ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG
INSTITUT FÜR INFORMATIK
Lehrstuhl für Mustererkennung und Bildverarbeitung

Rotation Invariant Object Detection with Matrix-Valued Kernels

Internal Report 1/08

Marco Reisert

January 2008

Rotation Invariant Object Detection with Matrix-Valued Kernels

Marco Reisert
Computer Science Department
Albert-Ludwigs-University Freiburg
79110 Freiburg, Germany
reisert@informatik.uni-freiburg.de

January, 2008

Abstract

This article presents a rotation invariant object detection system. We propose two main contributions. On the one hand we fuse two different approaches, the 'Parts and Structure'-model and the generalized Hough Transform. Local image patches cast votes for the locations of the parts of the searched object. Then, an explicit model is fitted to the estimated density functions for the different parts.

Secondly, we propose a parametric voting scheme that is based on the rotation equivariant matrix kernel framework. Equivariance means that, if a local appearance patch is rotated, then the spatial density for the object part position has to be rotated accordingly. Traditionally, this is accomplished by a normalization procedure together with an non-parametric voting scheme. We propose to model the mapping from the local appearance patch onto the spatial density by a equivariant matrix kernel machine. Therefore we model the spatial densities in a parametric manner. To compute this equivariant matrix kernel efficiently we propose a new type of steerable feature.

We compare our system to a classical non-parametric voting scheme and the state-of-the-art features. The experimental tasks are the detection of conifer pollen in microscopical images and the detection of airplanes in aerial photographs.

1 Introduction

This work addresses the problem of 2D object detection in a rotation invariant manner. Given some training samples of an object class, the goal is to find all instances of the object in an unknown scene independent of its position and orientation in the image plane. There are various application domains. They typically appear in the context of microscopy or remote sensing. Examples are medical applications, problems in structural biology or X-ray airport inspection systems.

One can distinguish between three main directions in object detection research. First we want to mention the classical approach. The idea is to detect the objects by a matched filter. A correlation coefficient between a template object and every possible image patch is computed. For rotation invariant detection the template is compared in all possible rotational poses. Such approaches are widely used in structural biology to count particles (see e.g. [Nicholson 01, Nicholson 04]). Obviously such exhaustive approaches are time consuming. The 'Viola-Jones' detector [Viola 01] overcomes this by a boosted cascade of Haar-like features which can efficiently be computed by the integral image. But the original algorithm is not rotation invariant and rotation-invariant generalizations [Barczak 05] are not straight-forward.

Secondly, we have the often called object category/class detection or recognition systems. Their aim is to learn in a semi- or unsupervised manner object categories. An object category is typically a collection of visually similar objects. For example, such systems try to learn the category of guitars or cars. The intra category variations are usually very large, e.g. it does not matter whether there is an acoustic or electronic guitar or what color or particular shape the guitars have. To learn such variations the training sets must be rather large. Most of such approaches are invariant to the position of the searched object, some of them are also invariant to scale-changes, but it is hard to find approaches which are invariant to rotations or reflections. Of course, a complete invariance to rotations if often unwanted, but a robustness is certainly desired. The robustness is achieved by presenting a large amount of training data to the system. The main idea of such systems is relying on the 'Parts and Structure' model introduced by Fischler and Elschlager [Fischler 73]. The key idea is the separation of 'semantic' and 'syntactic' information. Following [Fischler 73], "The semantic information, which is application dependent, is embodied in the specific portioning of the reference into coherent pieces", while "the syntactic information, which is relatively independent of the particular application, defines the class of description which the algorithm can process". In more recent terms (see e.g. [Fergus 03]) the se-

mantic information is referred to as the shape or model information. The syntactic information is often called the appearance information, which is represented by the description of local image patches with appropriate features. Modern instances of this approach are e.g. by Fergus et al. [Fergus 03, Fergus 05] or others [Torralba 04, Jurie 04, Felzenszwalb 05, Crandall 06, Schmid 96, Leung 98] to mention a few.

Finally, we have the voting based approaches, which are essentially based on the idea of the General Hough Transform (GHT) [Ballard 81] by Ballard. Several modern instances of the idea inherit the rotation invariance from the original GHT, for example [Yokono 04, Zhu 03, Aguado 02]. Also the SIFT [Lowe 04] approach by Lowe can be interpreted as some kind of Hough Transform. Most of these approaches are template-based, this means that already one training sample is enough to detect the object in a cluttered scene. The key idea is to learn from the local appearance of the object a most probable hypothesis for e.g. the center, orientation or scale of the object. To achieve rotation invariance the features describing the local appearance are usually steered with some local directional quantity. A more recent approach of this type is the Implicit Shape Model (ISM) by Leibe et al. [Leibe 04]. In contrast to the classical GHT-based approaches, the system is able to cope with larger intra-class variations by learning the probability for the object center given an appearance patch in a non-parametric way. To become efficient a codebook of local appearances is created and each entry is attributed with the spatial density for the center of the object. To get the final voting map all possible object positions get a vote weighted by the conditional probabilities corresponding to the current local patches. Originally, the ISM is not rotation invariant. An invariant generalization can be found in [Mikolajczyk 06].

This article has two main contributions. An object detection system is proposed that fuses the two latter ideas, the GHT-based approach by Ballard with the 'Parts and Structure'-model by Fischler and Elschlager. Secondly, we propose a new GHT voting scheme that is based on the equivariant matrix kernel framework. Contrarily to most of the former object detection systems that cast votes in a non-parametric way we propose a parametric voting scheme. To evaluate the equivariant matrix kernels efficiently we develop a new type of feature that is suited for the integration into the equivariant matrix kernels.

As already mentioned the first main contribution is the fusion of the first two mentioned ideas. On the one hand we keep the idea that the object consists of different parts that are modeled explicitly. On the other hand we gather the evidence for the presence of the object parts by a probabilistic voting scheme in the spirit of the GHT, while preserving the rotation invariance. Instead of interpreting the

patches around the located keypoints as object parts, the idea is to cast votes for the object part positions regarding the local appearance of the patches. In contrast to other approaches where the evidence for the presence of the parts directly rely on the keypoint's position, this approach is much more robust against the actual keypoint's location. It also preserves the flexibility and robustness of the Implicit Shape Model, i.e. the objects' parts are still modelled implicitly, which makes the approach more tolerant against articulations. But we also gain more reliability of the decision by verifying it with an explicit model in a second step.

