $\Omega_2 = w\Omega_1$ is selected in the second image I_2. In order to guarantee the co-variant detection of a region, some assumptions (such as visibility) on the image formation process must be satisfied at least locally. Since there is no way for the detector to know which regions are "good," an obvious strategy is to select several and regions and hope that at least some of them will satisfy the conditions.

Proposition 2 (Domain selection, or "co-variant detection")**.** *There exists a detector $d : I \mapsto \{\Omega^{(1)}, ..., \Omega^{(k)}, ...\}$ that selects co-variant regions, in the following sense. Consider a scene (S, ρ), two viewpoints g_t, $t = 1, 2$ and the generated images $I_t = h(S, \rho, g_t)$. For all compact patches $Q \subset S$ that are visible away from occluding boundaries from both viewpoints g_1 and g_2, projecting onto the regions $\Omega_t = \pi(g_t Q)$, $t = 1, 2$ respectively, one has*

$$\Omega_1 \in d(I_1) \Leftrightarrow \Omega_2 \in d(I_2).$$

The following proof is constructive and inspired by the work of Matas and coworkers [MCU02].

Proof. A region Ω_t of the image I_t is an *extremal region* if it is a compact connected component of a level set $L_t(c) \stackrel{\Delta}{=} \{x \mid I_t(x) \geq c\}$ of the image for some constant $c \geq 0$ (we use the definition of [CCM99].) All extremal regions that

[6] Finiteness can be re-introduced in the infinite resolution setting with additional weak assumptions on the regularity of the albedo (e.g. $\rho \in BV(S)$). However Fig. 2.5 shows that full scale invariance cannot be achieved with a finite detector, and one must relax this notion as well.

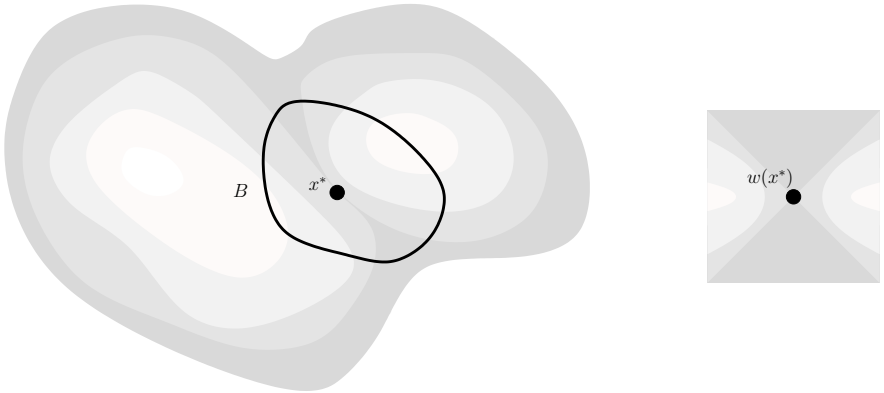


Figure 2.6: **Critical point.** A critical point x^* is a point where two level curves of $I(x)$ "meet". For a smooth function $I(x)$, this would be a saddle point; here we adapt this property to be invariant to homeomorphic deformation. We do this as follows. Consider the function $f(x_1, x_2) = x_1^2 - x_2^2$ defined on $\mathbb{R}^2$, portrayed on the right. We use $f(x)$ as a model of saddle point: We call x^* a *critical point* if there exists an open neighbor B and an homeomorphism $w : B \to \mathbb{R}^2$ such that $w(x^*) = 0$ and and each level set $L_{I|_B}(c)$ of $I(x)$, $x \in B$ is homeomorphically mapped to a level set $L_f(c')$ of f. This requires $I(x)$ to be locally equivalent to the "standard saddle" $f(x)$ up to a monotonic contrast change [CCM99] and an homeomorphic deformation of the domain. The property is obviously invariant per homeomorphism and local (i.e. if it is verified in the neighborhood B, it is verified in any neighborhood $B' \subset B$).

satisfy the local conditions of Lemma 3 are transformed by a homeomorphism as a result of a change of viewpoint.

Although (some of) the extremal regions satisfy the co-variance property, these are usually not countable. We need a mechanism to choose a small (countable) number of extremal regions in a way which is independent of domain transformations. We do this by considering the *critical points* x^* of $I_t(x)$, per Fig. 2.6; the detector d then selects those extremal regions Ω_t whose boundary $\partial\Omega_t$ contains at least a critical point x^* (Fig. 2.7). As the image domain is compact, at most a countable number of critical point can exist. Since the region is visible away from occluding boundaries, there is an entire neighborhood of the critical points which is homeomorphically mapped from I_1 to I_2. Furthermore, since homeomorphisms do not alter the topology of the neighborhood, critical points are mapped homeomorphically to critical points, hence they are repeatable across different viewpoints. □

Combining the previous results leads us to the following claim.

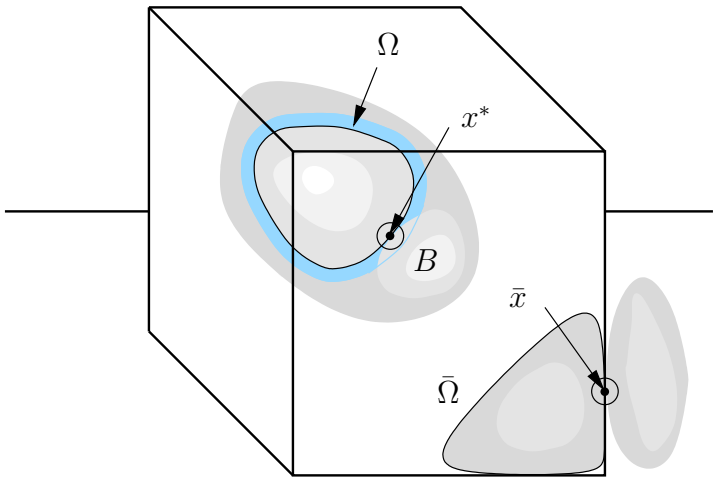


Figure 2.7: **Co-variant detector.** The co-variant detector selects the extremal regions Ω and $\bar{\Omega}$ because their borders $\partial\Omega$ and $\partial\bar{\Omega}$ contain the critical points x^* and $\bar{x}$. However, only Ω is visible and away from occluding boundaries, while $\bar{\Omega}$ is only visible. As such, only x^* is guaranteed to be repeatable (stable to a viewpoint change), while $\bar{x}$ can disappear as the viewpoint moves. Thus, although the detector selects both regions, only Ω is a "good region."

Theorem 1 (Existence of generic non-trivial viewpoint invariants)**.** *Consider a scene (S, ρ), viewpoints g_1, g_2 and the resulting images I_1, I_2. Then there exists a domain selection mechanism (detector) d and a local descriptor $\phi : I|_\Omega \mapsto \phi(I;\Omega) \in \mathcal{F}$ such that the collection of detector/descriptor pairs $F(I) \triangleq \{(\Omega, \phi(I;\Omega)),\ \Omega \in d(I)\}$ satisfy*

- ***(Invariance)** If $(\Omega_1, \phi_1) \in F(I_1)$ is such that Ω_1 is the projection of an unoccluded portion of the scene $Q \subset S$, that is also visible without occlusion from viewpoint g_2, then $(\Omega_2, \phi_2) \in F(I_2)$ and $\phi_1 = \phi_2$.*
- ***(Non-triviality)** In general, the descriptors $\phi(I;\Omega)$ varies as Ω varies in $d(I)$ (that is, they are not all identical).*

Proof. Invariance follows from Prop. 1 and Prop. 2. Non-triviality follows from the fact that the descriptors are albedo-discriminative (Prop. 1) and the counter-example in Figure 2.4. □

Remark 4 (Extension to ambient illumination invariants). A useful property of level sets and critical points is that they are invariant to affine scaling of the range of the image $I(x)$. This allow to relax the hypothesis of Prop. 2 to varying

ambient illumination and together with Remark 4, to extend Thm. 1 to viewpoint and (ambient) illumination invariants.

Remark 5 (Alternative co-variant selection). It should be noted that the usage of extremal regions identified by critical point is only one of many ways to detect co-variant regions under the assumption of visibility and Lambertian reflection. An interesting example is given in [LJ05], where co-variant regions are extracted by walking along "image geodetics." Unfortunately the method is not insensitive to changes in ambient illumination.

Another domain selection mechanism can be provided by a *segmentation procedure*: Assuming that the image is a piecewise smooth function, and that the locus of singularities is piecewise smooth [MS89], one can start from maxima and expand the regions until it reaches some singularity of the image. This occurs at edges and junctions, and can be performed following the guidelines of Lindeberg [Lin98].

2.3 Properties of viewpoint invariant features

In this section we study a few theoretical properties of viewpoint invariant features.

2.3.1 Shape discrimination

In order to construct a viewpoint invariant feature we need to eliminate the dependency of the image from viewpoint-induced transformations. Unfortunately, this process destroys the ability to discriminate scenes by their shape.

Definition 6. Consider a patch $Q_1 \subset S_1$ of a scene (S_1, ρ_1) and a patch $Q_2 \subset S_2$ of a scene (S_2, ρ_2). We say that the two patches have the *same appearance* if, and only if, there exist viewpoints g_1 and g_2 such that their projections coincide: $I_1(x) = \rho_1(X_1) = \rho_2(X_2) = I_2(x)$ for all $x = \pi(g_1 X_1) = \pi(g_2 X_2)$ with $X_1 \in Q_1$ and $X_2 \in Q_2$

Proposition 3 (Shape insensitivity)**.** *Let $\phi(I; \Omega) \in \mathcal{F}$ be a viewpoint invariant local descriptor as defined in Prop. 1 (but not necessarily constructed as in Prop. 1.) Consider two patches Q_1 and Q_2 of two scenes (S_1, ρ_1), (S_2, ρ_2) that have the same appearance in the sense of Def. 6, but different shape $S_1 \neq S_2$. Let g_1 and g_2 be viewpoints under which Q_1 and Q_2 projects onto $\Omega_1 = \pi(g_1 Q)$ and $\Omega_2 = \pi(g_2 Q)$ respectively. Then $\phi(I_1; \Omega_1) = \phi(I_2; \Omega_2)$ for all viewpoints g_1 and g_2 that do not occlude the patches.*

Proof. By the definition of viewpoint invariant local descriptor given in Prop. 1, the features $\phi(I_1; \Omega_1)$ and $\phi(I_2; \Omega_2)$ do not change as long as the change of viewpoint does not result in occlusions. Since the two descriptors coincide for at least the viewpoints for which the image of the two patches coincide (which exist by hypothesis), they are equal for any other pair of viewpoints that do not yield occlusions. □

The above proposition shows that viewpoint invariant statistics cannot be used to discriminate shape, no matter how such statistics are generated.

Remark 6 (Meaning of "shape"). To avoid confusion, note that here "shape" means the 3-D geometry of the scene S. If we have, say, a planar contour, which we can view as a binary image ρ, we can build a viewpoint invariant descriptor (e.g. [BMP02]) that can be legitimately used to recognize shape without searching for viewpoint during the matching procedure. Note, however, that the descriptor is *albedo-discriminative,* and it is only accidental that the albedo is used to represent (2-D) shape. Similarly note that the scene here includes everything visible, so the theorem does not apply to cases where the occluding boundary provides discriminative features, say to recognize a white sphere from a white cube on a black background. Finally, note that the notion of viewpoint can be generalized to an equivalence class under the action of a group, for instance the 3-D projective group, so that no *explicit* reconstruction is necessary during the matching phase.

Prop. 3 does not mean that one cannot discriminate scenes based on their shape, as we show next. Intuitively, given two images there are two possible scenarios: (a) the two images are generated by the same scene under different viewpoints, or (b) the two images portray different scenes. Local invariant features cannot discriminate between these two scenarios. However, an attempt to *reconstruct* the scene can discriminate, since only in the case (a) are the two images compatible with only one underlying scene. Note that we are barring visibility artifacts, lest given two images one can always construct a scene that generates them (e.g. glue the two images onto two sides of a cube). Further note that we are making the assumption of *generic* scene. One can easily construct scenes that, when viewed from a particular set of viewpoints, generate images that are compatible with a single underlying scene. These conditions, however, are *pathological* in the sense that a change in viewpoint destroys the compatibility and reveal that the images come from different scenes.

Proposition 4 (Recognition via reconstruction)**.** *Let the two patches Ω_{S_1} of scene (S_1, ρ_1) and Ω_{S_2} of scene (S_2, ρ_2) have the same generic radiance and project without occlusions from the generic viewpoints g_1 and g_2. If their images coincide*

under viewpoints (g_1, g_2)*, then either* $\Omega_{S_1} = \Omega_{S_2}$ *or, for almost all viewpoints* g_1'*, there exists no viewpoint* g_2' *such that the images are still the same.*

Proof. Since the radiance is generic, and so are the viewpoints, we can assume that $g_{12} \stackrel{\Delta}{=} g_1^{-1} g_2$ has non-trivial baseline (non-zero translation). If the two images come from the same scene, we can find *one* surface S and a viewpoint change g_{12} such that $I_1(\pi(g_{12}S(x))) = I_2(x)$. This is tantamount to reconstructing the shape of the scene and the viewpoint, which can be done uniquely up to a global scale factor under the stated genericity assumptions [JSY03]. Under the same genericity assumptions, two scenes generate images that are not compatible with a unique reconstruction, since the viewpoint is generic [MSK03]. □

2.3.2 Maximally discriminative feature

The basic idea of Prop. 1 is to annihilate the effect of a viewpoint change by absorbing in the descriptor all possible deformations of a given patch. There we considered as deformations the family of all two-dimensional homeomorphisms because any viewpoint induced deformation (of visible patches) falls within this class. It is intuitive, however, that not all possible homeomorphisms can be generated by a viewpoint change. Since the bigger is the class of transformation that we annihilate, the lesser discriminative is the resulting descriptor, we should understand which is the minimal class of transformations that we have to factor out to achieve viewpoint invariance.

Definition 7. We denote by $\mathcal{V}$ the set of viewpoint induced warps $v : \Omega_1 \to \Omega_2$, i.e. warp v that can be induced on a visible patch by a change of viewpoint that leaves the patch visible.

It is not difficult to characterize $\mathcal{V}$. For example, [SD96] shows that, as long as no camera sees the center of the other one, the deformation resulting from a viewpoint change is the composition of two homographies and a parallax.[7] Even for more general conditions, the warp is still *not* generic (for example, epipolar lines are transformed to other *lines*).

Unfortunately, factoring out $\mathcal{V}$ is not enough in order to get an invariant descriptor; as Fig. 2.8 illustrates, we need to consider the bigger class $\bar{\mathcal{V}}$ obtained by closing by composition the class $\mathcal{V}$. Annihilating $\bar{\mathcal{V}}$ is not only sufficient, but also necessary. Thus, by considering $\bar{\mathcal{V}}$ in place of the generic homeomorphisms in Prop. 1, the construction leads to the most discriminative viewpoint invariant de-

[7] If this condition is not satisfied, a zooming is also required.

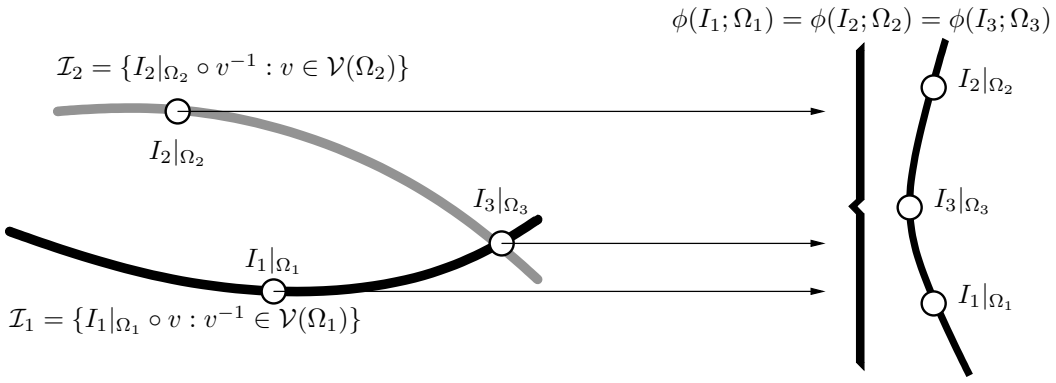


Figure 2.8: **Warps collapse due to invariance.** In order to build a viewpoint invariant descriptor we need to absorb not just the viewpoint transformations $v \in \mathcal{V}$, but their compositions too. In fact, consider a viewpoint invariant descriptor $\phi(I;\Omega)$ and two patches $I_1|_{\Omega_1}$ and $I_2|_{\Omega_2}$. The figure shows the families of patches $\mathcal{I}_t$, $t = 1, 2$ that are generated from $I_t|\Omega_t$ by applying all possible viewpoint warps $\mathcal{V}$ (the symbol $\mathcal{V}(\Omega_t)$ denotes the warps supported on Ω_t). By definition ϕ is constant over $\mathcal{I}_1$ and $\mathcal{I}_2$. Thus, if $\mathcal{I}_1$ and $\mathcal{I}_2$ intersect, say at $I_3|_{\Omega_3} = I_1|_{\Omega_1} \circ v_1^{-1} = I_2|_{\Omega_2} \circ v_2^{-1}$, then ϕ is constant on the *union* $\mathcal{I}_1 \cup \mathcal{I}_2$. But, in order to get $I_2|_{\Omega_2}$ from $I_1|_{\Omega_1}$ we need to use a warp $w = v_1^{-1} \circ v_2$ which is *not* in $\mathcal{V}$.

scriptor.[8] How much smaller is $\bar{\mathcal{V}}$ with respect to the class of all homeomorphisms is subject of current study.

Remark 7. We said that the descriptor of Prop. 1 is albedo-discriminative, while in this section we are guessing that descriptors even more discriminative are possible. This *not* a contradiction: in fact, the descriptor of Prop. 1 is discriminative as long as we consider albedos to be equivalent up to generic homeomorphism. Here is exactly the latter assumption that we want to relax.

2.4 A case study: 3-D corners

As an application of our theory, we develop a descriptor of a new kind of invariant features, corresponding to "3-D corners". We design a simple detector that selects

[8]The notion of maximal invariance is precise: it means that, if $\tilde{\phi}$ is another viewpoint invariant descriptor, then

$$\tilde{\phi}(I_1;\Omega_1) \neq \tilde{\phi}(I_2;\Omega_2) \Rightarrow \phi(I_1;\Omega_1) \neq \phi(I_2;\Omega_2).$$

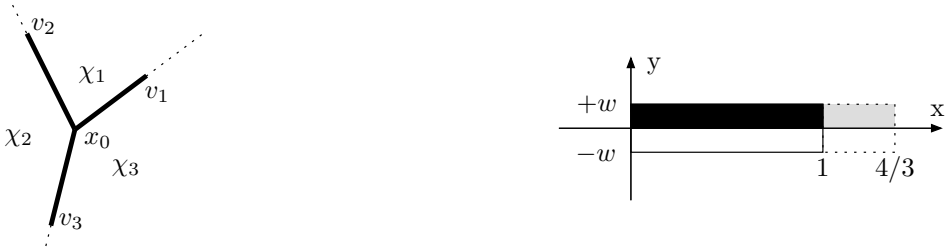


Figure 2.9: **Corner: deformation, detection, and canonization.** *Left*: a corner consists of n angular sectors $\chi_1, ..., \chi_n$ separated by edges $v_1, ..., v_n$. The length of the vectors $v_1, ..., v_n$ fix a point along each edge, which can be thought as a scale parameter. *Right:* the edge of the corner are detected by means of a simple parametric model by connecting pairs of Harris' points.

points where a suitable frame of reference can be easily attached. In particular, we focus on points $x_0 \in \Lambda$ that are projections of corners of the surface $S(x)$. A corner is a singular point of the object surface and cannot be approximated by a plane. Thus our descriptor works exactly under the conditions not supported by existing approaches [MS04, FTV03] (Figure 2.11).

2.4.1 Deformation under viewpoint change

We model a corner as a vertex with n planar faces. Barring occlusions, its image (see Figure 2.9) consists of n angular sectors, projections of the n faces, and a center x_0, projection of the vertex. These sectors are separated by edges, which we represent as vectors $v_i \in \mathbb{R}^2,\ i = 1, ..., n$. The length of the vectors will be used as a scale parameter.

When the viewpoint changes, the n faces of the corner are transformed by homographies, which we approximate by affine warps. This model locally captures the true transformation to an arbitrary degree of precision, which cannot be done by a single affine transformation. Since the corner surface is continuous, in the absence of occlusions so is the overall transformation. Thus, the n affine transformations are not independent and are fully specified by the mapping $x_0 \mapsto y_0$ of the center and the mappings $v_i \mapsto u_i,\ i = 1, ..., n$ of the edges (with their scales). Formally, let $\{\chi_i(x),\ i = 1, ..., n\}$ be a partition of $\mathbb{R}^2$ in n angular sectors, being $\chi_i(x)$ the indicator function of the i-Th. sector. We call *piecewise affine transformation* of degree n a function $w : \mathbb{R}^2 \to \mathbb{R}^2$ given by $w(x) = \sum_{i=1}^{n} \chi_i(x) A_i(x - x_0) + y_0,\ x \in \mathbb{R}^2$ where the matrices $A_i \in GL(2),\ i = 1, ..., n$ are chosen so that $w(x)$ is continuous.

A convenient parameterization of $w(x)$ is the tuple $(x_0, y_0, v_1, ..., v_n, u_1, ..., u_n)$, specifying the mapping of the center and of the edges. If we assume that the edges and the corresponding sectors are sorted counter-clockwise, the matrices $A_i,\ i = 1, ..., n$ are given by the equations $A_i v_i = u_i$ and $A_i v_{(i)_n+1} = u_{(i)_n+1}$, being $(i)_n$ the integer i modulo n. Let $\mathrm{PWA}_n,\ n \geq 2$ be the set of all piecewise affine transformations of degree n. For any fixed $n \geq 3$, the class does *not* form a group, since it is not closed under composition, but each transformation has an inverse $w^{-1}(x)$ of the same degree. The group-closure of $\mathrm{PWA}_n,\ n \geq 3$ is $\mathrm{PWA} = \bigcup_{n \geq 2} \mathrm{PWA}_n$.

Since the deformation of a corner under a viewpoint change is (locally) a PWA, PWAs are the minimal class of transformations with respect to which the feature has to be invariant, even though any more general class of transformations would fit. In particular, a PWA of degree m can be used for a corner that has $n < m$ physical edges, as long as the transformation is estimated consistently.

2.4.2 Feature detection

The detection process searches for corner structures in the image and attaches a reference frame to them. While there exist many possible procedures for detecting corners, including sketch primitives [GZW03] or matched filters [HF94], our emphasis here is not in proposing yet another detector, but rather in how to arrive at a viewpoint invariant once a structure has been detected. Therefore, we choose a simple if not somewhat naive detector, designed to provide directly the structures that we need.

The procedure is articulated as follows. Initially, a set of Harris points [HS88] $X = \{x_1, ..., x_n\}$ is extracted. These points are used as candidate corners and as evidence for edge-like structures in the image (we use the fact that some Harris points are located along edges, particularly nearby the edge terminations). The algorithm checks for each pair $(x_i, x_j) \in X^2$ whether the image portrays an edge connecting x_i to x_j. Edges are modeled using the parametric template

$$T(x, y; w) = \mathrm{sign}(y), \quad (x, y) \in [0, 1] \times [-w, w] \tag{2.4}$$

reminds what done in [BNM98] (Figure 2.9). The template is matched[9] to the image by normalized cross correlation (NCC). Once the set E of edges has been extracted, the procedure attaches a reference frame to each point $x_0 \in X$. All edges connected to x_0 are considered: first the localization of each edge is refined using the model (2.4); then edges with the same orientation are clustered, because they relate to the same image structure; finally the edge that best covers the full

[9]There exists some simple yet effective heuristics that one can use to pre-prune the set of candidate edges and speed-up significantly the algorithm.

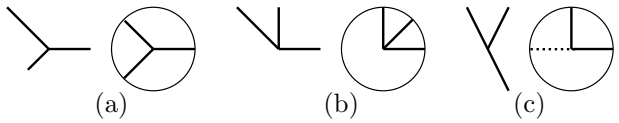


Figure 2.10: **Types of reference frame and their canonical configuration:** (a) all sectors are narrower than π radians; (b) a sector is wider than π radians and (c) T-junction.

extension of the underlying image structure is selected within each cluster. The selection uses an extension of the model (2.4) which represents explicitly the edge termination.

2.4.3 Feature canonization

Once a reference frame has been detected, we map it to a canonical configuration. In order to avoid singular configurations, we enforce the following conditions: (i) if all sectors are less than π radians wide, the normalized frame has n equally wide sectors; (ii) if one of the sectors is wider than π radians, we make this sector $3\pi/4$ radians wide and we fit evenly the others in the remaining $\pi/2$ radians[10]; (iii) if one sector is exactly π radians wide (T-junction), we delete one edge and we reduce to the former case[11]. Note that at least two edges are required to compute the PWA transformation. If a point has less than two edges attached to it, it is discarded.

These rules fix the canonical reference frame up to a rotation. The rotation can be partially eliminated by requiring that one edge maps to $(1, 0)$. However, any edge could do, and we are left with a discrete subgroup of rotations to choose from. If the corner has a sector wider than π radians, we use this to uniquely identify an edge and eliminate the ambiguity. This is possible because there is at most one such sector and the property is preserved under viewpoint changes. If all sectors are narrower than π radians, we use the sector with maximal mean albedo as reference.

[10]We do this because no PWA (nor viewpoint) transformation can make the wide sector smaller than π radians

[11]We do this because no PWA (nor viewpoint) transformation can change the π radians wide angle.

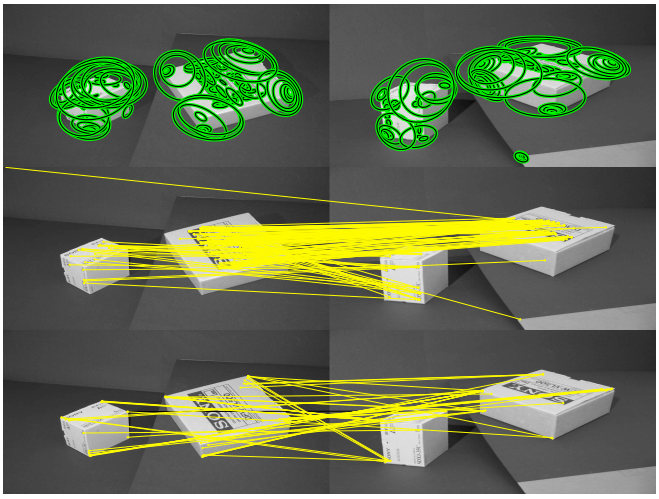


Figure 2.11: **Affine-invariant descriptors fail to capture non-planar structures:** (top) two images of the same scene with detected regions; (middle and bottom) correspondence established using affine invariant signatures respectively for planar (middle) and non planar (bottom) regions. Several non planar regions are detected by the low-level detector, but are not matched because of the large discrepancy in the corresponding descriptor, caused by the non-planar structure of the scene.

2.4.4 Feature description

Although the canonized features could be compared directly (e.g. by NCC), we compute a descriptor for each detected feature. This has two advantages: (1) makes the comparison much faster and (2) may absorb differences in the normalized features due to imprecise detections or unsatisfied assumptions (e.g. the surface is not Lambertian). Furthermore, most descriptors are insensitive to affine transformations of the albedo, so that we do not need to normalize explicitly the illumination. In the experiments we use the SIFT descriptor [Low04], one of the most widely used [Low04, MTS04]. We note however how this descriptor may not be as effective in our case as is for other kind of features. Indeed our canonized corners have strong oriented structures (the edges) in fixed position. This makes the SIFT descriptor (which is based on the gradient distribution) less discriminative.

Unilateral feature descriptors. The detector/descriptor works well under the assumptions we made. However, we wish to relax the hypotheses that the whole corner image is the projection of a single object. In fact, many corners are found

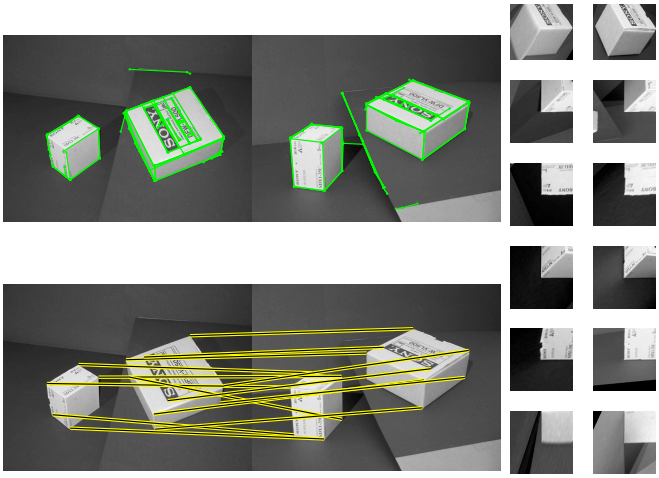


Figure 2.12: **General viewpoint invariants can match 3-D corners:** (top) detected reference frames; (bottom) matched "3-D features"; (right) examples of canonized features. Most of the "3-D features" that are detected but mismatched using an affine-invariant descriptor are correctly matched using a more general viewpoint-invariant model, in this case a "3-D corner."

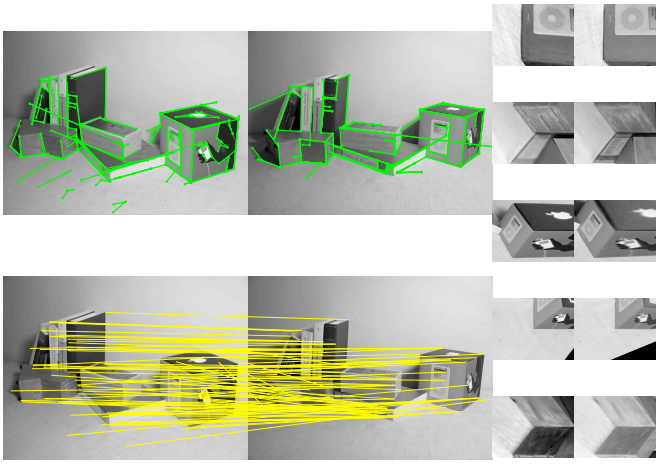


Figure 2.13: **Matching example:** (top) all the features detected in the first image are connected to their nearest neighbors in the second image; (bottom) all the features detected on the first image are connected to their nearest neighbors in the second image and (right) a variety of normalized features. Of 93 detected features, 32 are present and correctly matched in the second image.

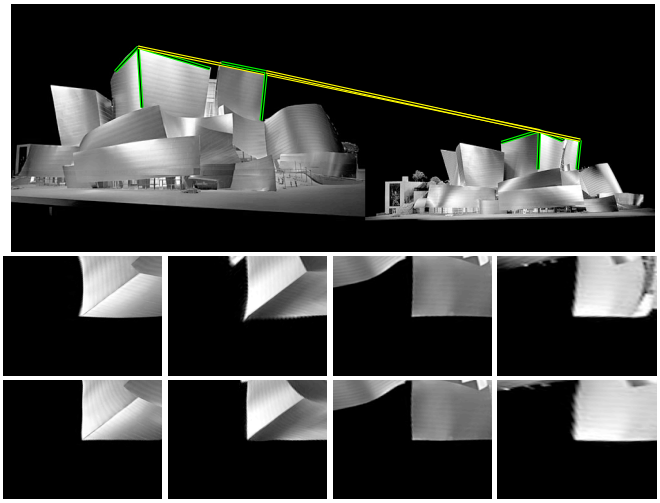


Figure 2.14: **Matching a challenging scene:** (top) two corners matched by our method; (middle) features canonized by a piecewise-affine transformation; (bottom) features canonized by a thin-plate spline transformation. Concert Hall, like all recent Gehry building, is challenging because severely non-Lambertian, and non-planar. Although the scene does not meet most of our working assumptions, a few corners are still matched (see also supplementary material).

on the boundaries of objects [SH05], and some sectors χ_i of the corner image (Figure 2.9) may belong to the background. Clearly, we are not supposed to incorporate the background into the feature if we want to preserve invariance. We solve the problem by computing multiple descriptors for each possible assignment of the faces to the foreground or the background. In practice, the most common cases (objects with convex corners) are covered if we do so only for sectors larger than π radians, thereby obtaining no more than two descriptors for each detected feature.

2.4.5 Experiments

In the first experiment we explore the domain of applicability of our technique, by showing that it operates when established detector/descriptor techniques fail. To illustrate our point, we have purposefully chosen a simple scene (Figure 2.11): even on such scenes, most of the current affine-invariant methods fail to establish correspondence. Also, note that our goal is not to compare our method with existing affine-invariant schemes, since our method works on top of them. Since, to the best of our knowledge, nobody has presented viewpoint invariant schemes

for non-planar scenes, we cannot do direct comparisons with any existing scheme.

As a typical representative [MTS04] of the class of affine invariant detectors, we selected the Harris-Affine detector [MS04]. Figure 2.11 shows that most of the non-planar detected regions are incorrectly matched using the affine descriptor: of 186 features detected in the first image, 53 are successfully matched, 68 are mismatched because the descriptor variability and 65 are not matched because the low-level detector fails to select the corresponding region. In contrast, Figure 2.12 shows the performance of our method on the corners of the same images: almost all 3-D corners are matched correctly. There is just one mismatch, due to the almost identical appearance of the exchanged features (last two feature pair in Figure 2.12), and two missing corners, which are not extracted by the Harris detector in the very first stage of the algorithm. An exact comparison with the affine-invariant detector is difficult because the latter finds several times the same structures; roughly speaking, however, 70% of the mismatches (due to missing features or discrepancy of the descriptor) of the affine detector are fixed by the "3-D corner" model. As an additional advantage, our method extracts just one feature for each 3-D structure, while the Harris-Affine detector generates many duplicate detections of these structures.

In the second experiment we test our method on a more complex scene, made of various objects presenting a variety of 3-D corners. Figure 2.13 shows the detected reference frames and the matching pairs. One third of the detected features in the first image are correctly matched to the corresponding features in the second. Therefore, the performance is similar to that of the Harris-Affine detector on the planar structures of Figure 2.11, but in our case for non-planar structures. Some feature pairs are shown as well: they illustrate the most typical canonical configurations.

In the last experiment (Figure 2.14) we test our method on a more challenging scene, where several of our working hypotheses are not verified. We match two images of an highly non-planar, non-Lambertian scene. Not only the scene contains many 3-D corners, but these have non planar faces as well. Moreover, the two images are at two significantly different scales. The figure shows two corners that our method is able to match nevertheless, together with the corresponding canonized features. Note that the features are quite different, because of both the reflections and the non planarity of the corner faces. Still, these canonized features are similar enough to be matched using the SIFT descriptor, illustrating the importance of viewpoint canonization. As a further example of this fact and of the generality of our framework, we show the same two corners normalized using a thin-plate spline deformation, estimated by tracking and rectifying the edges. The matching distances are slightly smaller ($0.28 \mapsto 0.15$ and $0.4 \mapsto 0.36$ respectively) using this deformation as we compensate for the curvature of the

edges.

Discussion

The material of this chapter is mainly theoretical: We clarify some misunderstandings that are lingering in the literature, where affine-invariant detectors/descriptors are often motivated by the non-existence of general-case viewpoint invariants following [BWR92]. Our results do not imply that affine-invariant descriptors are not useful. On the contrary, they may very well be the way to go, but we believe it is important that their motivations be clear and that overly restrictive assumptions are not imposed. Furthermore, by showing that viewpoint invariants are not shape-discriminative we validate "bags of features" approaches to recognition (see [DS04] and references therein), where spatial relations among features (i.e. shape) are either discarded or severely quantized or "blurred" [BMP02, BM01b]. Finally, we show that if instead of using a feature-based approach one factors out viewpoint as part of the matching process, then shape is discriminative indeed. This, however, requires (explicit or implicit) optimization with respect to the viewpoint, which may help explain some of the psycho-physical results following [SM71], where albedo is non-discriminative and therefore shape is the only "feature."

Formalizing the simplest instance of the recognition problem makes it immediate to see that features cannot improve the quality of "recognition by reconstruction," if that was theoretically and computationally viable. However, features can provide a principled, albeit suboptimal, representation for recognition: We have shown that under certain conditions viewpoint and illumination-invariant features can be constructed explicitly. Our framework allows comparison of existing methods and opens the way to design richer classes of detectors/descriptors. As an illustrative example, we introduce a 3-D corner descriptor that can be employed to establish correspondence when the state of the art fails because of violation of the local-planarity assumption.

CHAPTER 3

Optimal Support for Local Features

In Chapter 2 we noticed that viewpoint invariant features are limited, for all practical purposes, to the description of local image regions. The reason is that, in order to achieve invariance, one is forced to make assumptions on the image formation that are rarely satisfied globally. Among these, the most important is that the feature support must not be occluded during the viewpoint change. While very small features have the highest chance of avoiding occlusions, these are also the least informative, as they depend only on a small portion of the available data. Ideally, one would like the support of local features to be as large as possible, as long as no occlusion is intersected. Unfortunately, occlusions depend on the interaction of two or more views, and cannot be estimated from a single image. Therefore the optimal feature support cannot be determined when the feature is extracted (that is, during detection).

This chapter stands on a simple yet useful observation: While the feature support cannot be optimally chosen during detection, we can still optimize it during the matching process. In other words, we can start with small local features (say affine invariant) selected and normalized independently in training and test images, and expand their domain as part of the correspondence process. Correspondence amounts to a (non-rigid) registration task, and the dilation process yields a multi-view segmentation of the object of interest from clutter, including the detection of occlusions.

This process can be interpreted as the simultaneous registration and segmentation of deformable objects in multiple views starting from an "affine seed." It can also be thought of as an implicit 3-D reconstruction of the scene, which enables the recognition of non-planar objects and the discrimination of objects based on their shape (see Chapter 2).

We formalize the feature growth process as an optimization problem, and introduce efficient algorithms to solve it under three different deformation models. Our growth process can be thought of as a region-based segmentation scheme, but indeed it is quite different since it is *unilateral*, i.e. it requires a characterization of the foreground, but not of the background statistics.

Among the applications of our technique are general object recognition tasks, both *supervised*, i.e. given an uncluttered, unoccluded image ("template") of a

(3-D, possibly deformable) object of interest, find it in a cluttered image, and *unsupervised*, i.e. given two or more images all containing the same object under different clutter and occlusions, detect and localize the common object. Note that we concentrate on the recognition of specific objects, rather than object categories, as we do not allow intrinsic variability of the object other than the geometric deformations captured by the model. Our goal is to improve the discriminative power of local representations, so that finer discrimination can be performed: We do not just want to tell a face from a bottle; we want to tell *one* particular bottle from another, say with a scratch on it.

Our technique builds on local affine invariant features [MTS04, KB03]. By dilation and alignment, it increases their discriminative power as part of the matching process rather than directly as part of the representation, and extends their validity to non-planar, non-rigid objects. In principle, nothing prevents us from gathering such enlarged regions into constellations or bags [FH05, FHI00, FFP04, PLR04] although we will confine ourselves to studying *one* feature in isolation to better test the improvement relative to affine descriptors.

Since we combine region growing with registration, our work is also related to [Lhu98, WQ04], although these authors address the problem of global correspondence in a short baseline setting. [FTV05] propagates affine matches from an (unoccluded) image of an object (template) and a small set of initial seeds. None of these approaches model both viewpoint-induced deformations and the shape of the extracted regions explicitly.

As the selection of the interest region is not determined by the local image statistics alone, but is determined through the matching process, one can think of our technique as a *motion segmentation* procedure [ATW05, CS04, WAB03]. Finally, since the growth process takes part during the alignment (correspondence) from multiple views, our work relates to [CMT04, HB98, Bla05] and other tracking and long-baseline correspondence techniques, although it differs from them computationally.

In order to keep the implementation efficient, we work in a discrete rather than variational setting, much in the spirit of [BM02, SK05]. To this end, we introduce models of regularized region growth that are flexible and result in very efficient algorithms [Tsi95]. As opposed to [ZY96] and similarly to [PTK04], the segmentation is local and uses statistics only inside and in the immediate neighborhood of the region.

3.1 A model of feature growth

Recall that in Chapter 2 we defined images as a functions $I_t : \Lambda \to \mathbb{R}_+$, $t = 1, 2, ...$ Differently from Chapter 2, in this chapter we assume that $\Lambda = \{0, ..., M\} \times \{0, ..., N\}$ is a discrete set (lattice) and we extend $I_t(x)$ to non integer arguments by bilinear interpolation.

In addition to the domain $\Omega \subset \Lambda$ of a feature, we also consider a (binary or smooth) *window* function $H(x) : \Lambda \to [0, 1]$. Although related to the window $H(x)$, Ω is *not* necessarily its support $\operatorname{supp} H \triangleq \{x : H(x) \neq 0\}$; its precise meaning will be specified in Section 3.1.1. A *feature match* (H_i, w_{ij}) is a window $H_i(x)$ describing the interest region on one image $I_i(x)$, together with a regular warping (diffeomorphism) $w_{ij} : \mathbb{R}^2 \to \mathbb{R}^2$ onto the corresponding region in another image $I_j(x)$.

Our method starts with a putative feature match $(H, w) \triangleq (H_1, w_{12})$ initialized from an "affine seed" as in [MTS04, KB03]. An *affine seed* is a pair of corresponding elliptical regions (Ω_1, Ω_2) in $I_1(x)$ and $I_2(x)$ respectively. The regions are related by an affine warp $w(x) = Ax + T$, $(A, T) \in \mathbb{GL}(2) \times \mathbb{R}^2$ which is fixed (by $w\Omega_1 = \Omega_2$) up to a rotation $R(\theta) \in \mathbb{SO}(2)$. We estimate the residual rotation by maximizing the normalized cross-correlation (NCC) of the appearance of the two regions. The region Ω_1 and the transformation (A, T) are then used to initialize the feature match (H, w).

We grow the initial match by trading off dilation of the window $H(x)$ and quality of the alignment w, expressed by the following cost functional:

$$\begin{aligned} E(H, w, \mu) = &\sum_{x \in \Lambda} H(x)[(\mu(I_1) - I_2 \circ w)(x)]^2 \\ &- \alpha \left(\sum_{x \in \Lambda} H(x) - \beta \mathcal{R}(H) \right) + \gamma \mathcal{Q}(w). \end{aligned} \tag{3.1}$$

The first term is a sum of squared difference (SSD) representing the quality of the local alignment. Minimizing this term has two effects: to select the warp w that best aligns the interest regions and, of course, to shrink the region (the global minimizer of this term is $H(x) = 0 \ \forall \ x$). Minimizing the second term favors instead larger regions. There is a simple interpretation of the control parameter $\alpha \in \mathbb{R}_+$: $\sum H(x)$ can be thought as the area of the region[1] and α as the mean squared residual that we are willing to absorb within the region. The terms $\mathcal{R}(H)$ (Section 3.1.1) and $\mathcal{Q}(w)$ (Section 3.1.2) are regularization terms for the region H and warp w respectively. The function $\mu : \mathbb{R}_+^\Lambda \to \mathbb{R}_+^\Lambda$ is a pre-processing

[1] This is exactly the case for binary regions when $\beta = 0$.

operator that can be used to compensate for other factors affecting the range of the image, such as illumination (Section 3.1.3). The goal is to find H, w and μ by alternating minimization of E, which we discuss in the following sections.

3.1.1 Region model

The region growth is determined by a controlled evolution of the window function $H(x)$. There are several possible choices for the model of the window ranging from simple parametric models that allow only a limited set of shapes (ellipses, rectangles, etc.) to non-parametric models that enable regions with arbitrary shape (up to topological and smoothness constraints). In order to explore this spectrum of options, we experimented with three models, explained next. Since in this section we focus on the region only, we rewrite the cost (3.1) as

$$E(H) = \sum_{x \in \Lambda} H(x)(D(x)^2 - \alpha) - \alpha\beta\mathcal{R}(H) + \text{const.} \tag{3.2}$$

where we have defined the residual $D(x) \stackrel{\Delta}{=} \mu(I_1)(x) - (I_2 \circ w)(x)$.

Elliptical region. The first model, fully parametric, is a smoothed elliptical window

$$H(x; p) \stackrel{\Delta}{=} \phi(y^\top y), \quad y = A(p)^{-1}(x - T(p)), \quad x \in \mathbb{R}^2$$

where $\phi \in C^\infty(\mathbb{R}_+ \to [0, 1])$ is a non-increasing function such that $\phi(0) = 1$ and $\phi(+\infty) = 0$ and $(A(p), T(p)) \in \mathbb{GL}(2) \times \mathbb{R}^2$ is the affine map that brings the unit circle onto the elliptical region. The window is parametrized by the vector $p \in \mathbb{R}^6$ as $\operatorname{vec} A = (p_1, p_2, p_3, p_4)^\top$ and $T = (p_5, p_6)^\top$, where vec denotes the stacking operator. This model does not make explicit use of the feature support Ω; it is however handy to define it as the nominal support of the window $\Omega = \{x : H(x) > \tau\}$, for some small value of τ (e.g. $\tau = 1\%$).

Since this model is fully constrained, the regularization term $\mathcal{R}(H)$ in eq. (3.2) is unnecessary. The resulting minimization problem can be solved by Gauss-Newton (GN) or any other descent technique. In the experiments we combined steepest descent (SD) with GN for reliable and fast convergence.

Binary free-form region. A binary free-form region is the characteristic function $H(x) = \chi_\Omega(x)$ of a domain $\Omega \subset \Lambda$ that has 4-neighbors connectivity. The regularization term $\mathcal{R}(H)$ is the length of the 8-ways discrete perimeter $\pi_8(\Omega)$ of the set Ω. The representation allows for changes in topology, even if these are discouraged by the regularization (too many "holes" will increase the length of

Algorithm 1 Growing binary free-form regions

1: Pre-compute dilation cost table
   $\mathrm{Lkp} : \{0,1\}^8 \to \mathbb{R}$.
2: Make heap of
   $\{(D^2(x_+) - \alpha(1 - \beta\,\mathrm{Lkp}(\mathcal{N}_\Omega(x_+))), x_+), x_+ \in \partial_+\Omega\}$
3: **loop**
4: Pop minimal element (c_+, x_+) from the heap. Stop if $c_+ > 0$.
5: $\Omega \leftarrow \Omega \cup \{x_+\}$ and $H \leftarrow \chi_{\Omega_+}$.
6: Add missing 4-neighbors of x_+ to the heap.
7: Update the cost of the 4-neighbors of x_+ in the heap.
8: **end loop**

the perimeter). The cost functional (3.2) assumes the form

$$E(H) = \sum_{x\in\Lambda} H(x)(D(x)^2 - \alpha) + \alpha\beta\pi_8(\Omega) + \text{const.}$$

To maximize of $E(H)$ we add to Ω the pixel x_+ belonging to the outer border $\partial_+\Omega$ (dilation) or we remove the pixel x_- belonging to the inner border $\partial_-\Omega$ (contraction) that most decreases the cost function (SD). Here we discuss only dilation moves, as contraction moves are similar. The updated window $H_+ = \chi_{\Omega\cup\{x_+\}}$ has cost

$$E(H_+) \quad = \quad E(H) \;+\; (D(x_+)^2 \;-\; \alpha) \;+\; \alpha\beta\,(\pi_8(\Omega \cup \{x_+\}) - \pi_8(\Omega))\,. \quad (3.3)$$

The term $\pi_8(\Omega\cup\{x_+\})-\pi_8(\Omega)$ is very efficient to compute. In fact, it depends only on the tuple $\mathcal{N}_\Omega(x_+) \stackrel{\Delta}{=} (\chi_\Omega(x) : x \text{ is 8-neighbor of } x_+)$ and can be pre-computed and stored in a lookup table of just 256 entries, leading to Algorithm 1. This algorithm is similar to [Tsi95] and reminiscent of the discrete level-sets of [SK05].

This model has two drawbacks: the window $H(x)$ is not smooth and the amount of regularization that can be imposed on the shape of the region is limited by the discrete nature of the steps that are used in the descent (too much regularization can block growth). To overcome these restrictions we turn to the smooth free-form region model described next.

Smooth free-form region. A smooth free-form region is obtained by smoothing a binary free-form region. The window $H(x)$ is given by the convolution $(g_\sigma * \chi_\Omega)(x)$, $x \in \Lambda$ where g_σ is a Gaussian kernel of standard deviation σ. This yields a smooth window *and* a very efficient regularization criterion, as explained next.

Algorithm 2 Growing smooth free-form regions

1: $D_\sigma^2 \leftarrow g_\sigma * D^2$
2: $H \leftarrow g_\sigma * \chi_\Omega$
3: Make heap of
$\{(D_\sigma^2(x_+) - \alpha(2H(x_+) + g_0), x_+), x_+ \in \partial_+\Omega\}$
4: **loop**
5: Pop minimal element (c_+, x_+) from the heap. Stop if $c_+ > 0$.
6: $\Omega \leftarrow \Omega \cup \{x_+\}$ and $H \leftarrow H + g_\sigma * \delta_{x_+}$.
7: Add missing 4-neighbors of x_+ to the heap
8: Update the cost of neighbors within the support of the kernel g_σ in the heap.
9: **end loop**

In eq. (3.2) we set $\beta = 1$ and $\mathcal{R}(H) = \sum_{\Lambda/\Omega} H(x)$ so that

$$E(H) = \sum_{x\in\Lambda} H(x)D(x)^2 - \alpha \sum_{x\in\Omega} H(x) + \text{const.}$$

Since $H = g_\sigma * \chi_\Omega$ is small where the boundary of the region has high curvature or where the region is thin, the regularization favors compact and smooth regions. Like for binary regions, minimization of $E(H)$ is fast. Again we discuss only dilation moves. Given the window $H = g_\sigma * \chi_\Omega$, $\Omega \subset \Lambda$, we need to find the pixel $x_+ \in \partial_+\Omega$ for which the new window $H_+ = g_\sigma * \chi_{\Omega\cup\{x_+\}}$ has the lowest possible cost $E(H_+)$. We have

$$\begin{aligned} E(H_+) = \sum_{x\in\Lambda} D(x)^2 \left(g_\sigma * (\chi_\Omega + \delta_{x_+})\right)(x) - \alpha \sum_{x\in\Omega} \left(g_\sigma * (\chi_\Omega + \delta_{x_+})\right)(x) \\ = E(H) + (g_\sigma * D^2)(x_+) - \alpha(2H(x_+) + g_0) \quad (3.4) \end{aligned}$$

where $\delta_{x_+} = \delta(x - x_+)$ is the Kronecker's delta and $g_0 \stackrel{\Delta}{=} g_\sigma(0)$. The map $g_\sigma * D^2$ can be conveniently pre-computed, leading to Algorithm 2. Figure 3.1 shows some examples of regions grown using Algorithm 2; Figure 3.2 compares the three models as they grow the same affine match.

3.1.2 Warping model

The deformation induced on the image domain by changes in viewpoint can be rather complex, depending on the shape of the scene [VS05]. In particular, occlusions cause such a transformation to be globally non-invertible, and there is no way to distinguish a-priori an occlusion (a portion of the scene disappearing

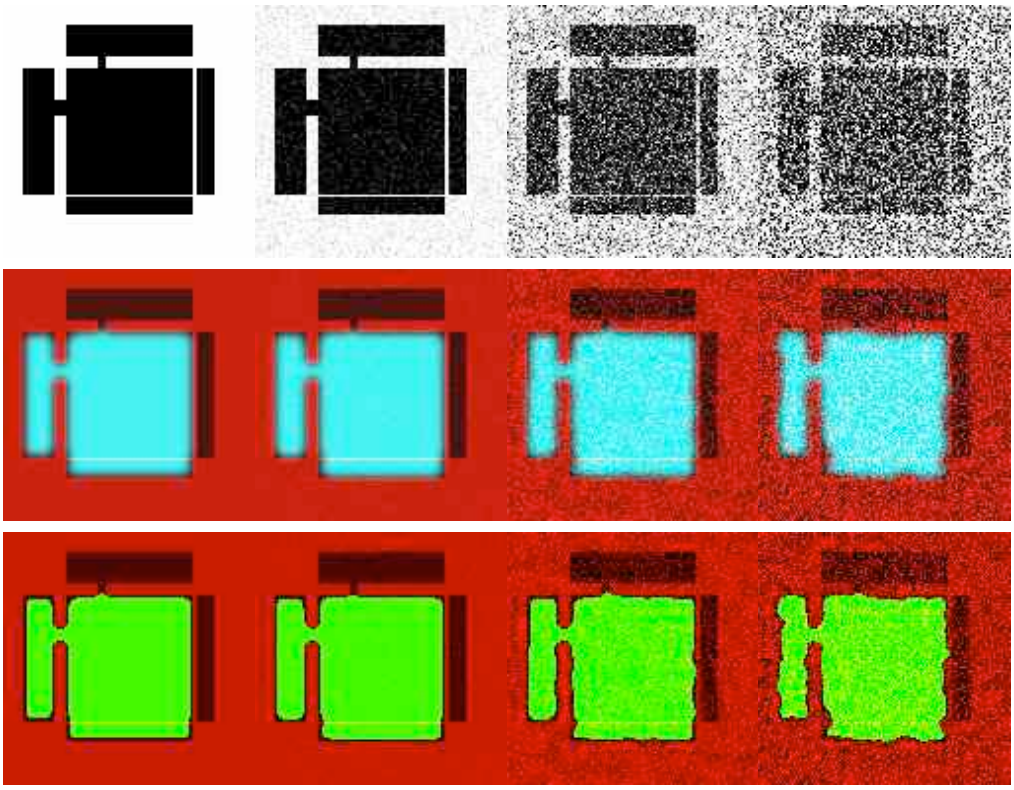
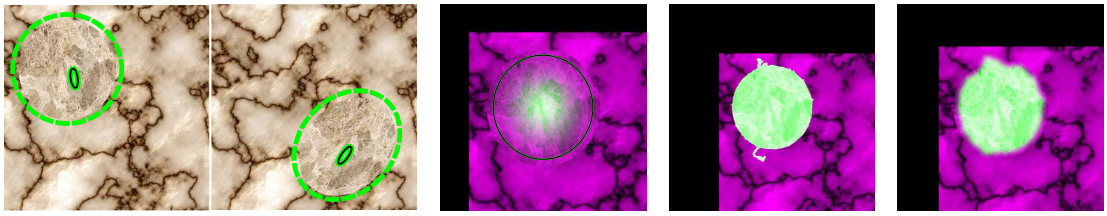


Figure 3.1: **Smooth free-from region.** We tested Algorithm 2 on the test images shown on the top. We show in the middle row (cyan) the smooth window $H(x)$ and in the bottom row (green) the support Ω. The parameter σ has been chosen so that the region can squeeze through the left corridor, but not the upper corridor, and past the bottom line, but not the right line. The computation requires a fraction of a second and the result is consistent even if a large amount of noise is injected. Note the regularizing effect of the kernel g_σ: the boundary of Ω is fairly smooth despite the fact that it is grown by discrete steps (one pixel per time).

under another) from a "collapse" (a portion of the scene being warped onto a subset of measure zero). Therefore, we have to impose restrictions on the local structure of the warping w (or equivalently on the motion and curvature of the underlying 3-D shape), for instance that it be locally continuous and bijective and, for reasons of computational efficiency, finitely parametrized.

We have experimented with three classes of transformations: affine, homography (corresponding to locally planar regions), and thin-plate spline. The last model is well suited to non-planar or deforming (non-rigid) scenes, but it is in



(a) (b) (c) (d)

Figure 3.2: **Effect of the region model in capturing correspondence between regions.** (a) A round region with different texture from the background (left) is moved and deformed affinely on the right image. Within these regions an affine seed is detected by Harris-Affine (in green). Note that the regions do not have well-defined (intensity) boundaries. The goal is to extend the seed to capture the elliptical region based on two-view correspondence. This can be thought of as a "stereoscopic texture segmentation" or as a "motion segmentation" procedure [CS04, WAB03, ATW05]. The effects of the choice of region are shown in (b)-(d). In (b) the region is by construction elliptical, and its domain captures the texture boundary. In (c) the region is free-form, but captures the elliptical shape, modulo some sprouts outside the region where the background happens to match in the two views; in (d) the sprouts are contained by the smooth free-form region.

general not globally invertible (thin-plate splines can fold).[2] We optimize the functional (3.1) using Gauss-Newton as in [BM02]. The derivation of the GN algorithm for these models is standard.