The second key contribution is the rotation equivariant learning of the voting function, that is the conditional probability for a object part position given a local appearance patch. In this case equivariance means that, if the local appearance patch is rotated, then the spatial density for the object part position has to be rotated accordingly. Traditionally, this is accomplished by a normalization or steering procedure together with an unparametric voting scheme. We propose to model this mapping from the local appearance patch onto the spatial density by a matrix kernel machine, while the equivariance is implicitly granted by the equivariance of the matrix kernels. To compute this equivariant matrix kernel efficiently we propose a new type of steerable (or equivariant) feature. This feature may be seen as the steerable version of the SIFT [Lowe 04] or GLOH [Mikolajczyk 05] features. The training algorithm is an ordinary kernelized regression with a quadratic loss. This has basically two reasons. It can be motivated from a maximum likelihood principle, and the second reason is that the quadratic loss is unitary invariant which is a necessary demand to get reasonable results in the presence of the equivariance constraint.

The article is organized as follows. Section 2 proposes the equivariant kernel framework. Three different types of equivariant kernels are proposed together with their implementations. In the following section the feature extraction stage is explained. In Section 4 we develop the object model in a probabilistic manner, where we still want to retain the generative nature of the GHT idea, i.e. only statistics about the object of interest is considered. Also the invariance demands are incorporated in the probabilistic framework. By interpreting the conditional density for the object part position given a local descriptor as a functional that maps the local descriptor onto a density function, we are able to reformulate the rotation invariance as an equivariance constraint. Section 5 proposes how the equivariant voting functions are parameterized and how they are trained by a kernelized regression scheme. The spatial relationships between the object parts are modelled by a full joint Gaussian model such that the training just involves ordinary estimation of covariance and mean. In Section 6 we explain how the part specific

voting maps are rendered and how the Gaussian model is fitted to them. In the experimental section we apply our system to two tasks, aircraft detection in aerial images and pollen detection in microscopical images. We compare our approach to standard non-parametric voting schemes and to the so called GLOH-features.

2 Equivariant Kernels for 2D rotations

First we give a short introduction into the equivariant kernel framework. Let X and Y be two Hilbert spaces. Our primary goal is to learn equivariant functions $\mathbf{f} : X \rightarrow Y$, that is

$$\mathbf{f}(\boldsymbol{\tau}_g^X \mathbf{x}) = \boldsymbol{\tau}_g^Y \mathbf{f}(\mathbf{x})$$

holds for all $\mathbf{x} \in X$ and $g \in SO(2)$, where $\boldsymbol{\tau}_g^X \in GL(X)$ denotes the group action in the input space of $\mathbf{f}$ and $\boldsymbol{\tau}_g^Y \in GL(Y)$ the group action on the output space. We assume that both group actions are unitary. The functions $\mathbf{f}$ will later model the mapping from the local appearance patch $\mathbf{x}$ onto a spatial probability density for the center or a part of the searched object.

The reproducing kernel of a vector-valued kernel Hilbert spaces is a matrix-valued function $\mathbf{K} : X \times X \rightarrow L(Y)$ (for definition see [Reisert 07] or [Micchelli 05]). Reproducing kernels of $SO(2)$-equivariant function spaces additionally fulfill the following equivariance property

$$\mathbf{K}(\boldsymbol{\tau}_g^X \mathbf{x}_1, \boldsymbol{\tau}_h^X \mathbf{x}_2) = \boldsymbol{\tau}_g^Y \mathbf{K}(\mathbf{x}_1, \mathbf{x}_2) \boldsymbol{\tau}_{h^{-1}}^Y$$

for all $g, h \in SO(2)$ and $\mathbf{x}_1, \mathbf{x}_2 \in X$. One can easily verify that linear combinations with $\mathbf{a}_i \in Y$

$$\mathbf{f}(\mathbf{x}) = \sum_{k=0}^{N-1} \mathbf{K}(\mathbf{x}, \mathbf{x}^k) \mathbf{a}_k$$

are equivariant functions. Essentially, there are three different ways to obtain equivariant matrix kernels: group integration, normalization or matching. We will shortly explain the three principles.

2.1 Group Integration Matrix Kernels

Let $K_0 : X \times X \rightarrow \mathbb{R}$ be a scalar-valued unitary basis kernel. Then we can obtain by group integration

$$\mathbf{K}(\mathbf{x}_1, \mathbf{x}_2) = \int_{SO(2)} K_0(\mathbf{x}_1, \boldsymbol{\tau}_g^X \mathbf{x}_2) \boldsymbol{\tau}_g^Y \, dg$$

an equivariant matrix kernel. We call this kernel Group Integration Matrix-Kernel, shortly GIM-kernel. We introduced this construction principle in [Reisert 07]. In fact, this approach is optimal: the equivariant representer theorem [Reisert 07] states that any equivariant minimizer in the RKHS of K_0 has to be a linear combination of GIM-kernels.

2.2 Normalization Kernels

Secondly, we can use the principle of normalization to obtain an equivariant behavior. Assume, that we have given a normalizer function $N : X \rightarrow SO(2)$ that turns $\mathbf{x}$ into a standard pose $\mathbf{x}_0$, that is $\boldsymbol{\tau}^X_{N(\mathbf{x})}\mathbf{x} = \mathbf{x}_0$. Such a normalizer function is characterized by the fact that $N(\boldsymbol{\tau}_g\mathbf{x}) = g^{-1}\, N(\mathbf{x})$. Then, we can show that

$$\mathbf{K}(\mathbf{x}_1, \mathbf{x}_2) = (\boldsymbol{\tau}^Y_{N(\mathbf{x}_1)})^{-1}\, K_0(\boldsymbol{\tau}^X_{N(\mathbf{x}_1)}\mathbf{x}_1, \boldsymbol{\tau}^X_{N(\mathbf{x}_2)}\mathbf{x}_2)\, \boldsymbol{\tau}^Y_{N(\mathbf{x}_2)}$$

is an equivariant kernel. The kernel K_0 is again a unitary scalar basis kernel. In fact, this kernel is also positive definite, we call it Normalization kernel, or N-kernel.

An inherent problem with the normalization approach is that reasonable normalizer procedures are discontinuous and give not always unique responses. Usually the normalization algorithm works as follows: a scalar feature $n(\mathbf{x})$, usually a gradient direction histogram of the patch $\mathbf{x}$, is maximized over all possible poses, that is

$$N(\mathbf{x}) = \operatorname*{argmax}_{g \in SO(2)} n(\boldsymbol{\tau}^X_g \mathbf{x})$$

By no means it it is assured that the maximum is unique. And even if is unique, there may be other local maxima that are very close to the global one. Thus, small distortion may cause discontinuous jumps of N that may be hazardous for further steps. To partially avoid such problems the following solution is proposed in the literature. All those local maxima are selected whose values are above e.g. 85% of the global maximum. This idea can also be easily incorporated in our kernel framework. Let N' be a set valued normalizer, i.e. N' returns a set of transformations $N'(\mathbf{x}) = \{g_1, g_2, \ldots\}$ that all normalize $\mathbf{x}$ to certain standard poses. Then, an equivariant kernel is given by

$$\mathbf{K}(\mathbf{x}_1, \mathbf{x}_2) = \sum_{\substack{g \in N'(\mathbf{x}_1) \\ h \in N'(\mathbf{x}_2)}} (\boldsymbol{\tau}^Y_g)^{-1}\, K_0(\boldsymbol{\tau}^X_g \mathbf{x}_1, \boldsymbol{\tau}^X_h \mathbf{x}_2)\, \boldsymbol{\tau}^Y_h, \tag{1}$$

This approach incorporates very gently known techniques that were introduced ad-hoc in the context of object detection and localization [Mikolajczyk 06, Teynor 07, Lowe 04] into the equivariant kernel framework that is the basis of this work.