Affine warp and homography. The affine warp and the homography are finite dimensional and they do not need to be regularized, so that $\gamma = 0$ in (3.1).

Thin-plate spline. The thin-plate spline warp is given by [Boo89]

$$w(x) = \begin{bmatrix} T & A & W \end{bmatrix} \begin{bmatrix} 1 \\ x \\ U(\|x - y^{(:)}\|) \end{bmatrix}$$

where (A, T) is an affine transformation, $W \in \mathbb{R}^{2\times K}$ is a matrix of weights, $y^{(:)} = (y^{(1)}, ..., y^{(K)})$ denotes collectively the K control points $y^{(k)} \in \mathbb{R}^2$ and $U(\|x - y^{(:)}\|) = \left[\|x - y^{(:)}\|^2 \log \|x - y^{(:)}\|^2\right]$ is the matrix of the radial basis functions of

[2] An interesting approach to this problem would be to regularize the warps based on priors on the shape [BYJ97, SIF05, VS06]. This, however, is beyond the scope of this chapter.

the spline. The matrices T, A and W are uniquely determined by the transformed control points $\bar{Y} = \begin{bmatrix} w(y^{(1)}) & \dots & w(y^{(K)}) \end{bmatrix}$, yielding a relation

$$w(x; \bar{Y}) = \begin{bmatrix} \bar{Y} & 0 \end{bmatrix} \phi(x; y^{(:)}), \quad \phi(x; y^{(:)}) \in \mathbb{R}^{K+3} \tag{3.5}$$

which is *linear* in the parameters $\bar{Y}$.

Regularization is controlled both by the number of points K and the *stiffness* (bending energy)

$$\mathcal{Q}(\bar{Y}) = \frac{\gamma}{2}(e_1 \otimes e_1 + e_2 \otimes e_2)^\top \operatorname{vec}(\bar{Y} S \bar{Y}^\top)$$

where $\otimes$ denotes the Kronecker's product, $S \in \mathbb{R}^{2\times 2}$ is the *stiffness matrix* and (e_1, e_2) is the standard basis of $\mathbb{R}^2$.

Optimization can be performed with GN,[3] but this is quite costly as each control point has a global influence on the warp. We drastically accelerate the computation by approximating the TPS by a piecewise-affine warp (PWA, [BM01a]) by imposing on its vertices the same regularization (stiffness) of the TPS. The PWA is intrinsically more efficient because each control point has an effect limited to a few triangles of the mesh. We also make use of the inverse compositional algorithm [BM01a] in place of GN, which is much faster.

3.1.3 Matching criterion

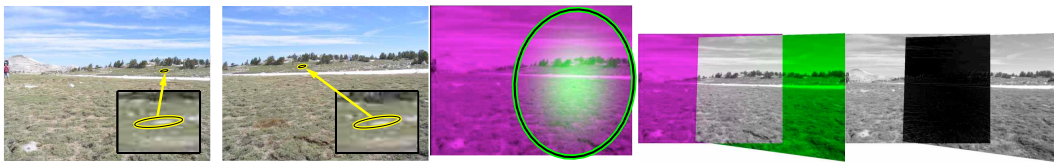
Appearance matching in model (3.1) uses a simple sum of squared difference criterion. The adjustment function $\mu = (\mu_1, \mu_2)$ is an affine scaling $\mu(I_1) = \mu_1 I_1 + \mu_2$ that accounts for global changes in the illumination. As such, μ can be determined in closed form given H and w; alternatively, its optimization can be combined in the GN iteration for w. Coarse illumination factors can be eliminated in other ways. For instance, in some of the experiments we normalize the images via

$$\mu(I) = \frac{I - g_\sigma * I}{\sqrt{g_\sigma * I^2 - (g_\sigma * I)^2}} \tag{3.6}$$

[3] As noted in [LY05], the linearity of (3.5) makes the estimation of the gradient efficient. In order to write the equations for the GN iteration, one also needs the gradient and the Hessian of the stiffness term, which are

$$\frac{\partial \mathcal{Q}(q)}{\partial q^\top} = \sum_{i=1}^{2} e_i^\top \bar{Y} S \otimes e_i^\top, \quad \frac{\partial^2 \mathcal{Q}(q)}{\partial q^\top \partial q} = S \otimes I_2$$

where $q \triangleq \operatorname{vec} \bar{Y}$ and $I_2 \in \mathbb{R}^{2\times 2}$ is the identity matrix (see [Kin96] for more details on the notation).



(a) Initial affine match **(b)** Grown window **(c)** Overlap **(d)** Residual

Figure 3.3: **Growing a mosaic from one feature.** A single feature detected and matched on two images related by an homography grows to capture the entire overlapping domain, yielding a projective mosaic. Here the region model is elliptical and the warp is an homography. (a) Initial affine matches (very small, so we inlay a magnified version) (b) interest region Ω (green ellipse) and window $H(x)$ (green shading) (c) overlap between the two images (perfect overlap results in unsaturated color) (d) residual.

where g_σ denotes an isotropic Gaussian kernel of variance $\sigma^2 I_2$, with I_2 the 2×2 identity matrix. Note, however, that this operator has to be applied to both I_1 and $I_2 \circ w$, which makes the optimization more complex. For further comments on the matching criteria, see Section 3.2.

3.2 Experiments

Global projective registration from a seed. The first simple experiment illustrates how a *single* local feature can grow to encompass the entire image. The two images of Fig. 3.3 are related by an homography; their registration yields a projective "mosaic" which can be obtained efficiently by matching a single feature and then growing it to capture the entire overlapping domain.

Growing increases discriminative power. The second experiment correlates the growth rate of the features to their initial overlap. In [MS03] the quality of an affine seed (Section 3.1) is evaluated by means of the *overlap error*

$$\epsilon \triangleq 1 - \frac{|\Omega_1 \cap \overline{w}^{-1}\Omega_2|}{|\Omega_1 \cup \overline{w}^{-1}\Omega_2|} \tag{3.7}$$

where $\overline{w}$ is the ground truth viewpoint deformation. We used the "viewpoint change" dataset of [MS03] and their code in order to extract Harris-Affine regions and compute ϵ.

We ran the algorithm on several affine seeds using elliptical regions and homo-

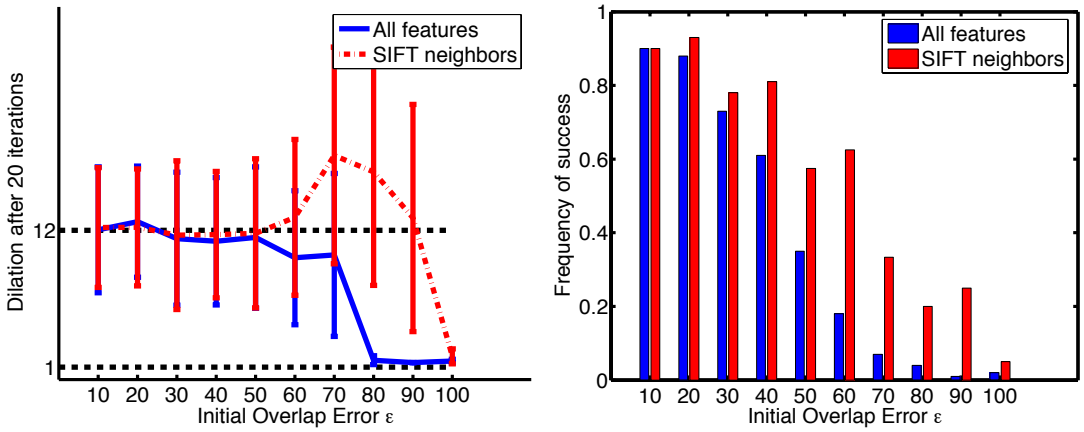


Figure 3.4: **Growing increases discriminative power.** On the left we show the average dilation ratio of the features as a function of the initial overlap error (3.7). As we stopped the algorithm after 20 iterations, the graph gives also an idea of the speed of the dilation. This statistic includes only features with a ratio ≥ 1. On the right we show how frequently this ratio is in fact bigger than 1 (dilation), again as a function of the initial overlap error. The experiment is repeated for all affine seeds and for affine seeds that are SIFT [Low04] neighbors. See text for further details.

graphies. As in the dataset there are almost no occlusions, correct seeds (overlap error less than 100%) should grow indefinitely and incorrect seeds (100% overlap error) should not grow at all. To check wether this is the case, in Fig. 3.4 we plot the average dilation ratio of the matches (left) and the probability of each match of being dilated (right) as a function of the initial overlap error. As desired, the algorithm grows quickly correct matches with up to 50% of initial overlap error and does not grow almost any of the incorrect matches. The algorithm does not perform equally well for correct seeds that have initial error exceeding 50%. This is because we focus on discriminating correct versus incorrect matches rather than trying to fix seeds of poor quality. If this is desired, a robust initialization step can be added (for instance as described in [FTV05]).

As our final goal is to discriminate features beyond the power of their descriptors, we repeated this experiment for those affine seeds that are also SIFT neighbors.[4] The performance of the algorithm does not deteriorate; in particular, almost all matches determined incorrectly by SIFT are invalidated by our criterion, while correct matches are preserved. As a side effect, the algorithm is also more robust to poor initial overlap, probably because seeds which are SIFT

[4]More precisely: for each region Ω_1 of image $I_1(x)$ we selected the three closest regions Ω_2 of image $I_2(x)$ in terms of SIFT distance.

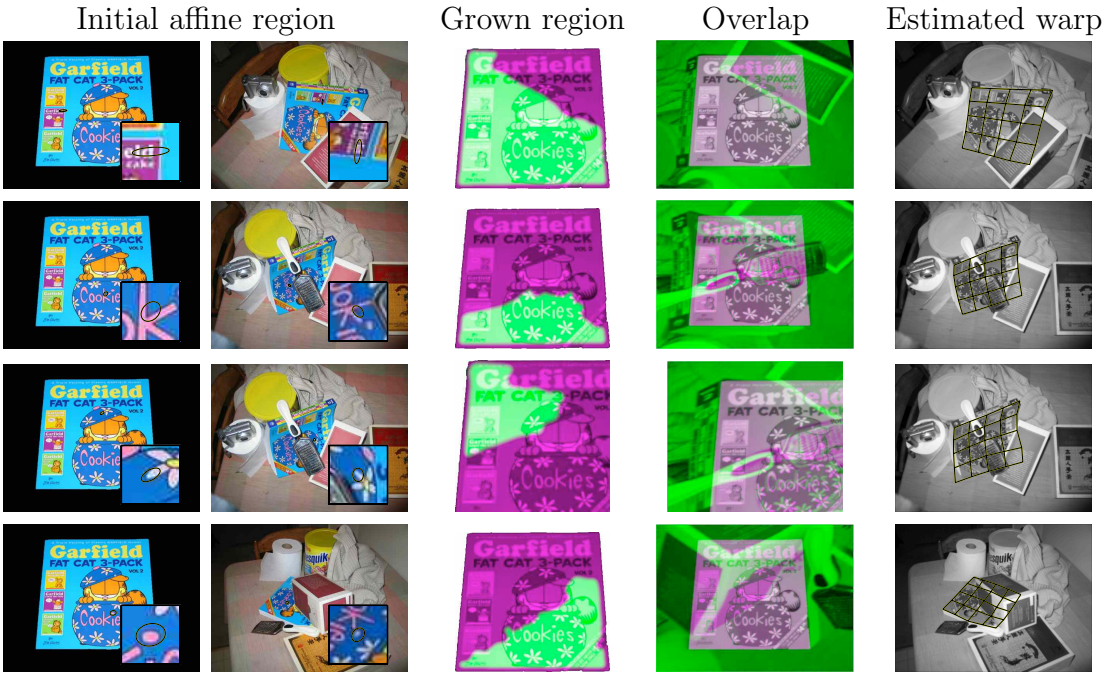


Figure 3.5: **Object detection in clutter.** The left column shows various training/test image pairs. Each pair shows the initial affine match that is grown by the algorithm. Since the object is non-rigid, we used the thin-plate spline model for the warp and the smooth free-form model for the region. In a few cases a portion of the visible area is not included in the region: This is due to the non-uniform illumination (difficult to see with the naked eye but quantitatively significant) which is not compensated by the global model (3.1). It would not be difficult to extend μ to account for more general contrast functions [CCM99] (Section 3.1.3 and 3.2).

neighbors have, if not good geometric correspondence, at least similar appearance.

Finding a known object in clutter. This experiment tests the capability of our method to find – in clutter – an object for which an uncluttered image is given as a training set (or "template"). It is similar in spirit to the experiments of [FTV05, FPZ05] and many other object recognition systems [Low04]. Since we use a deformable object (the Garfield book in Fig. 3.5), we use the thin-plate spline warp and the smooth free-form region model. The figure shows the detection/segmentation results, together with the alignment to the template and the estimated deformation. Note that the latter two quantities are meaningful only locally to the segmented area (so it is not a problem if the template does

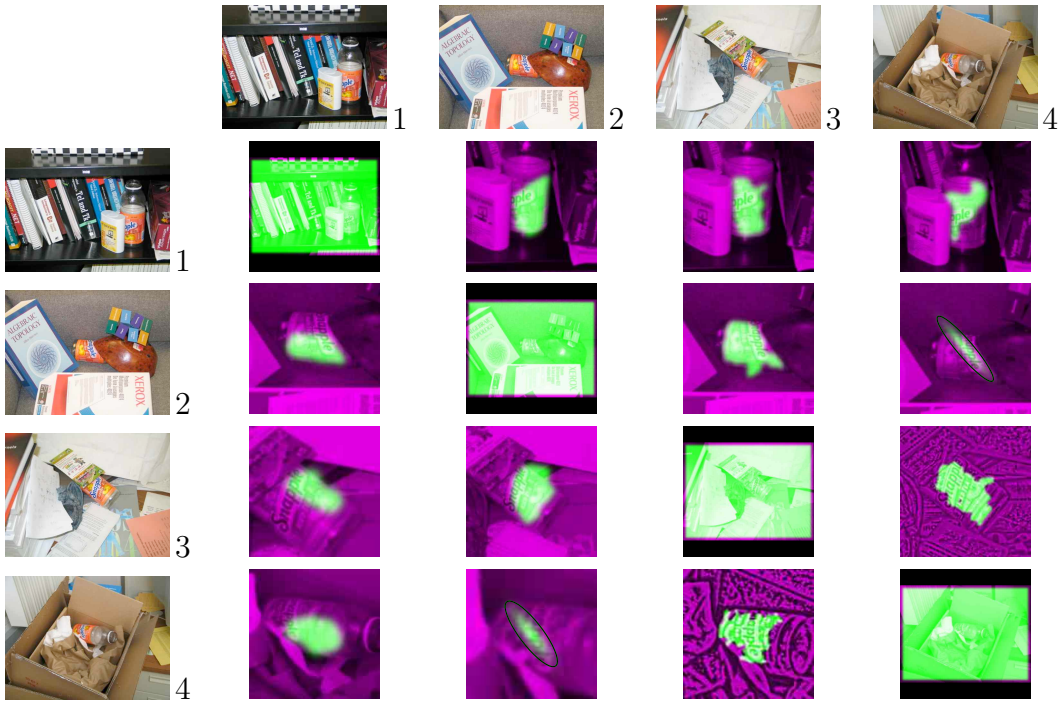


Figure 3.6: **Unsupervised detection in clutter.** We show a series of four images (1-4) portraying the same object (a bottle) in different clutter. The algorithm is tested on each image pair (in both directions) in order to detect the common object. The best SIFT matches of Harris-Affine features on the bottle are expanded and the best result for each pair is kept. Along the diagonal, as we test identical pairs, there is no deformation and the region extends to encompass the whole image domain. In the pairs $(4, 3)$ and $(3, 4)$ the complex reflectance and the particular image deformation (which has high stiffness) requires preprocessing according to eq. (3.6) in order to enable matching. In the pairs $(2, 4)$ and $(4, 2)$ part of the background is almost identical (once color is removed). Therefore, the SSD criterion cannot discriminate between the bottle and the background. One can overcome this ambiguity by using a more constrained region model (elliptical) at the cost of reducing the segmented area. A more principled solution is indicated in Section 3.2.

not align well outside that area.)

Detecting a common object in cluttered scenes. While the previous experiment can be thought of as "supervised" detection, since a template of the object is given, here we address "unsupervised" detection, that is the problem of

detecting the common portion of two images, without a pre-segmented sample of the region of interest. The data are four images that share a single object (a bottle). In Fig. 3.6 we show that the algorithm is generally capable of segmenting the common object from clutter (see the caption for more details).

Discussion

We have proposed a method that increases the information content of local features by maximizing their support. We have shown that the growth rate can be used to validate putative affine matches; the criterion is especially useful to verify matches that have been hypothesized on the basis of the distance between local descriptors. We have seen that the dilated support delineates segments of both known and unseen objects from images with clutter. The latter task is significantly more complex since no uncluttered, unoccluded view of the object of interest is ever available.

Unsupervised detection in clutter is complicated by the fact that certain portions of the background might match accidentally, which is especially easy if the background is uniform. This problem can be properly addressed by ensuring that the region grows where matching is *non-accidental*, that is in areas of the two images that have an appearance which is at the same similar *and* contains "enough structure" [KB03, Tri04]. While this constraint can be imposed as a regularization term on the region, a better solution is to substitute the SSD matching criterion with one that incorporates directly this requirement (see for example [HF01]). Thus the issue is more computational than theoretical, as these measures are significantly more expensive to optimize than SSD.

CHAPTER 4

Evaluating and Learning Viewpoint Invariant Local Features

In the previous chapters we have discussed local viewpoint invariant features and established conditions for their existence. Some of such conditions, such as Lambertian reflection, affine illumination, and no occlusions, are restrictive enough that invariant representations may seem to be inapplicable to most real-world problems. This is even more so for common viewpoint invariant features that, by marrying an affine model of the image deformation, implicitly assume that the local shape is flat. However, local viewpoint invariant features have been successfully applied to a large variety of tasks, suggesting that these methods are robust enough to absorb even relatively substantial deviations from their working assumptions.

In this chapter we focus on the problem of evaluating empirically local viewpoint invariant features in order to assess their applicability and to improve and learn them from data. So far, the effectiveness of features has been measured by recognition performance in an end-to-end task, where the features are one element of the decision process, together with the classifier and the dataset. An empirical test can tell which one is the better feature among the group being tested, but it tells us nothing on how a given feature can be improved, or how performance generalizes to different classifiers and different data sets.

In this chapter we introduce a different methodology for evaluating features. We call this *rational evaluation*.

The first thing we need is *ground truth.* If features were designed for optimality in an end-to-end task (in which case they would have to be co-designed with the classifier), then any labeled training set, along with standard decision-theoretic tools, would suffice. But features are not co-designed with the classifier, so they should be evaluated independently of it. For that we need ground truth. In this chapter we describe a way to design ground-truthed data to evaluate the effectiveness of a given feature based on its underlying (explicit or implicit) invariance assumptions. Such data consists of *synthetic images*, generated with a model that strictly includes the model underlying the invariance assumptions of a given feature. While ultimately an end-to-end system should be evaluated on

the recognition task performed on real images, there is no straightforward way to distill the role of features unless proper ground truth is available.

Once we have ground truth, we need to elucidate the various components of the feature design process, that includes a choice of image domain (the "feature detector"), a choice of image statistic computed on such a domain (the "feature descriptor"), and a choice of decision function ("feature matching") that becomes the elementary tool of the classifier downstream.

The effect of this procedure is not just a number to rank existing features based on how well they perform, when coupled with a given classifier, on a given dataset. A rational comparison also provides ways to improve the design of the feature, as we illustrate with an example. A similar approach could be followed to design better descriptors, and also better detector.

This chapter is part of a three-prong approach we have been developing for designing and evaluating local features: In [VF08] we provide a reliable open-source implementation of some of the most common local features. In this manuscript we describe a methodology to compare local features. Finally, in [Ved08] we provide code to generate synthetic test images, as well as a number of already rendered samples.

4.1 Empirical studies of local features

Because of their prominent role in recognition systems, local features have been the subject of considerable attention in the computer vision community. Due to the difficulty of extracting adequate ground truth, however, direct evaluation (i.e. not part of an end-to-end system) has been mostly limited to planar scenes [MTS04] designed to fit the conditions for which the features were designed. While local features are usually designed to be invariant to a simple class of transformations (say affine, or projective, corresponding to the assumption of planar scenes), it is the behavior of the feature in the presence of violations of such assumptions that determines its effectiveness. Therefore, it is important that the ground truth reflects conditions that supersede the underlying assumptions.

The need to test features on more challenging data has driven some to employ synthetic datasets [Roc03, LJ05], although the resulting images lacked in visual realism. More realistic data was used by [FB05] to infer ground truth via stereo. This procedure, however, is difficult to scale up to be representative of the complexity and variability of natural scenes. The most extensive collection of real objects to-date is [MP05], where a selection of (uncluttered) objects was placed on a calibrated turntable in front of a blue screen. Thousands of features were mapped from small-baseline image pairs to wide-baseline views in

a semi-automated fashion. A semi-synthetic data set was produced in [RB05] by gathering range images acquired with a laser scanner and generating a number of artificial views by rotating the data. [WB07] recognized the importance of obtaining wide-baseline feature deformation data for the study of viewpoint-invariant features and used structure from motion to estimate re-projection of point features from a large number of views of real scenes. Unfortunately this technique provides only limited ground truth information (i.e., sparse 3-D points estimated from the images themselves) and is laborious to collect, especially for controlled experimental conditions. To this date, however, there is no extensive data set that can scale up to an arbitrarily large number of scenes, where the geometry of the scene, its reflectance, the illumination, sensor resolution, clutter, and lens artifacts can be controlled and analyzed by the user.

In order to make a useful tool for evaluating features, however, it is not sufficient to generate (even a lot of) synthetic scenes with ground truth. We have to develop a methodology that allows us to evaluate different aspects of the feature matching process in isolation if we want to rationally improve the design of existing features. The following section does just that. While the nomenclature we introduce may seem like a burden to the reader at first, it will make the evaluation process more rigorous and unequivocal.

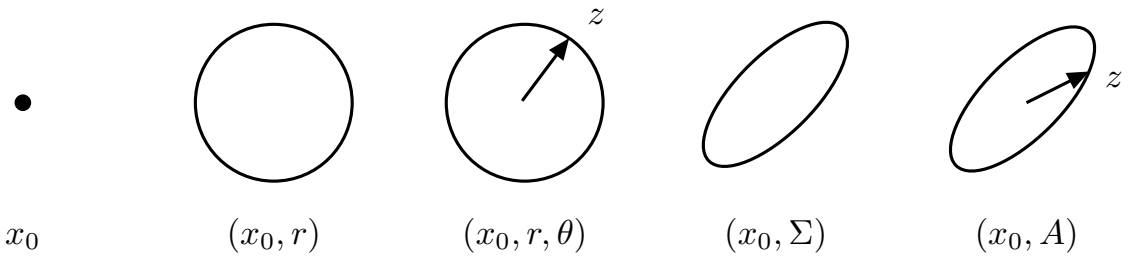
4.1.1 Feature extraction pipeline

The extraction of viewpoint invariant features was discussed in great detail in Chapter 2. To the end of this chapter, it is useful to break the extraction of invariant features in the following steps.

Detection. Given an image, the *co-variant detector*, or simply "detector", selects a number of image regions. It is designed to extract the same (deformed) regions as the image deforms under a viewpoint change. A specific detector (SIFT, Harris-Laplace, Harris-Affine) is compatible by design with a certain family of such deformations (usually a group, e.g. similarities, affinities [MTS04]). Section 4.2.1 develops a formal model of this step.

Canonization. The co-variant regions are *canonized*, i.e. deformed to a standard shape. This process compensates (in part or entirely) for deformations induced by the companion transformations. It is often assumed that such transformations form a group, and therefore they can be undone (inverted).

Description. The *descriptor* computes a statistic of the image on the canonized co-variant regions. This process may eliminate, or render the descriptor insensitive to, additional deformations which are not removed by canonization.



frame	**companion warps**	**fixed subset**
point	homeomorphisms	translations
disk	similarities	translations and scalings
oriented disk	similarities	similarities
ellipse	affinities	affinities up to residual rotation
oriented ellipse	affinities	affinities

Figure 4.1: **Feature frames. Top.** The figure depicts the five classes of feature frames, together with their parameters and the selected point z used to represent orientation. From left to right: point, disk, oriented disk, ellipse, oriented ellipse. **Bottom.** Association of frame types to companion warps used in this chapter.

Matching. A *similarity measure* is used to compare invariant descriptors to match regions in different images.

4.2 Constructing ground-truth

In Section 4.2.1 we introduce an idealized model of the output of co-variant detectors and in Section 4.2.2 a model of feature correspondences. These will be used in the empirical analysis in Section 4.2.3 and 4.2.4.

4.2.1 Modeling the detector

In Chapter 2 we have seen that viewpoint has a direct effect on the *geometry* of local features, resulting in a deformation of their support and appearance. The purpose of a *(co-variant) detector* is to select regions that warp according to, and hence track, image deformations induced by viewpoint changes.

There is a correspondence between the type of regions extracted by a detector and the deformations that it can handle. We distinguish transformations that

are (i) compatible with and (ii) fixed by a detector. For instance, a detector that extracts disks is compatible with, say, similarity transformations, but is not compatible with affine transformations, because these in general map disks to other type of regions. Still, this detector does not fix a full similarity transformation, because a disk is rotationally invariant and that degree of freedom remains undetermined. These ideas are clarified and formalized by the next definitions.

Frames. Typically one models the output of a detector as image regions, i.e. as subsets of the image domain [MTS04] . However, many popular detectors produce "attributed regions" instead (for example the SIFT detector [Low04] produces oriented disks rather than just disks). Since such attributed regions are ultimately used to specify image transformations, in this work we refer to them as "frames." Thus a *frame* is a set $\Omega \subset \mathbb{R}^2$ possibly augmented with a point $z \in \Omega$. For example, a disk is a set $\Omega = \{|x - x_0|_2 < r\}$ and an oriented disk is the combination (Ω, z) of a disk and a point $z \in \Omega$, $z \neq x_0$ representing its orientation[1] (as the line connecting the center x_0 to z). Here we consider the following classes of frames (see Fig. 4.1), that capture the output of most detectors found in the literature:

- **Points.** Points are determined by their coordinates x_0.
- **Disks.** Disks are determined by their center x_0 and radius r.
- **Oriented disks.** Oriented disks are determined by their center x_0, radius r and orientation θ.
- **Ellipses.** Ellipses are determined by their center x_0 and the moment of inertia (covariance) matrix

  $$\Sigma = \frac{1}{\int_\Omega dx} \int_\Omega (x - x_0)(x - x_0)^\top \, dx.$$

  Note that Σ has three free parameters.
- **Oriented ellipses.** Oriented ellipses are determined by the mapping $A \in GL(2)$ which brings the oriented unit circle Ω_c onto the oriented ellipse $\Omega = A\Omega_c$.

Frames fix deformations. Each type of frame (point, disk, oriented disk, etc.) can be used to fix (and undo, by canonization) certain image transformations.

[1] We prefer to use a point z rather than specifying the orientation as a scalar parameter because this representation is easier to work with and can be easily generalized to more complex feature frames.

In fact, given a pair of frames Ω_1 and Ω_2, the equation $\Omega_2 = w\Omega_1$ determines (partially or entirely) the warp w. Therefore, a frame Ω acts as a reference frame to specify deformations. This fact is captured by the following definitions:

- **Frame deformations.** For what concerns our discussion, an image deformation (warp) w is simply a transformation $\mathbb{R}^2 \to \mathbb{R}^2$ of the image domain, and wI denotes the image $I(w^{-1}(x))$. Such deformations apply to frames as well: Given a frame (Ω, z), the warped frame $w(\Omega, z)$ is the pair $(w\Omega, w(z))$. Note that, if present, the selected point z is moved too; later we will use the shorthand notation $w\Omega$, still meaning that the warp applies to both the set and the selected point z.
- **Closed, complete, and free frames.** Frames are *closed* under the deformations $\mathcal{W}$ if warping a frame by $w \in \mathcal{W}$ does not change their type. For example, disks and oriented disks are closed under similarity transformations and ellipses and oriented ellipses are closed under affine transformations. We say that a frame is *complete* for a certain set of transformation $\mathcal{W}$ if the equation $\Omega_2 = w\Omega_1$ admits at most one solution $w \in \mathcal{W}$. We also say that the frames are *free* on the set $\mathcal{W}$ (as in "free generators") if such an equation has a solution for all possible pairs of frames Ω_1 and Ω_2.

When analyzing a detector, it is important to specify both the type of frames it produces and the class of transformations that are assumed, which we call *companion warps*. Notice in fact that each frame type can be used in connection with different types of transformation, so both choices must be specified. In the rest of the chapter we focus on the most natural cases, summarized in Fig. 4.1. For instance, from the table we read that disks are used in conjunction with similarity transformations (their companion warps), but are expected to fix only a subset of them.[2]

4.2.2 Modeling correspondences

In the previous section we have modeled the detector as a mechanism that extracts (co-variant) frames. Operatively, the output of the detector is used to establish frame-to-frame correspondences between multiple images of the same object. For evaluation purposes, it is therefore necessary to extract sets of corresponding frames. This idea is captured by the following definitions.

[2]Notice also that frames (i) are closed under the companion warps, (ii) complete for a subset of these, and (iii) free on the complete subset. Property (iii) is not always satisfied by real detector. For instance, maximally stable extremal regions [MCU02] have arbitrary shape Ω, but their companion warps are just affine transformations. This means that the equation $\Omega_1 = w\Omega_2$ may not have a solution.

View sets (multiple views). A *view set* [LLF05] $\mathcal{V}$ is a collection of images $(I_1, \dots, I_n)$ of a scene taken under different viewpoints. Under Lambertian reflection and other assumptions [VS05], any image $I_j(x)$, $x \in \Lambda$ in a view set is obtained from any other image I_i by a deformation $I_j(x) = I_i(h_{ij}(x)) \doteq (h_{ji}I_i)(x)$. Such a deformation arises from the equation

$$h_{ij}(x) = \pi(R_{ij}\pi_j^{-1}(x) + T_{ij}), \quad x \in \Lambda \tag{4.1}$$

where π is the perspective projection and $\pi_j^{-1}(x)$ is the pre-image of pixel x from viewpoint j and (R_{ij}, T_{ij}) is the camera motion from view j to view i. Also note that, due to occlusions and other visibility artifacts, equations $I_j = h_{ji}I_i$ may have only local validity, but this is sufficient for the analysis of local features.

Co-variant frame sets (correspondences). A *(co-variant) frame set* $\mathcal{F}$ is a selection of frames $(\Omega_1, \dots, \Omega_n)$, one for each image of a view set $\mathcal{V} = (I_1, \dots, I_n)$, that are linked by the same deformations of the view set, i.e.

$$\Omega_i = h_{ij}\Omega_j$$

where h_{ij} is given by (4.1). It is useful to think of co-variant frames as collections of geometric elements (such as points, regions, bars and so on) that are "attached" to the images and deform accordingly. Co-variant frames define the support of features and, by tracking image deformations, enable canonization.

Frame sets enable canonization. By mapping a co-variant frame Ω_i to a *canonical variant* Ω_0, the equation $\Omega_0 = w_i\Omega_i$ defines a warp w_i which undoes the local image deformation in the sense that the local appearance w_iI_i is constant through the view set $i = 1, \dots, n$. For example, mapping an oriented disk Ω_i to the disk $\Omega_0 = w_i\Omega_i$ of unit radius and orientation $z = (0, 1)$ undoes the effect of a similarity transformation. Doing so for an un-oriented disk does the same up to a residual rotation.

Remark 8. Operatively, a detector can attach a frame to the local appearance only if this has enough "structure:" We can associate a disc to a radially symmetric blob, but we cannot (uniquely) associate an oriented disc to it because the image is rotationally symmetric. It should be noted, however, that this is irrelevant to the end of canonization: As long as the most specific frame is attached to each image structure, canonization will make the local appearance constant. For example, we cannot associate an oriented disk to a symmetric blob, but this is irrelevant as the residual rotation does not affect the local appearance by definition.

While so far we have just listed nomenclature, the next section will tie these concepts to the empirical process of evaluating features relative to ground truth.

4.2.3 Ground-truth correspondences

The main obstacle to the practical applicability of the concept of co-variant frames given in Section 4.2.1 is that the actual image transformations h_{ij} (4.1) are rarely of the idealized types because world surfaces are seldom flat, so the actual pixel motion $h(x)$ is more complex than a similarity or other simple transformation that we might assume. Furthermore, due to occlusion, folding, visibility and reflectance phenomena, images in a view set are rarely related to one another by simple deformations of their domains.

Therefore, we relax our requirement that the frames represent exactly the image deformations, and look for the best fit. We propose the following operational construction of a ground-truth frame set (i.e. of ground-truth correspondences):

1. We select a *reference view* $I_0 \in \mathcal{V}$ and an initial frame Ω_0 in I_0. Then, given an alternate view $I_i \in \mathcal{V}$, we map the points x of the frame Ω_0 to points $y = h(x)$ of the alternate view. To this end we use the three-dimensional ground truth in order to estimate the actual *motion* of the pixels from (4.1), which does not depend on the local appearance. Note that $h(x)$ is well defined even when some pixels $y = h(x)$ are occluded.

2. We search for the warp $w \in \mathcal{W}$ that best approximates h, for example by solving
$$w = \operatorname*{argmin}_{v \in \mathcal{W}} \int_{\Omega_0} \|h(x) - v(x)\|^2 \, dx. \tag{4.2}$$
Algorithms that solve efficiently this problem for the transformation classes $\mathcal{W}$ of interest are reported in Appendix 4.A. Notice that one can choose a cost different from (4.2), and we illustrate a different example in (4.3).

3. We map the frame Ω_0 to the frame $\Omega_i = w\Omega_0$ by the estimated warp w.

Occlusions and foldings. The procedure we have delineated is simple, but can be inadequate if the frame Ω_0 contains an occlusion or a strong depth discontinuity, which induces a highly non-linear or discontinuous motion $h(x)$. In such cases, instead of trying to capture the motion of all pixels simultaneously, one can expect the detector to track only the *dominant motion*, i.e. the motion of the background or the foreground, depending on which one occupies the larger portion of the region, or "patch." To this end, before executing step (4.2) we consider splitting the patch in half. We sort the pixels $x \in \Omega_0$ by depth and we search for a (relative) gap in the sorted values which is bigger than a threshold. If we find it, we restrict equation (4.2) only to the pixels before or after the split, based on majority rule.

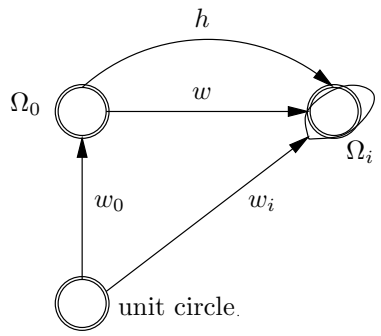


Figure 4.2: **Deviation.** The figure illustrates the quality index (4.3). Intuitively, the deviation is the norm of the difference of the true motion h and the estimated motion w, normalized by projecting on the unit disk (canonical configuration). Normalization reflects the fact that features are being canonized before the descriptor is computed.

Quality indices

The procedure just delineated finds, in each image I_i of a view set, the best matching frame Ω_i. However, not all matches are equal. Some may approximate very well the underlying image transformation, while others may be poor fits, due for instance to occlusions or strong non-linear distortions. For the evaluation of a real-world detector, it is important to assess which of these ground-truth matches are close to the idealized working assumptions, and which are not. To this end, we propose the following quality indices:

Deviation. This index measures the "non-linearity" of the warp. Let $w_0 = (A_0, T_0)$ and $w_i = (A_i, T_i)$ be the affine transformations that map the unit (oriented) circle on the reference frame Ω_0 and the alternate frame Ω_i; let w be the companion warp $\Omega_i = w\Omega_0$ that approximates the true motion $h(x)$. The deviation index is a normalized version of the average square residual $|h(x) - w(x)|^2$, obtained by conjugation with w_i:

$$\mathrm{dev}(w, h, \Omega_i) = \frac{1}{\pi} \int_{\{x:|x|<1\}} |w_i^{-1} \circ (h \circ w^{-1}) \circ w_i(x) - w_i^{-1} \circ w_i(x)|^2 \, dx. \quad (4.3)$$

The formula has a simple interpretation (Fig. 4.2). It is the average squared residual $|h(x) - w(x)|^2$ remapped to the canonized version of the frame Ω_i. Noting that, by definition, $w = w_i w_0^{-1}$ and all but h are affine warps, we

find (see Appendix 4.A)

$$\mathrm{dev}(w, h, \Omega_i) = \frac{1}{\pi} \int_{\{x:|x|<1\}} |A_i^{-1}(h \circ w_0(x) - w_i(x))|^2 \, dx. \tag{4.4}$$

In practice, we estimate the values of h on the pixels $\widehat{x}_i$ of the region Ω_0; in this case we use the formula

$$\mathrm{dev}(w, h, \Omega_i) \approx \frac{1}{|\Omega_0|} \sum_{\tilde{x}_i \in \Omega_0} |A_i^{-1}(h(\tilde{x}_i) - w(\tilde{x}_i))|^2 \tag{4.5}$$

which preserves its validity even if the region Ω_0 intersects the image boundaries.

Visibility. This is the portion of the frame Ω_i that falls inside the image boundaries.

Occlusion. This is the portion of the region Ω_0 that is occluded in the alternate view I_i. Occluded pixels $x \in \Omega_0$ are determined empirically by checking whether their pre-image from the reference view I_0 and the alternate view I_i correspond, i.e.

$$R_{i0}\pi_0^{-1}(x) + T_{i0} \neq \pi_i^{-1}(h(x)).$$

Splitting. This is the portion of frame Ω_0 which is accounted for in the estimation of the dominant motion and ranges from 1 (complete frame) to 1/2 (half frame).

Fig. 4.4 illustrates the quality indices for a number of co-variant frames.

4.2.4 Comparing ground-truth and real-world correspondences

One may regard a real-world detector as a mechanism that attempts to extract co-variant frames from the local appearance only. This task is difficult because, while the definition of correspondences (co-variant frames) is based on the knowledge of the ground-truth transformations h_{ij}, these are not available to the detector, and cannot be estimated by it as this would require operating on multiple images simultaneously [VS05].

There exist several mechanisms by which detectors are implemented in practice. The simplest one is to randomly extract a large number of feature frames so that eventually some frame sets will be filled "by chance". Albeit very simple, this process poses a high computational load on the matching step. More refined approaches, such as Harris, SIFT, attempt to attach feature frames to specific patterns of the local image appearance (for example SIFT attaches oriented disks

to "image blobs"). This enables the detector to explicitly "track" image transformations while avoiding the exhaustive approach of the random detectors. In general, constructing a co-variant detector requires that it be possible to associate co-variant frames to images based on the (local) appearance only. So, for example, we can associate disks to image "blobs," as long as we make sure that, as the blobs move, the disks move according.

No matter what the underlying principle on which a detector is based, the quality of the correspondences established by a real-world detector can be expected to be much lower than the ideal correspondences introduced in Section 4.2.3, which, under the limited expressive power of the regions extracted (e.g., disks are limited to similarity transformations), optimally approximate the actual image transformations. Thus ground-truth frame sets can be used to compare and evaluate the performance of the real-world detectors.

To asses the performance of a detector, we therefore measure how much the approximate co-variant frame $\tilde{\Omega}_i$ extracted by the detector deviates from the ground truth co-variant frame Ω_i defined in Section 4.2.3. To this end, we use the same criterion introduced in 4.3, and compare the deviation of the ground truth and estimated motions. Consider first the simpler case in which frames are complete for the companion transformations $\mathcal{W}$ (for example oriented disks for similarity transformations). Let $w_i = (A_i, T_i)$ and $\tilde{w}_i = (\tilde{A}_i, \tilde{T}_i)$ be the unique (by hypothesis) warps in $\mathcal{W}$ that bring the oriented unit circle to the frames Ω_i and $\tilde{\Omega}_i$. Let $w = \tilde{w}_i w_i^{-1}$ be the transformation mapping Ω_i to $\tilde{\Omega}_i$; the desired transformation h is the identity $\mathbf{1}$ and by plugging back into eq. (4.3) (see Appendix 4.A) we obtain the *oriented matching deviation*

$$\text{dev}(w, \mathbf{1}, \tilde{\Omega}_i) = \frac{1}{4}\|\tilde{A}_i^{-1}A_i - \mathbf{1}\|_F^2 + |\tilde{A}_i^{-1}(T_i - \tilde{T}_i)|^2 \tag{4.6}$$

where $\|\cdot\|_F$ is the Frobenius matrix norm.

In case the frames are not oriented, w_i and $\tilde{w}_i$ are known only up to right rotations R and $\tilde{R}$ and we have

$$\text{dev}(w, \mathbf{1}, \tilde{\Omega}_i) = \frac{1}{4}\|\tilde{R}^\top \tilde{A}_i^{-1}A_i R - \mathbf{1}\|_F^2 + |\tilde{A}_i^{-1}(T_i - \tilde{T}_i)|^2 \tag{4.7}$$

where we used the fact that the Euclidean norm $|\cdot|$ is rotationally invariant. We obtain the *un-oriented matching deviation* by minimizing over R and $\tilde{R}$ (see Appendix 4.A)

$$\text{dev}(w, \mathbf{1}, \tilde{\Omega}_i) \quad = \quad \frac{1}{4}\left(\|\tilde{A}_i^{-1}A_i\|_F^2 + 2(1 - \text{tr}[\Lambda])\right) \; + \; |\tilde{A}_i^{-1}(T_i \; - \; \tilde{T}_i)|^2. \tag{4.8}$$

where Λ is the matrix of the singular values of $\tilde{A}_i^{-1}\tilde{A}_i$.

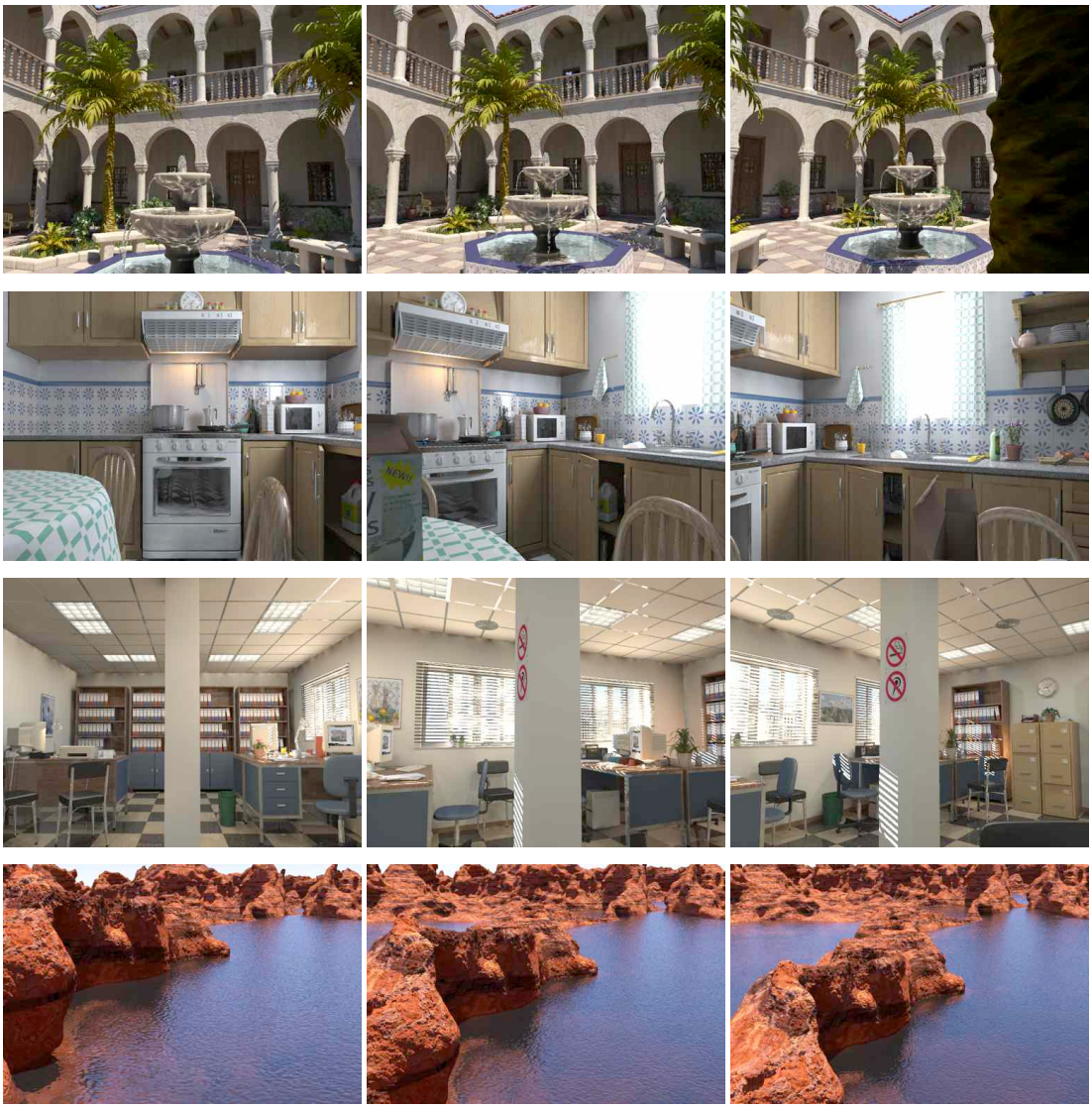


Figure 4.3: **View sets**. We show a small portion of a few view sets. These are synthetic rendering of scenes from [Piq06] and come with accurate ground truth. Each image requires several CPU hours to be generated. The data set, which required a large computer cluster to be computed, is available to the public at [Ved08].

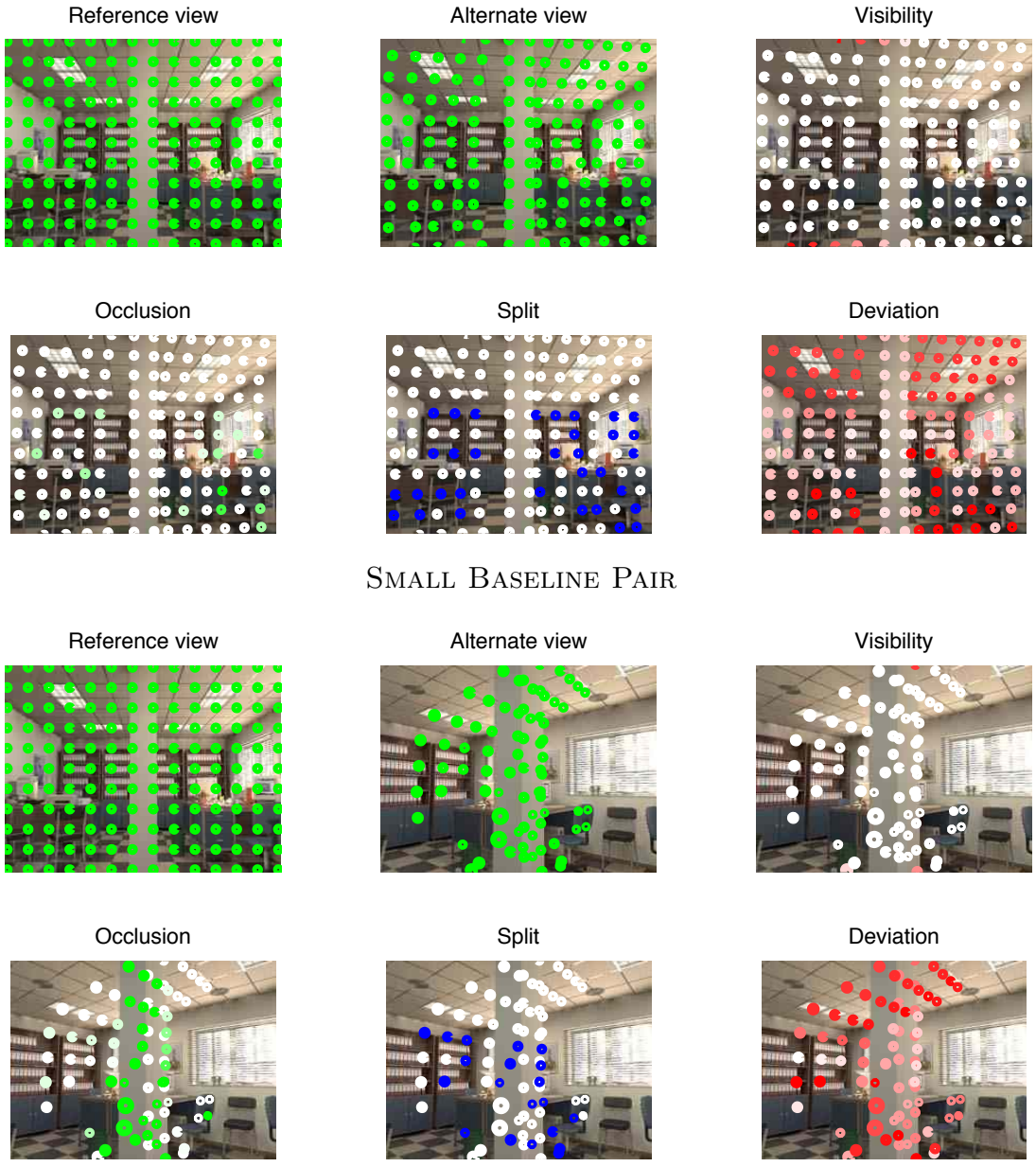


Figure 4.4: *Quality indices.* We show examples of the four quality indices (visibility, occlusion, split, and deviation) for a selection of of features in a small baseline pair (**top**) and wide baseline pair (**bottom**). Quality indices signal if, and for what reason, a certain match deviates from the idealized working assumptions. Brighter colors indicate higher quality patches.

Figure 4.5: **Frame sets (correspondences)**. We show portion of two ground-truth frame sets (Section 4.2.1) as canonized patches. Each patch is obtained by un-warping to a canonical configuration the corresponding co-variant frame. Note that, due to complex reflectance and geometric phenomena, canonization never yields perfectly aligned patches.

4.2.5 The data

Based on the concepts that we have introduced in the previous sections, we now describe a new dataset to evaluate and learn visual invariants. The dataset is composed as follows:

View sets. View sets are obtained from a number of three dimensional scenes shot from different vantage points (Fig. 4.3). Each image comes with accurate geometric ground truth information in the form of a *depth map*. This data can be acquired by means of special instrumentation (e.g. a dome and a laser scanner), but in this work we propose to use high quality synthetic images instead. This has the advantages that (a) no special instrumentation is needed; (b) much more accurate ground truth can be generated; (c) automated data extraction procedures can be easily devised. Our data is largely based on publicly available 3-D scenes developed by [Piq06] and generated by means of the freely available POV-Ray ray tracer.[3] Currently we work with a few such scenes that include natural as well as man-made environments; for each scene we compute a large number of views (from 300 to 1000) together with their depth map and camera calibration. The camera is moved to cover a large volume of space (it is more important to sample the camera translations rather than the camera rotations as additional orientations can be simulated exactly in post-processing by simple homographies).

Frame sets. For each view set we compute a number of co-variant frame sets (Fig. 4.5). This happens as follows:

- We choose a detector (e.g. SIFT) and a number of *reference views* in the view set.
- We run the detector on each reference view to extract reference frames.
- We re-map each reference frame to all other views as explained in Section 4.2.3 and we compute the four quality indices. The resulting collection of frames is a co-variant frame set. Based on the quality indices, frames can be filtered out in order to generate data of varying difficulty.
- Optionally, we run the detector on each non-reference view as well and we match each co-variant frame to a detected frame by minimizing the quality index introduced in Section 4.2.4. We then record the matching score and matched frames along with the co-variant frame set. This is

[3] We actually use a customized version to export the required ground truth data. Patches are available from [Ved08].

the approximation of the co-variant frame set obtained from the real-world detector.

In practice, only a few reference views (from 2 to 4) are selected for each view set. This alone is sufficient to generate several thousand frame sets, and most frame sets count dozens of frames from different views. Eventually it is easy to generate data in the order of millions frames. The data comes with quality indices so that interesting subsets can be easily extracted.

4.3 Learning to compare invariant features

The data developed in Section 4.2 can be used to:

1. Learn natural deformation statistics, similarly to [RB05], but in a wide-baseline setting.
2. Evaluate/learn detectors that compute good approximations of co-variant frames.
3. Evaluate/learn descriptors, given either the co-variant frame sets or the frame sets matched to the output of any practical co-variant detector.
4. Evaluate/learn similarity measures between descriptors.

Here we limit ourselves to the last task for the purpose of illustration of the use of the dataset. While the improvements we expect are limited, since we are only operating on the last ingredient of the feature matching pipeline, the results are readily applicable to existing systems.

More concretely, given a frame Ω_0 in a reference view I_0 and an alternate view I_1, we study two problems: (i) how to find the frame Ω_1 of I_1 that matches Ω_0 (Section 4.3.1) and (ii) when to accept a putative match $\Omega_0 \leftrightarrow \Omega_1$ in order to minimize the expected risk of making a mistake (Section 4.3.2). We focus on SIFT features (both detector and descriptor) because of their popularity, but any other similar technique could be studied in this fashion.

4.3.1 Learning to rank matches

Given a frame Ω_0 of the reference view I_0, its descriptor f_0 and an alternate view I_1, we order the frames $\Omega_1, \Omega_2, \ldots$ of I_1 based on the similarity ϕ of their

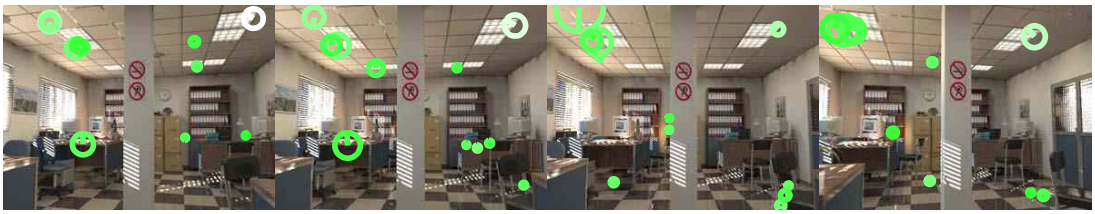


Figure 4.6: **Learning SIFT metric**. We show four views of a co-variant frame (the frame on the ceiling lamp) and the ten most similar SIFT features in term of their SIFT descriptors. Shades of green are proportional to the descriptor ϕ_2-similarity to the reference descriptor.

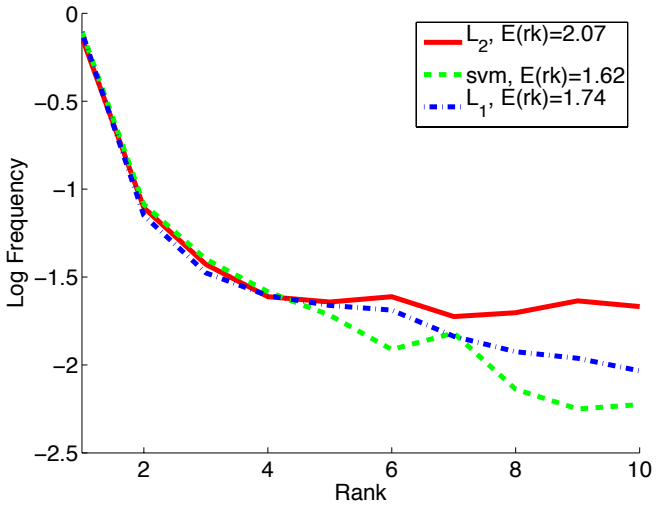


Figure 4.7: **Learn to rank matches**. The figure shows the distribution of the rank values of the correct feature match, averaged for frame sets in our database. The SVM-based ranking outperforms the naive ϕ_1 and ϕ_2 ranking, resulting in an expected rank of 1.62, 1.74 and 2.07 respectively.

descriptors f_1, f_2, ... to f_0, i.e.

$$\phi(f_0, f_1) \leq \phi(f_0, f_2) \leq \dots.$$

Ideally the similarity function ϕ is chosen in such a way that the correct matching frame Ω_i is ranked first.

Normally the similarity of a pair of SIFT descriptors is just their L_1 or L_2 distance, i.e.

$$\phi_p(f_0, f_1) \doteq \|f_0 - f_1\|_p, \qquad p = 1, 2.$$

Here we show how a similarity ϕ can be learned that outperforms both ϕ_1 and ϕ_2. We do this by setting up a learning problem as follows: Based on our ground-truth data, we sample pairs of corresponding descriptors (f_0, f_1) from a frame set and a pair of non-corresponding descriptors (f_0, f) randomly. We then learn a binary classifier $D(f_0, f_1)$ for the task of deciding whether f_0 and f_1 are the descriptors of corresponding features. Following [HBW04], we assume that the classifier is in the form $[\phi(f_0, f_1) \leq \tau]$ for some function ϕ (for example this is the case for a support vector machine (SVM) but one could use boosting as well [ZSG06]) and we use ϕ as a similarity measure.

Re-ranking. Since the goal is to improve ϕ_1 and ϕ_2 (which have already good performance), instead of choosing negative descriptors f completely at random, we select them among the descriptors of the alternate view that have ϕ_p-rank smaller or equal to 10 (Fig. 4.6). In testing, the learned similarity ϕ is then used to re-rank these top matches in hope of further improving their ranking. This approach has several benefits: (a) since the computation of ϕ is limited to a few features, testing speed is not a concern; (b) experimentally we verified that the top ten features include very often the correct match; (c) the learning problem has a much more restricted variability because features are ϕ_p-similar by construction.

Learning. We select about 500 frame sets (matched to actual SIFT frames – see Section 4.2.4) and we extract their reference frames Ω_0 and descriptors f_0; for each of them we select about 10 alternate views and we extract the relative co-variant frame Ω_1 and descriptor f_1. In this way, we form about 5,000 positive learning pairs (f_0, f_1). For each positive pair (f_0, f_1), we add about 10 negative pairs (f_0, f) formed as explained for a total of 55,000 examples. The data is used to learn an SVM with polynomial kernel.

Testing. While ϕ is optimized for classification, here we are interested in its ranking performance. Thus testing is done by taking a large number of fresh frame sets and averaging the ranking performance of ϕ over them. Fig.4.7 shows that learning can indeed improve the basic similarities.

4.3.2 Learning to accept matches

Once putative matches have been proposed, for example based on the similarity metric ϕ, we need to accept or reject them based on some notion of expected risk. In some applications we also want to order matches by reliability [CM05a]. [Low04] proposes a simple score that can be used to both accept and rank putative matches. With notation similar to the previous section, denote f_0 the descriptor

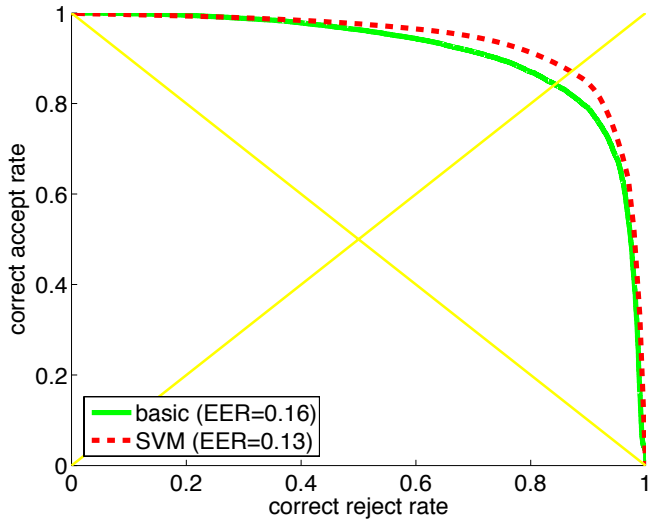


Figure 4.8: **Learning to accept matches.** The figure compares the ROC curves of the function $D(f_0, f_1, f_2)$ in its basic form and as learned by an SVM.

of a reference frame Ω_0 and by f_1 and f_2 the two ϕ-top matches in an alternate view. We define the belief that the match $\Omega_0 \leftrightarrow \Omega_1$ is correct as

$$P(\Omega_0 \leftrightarrow \Omega_1 | f_0, f_1, f_2) = 1 - \frac{\phi(f_1, f_0)}{\phi(f_2, f_0)}.$$

Here we use $\phi = \phi_2$ to be compatible with [Low04]. This quantity can be directly used to rank and accept matches, the latter by comparing it to a threshold τ, getting the decision function

$$D(f_0, f_1, f_2) = \left[P(\Omega_0 \leftrightarrow \Omega_1 | f_0, f_1, f_2) \leq \tau \right]. \tag{4.9}$$

As $D(f_0, f_1, f_2)$ is a decision function, it can also be learned by means of some standard technique, which we do next.

Data and learning. Data is obtained similarly to Section 4.3.1, with the obvious adaptations. Learning is still performed by an SVM based on a polynomial kernel.

Testing. In Fig. 4.8 we plot the ROC curve of (4.9) as τ is varied and the ROC curve of the SVM-based decision function $D(f_0, f_1, f_2)$. The equal error rate is lowered from 0.16 to 0.13 showing again that learning can be used to improve the basic method.

Discussion

We have presented an extensive, flexible, accurate ground-truthed dataset for matching local invariant features. Together with it, we have presented a methodology to evaluate local features, and illustrated their use to not only evaluate, but also improve current algorithms. Our analysis separates the effects of a feature detector, a descriptor, and a matching algorithm, and our dataset is aimed at facilitating the collection of natural image deformation statistics induced by viewpoint changes, and at incorporating them in the design of better features. A similar procedure can be followed to incorporate natural reflectance, illumination and occlusion statistics, which is obviously beyond the scope of this chapter. We have demonstrated the use of the dataset to improve on the matching score in matching SIFT features. Albeit the quantitative improvement is not stunning, it is sufficient to illustrate the potential advantage associated in the use of the dataset and the associated methodology for evaluating local features.

4.A Calculations

Justification of eq. (4.5)

By changing variable in (4.4) we obtain

$$\begin{aligned}\operatorname{dev}(w, h, \Omega_i) &= \frac{1}{\pi \det A_0} \int_{\Omega_0} |A_i^{-1}(h(\tilde{x}) - w(\tilde{x}))|^2 \, d\tilde{x} \\ &\approx \frac{1}{\pi \det A_0} \sum_{\tilde{x}_i \in \Omega_0} |A_i^{-1}(h(\tilde{x}_i) - w(\tilde{x}_i))|^2.\end{aligned}$$

Note that $\pi \det A_0$ is just the area of the region Ω_0, so we obtain (4.5).

Oriented matching deviation and Frobenius norm

Define the "size" of the linear deformation $A \in GL(2)$ the quantity

$$\|A\|^2 = \frac{1}{\pi} \int_{|x|<1} x^\top A^\top A x \, dx.$$

This is the average of the norm of the vector Ax as x is moved along the unit circle. We have

$$\|A\|^2 = \frac{1}{\pi} \operatorname{tr}\left[A^\top A \int_{|x|<1} x x^\top \, dx \right] = \frac{1}{4} \operatorname{tr}[A^\top A].$$

so this is just the Frobenius norm of A (up to a scale factor). Now consider the affine deformation $Ax + T$. We define analogously

$$\begin{aligned}\pi\|(A,T)\|^2 &= \int_{|x|<1} |Ax+T|^2\,dx \\ &= \int_{|x|<1} x^\top A^\top Ax\,dx + 2\int_{|x|<1} T^\top Ax\,dx + \int_{|x|<1} T^\top T\,dx.\end{aligned}$$

So the "Frobenius norm" of an affine deformation is

$$\boxed{\|(A,T)\|^2 = \frac{1}{\pi}\int_{|x|<1} |Ax+T|^2\,dx = \frac{1}{4}\operatorname{tr}[A^\top A] + |T|^2.}$$

This also justifies (4.6) because

$$\begin{aligned}\operatorname{dev}(w, \mathbf{1}, \tilde\Omega_i) &= \frac{1}{\pi}\int_{\{x:|x|<1\}} |\tilde A_i^{-1}(w_i(x) - \tilde w_i(x))|^2\,dx \\ &= \frac{1}{\pi}\int_{\{x:|x|<1\}} |(\tilde A_i^{-1}A_i - \mathbf{1})x + \tilde A_i^{-1}(T_i - \tilde T_i)|^2\,dx\end{aligned}$$

Unoriented matching deviation

Lemma 4. *Let A be a square matrix and Q a rotation of the same dimension and let $U\Lambda V^\top = A$ be the SVD of A. Then the rotation Q which minimizes the quantity* $\operatorname{tr}[QA]$ *is $UV^\top$ and the minimum is* $\operatorname{tr}[\Lambda]$.

Proof. Let $V\Lambda U^\top = A$ be the SVD decomposition of matrix A. We have $\operatorname{tr}[QA] = \operatorname{tr}[L\Lambda]$ where Λ is a diagonal matrix with non-negative entries and $L = U^\top QV$ is a rotation matrix. The trace is equal to $\sum_i L_{ii}\lambda_i$ where $0 \le L_{ii} \le 1$ and $L_{ii} = 1$ for all i if, and only if, L is the identity. So the optimal value of Q is $Q = UV^\top$.

Since the Frobenius norm is rotationally invariant, (4.7) can be written as

$$\|\tilde R^\top \tilde A_i^{-1}A_iR - \mathbf{1}\|_F^2 = \|\tilde A_i^{-1}\tilde A_i\|_F^2 - 2\operatorname{tr}[Q\tilde A_i^{-1}A_i] + 2, \qquad Q = R\tilde R_i^\top.$$

Minimizing this expression with respect to Q is equivalent to maximizing the term $\operatorname{tr}[Q\tilde A_i^{-1}A_iR]$. Let $V\Lambda U^\top = \tilde A_i^{-1}\tilde A_i$ be the SVD of $\tilde A_i^{-1}A_i$; Lemma 4 shows that the maximum is $\operatorname{tr}[\Lambda]$ (obtained for $Q = UV^\top$), yielding (4.8).

4.B Frame alignment algorithms

In this section we derive algorithms to minimize (4.2) in the cases of interest. The purpose of the following algorithm is to align a set of points $x_1^{(1)}, \dots, x_1^{(K)}$ to a set of points $x_2^{(1)}, \dots, x_2^{(K)}$ up to either an affine, rigid, or similarity motion.

Alignment by an affine motion

Let $x_2 = Ax_1 + T$ for an affine motion (A, T). We can transform this equation as

$$x_2 = Ax_1 + T = Bx = (x^\top \otimes I_{2\times 2}) \operatorname{vec} B, \qquad x = \begin{bmatrix} x_1 \\ 1 \end{bmatrix}, \quad B = \begin{bmatrix} A & T \end{bmatrix}$$

where $\otimes$ is the Kronecker product and vec is the stacking operator. We obtain one of these equations for each of the points $x_1^{(k)}$, $k = 1, \ldots, K$ to be aligned and solve them in the least-squares sense for the unknown B.

Alignment by a rigid motion

We give first a closed-form sub-optimal algorithm. This algorithm is the equivalent as the one proposed in [ZS97], but our development is straightforward.

Let $x_2 = Rx_1 + T$ be a rigid motion (R, T) and assume for the moment that the points are three dimensional. Let $R = \exp(\theta\widehat{r})$ where r, $|r| = 1$ is the axis of rotation, $\widehat{r}$ is the hat operator [MSK03], and $\theta > 0$ is the rotation angle. We use Rodrigues' formula [MSK03] $R = I + \sin\theta\widehat{r} + (1 - \cos\theta)\widehat{r}^2$ to get

$$\begin{aligned} x_2 &= Rx_1 + T = x_1 + \sin\theta\, \widehat{r}x_1 + (1 - \cos\theta)\, \widehat{r}^2 x_1 + T, \\ x_1 &= R^{-1}(x_2 - T) = x_2 - T - \sin\theta\, \widehat{r}(x_2 - T) + (1 - \cos\theta)\, \widehat{r}^2(x_2 - T). \end{aligned}$$

Adding the previous equations, collecting $\sin\theta\,\widehat{r}$, and using the trigonometric identity $\tan(\theta/2) = (1 - \cos\theta)/\sin\theta$ we obtain

$$\sin\theta\, \widehat{r}\left(x_1 - x_2 + T + \tan\frac{\theta}{2}\, \widehat{r}(x_1 + x_2 - T)\right) = 0.$$

It is easy to check that this condition is equivalent to $x_2 = Rx_1 + T$ for $|\theta| < \pi$. A sufficient condition is

$$x_1 - x_2 + T + \tan\frac{\theta}{2}\, \widehat{r}(x_1 + x_2 - T) = 0.$$

which can be rewritten as

$$x_1 - x_2 + \tan\frac{\theta}{2}\, \widehat{r}(x_1 + x_2) + z = 0, \qquad z = T - \tan\frac{\theta}{2}\, \widehat{r}T.$$

Since, no matter what r is, z spans all $\mathbb{R}^3$ as T varies, we can equivalently solve this equation linear in the unknowns $\tan(\theta/2)r$ and z in order to estimate the rigid transformation. As in the previous section, one obtains one of such equations for each of the points $x_1^{(k)}$, $k = 1, \ldots, K$ to be aligned and finds the solution in the least-squares sense.

If there is noise in the model, i.e. if $x_2 = Rx_1 + T + n$, we get the condition

$$x_1 - x_2 + \tan\frac{\theta}{2}\,\widehat{r}(x_1 + x_2) + z + \tan\frac{\theta}{2}\,\widehat{r}n = -n.$$

This means that for moderate rotations (away from $\pm\pi$) minimizing the l^2 residual of this equation is almost equivalent to minimizing the norm of n itself. However if θ approaches $\pm\pi$, then the term $\tan(\theta/2)\widehat{r}n$ will dominate, biasing the estimate.

The formulas work for the 2-D case with little adaptation. In this case we assume that all the points lie on the X-Y plane and the rotation vector is aligned to the Z axis, obtaining

$$x_1 - x_2 - \tan\frac{\theta}{2}\begin{bmatrix} x_{2,1} + x_{2,2} \\ -x_{1,1} - x_{1,2} \end{bmatrix} + z = 0.$$

Finally, the estimate can be refined by the iterative algorithm given in the next section (where one fixes the scale s to the constant 1).

Alignment by a similarity

There is no closed-form algorithm for this case. Instead, we estimate iteratively the translation T and the linear mapping sR. While the solution to the first problem is obvious, for the second consider the following equation:

$$\min_{s,R} \sum_k (x_2^{(k)} - sRx_1^{(k)})^\top (x_2^{(k)} - sRx_1^{(k)})$$
$$= \min_{s,R} \sum_k |x_2^{(k)}|^2 - 2s \sum_k {x_2^{(k)}}^\top Rx_1^{(k)} + s^2 \sum_k |x_1^{(k)}|^2. \quad (4.10)$$

Rewrite the cost function al as $c - 2bs + as^2$. The optimal value for s given a certain R is $s^* = b/a$ and the optimal value of the cost function is $a + c - 2b^2/a$. Note that only the term b is a function of R, while neither a nor c depend on it. As a consequence, the optimal value of R is obtained by solving the problem

$$\max_R b = \max_R \sum_k {x_2^{(k)}}^\top Rx_1^{(k)} = \max_R \sum_k \operatorname{tr}\left(Rx_1^{(k)} {x_2^{(k)}}^\top\right)$$

Thus we are simply maximizing the correlation of the rotated point $Rx_1^{(k)}$ and the target points $x_2^{(k)}$. By taking the derivative of the trace w.r.t. the rotation angle θ, we immediately find that the optimal angle is $\theta^* = \operatorname{atan}(w_2/w_1)$ where

$$w_1 = \sum_k |x_2^{(k)}||x_1^{(k)}| \cos\theta^{(k)}, \qquad w_2 = \sum_k |x_2^{(k)}||x_1^{(k)}| \sin\theta^{(k)}$$

where $\theta^{(k)}$ is the angle from vector $x_1^{(k)}$ to vector $x_2^{(k)}$.

Thus, in order to estimate R and s, we can first solve for the optimal rotation R^*, and then solve for the scale, which is obtained as

$$s^* = \frac{b}{a} = \frac{\sum_k x_2^{(k),\top} R^* x_1^{(k)}}{\sum_k |x_1^{(k)}|^2}.$$

The convergence of the alternating optimization can be greatly improved by removing the mean from $x_1^{(k)}$, $k = 1, \ldots, K$ as a pre-processing step.

CHAPTER 5

Ensemble Invariance and Joint Data Alignment

A building block in the conceptual construction of invariant and maximally discriminative features is the process of canonization (Section 1.1). One obstacle in the practical implementation of this idea is to constructively define the set of canonized features, or canonical set. For simple cases, such as affine invariant patches [MTS04], criteria such as imposing the symmetry of the structure tensor might be sufficient.[1] However, there is no universal way of specifying a canonical set and doing so may be difficult in practice. In particular, if the transformations and the data are complex, the geometry of the canonical set could be impossible to define by hand-crafted rules. Even when this is possible, one might ask which of the many possible canonical sets is the best in consideration of additional desirable properties.

In this chapter we study methods that can be used to *learn* canonizations from data in an unsupervised setting. We reformulate this problem as the one of *joint alignment*: Given a large collection of patterns (e.g., image patches), we deform them until they are aligned. The resulting set of aligned data constitutes the canonical set, which is described non-parametrically by the aligned prototypes themselves. The advantage of this approach is threefold: (i) we introduce a single construction to align a large variety of different data, (ii) the canonical set may have arbitrarily complex geometry as it is described non-parametrically, and (iii) it is possible to encourage the algorithm to find canonizations that exhibit useful properties.

5.1 Models of joint data alignment

Joint alignment is the process of removing from a collection of data the effect of undesired and irrelevant transformations. It serves both in discriminative modeling, to simplify the extraction of useful informations from the data, and in generative modeling, to reduce the data variability and make it easier capture it. For example, in order to *recognize* images of faces we are not interested in the face pose, and alignment can be used to eliminate pose from the problem.

[1] I.e., one defines as canonical images that have symmetric structure tensor

Similarly, if the goal is to *encode* images of faces, it is more convenient to factor out pose first, and then encode only the residual variability.

But what does it means to jointly align data? A possible conceptual model is the following: We assume that the observed data has been produced by transforming data originally aligned, and the goal is to recover the aligned data from the transformed one. This idea can be translated directly into a generative model with three components: The aligned data, the transformations, and the transformed data. Then the problem is to estimate the two unknown quantities (aligned data and transformations) from the observed one (transformed data). This is an inverse problem and is solved by making appropriate regularity assumptions on the underlying statistical model. We call this approach *alignment by fitting a generative model*, or, simply, *generative* approach.

In the context of joint alignment, the generative approach has been explored by *Transformed Component Analysis* (TCA) [FJ03, FJ99]. TCA models the aligned data as a Gaussian distribution, the samples of which are randomly transformed to obtain the observed data. TCA can be regarded as a version of Probabilistic Principal Component Analysis (PPCA) [TB99] augmented with data transformations. While conceptually straightforward, the generative approach has a number of issues. First, one must make assumptions about the statistics of the data that may be not well satisfied. Most importantly, the resulting optimization problems are complex and slow, making it hard to deal with large data. While there are known speedups [KJF07], these are limited to handle only simple transformations such as translations.

Since the goal of alignment is to simplify the data for further processing, estimating a full generative model may be an overkill. Instead, we may directly search for a transformed version of the data which is simpler, in the sense that it minimizes an appropriate measure of complexity. We call this approach *alignment by complexity reduction*. This idea has been proposed and popularized in the context of joint alignment by *image congealing* (IC) [Lea06]. In IC the data complexity is intended in the Shannon's (ensemble) sense as entropy and the algorithm attempts to transform the data in order to minimize this figure. IC is significantly simpler and more efficient than TCA, and enables aligning very large data with relatively complex transformations. An additional advantage is that IC does not make any particular assumption about the data statistics (albeit it requires regularity assumptions on the transformations).

In this chapter we study the complexity-reduction approach to joint alignment. We identify and address problems which were left open by previous works, providing a better understanding of its theoretical foundations and enabling new extensions.

First, we show that simplifying the data alone may remove not only the ir-

relevant variability, but also the useful information that one wants to preserve (Section 5.2). Take the example of aligning a collection of digital images up to a scaling of the domain. While in the continuous scalings are invertible transformations, this property is lost once we consider the discrete nature of digital images.[2] An algorithm which is based on minimizing entropy will try to use these "lossy" effects to improperly reduce the variability of the data, possibly converging to a degenerate solution. In IC this is implicitly avoided by imposing regularity on the transformations, which must be not "too lossy". Here we explicitly address the issue by imposing that the aligned data can be used to reconstruct the original data up to deformation and a small residual distortion. This casts IC in the framework of lossy compression (or lossy complexity reduction). So, as TCA is related to PPCA, IC is related to encoding methods such as Vector Quantization (VQ), Entropy Constrained Vector Quantization (ECVQ) and Rate Distortion (RD).

Second, we discuss which measures of complexity are suitable for joint alignment (Section 5.2.3). It is tempting to use off-the-shelf measures such as entropy and mutual information, and to regard joint alignment almost as a standard compression problem. However, complexity in the Shannon's sense, as captured by VQ, ECVQ and RD, does not reflect any structural property of the data. Roughly speaking, the goal of VQ, ECVQ and RD is to approximate the data with the smallest number of *codewords* (prototypes) possible, as complexity is simply the number of such codewords. The structure of the codewords themselves is irrelevant. So, while multiple data points may be aligned to a given codeword, and hence one another, there is *a-priori* no reason why codewords should be globally aligned, and so the data.

For the purpose of joint alignment, we need to link complexity to the regularity of the codebook. For instance, IC complexity is the sum of the entropies of all image pixels, regarded as statistically independent. While this complexity was introduced as an approximation to the actual entropy of the dataset, we note that it serves for the purpose of enforcing global alignment better than entropy alone (see Section 5.3). Once this is understood, we can devise other types of complexities that are be suitable for alignment. In particular, complexity can be used to enforce useful properties of the aligned dataset. As an example, we introduce a complexity term that can be used to encourage the aligned data to span a linear subspace of small dimension (Section 5.4).

Third, we discuss a fundamental issue that arise when the aligned data is continuous and one attempts to minimize its complexity. We show that the

[2]Due to aliasing, the cascade of interpolation-scaling-sampling required to approximate a continuous transformation in the discrete domain yields in general a *non* invertible transformation.

mere fact of measuring the complexity of a continuous variable introduces an additional distortion term that, if ignored, leads in some cases to degenerate solutions (Section 5.4.1).

We also develop efficient algorithms for all the cases that we consider, and we demonstrate the ability of such algorithms to align large datasets. We conclude the chapter with an application to the alignment of a large collection of natural image patches (Section 5.6). This can be used to address the difficult problem of sparse encoding of natural images under transformations that has recently interested several authors.

5.2 Joint alignment as compression

In IC [Lea06] alignment is obtained by searching for a transformed version of the data which minimizes entropy. Unfortunately, if the transformations are not bijective (as in most applications) this encourages the algorithm to annihilate *all* the data variability by preferring lossy transformations. For instance, in the problem of aligning NIST digits presented in Section 5.3.4 and later in the chapter, IC naturally shrinks digits to a single pixel, reducing the entropy to zero. In order to avoid this degenerate solutions, [Lea06] penalizes lossy transformations by an hand-crafted regularization term. Once this problem is understood, however, it is more natural to change the formulation to reflect the fact that simplification should not loose information besides the nuisance transformations. Formally, joint alignment becomes akin to a lossy compression problem, where one trades off simplicity of the aligned data and accuracy of the encoding.

In Section 5.2.1 we recall a few ideas from the theory of lossy compression, and in Section 5.2.2 and 5.2.3 we discuss how they can be adapted to the problem of joint alignment. The formulation is then compared to the one of IC in Section 5.2.4.

5.2.1 Lossy compression

Let $x \in \mathbb{R}^n$ be a continuous random variable (*datum*). In lossy compression we search for another (continuous or discrete) r.v. $y \in \mathbb{R}^n$ (*code*) with conditional density $p(y|x)$ that represents x concisely and accurately. Formally, we look for $p(y|x)$ that minimizes

$$E(y) = D(x, y) + \lambda \mathcal{C}(x, y). \tag{5.1}$$

The *distortion* $D(x, y)$ reflects the average error of the approximation, the *complexity* $\mathcal{C}(x, y)$ is average number of symbols required to encode y, and the parameter $\lambda \geq 0$ trades off the two terms. This general idea has been pursued in

various forms in the literature. In VQ [LBG80], y is restricted to $K < \infty$ *codewords*, all encoded with the same number of symbols. Thus $C(x, y) \propto \log K$ and VQ trades off the number of codewords for the average distance of the code y to the datum x. Entropy-Constrained Vector Quantization (ECVQ) [CLG89, BK93] generalizes this idea and represents codewords with a variable number of symbols. The optimal variable-length code uses an average number symbols equal to the entropy $H(y)$, so that $\mathcal{C}(x, y) = H(y)$. There also exist relaxations of VQ and ECVQ based on deterministic annealing (DA) [RM93] that use a regularized complexity term which depends jointly on x and y. Rate-Distortion (RD) [Sha48] is a further generalization to the problem of compressing long sequences of data. Shannon [Sha48] reduced this problem to the one of encoding a single instance x by a code y (which in this case can be a continuous r.v.) and complexity measure $\mathcal{C}(x, y) = I(x, y)$. This complexity gives also the *rate* (i.e. the average number of symbols per component of the sequence).

Notice that VQ and ECVQ are useful not only for compression, but also for clustering. Next, we look for choices of the distortion and complexity measures that are useful for the problem of joint alignment.

5.2.2 Distortion for alignment

The basic idea to reduce the joint alignment problem to a (lossy) compression problem is to search for a code y that represents the data x up to the action of the irrelevant transformations G and is "as simple as possible". Formally, we enable the code $y \in \mathbb{R}^m$ to represent the data $x \in \mathbb{R}^n$ up to the effect of a family G of transformations $g : \mathbb{R}^m \to \mathbb{R}^n$. This is achieved as follows. Starting from an arbitrary point-wise distortion measure $d_0 : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+$, we consider the *expected invariant distortion*

$$D(x, y) = E[d(x, y)], \qquad d(x, y) = \inf_{g \in G} d_0(x, gy). \tag{5.2}$$

Notice that the transformations $g \in G$ need not to have a special structure and, in particular, they are not required to form a group. There are several choices for the base distortion measure d_0; in the following we take the Euclidean distance $d_0(x, gy) = \|x - gy\|^2$.

5.2.3 Complexity for alignment

The complexity term $\mathcal{C}(x, y)$ of VQ, ECVQ, and RD essentially reflects the average number of symbols needed to index the code y (which in this case is a discrete variable). Unfortunately, this is not well suited for alignment because it is insensitive to the actual *values* of the codewords and cannot capture any of their

"structural" regularities. In particular, there is no natural way of encouraging the mutual alignment of the codewords.

In order to do this, we look for indexing mechanisms (and corresponding complexity measures) that are efficient precisely when y exhibits the desired regularity. For example, in IC the code y is a (random) binary image whose complexity $\mathcal{C}(x, y)$ is defined as the total entropy of its pixels, regarded as independent random variables. This gives the number of symbols required to index the codewords y if the design of the indexing mechanism *disregards the dependencies* among pixels. Clearly, this scheme is efficient if the pixels of y are independent. Since in practice pixels of natural images are highly correlated, this condition can be approached only if pixels are approximatively constant (across different samples of y). Thus minimizing this complexity encourages the mutual alignment of the codewords.

We study two complexity costs that have these characteristics. In Section 5.3 we consider a cost similar to standard IC, based on disregarding all the dependencies among pixels. Then in Section 5.4 we propose an alternative complexity term based on measuring the (properly normalized) entropy of a Gaussian distribution fitting the code y. This disregards all but the linear dependencies among pixels and encourages the code y to span a low dimensional subspace of $\mathbb{R}^m$.

5.2.4 Formal comparison with IC formulation

Our formulation of joint alignment reduces to the minimization of (5.1). It is interesting to compare this equation to IC formulation. There the code $y = gx$ is directly a function of the data point x and the random transformation g. The distortion term $\mathcal{D}(x, y)$ is replaced by an hand-crafted regularization $R(g)$ that penalizes transformations g which are too lossy.

5.3 Naive complexity

In this section we introduce a complexity term $\mathcal{C}(x, y)$ that, similarly to IC, disregards the dependencies among the components of the code $y \in \mathbb{R}^m$. We call this term *naive* as, similarly to naive Bayes, assumes independence of the data components. Differently from naive Bayes, in which independence trades off bias and over-fitting, here the independence assumption is exactly what makes the complexity term suitable for alignment.

Let $e_1, \dots, e_m$ be the vectors of the canonical basis of $\mathbb{R}^m$. We define

$$\mathcal{C}(x, y) = \frac{1}{K} \sum_{k=1}^{K} \sum_{j=1}^{m} h(\langle e_j, y_k \rangle)) = \sum_{j=1}^{m} E[-\log p_j(\langle e_j, y_k \rangle)]$$

where $h(\langle e_j, y_k \rangle)$ denotes the differential (or discrete) entropy of the projection $\langle e_j, y_k \rangle$. Concretely, given a data ensemble $x_1, \dots, x_K \in \mathcal{X}$, the density functions $p_j(\phi_j(y))$ are estimated from the samples $\langle e_j, y_1 \rangle, \dots, \langle e_j, y_K \rangle$ by a Parzen window estimator [DHS01] with Gaussian kernel $k_\sigma(y)$ of standard deviation σ (continuous case[3]), i.e.

$$p_j(\langle e_j, y \rangle) = \frac{1}{K} \sum_{k=1}^{K} k_\sigma(\langle e_j, y - y_k \rangle).$$

Thus

$$\mathcal{C}(x, y) = -\frac{1}{K} \sum_{j=1}^{m} \sum_{i=1}^{K} \log p_j(\langle e_j, y_k \rangle).$$

Let $G = \{g_\alpha : \alpha \in A\}$ be a parametric family of transformations. Given samples $\{x_1, \dots, x_K\}$ of the random variable x, the problem is then to find transformations $\{\alpha_1, \dots, \alpha_K\}$ and codes $\{y_1, \dots, y_K\}$ that minimize

$$E(\{\alpha_k, y_k\}) = \frac{1}{K} \sum_{k=1}^{K} \|x_k - g_{\alpha_k} y_k\|^2 - \lambda \frac{1}{K} \sum_{j=1}^{m} \sum_{i=1}^{K} \log p_j(\langle e_j, y_k \rangle). \tag{5.3}$$

Remark 9 (Continuous vs discrete entropy). It should be noted that in VQ and ECVQ the entropy $H(y)$ of the code is discrete even if x is a continuous random variable, as the number of codewords is finite. Here instead we consider either differential entropies or discrete entropies, depending on the nature of the data x. This fact introduces a subtle issue, because the meaning of differential entropy is itself *relative* to a quantization error ϵ which we did not account for yet. To understand the effect of this problem, notice that the differential entropy may be made arbitrarily small or large simply by scaling the random variable y. This problem is further discussed in Section 5.4.1.

[3]By using the Parzen window estimator we guarantee that the differential entropy of the distributions p_j is always lower bounded by the entropy of the kernel k_σ. This prevents the differential entropy to have arbitrary small negative values, and is similar to the regularization adopted in Section 5.4.

5.3.1 Application to the alignment of images

The main application of our method is the joint alignment of large collection of images. In this section we specialize (5.3) to solve this problem.

We start by specifying the nature of the data, of the codes and of the transformations. Images are functions defined on a discrete lattice, as in Chapter 3. To keep the notation consistent with the general case, however, in this chapter images are denoted as functions $x : (u, v) \mapsto \mathbb{R}_+$. Formally, the data point $x(u, v) \in \mathbb{R}$ is a (random) discrete image defined on the two dimensional lattice $\Omega = \{-r, -r + 1, \dots, r\}^2$ where r is a non-negative integer. Similarly, the code $y(u, v)$ is a (random) discrete image defined on a lattice $\Omega' = \{-r', \dots, r'\}$. The image x will also be identified with a matrix in $\mathbb{R}^{(2r+1)\times(2r+1)}$ or a vector in $\mathbb{R}^{(2r+1)^2}$, and similarly for the image y. Our goal is to remove from the random image x transformations $g : \mathbb{R}^2 \to \mathbb{R}^2$ of the real plane. For simplicity, we consider only affine transformations g_α where $\alpha = \begin{bmatrix} A & T \end{bmatrix}$, $T \in \mathbb{R}^2$, $A \in GL(2)$ and $g_\alpha(u, v) = A^{-1} \cdot (u, v) - A^{-1} \cdot T$ (notice that α parametrizes the inverse warp, as this simplifies the equations below). Applying the transformation g_α to an image $y(u', v')$ yields the image $(g_\alpha y)(u, v) = y(g_\alpha^{-1}(u, v))$, where the value $y(u', v')$ at the fractional point $(u', v') = g_\alpha^{-1}(u, v)$ is obtained by extending y to the real plane by bilinear interpolation and zero padding. Notice also that $(g_\alpha y)(u, v) = M(u, v; \alpha)y$ where $M(u, v; \alpha)$ is a linear operator independent of the particular image y. Since y is finite-dimensional, $M(u, v; \alpha)$ is just a row vector of mixing coefficients. Similarly, we will also use the notation

$$g_{\alpha_k} y_k = M(\alpha_k) y_k$$

where $g_{\alpha_k} y \in \mathbb{R}^{(2r+1)\times(2r+1)}$ and $M(\alpha_k)$ is a matrix of mixing coefficients determined by the transformation α_k and and the interpolation method in use.

The complexity of the aligned ensemble $\{y_1, \dots, y_K\}$ is computed as in Section 5.3. For each pixel $(u, v) \in \Omega'$ the density $p(y(u', v'))$, is estimated non parametrically from the samples $\{y_1(u, v), \dots, y_K(u, v)\}$ by the Parzen window estimator. The complexity of a pixel is thus

$$h(y(u, v)) = -\frac{1}{K} \sum_{k=1}^{K} \log p(y_k(u, v)).$$

Finally the overall cost function is obtained by summing over all pixels and averaging over all images:

$$E(\{\alpha_k, y_k\}) = \frac{1}{K} \sum_{i=1}^{K} \|x_k - g_{\alpha_k} y_k\|^2 - \lambda \frac{1}{K} \sum_{i=1}^{K} \sum_{(u,v)\in\Omega'} \log p(y_k(u, v))). \qquad (5.4)$$

Dealing with image boundaries. According to (5.4), only the portion of the code y which is mapped back within the bounding box of the image x is actually constrained by the distortion term $\|x - g_\alpha y\|^2$ (see Fig. 5.6). The other portion of the code y is determined uniquely by minimizing the complexity $\mathcal{C}(x, y)$. In some cases this introduces a discontinuity in the estimated code y which makes the optimization of (5.4) tricky. This could be alleviated for example by delimiting the domain of x by a Gaussian window rather than by a bounding box. If, however, the image x can be extended beyond its bounding box in a natural way then that information can be used to "fill the hole". We will get back to this issue in Section 5.5.

5.3.2 Optimization by coordinated descent

The transformation parameters α_k in (5.4) can be optimized by coordinate descent similarly to [FJ00]:

1: Estimate the probabilities $p(y(u, v))$, $(u, v) \in \Omega'$ from the samples $\{y_k(u, v) : k = 1, \dots, K\}$
2: For each datum $k = 1, \dots, K$ and for each component α_{jk} of the parameter vector α_k, try a few values of α_{jk}. For each value re-compute the cost function (5.4) and keep the best value of α_{jk}.
3: Repeat, refining the sampling step of the parameters.

This algorithm is appropriate if the dimensionality of the parameter vector α is reasonably small. Here we consider affine transformation, so that α is six-dimensional.

We are left with the problem of estimating the codes y_k. As a first order approximation (the final result will be refined by Gauss-Newton as explained in the next section), we bypass this problem and we simply set $y_k = g_{\alpha_k}^{-1} x_k$, exploiting the fact that the affine transformations g_α, as maps from $\mathbb{R}^2$ to $\mathbb{R}^2$, are invertible. Notice however that g_{α_k} is *not* invertible as an image transformation due to the approximation introduced by the discrete nature of the images. Eventually the distortion $\mathcal{D}(x, y)$ writes

$$\mathcal{D}(x, y) = \frac{1}{K} \sum_{k=1}^{K} \|(I - g_{\alpha_k} g_{\alpha_k}^{-1}) x_k\|^2$$

which is an empirical measure of the "non invertibility" of the image transformations g_{α_k}. It is interesting to compare this term with the *ad-hoc* regularization used by Learned-Miller in the original IC formulation [Lea06] (Section 5.2.4).

5.3.3 Optimization by Gauss-Newton

Compared with the standard formulation of IC, here we need to estimate both the transformation parameters α_k *and* the codes y_k. As promised in the previous section, this can be done by Gauss-Newton (GN).

Applying Gauss-Newton requires to take derivatives with respect to the pixel values $y_k(u,v)$. We exploit the fact that the variables $y_k(u,v)$ are continuous (notice that in IC [Lea06] these are discrete for the application to image alignment).

We still process a single image per time, reiterating several times across the whole ensemble $\{x_1,\dots,x_K\}$. For a given image x_k we update the warp parameters α_k and the codeword y_k simultaneously. We exploit the fact that, as the number K of images is usually large, the density $p(y_k(u,v))$ does not change significantly when only one of the codeword y_k is changed. Therefore $p(y_k(u,v))$ can be assumed constant in the computation of the gradient and the Hessian of the cost function (5.4). The gradient is given by

$$\frac{\partial E}{\partial \alpha_k^\top} = \sum_{(u,v)\in\Omega} 2\Delta_k(u,v)(\nabla g_{\alpha_k} y_k)(u,v)\frac{\partial g}{\partial \alpha_k^\top}(u,v),$$

$$\frac{\partial E}{\partial y(u',v')} = \sum_{(u,v)\in\Omega} 2\Delta_k(u,v)(M(u,v;\alpha_k)\delta_{uv}) - \sum_{(u',v')\in\Omega} \frac{\dot{p}(y_k(u',v'))}{p(y_k(u',v')}$$

where $\Delta_k = g_{\alpha_k} y_k - x_k$ is the reconstruction residual, $M(u,v;\alpha_k)$ is the linear map introduced in Section 5.4.2 and δ_{uv} is the vector of all zeros and a one in position (u,v).

The approximated Hessian of the cost function (5.4) can be obtained as follows. First, we use the Gauss-Newton approximation for the derivative w.r.t. the transformation parameters α_k

$$\frac{\partial^2 E}{\partial \alpha_k \partial \alpha_k^\top} \approx \sum_{(u,v)\in\Omega} 2\frac{\partial g^\top}{\partial \alpha_k}(u,v)\nabla^\top(g_{\alpha_k} y_k)(u,v)\nabla(g_{\alpha_k} y_k)(u,v)\frac{\partial g}{\partial \alpha_k^\top}(u,v)$$

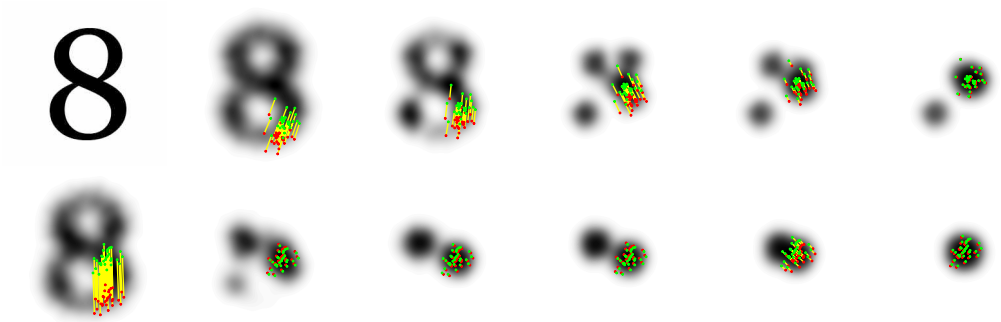


Figure 5.1: **Toy example. Top left.** We distort the patterns by applying translations drawn uniformly from the 8-shaped region (the center corresponds to the null translation). **Top.** We show the gradient based algorithm while it gradually aligns the patterns by reducing the complexity of the alignment y. Dark areas correspond to high values of the density of the alignment; we also superimpose the trajectory of one of the patterns. Unfortunately the gradient based algorithm, being a local technique, gets trapped in two local modes (the modes can however be fused in a post-processing stage). **Bottom.** The basic algorithm completely eliminates the effect of the nuisance transformations doing a better job of avoiding local minima. Although for this simple problem the basic search is more effective, on more difficult scenarios the extra complexity of the Gauss-Newton search pays off (see Section 5.3.4).

We then have

$$\frac{\partial^2 E}{\partial y_k(u',v')^2} = \sum_{(u,v)\in\Omega} 2(M(u,v;\alpha_k)\delta_{u'v'})^2 - \sum_{(u',v')\in\Omega} \frac{\ddot{p}(y_k(u',v')))p(y_k(u',v'))K - \dot{p}(y_k(u',v'))}{p(y_k(u',v'))^2}$$

$$\frac{\partial^2 E}{\partial y_k(u',v')\partial y_k(u'',v'')} = \sum_{(u,v)\in\Omega} 2(M(u,v;\alpha_k)\delta_{u'v'})(M(u,v;\alpha_k)\delta_{u''v''})$$

$$\begin{aligned}\frac{\partial^2 E}{\partial y_k(u',v')\partial \alpha_k^\top} &= \sum_{(u,v)\in\Omega} 2(M(u,v;\alpha_k)\delta_{u'v'})\nabla(g_{\alpha_k}y_k)(u,v)\frac{\partial g}{\partial \alpha_k^\top}(u,v) \\ &+ \sum_{(u,v)\in\Omega} 2\Delta_k(u,v)M(u,v;\alpha_k)\begin{bmatrix}D_1\delta_{u'v'} & D_2\delta_{u'v'}\end{bmatrix}\frac{\partial g}{\partial \alpha_k^\top}(u,v)\end{aligned}$$

where D_1 is the discrete linear operator used to compute the derivative of $y_k(u,v)$ along its first dimension u and D_2 the analogous operator for the second dimension v. The second term of the last equation gives a very small contribution and can be dropped.

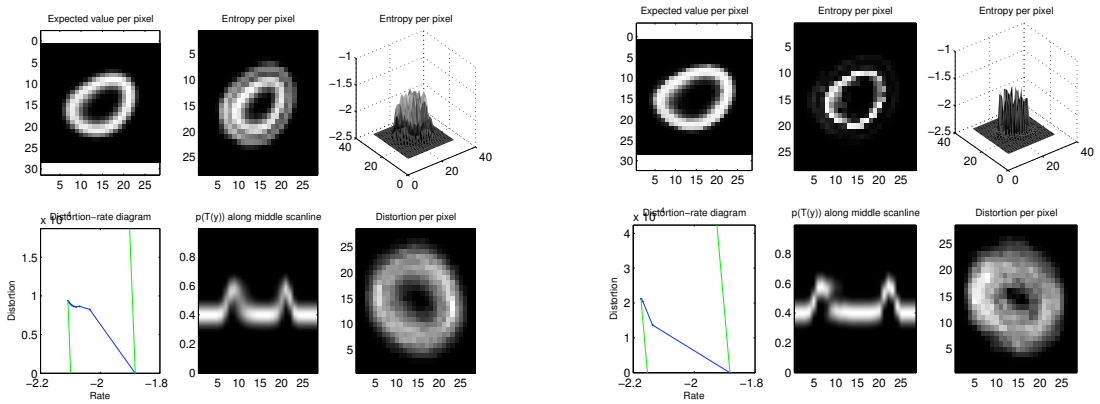


Figure 5.2: *Basic vs GN image alignment algorithms.* **Left.** We show the results of applying the basic image alignment algorithm of Section 5.3.2. The patterns are zeroes from the NIST Special Database 19. We show in writing order: The expected value $E[y(u,v)]$; the per-pixel entropy $h(y(u,v))$ (it can be negative as it is differential); a 3-D plot of the same function $h(y(u,v))$; the distortion-complexity diagram as the algorithm minimizes the function $D + \lambda R$ (in green we show some lines of constant cost); the density $p(y(u,v) = l)$ as (u,v) varies along the middle scan-line; and the per-pixel distortion $E[(x(u,v) - (g_\alpha y)(u,v))^2]$. **Right.** We demonstrate the GN algorithm of Section 5.3.3. The algorithm achieves a significantly better solution in term of the cost function (5.4). Moreover GN converges in only two sweeps of the dataset, while the basic algorithm after 10 sweeps is still slowly moving. This is due to the fact that GN selects both the best search direction and step size, resulting in a more efficient search strategy.

The equations are all straightforward and result in la linear system

$$\delta\theta^\top \left(\frac{\partial^2 E}{\partial\theta\partial\theta^\top} \right) = -\frac{\partial E}{\partial\theta^\top}$$

where the vector $\theta = (\alpha_k, y_k)$ has size in the order of the number of pixels of the codeword y_k. While this system is large, it is also extremely sparse an can be solved rather efficiently by standard methods [GL96].

5.3.4 Experiments

The first experiment (Fig.5.1) is a toy problem illustrating our method. We collect K patterns x_i, $i = 1, \ldots, K$ which are arrays of M 2D points $x_i = (x_{1i}, \ldots, x_{Mi})$. Such points are generated by drawing M i.i.d. samples from a 2-D Gaussian

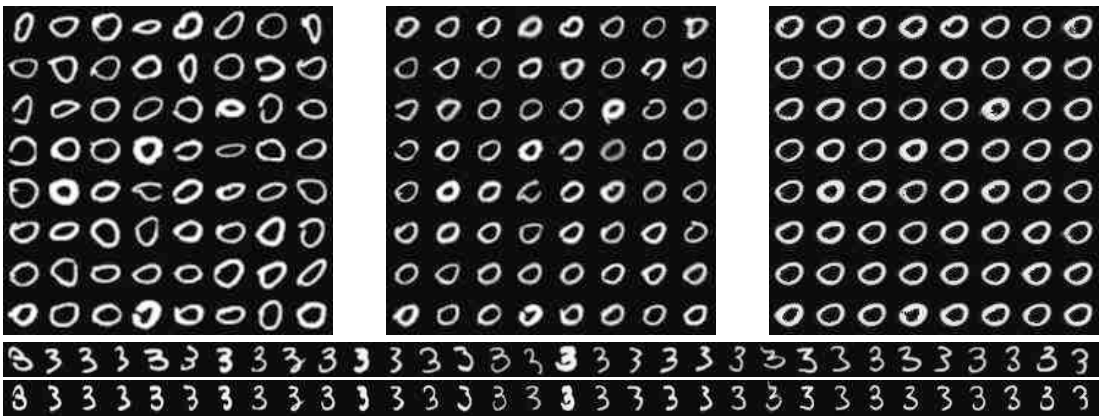
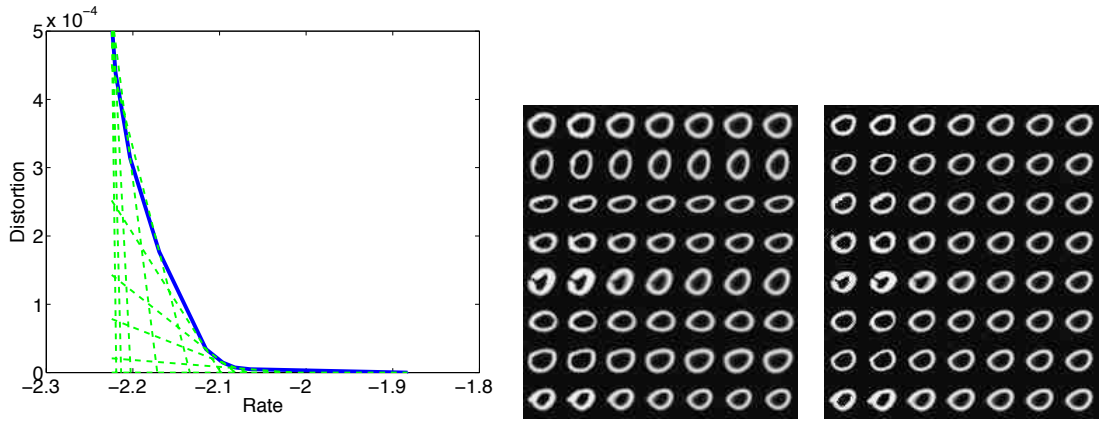


Figure 5.3: *Aligned patterns.* **Left.** A few patterns from NIST Special Database 19. **Middle.** Basic algorithm: Results are very similar to [Lea06], except that no regularization on the transformations is used. **Right.** GN algorithm: Patterns achieve a better alignment due to the more efficient search strategy; they also appear to be much more "regular" due to the noise cancellation effect discussed in Fig. 5.4. **Bottom.** More examples of patterns before and after GN alignment.



(a) Distortion-Complexity **(b)** Not aligned **(c)** Aligned

Figure 5.4: *Distortion-complexity balance.* We illustrate the effect of varying the parameter λ in (5.4). **(a)** Estimated distortion-complexity function $D(R)$. The green (dashed) lines have slope equal to λ and should be tangent to $D(R)$ (Section 5.3). **(b)** We show the alignment $T(w_i x)$ of eight patterns (rows) as λ is increased (columns). In order to reduce the entropy of the alignment, the algorithm "forgets" about specific details of each glyph. **(c)** The same as (b), but aligned.

distribution and adding a random translation $w_i \in \mathbb{R}^2$ to them. The distribution of the translations w_i is generic (in the example w_i is drawn uniformly from an 8-shaped region of the plane): This is not a problem as we do not need to make any particular assumptions on w besides that it is a translation. The distortion $d(x_i, y_i)$ is simply the sum of the Euclidean distances $\sum_{j=1}^{m} \|y_{ji} + w_i - x_{ji}\|^2$ between the patterns x_i and the transformed codes $w_i(y_i) = (y_{1i} + w_i, \dots, y_{mi} + w_i)$. The distribution $p(y_i)$ of the codes is assumed to factorize as $p(y_i) = \prod_{j=1} p(y_{ji})$ where the $p(y_{ji})$ are identical densities estimated by Parzen window from all the available samples $\{y_{ji}, j = 1, \dots, M,\ i = 1, \dots, K\}$.

In the second experiment (Fig. 5.2) we align hand-written digits extracted from the NIST Special Database 19. The results (Fig. 5.3) should be compared to the ones from [Lea06]: They are of analogous quality, but they were achieved without regularizing the class of admissible transformations. Despite this, we did not observe any of the aligned patterns to collapse. In Fig. 5.4 we show the effect of choosing different values of the parameter λ in the cost function (5.4). As λ is increased, the alignment complexity is reduced and the fidelity of the alignment is degraded. By an appropriate choice of λ, the alignment can be regarded as a "restoration" or "canonization" of the pattern which abstracts from details of the specific instance.

5.4 Structural complexity: The linear case

In this section we introduce a complexity term $\mathcal{C}(x, y)$ that characterizes the *linear dimensionality* (the number of dimensions of the linear subspace spanned by y) of the code $y \in \mathbb{R}^n$. We do so by constructing a description of y which is efficient when y spans a low-dimensional affine subspace of $\mathbb{R}^n$. To do this, first we approximate the density $p(y)$ of the code y with a Gaussian density $g(y)$, which captures the linear statistics of $p(y)$. Then, as if y had density $g(y)$, we use standard tools from rate-distortion theory to devise the optimal description of y and estimate its length (rate). The Gaussian density $g \in \mathcal{N}(\mu_g, \Sigma_g)$ which is closer to $p(y)$ in Kullback-Leibler (KL) divergence $\mathrm{kl}(p||g) = E_p[-\log g(y)] - h(p)$ is the one that matches the mean and the variance of y, i.e. $\mu_g = \mu_p = E_p[y]$, and $\Sigma_g = \Sigma_p = E_p[yy^\top] - \mu_p \mu_p^\top$. This Gaussian g yields an upper bound on the *rate* $R(\epsilon; p)$ (number of bits per symbol) required to describe y with some accuracy[4] ϵ: $R(\epsilon; p) \leq R(\epsilon; g)$. The rate-distortion function of a Gaussian source is known analytically [CT06], but in general its calculation requires computing the eigenvalues of Σ_g. If, however, we add to y a small Gaussian noise of variance

[4]The slack between $R(\epsilon; p)$ and $R(\epsilon; g)$ is irrelevant, as our goal is to characterize the *linear* dimensionality of the code y.

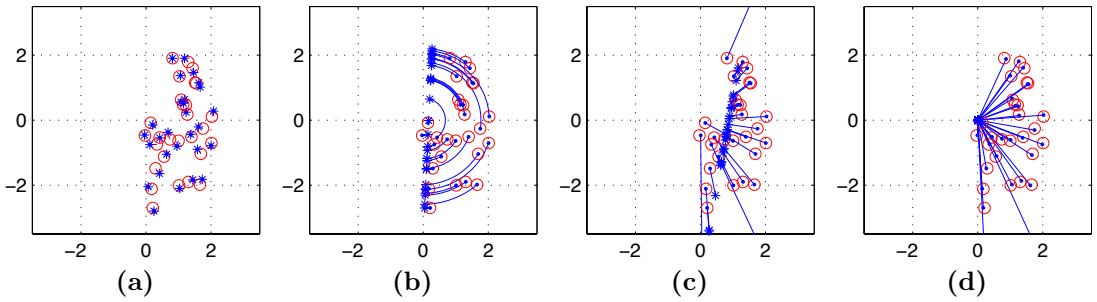


Figure 5.5: **Aligning 2-D points (see text).** (a) the data points $x_1, \dots, x_K$ (circles), the initial codes $y_1, \dots, y_K$ (stars — they are obtained by adding a small noise to the data) and reconstructions (dots — they coincide with the codes as initially $g_1 = \cdots = g_K = 1$); (b) removing the group G_1 of rotations from the data by mapping them to an affine subspace (line) — the curves show the trajectories mapping back the codes to the data; (c) removing the group G_2 of scalings from the data; (d) same as (c), except that the un-normalized complexity term (5.5) is used, which causes the solution to collapse on the origin.

ϵ^2 [YWS07], the formula is simply $R(\epsilon; g) = \frac{1}{2} \log \det \frac{\epsilon^2 I + \Sigma_g}{\epsilon^2}$ and we obtain

$$\mathcal{C}(x, y) = \frac{1}{2} \log \det \frac{\epsilon^2 I + \Sigma_p}{\epsilon^2} = \frac{1}{2} \log \det \left(I + \frac{\Sigma_p}{\epsilon^2} \right). \tag{5.5}$$

5.4.1 Degenerate solutions and normalization

It is easy to see that (5.5) decreases not only with the dimensionality of y, but also with its variance. Thus, if the transformations $g \in G$ enable reducing the variance of y without increasing the distortion of the reconstruction gy, then minimizing (5.5) yields a degenerate code (see also Fig. 5.5).

We remark that the same problem affects all similar formulations in which the code y is a continuous r.v. (for instance, it applies to some versions of IC [Lea06]). The reason is that the mere fact of *measuring* the complexity of the continuous r.v. y requires approximating it, as reflected by the error term ϵ in (5.5). Thus ϵ is an *additional distortion* which is not accounted for in (5.2). While it is possible to map the error ϵ back to the distortion $d(x, gy)$ through the transformation $g \in G$, doing so is cumbersome. Fortunately there is a simple short-cut that works well in practice. The idea is to tune ϵ adaptively as a fraction of the average variance

$E[\|y-\mu\|^2] = \operatorname{tr}\Sigma_p$ of the code itself. This yields the corrected complexity term[5]

$$\mathcal{C}'(x,y) = \frac{1}{2}\log\det\left(I + \frac{\Sigma_p}{\epsilon^2\operatorname{tr}\Sigma_p}\right). \tag{5.6}$$

Notice that (5.5) can still be used in place of (5.6) when the particular problem prevents the degenerate solution to be found (for instance, (5.5) works well for aligning images).