2.3 Maximum Matching Kernels

The third alternative is a registration or matching procedure. Instead of specifying a normalized pose of the considered patterns beforehand, the patterns are compared in all possible poses 'online' during the kernel evaluation. We again need some basis kernel K_0. Then,

$$\mathbf{K}(\mathbf{x}_1, \mathbf{x}_2) = K_0(\mathbf{x}_1, \boldsymbol{\tau}^X_{g^*}\mathbf{x}_2)\, \boldsymbol{\tau}^Y_{g^*}, \quad \text{where } g^* = \operatorname*{argmax}_{g \in \mathcal{G}} K_0(\mathbf{x}_1, \boldsymbol{\tau}^X_g \mathbf{x}_2)$$

is an equivariant indefinite kernel. We call such type of kernel Maximum Matching kernel, shortly MM-kernel. The MM-kernel approach has similar problems as the N-kernel. The maximum of the matching function $K_0(\mathbf{x}_1, \boldsymbol{\tau}^X_g \mathbf{x}_2)$ does not have to be unique and robust against small distortions of the input patterns. In fact, the same solution as for the N-kernel can also help to weaken this effect. Instead of just taking the maximum of the matching function $K_0(\mathbf{x}_1, \boldsymbol{\tau}^X_g \mathbf{x}_2)$ we take all local maxima that are within a range to the global one and average over all those.

2.4 Implementation

Both, the GIM-kernel and the MM-kernel, rely on the computation of a generalized cross correlation function

$$c(\phi) = K_0(\mathbf{x}_1, \boldsymbol{\tau}^X_\phi \mathbf{x}_2),$$

where ϕ denotes the rotation angle. It is well known that a fast computation of the cross correlation function is possible in the Fourier domain. Thus, we will introduce in the next section a novel local feature whose angular component is represented in the Fourier domain which allows a fast computation of $c(\phi)$.

By now, assume that the cross correlation $c(\phi)$ is given, and another issue is the representation of the output space Y. If the components of the output are also given in the Fourier domain, the computation of the GIM-kernel is quite easy. Because then, the representation $\boldsymbol{\tau}^Y_\phi$ of the group is a diagonal matrix with entries $e^{\mathbf{i}l\phi}$ on the diagonal. Hence, the GIM-kernel is also diagonal with the entries

$$K^l_{\text{GIM}}(\mathbf{x}_1, \mathbf{x}_2) = \int_0^{2\pi} c(\phi) e^{\mathbf{i}l\phi}\, d\phi$$

Thus, the GIM-kernel is just the inverse Fourier transform of the cross correlation function $c(\phi)$. Practically, the transformation is accomplished by the Fast Fourier Transform (FFT).

The MM-kernel algorithm is also as simple if we choose a Fourier representation in the output space. Just search for the set of maxima of the cross correlation function, call them ϕ_i^*, and return $e^{\mathbf{i}l\phi_i^*}$ weighted by the maximum as the lth kernel entry, that is

$$K^l_{\text{MM}}(\mathbf{x}_1, \mathbf{x}_2) = \sum_i c(\phi_i^*) e^{\mathbf{i}l\phi_i^*}$$

where the sum ranges over all local maxima of $c(\phi)$ that are above 85% of the global one.

For the implementation of the N-kernel we do not need to compute the cross correlation. During feature extraction we just have to steer the orientation of the feature with respect to some stable quantity, we used the maxima of the local gradient orientation histogram and save this orientation together with the extracted feature. Local feature extractors like SIFT or GLOH are based on this idea. To evaluate the kernel we just have to compute some similarity measure between the steered features and multiply it with the corresponding group actions, that is

$$K^l_{\text{N}}(\mathbf{x}_1, \mathbf{x}_2) = \sum_{i,j} K_0(\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(j)})\ e^{\mathbf{i}l(\phi_j - \phi_i)},$$

where $(\mathbf{x}_1^{(i)}, \phi_i)$ and $(\mathbf{x}_2^{(j)}, \phi_j)$ denote the set of steered features together with their steering orientations.

3 Rotation Equivariant Features in 2D

3.1 Keypoint Detection

The very first stage of the object detector is covered by a rotation invariant keypoint detector. We use a detector that is not invariant against scale changes. This has basically two reasons. The application domains that we consider later do not demand for a scale invariant treatment. For example, pollen grains have a typical size. Particles of different sizes, even if they look similar have to be rejected a priori. And secondly, there is a big disadvantage if we consider scale invariant keypoints. They are very rare in comparison to just rotation invariant keypoints. This makes particular problems if the objects are very small and have no inner

texture or structure. In the experiments we want to detect objects whose sizes are from about 20 up to 60 pixels. So, just keypoints below a scale of 60 pixels are important.

We use a curvature based key point detector [Lindeberg 98]. If we represent the image by an image function $\mathbf{I} : \mathbb{R}^2 \rightarrow \mathbb{R}$, then the used saliency map can be written as

$$S(\mathbf{I}) = (\nabla \mathbf{I})^\top H(\mathbf{I})(\nabla \mathbf{I}) - \frac{1}{2}\mathrm{Tr}(H(\mathbf{I}))\|\nabla \mathbf{I}\|$$

where $H(\mathbf{I})$ is the Hessian[1] of the image and $\nabla \mathbf{I}$ the gradient. Practically, the Hessian and the gradient are computed by finite differences on a slightly blurred version of the image $\mathbf{I}$. The proposed saliency map combines both, an edge and a corner detector, the gradient gives primarily response for edges while the Hessian gives response for corners.

The keypoints are located by a local maxima search on $S(\mathbf{I})$. The set of all detected keypoints is called $\mathcal{Z}$. Each keypoint $\vec{z} \in \mathcal{Z}$ is attributed with its local appearance patch $\mathbf{x}$. It describes the local neighborhood of the keypoint. We denote the location of the keypoints together with its local appearance patch by $\mathbf{X} = (\vec{z}, \mathbf{x})$. These are combined in the set

$$\mathcal{X} = \{\mathbf{X} \mid \vec{z} \in \mathcal{Z} \text{ and } (\vec{z}, \mathbf{x}) = \mathbf{X}\}.$$

The set $\mathcal{X}$ of local appearance patches will be the basis for all further considerations. We assume that the image $\mathbf{I}$ is completely described by $\mathcal{X}$.