Example 3 (Removing planar transformations). Consider $g : \mathbb{R}^2 \to \mathbb{R}^2$ acting on a set of K 2-D points $x_1, \dots, x_K \in \mathbb{R}^2$ (Fig. 5.5). We compute the transformations $g_1, \dots, g_K \in G$ and codes $y_1, \dots, y_K \in \mathbb{R}^2$ by minimizing the cost function

$$\begin{aligned} E(\{g_k, y_k\}) &= D(x,y) + \lambda\mathcal{C}'(x,y) \\ &= \frac{1}{K}\sum_{k=1}^{K}\|x_k - g_k y_k\|^2 + \frac{\lambda}{2}\log\det\left(I + \frac{YY^\top}{\epsilon^2\operatorname{tr}YY^\top}\right) \end{aligned} \tag{5.7}$$

where $Y = \begin{bmatrix} y_1 - \mu & \dots & y_K - \mu \end{bmatrix}$ is the matrix of the centered codes and $\mu = \sum_{k=1}^{K} y_k/K$ is the sample average. This can be done by gradient descent. In Fig. 5.5 we use this method to remove respectively rotations around the origin $G_1 = SE(2)$ and scalings $G_2 = \mathbb{R}$. The latter case clearly illustrates the importance of the normalization in (5.6), lest all points collapse to the origin.

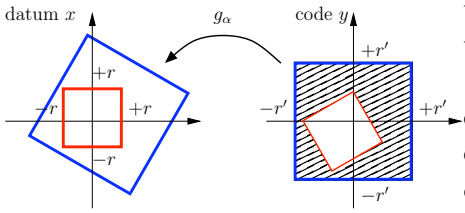
5.4.2 Application to images

Given samples $\{x_1, \dots, x_K\}$ of the random image x, the problem is then to find transformations $\{\alpha_1, \dots, \alpha_K\}$ and codes $\{y_1, \dots, y_K\}$ that minimize

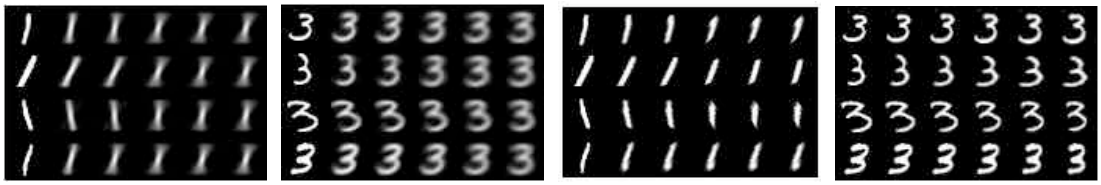
$$E(\{\alpha_k, y_k\}) = \frac{1}{K}\sum_{k=1}^{K}\|x_k - g_{\alpha_k} y_k\|^2 + \lambda\frac{1}{2}\log\det\left(I + \frac{YY^\top}{K\epsilon^2}\right), \tag{5.8}$$

where $Y = \begin{bmatrix} y_1 - \mu & \dots & y_k - \mu \end{bmatrix}$ is the matrix of centered codes $y_k - \mu$, and $\mu = \sum_{k=1}^{K} y_k/K$ is the sample average. Notice that (5.8) is formally identical to (5.7), except that the complexity term (5.5) is used as the normalization is not necessary (because warps cannot decrease the variance of the code without affecting the reconstruction accuracy).

[5] There is another interesting interpretation of (5.6). Let g' be the *isotropic* Gaussian closer in KL divergence to $p(y)$. Similarly to $g(y)$, this Gaussian matches the mean of y, and its (isotropic) variance is given by $\operatorname{tr}\Sigma_p/n$. Then the KL divergence $\mathrm{kl}(g\|g')$ of the Gaussians g and g' is equal to (up to constants) $\log\det(\Sigma_g/(\operatorname{tr}\Sigma_p/n))$. Thus minimizing (5.6) can be interpreted as making y as different as possible from an isotropic Gaussian distribution.



5.6: Boundaries: the code y is mapped back to the datum x by a transformation g_α. The portion of the code which is clipped by this operation (dashed area) needs not match x. If, however, x is extracted from a larger image or can otherwise be extended to the real plane, then the context of x can be used to fill the dashed area.



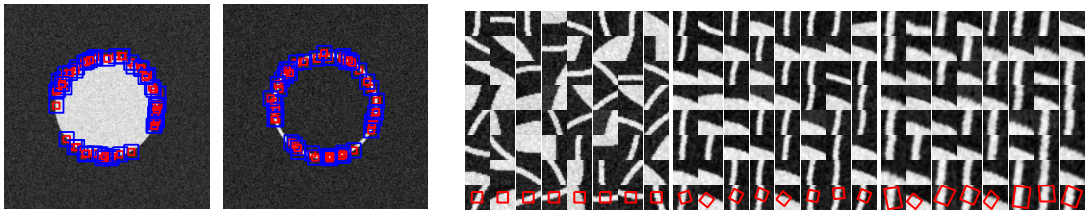
(a) codewords **(b)** codewords **(c)** aligned **(d)** aligned

Figure 5.7: Aligning NIST digits: (a), (b) show examples of original digits (first column) and of the codewords found by minimizing (5.8) for different values of λ. From second column to last: $\lambda = 1.5, 2, 2.5, 3, 3.5$. (c), (d) show the aligned versions of the same digits ($g_\alpha^{-1}x$) obtained by backprojection according to the optimal transformation.

Experiments. We explore the effect of minimizing the cost functional (5.8) on the NIST handwritten digits dataset. A simple gradient descent method was used to find the optimal set of codewords and transformation parameters $\{y_k, \alpha_k\}$. For each digit, a set of 500 samples was extracted and aligned. The result is shown in Fig. 5.7 for different values of the trade-off parameter λ. Note that increasing λ the structure of the codewords converges to a low-dimensional space, eventually collapsing to a zero dimensional space (a single template).

Once the algorithm has found the optimal codeword and transformation g_α for each sample, we can apply the reverse transformations g_α^{-1} on the original digits to obtain the aligned dataset, as shown in Fig. 5.7-(c,d). Figure 5.8 shows the mean of all the digits aligned in this way, compared to the mean before alignment. The result is qualitatively similar to IC.

Figure 5.8: Aligning NIST digits: Per-digit average of the original data (above) and of the aligned data (below). It can be seen that the average appears much sharper after the alignment process despite only affine transformations are removed.



(a) context **(b)** original **(c)** direct **(d)** gradient

Figure 5.9: Aligning bars and wedges by the efficient formulation of Section 5.5: (a) two images from which a number of patches are sampled (in red we show the actual patch, and in blue the context used to complete the code $y = g_\alpha^{-1} x$); (b) a few such patches; (c) alignment based on direct search of rotation and translation; (d) refinement based on gradient descent on the full six parameters of the affine transformation. Note that in (c), (d) all bars are aligned and so are edges; the algorithm found two "codewords" to represent the data.

5.5 An efficient variant for decimated affine transformations

In this section we derive a variant of the model (5.8) which is computationally more attractive. The key idea is that, instead of explicitly estimating the codes y_k, one could simply let $y_k = g_\alpha^{-1} x_k$ and avoid estimating the codes altogether. Unfortunately doing so requires in general to extend the image x beyond its bounding box (see Fig. 5.6 and Section 5.4.2), so this method can be used only if there is a reasonable way of doing so. For instance, in Fig. 5.9 small patches x are naturally extended by their context in the larger image and in Fig. 5.10 the hand-written digits are naturally extended by zero-padding.

By letting $y_k = g_\alpha^{-1} x_k$ the distortion term becomes $\|x - g_\alpha y\|^2 = \|x - g_\alpha(g_\alpha^{-1} x)\|^2$, which in general is *not* identically zero since the action of g_α on a discrete image is not necessarily invertible. In particular, $y = g_\alpha^{-1} x$ could be used

(a) original **(b)** direct **(c)** gradient **(d)** PCA

Figure 5.10: Aligning NIST digits: (a),(b) and (c) have been obtained as in Fig. 5.9; (d) shows the singular values of the three datasets (a),(b) and (c): notice the progressive reduction in the linear dimensionality of the data.

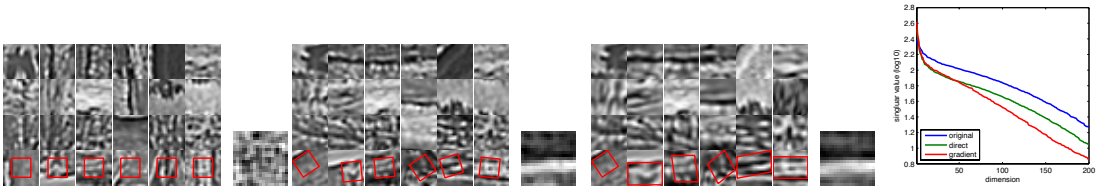
to decimate the data by mapping the data x to a constant code $g_\alpha^{-1}x$ which in turn would trivially decrease the complexity $\mathcal{C}(x, y)$. So the term $\|x - g_\alpha(g_\alpha^{-1}x)\|^2$ forces the code $y = g_\alpha^{-1}x$ to preserve information about the datum x.

Notice that IC uses implicit codes too [Lea06]. However, in place of the distortion term $\|x - g_\alpha(g_\alpha^{-1}x)\|^2$, IC simply penalizes transformations g_α that differ from the identity. This method has the advantage of speed and simplicity. Motivated by this observation, we experimented with a few surrogates of the distortion term and found that the simple function $\beta(x)/|\det(A)|$ approximates well $\|x - g_\alpha(g_\alpha^{-1}x)\|^2$. Here $\beta(x)$ is a constant which depends only on the datum x and can be estimated easily during pre-processing. Of course, this approximation is valid as long as the bounding box of the image x is mapped within the bounding box of the image y (Fig. 5.6), which can be enforced as a set of sixteen linear constraints $M\alpha + b \preceq 0$. It is convenient to incorporate these additional constraints into the energy function as a logarithmic barrier [BV04], yielding to the formulation

$$E(\{\alpha_k\}) = \frac{1}{K}\sum_{k=1}^{K}\left(\frac{\beta_k}{\det A_k} - \frac{1}{\gamma}\sum_{l=1}^{16}\log(-e_l^\top(M\alpha_k + b))\right) + \lambda\frac{1}{2}\log\det\left(I + \frac{YY^\top}{K\epsilon^2}\right), \quad (5.9)$$

where $\alpha_k = \begin{bmatrix}A_k & T_k\end{bmatrix}$, $Y = [g_{\alpha_1}^{-1}x_1 - \mu,\ \ldots,\ g_{\alpha_K}^{-1}x_K - \mu]$ is the matrix of the centered implicit codes and γ is the slope of the logarithmic barrier (we use a large value of γ so that the barrier has an effect only at the boundaries of the feasible region).

Optimization. We optimize (5.9) one image per time, looping over the entire dataset $x_1, \ldots, x_K$ several times. By doing this we can derive an efficient update rule for the transformations α_k. We start by noting that in (5.9) the only term that couples the different data is the entropic term through the covariance $C =$



(a) original **(b)** direct **(c)** gradient **(d)** PCA

Figure 5.11: Aligning natural image patches: We use the same conventions of Fig. 5.9. The alignment results may not be evident from the patches alone, but the mean of the data (small images on the right) reveals the structure found by the algorithm. (d) PCA analysis of (a), (b) and (c) reveals the decreasing linear complexity.

$I + YY^\top/K\epsilon^2$. Now fix the attention on a particular code y_k. As we vary y_k while keeping the other variables fixed, the matrix C becomes

$$C = C - \frac{(y_k - \mu)(y_k - \mu)^\top}{K\epsilon^2} + \frac{(y_k - \mu)(y_k - \mu)^\top}{K\epsilon^2} = \tilde{C} + \frac{(y_k - \mu)(y_k - \mu)^\top}{K\epsilon^2}. \quad (5.10)$$

We can expand the entropic term to the second order around $y_k = \mu$, obtaining

$$\frac{1}{2}\log\det\left(\tilde{C} + \frac{(y_k - \mu)(y_k - \mu)^\top}{K\epsilon^2}\right) \approx \frac{1}{2}(y_k - \mu)^\top \frac{\tilde{C}^{-1}}{K\epsilon^2}(y_k - \mu) + \text{const.}$$

which is a good approximation if $\|y_k - \mu\|/\epsilon\sqrt{K}$ is small, i.e. when K is large. Moreover for a large K we have $\tilde{C} \approx C$. Adding the other terms of (5.9) back, we get that, as long as only one image is changed and K is sufficiently large, (5.9) can be approximated well by

$$E(\alpha_k) \approx \frac{1}{K}\left(\frac{\beta_k}{\det A_k} - \frac{1}{\gamma}\sum_{l=1}^{16}\log(-e_l^\top(M\alpha_k + b))\right) + \frac{\lambda}{2\epsilon^2 K}(g_{\alpha_k}^{-1}x - \mu)C^{-1}(g_{\alpha_k}^{-1}x - \mu) + \text{const.} \quad (5.11)$$

which depends only on α_k. We use two algorithms to optimize (5.11). The first, dubbed *direct search*, simply tries a number of values of each parameter of the transformation α_k (this is basically the same strategy of IC). The second, dubbed *gradient search*, uses the efficient Gauss-Newton quadratic approximation of (5.11). In Fig. 5.10 we used the efficient formulation (5.11) and the two algorithms to align NIST digits, and we got alignment results analogous to the one obtained from the formulation (5.8).

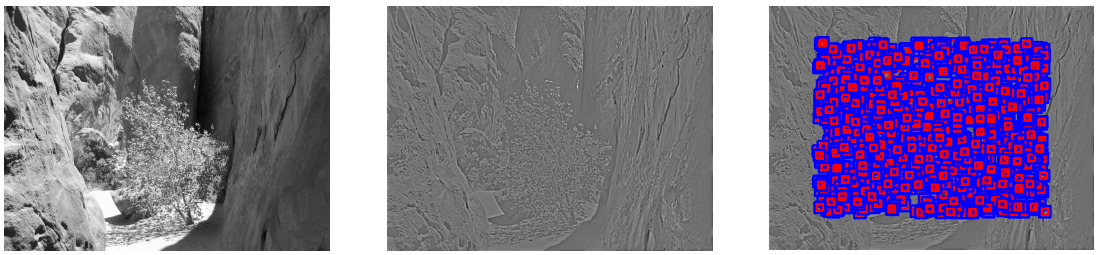


Figure 5.12: Extracting natural image patches: from left to right: a natural image (640×480); the image whitened and contrast normalized; selected 11×11 patches.

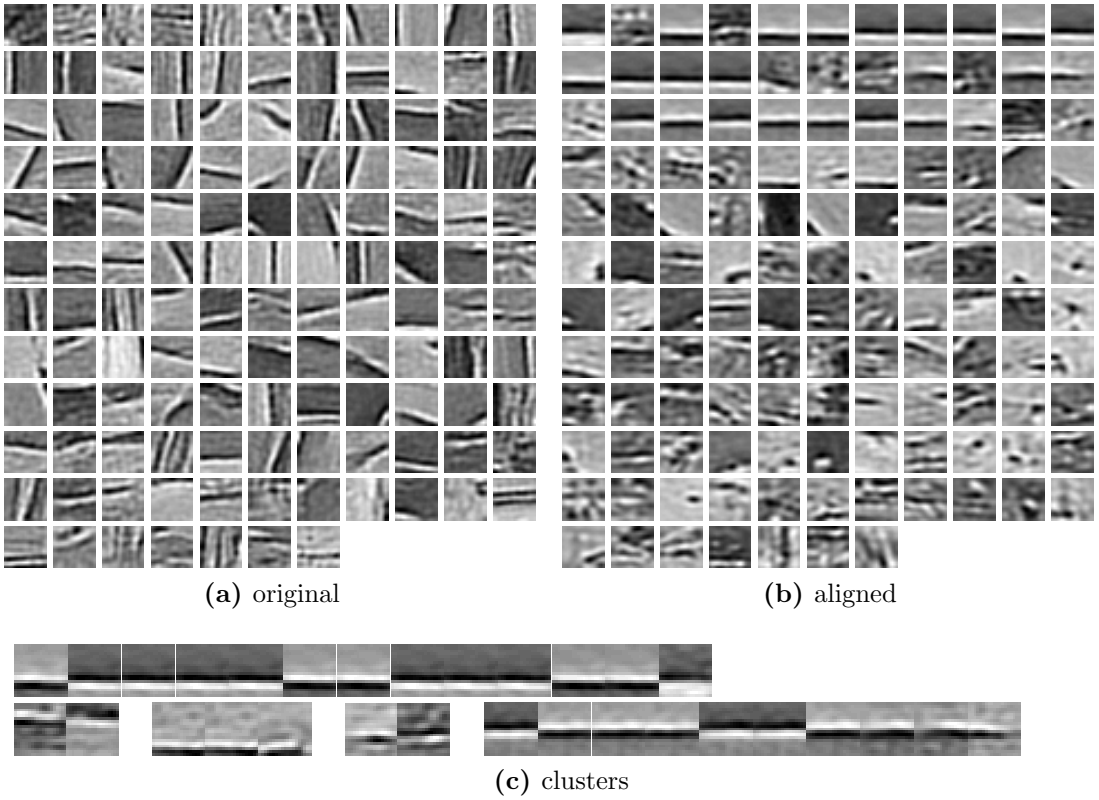
5.6 Aligning natural image patches

Starting from [OF96], there has been an emerging interest in studying sparse representations of natural images. Results from [OF96] show that, when a collection of natural patches are projected onto a linear basis whose coefficients have sparse statistics, structures such as bars, wedges and dots emerge, which resemble receptive fields of the human brain cortical areas V1, V2. Formally, given a collection $Y = \begin{bmatrix} y_1, \dots, y_K \end{bmatrix}$ of such natural patches, the sparse decomposition could be obtained by minimizing

$$E(A, B) = \|Y - BA\|_F^2 + \eta \sum_{qk} \log(1 + a_{qk}^2),$$
$$\text{subject to } \|b_q\|^2 = \beta > 0 \text{ for } q = 1, \dots, Q \quad (5.12)$$

where N is the number of pixels of each image y_k, $B = \begin{bmatrix} b_1 & \dots & b_Q \end{bmatrix} \in \mathbb{R}^{N \times Q}$ is the matrix of basis elements b_q and $A = [a_{qk}] \in \mathbb{R}^{Q \times K}$ is the matrix of coefficients a_{qk} and η a parameter controlling the sparsity of the solution. For this procedure to work well, natural images must be appropriately whitened and contrast normalized [OF96]. The result of minimizing (5.12) over $Q = 128$ basis elements on a collection of 5000 natural image patches extracted in such a way is shown in Fig. 5.13-(a).

From Fig. 5.13-(a) and the analogous results obtained by many other authors, it is evident that many of the structures found are similar, differing only by geometric parameters such as position, orientation and scale. Recently it has been argued by [GR05, OCC07] that these systematic transformations could be estimated and removed, obtaining a more compact representations which would also be invariant to such kind of geometric distortions.



(a) original **(b)** aligned

(c) clusters

Figure 5.13: Sparse decomposition of natural image patches: (a) decomposition of 5000 image patches (basis elements are ordered by decreasing Kurtosis of their coefficients). (b) decomposition of the same 5000 patches after alignment. (c) duplicate basis function detected in (b), showing significantly small complexity of the basis after alignment.

To this end [GR05, OCC07] extend the generative model (5.12) to account for geometric transformations and solve for basis, coefficients and geometric parameters (this is not dissimilar from [FJ99], except for the sparsity prior). Unfortunately this results in a large computation which is unstable. Moreover, we explicitly address the problem of boundaries, which are an important factor even when dealing with simple transformation such as translations.

Since our alignment algorithm is capable of decreasing the dimensionality of the linear embedding spanned by the data (no matter whether its statistic is sparse or Gaussian) it may be appropriate as a pre-processing step to align the collection of natural image patches before the sparse dimensionality reduction (5.12). The result of such alignment is shown in Fig. 5.11 and Fig. 5.13-(b) illustrates the result of applying the very same algorithm of Fig. 5.13-(a) to the

aligned data. Several observations can be made. First, a few dominant horizontal structures emerge, which evidently subsume many of the other structures found in Fig. 5.13-(a) at different orientations and translations. Second, such structures are found multiple times, with exactly the same appearance *and* position and orientation. To quantify this phenomenon, in Fig. 5.13-(a) we collapse similar basis elements until the reconstruction error in (5.12) increases less than 1% (we do this by iteratively collapsing the pair of most similar basis elements). This shows quantitatively that indeed several of the basis elements are redundant copies, created by the local optimization procedure used to minimize (5.12). Third, a number of relatively unstructured basis elements remain, which may indicate that the variety of strong structures has been significantly reduced by aligning the data.

Discussion

We have presented a novel approach to perform alignment with respect to transformations of the data that are not invertible. We have showed that a measure of complexity can be defined that is tailored to the postulated structure of the space where the codebook lives, and in particular we explore the case of affine subspaces of the embedding space of the raw data. We have presented efficient alignment algorithms that allow aligning large collections of handwritten digits and natural image patches, and more general real valued data.

CHAPTER 6

Invariant Boosted Learners

In the fist part of this thesis we studied how invariance properties of visual data can be exploited in preprocessing, by constructing invariant representations which facilitate subsequent analysis. We have seen that, while useful and of broad applicability, invariant representations are limited in several regards. For instance, in Chapters 2 and 3 we have seen that viewpoint invariant representations are essentially local and carry only partial information about the image. As a consequence, in some cases it may be preferable to bypass preprocessing and exploit invariance directly during analysis. In this chapter we propose, in particular, a method for learning a classifier that reflects invariance properties of the data (see also Section 1.2). This is an important special case, to which many object and category recognition problems can be reduced.

The simplest way to extend a learning algorithm to incorporate data invariance is to artificially modify the training data and make it representative of the irrelevant variations that need to be captured. This can be done by the addition of *virtual samples*, i.e. of samples obtained by transforming the available data by the irrelevant transformations [GC95, PB06]. This approach is conceptually straightforward and effective, but makes the training phase extremely slow and memory inefficient.

Instead of manipulating the training data, invariance can be incorporated in the cost function used for selecting the optimal classifier during training. Since the latter is usually the empirical error of the classifier, this can be done by computing analytically the error when random transformations are applied to the data (assuming to know the distribution of such transformations). This is, for example, the approach used in *Vicinal Risk Minimization* (VRM) [CWB00, PB06].

In this chapter we show how the latter idea can be applied to the AdaBoost algorithm in a way which results in a simple and powerful learning algorithm. We attach probability distributions to transformations that are local rather than global, as the extrapolation of samples for large transformations may not be reliable. This allows us to introduce tangent vector methods [PB06, SVL92] in our algorithm. To further limit the amount of computations, we introduce two additional elements: gradient descent and Haar wavelet projection. Our gradient

descent procedure allows to find good features automatically and efficiently so that exhaustive search over large databases can be avoided. Gradient methods have been also suggested in a scale-space scheme in VRM, but not in a boosting framework [CWB00]. Haar wavelet projection allows to dramatically reduce the amount of computations at test time by mapping each feature to a finite set of Haar wavelets and therefore enabling the use of integral images [POP98, VJ04b].

In the next section we revisit the basics of the binary classification problem; then, in Section 6.2 we introduce the tangent space approach and, finally, in Section 6.3 we present the Parzen-AdaBoost approach.

6.1 Binary classification

For the sake of simplicity, in this chapter we discuss only the binary classification problem. Extensions to multiple classes can be easily obtained, for instance, by using AdaBoost.MH [SS98]. In binary classification we are given a collection of N i.i.d.[1] observations $\mathbf{x}_1, \ldots, \mathbf{x}_N \in \mathcal{X}$ and labels $y_1, \ldots, y_N \in \{-1, +1\}$. The goal is to design a function $H : \mathcal{X} \mapsto \{-1, +1\}$, a *classifier*, that predicts the label y of a generic observation $\mathbf{x}$. As we deal with images, $\mathcal{X} \subseteq \mathbb{R}^d$, where d is the number of pixels of the image. We denote the unknown joint probability distribution density of $(\mathbf{x}, y)$ with $p(\mathbf{x}, y)$ and the *expectation* of a random variable w with respect to p with $E_p[w]$.

The *optimal classifier* H is the one that minimizes the so-called *01-loss expected risk*

$$\mathrm{Err}_p(H) \doteq E_p[I(H(\mathbf{x}) \neq y)] = \sum_{y=\{-1,+1\}} \int p(\mathbf{x}, y) I(H(\mathbf{x}) \neq y)\, d\mathbf{x} \tag{6.1}$$

where $I(\mathcal{A})$ is the indicator function of the event $\mathcal{A}$. Unfortunately, the optimal classifier cannot be computed as eq. (6.1) requires the distribution $p(\mathbf{x}, y)$, which is unknown. The common strategy is then to approximate $p(\mathbf{x}, y)$ by using the available set of samples $\{(\mathbf{x}_i, y_i),\ i = 1, \ldots, N\}$. One such approximation is to replace the true distribution $p(\mathbf{x}, y)$ with the *empirical distribution*

$$\widehat{p}_e(\mathbf{x}, y) = \frac{1}{N} \sum_{i=1}^{N} \delta(\mathbf{x} - \mathbf{x}_i)\delta(y - y_i) \tag{6.2}$$

where $\delta(\mathbf{x})$ and $\delta(y)$ are Dirac's deltas. This choice leads to the minimization of

[1] The notation i.i.d. stands for *independent and identically distributed*.

the *empirical loss*[2]

$$\mathrm{Err}_{\widehat{p}_e}(H) \propto -\frac{1}{N}\sum_{i=1}^{N} y_i H(\mathbf{x}_i). \tag{6.3}$$

Alternatively to eq. (6.2), one can consider *Parzen's windows* [DHS01]

$$\widehat{p}_g(\mathbf{x}, y) = \frac{1}{N}\sum_{i=1}^{N} g_\Sigma(\mathbf{x} - \mathbf{x}_i)\delta(y - y_i) \tag{6.4}$$

where $g_\Sigma(\mathbf{x})$ denotes a zero-mean Gaussian distribution with covariance Σ. In this case, the classification error is

$$\begin{aligned} \mathrm{Err}_{\widehat{p}_g}(H) &\propto E_{\widehat{p}_g}[-yH(\mathbf{x})] \\ &= -\frac{1}{N}\sum_{i=1}^{N}\int g_\Sigma(\mathbf{x} - \mathbf{x}_i) y_i H(\mathbf{x}) d\mathbf{x} \end{aligned} \tag{6.5}$$

An interesting observation in [CWB00] that we exploit in our algorithm is that Parzen's windows method can be formulated as the empirical loss of a smoothed classifier H, i.e.,

$$\mathrm{Err}_{\widehat{p}_g}(H) = \mathrm{Err}_{\widehat{p}_e}(g_\Sigma * H). \tag{6.6}$$

6.1.1 Discrete AdaBoost

In boosting one builds a classifier H by combining additively several so-called *weak classifiers* [FHT00, HTF01]. The key idea is that, as long as the weak classifiers do better than chance, it is possible to *boost* their performance by combining them linearly.

In Discrete AdaBoost M weak classifiers $f_m : \mathcal{X} \mapsto \{-1, +1\}$ are combined to yield an *auxiliary function* F_M and the corresponding *strong classifier* H_M

$$F_M(\mathbf{x}) = \textstyle\sum_{m=1}^{M} c_m f_m(\mathbf{x}), \quad H_M(\mathbf{x}) = \mathrm{sign}\left(F_M(\mathbf{x})\right) \tag{6.7}$$

with parameters $c_1, \ldots, c_M \in \mathbb{R}$. Rather than directly minimizing the empirical error (6.3), AdaBoost minimizes the *exponential loss*

$$E_{\widehat{p}_e}[e^{-yF_M(\mathbf{x})}] \tag{6.8}$$

which bounds from above the empirical error (6.3) as $e^{-yF_M(\mathbf{x})} \geq I(H_M(\mathbf{x}) \neq y)$, $\forall M$. To limit the computational burden the auxiliary function F_M is built

[2] In our notation $A(\mathbf{x}) \propto B(\mathbf{x})$ if and only if there exist constants $a > 0, b$ such that $A(\mathbf{x}) = aB(\mathbf{x}) + b$.

iteratively. Given F_{m-1} one searches for the optimal update $c_m f_m$ such that the exponential loss (6.8) is minimized. Every iteration can be written recursively by means of weights $w(\mathbf{x}, y) = e^{-yF_{m-1}(\mathbf{x})}$, which automatically concentrate the error on the difficult samples. The algorithm is summarized in Algorithm 3.

Algorithm 3 Discrete AdaBoost

1: Initialize $F_0(\mathbf{x}) = 0$ for all $x \in \mathcal{X}$.
2: Initialize $w(\mathbf{x}_i, y_i) = 1/N$ for all $i = 1, 2, \ldots, N$.
3: **for** $m = 1$ to M **do**
4: Find the weak classifier $f_m \in \mathcal{F}$ that minimizes

$$\mathrm{Err}_q(f) \propto \sum_{i=1}^{N} w(\mathbf{x}_i, y_i) I(f(\mathbf{x}_i) \neq y_i)$$

where the distribution q is defined as

$$q(\mathbf{x}, y) \propto \sum_{i=1}^{N} w(\mathbf{x}, y) \delta(\mathbf{x} - \mathbf{x}_i) \delta(y - y_i);$$

5: Let

$$c_m \leftarrow \frac{1}{2} \log \frac{1 - \mathrm{Err}_q(f_m)}{\mathrm{Err}_q(f_m)};$$

6: Update the weights

$$w(\mathbf{x}_i, y_i) \leftarrow w(\mathbf{x}_i, y_i) e^{-y_i c_m f_m(\mathbf{x}_i)};$$

7: Update the auxiliary function $F_m = F_{m-1} + c_m f_m$.
8: **end for**

6.2 Invariance and tangent spaces

As mentioned in the Introduction, in many computer vision applications one is interested in classifying objects in images irrespectively of translations $t \in \mathbb{R}^2$, rotations[3] $R \in SO(2)$, scalings $s \in [0, +\infty)$, and changes in intensity due to contrast $b \in [0, +\infty)$. Let $\alpha = [t \ R \ s \ b] \in \mathcal{L}$ be a vector lumping all the transformation parameters. Then, given a sample $\mathbf{x}$, the transformed sample $\mathbf{x}_\alpha$ is defined as

$$\mathbf{x}_\alpha \doteq T(\mathbf{x}, \alpha) \tag{6.9}$$

[3] $SO(2)$ denotes the *special orthogonal* group of 2-D rotation matrices.

where $T : \mathcal{X} \times \mathcal{L} \mapsto \mathcal{X}$ is the *morphing function* defined via

$$T(\mathbf{x}, \alpha)(x) \doteq b\mathbf{x}(sRx + t) \tag{6.10}$$

and $x \in \mathbb{R}^2$ denotes the 2-D coordinates of $\mathbf{x}$. Let $p(\alpha)$ be the probability density of a transformation α. Then, one could incorporate invariance to transformations in the empirical error as follows

$$\mathrm{Err}^{inv}_{\widehat{p}_e}(H) \propto -\frac{1}{N}\sum_{i=1}^{N}\int p(\alpha) y_i H(T(\mathbf{x}_i, \alpha)) d\alpha. \tag{6.11}$$

Virtual samples can be easily generated by letting $p(\alpha) = \sum_{j=1}^{K} \delta(\alpha - \alpha_j)$ for a certain set of transformations $\{\alpha_j\}_{j=1,2,\ldots,K}$. However, this has the immediate effect of multiplying the size of the data set of the samples by a factor K. Moreover, notice that virtual samples generated for large transformations may not reliably substitute real samples due to missing data, sampling, and quantization. This is the case, for instance, when we scale, translate, or rotate an image. Therefore, we consider incorporating invariance to transformations only locally at each sample. To do so, we use the tangent vector approach [SVL92], i.e., we approximate the global transformation at each sample as

$$T(\mathbf{x}, \alpha) \simeq \mathbf{x} + \sum_{k=1}^{K} L_k(\mathbf{x}, \alpha_k^0)(\alpha_k - \alpha_k^0) \tag{6.12}$$

where α^0 is the identity transformation, which we can assume identically zero without loss of generality, and $L_k : \mathcal{X} \times \mathcal{L} \mapsto \mathcal{X}$ are local transformations defined as

$$L_k(\mathbf{x}, \alpha^0) \doteq \left.\frac{\partial T}{\partial \alpha_k}(\mathbf{x}, \alpha)\right|_{\alpha=\mathbf{0}}. \tag{6.13}$$

Note that L_k are operators that generate the whole space of local transformations (a Lie algebra of local transformations). Such operators can be computed analytically or, more easily, by using finite differences. Finally, to enforce locality we assume that the prior $p(\alpha)$ is Gaussian with mean $\alpha_0 = \mathbf{0}$ and diagonal covariance matrix Ψ.[4] By substituting this prior in eq. (6.11) and by using the tangent vector approximation we obtain

$$\mathrm{Err}^{inv}_{\widehat{p}_e}(H) \propto -\frac{1}{N}\sum_{i=1}^{N} y_i \left(g_{\Sigma_i} * H\right)(\mathbf{x}_i). \tag{6.14}$$

[4]One way to estimate Ψ is, for example, via cross-validation. However, in our experiments we manually fix Ψ to the maximum amount possible and within the limits imposed by linearization.

where $\Sigma_i = L(\mathbf{x}_i, \mathbf{0})\Psi L(\mathbf{x}_i, \mathbf{0})^T$ and we have defined $L(\mathbf{x}_i, \mathbf{0}) = [L_1(\mathbf{x}_i, 0)\ L_2(\mathbf{x}_i, 0)\ \ldots\ L_K(\mathbf{x}_i, 0)]$. Notice that the above equation can also be readily interpreted as Parzen's windows error where the covariance of the Gaussian kernel in eq. (6.3) is $\Sigma = \Sigma_i$, i.e.,

$$\mathrm{Err}^{inv}_{\widehat{p}_e}(H) = \mathrm{Err}_{\widehat{p}_e}(g_\Sigma * H) = \mathrm{Err}_{\widehat{p}_g}(H). \tag{6.15}$$

Let us now consider the case of additive Gaussian noise $w \sim p(w)$. This case is particularly interesting as it corresponds to no a-priori knowledge where every pixel of the image $\mathbf{x}$ is affected by an unknown disturbance. In this case, the tangent vectors cover the whole space $\mathcal{X}$ and Σ becomes a diagonal matrix, thus yielding *isotropic smoothing*. In other words, when samples are affected by additive Gaussian noise that is independent at each pixel, virtual samples lie in a sphere around the original image sample. This has the effect of increasing the classifier margin [Vap95].

6.3 Parzen-AdaBoost

So far, AdaBoost has been based on the empirical distribution in eq. (6.2). We now look at the extension of AdaBoost to Parzen's windows eq. (6.4) because, as we have seen in Section 6.2, it allows us to incorporate invariance to a prescribed set of transformations.

Similarly to Discrete AdaBoost, our algorithm is based on the following inequality

$$\frac{1}{2}E_{\widehat{p}_e}\left[1 - y\mathrm{sign}(g_\Sigma * F(\mathbf{x}))\right] \leq E_{\widehat{p}_e}[e^{-yg_\Sigma * F(\mathbf{x})}]. \tag{6.16}$$

The auxiliary function F of a strong classifier $H(\mathbf{x}) = \mathrm{sign}\ F(\mathbf{x})$ is written as a summation $F_M = \sum_{m=1}^{M} c_m f_m$, so that $g_\Sigma * F_M = \sum_{m=1}^{M} c_m\,(g_\Sigma * f_m)$ and this corresponds to smoothing each weak classifier. This approach has three advantages: First, eq. (6.16) guarantees that by minimizing the right-hand side one improves the empirical error eq. (6.3) of the strong classifier $\mathrm{sign}(g_\Sigma * F)$. Second, by selecting weak classifiers based on eq. (6.16) one automatically incorporates invariance.[5]

[5]Later, we will see that the strong classifier F approximately minimized Parzen's loss (6.6). When F is a single weak classifier of the form $F(\mathbf{x}) = c\ \mathrm{sign}(\gamma_1^T\mathbf{x} + \gamma_0)$ the bound $E_{\widehat{p}_e}\left[1 - y\mathrm{sign}(g_\Sigma * F(\mathbf{x}))\right]/2 \leq E_{\widehat{p}_e}[e^{-yg_\Sigma * F(\mathbf{x})}]$ is guaranteed for any $c < 2.678$. When F consists of more than one weak classifier, we assume that only one weak classifier is changing sign "close to" a sample. In this context, the notion of distance from a sample is based on Σ. In this case, we have that $F(\mathbf{x}) = c\ \mathrm{sign}(\gamma_1^T\mathbf{x} + \gamma_0) + \theta$ where θ is constant (near a sample) and collects all the decisions from the other weak classifiers. The above bound is again guaranteed for $c < 0.693$.

Third, the minimization of eq. (6.16) results in a very efficient algorithm, as few weak classifiers are required.[6] We call this novel method *Parzen-AdaBoost*.

Our strategy is to find a recursive iteration along the lines of Discrete AdaBoost to minimize Parzen's windows loss in eq. (6.5). Similarly to Discrete AdaBoost we have a strong classifier $H(\mathbf{x}) = \text{sign}(g_\Sigma * F_M(\mathbf{x}))$, where $F_M(\mathbf{x}) = \sum_{m=1}^{M} c_m f_m(\mathbf{x})$, and define Parzen's windows exponential loss as

$$E_{\widehat{p}_e}\left[e^{-y\ g_\Sigma * F_M(\mathbf{x})}\right]. \tag{6.17}$$

Then, given a classifier F_{m-1} we search for the optimal update $c_m f_m$ such that eq. (6.16) is minimized. Thanks to the exponential form, we can separate the update from the current classifier so that

$$E_{\widehat{p}_e}\left[e^{-y\ g_\Sigma * (F_{m-1} + c_m f_m)(\mathbf{x})}\right] = E_q\left[e^{-y\ c_m g_\Sigma * f_m(\mathbf{x})}\right] \tag{6.18}$$

where

$$\begin{array}{rcl} q(\mathbf{x}, y) & = & \frac{1}{N}\sum_{i=1}^{N} w(\mathbf{x}_i, y_i)\delta(\mathbf{x} - \mathbf{x}_i)\delta(y - y_i) \\ w(\mathbf{x}_i, y_i) & = & e^{-y_i\ g_{\Sigma_i} * F_{m-1}(\mathbf{x}_i)}. \end{array} \tag{6.19}$$

Thus, at the m-th iteration we need to minimize

$$\sum_{i=1}^{N} w(\mathbf{x}_i, y_i) e^{-y_i c_m g_\Sigma * f_m(\mathbf{x}_i)} \tag{6.20}$$

with respect to f_m and c_m. Finding a closed form solution for this minimization problem is not straightforward. In this chapter we follow an approach proposed by Friedman [Fri01] and consider the minimization of eq. (6.20) in function space. First, we compute the derivative of eq. (6.20) along an arbitrary weak classifier $h : \mathcal{X} \mapsto \mathbb{R}$. This yields

$$\lim_{\epsilon \to 0} \frac{E_q\left[e^{-y\ g_\Sigma * \epsilon h(\mathbf{x})}\right] - E_q\left[e^0\right]}{\epsilon} = -E_q\left[y\ g_\Sigma * h(\mathbf{x})\right]. \tag{6.21}$$

Then we search for the weak classifier f_m that results in the steepest descent, i.e., that maximizes

$$E_q\left[y\ g_\Sigma * f_m(\mathbf{x})\right] \propto -\,\text{Err}_q(f_m) \tag{6.22}$$

where

$$\text{Err}_q(f_m) = E_q\left[\frac{1 - y\ g_\Sigma * f_m(\mathbf{x})}{2}\right]. \tag{6.23}$$

Thus we are searching for the weak classifier f_m that, after smoothing, has the minimum weighed risk. Once f_m has been chosen, we need to compute the

[6] This algorithm can also be cast as a Real-Boost method where we search for the optimal weak classifiers among the smooth and transformation invariant ones.

optimal step c_m. To this end, we consider the first and second derivative of eq. (6.20) with respect to c_m, i.e.,

$$\begin{aligned} \frac{\partial}{\partial c_m} E_q\left[e^{-y\ c_m g_\Sigma * f_m(\mathbf{x})}\right] &= E_{q'}[-y\ g_\Sigma * f_m(\mathbf{x})] \\ \frac{\partial^2}{\partial c_m^2} E_q\left[e^{-y\ c_m g_\Sigma * f_m(\mathbf{x})}\right] &= E_{q'}[(g_\Sigma * f_m)^2(\mathbf{x})] \end{aligned} \tag{6.24}$$

where $q'(\mathbf{x}) = q(\mathbf{x})e^{-y\ c_m g_\Sigma * f_m(\mathbf{x})}$ is the iteratively updated weighed distribution. Thus, starting from

$$c_m \leftarrow \frac{1}{2}\log\frac{1-\mathrm{Err}_{q'}(f_m)}{\mathrm{Err}_{q'}(f_m)} = \frac{1}{2}\log\frac{1+E_{q'}[y\ g_\Sigma * f_m(\mathbf{x})]}{1-E_{q'}[y\ g_\Sigma * f_m(\mathbf{x})]} \tag{6.25}$$

we get the Gauss-Newton update

$$c_m \leftarrow \frac{E_{q'}[y\ g_\Sigma * f_m(\mathbf{x})]}{E_{q'}[(g_\Sigma * f_m)^2(\mathbf{x})]}. \tag{6.26}$$

The algorithm is summarized in Algorithm 4.

Algorithm 4 Parzen-AdaBoost

1: Initialize $F_0(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathcal{X}$.
2: Initialize $w(\mathbf{x}_i, y_i) = 1/N$ for all $i = 1, 2, \ldots, N$.
3: **for** $m = 1$ to M **do**
4: Search for the weak classifier $f_m \in \mathcal{F}$ that minimizes $\mathrm{Err}_q(f_m)$ given in eq. (6.23), where $q(\mathbf{x}, y)$ is given in eq. (6.19). In alternative to the exhaustive search, use the gradient descent method described in Section 6.3.2.
5: Initialize $q' \leftarrow q$.
6: Initialize

$$c_m \leftarrow \frac{1}{2}\log\frac{1-\mathrm{Err}_{q'}(f_m)}{\mathrm{Err}_{q'}(f_m)}. \tag{6.27}$$

7: **while** not converged **do**
8: Calculate $E_{q'}[(g_\Sigma * f_m)^2(\mathbf{x})]$.
9: Calculate $E_{q'}[y\ g_\Sigma * f_m(\mathbf{x})]$.
10: Set $\delta \leftarrow E_{q'}[y\ g_\Sigma * f_m(\mathbf{x})]/E_{q'}[(g_\Sigma * f_m)^2(\mathbf{x})]$.
11: Update $c_m \leftarrow c_m + \delta$.
12: Update $q'(\mathbf{x}, y) \leftarrow q'(\mathbf{x}, y)e^{-\delta y\ g_\Sigma * f_m(\mathbf{x})}$.
13: **end while**
14: Update the auxiliary function $F_m \leftarrow F_{m-1} + c_m f_m$.
15: **end for**

6.3.1 Linear classifiers

In the simplest instance of a classifier $H(\mathbf{x}) = \text{sign } f_1(\mathbf{x})$ the auxiliary function f_1 is linear, i.e.,

$$f_1(\mathbf{x}) = \text{sign}(\gamma_0 + \langle \gamma_1, \mathbf{x} \rangle) \tag{6.28}$$

where $\gamma_0 \in \mathbb{R}$ and $\gamma_1 \in \mathcal{X}$. Graphically, this corresponds to defining a hyper-plane that separates the space of the input images $\mathbf{x}$ into two complementary hyper-volumes. The vector γ_1 defines the normal to the hyper-plane. In practice, γ_1 can be rearranged as an image and be seen as a *feature*. As we will see in the later sections, we can approximate γ_1 with Haar wavelets and improve the computational efficiency of the classifier. In the case of Parzen's windows in eq. (6.4) we immediately find that

$$(g_\Sigma * H)(\mathbf{x}) = \text{erf}\left(\frac{\gamma_0 + \langle \gamma_1, x \rangle}{\sqrt{2\gamma_1^\top \Sigma \gamma_1}} \right) \tag{6.29}$$

and thus the approximate Parzen's windows loss eq. (6.5) becomes

$$\text{Err}_{p_w}(H) = \frac{1}{2} - \frac{1}{2N} \sum_{i=1}^{N} y_i \text{erf}\left(\frac{\gamma_0 + \langle \gamma_1, x_i \rangle}{\sqrt{2\gamma_1^T \Sigma_i \gamma_1}} \right) \tag{6.30}$$

where erf is the *error function* and is defined as

$$\text{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2}\, dt. \tag{6.31}$$

Tangent vectors can then be readily incorporated as suggested in sec. 6.2 by defining $\Sigma_i = L(\mathbf{x}_i, \mathbf{0}) \Psi L(\mathbf{x}_i, \mathbf{0})^T$.

6.3.2 Optimizing linear weak classifiers

The first step in Parzen-AdaBoost is to search for a weak classifier f_m that minimizes eq. (6.23). Typically, one defines a very large set of weak classifiers and then performs an exhaustive search to determine the optimal one. In addition to being rather time-consuming, this procedure yields cumbersome and computationally inefficient strong classifiers when the set of weak classifiers is not chosen purposefully. In this section we suggest a method to automatically design weak classifiers via a gradient descent procedure.

We restrict the weak classifiers to be linear so that, as shown in sec. 6.3.1,

$$f_m(\mathbf{x}) = \text{sign}\left(\gamma_0 + \langle \gamma_1, \mathbf{x} \rangle\right) \tag{6.32}$$

where $\gamma_0 \in \mathbb{R}$ and $\gamma_1 \in \mathcal{X}$. The initialization of the parameters (γ_0, γ_1) is done by selecting a random vector γ_1 and then by a simple line search on the other parameter γ_0. In our algorithm, however, we implement a more efficient method for the initialization of γ_0 based on sorting the responses $\langle \gamma_1, \mathbf{x}_i \rangle$, that we do not report here for lack of space. Once the parameters have been initialized, we compute the gradient of

$$\Phi(\gamma_0, \gamma_1) \doteq \sum_{i=1}^{N} w(\mathbf{x}_i, y_i) \mathrm{erf}\left(\frac{\gamma_0 + \langle \gamma_1, \mathbf{x}_i \rangle}{\sqrt{2\gamma_1^\top \Sigma_i \gamma_1}} \right). \tag{6.33}$$

with respect to (γ_0, γ_1). This results in

$$\begin{array}{rcl} \dfrac{\partial \Phi}{\partial \gamma_0} & = & \sum_{i=1}^{N} w(\mathbf{x}_i, y_i) \dot{\mathrm{erf}}\left(\frac{\gamma_0 + \langle \gamma_1, \mathbf{x} \rangle}{\sqrt{2\gamma_1^\top \Sigma_i \gamma_1}} \right) \frac{1}{\sqrt{2\gamma_1^\top \Sigma_i \gamma_1}} \\ \dfrac{\partial \Phi}{\partial \gamma_1^\top} & = & \sum_{i=1}^{N} w(\mathbf{x}_i, y_i) \dot{\mathrm{erf}}\left(\frac{\gamma_0 + \langle \gamma_1, \mathbf{x} \rangle}{\sqrt{2\gamma_1^\top \Sigma_i \gamma_1}} \right) \frac{1}{\sqrt{2\gamma_1^\top \Sigma_i \gamma_1}} \\ & & \cdot \left(\mathbf{x} - \frac{\gamma_0 + \langle \gamma_1, \mathbf{x} \rangle}{\gamma_1^\top \Sigma_i \gamma_1} \Sigma_i \gamma_1 \right) \end{array} \tag{6.34}$$

where $\dot{\mathrm{erf}}(z)$ denotes the derivative of $\mathrm{erf}(z)$ with respect to z. Let $\nu_0 \doteq \dfrac{\partial \Phi}{\partial \gamma_0}$ and $\nu_1 \doteq \dfrac{\partial \Phi}{\partial \gamma_1}$ be the update directions of γ_0 and γ_1. Then, we can update the weak classifier via

$$\begin{array}{rcl} \gamma_0 & \leftarrow & \gamma_0 + \lambda \nu_0 \\ \gamma_1 & \leftarrow & \gamma_1 + \lambda \nu_1 \end{array} \tag{6.35}$$

given a step $\lambda > 0$. To determine the optimal update step λ, consider

$$\begin{array}{l} \Phi(\gamma_0 + \lambda \nu_0, \gamma_1 + \lambda \nu_1) = \\ \qquad \sum_{i=1}^{N} w(\mathbf{x}_i, y_i) \mathrm{erf}\left(\frac{\beta_{1,i} + \lambda \beta_{2,i}}{\sqrt{2\beta_{3,i} + 4\beta_{4,i}\lambda + 2\beta_{5,i}\lambda^2}} \right) \end{array} \tag{6.36}$$

where

$$\begin{array}{lll} \beta_{1,i} = \gamma_0 + \langle \gamma_1, \mathbf{x}_i \rangle & \beta_{2,i} = \nu_0 + \langle \nu_1, \mathbf{x}_i \rangle & \\ \beta_{3,i} = \gamma_1^\top \Sigma_i \gamma_1 & \beta_{4,i} = \nu_1^\top \Sigma_i \gamma_1 & \beta_{5,i} = \nu_1^\top \Sigma_i \nu_1 \end{array}$$

These equations can be used to compute the energy function for several values of λ rather efficiently. However, to improve efficiency even further we perform a single Gauss-Newton step. Then, we compute the gradient and the Hessian of eq. (6.33) with respect to λ and evaluate them in $\lambda = 0$

$$\begin{array}{rcl} \dfrac{\partial \Phi}{\partial \lambda}(0) & = & \dot{\Phi}(\gamma_0, \gamma_1) \frac{\beta_{2,i}}{\beta_{3,i}^{1/2}} - \frac{\beta_{1,i}\beta_{4,i}}{\beta_{3,i}^{3/2}} \\ \dfrac{\partial^2 \Phi}{\partial \lambda^2}(0) & = & 2\dot{\Phi}(\gamma_0, \gamma_1) \left(\frac{\beta_{2,i}\beta_{4,i}}{\beta_{3,i}^{1/2}} - 3\frac{\beta_{1,i}\beta_{4,i}^2}{\beta_{3,i}^{5/2}} - \frac{\beta_{1,i}\beta_{5,i}}{\beta_{3,i}^{3/2}} \right) \end{array} \tag{6.37}$$

where

$$\dot{\Phi}(\gamma_0, \gamma_1) \doteq \sum_{i=1}^{N} w(\mathbf{x}_i, y_i) \mathrm{erf}\left(\frac{\beta_{1,i}}{\sqrt{2\beta_{3,i}}}\right) \tag{6.38}$$

Finally, $\lambda \leftarrow -\frac{\partial \Phi}{\partial \lambda}(0) \left(\frac{\partial^2 \Phi}{\partial \lambda^2}(0)\right)^{-1}$.

6.3.3 Projection onto Haar wavelets

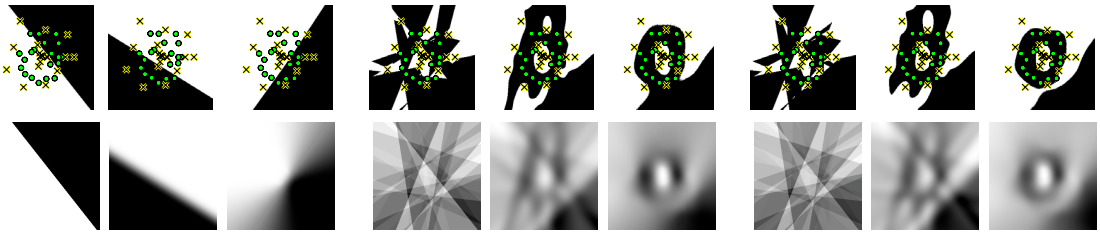
In this section we suggest a simple step to considerably speed up the performance of the classifier at run-time. So far we have been concerned with finding the weak classifiers f_m that minimize a certain exponential loss. We have not considered, however, that the computation of the inner product $\langle \gamma_1, \mathbf{x} \rangle$ is rather intensive for a generic vector γ_1. This is because each element in γ_1 needs to be multiplied by the corresponding element of the image $\mathbf{x}$. Nevertheless, if the elements of γ_1 lie in a rectangular region and have a constant value, the computations can be dramatically reduced by using the well-known *integral image* method [POP98, VJ04b]. More in general, one can use Haar wavelets to compose several rectangular regions and obtain more advanced weak classifiers. In our algorithm, we *project* the weak classifier f_m by truncating its Haar wavelet decomposition. We insert this projection immediately before the update step of the auxiliary function F_m. In the next section we will see that only a few Haar wavelets are required to achieve the desired classification performance.

6.4 Experiments

The proposed algorithm has been thoroughly tested on both synthetic and real data sets. In both cases we illustrate the effects of incorporating isotropic smoothing, gradient descent, tangent vectors, and Haar wavelet projection.

6.4.1 Synthetic data experiments

In Figure 6.1 we show a first experiment on two-dimensional (2-D) data to emphasize the efficacy of smoothing and the tangent vector method with few samples. The synthetic data is invariant to 2-D rotations. We divide the plots into 3 groups (a), (b), and (c) where: (a) corresponds to a strong classifier with only 1 weak classifier, (b) to 100 weak classifiers, and (c) to 300 weak classifiers. For each group the left image shows the classification result of Discrete AdaBoost, the middle image shows the result of AdaBoost with isotropic smoothing only,



(a) (b) (c)

Figure 6.1: Smoothing and tangent vectors. Within each of the three groups (a), (b), and (c) we show the classification result of Discrete AdaBoost (left), AdaBoost with isotropic smoothing (middle), and AdaBoost with tangent vectors (right). Group (a) shows the classification result with 1 weak classifier, group (b) shows the classification results with 100 weak classifiers, and group (c) shows the classification results with 300 weak classifiers. The top row illustrates the decision region: White corresponds to the region where points are classified as crosses and black to the region where points are classified as circles. The bottom row shows the response of the auxiliary function F, i.e., before thresholding with sign. Notice that group (a) shows clearly the effect of introducing isotropic smoothing (middle image) and that of introducing an anisotropic smoothing (right image). Notice also how the standard AdaBoost method suffers from overfitting, while isotropic smoothing and, in particular, tangent vector methods do not.

and the right image shows the result of AdaBoost with tangent vectors. The top row displays the decision region where white corresponds to points that are classified as crosses and black corresponds to points that are classified as circles. The bottom row shows the response of the auxiliary function F. One can immediately see that while Discrete AdaBoost suffers from overfitting, the other two methods can cope well with few samples. In particular, as this data is invariant to rotations, AdaBoost with tangent vectors obtains the best results.

In the second experiment, the synthetic data set consists of 24×24 pixels images of 4 shapes: a circle, a triangle, a star, and a square. To each shape we apply all the transformations listed in Section 6.2 and, in addition, skewness. Some samples from each data set are shown in Figure 6.2.