3.2 Feature Extraction

One key issue of our detection system is the rotation equivariant modelling of the voting function $\mathbf{f}$ that casts votes for the object center and parts, respectively. We will obtain the equivariance by the use of the proposed equivariant kernel framework. We will consider Group Integration Matrix-kernels (GIM-kernel), Normalization-kernels (N-kernels) and Maximum Matching-kernels (MM-kernels). We have already mentioned that the evaluation of GIM-kernels and MM-kernels both need to compute cross-correlations between the incoming patterns. Thus, we

[1] If the image is given by $\mathbf{I}(\vec{u}) = \mathbf{I}(x, y)$ then the Hessian matrix $H \in \mathbb{R}^{2\times 2}$ is given by

$$H(\mathbf{I}) = \begin{pmatrix} \partial_{xx}\mathbf{I}(u, v) & \partial_{yx}\mathbf{I}(x, y) \\ \partial_{xy}\mathbf{I}(u, v) & \partial_{yy}\mathbf{I}(x, y) \end{pmatrix},$$

where ∂ denotes the partial derivative

need a representation that allows us to compute such cross-correlations efficiently. It is well known that the Fourier domain is well suited for this, because one can make use of the Fast Fourier Transform (FFT).

For each patch $\mathbf{x}$ we compute a feature vector $\boldsymbol{\Psi}(\mathbf{x})$. Let $\boldsymbol{\psi}_n^l(\mathbf{x})$ denote the components of this feature vector, where the index l corresponds to the Fourier representation and n is an additional feature dimension. We have two demands for the feature function. It should be discriminative and robust like e.g. SIFT [Lowe 04] or GLOH [Mikolajczyk 05] features and on the other hand we have to represent the angular direction of the feature in the Fourier domain to allow a fast computation of the cross correlation. The second demand can be expressed as

$$\boldsymbol{\psi}_n^l(\boldsymbol{\tau}_\phi^X \mathbf{x}) = e^{\mathbf{i}l\phi} \boldsymbol{\psi}_n^l(\mathbf{x}).$$

Note, that is property is nothing else than equivariance of the feature function with respect to rotations, or in other terms one may also call this feature a steerable feature, according to the steerability property known from image filters (see e.g. [Perona 95]).

Following the style of the SIFT features we propose a feature that is also based on the gradient of the image patch. We denote the gradient at position $\vec{u}$ as $\nabla\mathbf{x}(\vec{u})$. For convenience we assume that the position of the keypoint is shifted to the origin. Basically, the feature is a joint histogram over the distance from the origin and two angle-like quantities. In Figure 1 we visualize the configuration of the three quantities. The first angle is the cosine between the position vector $\vec{u}$ and the gradient direction $\nabla\mathbf{x}(\vec{u})$. The second one is the absolute angle of the position $\arg(\vec{u})$. We represent the latter histogram dimension in the Fourier domain. Formally, we can write the feature as

$$\boldsymbol{\psi}_n^l(\mathbf{x}) = \int\limits_{\|\vec{u}\|<d_{\max}} \delta\left(\frac{\vec{u}^\top \nabla\mathbf{x}(\vec{u})}{\|\vec{u}\|\|\nabla\mathbf{x}(\vec{u})\|} - a_n\right) \delta(\|\vec{u}\| - d_n) \|\nabla\mathbf{x}(\vec{u})\| \; e^{\mathbf{i}\arg(\vec{u})} d\vec{u},$$

where δ denotes the Dirac delta functions and a_n and d_n are the centers of the histogram bins for the distance and the relative angle, respectively.

The algorithm for the computation of the features is given in Algorithm 1. We call them Equivariant Gradient Histograms, shortly EGH-features.

As we are highly interested in small and compact features we examined how to keep the number of bins as low as possible. We found that an equiareal binning for the distance is advantageous. Equiareal means that each distance bin should

Algorithm 1 Equivariant Gradient Histogram

Input: A patch $\mathbf{x}(\vec{u})$ and a quantization scheme $Q : \mathbb{R} \times \mathbb{R} \to \mathbb{N}$

Output: Feature array $\boldsymbol{\psi}_n^l \in \mathbb{C}$ with $n = 0, \ldots, m-1$ and $l = 0, \ldots, l_{\max}$.

1: Initialize feature array $\boldsymbol{\psi}_n^l$ with zeros for all n and l.
2: **for** all pixels $\vec{u}$ in the gradient patch $\nabla\mathbf{x}$ **do**
3: Compute $d = \|\vec{u}\|$ and $a = \frac{\vec{u}^\top \nabla\mathbf{x}(\vec{u})}{\|\vec{u}\|\|\nabla\mathbf{x}(\vec{u})\|}$.
4: Quantize (d, a) into the discrete bin $n = Q(d, a)$.
5: Accumulate $\boldsymbol{\psi}_n^l = \boldsymbol{\psi}_n^l + \|\mathbf{x}(\vec{u})\| e^{\mathbf{i} l \arg \vec{u}}$ for all l.
6: **end for**

be responsible for the same amount of area in the patch and thus for the same number of pixels. We depicted this in Figure 1. The shaded areas correspond to different bins for d. As the area of the circle grows quadratically in d, we can achieve an equiareal binning by a traditional equidistant binning of $\sqrt{d}$. So we were able to choose between 4 and 6 bins for d while keeping the robustness and discriminativity high.

3.3 Computation of the Cross-Correlation

Based on these features we are able to compute the cross correlation $c(\phi) = K_0(\mathbf{x}_1, \boldsymbol{\tau}_\phi^X \mathbf{x}_2)$ in a fast manner. Therefore, we have to assume that K_0 is a dot-product kernel, meaning that it has the form

$$K_0(\mathbf{x}_1, \mathbf{x}_2) = \Gamma(\underbrace{\langle \boldsymbol{\Psi}(\mathbf{x}_1), \boldsymbol{\Psi}(\boldsymbol{\tau}_\phi^X \mathbf{x}_2) \rangle}_{c_{\text{linear}}(\phi)}),$$

where Γ is a nonlinear function. From standard kernel literature [Scholkopf 02] we know that applying any analytic function $\Gamma : \mathbb{R} \to \mathbb{R}$ with positive Taylor expansion coefficients to a kernel yields again a kernel. To obtain $c_{\text{linear}}(\phi)$ we have to compute an inner product separately for each frequency and apply on the result an inverse Fourier transform.

$$c_{\text{linear}}(\phi) = \sum_{l=0}^{l_{\max}} \left(\sum_{n=1}^{m} \boldsymbol{\psi}_n^l(\mathbf{x}_1) \overline{\boldsymbol{\psi}_n^l(\mathbf{x}_2)} \right) e^{\mathbf{i} l \phi}.$$