We lump triangles and squares into class 1 and stars and circles into class 2. Then, we train a strong classifier by changing the isotropic smoothing parameter, by enabling or not the gradient descent on weak classifiers, by incorporating or not the tangent vectors, and by changing the number of samples in the training set. The results of several combinations of these features are shown in Figures 6.3

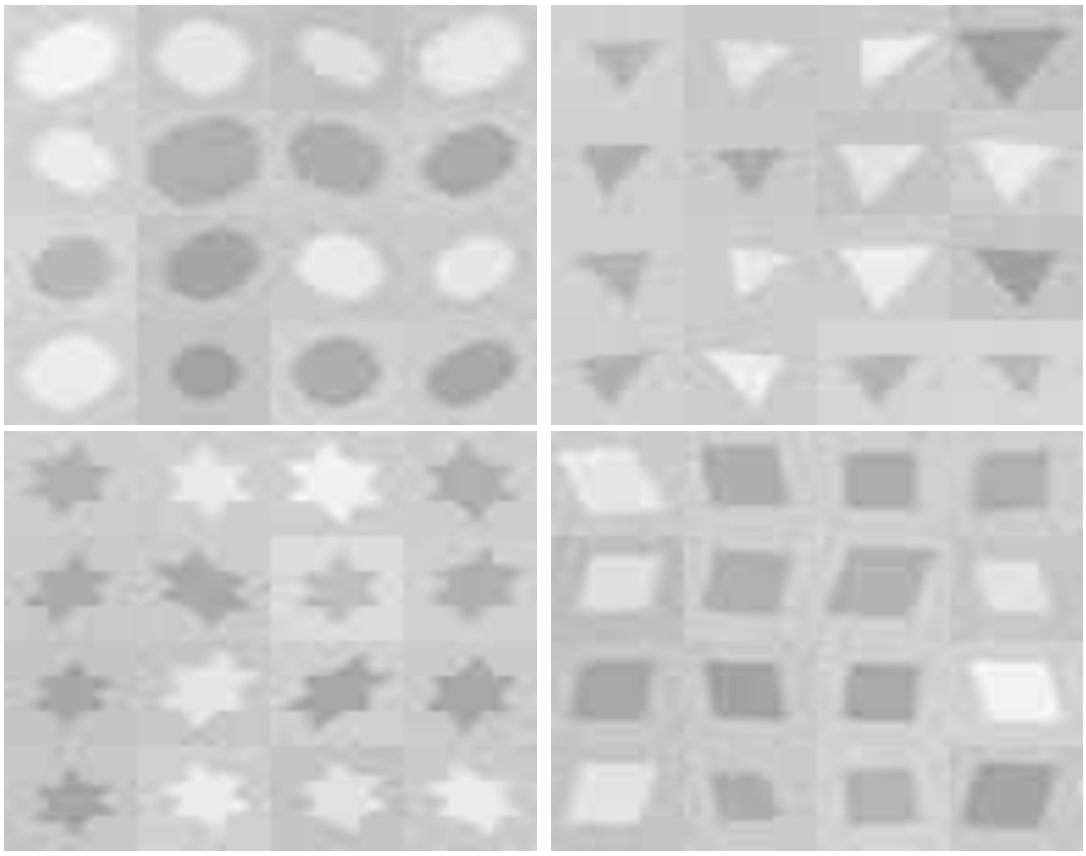


Figure 6.2: Some samples from the synthetic data set of the second experiment. Circles, triangles, stars, and squares are translated, rotated, scaled, skewed, undergo changes in contrast and brightness, and have additive Gaussian noise.

and 6.4.

Notice that in Figure 6.3 the performance at run-time improves when tangent vectors are used and even more when gradient descent is enabled. All experiments share the same data set and the same initial set of weak classifiers. Furthermore, notice how the proposed algorithm can cope well with few data samples in the training set as the performance with 25 elements yields a 10% test error. As the test error converges almost immediately when the proposed method is used, only a few weak classifiers are needed to achieve a very low test error (i.e., 8 weak classifiers for $\sim 2\%$ test error).

In Figures 6.4 we show the performance of Parzen-AdaBoost when each linear weak classifier is projected onto a Haar wavelet basis and a fixed number of components is retained. Notice that the algorithm achieves 10% test error already with 20 samples, and that on average 25 components are sufficient to capture the

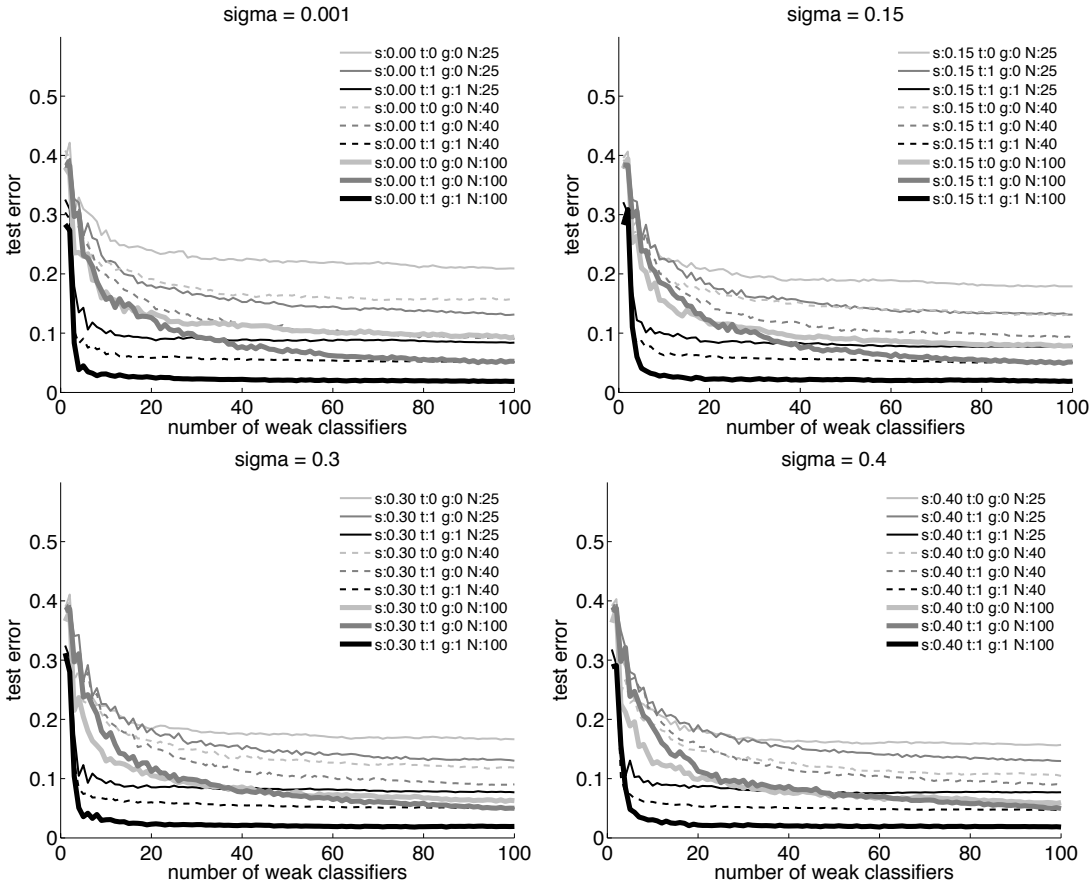


Figure 6.3: Classification results on synthetic data. The ordinate axis shows the test error of the proposed AdaBoost method for several configurations of number of samples in the training data set (N), usage of gradient descent (g), usage of the tangent vectors (t), and level of isotropic smoothing (s).

variability of the weak classifiers.

6.4.2 Real data experiments

As real data set we use the Viola-Jones faces data set which is publicly available [VJ04a]. In Figure 6.5 we show the same tests performed in the previous section. Notice that the results resemble very closely the results obtained in the case of synthetic data. The only exception is for the case of 25 samples when both gradient descent and tangent vectors are used. In this case the method does still better than Discrete AdaBoost, but worse than using only tangent vectors. As this does not happen for larger training sets, we conjecture that gradient descent

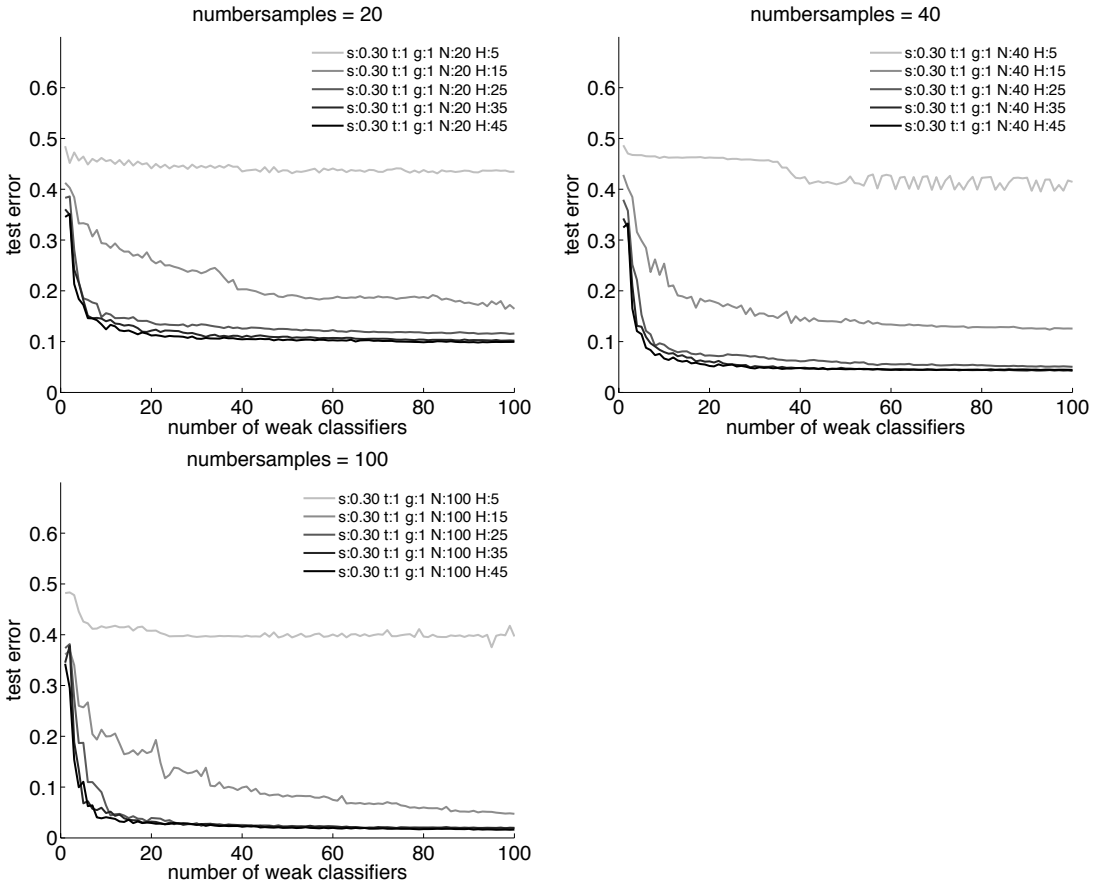


Figure 6.4: Classification results on synthetic data. The ordinate axis shows the test error of the proposed AdaBoost method with Haar wavelet projection. The plot shows how the performance changes as we increase the number of Haar wavelets of each weak classifier (H).

may overfit data on extremely small training sets.

As in the previous section, we test the performance of the proposed algorithm when the weak classifiers are projected onto a Haar wavelet basis. Figure 6.6 shows the results for 5, 15, 25, 35, 45, and 55 components in the truncated Haar wavelet series. Notice that the performance does not change visibly when more than 25 components are kept. In Figure 6.7 we show the result of training the method on $8,000$ samples (almost half of the database) and show a final comparison between the main features: optimization of the weak classifiers via gradient descent and usage of tangent vectors. Again the performance shows a consistent behavior. Not only AdaBoost based on tangent vectors and gradient

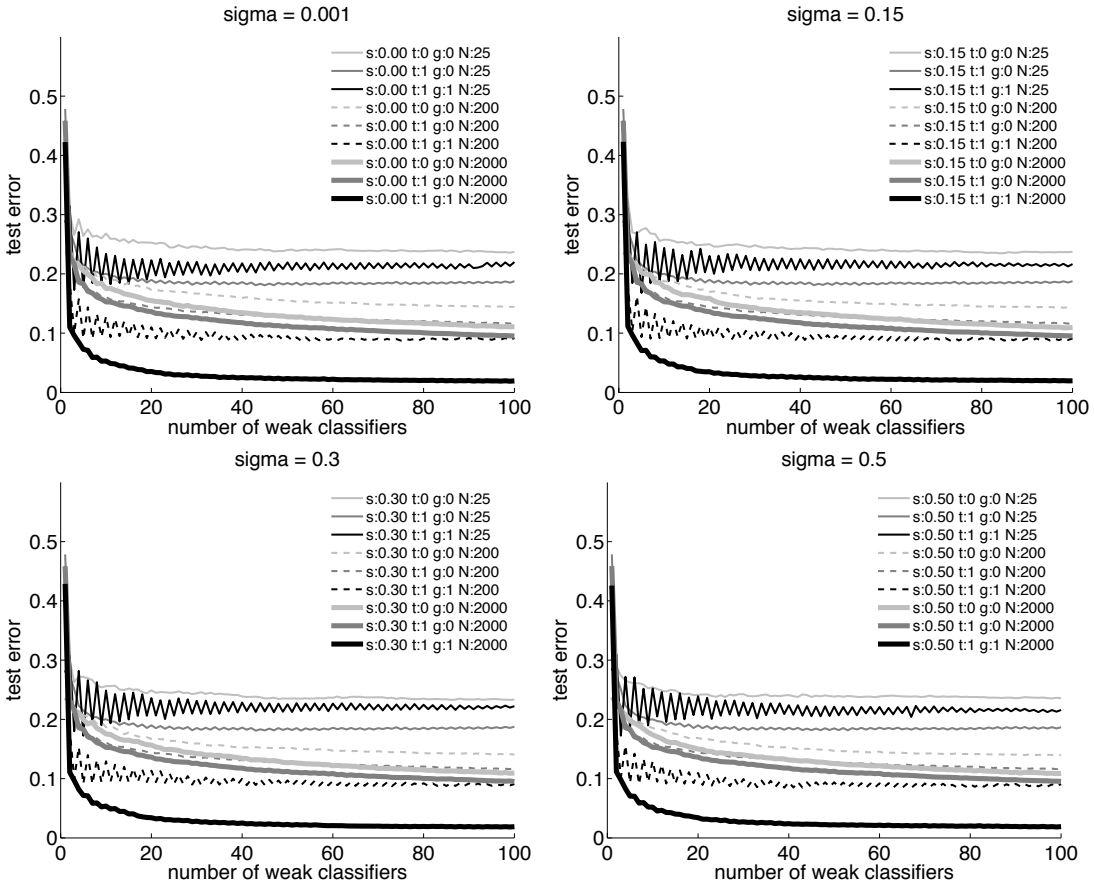


Figure 6.5: Faces data set. Performance evaluation for varying isotropic smoothing (s), gradient descent (g), tangent vectors (t), and number of samples in the training set (N).

descent achieves the best performance in these experiments by and large, but also it does so more quickly. Having quick convergence to the final test error means that fewer weak classifiers can be used, with considerable reduction of the computations at run-time.

Discussion

We have presented a novel boosting method that incorporates several features. We start by extending the traditional AdaBoost framework to Parzen's windows and then show how this can be used to incorporate invariance to geometric and photometric transformations of the data set. Furthermore, our formulation al-

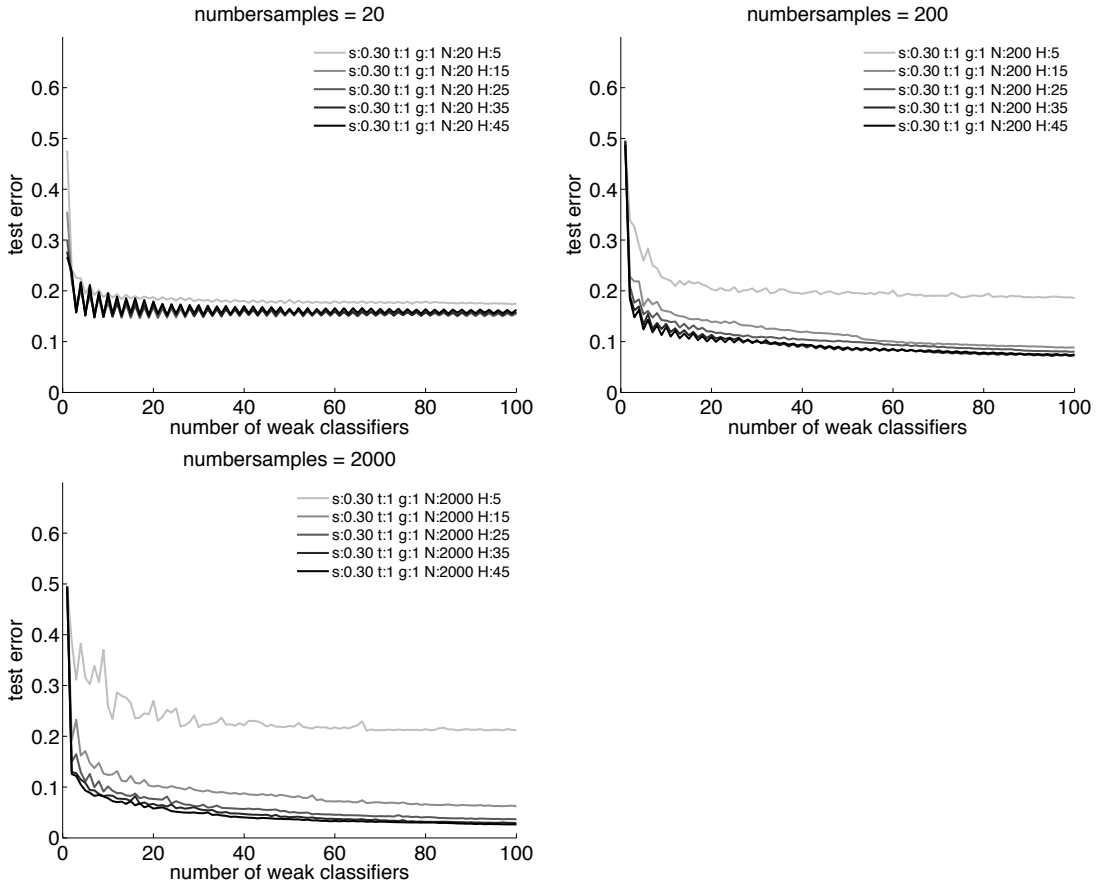


Figure 6.6: Faces data set. As in the synthetic data set, the ordinate axis shows how the performance changes as we vary the number of Haar wavelet components of each weak classifier (H).

lows the introduction of gradient descent to find optimal weak classifiers and of Haar wavelet projection to improve the computational efficiency at run-time. We demonstrate that our method has strong generalization properties on both synthetic and real data.

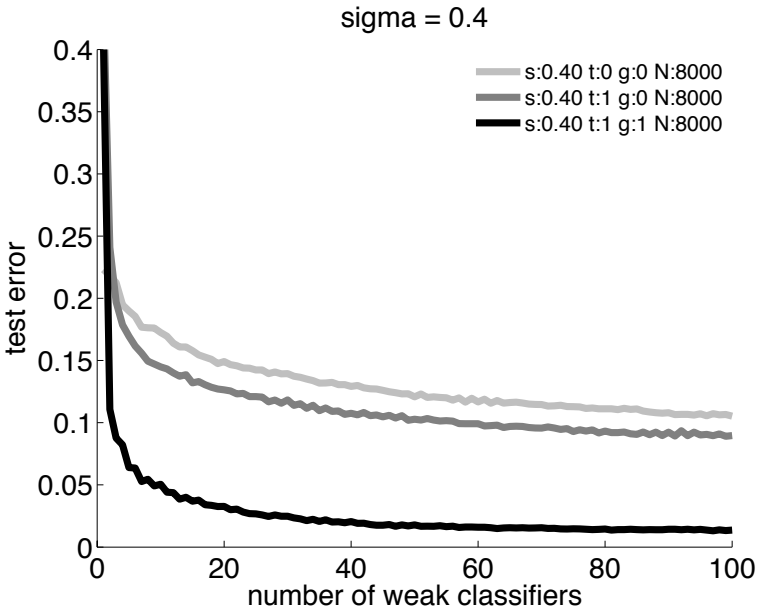


Figure 6.7: Faces data set. In this experiment we perform training on about half of the data set. The results are once again consistent with the performance in the previous cases, showing that the method scales well with the size of the training set.

CHAPTER 7

Relaxed Matching Kernels

Many visual tasks, such as visual recognition, categorization or three-dimensional reconstruction, hinge on establishing at least partial correspondence between different images affected by intrinsic variability of the underlying scene (e.g. "chair"), as well as by variability due to changes in viewpoint and illumination. In order to mitigate the effects of occlusions of line-of-sight, many methods employ a representation of the image in terms of local invariant features, as discussed in Chapters 2 through 5 of this thesis. The way local features are integrated in a global representation of the image, however, varies greatly. At one end of the spectrum are so-called "constellation models" that allow for affine transformations of feature locations [WWP00], or more general "deformable templates" that allow for more general transformations, for instance represented by a finite-dimensional thin-plate spline [Boo89]. At the other end of the spectrum are so-called "bags of features" (BoF) methods [CDD04], that discard the location of the features altogether [GD06a].

The fact that BoF methods have been so successful in visual categorization tasks may seem surprising. A possible reason is that, as we showed in Chapter 2, achieving viewpoint invariant image representations forces to discard shape information. However, this does not necessarily mean that a fully invariant representation is preferable to one which is perhaps less invariant, but more discriminative. In this context, works such as [LSP06, LS07] proposed variants of the bag-of-feature model that tries to capture part of the spatial information as well. In particular, they propose kernels for image comparison which are based on a bag-of-feature representation augmented with spatial information.

In this chapter we build upon those works and define a general family of kernels, called "relaxed matching kernels" (RMK) (Section 7.2). This family include as special cases and unifies existing approaches such as the pyramid matching kernel [GD06a], the spatial pyramid matching kernel [LSP06] as well as the proximity distribution kernel [LS07]. We study interesting properties shared by these kernels and we show that all of them can be computed efficiently. This helps understanding the difference between these approaches, and at least in one case it highlights inconsistencies in the weighting scheme and suggests a better kernel. More importantly, our approach allows us to *define new kernels*, for instance

the "graph-matching kernel" (GMK) and agglomerative-information-bottleneck kernel (AIBMK) proposed in Section 7.3.

In Section 7.4.1 we test GMK on matching graphs of generic features, such as those used in the "sketch" [GZW03], for wide-baseline correspondence. We show that, even when features are ambiguous and their identity becomes unstable due to viewpoint changes, the graph matching is robust enough to absorb much of the variability. Finally, in Sect 7.4.1 we compare various kernels on the task of object recognition on benchmark datasets such as Graz-02 and Pascal-05. We show that our kernels are very competitive with respect to state of the art [TS07, LS07]. We also show, however, that a good baseline implementation of bag-of-features is very competitive with this more advanced methods, an is capable to outperform those and previously published sate-of-the-art results in some cases.

7.1 Bag-of-features and beyond

Constructing the Bag-of-Features (BoF) representation [CDD04] of an image starts from the extraction of local image features. First, the image I is decomposed in a number of interest regions. To this end, several operators (feature detectors) are available, ranging from the selection of random patches [NJT06] to the extraction of scale or affine covariant blobs and corners [MTS04]. This results in a list $l_1, \ldots, l_N$ of feature locations (and the associated regions). Then the appearance of each region is encoded by a compact but discriminative statistic (feature descriptor). Again, several operators can be used, many of which are based on computing an histogram of the image intensities or gradients [Low04]. This results in a second list $d_1, \ldots, d_N$ of feature descriptors.

The locations $l_1, \ldots, l_N$ are then disregarded and the image is represented by the distribution of the feature descriptors $d_1, \ldots, d_N$ alone. The distribution is estimated by quantizing the descriptor space $\mathcal{F}$ and then computing an histogram[1] $h(b)$ of the occurrence of the quantized descriptors (it is also possible to avoid quantizing altogether [PD06]). The quantization $B \subset 2^{\mathcal{F}}$ may be obtained by a variety of methods, such as K-means or regular partitioning [GD06a, TS07]. By analogy with the bag-of-words model of text analysis, the quantized descriptors $b_1, \ldots, b_N \in B$ are also called *visual words* and the quantization B *visual dictionary*.

Comparing two images I^1 and I^2 is then reduced to evaluating the similarity $K(h^1, h^2)$ of the respective bag-of-features $h^k(b)$, $k = 1, 2$ representations. Recently [ZML06] has shown that the χ^2 Radial Basis Function (RBF) kernel (Section 7.2) yields particularly good performances in object categorization with

[1] We assume that histograms are normalized to one.

the advantage of being directly operable in an SVM classifier.

A problem with the dictionary approach to BoF is the choice of the resolution of the visual dictionary B. An excessively fine quantization causes features from two images to never match (overfitting), while an excessively coarse quantization yields non-discriminative histograms (bias). Grauman et al. [GD06a] proposed *Pyramid Matching Kernel* to overcome this issue. The idea is to work with a sequence of R progressively coarser dictionaries $B_0, B_1, \ldots, B_{R-1}$ and to define a similarity measure as a positive combination of the BoF similarities at the various levels. The formulation yields a proper Mercer (positive definite) kernel.

While BoF is a powerful paradigm, disregarding completely the image geometry limits the discriminative power of the representation. Several attempts have been made to extend BoF to account for geometric information. The easiest way is to append the interest point locations to the descriptors (Section 4 of [GD06a]). Lazebnik et al. [LSP06] extend this idea and introduce the *Spatial Pyramid Matching Kernel* (SPMK): They propose to use quantized pairs (l_i, d_i) of interest point location-descriptor as element of the base visual dictionary B_0. The pyramid $B_0, B_1, \ldots, B_{R-1}$ is then formed by coarsening the quantization of the location component l only. In this way, the representation captures the distribution of both the appearance and location of the interest points.

A limitation of this approach is that, since the location l is expressed in absolute coordinates, the representation is unsuitable for objects which present large variations in pose. To address this issue, Ling et al. [LS07] introduced the *Proximity Distribution Kernel* (PDK). The idea is to start from triplets (d_i, d_j, ρ_{ij}), where d_i and d_j are interest points descriptors and ρ_{ij} is their (nearest neighbors) distance. Successive relaxations merge increasing values of the ρ component (Section 7.2). Since ρ is a relative quantity, the limitation of SPMK is removed.

7.2 Relaxed matching kernels

In this section we introduce the "relaxed matching kernels" which generalize PMK, SPMK and PDK.

Construction. Let $B_0 \subset 2^{\mathcal{F}}$ a quantization of the feature space $\mathcal{F}$ (base visual dictionary). To obtain coarser quantizations B_r, we recursively merge bins $b \in B_0$ (Fig 7.1). The result of this process is an *agglomerative tree*, whose nodes are bins and parents are obtained from children by merging.[2]

[2]In practice the tree might be a forest if one stops merging before all bins are merged into one (but one can always assume that the latter is the case).

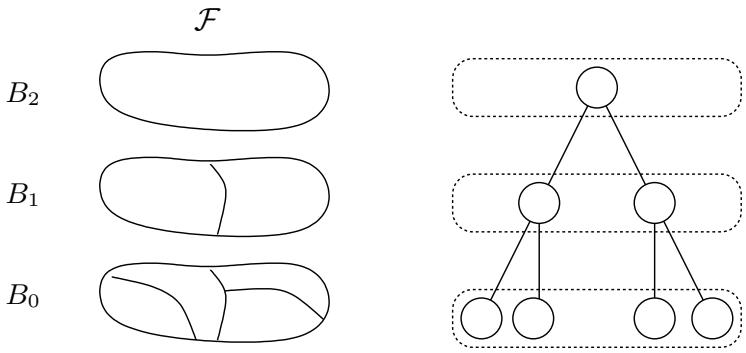


Figure 7.1: **RMK construction: agglomerative tree**. *Left.* The feature space $\mathcal{F}$ and a sequence of three relaxations B_0, B_1 and B_2. *Right.* The agglomerative tree represents the merging operations transforming a relaxation to the next. Each relaxation B_r corresponds to a cut of the tree (dotted boxes).

The base dictionary B_0 corresponds to the leaves of the agglomerative tree and the coarser dictionaries B_r correspond to tree "cuts". A *cut* (Fig. 7.1) is just a subset B_r of the tree nodes such that any leaf $b \in B_0$ is descendent of exactly one node $b' \in B_r$ of the cut. Cuts have the property of preserving the mass of the dictionary: If $h_{B_0}(b), b \in B_0$ is an histogram on the finer dictionary B_0, then its projection $h_{B_r}(b)$ on the cut B_r satisfies

$$\sum_{b \in B_0} h_{B_0}(b) = 1 = \sum_{b \in B_r} h_{B_r}(b).$$

We compare images I^k, $k = 1, 2$ by comparing histograms of features defined on corresponding cuts. Given a cut B_r, the similarity measure is given by

$$F_r = k_1(h^1_{B_r}, h^2_{B_r}) = \sum_{b \in B_r} \min\{h^1_{B_r}(b), h^2_{B_r}(b)\} \tag{7.1}$$

To make the match robust, we adopt a "multiscale" approach. We consider multiple cuts B_r at increasing *relaxation levels* $r = 0, 1 \ldots, R-1$ and define

$$K(h^1, h^2) = \sum_{r=0}^{R-1} w_r F_r, \tag{7.2}$$

where $w_r \geq 0$ is a sequence of weights that establish the relative importance of the relaxations. We define this quantity *relaxed matching kernel* (RMK).

Base kernel, Mercer's condition, and RBF. The RMK is a positive definite (p.d.) kernel [SS02] since each term $k_1(h^1_{B_r}, h^2_{B_r})$ of the summation (7.1) is

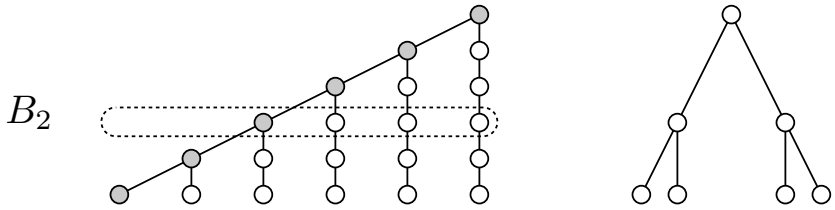


Figure 7.2: **RMK: agglomerative trees for PDK, PMK and SPMK.** *Left.* PDK relaxations merge successive values of the distance component ρ, yielding a "linear" agglomerative tree. As an illustration, we highlight the cut corresponding to relaxation B_2. PDK fails to be a proper RMK, however, as it considers only the shaded nodes and is not normalized. *Right.* PMK and SPMK relaxations are obtained by merging octaves of the scale space, yielding a "logarithmic" agglomerative tree.

p.d. [HB05] and the weights w_r are non-negative [SS02]. Interestingly, Hein et al. [HB05] provide a whole family of base kernels that can be substituted to the l_1 kernel k_1 in (7.1). This family includes the χ^2 and Hellinger's kernels

$$k_{\chi^2}(p,q) = 2\sum_i \frac{p_i q_i}{p_i + q_i}, \quad k_H(p,q) = \sum_i \sqrt{p_i q_i}.$$

All of these choices yield p.d. RMKs (another useful property is that the kernels are normalized to one, i.e. $k(p,p) = 1$).

Finally, each base kernel corresponds to a distance $d^2(p,q)$ by the formula $d^2(p,q) = 2 - 2k(p,q)$. So, for instance, $k_1(p,q)$ corresponds to $d_1^2(p,q) = \|p-q\|_1$ and the χ^2 and Hellinger's kernels correspond to

$$d^2_{\chi^2}(p,q) = \sum_i \frac{(p_i - q_i)^2}{p_i + q_i}, \quad d^2_H(p,q) = \sum_i (\sqrt{p}_i - \sqrt{q}_i)^2.$$

These distances can be used to define corresponding RBF kernels by setting $k(p,q) = \exp(-\gamma d^2(p,q))$, where $\gamma > 0$ is a tuning parameter. These kernels are also p.d. [HB05, BFC02].

This flexibility in the choice of the base kernel is interesting as, for instance, [ZML06] showed that the χ^2 RBF kernel may perform better than the l_1 kernel (on which PMK, SPMK and PDK are based) for the task of object categorization.

PMK, SPMK, and PDK. The RMK construction encompasses the approaches discussed in Set. 7.1. In PMK the feature space $\mathcal{F}$ is the set of descriptors d, B_0 is a regular partition of $\mathcal{F}$ and B_r are obtained by recursively merging

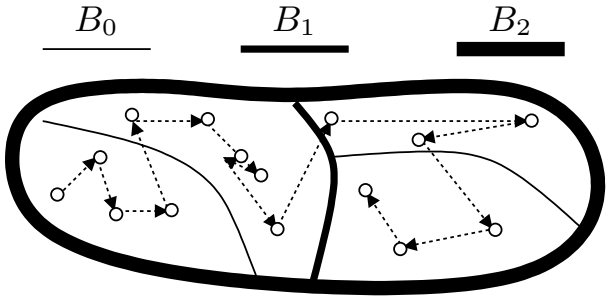


Figure 7.3: **RMK computation: feature visit order.** The figure shows the feature space $\mathcal{F}$ and the quantization B_0, B_1 and B_2 of Fig. 7.1. The dots represents the features f_i^k and the dotted arrows a possible visiting order. Notice that the visit traverses all the features of a bin $b \in B_r$ before passing to the successive bin $b' \in B_r$, for all relaxations $r = 0, 1, 2$.

such partitions, reducing by half the resolution of the quantization. The SPMK is similar, except that relaxations operate on the location component l of the features (Section 7.1). The corresponding agglomerative tree height is logarithmic in the size of the base dictionary B_0 (Fig. 7.2).

In PDK B_0 is obtained by quantizing the descriptor component d_i and d_j of the triplets (d_i, d_j, ρ_{ij}) (ρ_{ij} is already discrete). Then the successive relaxations B_r are defined by merging triplets that have distance $\rho_{ij} \leq r + 1$. Still PDK is not a proper RMK because (a) the histograms are not normalized and (b) at each level the comparison (7.1) is defined as

$$k_{\text{PDK}}(h^1_{B_r}, h^2_{B_r}) = \sum_{d_1,d_2} \min \left\{ \sum_{\rho \leq r+1} h^1_{B_0}(d_1, d_2, \rho), \sum_{\rho \leq r+1} h^2_{B_0}(d_1, d_2, \rho) \right\}$$

and misses part of the mass. Specifically, the RMK version of PDK (Fig. 7.2) yields

$$k_{\text{PDK/RMK}}(h^1_{B_r}, h^2_{B_r}) = k_{\text{PDK}}(h^1_{B_r}, h^2_{B_r}) + \sum_{d_1,d_2} \sum_{\rho > r+1} \min \left\{ h^1_{B_0}(d_1, d_2, \rho), h^2_{B_0}(d_1, d_2, \rho) \right\}.$$

Meaning of the weights. Define $W_r = \sum_{q=0}^{r} w_q$ and $f_r = F_r - F_{r-1}$, $W_{-1} =$

$F_{-1} = 0$. Then the RMK (7.2) may be rewritten as

$$K = \sum_{r=0}^{R-1} w_r F_r = \sum_{r=0}^{R-1} (W_{R-1} - W_{r-1}) f_r. \tag{7.3}$$

An interesting property of the successive relaxations, proved in Theorem 2, is that F_r is a monotonically increasing quantity (for a large choice of base kernels, including all the popular ones). Moreover, if the last relaxation level corresponds to merging the whole feature space into a single bin, since the base kernel is normalized we also have $F_R = 1$. Therefore we can interpret F_r as a cumulative distribution function and the summation (7.3) as the expected value $K = E_{f_r}[W_{R-1} - W_{r-1}]$ of the function $W_{R-1} - W_{r-1}$ of the random variable r with (discrete) density f_r. Notice that f_r assigns more mass to the relaxation levels r for which there is an abrupt increase in the matching score F_r. Since $W_{R-1} - W_{r-1}$ decays with increasing relaxation r (the weights are positive), this means that the score is large if the two image statistics match early in the relaxation sequence. In other words, the kernel is looking for the finer relaxation level for which the statistics match well.[3]

For instance the PMK and SPMK kernels have exponentially decaying integral weights of the form $W_r \propto -e^{-\lambda r}$, $\lambda > 0$ (up to a positive factor and offset). In fact, computing the differences $W_r - W_{r-1}$ yields $w_r \propto e^{-\lambda r}$ and we have

$$K_{\text{PMK}} \propto \sum_{r=0}^{R-1} (e^{-\lambda r} - e^{-\lambda R}) f_{r-1} \propto \sum_{r=0}^{R-1} e^{-\lambda r} F_r.$$

For the PDK/RMK kernel we have $w_r = 1$, $W_r = r$ and

$$K_{\text{PDK}} = \sum_{r=0}^{R-1} F_r = \sum_{r=0}^{R-1} (R - r) f_r$$

so the weights are linearly decaying.

Computation. We show next that computing an RMK it is a fast operation as it it is linear in the number of features and relaxation levels.[4]

Let f_i^k, $i = 1, \ldots, N_k$, $k = 1, 2$ be the features extracted from images I^1 and I^2 and quantized to the base level B_0. Let F_r, L_r^1, L_r^2, $r = 0, \ldots, R-1$ be three accumulators initialized to zero.

[3]This also suggests why counting the same features at multiple relaxation levels do not really introduce bias in the comparison

[4]The complexity is $O(NR)$ where $N = N_1 + N_2$ is the number of features from the two images to be compared and R is the number of relaxations. The algorithm is also space efficient as it requires only $O(N + R)$ memory.

First, we show how to calculate F_r according to the definition (7.1) for a fixed relaxation level r. To do this, we need to compare histograms $h^1_{B_r}$ and $h^2_{B_r}$ defined over bins $B_r = \{b_{r1}, \dots, b_{rM}\}$. We start by visiting all the features f^k_i that belong to the first bin b_{r1}, incrementing the value of the respective accumulators L^k_r. When there are no more features in b_{r1}, we compute $\min\{h^1_{B_r}(b_{r1}), h^2_{B_r}(b_{r2})\} = \min\{L^1_r, L^2_r\}$ as of equation (7.1), accumulate the result to F_r, set L^1_r and L^2_r to zero, and proceed to the next bin b_{r2}. When all bins $b_{rm} \in B_r$ are exhausted, F_r holds the value (7.1).

This process can be extended to work *simultaneously* for all relaxation levels $r = 0, \dots, R-1$. This is possible because bins b_{ri} at level r are fully contained in bins $b_{r+1,j}$ at level $r+1$, so visiting the features belonging to $b_{r+1,j}$ can be done by visiting the features belonging respectively to all the bins $b_{ri} \subset b_{r+1,j}$ in order, and so on recursively (Fig. 7.3). So it suffices to scan the features once (in the proper order) accumulating their mass to $L^k_1, \dots, L^k_{R-1}$. Whenever the visit crosses a bin boundary at some level r, the algorithm adds $\min\{L^1_r, L^2_r\}$ to F_r, resets L^1_r and L^2_r and moves on.[5]

7.3 Two novel RMKs

To illustrate the flexibility of the RMK construction, we introduce two new matching kernels.

Graph Matching Kernel. Graphs have been used extensively for representing and matching images. Usually a graph is constructed by connecting interest points or other features in structures such as constellations, and sketches (see for instance [FFP03, FPZ05, LZW07, FH05] and references therein). Matching graphs however is difficult due to the high instability of such structures and the combinatorial complexity of the search. Roughly speaking, three approaches are used: (i) focus on simple structures (such as small graphs, trees or stars) that enable exhaustive search [FPZ05, FH05], (ii) use statistical searching procedures (e.g. RANSAC, Swendsen-Wang sampling [LZW07]), and (iii) use approximated matching methods (e.g. spectral methods [QH06]).

Here we experiment with a loose but robust voting scheme, reminiscent of [KI02] and PDK, based on comparing interest point pairs. Consider a graph G whose nodes are interest points $l_1, \dots, l_N$ with associated descriptors $d_1, \dots, d_N$. Let $G = \{e_m, m = 1, \dots, M\}$ be the collection of edges forming the graph, where $e_m = \{l_i, l_j\}$ is an (unordered) pair of image locations. Let ρ_{ij} be the graph

[5] A further speed-up is obtained if features are pre-merged at the finer relaxation level B_0 before running the algorithm. This is especially useful for kernel such as PDK which compare pairs of interest point and may have large feature sets.

distance from l_i to l_j (i.e. the length of the shortest path connecting l_i to l_j). We construct an RMK by considering triplets (d_i, d_j, ρ_{ij}) as the base features. We quantize the descriptor space as in PDK or SPMK (ρ has already a discrete structure) to obtain the base dictionary B_0. We then define the successive relaxation levels $B_1, \ldots, B_{R-1}$ by merging values of the index ρ, using the linear scheme of PDK/RMK. So the kernel has the form

$$K(G^1, G^2) = \sum_{r=0}^{R-1} w_r \sum_{(d_i,d_j,\rho)\in B_r} k(h^1_{B_r}(d_i, d_j, \rho), h^2_{B_r}(d_i, d_j, \rho)).$$

In the following we refer to this kernel as *Graph Matching Kernel* (GMK). GMK checks for the presence of edges between images features, as specified by the graph structure. Despite this fact, in the limit when all nodes have unique identifiers, $K(G^1, G^2)$ assumes its maximum value $\sum_{r=0}^{R-1} w_r$ if, and only if, $G^1 \equiv G^2$.

Agglomerative Information Bottleneck Kernel. As a second example of RMK, we introduce a kernel similar in spirit to PMK. We start by a basic quantization B_0 of the feature descriptors d_i (we discard the locations l_i). Then we define the successive relaxations B_r by iteratively merging bins of the base dictionary B_0. However, instead of guiding the merges based on descriptor similarity (as PMK does), we use Agglomerative Information Bottleneck (AIB, [ST99]) to obtain a sequence of binary merges. AIB produces a sequence of relaxations B_r so that the information $I(d, c)$ between the feature descriptor $d \in B_r$ (regarded as a random variable) and the class label c is maximally preserved. We use $w_r = I_r$ to penalize coarser relaxations which correspond to uninformative dictionaries, where I_r is the residual information $I(d, c)$ at the relaxation level r. We call this *Agglomerative Information Bottleneck Matching Kernel* (AIBMK).

7.4 Experiments

7.4.1 GMKs to match unstable graphs

The first experiment (Fig. 7.4) illustrates graph matching by GMK. Given an image I^k, we construct a graph as follows: we run Canny's edge detector on the image, we extract straight edge segments, and we complete the graph by constrained Delaunay triangulation. We then extract SIFT keys at the node locations (fixed window size and orientation) using software from [VF08] and we create a dictionary of only sixteen visual words (such a vocabulary is not very distinctive but quite invariant). This yields graphs G^1 and G^2 from the two images.

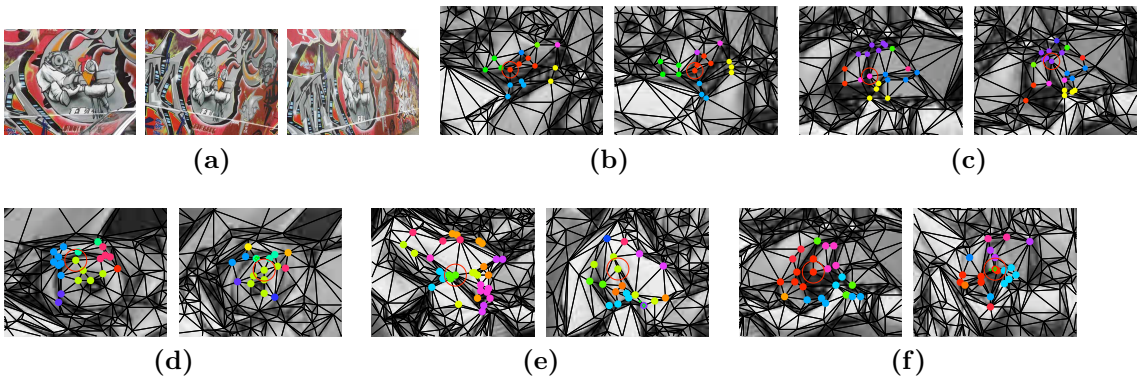
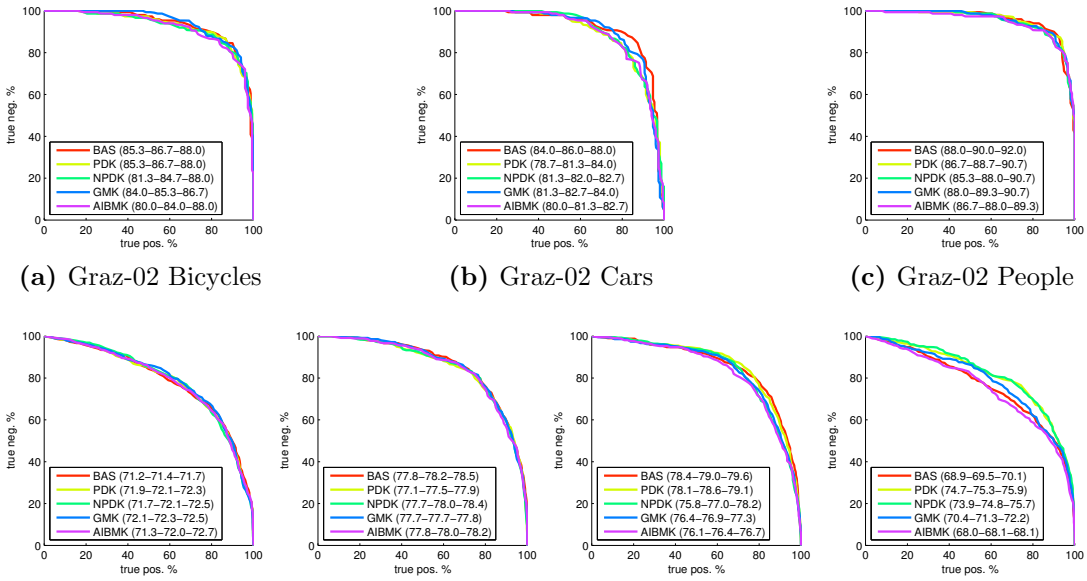


Figure 7.4: **GMK: robustness evaluation.** (a) A few images from [MTS04]. The data consists of six image: a frontal view five other views from, 20 to 60 degrees of slant. Here we construct a graph by downsampling the images by half, computing a Canny edge map and running constrained Delaunay triangulation. We then compute SIFT features at nodes (fixed orientation and window size of 20 pixels). This construction is *not* affine invariant and the resulting graph is highly unstable. We make the node labels as invariant as possible by choosing a small dictionary size (64 bins). We then match each subgraph $S^1(l_i)$ in the frontal view to similar graphs $S^k(l_j)$ in the other views (we do not try to remove ambiguous matches). Using the ground truth homography, we record the graph distance from the center of the best matching subgraph to the actual reprojection. (b) a match at graph distance 0 from the 20^o views pair. (c) A match with graph distance 1 – the overlap is still very good. (d)-(f) two matches at 30^o. (f) A match at 50^o. Up to 20^o of slant 83% of the match are within graph distance 2. At 30^o this number reduces to 57%. After that the deformation of the descriptors is excessive and matching becomes unreliable.

We then select a location l_i in the first image and extract a subgraph $S^1(l_i) \subset G^1$, defined as the union of l_i with its neighbors at (graph) distance not greater than $T = 2$. Then we try to match $S^1(l_i)$ to $S^2(l_j)$ for all similarly constructed subgraphs in the second image. Notice the large variation in the structure of the graphs being matched, due both to instability of the construction of the image graphs G^k and the selection of the subgraphs S^k. We evaluate quantitatively how many subgraphs can be successfully matched in a test sequence from [MTS04]. This data is devised to evaluate affine invariant descriptors; here we show that RMK is robust enough to match unstable interest points graphs.



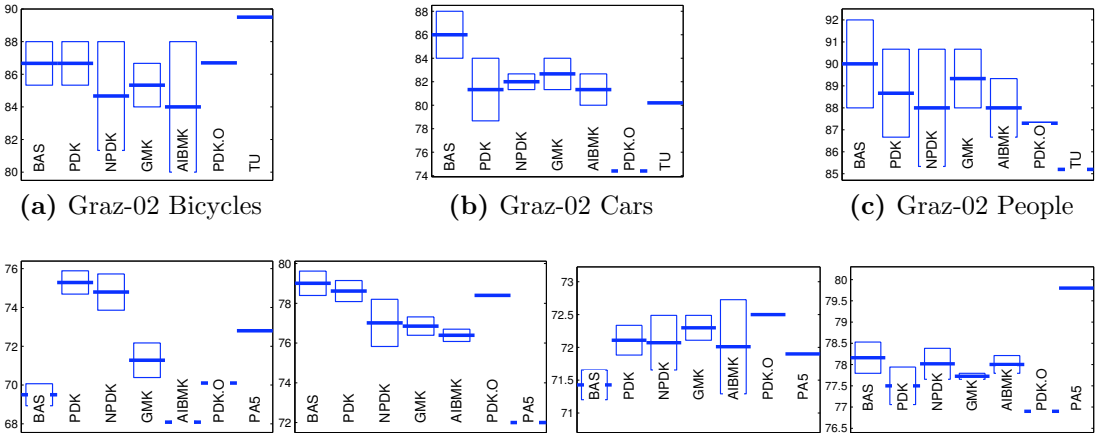
(a) Graz-02 Bicycles **(b)** Graz-02 Cars **(c)** Graz-02 People

(d) Pascal-02 People **(e)** Pascal-02 Motorbikes **(f)** Pascal-02 Cars **(g)** Pascal-02 Bicycles

Figure 7.5: **ROC curves for Pascal-05 and Graz-02.** We compare the average ROCs obtained in several runs of the various algorithms (we average ROC curves along lines from the origin; in this way the curve passes by the average equal-error-rate point).

7.4.2 RMKs for object recognition

We evaluate GMK, AIBMK, PDK, PDK/RMK in object recognition experiments on the Graz-02 and Pascal-05 datasets (mainly for the sake of comparison with previous related approaches). We also compare the methods against the baseline BoF as described by [ZML06], which we summarize next. Each image is normalized so that the largest side measures 640 pixels. Then the Harris and Laplace operators are used to extract multiscale interest points using publicly available code from [Dor05]. We remove features of scale below 2.5 pixels (we also remove duplicate features due to a bug in the software). As in [ZML06], we fix the orientation of the patches to a nominal value (i.e. the interest points are not rotationally invariant). At each interest point we compute a SIFT descriptor. The visual vocabulary is formed by running k-means with $k = 200$ (Lloyd algorithm) for each category independently, and then joining the dictionaries. Bag of features are compared by the χ^2 RBF kernel, which performs better on average. The GMK, AIBMK, PDK, PDK/RMK also use the same χ^2 basis kernel and the RBF transformation. We use an SVM in all experiments. The parameter C of



(a) Graz-02 Bicycles **(b)** Graz-02 Cars **(c)** Graz-02 People

(d) Pascal-05 Bicycles **(e)** Pascal-05 Cars **(f)** Pascal-05 People **(g)** Pascal-05 Motorbikes

Figure 7.6: **Equal Error Rates for Pascal-05 and Graz-02.** We report the maximum, minimum and average EER for each algorithm in multiple runs (as the construction of the dictionary is randomized). The variability, especially in the smaller Graz-02 dataset, is relatively large. This makes it difficult to compare directly to previous work, which do not report this information. Here PDK.O refers to [LS07], TU to [TS07] and PA5 to Pascal-05 winner. All algorithms, whether they use spatial information or not, are very close. The baseline algorithm performs as well or better in many of the cases, and it is very close to the best algorithm in the others. Makes exception Pascal-05 bikes, where we were able to obtain the better performance by method exploiting the spatial structure.

the SVM [SS02] is learned by 10-fold cross validation. The graph used in GMK is computed by Delaunay triangulation of the points (we do not extract edges).

For Graz-02 we use the same training and testing sets of [LS07, TS07]. For Pascal-05 we use the training and validation sets from the challenge as training data and the test-2 (difficult) test set as testing data. Results are compared in Table 7.6 against [LS07, TS07] and the winner of Pascal-05 VOC competition. ROC curves are reported in Fig. 7.5.

Our kernels are competitive, outperforming previous state of the art in four of the seven categories. Our implementation of PDK outperforms the original paper [LS07] in all but one cases, perhaps due to the fact that we use the χ^2 and RBF combination. We also compare favorably to [TS07] and the Pascal-05 winner.

We should note, however, than in most cases the advantage of one method on another is small (see for instance GR-bicycles). In particular, the baseline

algorithm performs in practice as well and in some case better than these more sophisticated kernels and [TS07] (which uses dense features and a large vocabulary as opposed to sparse feature and a small vocabulary).

Discussion

We have introduced RMK as a generalization of popular kernels for image categorizations. The formulation defines a large space of possible useful kernels, and suggests modifications and improvements to the current ones. We also have introduced a novel interpretation of the kernel weights and showed the monotonicity property of the relaxed similarity scores (7.1). These observations transfer directly to previous method as well.

We have introduced two new examples of RMKs: the GMK and AIBMK kernels. GMK have been demonstrated successfully for matching graphs of features in a wide-baseline matching experiment. We also have tested our kernels on object categorization on Pascal-05 and Graz-02. However, we also noticed that a baseline BoF formulation is often as competitive, which, we hope, will stimulate a useful debate in the community.

7.A Proofs

We study the parametric family of kernels among histograms given by $K(p,q) = \sum_i k_{\alpha|\beta}(p_i, q_i)$, where [HB05]

$$k_{\alpha|\beta} = \frac{p_i + q_i}{2} - \frac{1}{2Z}\left[\left(\frac{p_i^\alpha + q_i^\alpha}{2}\right)^{\frac{1}{\alpha}} - \left(\frac{p_i^\beta + q_i^\beta}{2}\right)^{\frac{1}{\beta}}\right] \tag{7.4}$$

where $\alpha \geq 1$ and $\beta \in [-\infty, -1] \cup [\frac{1}{2}, \alpha]$ and the normalization constant Z is equal to $2^{-\frac{1}{\alpha}} - 2^{-\frac{1}{\beta}}$ if $\beta > 0$ and to $2^{-\frac{1}{\alpha}}$ if $\beta < 0$. l_1, Hellinger's and χ^2 kernels are obtained for (α, β) equal to $(\infty, 1)$, $(1, \frac{1}{2})$ and $(1, -1)$ respectively. In the following we restrict to the case $\beta \leq 1$ (we verified by simulation that these results do not always hold if $\beta > 1$).

Lemma 5. *Let $x_1, x_2, y_1, y_2 \in \mathbb{R}_+$ be non negative numbers and let $k = k_{\alpha|\beta}$ as defined above. Moreover, let $\beta \leq 1$. Then*

$$k(x_1 + x_2, y_1 + y_2) \geq k(x_1, y_1) + k(x_2, y_2)$$

Proof. Let $f_\alpha(x_i, y_i) = (x_i^\alpha + y_i^\alpha)^{1/\alpha}$. Since $\alpha \geq 1$, by Minkowsky's inequality[6]

[6]I.e. by th l^α-triangle inequality applied to vectors (x_1, y_1) and (x_2, y_2).

$f_\alpha(x_1 + x_2, y_1 + y_2) \leq f_\alpha(x_1, y_1) + f_\alpha(x_2, y_2)$. Minkowsky's inequality reverses when the exponent is smaller than 1, for which $f_\beta(x_1 + x_2, y_1 + y_2) \leq f_\beta(x_1, y_1) + f_\beta(x_2, y_2)$. Substituting back in (7.4) we obtain the desired inequality. □

Theorem 2 (Monotonicity of the kernel)**.** *Let $p, q \in \mathbb{R}^n_+$ be non-negative real vectors. Let W be a stochastic matrix (i.e. $W \in \mathbb{R}^{m\times n}_+$, $\mathbf{1}^\top W = \mathbf{1}^\top$). Let $K(p, q)$ defined as above, with $\beta \leq 1$. Then*

$$K(Wp, Wq) \geq K(p, q).$$

Proof. We have

$$K(Wp, Wq) = \sum_i K\left(\sum_j w_{ij}p_j, \sum_j w_{ij}q_j\right)$$

Applying iteratively the lemma $n - 1$ times yields

$$K(Wp, Wq) \geq \sum_i \sum_j K(w_{ij}p_j, w_{ij}q_j).$$

But K is homogeneous (i.e. $K(cx, cy) = cK(x, y)$), so

$$K(Wp, Wq) \geq \sum_j \sum_i w_{ij}K(p_j, q_j) = K(p, q).$$

□

CHAPTER 8

Fast Data Clustering and Quick Shift

Clustering is a fundamental primitive of many vision algorithms. An application relevant to us is the construction of the so called "visual dictionaries" used in bag-of-features image representations for the recognition of object categories (refer to Chapter 7). Recall that a visual dictionary is a partition of the space of local image features into a number of "visual words". Ideally, visual words should be as informative as the original features and contain less irrelevant details than them. Perhaps surprisingly, however, clustering methods that are commonly used for the design of such dictionaries are *not* tailored to this end. A typical example is k-means, whose purpose is to approximate the feature (data) space with a small number k of prototypes: These are apparently unrelated to the task of characterizing object categories.

A possible reason why clustering methods such as k-means may work in this context is that the distribution of local image features often reflects invariance properties of the data. In particular, local image features may distribute in modes, with each mode capturing a prototypical visual structure and some of its irrelevant variations. Therefore, by representing such modes, k-means may generate useful visual words.