In practice, we use a Fast Fourier Transform (FFT) for the computation of the outer sum. In the experiments we have chosen the following four (non)-linearities

$$\Gamma_1(t) = t, \ \Gamma_2(t) = t^2, \ \Gamma_3(t) = e^{\lambda t}, \ \Gamma_4(t) = 1 + \lambda t + \lambda^2 t^2 / 2, \tag{2}$$

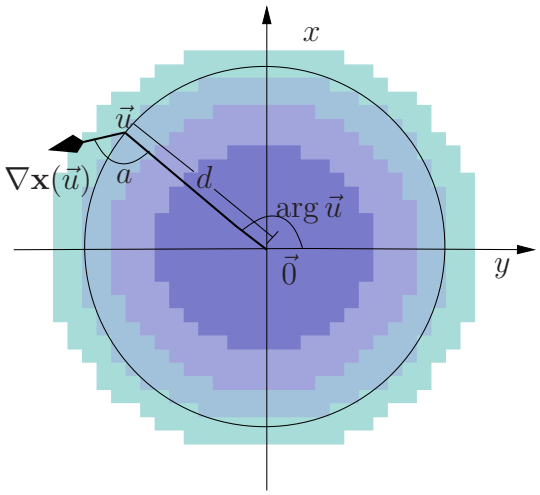


Figure 1: **The geometric interpretation of the feature computation.** For each pixel $\vec{u}$ the configuration is described by three quantities: The angle $\arg(\vec{u})$, the length $\|\vec{u}\|$ and the relative angle between the gradient $\nabla \mathbf{x}(\vec{u})$ and $\vec{u}$. The shaded areas indicate the histogram-bins for the length $\|\vec{u}\|$. In this case, 4 bins are distributed over a radius of 14 pixels, which is a typical choice in our experiments. The bin sizes are chosen such that approximately the same number of pixels are assigned to each bin.

where $\gamma \in \mathbb{R}$. The first one corresponds to a linear kernel, the second one to a quadratic kernel, the third one is an exponential kernel that is closely related to a Gaussian RBF kernel and the fourth is an approximation of the exponential kernel.

4 The Object Model

One key assumption is that the input image is completely characterized by the feature set $\mathcal{X}$. In practice, this assumption is reasonable if the number of keypoints is large enough. It further simplifies the following statistical considerations. We assume that our object consists of M different parts. Each part has a position $\vec{c}_j$. The whole configuration of all object parts is the vector consisting of the object positions $\vec{\mathbf{c}} = [\vec{c}_0, \ldots, \vec{c}_{M-1}]$. The goal of the detection process is to find a image portion $\mathcal{I} \subset \mathcal{X}$ and a configuration $\vec{\mathbf{c}}$ such that the posterior probability $p(\vec{\mathbf{c}} \,|\, \mathcal{I}, \theta)$ is maximized. That is the probability that the object configuration $\vec{\mathbf{c}}$ is observed when seen the featureset $\mathcal{I}$ for a given set of model parameters θ. In the following

we often omit the θ-dependency as it is mostly formal ballast. Always assume it as implicitly given, if necessary we explicitly refer to it.

Using Bayes' rule we can decompose the posterior as follows

$$p(\vec{\mathbf{c}} \,|\, \mathcal{I}) = p(\mathcal{I}|\vec{\mathbf{c}}) \frac{p(\vec{\mathbf{c}})}{p(\mathcal{I})},$$

which makes it possible to model the appearance term $p(\mathcal{I}|\vec{\mathbf{c}})$ and the shape prior $p(\vec{\mathbf{c}})$ independently. Following the terms of Fischler and Elschlager [Fischler 73], we decompose the problem into a 'syntactic' and 'semantic' part. The main independence assumption is that the likelihood of seeing the image $\mathcal{I}$ given the object parts decomposes into independent distribution over the single objectparts.

$$p(\mathcal{I}|\vec{\mathbf{c}}) = \prod_{j=0}^{M-1} p(\mathcal{I}|\vec{c}_j)$$

Until now we have followed the common statistical framework known from active shape models or object recognition, see for example [Felzenszwalb 05]. Usually the distributions $p(\mathcal{I}|\vec{c}_j)$ are now directly modelled. But we want to point out an alternative way. To apply the idea of gathering the evidence for the presence of object parts by a voting scheme, Bayes' theorem is applied again on the densities for the parts. Following this we obtain

$$p(\vec{\mathbf{c}} \,|\, \mathcal{I}) = p(\vec{\mathbf{c}}) \prod_{j=0}^{M-1} \frac{p(\vec{c}_j \,|\, \mathcal{I})}{p(\vec{c}_j)}.$$

Due to the invariance constraints the absolute positions of the parts have to be ignored, so we neglect the marginals $p(\vec{c}_j)$ in further considerations. Considering the densities $p(\vec{c}_j \,|\, \mathcal{I})$ the idea of voting for the presence of object parts becomes obvious. Interpreting $\mathcal{I}$ as a set of samples drawn from a underlying density $p(\mathbf{X})$ representing the image under consideration, we make the following estimation. The probability for a object part's position can be written as $\int p(\vec{c}_j|\mathbf{X})p(\mathbf{X})d\mathbf{X}$ by the law of the total probability. We can approximate this by

$$p(\vec{c}_j \,|\, \mathcal{I}) = \sum_{\mathbf{X} \in \mathcal{I}} p(\vec{c}_j \,|\, \mathbf{X}) \tag{3}$$

In conclusion, we have to learn the distributions $p(\vec{c}_j \,|\, \mathbf{X})$ that perform the votes for the object parts, and $p(\vec{\mathbf{c}})$ that priors the shape in terms of the configuration of the object parts. Both have to fulfill several invariance constraints that are discussed in the following.

4.1 Invariance of the Voting Function

First we investigate the distributions which vote for the centers of the object parts. They have to be invariant against rotations and translations:

$$\begin{aligned} \textit{Translation:} \quad p(\vec{c}\,|\,(\vec{z},\mathbf{x})) &= p(\vec{c}+\vec{t}\,|\,(\vec{z}+\vec{t},\mathbf{x})) \\ \textit{Rotation:} \quad p(\vec{c}\,|\,(\vec{z},\mathbf{x})) &= p(R_\varphi\vec{c}\,|\,(\vec{z},\boldsymbol{\tau}_\varphi^X\mathbf{x})) \end{aligned}$$

Invariance against translations can easily be obtained by just learning the probabilities relative to the position of the observed keypoint, i.e. we normalize the distribution and just learn $p(\vec{c}-\vec{z}\,|\,(\vec{0},\mathbf{x}))$. For the rotation it is different. We do not have any anchor point to make the angular coordinates 'absolute' in some way. In the SIFT framework by Lowe [Lowe 04] local maxima of the gradient orientation histogram are used to normalize the features with respect to the angular coordinates. We want to achieve the invariance alternatively by the use of equivariant kernel methods.