The ability of k-means of capturing the data modes is limited by the fact that the algorithm uses a parametric model of the cluster shapes. If more fine grained information needs to be recovered, different approaches can be used. For computer vision applications, the most notable example is probably mean shift. Mean shift [FH75, Che95, CM02b] is a popular non-parametric clustering algorithm based on the idea of associating each data point to a mode of the underlying probability density function. This simple criterion has appealing advantages compared to other traditional clustering techniques: The structure of the clusters may be rather arbitrary and the number of clusters does not need to be known in advance.

Mean shift is not the only "mode seeking" clustering algorithm. Other examples include earlier graph-based methods [KNF76] and, more recently, *medoid shift* [SKK07]. Unlike mean shift, medoid shift extends easily to non-Euclidean spaces. In fact, mean shift is essentially a gradient optimization algorithm [Che95,

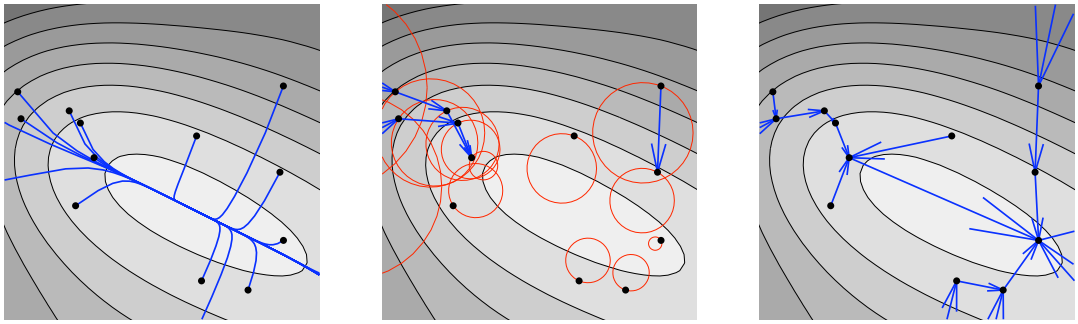


Figure 8.1: **Mode seeking algorithms.** Comparison of different mode seeking algorithms (Section 8.1) on a toy problem. The black dots represent (some of) the data points $x_i \in \mathcal{X} \subset \mathbb{R}^2$ and the intensity of the image is proportional to the Parzen density estimate $P(x)$. **Left.** Mean shift moves the points uphill towards the mode approximately following the gradient. **Middle.** Medoid shift approximates mean shift trajectories by connecting data points. For reason explained in the text and in Fig. 8.2, medoid shifts are constrained to connect points comprised in the red circles. This disconnects portions of the space where the data is sparse, and can be alleviated (but not solved) by iterating the procedure (Fig. 8.2). **Right.** Quick shift (Section 8.2) seeks the energy modes by connecting nearest neighbors at higher energy levels, trading-off mode over- and under-fragmentation.

CM02b, YL07] and the gradient cannot be defined unless the data space is endowed with a Hilbert space structure. While there have been attempts to generalize mean shift to non-linear manifolds by working on the tangent space [SM06], medoid shift adopts a simpler and more general strategy, which does not require the explicit computation of the gradient in the first place. Moreover, the algorithm is non-iterative and there is no need for a stopping heuristic. Its biggest disadvantage is its computational complexity [SKK07]. Depending on the implementation, medoid shift requires between $O(dN^2 + N^3)$ and $O(dN^2 + N^{2.38})$ operations to cluster N points, where d is the dimensionality of the data. On the other hand, mean shift is only $O(dN^2T)$, where T is the number of iterations of the algorithm, and clever implementations yield $dT \ll N$.

In this chapter we show that the computational complexity of Euclidean medoid shift is only $O(dN^2)$ (with a small constant), which makes it faster (not slower!) than mean shift (Section 8.2). We then generalize this result to a large family of non-Euclidean distances by using kernel methods [Sch01], showing that in this case the complexity is bounded by the effective dimensionality of the kernel space (Section 8.2). Working with kernels has other advantages: First, it

extends to mean shift (Section 8.3); second, it gives an explicit interpretation of non-Euclidean medoid shift; third, it suggests why such generalized mode seeking algorithms skirt the *curse of dimensionality*, despite estimating a density in complex spaces (Section 8.3). In summary, we show that kernels extends mode seeking algorithms to non-Euclidean spaces in a simple, general and efficient way.

Can we conclude that medoid shift should replace mean shift? Unfortunately, not. We show that the weak point of medoid shift is its inability to identify consistently all the modes of the density (Section 8.1). This fact was addressed implicitly by [SKK07] who reiterate medoid shift on a simplified dataset (similar to [Car06]). However, this compromises the non-iterative nature of medoid shift and changes the underlying density function (which may be undesirable). Moreover, we show that this fix does not always work (Fig. 8.2).

We address this issue in two ways. First, we propose using medoid shift to simplify the data and initialize the more accurate mean shift algorithm (Section 8.4.2 and Section 8.4.3). Second, we propose an alternative mode seeking algorithm that can trade off mode over- and under-fragmentation (Section 8.2). This algorithm, related to [KNF76], is particularly simple and fast, yields surprisingly good segmentations, and returns a one parameter family of segmentations where model selection can be applied.

We demonstrate these algorithms on three tasks (Section 8.4): Clustering on a manifold (Section 8.4.1), image segmentation (Section 8.4.2), and clustering image signatures for automatic object categorization (Section 8.4.3). The relative advantages and disadvantages of the various algorithms are discussed.

8.1 Mode seeking

Given N data points $x_1, \ldots, x_N \in \mathcal{X} = \mathbb{R}^d$, a *mode seeking* clustering algorithm conceptually starts by computing the *Parzen density estimate*

$$P(x) = \frac{1}{N} \sum_{i=1}^{N} k(x - x_i), \quad x \in \mathbb{R}^d \tag{8.1}$$

where $k(x)$ can be a Gaussian or other window.[1] Then each point x_i is moved towards a mode of $P(x)$ evolving the trajectory $y_i(t)$, $t > 0$ uphill, starting from $y_i(0) = x_i$ and following the gradient $\nabla P(y_i(t))$. All the points that converge to the same mode form a cluster.

[1]The term "kernel" is also used in the literature. Here we use the term "window" to avoid confusion with the kernels introduced in Section 8.2.

A mode seeking algorithm needs (i) a numerical scheme to evolve the trajectories $y_i(t)$, (ii) a halting rule to decide when to stop the evolution and (iii) a clustering rule to merge the trajectory end-points. Next, we discuss two algorithms of this family.

Mean Shift. Mean shift [FH75, CM02b] is based on an efficient rule to evolve the trajectories $y_i(t)$ when the window $k(x)$ can be written as $\psi(\|x\|_2^2)$ for a convex function $\psi(z)$ (for instance the Gaussian window has $\psi(z) \propto \exp(-z)$). The idea is to bound the window from below by the quadric $k(z') \geq k(z) + (\|z'\|_2^2 - \|z\|_2^2)\dot{\psi}(\|z\|_2^2)$. Substituting in (8.1) yields

$$P(y') \geq P(y) + \frac{1}{N}\sum_{j=1}^{N}(\|y' - x_j\|_2^2 - \|y - x_j\|_2^2)\dot{\psi}(\|y - x_j\|_2^2), \tag{8.2}$$

and maximizing this lower bound at $y = y_i(t)$ yields the mean-shift update rule

$$y_i(t+1) = \operatorname*{argmax}_y \frac{1}{N}\sum_{j=1}^{N}\|y - x_j\|_2^2\dot{\psi}(\|y_i(t) - x_j\|_2^2) = \frac{\sum_{j=1}^{N}\dot{\psi}(\|y_i(t) - x_j\|_2^2)x_j}{\sum_{j=1}^{N}\dot{\psi}(\|y_i(t) - x_j\|_2^2)}. \tag{8.3}$$

If the profile $\psi(z)$ is monotonically decreasing, then $P(y_i(t)) < P(y_i(t+1))$ at each step and the algorithm converges in the limit (since E is bounded [CM02b]). The complexity is $O(dN^2T)$, where d is the dimensionality of the data space and T is the number of iterations. The behavior of the algorithm is illustrated in Fig. 8.1.

Medoid Shift. Medoid shift [SKK07] is a modification of mean shift in which the trajectories $y_i(t)$ are constrained to pass through the points x_i, $i = 1, \ldots, N$. The advantage of medoid shift are: (i) only one step $y_i(1)$, $i = 1, \ldots, N$ has to be computed for each point x_i (because $y_i(t+1) = y_{y_i(t)}(1)$), (ii) there is no need for a stopping/merging heuristic (as these conditions are met exactly), and (iii) the data space $\mathcal{X}$ may be non-Euclidean (since to maximize (8.4) there is no need to compute derivatives). Eventually, points are linked by steps into a forest, with clusters corresponding to trees. The algorithm is illustrated in Fig. 8.1.

According to [SKK07], the main drawback of medoid shift is speed. In fact, maximizing (8.3) restricted to the dataset amounts to calculating

$$y_i(1) = \operatorname*{argmax}_{y \in \{x_1, \ldots, x_N\}} \frac{1}{N}\sum_{j=1}^{N}\mathsf{d}^2(y, x_j)\dot{\phi}(\mathsf{d}^2(x_j, x_i)) \tag{8.4}$$

where $\mathsf{d}^2(x, y) = \|x - y\|_2^2$ in the Euclidean case. A basic implementation requires $O(N^3 + dN^2)$ operations, assuming $O(d)$ operations to evaluate $\mathsf{d}^2(x, y)$. However,

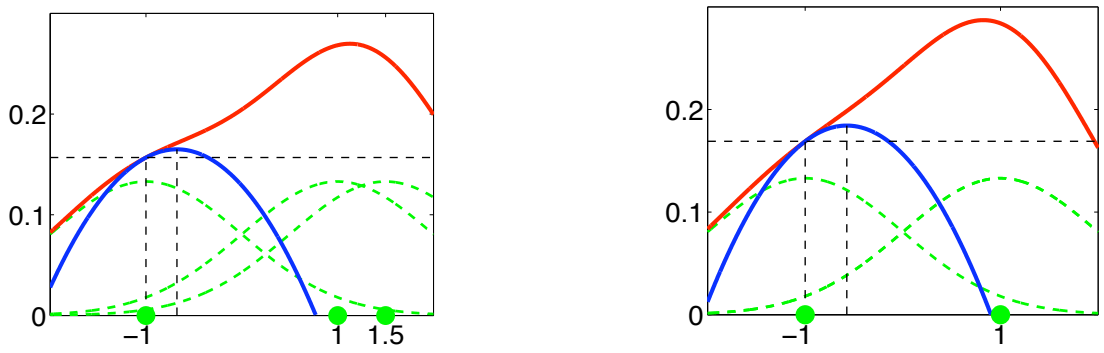


Figure 8.2: **Medoid shift over-fragmentation. Left.** We apply medoid shift to cluster points $-1, +1, +1/2 \in \mathbb{R}$ using a Gaussian window of variance $\sigma^2 = 1$ (dashed green lines). The density $P(x)$ (red curve; Section 8.1) has a single mode, but medoid shift fails to move the point -1 towards the mode (i.e. $y_{-1}(1) = -1$). The reason is that the quadratic lower bound (8.2) (blue curve) is larger at -1 than it is at $+1$ or $+1/2$. Notice that mean shift would have moved -1 towards the mode by a small, but finite amount, eventually extracting the single mode. **Right.** The problem is not solved even if medoid shift is reiterated [SKK07] on the two modes -1 and $+1$ (where $+1$ has double mass), even if the density $P(x)$ *does* become blurrier [Car06, SKK07].

by defining matrices $D_{kj} = \mathsf{d}^2(x_k, x_j)$ and $F_{ki} = \dot{\phi}(D_{ik})/N$, we can rewrite (8.4) as

$$y_i(1) = \operatorname*{argmax}_{k=1,\dots,N} \sum_{j=1}^{N} D_{kj} F_{ji} = \operatorname*{argmax}_{k=1,\dots,N} e_k^\top D F e_i \tag{8.5}$$

where e_i denotes the i-th element of the canonical basis.[2] As noted in [SKK07], $O(N^{2.38})$ operations are sufficient by using the fastest matrix multiplication algorithm available. Unfortunately the hidden constant of this algorithm is too large to be practical (see [Knu98], pag. 501). Thus a realistic estimate of the time required is more pessimistic than what suggested by the asymptotic estimate $O(dN^2 + N^{2.38})$.

Here we note that a more delicate issue with medoid shift is that it may fail to properly identify the modes of the density $P(x)$. This is illustrated in Fig. 8.2, where medoid shift fails to cluster three real points $-1, +1$ and $+1/2$, finding two modes -1 and $+1$ instead of one. To overcome this problem, [SKK07] applies medoid shift iteratively on the modes (in the example -1 and $+1$). However this solution is not completely satisfactory because (i) the underlying model $P(x)$

[2] For instance $e_2 = \begin{bmatrix} 0 & 1 & 0 & \dots 0 \end{bmatrix}^\top$.

is changed (similarly to *blurry* mean shift [FH75, Che95]) and (ii) the strategy does not work in all cases (for instance, in Fig. 8.2 points -1 and $+1$ still fail to converge to a single mode).

Finally, consider the interpretation of medoid shift. When $\mathcal{X}$ is a Hilbert space, medoid (and mean) shift follow approximately the gradient of the density $P(x)$ (by maximizing the lower bound (8.3)). The gradient itself depends crucially on the inner product defined on $\mathcal{X}$ (which encodes the cost of moving along each direction [SYM07]). When $\mathcal{X}$ is *not* an Hilbert space, the gradient cannot be defined, but the term $\mathsf{d}^2(x, y)$ in (8.4) has a similar direction-weighing effect. In later sections we will make this connection more explicit.

8.2 Fast clustering

Faster Euclidean Medoid Shift. We show that the complexity of Euclidean medoid shift is only $O(dN^2)$ (with a small constant) instead of $O(dN^2 + N^{2.38})$ (with a large constant) [SKK07]. Let $X = \begin{bmatrix} x_1 & \dots & x_N \end{bmatrix}$ be the data matrix. Let $n = (X^\top \odot X^\top)\mathbf{1}$ be the vector of the squared norms of the data, where $\mathbf{1}$ denotes the vector of all ones and $\odot$ the Hadamard (component wise) matrix product. Then we have

$$D = \mathbf{1}n^\top + n\mathbf{1}^\top - 2X^\top X, \qquad DF = n(\mathbf{1}^\top F) + \mathbf{1}(n^\top F) - 2X^\top(XF). \tag{8.6}$$

The term $\mathbf{1}(n^\top F)$ has constant columns and is irrelevant to the maximization (8.5). Therefore, we need to compute

$$DF \propto n(\mathbf{1}^\top F) - 2X^\top(XF), \quad n = (X^\top \odot X^\top)\mathbf{1} = (I \odot X^\top X)\mathbf{1} \tag{8.7}$$

where I is the identity matrix.[3] It is now easy to check that each matrix product in (8.7) requires $O(dN^2)$ operations only.

Kernel Medoid Shift. An advantage of medoid shift is the possibility of computing (8.4) for distances $\mathsf{d}^2(x, y)$ other than the Euclidean one [SKK07]. The decomposition (8.6) can still be carried out if the distance $\mathsf{d}^2(x, y)$ can be expressed as $K(x, x) + K(y, y) - 2K(x, y)$ for an appropriate positive definite (p.d.)

[3] And we used the fact that $(I \odot AB)\mathbf{1} = (B^\top \odot A)\mathbf{1}$.

kernel[4] K [Sch01]. Then we have $D = \mathbf{1}n^\top + n\mathbf{1}^\top - 2K$, and

$$DF \propto n(\mathbf{1}^\top F) - 2KF, \quad n = (I \odot K)\mathbf{1}.$$

Unfortunately, the multiplication KF is still $O(N^{2.38})$. However we can search for a low-rank decomposition $G^\top G$ of K (we assume, without loss of generality, that K is centered[5]). If G is a decomposition of rank d, then

$$DF \propto n(\mathbf{1}^\top F) - 2G(G^\top F), \quad n = (I \odot G^\top G)\mathbf{1} = (G^\top \odot G^\top)\mathbf{1}$$

can still be computed in $O(dN^2)$ operations. The cost of decomposing K is typically around $O(d^2N)$ [FS01, BJ02]. See Fig. 8.3 for a basic implementation.

Quick Shift. In order to seek the mode of the density $P(x)$, it is not necessary to use the gradient or the quadratic lower bound (8.2). Here we propose *quick shift*, which simply moves each point x_i to the nearest neighbor for which there is an increment of the density $P(x)$. In formulas,

$$y_i(1) = \operatorname*{argmin}_{j:P_j > P_i} D_{ij}, \quad P_i = \frac{1}{N}\sum_{j=1}^{N} \phi(D_{ij}). \tag{8.8}$$

Quick shift has four advantages: (i) simplicity; (ii) speed ($O(dN^2)$ with a small constant); (iii) generality (the nature of D is irrelevant); (iv) a tuning parameter to trade off under- and over-fragmentation of the modes. The latter is obtained because there is no *a-priori* upper bound on the length D_{ij} of the shifts $y_i(0) \to y_i(1)$. In fact, the algorithm connects all the points into a single tree. Modes are then recovered by breaking the branches of the tree that are longer than a threshold τ. Searching τ amounts to performing model selection and balances under- and over-fragmentation of the modes. The algorithm is illustrated in Fig. 8.1.

Quick shift is related to the classic algorithm from [KNF76]. In fact, we can rewrite (8.8) as

$$y_i(1) = \operatorname*{argmax}_{j=1,\dots,N} \frac{\operatorname{sign}(P_j - P_i)}{D_{ij}}, \quad \text{and compare it to } y_i(1) = \operatorname*{argmax}_{j:d(x_j,x_i)<\tau} \frac{P_j - P_i}{D_{ij}} \tag{8.9}$$

[4]The kernel K should not be confused with the Parzen window $k(z)$ appearing in (8.1). In the literature, it is common to refer to the Parzen window as "kernel", but in most cases it has rather different mathematical properties than the kernel K we consider here. An exception is when the window is Gaussian, in which cases $k(\mathsf{d}^2(x,y))$ is a p.d. kernel. In this case, we point out an interesting interpretation of mean shift as a local optimization algorithm that, starting from each data point, searches for the pre-image of the global data average computed in kernel space. This explains the striking similarity of the mean shift update Eq. (8.3) and Eq. (18.22) of [SS02].

[5]K is *centered* if $K\mathbf{1} = 0$. If this is not the case, we can replace K by $K' = HKH$, where $H = I - \mathbf{1}\mathbf{1}^\top/N$ is the so-called *centering matrix*. This operation translates the origin of the kernel space, but does not change the corresponding distance.

as given by [KNF76]. Notice that $(P_j - P_i)/D_{ij}$ is a numerical approximation of the gradient of E in the direction $x_j - x_i$. The crucial difference is that maximizing the gradient approximation must be done in a neighborhood of each point defined *a-priori* by the choice of the parameter τ. Thus model selection in [KNF76] requires running the algorithm multiple times, one for each value of τ. In contrast, quick shift returns at once the solutions for all possible values of τ, making model selection much more efficient.

8.3 Cluster refinement

In the previous section we introduced fast kernel medoid shift as an accelerated version of non-Euclidean medoid shift. Since medoid shift may over-fragment modes, quick shift was then proposed as a method to control under- and over-fragmentation by the choice of a parameter τ. No algorithm, however, guarantees the same accuracy of the slower mean shift.

It is then natural to ask whether mean shift could be extended to work in a non-Euclidean setting. [SKK07] cites the problem of defining the mean as the major obstacle to this idea. [SM06] addresses this issue by defining mean shift vectors on the tangent space of a non-linear manifold, but no proof of convergence is given, and the applicability is limited by the fact that the data space needs to have a manifold structure known analytically.

A simple solution to this problem is to extend kernel medoid to a corresponding kernel mean shift procedure. Let $K(\cdot,\cdot)$ be a p.d. kernel on the data space $\mathcal{X}$. Then $K(x,\cdot)$ is an element of the so called *reproducing kernel Hilbert space* $\mathcal{H}$ [SS02], whose inner product is defined by letting $\langle K(x,\cdot), K(y,\cdot)\rangle_{\mathcal{H}} = K(x,y)$. Points $x \in \mathbb{R}^d$ are then identified with elements $K(x,\cdot)$ of the Hilbert space. Given this identification, we can write $\langle \cdot, x\rangle_{\mathcal{H}}$ for $\langle \cdot, K(x,\cdot)\rangle_{\mathcal{H}}$.

Kernel mean shift computes a "density[6]" on $\mathcal{H}$

$$P(y) = \frac{1}{N}\sum_{j=1}^{N} k(\mathsf{d}^2_{\mathcal{H}}(y, x_j)), \qquad y \in \mathcal{H} \tag{8.10}$$

where $\mathsf{d}^2_{\mathcal{H}}(x_j, y) = \langle y, y\rangle_{\mathcal{H}} + \langle x_j, x_j\rangle_{\mathcal{H}} - 2\langle y, x_j\rangle_{\mathcal{H}}$. Notice that $y \in \mathcal{H}$, unlike standard mean shift, does not belong necessarily to the data space $\mathcal{X}$ (up to the identification $x \equiv K(x,\cdot)$). However, if $k(z)$ is monotonically decreasing, then maximizing w.r.t. y can be restricted to the linear subspace $\operatorname{span}_{\mathcal{H}} X = \operatorname{span}_{\mathcal{H}}\{x_1, \ldots, x_n\} \subset \mathcal{H}$ (if not, the orthogonal projection of y onto that space decreases simultaneously all terms $\mathsf{d}^2_{\mathcal{H}}(x_j, y)$).

[6] The interpretation is discussed later.

(Kernel) Mean Shift

```
function Z = meanshift(G, sigma)

[d,N]  = size(G) ;
oN     = ones(N,1) ;
od     = ones(d,1) ;
n      = (G'.*G')*od ;

Z = G ;
T = 100 ;
for t=1:T
  m = (Z'.*Z')*od ;
  D = m*oN' + oN*n' - 2*(Z'*G) ;
  F = - exp(- .5 * D' / sigma^2) ;
  Y = F ./ (oN * (oN'*F)) ;
  Z = G*Y ;
end
```

(Kernel) Medoid Shift

```
function map = medoidshift(G, sigma)

[d,N]  = size(G) ;
oN     = ones(N,1) ;
od     = ones(d,1) ;
n      = (G'.*G')*od ;

D = n*oN' + oN*n' - 2*(G'*G) ;
F = - exp(- .5 * D' / sigma^2) ;
Q = n * (oN'*F) - 2 * G' * (G*F) ;

[drop,map]  = max(Q) ;
```

Figure 8.3: **Kernel mean and medoid shift algorithms.** We show basic MATLAB implementations of two of the proposed algorithms. Here $K = G^\top G$ is a low-rank decomposition $G \in \mathbb{R}^{d \times N}$ of the (centered) kernel matrix and sigma is the (isotropic) standard deviation of the Gaussian Parzen window. Both algorithms are $O(dN^2)$ (for a fixed number of iterations of mean shift), reduce to their Euclidean equivalents by setting $G \equiv X$ and $Z \equiv Y$, and can be easily modified to use the full kernel matrix K rather than a decomposition $G^\top G$ (but the complexity grows to $O(N^3)$).

Therefore, we can express all calculations relative to $\operatorname{span}_{\mathcal{H}} X$. In particular, if $K_{ij} = K(x_i, x_j)$ is the kernel matrix, we have $\mathsf{d}^2_{\mathcal{H}}(x_j, y) = y^\top K y + e_j^\top K e_j - 2 e_j^\top K y$ where e_j is the j-th vector of the canonical basis and y is a vector of N coefficients. As in standard mean shift, the shifts are obtained by maximizing the lower bound

$$y_i(t+1) = \operatorname*{argmax}_{y \in \mathbb{R}^N} \sum_{j=1}^{N} (y^\top K y + e_j^\top K e_j - 2 e_j^\top K y) \dot{\phi}(\mathsf{d}^2_{\mathcal{H}}(x_j, y_i(t))).$$

Deriving w.r.t. y and setting to zero yields the update equation

$$y_i(t+1) = \frac{1}{\mathbf{1}^\top F e_i} (F e_i), \quad F_{ji} = \dot{\phi}(D_{ij}), \quad D_{ij} = \mathsf{d}^2_{\mathcal{H}}(y_i(t), x_j). \tag{8.11}$$

Low-rank approximation. Similarly to medoid shift, we can accelerate the algorithm by using a low-rank decomposition $K = G^\top G$ of the (centered) kernel matrix. It is useful to switch to matrix notation for all the quantities. Let $Y = \begin{bmatrix} y_1, & \dots & y_M \end{bmatrix}$ be the trajectory matrix and define $Z = GY$ the reduced

coordinates.[7] The distance matrix D can be written compactly as

$$D = m\mathbf{1}^\top + \mathbf{1}n^\top - 2Y^\top K = m\mathbf{1}^\top + \mathbf{1}n^\top - 2Z^\top G;$$

where

$$m = (Y^\top \odot Y^\top K)\mathbf{1} = (Z^\top \odot Z^\top)\mathbf{1}, \quad n = (I \odot K)\mathbf{1} = (G^\top \odot G^\top)\mathbf{1}.$$

At each iteration D is calculated in $O(dN^2)$ operations. Then $F = \dot{\phi}(D^\top)/N$ is evaluated component-wise. Finally the trajectories Y (or equivalently Z) are updated by

$$Y \leftarrow F\,\mathrm{diag}(F\mathbf{1})^{-1}, \qquad Z \leftarrow GY.$$

in $O(N^2)$ operations. Notice that, by setting $G \equiv X$ and $Z \equiv Y$ in these equations, we obtain Euclidean mean shift back. See Fig. 8.3 for a basic implementation.

Interpretation, Regularization and Scaling. In Euclidean mean shift the function $P(x)$ is a non-parametric estimate of a probability density. Does the same interpretation hold in kernel space? For any fixed data set of size N, we can restrict our attention to the subspace space $\mathrm{span}_{\mathcal{H}} X \subset \mathcal{H}$ and interpret $P(x)$ as a probability density on this finite-dimensional space. Unfortunately, the number of dimensions of this space may be as large as the number of data points N, which makes the Parzen density estimate $P(x)$ inconsistent (in the sense that the variance does not converge to zero as $N \to \infty$). So how do we make sense of kernel mean shift? The idea is to use the fact that most of the dimensions of $\mathrm{span}_{\mathcal{H}} X$ are often unimportant. Formally, consider the eigen-decomposition $K = V\Sigma V^\top = G^\top G$, $G = \Sigma^{\frac{1}{2}} V^\top$ of the (centered) kernel matrix K. Assume $\Sigma^{\frac{1}{2}} = N\,\mathrm{diag}(\sigma_1, \ldots, \sigma_N)$, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_N$. According to this decomposition, vectors $x, y \in \mathrm{span}_{\mathcal{H}} X$ can be identified with their coordinates $g, z \in \mathbb{R}^N$ so that $\langle x, y\rangle_{\mathcal{H}} = \langle g, z\rangle$. Moreover the data matrix $G = \begin{bmatrix} g_1 & \ldots & g_n \end{bmatrix}$ has null mean[8] and covariance $GG^\top/N = \Sigma/N = \sigma_1^2\,\mathrm{diag}(\lambda_1^2, \ldots, \lambda_N^2)$. If λ_i decay fast, the effective dimension of the data can be much smaller than N.

The simplest way to regularize the Parzen estimate is therefore to discard the dimensions above some index d (which also improves efficiency). Another option is to blur the coordinates z by adding a small Gaussian noise η of isotropic standard deviation ϵ, obtaining a regularized variable $z' = z + \eta$. The components of z with smaller variance are "washed out" by the noise, and we can obtain a consistent estimator of z' by using the regularized Parzen estimate $\sum_{i=1}^N (g_\epsilon * k)(z_i)$ (the same idea is implicitly used, for instance, in kernel Fisher discriminant analysis [MRW99], where the covariance matrix computed in kernel space is regularized

[7] Similarly, the data matrix X has reduced coordinates equal to G.

[8] Because K is assumed to be centered, so that $\mathbf{1}^\top G^\top (G\mathbf{1}) = \mathbf{1}^\top K \mathbf{1} = 0$.

by the addition of $\epsilon^2 I$). This suggests that using a kernel with sufficient isotropic smoothing may be sufficient.

Finally, we note that, due to the different scalings $\lambda_1, \ldots, \lambda_N$ of the linear dimensions, it might be preferable to use an adapted Parzen window, which retains the same proportions [Sai02]. This, combined with the regularization ϵ, suggests us to scale each axis of the kernel by $\sqrt{\sigma^2 \lambda_i^2 + \epsilon^2}$.[9]

[9]So far we disregarded the normalization constant of the Parzen window $k(x)$ as it was irrelevant for our purposes. If, however, windows $k_\sigma(x)$ of variable width σ are used [CRM01], then the relative weights of the windows become important. Recall that in the d dimensional Euclidean case one has $k_\sigma(0)/k_{\sigma'}(0) = (\sigma'/\sigma)^d$. In kernel space therefore one would have

$$\frac{k_\sigma(0)}{k_{\sigma'}(0)} = \sqrt{\prod_{i=1}^{N} \frac{\sigma'^2 \lambda_i^2 + \epsilon^2}{\sigma^2 \lambda_i^2 + \epsilon^2}}.$$

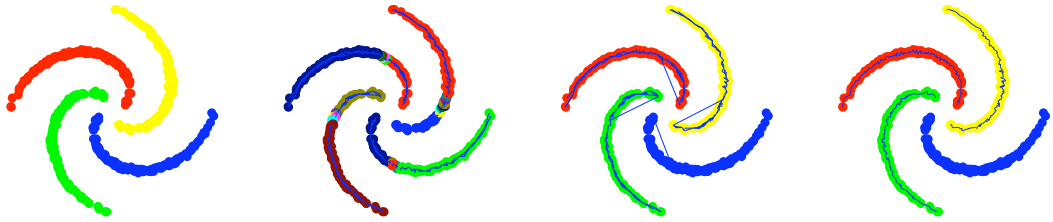


Figure 8.4: **Clustering on a manifold.** By using kernel ISOMAP we can apply kernel mean and medoid shift to cluster points on a manifold. For the sake of illustration, we reproduce an example from [SKK07]. From left to right: Kernel mean shift (7.8s), *non-iterated* kernel medoid shift (0.18s), iterated kernel medoid shift (0.48s), quick shift (0.12s). We project the kernel space to three dimensions $d = 3$ as the residual dimensions are irrelevant. All algorithms but non-iterated medoid shift segment the modes successfully. Compared to [SKK07], medoid shift has complexity $O(dN^2)$, (with a small constant and $d = 3 \ll N$) instead of $O(N^3)$ (small constant) or $O(N^{2.38})$ (large constant)

8.4 Applications

8.4.1 Clustering on manifolds

[SKK07] applies medoid shift to cluster data on manifolds, based on the distance matrix D calculated by ISOMAP. If the kernel matrix $K = HDH'/2$, $H = I - \frac{1}{N}\mathbf{1}\mathbf{1}^\top$ is p.d., we can apply directly kernel mean or medoid shift to the same problem. If not, we can use the technique from [CC06] to regularize the estimate and enforce this property. In Fig. 8.4 this idea is used to compare kernel mean shift, kernel medoid shift and quick shift in a simple test case.

8.4.2 Image segmentation

Image segmentation is a typical test case for mode seeking algorithms [CM02b, PD07, SKK07]. Usually mode seeking is applied to this task by clustering data $\{(p, f(p)), p \in \Omega\}$, where $p \in \Omega$ are the image pixels and $f(p)$ their color coordinates (we use the same color space of [CM02b]).

As in [CM02b], we apply mean shift to segment the image into super-pixels (mean shift variants can be used to obtain directly full segmentations [Car06, PD07, YL07]). We compare the speed and segmentation quality obtained by using mean shift, medoid shift, and quick shift (see Fig. 8.5 for further details).

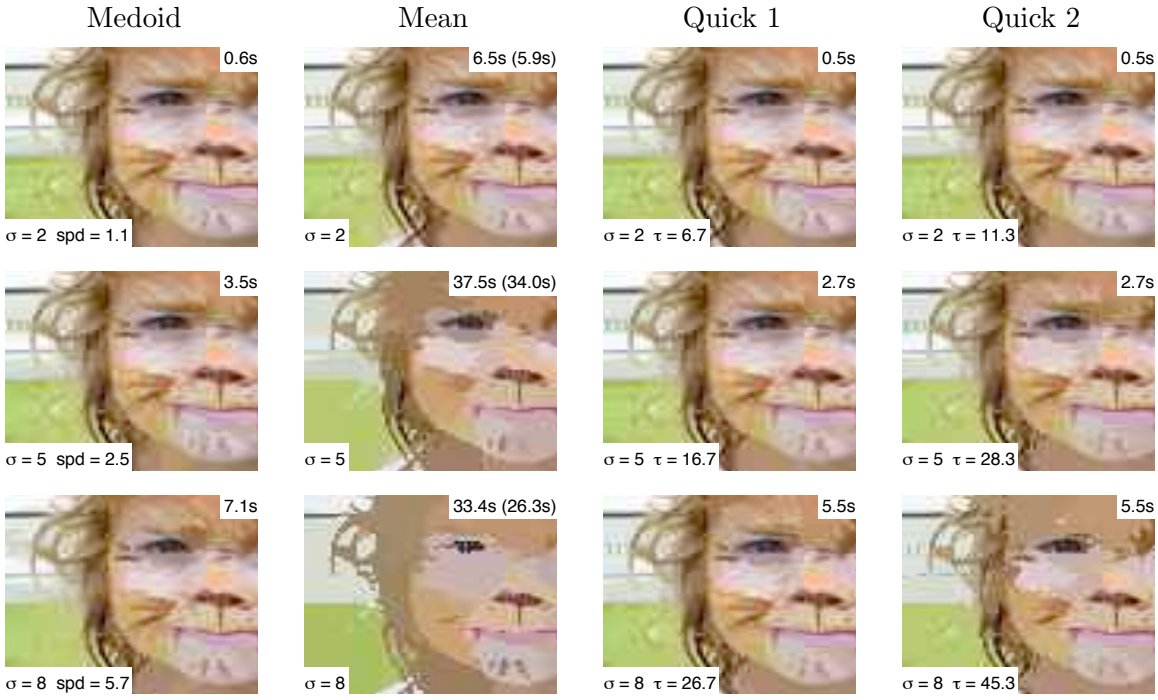


Figure 8.5: **Image segmentation.** We compare different mode seeking techniques for segmenting an image (for clarity we show only a detail). We report the computation time in seconds (top-right corner of each figure). In order to better appreciate the intrinsic efficiency advantages of each method, we use comparable vanilla implementations of the algorithms (in practice, one could use heuristics and advanced approximation techniques [YDG03] to significantly accelerate the computation). We use a Gaussian kernel of isotropic standard deviation σ in the spatial domain and use only one optimization: We approximate the support of the Gaussian window by a disk of radius 3σ (in the spatial domain) which results in a sparse matrix F. Therefore the computational effort increases with σ (top to bottom). The results are discussed in the text.

Mean shift is equivalent to [CM02b] and can be considered a reference to evaluate the other segmentations. Non-iterative medoid shift (first column) overfragments significantly (see also Fig. 8.2), which in [SKK07] is addressed by reiterating the algorithm. However, since our implementation is only $O(dN^2)$, medoid shift has at least the advantage of being much faster than mean shift, and can be used to speed up the latter. In Fig. 8.5 we compare the time required to run mean shift from scratch and from the modes found by medoid shift. We report the speedup (as the number of modes found by medoid shift over the number of pixels), the computation time of medoid+mean shift and, in brackets, the com-

putation time of the mean shift part only. Interestingly, the efficiency increases for larger σ, so that the overall computation time actually *decreases* when σ is large enough.

Finally, we show the result of quick shift segmentation (last two columns) for increasing values of the regularization parameter τ. Notice that quick shift is run only once to get both segmentations (Section 8.2) and that the algorithm is in practice much faster than the other two, while still producing reasonable super-pixels.

8.4.3 Clustering bag-of-features

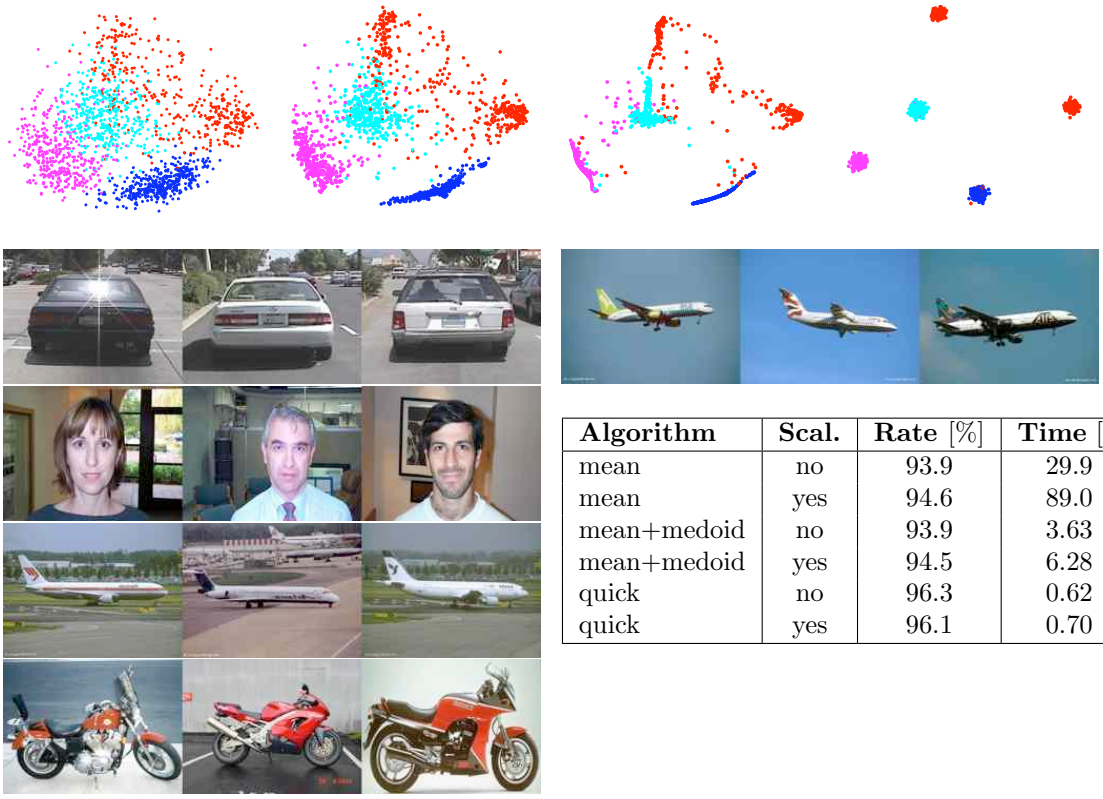
The interesting work [HB05] introduces a large family of positive definite kernels for probability measures which includes many of the popular metrics: χ^2 kernel, Hellinger's kernel, Kullback-Leibler kernel and l^1 kernel. Leveraging on these ideas, we can use kernel mean shift to cluster *probability measures*, and in particular histograms, such as the ones arising in bag-of-features [CDD04] or similar representations. In the rest of the section we experiment with the χ^2 kernel

$$K_{\chi^2}(x, y) = 2 \sum_{b=1}^{B} \frac{x_b y_b}{x_b + y_b}$$

where x and y are histograms of B bins.

Inspired by [GD06b], we attempt to automatically infer the object categories of Caltech-4 in a completely unsupervised setting. We select at random 1600 images from the categories bike, airplanes, cars and faces. Instead of the more sophisticated representation of [GD06b], we compute a basic bag-of-feature image representation as suggested by [ZML06]: We extract multiscale Harris and DoG interest points (of fixed orientation; see [ZML06] and ref. therein) and calculate SIFT descriptors [Low07], obtaining about 10^3 features per image. We then generate a vocabulary of 400 visual words by clustering a random selection of such descriptors by using k-means. For each image, we compute a bag-of-feature histogram x by counting the number of occurrences of each visual word in that image. Finally, we use the χ^2 kernel to generate the kernel matrix, that we feed to our clustering algorithms.

In Fig. 8.6 we compare kernel mean shift, kernel mean shift initialized by medoid shift, and quick shift. The problem we solve is considerably harder than [GD06b], since in our case the number of clusters (categories) is unknown. All algorithms discover five (rather than four) categories (Fig. 8.6), but the result is quite reasonable since the category airplanes contains two distinct and visually quite different populations (grounded and airborne airplanes). Moreover,



Algorithm	Scal.	Rate [%]	Time [s]
mean	no	93.9	29.9
mean	yes	94.6	89.0
mean+medoid	no	93.9	3.63
mean+medoid	yes	94.5	6.28
quick	no	96.3	0.62
quick	yes	96.1	0.70

Figure 8.6: **Automatic visual categorization**. We use kernel mean shift to cluster bag-of-features image descriptors of 1600 images from Caltech-4 (four visual categories: airplanes, motorbikes, faces, cars). **Top.** From left to right, iterations of kernel mean shift on the bag-of-features signatures. We plot the first two dimensions of the rank-reduced kernel space (z vectors) and color the points based on the ground truth labels. In the rightmost panel the data converged to five points, but we artificially added random jitter to visualize the composition of the clusters. **Bottom.** Samples from the five clusters found (notice that airplane are divided in two categories). We also report the clustering quality, as the percentage of correct labels compared to the ground truth (we merge the two airplanes categories into one), and the execution time. We use basic implementations of the algorithms, although several optimizations are possible.

compared to [GD06b] we do not try to explicitly separate an object from its background, but we use a simple holistic representation of each image.

The execution time of the algorithms (Fig. 8.6) is very different. Mean shift is relatively slow, at least in our simple implementation, and its speed greatly improves when we use medoid shift to initialize it. However, consistently with our image segmentation experiments, quick shift is much faster.

We also report the quality of the learned clusters (after manually merging the two airplane subcategories) as the percentage of correct labels. Our algorithm performs better than [GD06b], that uses spectral clustering and reports 94% accuracy on selected prototypes and as low as 85% when all the data are considered; our accuracy in the latter case is at least 94%. We also study rescaling as proposed in Section 8.3, showing that it (marginally) improves the results of mean/medoid shift, but makes the convergence slower. Interestingly, however, the best performing algorithm (not to mention the fastest) is quick shift.

Discussion

In this chapter we exploited kernels to extend mean shift and other mode seeking algorithms to a non-Euclidean setting. This also clarifies issues of regularization and data scaling when complex spaces are considered. In this context, we showed how to derive a very efficient version of the recently introduced medoid shift algorithm, whose complexity is *lower* than mean shift. Unfortunately, we also showed that medoid shift often results in over-fragmented clusters. Therefore, we proposed to use medoid shift to initialize mean shift, yielding a clustering algorithm which is both efficient and accurate.

We also introduced quick shift, which can balance under- and over-fragmentation of the clusters by the choice of a real parameter. We showed that, in practice, this algorithm is very competitive, resulting in good (and sometimes better) segmentations compared to mean shift, at a fraction of the computation time.

CHAPTER 9

Summary of Findings

In this thesis we have discussed methods for exploiting invariance of visual data for the solution of foundational vision problems such as object and category recognition and scene matching. We divided such methods into two broad families: the construction of invariant representations and learning with invariance constraints. In this chapter we schematically summarize our findings.

Invariant representations.

- We have found that general viewpoint invariant features exist, if one consider, along with geometry, the photometric properties of a three dimensional scene. We have also found that such general invariants cannot be shape-discriminative, but that discriminating shape is possible through reconstruction. We have found experimentally that affine invariant features fail at scene singularities (3-D corners) where a more general invariant can be successful, suggesting that looking beyond planar patches may be worthwhile (Chapter 2)

- We have found that, while occlusions require viewpoint invariant features to be local, the discriminative power of local features can be significantly improved by optimizing their support during matching. We have validated experimentally this idea by demonstrating applications to supervised and unsupervised detection of objects and the validation of putative feature matches (Chapter 3).

- We have seen that complex ground truth is necessary in order to rationally evaluate local viewpoint invariant features. We have found that synthetic images can be used to obtain the required data. We have shown that, by using open source software and freely available 3-D models, the simulations of the physical processes underlying the generation of such images can be carried at a level of granularity which is much finer than the one assumed by the features object of our study. We have also introduced a methodology that, along with our data, enables to assess the impact of complex visual phenomena (non linear deformations, parallax, occlusions, etc.) on the performance of local viewpoint invariant features (Chapter 4).

- We have found that joint data alignment can be used to learn invariant representations through canonization. We have found that formulating alignment as the problem of simplifying the data, as previously proposed, is ill-posed and may yield degenerate solutions. We have seen that, instead, the problem is well-posed if one searches for a simple *representation* of the data by enforcing the reconstruction property. We have seen that this casts joint data alignment as a lossy compression problem. We have discovered that entropy (in the Shannon's sense) may not be optimal for the task of alignment and proposed novel measures of complexity that are better adapted to this task. We have experimentally validated our formulation on different problems, including the one of aligning a large collection of natural image patches (Chapter 5).

Learning with invariance constraints.

- We have found that data invariance can be incorporated effectively and efficiently in AdaBoost by smoothing the weak classifiers along the invariance directions. We have seen that such regularized weak classifiers effectively capture the data invariance and are also quite convenient for training. The are two advantages: (i) by learning weak classifiers by gradient descent we can significantly reduce the memory requirements compared to conventional implementations of AdaBoost and (ii) such weak classifiers are more expressive than basic Haar wavelets (or similar elementary classifiers), so that fewer of them are needed. We have also seen that the resulting strong classifier can be projected back onto Haar wavelets for efficient evaluation (Chapter 6).

- We have seen that recent positive definite kernels for image comparison, based on the idea of re-organizing the feature space into a pyramid, can be captured and generalized by the idea of relaxed matching kernels. We have seen that there is great freedom in the construction of such kernels, ranging from the choice of the base kernel to the way the pyramid is formed. Nevertheless, we have seen that all kernels can be computed efficiently by the same algorithm and that they share a few useful properties. We have also introduced a novel interpretation of the weights used for the construction of such kernels which sheds light on their interpretation. We have introduced two novel relaxed matching kernels and compared them on standard benchmark data with previous kernels. We have noticed that, despite some of these kernels incorporate explicitly spatial information, a flat representation such as bag-of-features is often very competitive. This suggests that more work is needed to exploit the full potential of geometric information in such approaches (Chapter 7).

- We have found that the medoid shift clustering algorithm can be significantly accelerated by exploiting the Hilbert structure of the data space. We have found that kernels can be used to generalized mean shift to operate in non-Euclidean spaces, along with medoid shift. We have seen that medoid shift is more efficient than mean shift with our innovations, but tends to significantly over fragment the data modes. We have introduced a novel algorithm, quick shift, that can be used to overcome this limitation, yielding high quality clusters, while at the same time being much faster than both mean shift and medoid shift (Chapter 8).

Other contributions.

- We have contributed a robust filtering framework, KALMANSAC, and applied it to the task of on-line structure from motion. We have seen that this technique enables handling very high outlier rates (much higher than conventional robust filters) being still suitable for on-line estimation. KALMANSAC has been demonstrated on both synthetic and real structure-from-motion data (Appendix A).

- We have studied the singularities that emerge in structure from forward motion, showing that these are caused by the interaction of feature reprojection and unbounded depth estimates. We have shown that, by enforcing that no depth may become arbitrarily small, the singularities can be removed (Appendix B).

- As a service to the community, we have made available several open source reference-quality implementations of fundamental algorithms. Such contributions have been well received by the community (Applendix C).

APPENDIX A

Robust Filtering for Structure From Motion

Structure From Motion (SFM) is a mature area of computer vision where significant success has been attained during the last decade: We now have consumer products [2D3] that can estimate 3-D camera pose and point-wise structure of the scene from a collection of images in a fully automatic fashion. Robust statistical [SWB07] inference plays a crucial role in the practical implementation of most SFM systems, since the correspondence mechanisms are often based on low-level assumptions that are violated in practice. In particular, RANSAC [FB81a], along with its many variants, has become the method of choice, owing to its ability to operate in the presence of a large proportion of "outliers[1]." By contrast, vision has so far failed to materialize as a reliable sensory modality in real-time control applications, where data has to be processed in a causal fashion[2] as part of a closed-loop system. We attribute part of this failure to the lack of availability of suitable robust inference techniques that can be applied in causal data processing (there are notable exceptions, e.g. [Nis03a]). Note that batch-processing based SFM algorithms, together with the associated techniques for handling outliers, cannot be directly applied on these problems as they introduce destabilizing delays in the feedback loop [Kai80]. On the other hand, existing robust filtering techniques, which we review in Section A.1, either cannot tolerate a large proportion of outliers, or are not suitable for real-time implementation.

Therefore, we turn our interest to *causal* robust statistical inference. This problem arises when there is some hidden variable of interest that evolves over time, and the observations are either related to the hidden states by a simple statistical model, or they are "outliers." The goal is to infer the hidden variables despite outliers, and to do so causally, i.e. only using data up to the current time. For example, the hidden variable could be ego-motion, and the measurements are point correspondences from a low-level tracker. This is important for the real-time estimation and segmentation of structure and motion in vision-based control and robotics, for instance in tracking, manipulation, navigation, and surveillance. The goal is not just to do it fast, but to do it causally to avoid delays in the loop.

[1] We will define the notion of outlier properly in Section A.2; for now an intuitive acception suffices.

[2] I.e. the estimate at time t can only use measurements up to time t.

A.1 On-line structure from motion

Robust statistical inference in a dynamic context is a particular type of non-linear filtering problem. Given a probabilistic description of the uncertainty, as well as of the outlier generation mechanism, one can write the equations that govern the evolution of the conditional density of the hidden states (variables of interest as well as inlier/outlier distribution at time t) given all measurements available up to time t. The optimal filter evolves an estimate of such a density starting from some initial condition and is easily derived formally (e.g. [Jaz70] p. 174). From such a density, one would then have to construct some *point estimate*, for instance the maximum likelihood, the maximum a-posteriori, or the least mean square or median estimate.

In practice the conditional density can only be integrated numerically except for a handful of cases that does not include ours. In general there is a variety of numerical schemes available, including a plethora of particle evolution schemes [Liu01, DFG01]. However, ultimately we are not interested in the entire conditional density, but in a point estimate. Since the conditional density can only be approximated, a point-estimate computed from it is an approximation too. At that point, we may settle for a more efficient scheme that yields an (approximate) point estimate at the outset. There is another more philosophical reason to prefer a point-estimator: Often if a problem is well formulated the designer has reasons to believe that what we are looking for is a unique entity (e.g. ego-motion), and the multi-modality of the conditional density is only due to the output statistics (e.g. outliers). Therefore, estimating the entire conditional density would be an overkill. We make the assumption that the posterior density of the hidden variables would be unimodal if it were not for outliers. We therefore design a point estimator that only attempts to model the evolution of the dominant mode, rather than spreading computational resources thin by evolving particles at the tails of the distribution.

This work relates to a large body of literature in robust statistics that exploit heuristics related to sample consensus. A prototype algorithm of this class is RANSAC [FB81b], and the many variants that have been proposed to improve its efficiency [CM02a, CM05b, TM02], and robustness [TZ00, TD03, Nis03b]. Our work can be interpreted as a way to make RANSAC work in a causal fashion.

The goal of our work is to extend random sample consensus techniques to a dynamic context, so we can use them to handle outliers in real-time applications. We use tracking and SFM as examples, but by no means is our work limited to these applications. We illustrate our scheme on two simple problems that can be used to implement a robust object tracker and a robust ego-motion estimator. We show experiments where up to 85% of the measurements are outliers,

where previous robust filters fail, and where a large number of particles would be necessary in order to successfully capture the dominant mode. Although a batch processing of the data necessarily would give better estimates, it would require waiting for all data (or at least for a "large enough" temporal window) to be collected, thereby introducing destabilizing delays in a vision-based control scenario. Our scheme compares favorably with batch RANSAC, in the sense of being in the ballpark in terms of accuracy and robustness, despite only processing data in a causal fashion.

The only other scheme for causal robust inference in the context of computer vision that we are aware of is [Nis03b] which provided ways to expedite non-causal sampling schemes so they can be implemented fast enough to be used in real-time, whereas we propose causal processing algorithms to perform robust statistical inference. Whereas [Nis03b] is limited to processing a window of measurements, with no long-term memory, we integrate information causally from time 0 to time t and provide a long-term, albeit approximate, estimate of the conditional density. Our approach does not have an intrinsic delay, and is therefore more suitable to control applications that [Nis03b] that must carry a buffer of frames in order to run its version of RANSAC.

A.2 Problem statement

We are interested in some quantity $x_t \in \mathbb{R}^M$ (the "state") that evolves in time but that we cannot measure directly. In the simplest case the time dependency can be described by an ordinary difference equation (ODE), up to some "model uncertainty" v_t that we assume to be well captured by a simple statistical model, say a Gaussian process. In the simplest case the ODE is linear, $x_{t+1} = Ax_t + v_t$, $A \in \mathbb{GL}(M)$, and without loss of generality we can assume the uncertainty to be white and zero-mean[3] $\{v_t\}_{t\in\mathbb{Z}} \sim \mathcal{N}(0, R_v)$. Although we cannot measure x_t, we are given measurements $\{y_t \in \mathbb{R}^N\}_{t\in\mathbb{Z}}$ that come from one of two possible sources: At some time $t \in \mathbb{Z}$, **either** y_t is an instantaneous function of x_t, up to some measurement error that can be described by a simple statistical model, **or** y_t is "completely unrelated" to x_t. In the former case, the simplest instance is a linear model, $y_t = Cx_t + n_t$ where $C \in \mathbb{R}^{N\times M}$ and $\{n_t\}_{t\in\mathbb{Z}} \sim \mathcal{N}(0, R_n)$. In the latter case "completely unrelated" means that there is no statistical dependency between y_t and x_t that is simple enough for us to care to model it explicitly. Instead, in this case we call y_t an *outlier* and we wish to bar it from contributing to the inference of x_t. We do not know a-priori whether y_t is a valid measurement (inlier) or

[3]Lest we can consider the mean to be part of the state and we can pre-whiten the filter by simple (linear) projection operators [Jaz70]

an outlier, so we can model the choice with a stochastic indicator process χ_t: $y_t = \text{diag}(\chi_t)\,(Cx_t + n_t) + (I - \text{diag}(\chi_t))\nu_t$ where diag is an operator that maps a vector χ_t into a diagonal matrix, and ν_t is a stochastic process statistically independent of the other variables.

Since we allow for uncertainty in the evolution of x_t and in the measurement process, we have to specify what we mean by inference, which we will do shortly. To that end, we summarize the model that generates the data as follows[4]

$$\begin{cases} x_{t+1} = Ax_t + v_t & v_t \sim \mathcal{N}(0, R_v) \\ \chi_{t+1} = g(\chi_t, \mu_t) & n_t \sim \mathcal{N}(0, R_n) \\ y_t = \text{diag}(\chi_t)\,(Cx_t + n_t) + (I - \text{diag}(\chi_t))\nu_t \end{cases} \tag{A.1}$$

where the evolution of χ_{t+1} is written formally as a function $g : \{0,1\}^N \times \mathbb{R}^N \longrightarrow \{0,1\}^N$ of some unknown "input" μ_t that guarantees that χ_t remains an indicator function. Given measurements ("output" of this model), i.e. realizations of the process $\{y_t\}_{t\in\mathbb{Z}}$, we want to infer the state x_t, by employing only inlier measurements. We are thus in the realm of robust statistical inference, and in particular, since the inference concerns the state of a dynamical model, this problem is known as *robust filtering* [Hub81]. We indicate with $y_\tau^t \doteq (y_\tau, \ldots, y_t)$ a realization of the process $\{y_t\}$ from time τ to t, and we omit the subscript when $\tau = 0$. We denote by $y_t(i)$ the i-th component of the vector y_t.