To apply the equivariant kernel framework we reformulate the invariance constraint. Interpreting $p(\vec{c}\,|\,(\vec{0},\mathbf{x}))$ as a functional that is mapping a patch $\mathbf{x}$ onto a probability distributions over $\vec{c}$, the rotation invariance is translated to an equivariance constraint for this functional. Let us formulate this more precisely. We define a function $\mathbf{f} : X \rightarrow Y$ that takes a local descriptor $\mathbf{x}$ and maps it onto a vectorial output $\mathbf{f}(\mathbf{x}) \in Y$. The components of the output are interpreted as probabilities for the occurrence of the object part at a specific location. That is, each component of this vector is indexed by a position $\vec{c}$. To access these components of the vector we just have to compute an inner product with an unit vector given its only contribution at one specific entry that is indexed by $\vec{c}$. More formally, we have

$$\mathbf{e}_{\vec{c}}^\top\mathbf{f}(\mathbf{x}) = p(\vec{c}\,|\,(\vec{0},\mathbf{x})).$$

By $\mathbf{e}_{\vec{c}}$ we have denoted the unit vector that selects the appropriate entry. To understand that the rotation invariance of the density p is equivalent to the equivariance of the function $\mathbf{f}$ we consider a rotation g by angle φ around the origin,

$$\begin{aligned} p(\vec{c}\,|\,(\vec{0},\boldsymbol{\tau}_\varphi^X\mathbf{x})) &= \mathbf{e}_{\vec{c}}^\top\mathbf{f}(\boldsymbol{\tau}_\varphi^X\mathbf{x}) \overset{\text{eq.}}{=} \mathbf{e}_{\vec{c}}^\top\boldsymbol{\tau}_\varphi^Y\mathbf{f}(\mathbf{x}) \\ &\overset{\text{uni.}}{=} (\boldsymbol{\tau}_{\text{-}\varphi}^Y\mathbf{e}_{\vec{c}})^\top\mathbf{f}(\mathbf{x}) \overset{\text{def.}}{=} (\mathbf{e}_{R_\varphi^{\text{-}1}\vec{c}})^\top\mathbf{f}(\mathbf{x}) = p(R_\varphi^{\text{-}1}\vec{c}\,|\,(\vec{0},\mathbf{x})) \end{aligned}$$

which is equivalent to the rotation invariance of p. First we have used the equivariance of $\mathbf{f}$ and then the unitarity of the rotation and then the definition that $\mathbf{e}_{\vec{c}}$ is a vector that evaluates at position $\vec{c}$.

4.2 Invariance of the Shape Density

Also the shape prior $p(\vec{\mathbf{c}})$ has to fulfill translation and rotation invariance, namely

$$p(\vec{\mathbf{c}}) = p(R_\varphi \vec{\mathbf{c}} + \vec{t}\mathbf{1}),$$

where $R_\varphi \vec{\mathbf{c}} + \vec{t}\mathbf{1}$ denotes a rotation and translation of the whole configuration. To model $p(\vec{\mathbf{c}})$ we want to use a full joint gaussian model. But first, consider the invariance constraints. To include them we make the expansion

$$p(\vec{\mathbf{c}}) = p(\vec{\mathbf{c}}, \vec{v}, \phi) = p(\vec{\mathbf{c}} \,|\, \vec{v}, \phi)\; p(\vec{v}, \phi). \tag{4}$$

where $\vec{v}$ denotes the 'center' of the configuration and ϕ the orientation. Both are deterministically dependend on $\vec{\mathbf{c}}$. Because no orientation and position of the objects should be favored one has to neglect $p(\vec{v}, \phi)$. Due to the invariance constraint the remaining part has to behave like

$$p(\vec{\mathbf{c}} \,|\, \vec{v}, \phi) = p(e^{-\mathrm{i}\phi}\vec{\mathbf{c}} - \vec{v}\mathbf{1} \,|\, \vec{0}, 0)$$

and hence we only have to learn $p(\vec{\mathbf{c}} \,|\, \vec{0}, 0)$ where $\vec{\mathbf{c}}$ is constraint to be in a fixed position and orientation.

5 Training of the Model

We assume that a set of training images is given, where in each image the training objects are marked by bounding boxes and the object parts are labeled by their absolute positions. It is assumed that all images have the same orientation. The learning process is two fold, on the one hand we have to estimate the densities $p(\vec{c}_j \,|\, \mathbf{X})$ for each object part $\vec{c}_j$ and on the other hand the shape prior $p(\vec{\mathbf{c}})$ that covers the interrelations between the object parts.

Consider one training object marked by a bounding box. We select all keypoints $\vec{z}^k$ and their associated features $\mathbf{x}^k$ that lie inside the bounding box. Secondly, we gather the M labeled part positions $\vec{c}_j$ of the object and compose thereof M object specific training sets, each consisting of all tuples $(\vec{c}_j, \mathbf{X}^k) = (\vec{c}_j, (z^k, \mathbf{x}^k))$ for all $\mathbf{X}^k$ that lie inside the bounding box. To train $p(\vec{c}_j \,|\, \mathbf{X}) = p(\vec{c}_j \,|\, (\vec{z}, \mathbf{x}))$ we union all object specific training sets and obtain for each object part j one training set $\mathcal{T}_j = \{(\vec{c}_j^{\,k}, \mathbf{X}^k) \mid k = 0, \ldots, N-1\}$. The number N denotes the total number of training patches.

We already mentioned that we want to model the densities $p(\vec{c}_j \mid \mathbf{X})$ by a function $\mathbf{f}_j : X \rightarrow Y$ that gets as input a local appearance patch and returns a probability density over the object part position $\vec{c}_j$. We model this density parametrically by the use of a circular harmonic expansion such that their representations are compatible with the proposed equivariant matrix kernels. The function $\mathbf{f}$ is trained by an ordinary kernelized ridge regression scheme with a quadratic loss.

Secondly, we have to train the shape prior $p(\vec{\mathbf{c}})$. Each training object together with its part positions $\vec{\mathbf{c}} = [\vec{c}_0, ..., \vec{c}_{N-1}]$ serve as one training instance. The shape prior $p(\vec{\mathbf{c}})$ will be modelled by a simple joint gaussian model, i.e. the training procedure is just the usual estimation of mean and covariance of the object part positions.