In the absence of more detailed information on the outlier process ν_t, we will assume that these variables are uniformly distributed and independent, i.e. $p(\nu_t(i)) = 1/\eta$ where η is a nominal spreading value. This simple model has already been proven successful in applications similar to our [TZ00]. Moreover, we will assume that the processes $\chi^t(i)$ and $\chi^t(j)$ are independent for $i \neq j$. By using these assumption, we can get for example the density of $y_t(i)$ conditioned on the state x_t and χ_{t-1} as

$$\begin{aligned} p(y_t(i)|x_t, \chi_{t-1}) = p(y_t(i)|x_t, \chi_t(i)) \\ \times P(\chi_t(i) = 1|\chi_{t-1}(i)) + P(\chi_t(i) = 0|\chi_{t-1}(i))/\eta \end{aligned} \tag{A.2}$$

where $p(y_t(i)|x_t, \chi_t(i))$ can be derived from the density of the measurement noise $n_t(i)$ and $P(\chi_{t-1}(i)|\chi_t(i))$ is the transition probability encoded by $g(\cdot, \mu_t)$.

A.2.1 Optimal filter and its infeasibility

Ideally, given the observation y^t, we would like to obtain the posterior density $p(x_t, \chi_t|y^t)$. This problem has a well known solution in term of a recursive filter:

[4]The generative model can be specified in terms of evolution of the joint density of $\{x_t, y_t\}$, or in terms of the evolution of the generic realization x_t, y_t. We choose the latter because of its simplicity, although the two are entirely equivalent.

The evolution of the conditional density is immediate to derive using Bayes' rule and Chapman Kolmogorov's equations ((6.61) [Jaz70]):

$$\begin{cases} p(x_{t+1}, \chi_{t+1}|y^{t+1}) \propto p(y_{t+1}|x_{t+1}, \chi_{t+1}) \cdot \\ \qquad\qquad \cdot \int p(x_{t+1}, \chi_{t+1}|x_t, \chi_t) dP(x_t, \chi_t|y^t), \\ p(x_0, \chi_0|\emptyset) \sim p_0, \end{cases} \tag{A.3}$$

where $p_0 \sim p(x_0, \chi_0|\emptyset)$ is an initial estimate. The integrand $p(x_{t+1}, \chi_{t+1}|x_t, \chi_t)$ can be factored as:

$$p(x_{t+1}, \chi_{t+1}|x_t, \chi_t) = p(x_{t+1}|x_t) p(\chi_{t+1}|\chi_t) \tag{A.4}$$

under the assumptions we make in the previous section, and the transition probability $p(\chi_{t+1}|\chi_t) \doteq \pi_{ij}$ can be further specified as part of the model (A.1). Unfortunately, computing the integral above in the most general case is out of the question because $p(x_t, \chi_t|y^t)$ depends on the entire history χ^t of χ_t, via y^t, and is therefore a mixture of Gaussians with an exponential number of modes in both time t and the number of observation N. This is where we need to introduce approximations, and several options are available, from "sum-of-Gaussian" filters [AS72] to "interactive multiple models" [BL98], to various forms of generalized pseudo-Bayesian filters (e.g. GPB1, [BL98] p. 445). Each of these filters is based on a different heuristic, and it is impossible to prove general properties or approximation bounds in general. These filters arose in radar signal processing, where switches occur rarely between a small number of models (targets), and are not well suited to our application where the set of possible subsets of inliers is large.

A.2.2 Sampling of filters

Given *any* choice of putative inliers $\tilde{\chi}^t$, one could easily (i.e. using linear operations) solve equation (A.3) in the state x_t by employing a classical Kalman filter. In fact, given $\tilde{\chi}^t$, the posterior density $p(x_{t+1}|y^t, \tilde{\chi}^t)$ stays Gaussian. This suggests concentrating on the easier problem of getting the distribution on the continuous state x_t for a point estimate of the discrete state $\widehat{\chi}^t$. A natural criterion for the choice of $\widehat{\chi}^t$ is to maximize the posterior density

$$\widehat{x}_t, \widehat{\chi}^t \doteq \arg\max_{x_t, \chi_t} p(x_t, \chi^t|y^t). \tag{A.5}$$

Note that the function $p(x_t, \chi^t|y^t)$ is proportional to $p(x_t|y^t, \chi^t) p(\chi^t|y^t)$. Once χ^t is given as $\tilde{\chi}^t$, our system (A.1) reduces to a simple linear system. Therefore, the maximum likelihood estimator for $\widehat{x}_t(\tilde{\chi}^t) = \arg\max_{x_t} p(x_t, \tilde{\chi}^t|y^t)$ is given by the Kalman filter [Jaz70]. Unfortunately maximizing in the whole history χ^t is still a doubly exponential problem.

Algorithm 5 Sampling of filters.

1: **Initialization:** Randomly choose a set of assignments $\Upsilon \subset \{0,1\}^N$ of inliers/outliers among the the measurements.
2: **for all** assignments $\chi^\star$ in Υ **do**
3: Fix $\widehat{\chi}^t$ to $\widehat{\chi}_0 = ... = \widehat{\chi}_t = \chi^\star$.
4: Estimate $\widehat{x}_t$ using the Kalman filter.
5: Compute the MAP *score* of the assignment $\chi^\star$ as $p(y^t, \widehat{x}_t, \widehat{\chi}^t)$ which is proportion to $p(\widehat{x}_t, \chi^t | y^t)$.
6: **end for**
7: **Validation:** Choose $\widehat{x}_t$ and $\chi^\star$ that yield the maximum score.

The complexity can be reduced drastically by assuming χ_t to be constant over time, i.e. $\chi_0 = ... = \chi_t$, which is acceptable as long as the observations tend to preserve their inlier/outlier status. Still the possible assignments of χ_t are exponential in the number of observations. This can be addressed by sampling randomly the solution space, leading to the procedure summarized in Algorithm 5.

The three major shortcomings of Algorithm 5 are: (i) the constancy of $\chi_0, ..., \chi_t$ is too stringent, especially for large time t; (ii) naively sampling the space of assignments for χ_t, although in the limit provably solves the problem, is in practice too slow; (iii) the complexity is not constant but grows linearly with t. These issues will be addressed by the KALMANSAC procedure described in the next section.

A.3 KALMANSAC

In this section, we present an approximate solution that improves Algorithm 5 by (i) letting χ_t change over time, (ii) using an efficient sampling scheme and (iii) limiting the computational complexity to be constant for all times t.

We start from equation (A.3) and make our assumption explicit. We are going to assume that, at every instant of time, a best estimate of the inliers $\widehat{\chi}_t$ is available, as part of the solution of $\widehat{x}_t, \widehat{\chi}_t = \arg\max p(x_t, \chi_t | y^t)$. Compounding these choices over time we get the best causal estimate $\widehat{\chi}^t$ up to time t. We re-write $p(x_t, \chi_t | y^t) \propto p(x_t | \chi_t, y^t) p(\chi_t | y^t)$. Now we assume that

$$p(\chi_t | y^t) = \delta(\chi_t - \widehat{\chi}_t), \tag{A.6}$$

Algorithm 6 KALMANSAC.

1: **Initialization:** We are given the best choice of inliers uptime to time t, $\widehat{\chi}_t$, and the current best estimate of the state conditional density $p(x_t|\widehat{\chi}_t, y^t)$. Extract a subset $\Upsilon \subset \{0,1\}^N$ of the *minimal set* of assignments of inliers/outliers among the measurements.
2: **for all** $\chi^\star_{t+1}$ in the set Υ **do**
3: Initialize χ_{t+1} with $\chi^\star_{t+1}$.
4: **repeat** {Alternating maximization in χ_{t+1} and x_{t+1}}
5: Fix χ_{t+1} and compute $\arg\max_{x_{t+1}} p(x_{t+1}|\chi_{t+1}, y^{t+1})$ by reading off the updated state $\widehat{x}(\chi_{t+1})$ from one step of the Kalman filter.
6: Fix $x_{t+1} = \widehat{x}_{t+1}$ and estimate $\widehat{\chi}_{t+1} = \arg\max_{\chi_{t+1}} p(x_{t+1}, \chi_{t+1}|y^{t+1})$
7: Set $\chi_{t+1} = \widehat{\chi}_{t+1}$
8: **until** maximum amount of iterations has been reached or until the estimated state $\widehat{x}_{t+1}$ and estimated indicator $\widehat{\chi}_{t+1}$ do not change.
9: **end for**
10: **Validation** Select $\widehat{x}_{t+1}$ and $\widehat{\chi}_{t+1}$ that yield the maximum score $p(y^{t+1}, x_{t+1}, \chi_{t+1})$.

and equation (A.3) reduces to

$$p(x_{t+1}, \chi_{t+1}|y^{t+1}) \propto p(y_{t+1}|x_{t+1}, \chi_{t+1}) p(\chi_{t+1}|\widehat{\chi}_t) \times \int p(x_{t+1}|x_t) dP(x_t|\widehat{\chi}_t, y^t). \quad \text{(A.7)}$$

From equation (A.7), it is easy to check (recursively) that $p(x_t|\widehat{\chi}_t, y^t)$ stays as a Gaussian.

The maximization of equation (A.7) jointly in x_{t+1} and χ_{t+1} is still problematic because of the exponential number of possible assignments of χ_{t+1}. In the next section we will show an efficient sampling scheme that solves this problem.

A.3.1 Searching for inliers

We use equation (A.7) as the basis for our sampling filter. We start with an initial choice of inliers $\widehat{\chi}_0$ and with an initial density $\widehat{p}_0 \sim p(x_0|\emptyset)$. Now, at a generic time t, we assume we are given $\widehat{\chi}_t$ and $\widehat{p}_t \doteq p(x_t|\widehat{\chi}_t, y^t)$. We now want to generate the new maximum a-posteriori estimate $\widehat{\chi}_{t+1}, \widehat{x}_{t+1} \doteq \arg\max p(x_{t+1}, \chi_{t+1}|y^{t+1})$, or more in general $\widehat{\chi}_{t+1}$ and the entire mode $p(x_{t+1}|\widehat{\chi}_{t+1}, y^{t+1})$ since that will come for free from the Kalman filter and we will need it at the next step $t+1$.

Once the indicator χ_{t+1} is fixed, we can compute from equation (A.7) the

maximum a-posteriori state as a function of our choice

$$\widehat{x}_{t+1}(\chi_{t+1}) \doteq \arg\max_{x_{t+1}} p(x_{t+1}, \chi_{t+1}|y^{t+1})$$

using the Kalman filter. Now, we notice that in addition to the optimal choice of inliers $\widehat{\chi}_{t+1}$, there will be many more that yield nearly identical score, in particular all subsets of the optimal set of inliers. On the other hand, the presence of even a single outlier will severely affect the posterior. This suggests (i) to choose χ_{t+1} from the minimal set[5], thus increasing the chance of picking a *good* indicator, i.e. a choice of inliers that does not include any outliers; (ii) since $p(\widehat{x}_{t+1}(\chi_{t+1}), \chi_{t+1}|y^{t+1}) \simeq p(\widehat{x}_{t+1}(\tilde{\chi}_{t+1}), \chi_{t+1}|y^{t+1})$ as long as χ_{t+1} and $\tilde{\chi}_{t+1}$ are good indicators, we can estimate $\widehat{x}_{t+1}$ using either one; (iii) we can then improve the likelihood $p(y^{t+1}|\widehat{x}_{t+1}(\chi_{t+1}), \chi_{t+1})$ by maximizing $\widehat{\chi}_{t+1} = \arg\max_{\chi_{t+1}} p(\widehat{x}_{t+1}, \chi_{t+1}|y^{t+1})$. These three observations above, which are at the core of any RANSAC algorithm, yield our *KALMANSAC* algorithm, summarized in Algorithm 6.

One limitation of the KALMANSAC algorithm is that it relies completely on the previous estimate $\widehat{\chi}_t$ (because of the assumption (A.6)). Since the recovered $\widehat{\chi}_t$ is obtained from an approximate solution, this might prevent the algorithm from converging to the optimal solution. However, by relaxing the assumption on $p(\chi_t|y^t)$, we face a problem of exponential complexity as we have seen in Section A.2.1. We address this problem by using a *limited memory filter*, which trades off optimality for constant complexity, as we discuss in the next section.

A.3.2 Back-tracing: limited memory filter

The algorithm presented in Section A.3.1 provides an approximation of the maximum likelihood estimate of x_t together with an approximation of its covariance (from the Kalman filter) following a myopic estimate of the set of inliers χ_t based only on the current observations. Our main observation is that the best estimate $\widehat{x}_t$ available at time t may be affected by the approximations of our procedure, and therefore at the time step $t+1$ we choose to re-estimate it by using also the new measurements. More in general, we can do so for τ steps back in time. This leads to a limited memory filter ([Jaz70], p. 318). The steps of the algorithm are exactly the ones described in the previous section, except that the samples to be drawn are not for χ_{t+1}, but for $\chi_{t-\tau}^{t+1}$ for some $\tau \geq 0$, and the computational step involves a τ-step prediction and update for the process x_t, which are

[5]The *minimal set* is the set made of all assignments inlier/outlier with the minimum number of inliers so that the unknown parameters of the model can be unambiguously recovered. In our context, minimality is naturally connected to the selection of measurements that are sufficient to make the system observable [Kai80].

both standard for the Kalman filter, and the Viterbi algorithm for the process χ_t. As for the sampling of χ_t, we exploit the observation of Section A.2.2 that the inlier/outlier state tends to be preserved, and we let the samples $\chi_{t-\tau}^{t+1}$ to be constant across the time frame and equal to $\widehat{\chi}_t$ (line 1 of Algorithm 6). Notice that in the subsequent steps the assignment may change within the time-frame $[t-\tau, t+1]$.

Another benefit of using more steps of back-tracing is that each observation is checked for consistency with the inlier model across consecutive time step, making more accurate its classification as inlier or outlier. This only requires setting the transition probability $P(\chi_t|\chi_{t-1})$ to a function which penalizes switches of state (i.e. inliers becoming outliers and vice-versa).

Since sampling sets of inliers that can change arbitrarily, using a long interval τ is an overkill. Therefore, in the experimental section we have tested a version of the limited memory filter with $\tau = 2$.

A.3.3 Extension to non-linear models

Equation (A.7) is valid for models far more general than (A.1). The advantage of a linear model is to allow computing (A.7) using linear operations to evolve conditional mean and covariance. However, the sampling scheme proposed in Section A.3.1 is valid for any type of model, provided one has at least an approximate procedure to integrate (A.7). These include various types of approximations, from the extended Kalman filter (EKF, [Jaz70] p. 332) to numerical integration, even to particle filters. In the experimental section we will illustrate the performance of our sampling scheme with an EKF used to estimate structure from motion.

A.3.4 Accelerating the convergence

Although KALMANSAC samples minimal assignments of the inlier process χ_t, the number of samples required to find at least one good assignment with a certain probability is still large for large amounts of outliers. See for instance [TZ00] for a discussion on how to choose the right number of samples based on the percentage of the outliers and the size of minimal set. One way to reduce such number of samples is to employ a non-uniform sampling strategy that selects good assignments with high probability.

Our method is similar in spirit to [TM02, CM05b]. The basic idea is to extract the same set of M samples that the vanilla RANSAC would choose, but in an order that focuses first on the observations that we believe are inliers. The advantage of this approach is that one can use any sort of information to define the likely candidates, while being in the worst case equivalent to RANSAC when

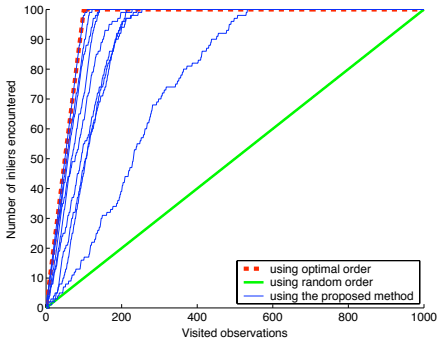


Figure A.1: *Accelerating the convergence. The figure shows the function* $f(j)$, $j = 1, ..., N$ *(see text) for various orderings: (dashed curve) ideal ordering with all inliers first; (dotted curve) random order; (solid curves) order obtained by the singleton inlier probability for the first 10 iterations of the filter. This figure is obtained based on the tracking problem with* 1000 *features and 85% outliers (see Section A.4).*

all M samples have been drawn [CM05b]. We use the prediction of the filter to infer the ordering of the observations.

One way of evaluating an ordering $(\sigma_1, ..., \sigma_N) \in S(N)$, $S(N)$ being the symmetric group of order N, is to consider the function $f(j) = \sum_{i=1}^{j} \chi_t(\sigma_j)$, $j = 1, ..., N$ (see Figure A.1) which counts how many inliers are encountered while visiting the observations in the specified order. The best possible order goes first through all inliers, then through all outliers. Then, we formulate the choice of the ordering as the N optimization problems (for $j = 1, ..., N$)

$$\max_{\sigma_1,...,\sigma_j} E[f(j)|y_t, \widehat{\chi}^{t-1}] = \max_{\sigma_1,...,\sigma_j} \sum_{i=1}^{j} S_t(\sigma_j), \tag{A.8}$$

where N is the number of observations and $S_t(i) = P[\chi_t(i)|y^t, \widehat{\chi}^{t-1}]$ is the *singleton inlier probability*. The N problems are solved *simultaneously* by simply ordering the observations by decreasing $S_t(i)$, which can be computed as

$$S_t(i) \propto \int p(y_t|\chi_t^j, \widehat{\chi}_{t-1}, x_t) p(x_t|y^t, \widehat{\chi}^{t-1})\, dx_t \tag{A.9}$$

Although this integral is computationally expensive, it can be approximated by setting $p(x_t|y^t, \widehat{\chi}^{t-1}) = \delta(x_t - x'_t)$, x'_t being the prediction of the filter at time t. This results in the score $\tilde{S}_t(i) \propto P(\chi_t(j)|\widehat{\chi}_{t-1}) p(y_t|\chi_t^j, \widehat{\chi}_{t-1}, x'_t)$, which is very quick to compute and in practice yields excellent ordering results, as one can see in Figure A.1.

A.4 Experiments

In this section we illustrate the various features of our algorithm on two examples: an 2-D tracker, where the model used is a second-order random walk that fits equation (A.1), and structure from motion, where we adopt a model borrowed from [AP95, CFJ02], which is non linear. In this case, the computation of the maximum a-posteriori desnities be approximated by an extended Kalman filter, but the structure of the algorithm presented in Section A.3.1 is unaltered. In order to perform systematic and controlled test, we employ simulation for the first case. For the more complex case of SFM we show results with both synthetic and real image sequences. We do so for the benefit of the skeptical reviewers, although the performance of our sampling scheme is best tested on simulation where the parameters of the experiment, which include a large variety of factors depending on the applications, can be carefully controlled.

A.4.1 Tracking

In this first set of experiments we choose to test the KALMANSAC on a simple 2-D object tracker. We consider tracking a group of 2-D points $y^i \in \mathbb{R}^2$, $i = 1, \ldots, N$ that evolve in time according to a similarity transformation, i.e. that satisfy the following model:

$$y_t^i = s_t R_t (y_0^i + T_t) \quad i = 1, 2, \ldots, N \tag{A.10}$$

where $s_t \in \mathbb{R}$ is the isotropic scaling, $R_t \in SO(2)$ is the 2D rotation and $T_t \in \mathbb{R}^2$ is the translation. Rather than representing the state as the vector containing s, R and T, we use an equivalent alternative representation that yields a linear system of equations. We define two variables $a_t \in \mathbb{R}^2$ and $b_t \in \mathbb{R}^2$, such that

$$a_t = s_t R_t T_t \quad \text{and} \quad b_t = s_t R_t \left[\begin{smallmatrix}1\\0\end{smallmatrix}\right] + a_t. \tag{A.11}$$

Then, it is immediate to obtain the expressions of s_t, R_t and T_t as a function of a_t and b_t. By substituting these expressions into equation (A.10) we obtain that

$$y_t^i = (I - Q(y_0^i))a_t + Q(y_0^i)b_t \quad i = 1, 2, \ldots, N \tag{A.12}$$

where $Q(y_0^i) = [y_0^i \quad (y_0^i)^\perp]$. Now, let $P_t = [a_t^T \quad b_t^T]^T \in \mathbb{R}^4$. We then choose a second-order random walk as a model for the dynamics, i.e.

$$P_{t+1} = P_t + V_t \quad \text{and} \quad V_{t+1} = V_t + n_t \tag{A.13}$$

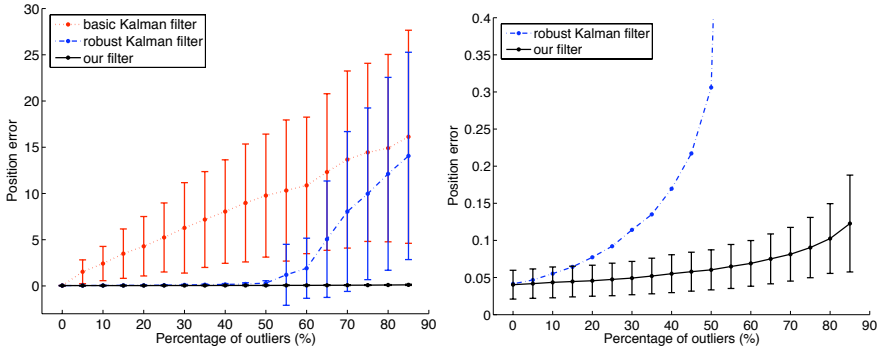


Figure A.2: **2D tracking experiment**. Left: performance comparison between the classic Kalman Filter (KF) (dotted), the robust KF (dot-dashed) and the KALMANSAC (solid). As one can see the KALMANSAC can produce very accurate estimates of the state up to 85% outliers, while the robust KF fails at 50% outliers and the classic KF as soon as outliers appear in the measurements. Right: a zoomed-in version of the plot highlight the difference in performance between the robust KF and KALMANSAC.

where $n_t \sim \mathcal{N}(0, R)$. As a consequence, the dynamical system corresponding to equation (A.1) is defined as follows:

$$\begin{cases} A = \left[\begin{smallmatrix} I & I \\ 0 & I \end{smallmatrix}\right] \in \mathbb{R}^{8\times 8} \\ C = [C_1^T, \ldots, C_N^T]^T \\ \qquad \text{where } C_i \doteq \left[\begin{smallmatrix} I - Q(y^i(0)) & Q(y^i(0)) \\ 0 & 0 \end{smallmatrix}\right] \in \mathbb{R}^{4\times 8} \end{cases} \tag{A.14}$$

and the state $x_t \doteq [P_t^T \;\; V_t^T]^T \in \mathbb{R}^8$ with initial conditions $x_0 = [0\ 0\ 1\ 0\ 0\ 0\ 0\ 0]^T$.

To evaluate the performance of KALMANSAC on this dynamical system, we compare it with a classic *Kalman filter (KF)*, and with the *Robust KF* [KY98]. The robust KF is a modification of the Kalman filter that uses a robust model of the observations [Hub81] to obtain an M-estimate of the state rather than the usual MAP-estimate. Contrary to our algorithm, the robust KF is capable of detecting outliers only based on the current estimate of the measurement prediction error, and, because of the weighing scheme used, is not completely resilient to the detected outliers. Moreover, it does not provide an explicit estimation of the process χ_t.

We choose $N = 100$ and generate 100 sequences of measurements each contaminated with $0\%, 5\%, \ldots, 85\%$ of outliers. On this data we run the classic KF, the robust KF and KALMANSAC. We compute the mean and the standard deviation of the state estimation error over the 100 trials and plot them

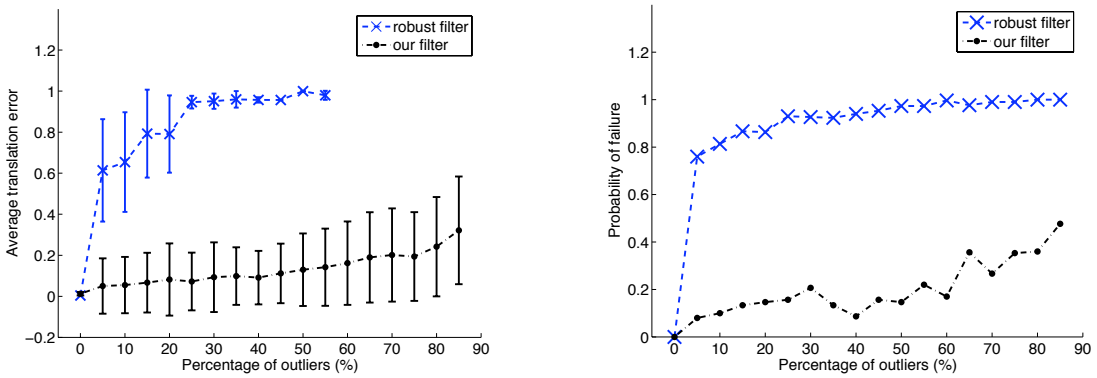


Figure A.3: **Mean and standard deviation error of motion estimation versus increasing proportions of outliers.** Left: The robust EKF (blue solid line) diverges immediately as soon as measurements contain 5% of outliers. Rather, KALMANSAC (black solid line) returns consistent motion estimates with a very low error even up to 75% outliers. Right: We plot the frequency of failure of both the robust EKF (blue solid line) and KALMANSAC (black solid line). Notice that the robust EKF is confused almost immediately by outliers as opposed to the KALMANSAC that starts to be confused half of the times when outliers are more than 80%.

in Figure A.2. Because of the enhanced sampling scheme (Section A.3.4), only 20 to 100 samples (depending on the outlier concentration) need to be drawn at each time step. Notice that the classic KF starts to return inconsistent results as soon as some outliers appear in the measurements. The robust KF can instead tolerate up to 50% outliers, but then rapidly degenerates. KALMANSAC proves to be very resistant to outliers, maintaining consistent estimates up to 85% outliers. To enhance the difference in performance between the robust KF and the KALMANSAC the plot in Figure A.2 has been repeated with a smaller range in the position error axis.

A.4.2 Structure from motion

In this section, we carry out experiments on a non-linear system that is suited to solve structure from motion under the assumption that (both linear and rotational) accelerations are a Brownian motion in time. As mentioned above, we adopt a model borrowed from [AP95, CFJ02].

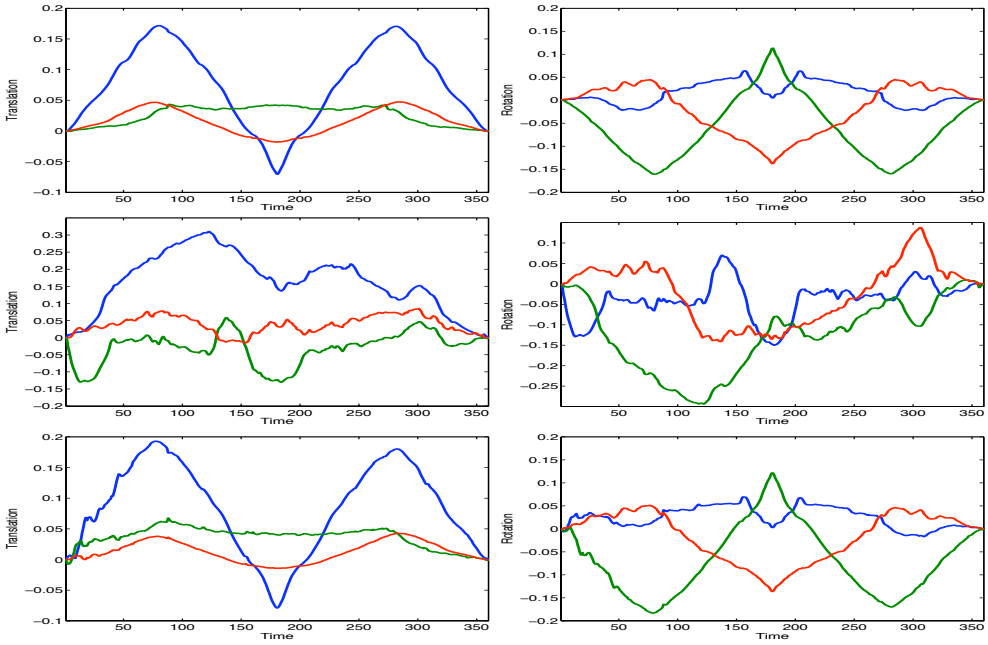


Figure A.4: Comparison between the robust filter and KALMANSAC based on the camera motion estimates (the sequence has been played back and forth once). The left column shows the translation error and the right column shows the rotation error (expressed in exponential coordinates). Top row: Pseudo-ground truth of the camera motion obtained by manually eliminate the outliers from data. Middle row: camera motion estimated by the robust filter. Bottom row: camera motion estimated by the KALMANSAC algorithm. One can see clearly the robust filter cannot make sense of the data at all because of the outliers while the KALMANSAC is able to estimate the camera motion.

Synthetic data. The synthetic scene is composed of 200 points. The camera rotates around the points, with center of rotation on the center of mass of the structure. We re-scale both translation and structure by fixing the depth coordinate of one of the points to 1. In this experiments, we want to show how the different implementations respond on average to different amounts of outliers. We simulate outliers as 3-D points whose projections follow a random walk of the second-order (diffusion). We choose proportions of $0\%, 5\%, ..., 85\%$ of outliers and for each of the filters we run 100 experiments and store the estimated motion. Then, we compute the mean and standard deviation for each outlier proportion and for both the robust EKF (an extension of [KY98] to nonlinear systems) and KALMANSAC and plot it in Figure A.3. As one can see, the robust EKF consistently fails to produce any sensible estimate of motion as soon as

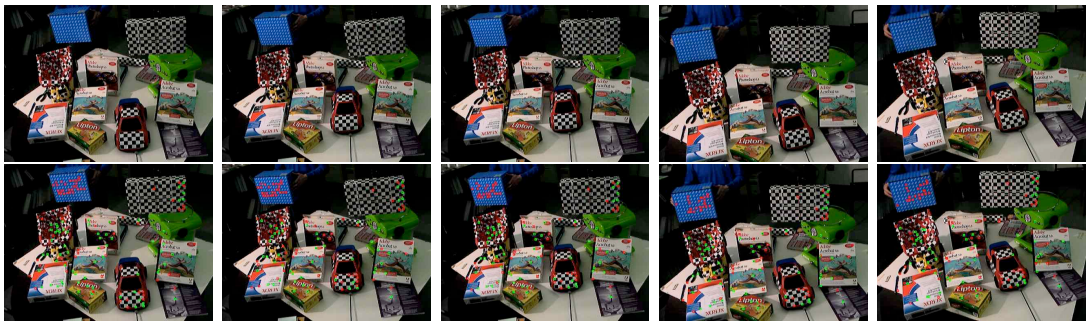


Figure A.5: **Experiments with real data.** Top row: 5 frames extracted from the real sequence (180 frames in total). The independently moving objects are: The car, the top checkerboard box and the blue box on the top-left corner of the image. The camera is moving sideways while these objects are also moving independently. Bottom row: Corresponding images with tracked features superimposed. The features are marked with either red or green squares. The green squares are the features considered as inliers while the red squares are considered outliers by the KALMANSAC.

some outliers are introduced in the measurements (Figure A.3, plot to the left). KALMANSAC can produce a sensible estimate of motion up to 75% of outliers (Figure A.3, plot to the left). In Figure A.3, right, we also show how frequently both filters diverge. We do so because in the case of structure from motion the recovery of motion parameters may be unsuccessful even when there are no outliers. Indeed, the same measurements (up to noise) may be generated by different configurations of points and motion so that recovery of motion parameters is an ambiguous process. In Figure A.3, plot to the right, one can see that while the robust EKF fails almost always as soon as we have 5% outliers, KALMANSAC starts to fail half of the times when we have more than 80% outliers.

Real data. In this set of experiments we test the robust EKF and KALMANSAC, on a real sequence (Figure A.5) where 3 independent objects that are moving within a rigid scene. These 3 objects plus additional T-junctions and reflections generate more than 60% of outliers in our measurements. Similarly to the case of synthetic data, in Figure A.4 we compare the performance of the two filters by comparing the estimated motion to motion that has been estimated with a classic EKF by manually discarding the outliers (pseudo-ground truth). As one can see the overall performance reflects the experiments on synthetic data: While the robust EKF fails to recover the motion parameters, KALMANSAC returns an estimate very similar to the pseudo-ground truth.

Discussion

We have presented an algorithm for causal robust statistical inference of the state of a dynamical model. Since the optimal solution is computationally prohibitive, we have proposed a random sampling approach that propagates the best current estimate of the set of inliers $\widehat{\chi}_t$, together with the state conditional density $p(x_t|\widehat{\chi}^t, y^t)$. We have derived this algorithm from the optimal filter, clearly highlighting the assumptions that underlie our approximation. We have validated our scheme experimentally, on both real and controlled synthetic experiments, and shown that it can operate successfully in the presence of a large proportion of outliers where existing robust filtering schemes fail.

APPENDIX B

Singularities in Structure from Forward Motion

Structure From Motion (SFM), the problem of reconstructing the 3D structure of a scene and the motion of a camera from a collection of images, can be counted among the success stories of Computer Vision. Over the course of the past twenty years, the geometry of the problem has been elucidated, and an extensive corpus of analysis and algorithms has been collated in several textbooks, e.g. [HZ00]. Some have even made their way into the real-world as commercial products. However, despite all the success, *forward motion still presents a challenge to existing SFM algorithms.* This problem rarely occurs in match-moving and virtual object insertion, for which most commercial algorithms are designed, but has become painfully patent with the recent push in autonomous driving.

The difficulties with forward motion are due in part to the limited lifetime of point-feature tracks: The most informative features are in the peripheral vision and they quickly move out of the visual field. This can be addressed by enlarging the field of view, all the way to omnidirectional cameras, which explains their popularity in robotic navigation. A less-easily fixed difficulty with forward motion is the presence of a *large number of local minima in the least-squares landscape of the reprojection error,* which many existing algorithms try to minimize. These local minima have been studied in some detail by Oliensis [Oli05] and Chiuso et al. [CBS00], using tools first introduced by Heeger and Jepson [HJ90] and Golub and Pereyra [GP73]. All have shown the presence of local minima corresponding to well-known ambiguities (Necker-reversal, plane-translation and bas-relief) and, more importantly for our problem, they have shown that *the least-squares reprojection error has singularities* corresponding to translation in the direction of point features, which introduce a fine-scale topology with many local minima around the true direction of translation (see [Oli05], figures 1 and 8, or [CBS00] figure 7). In order to overcome this problem, semi-global approaches based on convex optimization and fractional programming have been recently proposed [ACK06, KH05], but the resulting algorithms are too slow to run in real time for navigation applications, and the actual hypotheses to guarantee global convergence are not strictly satisfied for central projection and forward motion. Thus we are left with finding simpler ways to handle the strange geometry of the reprojection error surface. *Is such a geometry a product of the mathematics we are*

using, for instance the choice of L^2 to measure reprojection error, *or is it intrinsic to the problem?* Can a different choice of norm eliminate the singularities? Is there some reasonable assumption we can make on the scene or on the inference process that will make the singularities vanish?

In this chapter we show that the singularities are a byproduct of the mathematics, and can be easily eliminated. Specifically, we prove that imposing a bound on the depth of the scene makes the least-squares reprojection error continuous. We also show how such an assumption can be easily embedded in an iterative algorithm. It does not, however, show that local minima disappear altogether. In fact, only continuity can be guaranteed analytically, not convexity, and an empirical analysis shows that some local minima still exist.

B.1 CBS diagrams

In this section we will introduce the notation and rephrase some of the results of [CBS00] and [Oli05] for completeness.

Let $X_1, \dots, X_N \in \mathbb{R}^3$ be points on a rigid body (the scene). Let $x_i = X_i/|X_i|$ be the projection of the i-th point taken from a given vantage point on a spherical imaging surface. Let v and ω be the instantaneous linear and angular velocities of the scene (rigid body) relative to the camera. The motion causes the feature projections x_i to move on the imaging surface with velocities z_i. Our goal is to recover both the structure and motion from the measurements $x_1, \dots, x_N$ and $z_1, \dots, z_N$. To this end we note that the i-th measured velocity is given by

$$z_i = \omega \times x_i + \lambda_i \left(v - \langle v, x_i \rangle x_i\right) + n_i \tag{B.1}$$

where $\lambda_i = 1/|X_i|$ is the inverse of the distance of the i-th feature from the camera center and n_i is measurement noise. As z_i, n_i and $\omega \times x_i$ belong to the tangent space $T_{x_i}(\mathbb{S}^2)$ of the spherical imaging surface, the norm of the residual does not change if we pre-multiply by $-\widehat{x}_i = -x_i\times$, obtaining

$$-\widehat{x}_i z_i = \widehat{x}_i^2 \omega - \lambda_i \widehat{x}_i v - \widehat{x}_i n_i. \tag{B.2}$$

As the norm and statistics of $\pm\widehat{x}_i n_i$ and n_i are the same, we set $y_i = -\widehat{x}_i z_i$ and define the fitting cost

$$E(v, \omega, \lambda_1, ..., \lambda_N) = \sum_{i=1}^{N} \|n_i\|^2 = \sum_{i=1}^{N} \|y_i - \widehat{x}_i^2 \omega + \lambda_i \widehat{x}_i v\|^2. \tag{B.3}$$

The goal of SFM, for the purpose of our analysis, is to find the minimizers of the cost function E.

B.2 Reduced CBS diagrams

Heeger and Jepson introduced subspace projection into their SFM algorithm in 1990 [HJ90]. They showed that SFM can be reduced to a two-dimensional optimization problem. Their approach was later used by Chiuso et al. [CBS00] to study local minima in SFM after noticing that the local extrema in the reduced optimization are related to local minima in the original one. The two-dimensional landscape can be visualized in CBS diagram. Here we re-derive some of the properties of CBS diagrams since these results will be used later in proving continuity of the reduced cost function.

The squared residual $E(v, \omega, \lambda_1, \ldots, \lambda_N)$ is a function of both the inverse depths of the observed points (features) $\lambda_1, \ldots, \lambda_N$ (structure) and the camera angular ω and linear v velocities (motion). The idea of [HJ90] is to solve for all variables but linear velocity v. This gives a *reduced function* $E^*(v) = E(v, \omega^*(v), \lambda_1^*(v), \ldots, \lambda_N^*(v))$ of v only. The CBS diagram is then the function

$$E^*(v) = \min_{\omega \in \mathbb{R}^3, \lambda_1, \ldots, \lambda_N \in \mathbb{R}} E(v, \omega, \lambda_1, \ldots, \lambda_N) \tag{B.4}$$

An algebraically simplified variant, called *weighted* CBS diagram, is obtained by optimizing weighted residuals $w_i \|n_i\|^2$.

B.2.1 Properties

The function (B.4) enjoys several useful properties. First we note that fixing v turns the minimization (B.4) into a simple least-squares problem. To make this more explicit, define $y \doteq \begin{bmatrix} y_1^T & y_2^T & \ldots & y_N^T \end{bmatrix}^T$,

$$\Phi(v) \doteq \begin{bmatrix} \widehat{x}_1^2 & -\widehat{x}_1 v & 0 & \ldots & 0 \\ \widehat{x}_2^2 & 0 & -\widehat{x}_2 v & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \widehat{x}_N^2 & 0 & 0 & \ldots & -\widehat{x}_N v \end{bmatrix} \tag{B.5}$$

$a \doteq \begin{bmatrix} \omega^T & \lambda_1 & \lambda_2 & \ldots & \lambda_N \end{bmatrix}^T$, so that

$$E^*(v) = \min_{a \in \mathbb{R}^{3+N}} E(v, a) = \min_{a \in \mathbb{R}^{3+N}} \|y - \Phi(v) a\|^2. \tag{B.6}$$

Note that the optimal value $a(v)$ of the linear variable a – angular velocity and structure – for a given linear velocity v is given by

$$a(v) = \Phi(v)^\dagger y \tag{B.7}$$

where $\Phi(v)^\dagger$ is the pseudo-inverse of the matrix $\Phi(v)$.

B.2.1.1 Singular configurations

We say that a vector u is *proportional* to a vector v, $u \propto v$ in symbols, if there exists a number $\lambda \in \mathbb{R}$ such that $u = \lambda v$. Note that $0 \propto v$ for each v.

Proposition 5. *Consider features $X_1, \dots, X_N \in \mathbb{R}^3$, $N \geq 1$ and a vector $v \in \mathbb{S}^2$, such that $X_i \not\propto v$ for $i = 1, \dots, N$. The kernel of the matrix $\Phi(v)$ is not empty if, and only if, we can find a vector $\omega \in \mathbb{S}^2$ such that*

$$X_i^\top Q(v, \omega) X_i = 0, \tag{B.8}$$

$$Q(v, \omega) = \frac{1}{2}(v\omega^\top + \omega v^\top - 2\langle v, \omega \rangle I). \tag{B.9}$$

for all $i = 1, \dots, N$.

Proof. In order for the kernel of $\Phi(v)$ not to be empty, the following condition must hold

$$\exists\, (\omega, \lambda_1, \dots, \lambda_n) \;\neq\; 0 \;:\quad \widehat{X}_i^2 \omega \,-\, \widehat{X}_i v \lambda_i \;=\; 0, \quad i \;=\; 1, \dots, N. \tag{B.10}$$

Since we assumed that $\widehat{X}_i v \neq 0$, if we choose $\omega = 0$ then equation (B.10) is satisfied if and only if all λ_i vanish, thus ω has to be different from zero. Hence, from (B.10) we get the following condition by taking the scalar product with v.

$$\exists \omega \in \mathbb{S}^2 : \quad \langle \widehat{X}_i^2 \omega, v \rangle = 0, \quad i = 1, \dots, N. \tag{B.11}$$

The reverse implication is also true. Let us assume that (B.11) holds, we have $\langle \widehat{X}_i^2 \omega, v \rangle = 0$, $\langle \widehat{X}_i^2 \omega, X_i \rangle = 0$ and we know that X_i and v are linearly independent, so $\widehat{X}_i^2 \omega$ is directed along the vector $\widehat{X}_i v$, which means that there exists a λ_i (possibly null) such that (B.10) holds.

We rewrite (B.11) by using the identity $(a \times b) \times c = \langle a, c \rangle b - \langle b, c \rangle a$ to get $\widehat{X}_i^2 \omega = \langle X_i, \omega \rangle X_i - \langle X_i, X_i \rangle \omega$ and the equivalent conditions

$$\begin{aligned} v^\top (X_i X_i^\top \omega - (X_i^\top X_i)\omega) = \\ X_i^\top (v\omega^\top - v^\top \omega I) X_i = X_i^\top M X_i = 0, \end{aligned} \tag{B.12}$$

$$M = v\omega^\top - v^\top \omega I. \tag{B.13}$$

This is the equation of a quadric and is more easily studied by symmetrizing the matrix M, obtaining the equivalent constraints

$$X_i^\top Q X_i = 0, \quad i = 1, \dots, N, \quad Q = \frac{1}{2}(M + M^\top). \tag{B.14}$$

□

If $v \neq w$ it is easy to verify that the eigenvectors and eigenvalues of Q are given by

$$z_1 = v + \omega, \qquad \lambda_1 = \tfrac{1}{2}(1 - \langle v, \omega \rangle), \tag{B.15}$$

$$z_2 = v - \omega, \qquad \lambda_2 = -\tfrac{1}{2}(1 + \langle v, \omega \rangle), \tag{B.16}$$

$$z_3 = v \times \omega, \qquad \lambda_3 = -\langle v, \omega \rangle. \tag{B.17}$$

If $v = \pm\omega$, then $Q = \pm\widehat{v}^2$ and the eignevectors-eigenvalues are given by

$$z_1 = v, \qquad \lambda_1 = 0 \tag{B.18}$$

$$z_{2,3} \in v^{\perp}, \qquad \lambda_{2,3} = \mp 1. \tag{B.19}$$

Equation (B.8) then defines a double elliptical cone which contains v and ω, directed either along $v+\omega$ with major axis along $v-\omega$ (if $\langle v, \omega \rangle > 0$), or directed along $v - \omega$ with major axis along $v + \omega$ (if $\langle v, \omega \rangle < 0$). In case $\langle v, \omega \rangle = 0$ the cone degenerates to the union of the two planes $\langle X, v \rangle = 0$ and $\langle X, \omega \rangle = 0$.

We observe that:

- $\Phi(v)$ is $3N \times 3 + N$ so for $N = 1$ the kernel is not empty. Indeed $(\omega, \lambda_1) = (X_1, 0)$ belongs to the kernel.
- For $N = 2$ the kernel is also not empty. The conditions of Proposition 5 are in fact satisfied for $\omega \propto (v \times x_1)/\langle v, x_2 \rangle - (v \times x_2)/\langle v, x_1 \rangle + x_1 \times x_2$.
- If $N \geq 3$, then $\Phi(v)$ has full rank $N+3$ as long the quadratic equation (B.8) is not satisfied for some $\omega \in \mathbb{S}^2$.

The previous observations motivate the following definition:

Definition 8. The points $X_1, \ldots, X_N$ ($N \geq 3$), are said to be in *general position for the linear velocity* $v \neq 0$ if, and only if,

1. $X_i \not\propto v, \quad i = 1, \ldots, N,$
2. there exists no angular velocity ω such that $X_i^{\top} Q(v, \omega) X_i = 0$ for all $i = 1, \ldots, N$.

We also say that the points are in *general position* if they are in general position relatively to all velocities v for which point 1 above is satisfied.

We can now rephrase Proposition 5 in the following remark.

Remark 10. If points $X_1, \ldots, X_N$ are in general position for $v \neq 0$, then the matrix $\Phi(v)$ has full rank.

B.2.1.2 Correspondence of minimizers

Proposition 6. *The reduced cost function $E^*(v)$ enjoys the following properties. Let $\Omega = \mathbb{S}^2 \setminus \{x_1, \dots, x_N\}$ the open subset of the imaging sphere that does not include the feature projections and let $\{x_1, \dots, x_N\}$ be in general position. Then:*

- *If v is a critical point (or global minimizer) of $E^*(v)$ for $v \in \Omega$, then $(v, a(v))$ is a critical point (or global minimizer) of $E(v, a)$ for $(v, a) \in \Omega \times \mathbb{R}^{N+3}$.*
- *If (v, a) is a global minimizer of $E(v, a)$, then v is a global minimizer of $E^*(v)$.*

Proof. This kind of problems is discussed in [GP73]. The rank of $\Phi(v)$ is constant for all $v \in \Omega$. The result is then an immediate application of Thm. 2.1 therein. □

Prop. 6 says that all the critical points of the reduced cost function $E^*(v)$ (except at most singularities corresponding to the feature projections) correspond to critical points of the full cost function $E(v, a)$. It also says that global minimizers of the full and reduced cost functions correspond.

What the proposition does *not* say is that critical points of $E(v, a)$ are reflected by the reduced cost function $E^*(v)$. This is however easily seen. Since the points are in general position, $\Phi(v)$ never drops rank and $a(v)$ and $E^*(v)$ are differentiable. Let (v, a) be a critical point of $E(v, a)$. Then $a = a(v)$ and $\nabla E(v, a(v)) = 0$, so that

$$\frac{\partial E^*}{\partial v}(v) = \frac{\partial E}{\partial v}(v, a(v)) + \frac{\partial E}{\partial a}(v, a(v)) \frac{\partial a}{\partial v}(v) = 0. \tag{B.20}$$

So, except for singularities or pathological cases, the local minima of the reduced and full costs correspond.

B.2.2 Computation of the reduced functional

The reduced cost function (B.4) can be computed efficiently in two steps: Optimization of the depths given the motion, optimization of the rotation given the translation.

- **Optimizing the depths given the motion.** We fix ω and v in (B.3), obtaining for each λ_i its optimal value

$$\lambda_i = \frac{v^\top \widehat{x}_i^\top}{v^\top \widehat{x}_i^\top \widehat{x}_i v} \left(\widehat{x}_i^2 \omega - y_i \right), \tag{B.21}$$

which give the cost function

$$E(v,\omega) \;=\; \sum_{i=1}^{N} \|n_i\|^2 \;=\; \sum_{i=1}^{N} \left\| \left(I - \frac{\widehat{x}_i v v^\top \widehat{x}_i^\top}{v^\top \widehat{x}_i^\top \widehat{x}_i v} \right) \left(y_i - \widehat{x}_i^2 \omega \right) \right\|^2 . \quad \text{(B.22)}$$

The depth λ_i of a feature x_i parallel to the motion direction v does not affect the cost function, is undetermined and the corresponding operator $\widehat{x}_i v v^\top \widehat{x}_i^\top / v^\top \widehat{x}_i^\top \widehat{x}_i v$ is null. Moreover, as v approaches x_i, the limit of such operator does not exist (the directional limit, however, is well defined). This property of the residuals n_i generates singularities in correspondence of features in the cost function.

- **Optimizing the rotation given the translation.** Assume $v \not\propto x_i$ for all features x_i. Recall that $I = uu^\top - \widehat{u}^2$. Let $u = \widehat{x}_i v / \|\widehat{x}_i v\|$, so that

$$I - \frac{\widehat{x}_i v v^\top \widehat{x}_i^\top}{v^\top \widehat{x}_i^\top \widehat{x}_i v} = -\widehat{u}^2 \quad \text{(B.23)}$$

$$\|n_i\|^2 = \| -\widehat{u}^2 (y_i - \widehat{x}_i^2 \omega)\|^2 = \|\widehat{u}(y_i - \widehat{x}_i^2 \omega)\|^2 . \quad \text{(B.24)}$$

We now use the fact that $\widehat{u} = (v x_i^\top - x_i v^\top)/\|\widehat{x}_i v\|$, and that the residual $y_i - \widehat{x}_i^2 \omega$ is orthogonal to x_i to write

$$E(v,\omega) \;=\; \sum_{i=1}^{N} \|n_i\|^2 \;=\; \sum_{i=1}^{N} \frac{\|x_i v^\top (y_i - \widehat{x}_i^2 \omega)\|^2}{\|\widehat{x}_i v\|^2} \;=\; \sum_{i=1}^{N} \frac{(v^\top (y_i - \widehat{x}_i^2 \omega))^2}{\|\widehat{x}_i v\|^2} . \quad \text{(B.25)}$$

The latter expression is a standard least-squares estimation problem for ω. Note that estimating the velocity v is much harder, as the weights $\|\widehat{x}_i v\|$ depend on it. Note also that the residual $y_i - \widehat{x}_i^2 \omega$ is orthogonal to x_i, so that when the denominator $\|\widehat{x}_i v\|$ is small, because v approaches x_i, the numerator is small too. Solving for ω yields

$$\omega = \left(\sum_{i=1}^{N} \frac{\widehat{x}_i^2 v v^\top \widehat{x}_i^2}{\|\widehat{x}_i v\|^2} \right)^{-1} \sum_{i=1}^{N} \frac{\widehat{x}_i^2 v v^\top y_i}{\|\widehat{x}_i v\|^2} . \quad \text{(B.26)}$$

B.3 Bounded depth

In this section we extend the previous results and present the main contribution of this chapter, which is a characterization of the reduced cost function when the reconstructed depths are bounded.

Singularities (discontinuities) at feature locations $v = x_1, \ldots, x_N$ of the reduced cost function $E^*(v)$ are possible because the feature depths can be made arbitrarily small. This is reflected in equation (B.3): when a coefficient $\widehat{x}_i v \to 0$ as $v \to x_i$, the corresponding variable (inverse depth) λ_i approaches infinity to counter-balance. Thus we propose to work with a "regularized" reduced cost function $E^*_\alpha(v)$ defined as

$$E^*_\alpha(v) = \min_{\omega \in \mathbb{R}^3,\ |\lambda_i| \leq \alpha} E(v, \omega, \lambda_1, \ldots, \lambda_N). \tag{B.27}$$

The following lemmas are meant to establish the continuity of $E^*_\alpha(v)$ and the correspondence of global and local minimizers, by showing results in a more general setting (extending Proposition 6 with respect to correspondence of minimizers).

B.3.1 Continuity

Consider a function $E : V \times A \to \mathbb{R}$, where V is a topological space and A is a subset of $\mathbb{R}^k$, and let $E^*(v) = \min_{a \in A} E(v, a)$. In general, the continuity of E is not sufficient for E^* to be continuous, as it is only sufficient for the upper semicontinuity of E^*. However, if A is compact then E^* is indeed continuous, as shown in the following lemma.

Lemma 6. *Let E and E^* be defined as above. If E is continuous, then E^* is upper semicontinuous; moreover, if A is compact then E^* is also continuous.*

Proof. Consider the level sets

$$L^s_{E^*} = \{v \in V : E^*(v) < s\}, \tag{B.28}$$
$$L^s_E = \{(v, a) \in V \times A : E(v, a) < s\}. \tag{B.29}$$

Then

$$L^s_{E^*} = \pi_V (L^s_E), \tag{B.30}$$

where $\pi_V : V \times A \to V$ is the canonical projection. Since the level set L^s_E is open (E is continuous), its projection is open, too. The openness of $L^s_{E^*}$ for every s proves the upper semicontinuity of E^*.

Now if A is compact, we consider the level sets

$$W^s_{E^*} = \{v \in V : E^*(v) \leq s\}, \tag{B.31}$$
$$W^s_E = \{(v, a) \in V \times A : E(v, a) \leq s\}, \tag{B.32}$$

we also have

$$W^s_{E^*} = \pi_V (W^s_E), \tag{B.33}$$

since $E(v,\cdot)$ attains its minimum value in A. Since the level set W_E^s is closed (E is continuous) and A is compact, the projection is closed, too. This can be easily seen by taking a point $v \notin W_{E^*}^s$ and a covering of $\{v\} \times A$ with open subsets belonging to $W_E^{s\,c}$, then extracting a finite covering and considering the intersection of their projections on V to obtain a neighborhood of v in $W_{E^*}^{s\,c}$.

Thus, if A is compact E^* is both upper and lower semicontinuous, completing the proof. □

The result stated in the last lemma applies to the regularized cost function E_α^* defined in (B.27), since the bound on λ_i restricts the space of parameters to a compact space (there is a natural bound on the parameter ω, given the bound α on the parameters λ_i — see Lemma 11) . The use of the cost function (B.27) is in this sense justified.

B.3.2 Correspondence of minimizers

Now we see which results on correspondence of local and global minima we can obtain in this general setting.

Let a^* be a function defined from V to A such that

$$E^*(v) = \min_{a \in A} E(v,a) = E(v, a^*(v)). \tag{B.34}$$

For all the following lemmas we assume that E is continuous.

Lemma 7. *$\bar{v}$ is a global minimizer for E^* if, and only if $(\bar{v}, a^*(\bar{v}))$ is a global minimizer for E.*

Proof. It follows trivially from the equation $\min_{v \in V} E^*(v) = \min_{v \in V, a \in A} E(v,a)$. □

Lemma 8. *If $\bar{v}$ is a local minimizer for E^*, then $(\bar{v}, a^*(\bar{v}))$ is a local minimizer for E.*

Proof. Let $U \ni \bar{v}$ be a neighborhood such that $\bar{v}$ is a minimizer for E^* in U. Then for $v \in U$ and $a \in A$ we have

$$E(v,a) \geq E(v, a^*(v)) = E^*(v) \geq \\ \geq E^*(\bar{v}) = E(\bar{v}, a^*(\bar{v})), \tag{B.35}$$

so $(\bar{v}, a^*(\bar{v}))$ is a minimizer for E in $U \times A$. □

The converse is not true in general, so we consider an additional constraint on the regularity of a^*.

Lemma 9. *If $(\bar{v}, a^*(\bar{v}))$ is a local minimizer for E and a^* is continuous in $\bar{v}$, then $\bar{v}$ is a local minimizer for E^*.*

Proof. Let $W \ni (\bar{v}, a^*(\bar{v}))$ be a neighborhood such that $(\bar{v}, a^*(\bar{v}))$ is a minimizer for E in W. Since the mapping $\phi : v \mapsto (v, a^*(v))$ is continuous in $\bar{v}$, then $\phi^{-1}(W)$ contains a neighborhood U of $\bar{v}$. For $v \in U$, we have

$$E^*(v) = E(v, a^*(v)) \geq E(\bar{v}, a^*(\bar{v})) = E^*(\bar{v}), \tag{B.36}$$

so $\bar{v}$ is a minimizer for E^* in U. □

The following lemma makes it easy to assess the continuity of the function a^*, relying on the uniqueness of the minimizer.

Lemma 10. *If A is compact and $a^*(v)$ is the unique minimizer of $E(v, \cdot)$ for each v, then a^* is continuous.*

Proof. Consider a sequence $v_n \to v$. Then for $\epsilon > 0$ by continuity of E we have that for n large enough

$$E(v_n, a^*(v_n)) \leq E(v_n, a^*(v)) < E(v, a^*(v)) + \epsilon.$$

By contradiction, assume that $a^*(v_n)$ does not converge to $a^*(v)$. Then, because A is compact, there exists a subsequence $a^*(v_{n_k})$ that has a limit $\bar{a} \in A$ with $\bar{a} \neq a^*(v)$. Therefore

$$E(v, a^*(v)) \quad \leq \quad E(v, \bar{a}) \quad = \quad \lim_{k \to \infty} E(v_{n_k}, a(v_{n_k})) \quad < \quad E(v, a^*(v)) \; + \; \epsilon$$

for all $\epsilon > 0$. Hence $E(v, \bar{a}) = E(v, a^*(v))$ and $\bar{a} = a^*(v)$. □

Now we apply the previous lemmas to the bounded cost function E^*_α defined in (B.27).

Proposition 7. *Let $\{x_1, \ldots, x_N\}$ be in general position. Then, the reduced cost function E^*_α is continuous on $\mathbb{S}^2$. Moreover, the local minima of E^*_α on the domain $\mathbb{S}^2 \setminus \{x_1, \ldots, x_N\}$ are in one-to-one correspondence with the local minima of E on $\mathbb{S}^2 \setminus \{x_1, \ldots, x_N\} \times \mathbb{R}^3 \times \{|\lambda_i| \leq \alpha\}$.*

Proof. Lemma 6 (continuity) applies to E^*_α since the cost function E defined in (B.3) is continuous and the minimization is restricted to a minimization on a compact space (see also Lemma 11). Lemma 10 applies since for each v in $\mathbb{S}^2 \setminus \{x_1, \ldots, x_N\}$ the function $E(v, \cdot)$ is strictly convex and has a unique minimizer, hence both Lemmas 8, 9 hold. Together with the uniqueness of the minimizer, Lemmas 7, 8, 9 prove the one-to-one correspondence of local and global minima as stated. □

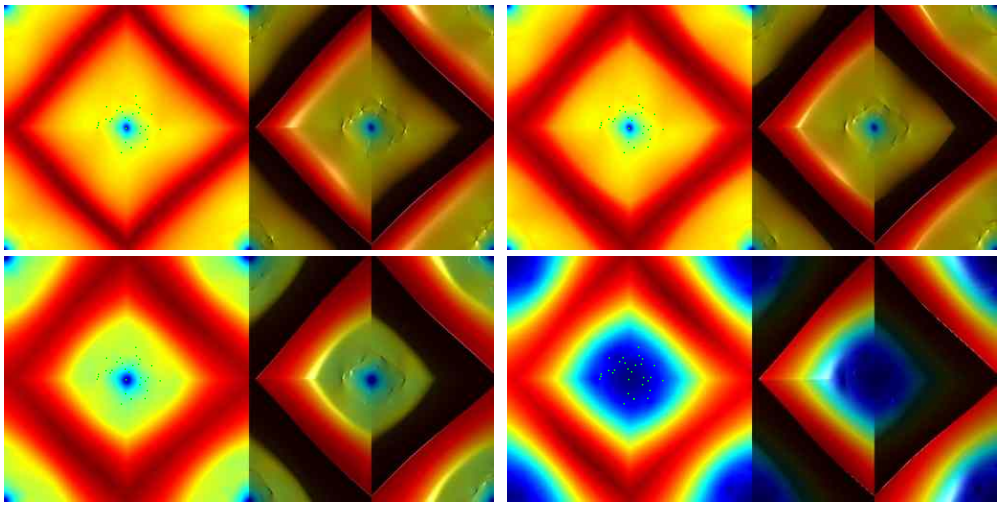


Figure B.1:]
CBS diagrams with various depth constraints. Top-left: no constraint; top-right: $|X_i| > 0.5L$ (L is the minimum distance of features from camera); bottom-left: $|X_i| > 0.8L$; bottom-right: $|X_i| > L$.

This sequence of lemmas constitutes the contribution of this chapter. A final remark shows that it is not possible to extend the correspondence of minimizers for minimizers that happen to be exactly on a feature point.

Remark 11. There are configurations of feature points $\{x_1, \ldots, x_N\}$ such that E a has local minimum for $v = x_N$, while E^* does not have a local minimum in x_N.

In the next section we will illustrate empirically the effect of bounded depth on CBS diagrams.

B.4 Experiments

In this section we illustrate the behavior of the CBS diagrams when bounding the underlying depths. Notice that this is not straightforward, because the CBS diagrams *do not depend on depth*, at least not directly, because it has been eliminated by subspace projection [HJ90].

In figure B.1 we show the CBS diagrams computed using various bounds on the scene depth. Each plot is divided into two parts, for each one the cost function is represented through a mapping of the domain $\mathbb{S}^2$ onto a square (by projecting onto a octahedron) and with color encoding. The left part shows just the function

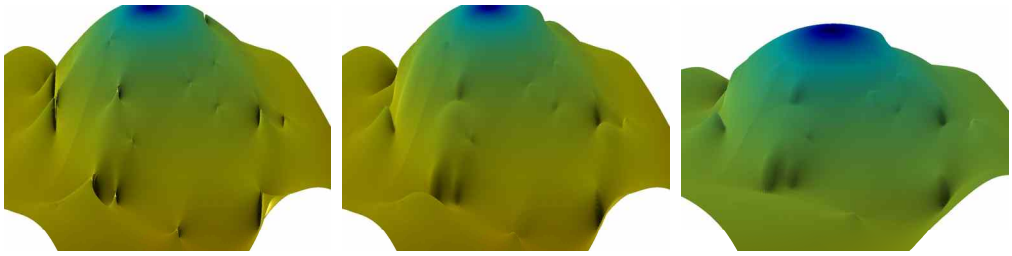


Figure B.2: 3D representation of CBS diagrams with various depth constraints. Top-left: no constraint; top-right: $|X_i| > 0.5L$; bottom: $|X_i| > 0.8L$.

no bound	86.2%
$\|X_i\| > 0.01L$	87.1%
$\|X_i\| > 0.05L$	87.1%
$\|X_i\| > 0.1L$	87.2%
$\|X_i\| > 0.5L$	87.1%
$\|X_i\| > 0.9L$	88.6%
$\|X_i\| > 1.5L$	69.2%

Table B.1: Percentage of successes for gradient descent algorithm with bounded depths. L is the minimum distance of the scene points from the camera (note that the unit of measure of distances is chosen so that the instantaneous velocity has unit norm).

values and the feature points in green, while the right part shows the function with simulated 3D appearance. The four plots start from no boundary (top left) and go to an extremely high boundary value (bottom right), which actually shows that imposing a bound on depth that is too strict eventually produces deep changes in the structure of the cost function, as expected.

The structure of the cost function near singularities, and its modifications with bounded depths, are better appreciated in the closeups in figure B.2, where the opposite of the function is shown (so that minima are represented as peaks). It is clear that the bound on depths has the effect of removing the singularities (as proved) and also that many of the local minima disappear, even though local peaks are still present.

We evaluated the improvement obtained by minimizing a regularized cost function when the algorithm used for minimization is a simple gradient descent, in order to provide a quantitative appraisal of the change in structure that the cost

function undergoes when considering bounded depths. For these experiments, we generated 10000 scenes of 25 random points each, selected according to a Gaussian distribution with standard deviation 0.5, at a distance of 2.5 from the camera. The measurements were generated by an instantaneous motion of the camera along the forward direction with random rotational component, and the descent algorithm was initialized randomly. The step on the direction of greatest descent was selected using a backtracking strategy, choosing the first point where the cost function decreases.

As can be seen from Table B.1, the fraction of gradient descent runs that correctly find the global minimum is consistently greater when the bound on the depths is enforced, even for loose bounds. Eventually, imposing a very strict boundary disrupts entirely the structure of the cost function, as shown by the last row of the table.

Discussion

We have proven that imposing a bound on the depth of the reconstructed scene makes the least-squares reprojection error continuous. This result shows that many of the local minima induced by singularities in the least-squares cost function that plague existing SFM algorithms when applied to autonomous navigation, where most of the motion is forward, do not really exist and can be simply avoided by altering the main iteration in the algorithms to enforce bounded depth. [1]

Our theoretical results can be visualized on the two-dimensional CBS diagrams. Note that we are imposing bounds on depth, and the CBS diagram does not depend on depth, so the reader will have to actually read the proofs to appreciate how this comes about.

Because the local structure is affected by the location of feature points used in the computation of the reduced cost, one could conceive sampling strategies where only subsets of the available measures are chosen. This way spurious minima due to the configuration of points change, whereas the only stable minimum under the sampling procedure should be the one due to the actual motion. This, however, is beyond the scope of this chapter and is the subject of ongoing work.

[1] An anonymous reviewer suggested that our analysis should be extended to other cost functionals, such as Sampson's distance [Sam82], that might be smooth without the need for boundedness or positivity constraints. However, we have focused on the least-squares reprojection error since this is the cost functional that was used to analyze the structure of local minima.

B.A Details

Lemma 11. *Let E be as defined in* (B.3) *and fix a bound $\alpha > 0$ on the inverse depths, i.e. restric the domain of E to the set $\mathbb{S}^2 \times A$, where $A = \mathbb{R}^3 \times [-\alpha, \alpha]^N$. Assume that there exists two features X_i and X_j which are linearly independent. Then there exists a constant B such that, for any given unit velocity v, any minimizer $(\omega, \lambda_1, \ldots, \lambda_N) \in A$ of $E(v, \cdot) : A \to \mathbb{R}$ must satisfy $\|\omega\| \leq B$.*

Proof. Each summand in (B.3) satisfies $\|y_i - \widehat{x}_i^2\omega + \lambda_i \widehat{x}_i v\| \geq \|\widehat{x}_i^2\omega\| - \|y_i\| - \alpha\|\widehat{x}_i v\|$. Moreover, since there are at least two linearly independent features X_i and X_j, we can find $\gamma > 0$ such that for all $\omega \in \mathbb{R}^3$ we have $\max\{\|\widehat{x}_i^2\omega\|, \|\widehat{x}_j^2\omega\|\} \geq \gamma\|\omega\|$. Hence we get the bound

$$E(v, \omega, \lambda_1, \ldots, \lambda_N) \geq \sum_i \|\widehat{x}_i^2\omega\| - C_1 \geq \gamma\|w\| - C_1$$

for some constant C_1. Therefore $\lim_{\|\omega\| \to \infty} E = \infty$ uniformly in v and the minimization problem can be restricted to a limited subset of A. □

APPENDIX C

An Open Implementation of SIFT

This appendix describes an implementation of the Scale-Invariant Transform Feature (SIFT) detector and descriptor [Low04]. The implementation is designed to produce results compatible to Lowe's original version.[1] Designed for the MATLAB environment, it is broken down into several M and MEX files that enable running only portion of the algorithm for increased flexibility.

The SIFT *detector* extracts from an image a collection of *frames* or *keypoints*. These are oriented disks attached to blob-like image structures. As the image translates, rotates and scales, the frames track the blobs and thus the image deformation. By *canonization*, i.e. by mapping the frames to a standard shape (a canonical disk), the effect of the deformation on keypoint appearance is removed.

The SIFT *descriptor* is a coarse description of the edge found in the frame. Due to canonization, descriptors are invariant to translations, rotations and scalings and are designed to be robust to residual small distortions.

The SIFT detector and descriptor are discussed in depth in [Low04]. Here we only describe the interface to our implementation and some technical details.

C.1 User reference: the sift function

The SIFT detector and the SIFT descriptor are invoked by means of the function `sift`, which provides a unified interface to both.

Example 4 (Invocation). The following lines run the SIFT detector and descriptor on the image `data/test.jpg`.

```
I = imread('data/test.png') ;
I = double(rgb2gray(I)/256) ;
[frames,descriptors] = sift(I, 'Verbosity', 1) ;
```

The pair option-value `'Verbosity',1` causes the function to print a detailed progress report.

[1] See `http://www.cs.ubc.ca/\~lowe/keypoints/`

The sift function returns a $4 \times K$ matrix frames containing the SIFT frames and a $128 \times K$ matrix descriptors containing their descriptors. Each frame is characterized by four numbers which are in order (x_1, x_2) for the center of the frame, σ for its scale and θ for its orientation. The coordinates (x_1, x_2) are relative to the upper-left corner of the image, which is assigned coordinates $(0, 0)$, and may be fractional numbers (sub-pixel precision). The scale σ is the smoothing level at which the frame has been detected. This number can also be interpreted as size of the frame, which is usually visualized as a disk of radius 6σ. Each descriptor is a vector describing coarsely the appearance of the image patch corresponding to the frame (further details are discussed in Appendix C.A.3). Typically this vector has dimension 128, but this number can be changed by the user as described later.

Once frames and descriptors of two images I_1 and I_2 have been computed, siftmatch can be used to estimate the pairs of matching features. This function uses Lowe's method to discard ambiguous matches [Low04]. The result is a $2 \times M$ matrix, each column of which is a pair (k_1, k_2) of indices of corresponding SIFT frames.

Example 5 (Matching). Let us assume that the images I1 and I2 have been loaded and processed as in the previous example. The code

```
matches = siftmatch(descriptors1, descriptors2) ;
```

stores in matches the matching pairs, one per column.

The package provides some ancillary functions; you can

- use plotsiftframe to plot SIFT frames;
- use plotsiftdescriptor to plot SIFT descriptors;
- use plotmatches to plot feature matches;
- use siftread to read files produced by Lowe's implementation.

Example 6 (Visualization). Let I1, I2 and matches be as in the previous example. To visualize the matches issue

```
plotsiftmatches(I1,I2,frames1,frames2,matches)
```

The sift function has many parameters. The default values have been chosen to emulate Lowe's original implementation. Although our code does not result in frames and descriptors that are 100% equivalent, in general they are quite similar.

C.1.1 Scale space parameters

The SIFT detector and descriptor are constructed from the *Gaussian scale space* of the source image $I(x)$. The Gaussian scale space is the function

$$G(x;\sigma) \stackrel{\Delta}{=} (g_\sigma * I)(x)$$

where g_σ is an isotropic Gaussian kernel of variance $\sigma^2 I$, x is the spatial coordinate and σ is the scale coordinate. The algorithm make use of another scale space too, called *difference of Gaussian (DOG)*, which is, coarsely speaking, the scale derivative of the Gaussian scale space.

Since the scale space $G(x;\sigma)$ represents the same information (the image $I(x)$) at different levels of scale, it is sampled in a particular way to reduce redundancy. The domain of the variable σ is discretized in logarithmic steps arranged in O octaves. Each octave is further subdivided in S sub-levels. The distinction between octave and sub-level is important because at each successive octave the data is spatially downsampled by half. Octaves and sub-levels are identified by a discrete *octave index* o and *sub-level index* s respectively. The octave index o and the sub-level index s are mapped to the corresponding scale σ by the formula

$$\sigma(o,s) = \sigma_0 2^{o+s/S}, \quad o \in o_{\min} + [0,...,O-1], \quad s \in [0,...,S-1] \tag{C.1}$$

where σ_0 is the base scale level.

The `sift` function accepts the following parameters describing the Gaussian scale space being used:

- `NumOctaves`. This is the number of octaves O in (C.1).
- `FirstOctave`. Index of the first octave $o_{\min}$: the octave index o varies in $o_{\min}, ..., o_{\min}+O-1$. It is usually either 0 or -1. Setting $o_{\min}$ to -1 has the effect of doubling the image before computing the Gaussian scale space.
- `NumLevels`. This is the number of sub-levels S in (C.1).
- `Sigma0`. Base smoothing: This is the parameter σ_0 in (C.1).
- `SigmaN`. Nominal pre-smoothing: This is the nominal smoothing level of the input image. The algorithm assumes that the input image is actually $(g_{\sigma_n} * I)(x)$ as opposed to $I(x)$ and adjusts the computations according. Usually σ_n is assumed to be half pixel (0.5).

C.1.2 Detector parameters

The SIFT frames (x, σ) are points of local extremum of the DOG scale space. The selection of such points is controlled by the following parameters:

- `Threshold`. Local extrema threshold. Local extrema whose value $|G(x, ; \sigma)|$ is below this number are rejected.
- `EdgeThreshold`. Local extrema localization threshold. If the local extremum is on a valley, the algorithm discards it as it is too unstable. Extrema are associated with a score proportional to their sharpness and rejected if the score is below this threshold.
- `RemoveBoundaryPoints`. Boundary points removal. If this parameter is set to 1 (true), frames which are too close to the boundary of the image are rejected.

C.1.3 Descriptor parameters

The SIFT descriptor is a weighed and interpolated histogram of the gradient orientations and locations in a patch surrounding the keypoint. The descriptor has the following parameters:

- `Magnif`. Magnification factor m. Each spatial bin of the histogram has support of size $m\sigma$, where σ is the scale of the frame.
- `NumSpatialBins`. Number of spatial bins. Together with the next parameter, this number defines the extension and dimension of the descriptor. The dimension of the descriptor (the total number of bins) is equal to $\texttt{NumSpatialBins}^2 \times \texttt{NumOrientBins}$ and its extension (the patch where the gradient statistic is collected) has radius $\texttt{NumSpatialBins} \times m\sigma/2$.
- `NumOrientBins`. Number of orientation bins.

C.1.4 Direct access to SIFT components

The SIFT code is decomposed in several M and MEX files, each implementing a portion of the algorithm. These programs can be run on their own or replaced. Appendix C.A contains information useful to do this.

Example 7 (Computing the SIFT descriptor directly). Sometimes it is useful to run the descriptor code alone. This can be done by calling the function `siftdescriptor` (which is actually a MEX file.) See the function help for further details.

Symbol	Description
$o \in [o_{\min}, o_{\min} + O - 1]$	Octave index and range
$s \in [s_{\min}, s_{\max}]$	Scale index and range
$\sigma(o, s) = \sigma_0 2^{o+s/S}$	Scale coordinate formula
σ_0	Base scale offset
M_0, N_0	Base spatial resolution (octave $o = 0$)
$N_o = \lfloor \frac{N_0}{2^o} \rfloor$, $M_o = \lfloor \frac{M_0}{2^o} \rfloor$	Octave lattice size formulas
$x_o \in [0, ..., N_o] \times [0, ..., M_o]$	Spatial indexes and rages
$x = 2^o x_o$	Spatial coordinate formula
$F(\cdot, \sigma(o, \cdot))$	Octave data
$G(x, \sigma)$	Gaussian scale space
$D(x, \sigma)$	DOG scale space

Figure C.1: *Scale space parameters.* The SIFT descriptors uses two scale spaces: a Gaussian scale space and a Difference of Gaussian scale space. Both are described by these parameters.

C.A Internals

C.A.1 Scale spaces

Here a *scale space* is a function $F(x, \sigma) \in \mathbb{R}$ of a spatial coordinate $x \in \mathbb{R}^2$ and a scale coordinate $\sigma \in \mathbb{R}_+$. Since a scale space $F(\cdot, \sigma)$ typically represents the same information at various scales $\sigma \in \mathbb{R}$, its domain is sampled in a particular way in order to reduce the redundancy.

The scale coordinate σ is discretized in logarithmic steps according to

$$\sigma(s, o) = \sigma_0 2^{o+s/S}, \qquad o \in \mathbb{Z}, \quad s = 0, ..., S - 1$$

where o is the *octave index*, s is the *scale index*, $S \in \mathbb{N}$ is the *scale resolution* and $\sigma_0 \in \mathbb{R}_+$ is the *base scale offset*. Note that it is possible to have octaves of negative index.

The spatial coordinate x is sampled on a lattice with a resolution which is a function of the octave. We denote x_o the spatial index for octave o; this index is mapped to the coordinate x by

$$x = 2^o x_o, \qquad o \in \mathbb{Z}, \quad x_o \in [0, ..., N_o - 1] \times [0, ..., M_o - 1].$$

where (N_o, M_o) is the spatial resolution of octave o. If (M_0, N_0) is the the resolution of the base octave $o = 0$, the resolution of the other octaves is obtained

as

$$N_o = \lfloor \frac{N_0}{2^o} \rfloor, \qquad M_o = \lfloor \frac{M_0}{2^o} \rfloor.$$

It will be useful to store some scale levels twice, across different octaves. We do this by allowing the parameter s to be negative or greater than S. Formally, we denote the range of s as $[s_{\min}, s_{\max}]$. We also denote the range of the octave index o as $[o_{\min}, o_{\min} + O - 1]$, where $O \in \mathbb{N}$ is the total number of octaves. See Figure C.1 for a summary of these symbols.

The SIFT detector makes use of the two scale spaces described next.

Gaussian Scale Space. The *Gaussian scale space* of an image $I(x)$ is the function

$$G(x, \sigma) \stackrel{\Delta}{=} (g_\sigma * I)(x)$$

where the scale $\sigma = \sigma_0 2^{o+s/S}$ is sampled as explained in the previous section. This scale space is computed by the function gaussianss. In practice, it is assumed that the image passed to the function gaussianss is already pre-smoothed at a nominal level σ_n, so that $G(x, \sigma) = (g_{\sqrt{\sigma^2 - \sigma_n^2}} * I)(x)$. As suggested in [Low04], the pyramid is computed incrementally from the bottom by successive convolutions with small kernels.

Difference of Gaussians scale space. The *difference of Gaussians (DOG) scale space* is the scale "derivative" of the Gaussian scale space $G(x, \sigma)$ along the scale coordinate σ. It is given by

$$D(x, \sigma(s, o)) \stackrel{\Delta}{=} G(x, \sigma(s + 1, o)) - G(x, \sigma(s, o)).$$

It is obtained from the Gaussian scale space by the diffss function.

Remark 12 (Lowe's parameters). Lowe's implementation uses the following parameters:

$$\sigma_n = 0.5, \quad \sigma_0 = 1.6 \cdot 2^{1/S}, \quad o_{\min} = -1, \quad S = 3$$

In order to compute the octave $o = -1$, the image is doubled by bilinear interpolation (for the enlarged image $\sigma_n = 1$). In order to detect extrema at all scales, the Difference of Gaussian scale space has $s \in [s_{\min}, s_{\max}] = [-1, S + 1]$. Since the Difference of Gaussian scale space is obtained by differentiating the Gaussian scale space, the latter has $s \in [s_{\min}, s_{\max}] = [-1, S + 2]$. The parameter O is set to cover all octaves (i.e. as big as possible.)

C.A.2 The detector

The SIFT frames (or "keypoints") are a selection of (sub-pixel interpolated) points (x, σ) of local extremum of the DOG scale-space $D(x, \sigma)$, together with

an orientation θ derived from the spatial derivative of the Gaussian scale-space $G(x, \sigma)$. For what concerns the detector (and being in general different for the descriptor), the "support" of a keypoint (x, σ) is a Gaussian window $H(x)$ of deviation $\sigma_w = 1.5\sigma$. In practice, the window is truncated at $|x| \leq 4\sigma_w$.

The Gaussian and DOG scale spaces are derived as in Section C.A.1. In this Section, the parameters S, O, $s_{\min}$, $s_{\max}$, $o_{\min}$, σ_0 refer to the DOG scale space. The Gaussian scale space has exactly the same parameters of the DOG scale space except for $s_{\max}^{\text{DOG}}$ which is equal to $s_{\max} - 1$.

The extraction of the keypoints is carried one octave per time and articulated in the following steps:

- **Detection.** Keypoints are detected as points of local extremum of $D(x, \sigma)$ (Section C.A.1). In the implementation the function `siftlocalmax` extracts such extrema by looking at $9 \times 9 \times 9$ neighborhoods of samples.

  As the octave is represented by a 3D array, the function `siftlocalmax` returns indexes k (in MATLABconvention) that are to be mapped to scale space indexes (x_1, x_2, s) by

  $$k - 1 = x_2 + x_1 M_o + (s - s_{\min}) M_o N_o.$$

  Alternatively, one can use `ind2sub` to map the index k to a subscript (i, j, l) and then use

  $$x_1 = j - 1, \qquad x_2 = i - 1, \qquad s = l - 1 + s_{\min}.$$

  Because of the way such maxima are detected, one has always $1 \leq x_2 \leq M_o - 2$, $1 \leq x_1 \leq N_o - 2$ and $s_{\min} + 1 \leq s \leq s_{\max} - 1$.

  Since we are interested both in local maxima and minima, the process is repeated for $-G(x, \sigma)$. (If only positive maxima and negative minima are of interest, another option is to take the local maxima of $|G(x, \sigma)|$ directly, which is quicker.)

- **Sub-pixel refinement.** After being extracted by `siftlocalmax`, the index (x_1, x_2, s) is fitted to the local extremum by quadratic interpolation. At the same time, a threshold on the "intensity" $D(x, \sigma)$ and a test on the "peakedness" of the extremum is applied in order to reject weak points or points on edges. These operations are performed by the `siftrefinemx` function.

  The edge rejection step is explained in detail in the paper [Low04]. The sub-pixel refinement is an instance of Newton's algorithm.

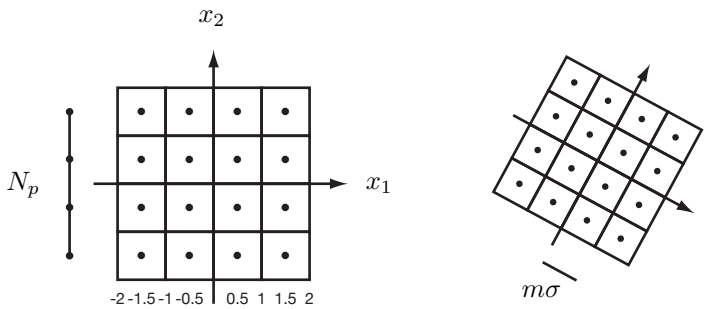


Figure C.2: *SIFT descriptor layout.* The actual size of a spatial bin is $m\sigma$ where σ is the scale of the keypoint and $m = 3.0$ is a nominal factor.

- **Orientation.** The orientation θ of a keypoint (x, σ) is obtained as the predominant orientation of the gradient in a window around the keypoint. The predominant orientation is obtained as the (quadratically interpolated) maximum of the histogram of the gradient orientations $\angle\nabla G(x_1, x_2, \sigma)$ within a window around the keypoint. The histogram is weighed both by the magnitude of the gradient $|\nabla G(x_1, x_2, \sigma)|$ and a Gaussian window centered on the keypoint and of deviation 1.5σ (the Gaussian window defines the region of interest as well). After collecting the data in the bins and before computing the maximum, the histogram is smoothed by a moving average filter.

  In addition to the global maximum, each local maximum with a value above 0.8% of the maxium is retained as well. Thus for each location and scale multiple SIFT frames might be generated.

  These computations are carried by the function siftormx.

C.A.3 The descriptor

The SIFT descriptor of a keypoint (x, σ) is a local statistic of the orientations of the gradient of the Gaussian scale space $G(\cdot, \sigma)$.

Histogram layout. The SIFT descriptor (Figure C.2) is an histogram of the gradient orientations augmented and 2-D locations in the support of the SIFT frame. Formally, the domain of the histogram are the tuples $(x, \theta) \in \mathbb{R}^2 \times \mathbb{R}/\mathbb{Z}$. The bins form a three dimensional lattice with $N_p = 4$ bins for each spatial direction and $N_o = 8$ bins for the orientation for a total of $N_p^2 N_o = 128$ components (these numbers can be changed by setting the appropriate parameters). Each spatial bin is square with unitary edge. The window $H(x)$ is Gaussian with

deviation equal to half the extension of the spatial bin range, that is $N_p/2$.

Keypoint normalization. In order to achieve invariance, the histogram layout is projected on the image domain according to the frame of reference of the keypoint. The spatial dimensions are multiplied by $m\sigma$ where σ is the scale of the keypoint and m is a nominal factor (equal to 3.0 by default). The layout is also rotated so that the axis x_1 is aligned to the direction θ of the keypoint.

Weighing. The histogram is weighed by the gradient modulus and a Gaussian windowed and tri-linearly interpolated. More in detail, each sample $(x_1, x_2, \angle\nabla G(x, \sigma))$ is

- weighed by $|\nabla G(x, \sigma)|$;
- weighed by the gaussian window $H(x)$;
- projected on the centers of the eight surrounding bins;
- summed to each of this bins proportionally to its distance from the respective center.

Remark 13. (Lowe's impelmentation) In order to achieve full compatibility with Lowe's original implementation, one needs to pay attention to many little details as the memory layout of the descriptor and the convention for the gradient orientations. These are detailed in the source code.

References

[2D3] Boujou from 2D3 Limited. http://www.2d3.com.

[ACK06] S. Agarwal, M. Chandraker, F. Kahl, D. Kriegman, and S. Belongie. "Practical global optimization for multiview geometry." In *Proc. ECCV*, 2006.

[AP95] A. Azarbayejani and A. Pentland. "Recursive estimation of motion, structure and focal length." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **17**(6):562–575, 1995.

[AS72] D. L. Alspach and H. W. Sorenson. "Nonlinear Bayesian estimation using Gaussian sum approximation." *IEEE Trans. Aut. Contr.*, **17(4)**:439–448, 1972.

[ATW05] M. Allan, M. K. Titsias, and C. K. I. Williams. "Fast Learning of Sprites using Invariant Features." In *Proc. BMVC*, 2005.

[Bas77] D. Basu. "On the Elimination of Nuisance Parameters." *Journal of the American Statistical Associtation*, **72**(358), 1977.

[Ber85] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis*. Springer-Verlag, 1985.

[BFC02] S. Belongie, C. Fowlkes, F. Chung, and J. Malik. "Spectral Partitioning with Indefinite Kernels Using the Nyström Extension." In *Proc. ECCV*, 2002.

[BJ02] F. R. Bach and M. I. Jordan. "Kernel independent componet analysis." *Journal of Machine Learninig Research*, **3**(1), 2002.

[BK93] J. Buhmann and H. Kühnel. "Vector Quantization with Complexity Costs." *IEEE Trans. on Information Theory*, **39**, 1993.

[BL98] Y. Bar-Shalom and X.-R. Li. *Estimation and tracking: principles, techniques and software*. YBS Publishing, February 1998.

[Bla05] A. Blake. "Visual tracking: a research roadmap." In O. Faugerasn, Y. Chen, and N. Paragios, editors, *Mathematical Models of Computer Vision: The Handbook*. Springer, 2005.

[BM01a] S. Baker and I. Matthews. "Equivalence and Efficiency of Image Aligment Algorithms." In *Proc. CVPR*, 2001.

[BM01b] A. Berg and J. Malik. "Geometric blur for template matching." In *Proc. CVPR*, 2001.

[BM02] S. Baker and I. Matthews. "Lucas-Kanade 20 Years On: A Unifying Framework: Part 1." Technical Report CMU-RI-TR-02-16, CMU, 2002.

[BMP02] S. Belongie, J. Malik, and J. Puzicha. "Shape Matching and Object Recognition Using Shape Contexts." *IEEE Trans. Pattern Anal. Mach. Intell.*, **24**(4):509–522, 2002.

[BNM98] Simon Baker, Shree K. Nayar, and H. Murase. "Parametric Feature Detection." *IJCV*, **27**(1):27–50, 1998.

[Boo89] F. L. Bookstein. "Principal Warps: Thin-Plate Splines and the Decomposition of Deformations." *PAMI*, **11**(6):567–585, 1989.

[BV04] S. Boyd and L. Vanderberghe. *Convex Optimization*. Cambridge University Press, 2004.

[BWR92] J. B. Burns, R. S. Weiss, and E. M. Riseman. "The non-existence of general-case view-invariants." In *Geometric Invariance in Computer Vision*, 1992.

[BYJ97] M. J. Black, Y. Yacoob, A. D. Jepson, and D. J. Fleet. "Learning Parametrized Models of Image Motion." In *Proc. CVPR*, 1997.

[Car06] M. Carreira-Perpiñán. "Fast Nonparametric Clustering with Gaussian Blurring Mean-Shift." In *Proc. ICML*, 2006.

[CBJ00] H. F. Chen, P. N. Belhumeur, and D. W. Jacobs. "In Search of Illumination Invariants." In *Proc. CVPR*, 2000.

[CBS00] A. Chiuso, R. Brockett, and S. Soatto. "Optimal Structure from Motion: Local Ambiguities and Global Estimates." *IJCV*, 2000.

[CC06] H. Choi and S. Choi. "Robust kernel Isomap." *Pattern Recognition*, 2006.

[CCM99] V. Caselles, B. Coll, and J.-M. Morel. "Topographic Maps and Local Contrast Changes in Natural Images." *IJCV*, **33**(1), 1999.

[CDD04] G. Csurka, C. R. Dance, L. Dan, J. Willamowski, and C. Bray. "Visual Categorization with Bags of Keypoints." In *Proc. ECCV*, 2004.

[CFJ02] A. Chiuso, P. Favaro, H. Jin, and S. Soatto. "Structure from Motion Causally Integrated over Time." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **24**(4):523–535, April 2002.

[Che95] Y. Cheng. "Mean Shift, Mode Seeking, and Clustering." *PAMI*, **17**(8), 1995.

[CLG89] P. A. Chou, T. Lookabaugh, and R. M. Gray. "Entropy-Constrained Vector Quantization." *IEEE Trans. on Acoustics, Speech, and Signal Processing*, **37**(1), 1989.

[CM02a] O. Chum and J. Matas. "Randomized RANSAC with $T_{(d,d)}$ test." In *British Machine Visison Conference*, pp. 448–457, 2002.

[CM02b] D. Comaniciu and P. Meer. "Mean Shift: A Robust Approach Toward Feature Space Analysis." *PAMI*, **24**(5), 2002.

[CM05a] O. Chum and J. Matas. "Matching with PROSAC – Progressive Sample Consensus." In *Proc. CVPR*, 2005.

[CM05b] O. Chum and J. Matas. "Matching with PROSAC – Progressive Sample Consensus." In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, June 2005.

[CMT04] T. F. Cootes, S. Marsland, C. J. Twining, C. J. Taylor, and K. Smith. "Groupwise Diffeomorphic Non-rigid Registration for Automatic Model Building." In *Proc. ECCV*, 2004.

[CRM01] D. Comaniciu, V. Ramesh, and P. Meer. "The Variable Bandwidth Mean Shift and Data-Driven Scale Selection." In *Proc. ICCV*, 2001.

[CS04] D. Cremers and S. Soatto. "Motion Competition: A Variational Approach to Piecewise Parametric Motion Segmentation." *IJCV*, 2004.

[CT06] T. M. Cover and J. A. Thomson. *Elements of Information Theory*. Wiley, 2006.

[CWB00] Olivier Chapelle, Jason Weston, Leon Bottou, and Vladimir Vapnik. "Vicinal Risk Minimization." In *NIPS*, pp. 416–422, 2000.

[DFG01] A. Doucet, N. de Freitas, and N. Gordon. *Sequential Monte Carlo methods in practice*. Springer Verlag, 2001.

[DHS01] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Inerscience, 2001.

[Dor05] G. Dorkó. "Scale and Affine Invariant Local Detectors and Descriptors." Technical report, INRIA, 2005.

[DS04] G. Dorkó and C. Schmid. "Object Class Recognition Using Discriminative Local Features." *PAMI (submitted)*, 2004.

[DZZ04] D.Chetverikov, Z.Megyesi, and Z.Jankó. "Finding Region Correspondences for Wide Baseline Stereo." In *Proc. ICPR*, volume 4, pp. 276–279, 2004.

[FB81a] M. A. Fischler and R. C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." *Communications of the ACM*, **24**(6):381–395, 1981.

[FB81b] M. A. Fischler and R. C. Bolles. "RANSAC, random sampling consensus: a paradigm for model fitting with applications to image analysis and automated cartography." *Comm. ACM*, **26**:381–395, 1981.

[FB05] F. Fraundorfer and H. Bischof. "A novel performance evaluation method of local detectors on non-planar scenes." In *Proc. CVPR*, 2005.

[FFP03] L. Fei-Fei, R. Fergus, and P. Perona. "A Bayesian Approach to Unsupervised One-Shot Learning of Object Categories." In *Proc. ICCV*, 2003.

[FFP04] L. Fei-Fei, R. Fergus, and P. Perona. "Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories." In *CVPR Workshop*, 2004.

[FH75] K. Fukunaga and L. D. Hostler. "The Estimation of the Gradient of a Density Function, with Applications in Pattern Recognition." *IEEE Trans. on Information Theory*, **21**(1), 1975.

[FH05] P. F. Felzenszwalb and D. P. Huttenlocher. "Pictorial Structures for Object Recognition." *IJCV*, **61**(1), 2005.

[FHI00] D. A. Forsyth, J. Haddon, and S. Ioffe. "Finding objects by grouping primitives." In David A. Forsyth, L. Mundy, Vito Di Gesù, and Roberto Cipolla, editors, *Shape, contour and grouping in computer vision*. Springer-Verlag, 2000.

[FHT00] J. Friedman, T. Hastie, and R. Tibshirani. "Additive logistic regression: a statistical view of boosting." *Annals of Statistics*, **28**(2):337–407, 2000.

[FJ99] B. J. Frey and N. Jojic. "Transformed Component Analysis: Joint Estimation of Spatial Transformations and Image Components." In *Proc. ICCV*, 1999.

[FJ00] B. J. Frey and N. Jojic. "Transformation-Invariant Clustering and Dimensionality Reduction Using EM." *PAMI*, 2000.

[FJ03] B. J. Frey and N. Jojic. "Transformation-Invariant Clustering Using the EM Algorithm." *PAMI*, **25**(1), 2003.

[FPZ03] R. Fergus, P. Perona, and A. Zisserman. "Object class recognition by unsupervised scale-invariant learning." In *Proc. CVPR*, 2003.

[FPZ05] R. Fergus, P. Perona, and A. Zisserman. "A Sparse Object Category Model for Efficient Learning and Exhaustive Recognition." In *Proc. CVPR*, 2005.

[Fri01] J. H. Friedman. "Greedy Function Approximation: a Gradient Boosting Machine." *The Annals of Statistics*, 2001.

[FS01] S. Fine and K. Scheinberg. "Efficient SVM Training Using Low-Rank Kernel Representations." *Journal of Machine Learninig Research*, 2001.

[FTV03] V. Ferrari, T. Tuytelaars, and L. Van Gool. "Wide-baseline Multiple-view Correspondences." In *Proc. CVPR*, 2003.

[FTV05] V. Ferrari, T. Tuytelaars, and L. Van Gool. "Simultaneous Object Recognition and Segmentation from Single or Multiple Model Views." *IJCV*, 2005.

[GC95] F. Girosi and N. T. Chan. "Prior knowledge and the creation of "virtual" examples for RBF networks." In *Neural Networks for Signal Processing*, pp. 201–21, 1995.

[GD06a] K. Grauman and T. Darrell. "Pyramid Match Kernels: Discriminative Calssification with Sets of Image Features." Technical Report MIT-CSAIL-TR-2006-020, MIT, 2006.

[GD06b] K. Grauman and T. Darrell. "Unsupervised Learning of Categories from Sets of Partially Matching Image Features." In *Proc. CVPR*, 2006.

[GL96] G. H. Golub and C. F. Van Loan. *Matrix Computations*. The Johns Hopkins University Press, 1996.

[GP73] G. H. Golub and V. Pereyra. "The Differentiation of Pseudo-Inverses and Nonlinear Least Squares Problems Whose Varaibles Separates." *SIAM Journal on Numerical Analysis*, **10**(2), 1973.

[GR05] D. B. Grimes and R. P. N. Rao. "Bilinear Sparse Coding for Invariant Vision." *Neural Computation*, **17**, 2005.

[GZW03] C. Guo, S.-C. Zhu, and Y. N. Wu. "Towards a Mathematical Theory of Primal Sketch and Sketchability." In *Proc. ICCV*, p. 1228, 2003.

[HB98] G. D. Hager and P. N. Belhumeur. "Efficient Region Tracking With Parametric Models of Geometry and Illumination." *PAMI*, 1998.

[HB05] M. Hein and O. Bousquet. "Hilbertian Metrics and Positive Definite Kernels on Probability Measures." In *Proc. AISTAT*, 2005.

[HBW04] T. Hertz, A. Bar-Hillel, and D. Weinshall. "Learning Distance Functions for Image Retrieval." In *Proc. CVPR*, 2004.

[HF94] L. Haglund and D. J. Fleet. "Stable Estimation of Image Orientation." In *Proc. ICIP*, pp. 68–72. IEEE, 1994.

[HF01] G. Hermosillo and O. Faugeras. "Dense Image Matching with Global and Local Statistical Criteria: A Variational Approach." In *Proc. CVPR*, 2001.

[HJ90] D.J. Heeger and A.D. Jepson. "Subspace Methods for Recovering Rigid Motion, Part II: Theory." In *RBCV-TR*, 1990.

[HS88] C. Harris and M. Stephens. "A combined corner and edge detector." In *Proc. of The Fourth Alvey Vision Conference*, pp. 147–151, 1988.

[HTF01] T. Hastie, R. Tibishirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.

[Hub81] P. J. Huber. *Robust statistics*. Wiley, New York, 1981.

[HZ00] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[Jaz70] A.H. Jazwinski. *Stochastic Processes and Filtering Theory*. Academic Press, 1970.

[JSY03] H. Jin, S. Soatto, and A. J. Yezzi. "Multi-view stereo beyond Lambert." *IJCV*, **1**:171–178, 2003.

[Kai80] T. Kailath. *Linear Systems*. Prentice Hall, 1980.

[KB03] T. Kadir and M. Brady. "Scale Saliency: A Novel Approach to Salient Feature and Scale Selection." In *Intl. Conference Visual Information Engineering*, 2003.

[KH05] F. Kahl and D. Henrion. "Globally optimal estimates for geometric reconstruction." In *Proc. ICCV*, 2005.

[KI02] H. Kashima and A. Inokuchi. "Kernels for Graph Classification." In *ICDM Workshop on Active Mining*, 2002.

[Kin96] D. B. Kinghorn. "Integrals and Derivatives for Correlated Gaussian Fuctions Using Matrix Differential Calculus." *International Journal of Quantum Chemestry*, **57**:141–155, 1996.

[KJF07] A. Kannan, N. Jojic, and B. J. Frey. "Fast Transformation-Invariant Component Analysis." *IJCV*, **77**(1–3), 2007.

[KNF76] W. L. G. Koontz, P. Narendra, and K. Fukunaga. "A Graph-Theoretic Approach to Nonparametric Cluster Analyisis." *IEEE Trans. on Computers*, **c-25**(9), 1976.

[Knu98] D. Knuth. *The Art of Computer Programming: Seminumerical Algorithms*, volume 2. Third Edition, 1998.

[KRS04] A. Kaplan, E. Rivlin, and I. Shimshoni. "Robust feature matching across widely separated color images." In *Proc. CVPR*, 2004.

[KS70] J. D. Kalbfleisch and D. A. Sprott. "Application of Likelihood Methods to Models Involving Large Number of Parameters." *Journal of The Royal Statistical Society, Series B*, **32**(2), 1970.

[KY98] K.R. Koch and Y. Yang. "Robust Kalman filter for rank deficient observation models." *J. Geodesy*, **72**(7-8):436–41, 1998.

[LBG80] Y. Linde, A. Buzo, and R. M. Gray. "An Algorithm for Vector Quantizer Design." *IEEE Trans. on Communications*, 1980.

[Lea06] E. G. Learned-Miller. "Data Driven Image Models through Continuous Joint Alignment." *PAMI*, **28**(2), 2006.

[Lhu98] M. Lhuillier. "Efficient Dense Matching for Textured Scenes Using Region Growing." In *Proc. ECCV*, 1998.

[Lin98] T. Lindeberg. "Feature Detection with Automatic Scale Selection." *IJCV*, **30**(2):77–116, 1998.

[Liu01] J. Liu. *Monte Carlo strategies in scientific computing*. Springer Verlag, 2001.

[LJ05] H. Ling and D. W. Jacobs. "Deformation Invariant Image Matching." In *Proc. ICCV*, 2005.

[LLF05] V. Lepetit, P. Lagger, and P. Fua. "Randomized Trees for Real-Time Keypoint Recognition." In *Proc. CVPR*, 2005.

[Low04] D. G. Lowe. "Distinctive Image Features from Scale-Invariant Keypoints." *IJCV*, **2**(60):91–110, 2004.

[Low07] D. Lowe. "Implementation of the Scale Invariant Feature Transform." `http://www.cs.ubc.ca/~lowe/keypoints/`, 2007.

[LS07] H. Ling and S. Soatto. "Proximity Distribution Kernels for Geometric Context in Category Recognition." In *Proc. CVPR*, 2007.

[LSP04] S. Lazebnik, C. Schmid, and J. Ponce. "Semi-Local Affine Parts for Object Recognition." In *Proc. BMVC*, pp. 959–968, 2004.

[LSP06] S. Lazebnik, C. Schmid, and J. Ponce. "Beyond Bag of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories." In *Proc. CVPR*, 2006.

[LY05] J. Lim and M.-H. Yang. "A Direct Method for Modeling Non-rigid Motion with Thin Plate Spline." In *Proc. CVPR*, 2005.

[LZ93] M. S. Langer and S. W. Zucker. "Diffuse Shading, Visibility Fields, and the Geometry of Ambient Light." In *Proc. CVPR*, 1993.

[LZW07] L. Lin, S.-C. Zhu, and Y. Wang. "Layered Graph Matching with Graph Editing." In *Proc. CVPR*, 2007.

[MB99] S. MacLane and G. Birkhoff. *Algebra*. AMS, 1999.

[MCU02] J. Matas, O. Chum, M. Urban, and T. Pajdla. "Robust Wide Baseline Stereo from Maximally Stable Extremal Regions." In *Proc. BMVC*, 2002.

[MP05] P. Moreels and P. Perona. "Evaluation of Features Detectors and Descriptors based on 3D objects." In *Proc. ICCV*, 2005.

[MRW99] S. Mika, G. Rätsch, J. Weston, B. Schölkopf, and K.-R. Müller. "Fisher Discriminant Analysis with Kernels." In *Proc. IEEE Neural Networks for Signal Processing Workshop*, 1999.

[MS89] D. Mumford and J. Shah. "Optimal approximations by piecewiese smooth functions and associated variational problems." *Comm. Pure and Applied Mathematics*, 1989.

[MS03] K. Mikolajczyk and C. Schmid. "A performance evaluation of local descriptors." In *Proc. CVPR*, 2003.

[.]

[MS04] K. Mikolajczyk and C. Schmid. "Scale & affine invariant interest point detectors." *IJCV*, **11**(60):63–86, 2004.

[MSK03] Y. Ma, S. Soatto, J. Kosecka, and S. S. Sastry. *An Invitation to 3-D Vision*. Springer Verlag, 2003.

[MTS04] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. "A Comparison of Affine Region Detectors." *IJCV*, **1**(60):63–86, 2004.

[Great overview of affine covariant detector.]

[Nis03a] D. Nistér. "Preemptive RANSAC for Live Structure and Motion Estimation." In *Proc. ICCV*, 2003.

[Nis03b] D. Nister. "Preemptive ransac for live structure and motion estimation." In *Proc. Intl. Conf. on Computer Vision*, pp. 199–206, 2003.

[NJT06] E. Nowak, F. Jurie, and B. Triggs. "Sampling Strategies for Bag-of-Features Image Classification." In *Proc. ECCV*, 2006.

[OCC07] B. A. Olshausen, C. Cadieu, J. Culpepper, and D. K. Warland. "Bilinear Models of Natural Images." In *Proc. SPIE*, volume 6492, 2007.

[OF96] B. A. Olshausen and D. J. Field. "Emergence of simple-cell receptive field properties by learning a sparse code for natural images." *Nature*, 1996.

[Oli05] J. Oliensis. "The least-squares error for structure from infinitesimal motion." *IJCV*, 2005.

[PB06] Alexei Pozdnoukhov and Samy Bengio. "Invariances in kernel methods: From samples to objects." *Pattern Recogn. Lett.*, **27**(10):1087–1097, 2006.

[PD06] F. Perronnin and C. Dance. "Fisher Kenrels on Visual Vocabularies for Image Categorizaton." In *Proc. CVPR*, 2006.

[PD07] S. Paris and F. Durand. "A Topological Approach to Hierarchical Segmentation using Mean Shift." In *Proc. CVPR*, 2007.

[Piq06] J. V. Piqueres. "The Persistence of Ignorance." http://www.ignorancia.org/, 2006.

[PLR04] J. Ponce, S. Lazebnik, F. Rothganger, and C. Schmid. "Toward True 3D Object Recognition." In *Reconnaisance de Formes et Intelligence Artificelle*, 2004.

[POP98] Constantine P. Papageorgiou, Michael Oren, and Tomaso Poggio. "A General Framework for Object Detection." In *International Conference on Computer Vision*, pp. 555–562, 1998.

[PTK04] E. Pichon, A. Tannenbaum, and R. Kikinis. "A statistically based flow for image segmentation." *Medical Image Analysis*, 2004.

[PZ98] P. Pritchett and A. Zisserman. "Wide Baseline Stereo Matching." In *Proc. ICCV*, pp. 754–760, 1998.

[QH06] H. Qiu and E. R. Hancock. "Graph matching and clustering using spectral partitions." *Pattern Recognition*, **39**, 2006.

[RB05] S. Roth and M. J. Black. "On the Spatial Statistics of Optical Flow." In *Proc. ICCV*, 2005.

[Rem84] M. Rémon. "On a Concept of Partial Sufficiency: L-Sufficiency." *International Statistical Review*, 1984.

[RM93] K. Rose and D. Miller. "A Deterministic Annealing Algorithm for Entropy-Constrained Vector Quantizer Design." In *Proc. of IEEE Conf. on Signals, Systems and Computers*, 1993.

[Roc03] P. I. Rockett. "Performance Assesment of Feature Detection Algorithms: A Methodology and Case Study on Corner Detectors." *IEEE Trans. on Image Processing*, **12**(12), 2003.

[Sai02] S. R. Sain. "Multivariate locally adaptive density estimation." *Comp. Stat. and Data Analysis*, **39**, 2002.

[Sam82] P. D. Sampson. "Fitting conic section to "very scattered" data: An iterative refinement of the Bookstein algorithm." *Computer Vision, Graphics and Image Processing*, **18**, 1982.

[Sch01] B. Schölkopf. "The Kernel Trick for Distances." *Proc. NIPS*, 2001.

[SD96] S. M. Seitz and C. R. Dyer. "View morphing." In *Proc. SIGGRAPH*, pp. 21–30, New York, NY, USA, 1996. ACM Press.

[SH05] A. Stein and M. Hebert. "Incorporating Background Invariance into Feature-Based Object Recognition." In *Seventh IEEE Workshop on Applications of Computer Vision (WACV)*, 2005.

[Sha48] C. Shannon. "A Mathematical Theory of Communications." *The Bell System Tech. Journal*, **27**, 1948.

[SIF05] M. Salzmann, S. Ilic, and P. Fua. "Physically Valid Shape Parameterization for Monocular 3-D Deformable Surface Tracking." In *Proc. BMVC*, 2005.

[SK05] Y. Shi and W. C. Karl. "A Fast Level Set Method Without Solving PDEs." In *IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing*, 2005.

[SKK07] Y. A. Sheikh, E. A. Khan, and T. Kanade. "Mode-seeking by medoidshifts." In *Proc. CVPR*, 2007.

[SM71] R. N. Shepard and J. Metzler. "Mental rotation of three-dimensional objects." *Science*, **171**:701–703, 1971.

[SM06] R. Subbarao and P. Meer. "Nonlinear Mean Shift for Clustering over Analytic Manifolds." In *Proc. CVPR*, 2006.

[SS98] Robert E. Schapire and Yoram Singer. "Improved boosting algorithms using confidence-rated predictions." In *Computational learning theory*, pp. 80–91, 1998.

[SS02] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.

[ST99] N. Slonim and N. Tishby. "Agglomerative Information Bottleneck." In *Proc. NIPS*, 1999.

[SVL92] P. Simard, B. Victorri, Y. LeCun, and J. Denker. "Tangent Prop-A Formalism for Specifying Selected Invariances in an Adaptive Network." *Advances in Neural Information Processing Systems 4*, pp. 895–903, 1992.

[SWB07] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. "Robust Object Recognition with Cortex-Like Mechanisms." *PAMI*, 2007.

[SYM07] G. Sundaramoorthy, A. Yezzi, and A. Mennucci. "Sobolev Active Contours." *Int. J. Comput. Vision*, **73**(3), 2007.

[TB99] M. E. Tipping and C. M. Bishop. "Probabilistic Principal Component Analysis." *Journal of The Royal Statistical Society, Series B*, **61**(3), 1999.

[TD03] P. H. S. Torr and C. Davidson. "IMPSAC: synthesis of importance sampling and random sample consensus." *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **25**(3):354–364, March 2003.

[TM02] B. Tordoff and D. Murray. "Guided sampling and consensus for motion estimation." In *Proc Euro. Conf. on Computer Vision*, pp. 82–96, May 2002.

[Tri04] B. Triggs. "Detecting Keypoints with Stable Position, Orientation, and Scale under Illumination Changes." In *Proc. ECCV*, 2004.

[TS07] T. Tuytelaars and C. Schmid. "Vector Quantizing Feature Space with a Regular Lattice." In *Proc. ICCV*, 2007.

[Tsi95] J. N. Tsitsiklis. "Efficient Algorithms for Globally Optimal Trajectories." *IEEE Trans. on Automatic Control*, **40**(9), 1995.

[TZ00] P. H. S. Torr and A. Zisserman. "MLESAC: A new robust estimator with application to estimating image geometry." *Computer Vision and Image Understanding*, **78**:138–156, 2000.

[Vap95] V. N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

[Ved08] A. Vedaldi. "A Ground-Truthed Dataset For Evaluation and Learning of Viewpoint Invariant Features." `http://vision.ucla.edu/gtvpi`, 2008.

[VF08] A. Vedaldi and B. Fulkerson. "VLFeat: An Open and Portable Library of Computer Vision Algorithms." `http://vision.ucla.edu/vlfeat`, 2008.

[VJ04a] P. Viola and M. Jones. "Face Database." `http://www.cs.ubc.ca/~pcarbo/viola-traindata.tar.gz`, 2004.

[VJ04b] P. Viola and M. Jones. "Robust Real-time Face Detection." *IJCV*, 2004.

[VS05] A. Vedaldi and S. Soatto. "Features for Recognition: Viewpoint Invariance for Non-Planar Scenes." In *Proc. ICCV*, 2005.

[VS06] A. Vedaldi and S. Soatto. "Viewpoint Induced Deformation Statistics and the Design of Viewpoint Invariant Features: Singularities and Occlusions." In *Proc. ECCV*, 2006.

[WAB03] J. Wills, S. Agarwal, and S. Belongie. "What Went Where." In *Proc. CVPR*, 2003.

[WB07] S. A. J. Winder and M. Brown. "Learning Local Image Descriptors." In *Proc. CVPR*, 2007.

[WQ04] Y. Wei and L. Quan. "Region-Based Progressive Stereo Matching." In *Proc. CVPR*, 2004.

[WWP00] M. Weber, M. Welling, and P. Perona. "Unsupervised Learning of Models for Recognition." In *Proc. ECCV*, 2000.

[YDG03] C. Yang, R. Duraiswami, N. A. Gumerov, and L. Davis. "Improved Fast Gauss Transform and Efficient Kernel Density Estimation." In *Proc. ICCV*, 2003.

[YL07] X. Yuan and S. Z. Li. "Half Quadric Analysis for Mean Shift: with Extension to A Sequential Data Mode-Seeking Method." In *Proc. CVPR*, 2007.

[YWS07] A. Yang, J. Wright, S. S. Sastry, and Y. Ma. "Unsupervised Segmentatiton of Natural Images via Lossy Data Compression." *CVIU (submitted)*, 2007.

[ZML06] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. "Local features and kernels for classification of texture and object categories: A comprehensive study." *IJCV*, 2006.

[ZS97] H. Zhuang and R. Sudhakar. "Simultaneous Rotation and Translation Fitting of Two 3-D Point Sets." *IEEE Trans. on Systems, Man, and Cybernetics*, **27**(1), 1997.

[ZSG06] S. K. Zhou, J. Shao, B. Georgescu, and D. Comaniciu. "BoostMotion: Boosting a discriminative similarity function for motion estimation." In *Proc. CVPR*, 2006.

[ZY96] S.-C. Zhu and A. Yuille. "Region Competition: Unifying Snakes, Region Growing, and Bayes/MDL for Multiband Image Segmentation." *PAMI*, **18**(9):884–900, 1996.