5.1 Parametric Representation of the Voting Functions

As already depicted in Section 4.1 we can interpret the conditional probability function as some vector valued function $\mathbf{f}(\mathbf{x})$, where the vector entries correspond to probabilities for $\vec{c}$. We omit here the dependency on the part number j, because the considerations are for all object parts the same. Recall that the output domain of the function $\mathbf{f}$ are a spatial probability density, the probability values are accessed by an inner product with an unit vector, i.e. $\mathbf{e}_{\vec{c}}^{\top}\mathbf{f}(\mathbf{x}) = p(\vec{c} \mid (\vec{0}, \mathbf{x}))$. The question is, what is an appropriate parametric model for this density. To make the output of $\mathbf{f}$ compatible with the proposed matrix kernels we make a decomposition into an radial and an angular part as mentioned before. We propose to use the following representation:

$$\mathbf{e}_{\vec{d}}^{\top}\mathbf{f}(\mathbf{x}) = \sum_{l=-l_{\max}}^{l_{\max}} \sum_{i=0}^{E-1} f^{l,i}(\mathbf{x})\; e^{\mathbf{i}l\arg(\vec{d})}\; \epsilon_i(\|\vec{d}\|), \tag{5}$$

where $\arg(\vec{d})$ is the angle of the vector $\vec{d}$. The functions ϵ_i are triangular shaped envelope functions only depending on the distance $d = \|\vec{d}\|$, i.e.

$$\epsilon_i(d) = \begin{cases} \frac{r_i - d}{\Delta} & \text{for} \quad r_i - \Delta < d < r_i \\ 1 - \frac{r_i - d}{\Delta} & \text{for} \quad r_i < d < r_i + \Delta \\ 0 & \text{otherwise} \end{cases},$$

where the r_i are fixed centers of the radial parts ϵ_i. The function ϵ_0 in the center is treated specifically to avoid a hole around the origin.

The expansion coefficients $f^{l,i}(\mathbf{x})$ can be interpreted as a representation of $\mathbf{f}$ in polar coordinates, where the part depending on the angle is expanded in Fourier

domain, corresponding to index the l, and the radial part, corresponding to the index i. In Figure 2 we show two examples. The functions are sinusodials of angular frequency 2 and 3 that are swapping around the origin of the plane. For example, the upper function in Figure 2 has its only entries at $f^{2,4} = f^{-2,4} = 1$.

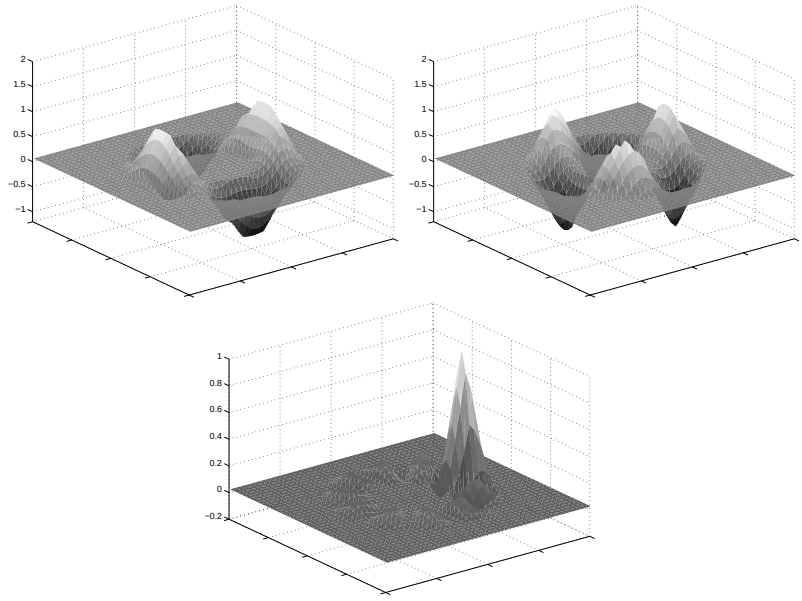


Figure 2: **Example for the modelling of the voting function.** The graph on the left corresponds to a function with $f^{l,i} = 0$ except for $l = \pm 2$ at one particular distance, specified by the index i. The function in the middle has only entries at $l = \pm 3$. The function on the right is an example for the unit vector $\mathbf{e}_{\vec{c}}$. The Fourier expansion is truncated after 10 terms. One can clearly the artefacts that produced by this truncation.

We have to ask what are the expansion coefficients $e^{l,i}_{\vec{c}}$ of the unit vector $\mathbf{e}_{\vec{c}}$ in this representation. They will play later an essential role in the training stage. Interpreted as a probability density it is just the density of the certain event at position $\vec{c}$, that is, a density that has clear peak at position $\vec{c}$. We cannot expect from the representation proposed in equation (5) that it can model arbitrary fine peaks at some position $\vec{c}$. Due to the fixed discretization introduced by the triangular shaped radial functions it can even happen that peaks at different positions will show different artefacts. To get a smooth representation and a symmetric

interpolation kernel we choose

$$\mathbf{e}_{\vec{c}}^{\top}\mathbf{e}_{\vec{d}} = \sum_{l=-l_{\max}}^{l_{\max}} \sum_{i=0}^{E-1} \underbrace{\epsilon_i(\|\vec{c}\|)e^{-\mathbf{i}l\arg(\vec{c})}}_{e_{\vec{c}}^{l,i}} \; e^{\mathbf{i}l\arg(\vec{d})} \; \epsilon_i(\|\vec{d}\|)$$

In Figure 2 we show how a unit vector $\vec{e}_{\vec{c}}$ looks like. We used a Fourier expansion up to $l_{\max} = 10$. We can clearly see the artefacts that are created by this truncation.

5.2 Kernelized Regression

The key issue is to learn the conditionals $p_\theta(\vec{c}\,|\,\mathbf{X}) = p(\vec{c}\,|\,\mathbf{X}, \theta)$ from a set of samples $\mathcal{T} = \{(\vec{c}^{\,k}, \mathbf{X}^k) \mid k = 0, \ldots, N-1\}$. The subscript θ denotes the set of internal parameters of the estimated distribution. We already mentioned that it is sufficient to learn $p_\theta(\vec{c}\,|\,(\vec{0}, \mathbf{x}))$ due to the translational invariance. We assume in the following that all training samples are given in the form $(\vec{c}^{\,k}, (\vec{0}, \mathbf{x}^k))$.

The goal is to find the maximum likelihood estimate of the distribution $p_\theta(\vec{c}\,|\,(\vec{0}, \mathbf{x}))$ given the training samples. We have embedded the distribution in a vector space by setting $p_\theta(\vec{c}\,|\,(\vec{0}, \mathbf{x})) = \mathbf{e}_{\vec{c}}^{\top}\mathbf{f}_\theta(\mathbf{x})$. So, we have to formulate the maximum likelihood criterion for the function $\mathbf{f}_\theta(\mathbf{x})$ directly. It is well known that a hypothesis minimizing the KL-divergence with respect to the empirical distribution also maximizes the likelihood of the given sample. In [Abe 01] it is shown that learning with respect to the KL-divergence is related to learning with respect to the quadratic distance in the sense that one have to minimize

$$L(\theta) = \sum_{k=0}^{N-1} \|\mathbf{f}_\theta(\mathbf{x}^k) - \mathbf{e}_{\vec{c}^{\,k}}\|^2 \tag{6}$$

with respect to the parameters θ. But, the equivalence only holds if $\mathbf{f}$ is contrained to be a density, i.e. all entries are positive and sum up to 1. If we want to solve this problem by an ordinary kernelized regression we are not able to force $\mathbf{f}_\theta(\mathbf{x})$ to behave strictly like a density. But, for a kernel of local support, for example the exponential kernel, the solution behaves very much like the maximum likelihood solution. As usual in kernelized regression, the coefficients $f_\theta^{l,i}(\mathbf{X})$ are modelled by a linear combination of kernel evaluation

$$f_\theta^{l,i}(\mathbf{x}) = \sum_{k=0}^{N-1} K^l(\mathbf{x}, \mathbf{x}^k)\alpha_k^{l,i}$$

The set θ of complex weighting factors $\theta = \{\alpha_k^{l,i} \in \mathbb{C} \,|\, k = 0, \ldots, N-1,\ l = 0, \ldots, l_{\text{max}}\ , i = 0, \ldots, E-1\}$ have to be learned. We can let the Fourier index range only from 0 to l_{max} because the function that we want to learn is real valued, then components with the negative index are the complex conjugate of the positive ones. The minimization problem (6) translates directly to the problem of solving for each l and i the linear equations:

$$\sum_{k=0}^{N-1} K^l(\mathbf{x}^j, \mathbf{x}^k)\alpha_k^{l,i} = e_{\vec{c}^j}^{l,i}.$$

If we truncate the Fourier expansion after l_{max} coefficients and use E components for the radial part of the function, we have to solve in total $(l_{\text{max}} + 1) \cdot E$ linear equations of dimension N.

5.3 Training of the Shape Prior

To model the shape density $p(\vec{\mathbf{c}}|\vec{v}, \phi)$ we want to use a M-dimensional complex gaussian model. This means that we interpret the positions $\vec{c} \in \mathbb{R}^2$ as numbers in the complex plane $\mathbb{C}$. We will indicate this by omitting the vector arrows, i.e. we write c instead of $\vec{c}$ and $\mathbf{c}$ instead of $\vec{\mathbf{c}}$. This approach has two advantages. At first, it forces the one-dimensional marginals (real two-dimensional) of any Gaussian to be isotropic (see appendix). This is a reasonable assumption in the presence of the invariance constraints. The second reason is that the complex representation is very well suited for the optimization procedure applied later.

We already sketched how to include the invariance constraints by a normalization approach. Thus, given the covariance matrix $\mathbf{C} \in \mathbb{C}^{M \times M}$ and the mean $\mathbf{m} \in \mathbb{C}^M$ the distribution has the form

$$p(\mathbf{c}|v, \phi, \mathbf{m}, \mathbf{C}) \propto e^{-(\mathbf{c}-(e^{\mathbf{i}\phi}\mathbf{m}+v\mathbf{1}))^\top \mathbf{C}^{-1}(\mathbf{c}-(e^{\mathbf{i}\phi}\mathbf{m}+v\mathbf{1}))},$$

The Gaussian assumption can also be interpreted as a 'spring'-model as proposed by Fischler and Elschlager [Fischler 73] to model the shape variations. The mean $\mathbf{m}$ can be interpreted as the mean or standard shape of the object part constellation. The covariance matrix models the stiffness of the connection between the parts.

To learn the model we always assume position and orientation normalized objects, i.e. we learn $p(\mathbf{c}|0, 0)$. It is assumed that the user has labeled a sufficient set of training objects by manually selecting the M object parts. The learning process

itself is a usual estimation of the mean and covariance parameters. We further assume an isotropic Gaussian error model of the observed training samples, that is we regularize the covariance estimate by $\mathbf{C}_{\text{est}} + \sigma_{\text{reg}}^2 \mathbf{I}$, which is, of course, also useful in the absence of enough training samples. For σ_{reg} we choose values in the range from one to several pixel units.

6 Detection

As already explained the goal is to find an appropriate subset $\mathcal{I}$ of feature points and a configuration $\mathbf{c}$ such that the posterior probability

$$p(\vec{\mathbf{c}} \,|\, \mathcal{I}) \propto p(\vec{\mathbf{c}}) \prod_{j=0}^{M-1} p(\vec{c}_j \,|\, \mathcal{I})$$

is maximized. We divide the problem into two parts. First we make an initial hypothesis. We select one specific part of the object a priori. This part should be the most prominent and easily detectable of all parts. For this part, let us say $\vec{c}_0$ without restriction of generality, we compute the voting map $p(\vec{c}_0 \,|\, \mathcal{X})$ and search for all local maxima above a certain threshold. These maxima are designated as our initial hypotheses. For each maximum we gather all neighboring keypoints within a specific distance into the image portion $\mathcal{I}$. For this portion we search for an optimal configuration $\vec{\mathbf{c}}$ by an iterative optimization procedure.

6.1 Rendering of the Voting Maps and Finding Initial Hypotheses

To find the initial hypothesis we select the part $\vec{c}_0$ of the object a priori. We have to compute the voting map $p(\vec{c}_0|\mathcal{X})$ relying on all detected keypoints, according to equation (3).

As explained, the corresponding functions $\mathbf{f}_0(\mathbf{x})$ do not exactly behave like a distribution. Consider a feature $\mathbf{x}$ which is in terms of the feature distance $\|\mathbf{x}-\mathbf{x}^k\|$ very far away from the training samples $\mathbf{x}^k$. If we use an exponential GIM-kernel (a kernel with local support) the function $\mathbf{f}_0(\mathbf{x})$ tends towards zero. And thus, also the mean $\mathbf{1}^\top \mathbf{f}_0(\mathbf{x})$ tends towards zero. This is definitely not an appropriate behavior for a density, because for any density it always must hold $\mathbf{1}^\top \mathbf{f}_0(\mathbf{x}) = 1$. But, in the actual implementation we neglect this effect. This is justifiable because

for the sample very far from the training samples we do not have any knowledge, so it is reasonable that they do not contribute to the overall probability.

We directly use the contribution of the function $\mathbf{f}_0(\mathbf{x})$ from all observed local appearances $\mathbf{x}^k$ in the image under consideration to approximate the density:

$$p(\vec{c}_0|\mathcal{X}) \approx \sum_{(\vec{z}^k,\mathbf{x}^k)\in\mathcal{X}} \mathbf{f}_0(\mathbf{x}^k)^\top \mathbf{e}_{(\vec{c}_0-\vec{z}^k)},$$

In practice the voting map is rendered at a lower resolution than the original image. This is mainly due to complexity reasons. We use sizes from half up to fourth of the original resolution, depending on the size of searched object. In Figure 3 we show an example of a rendered voting map.

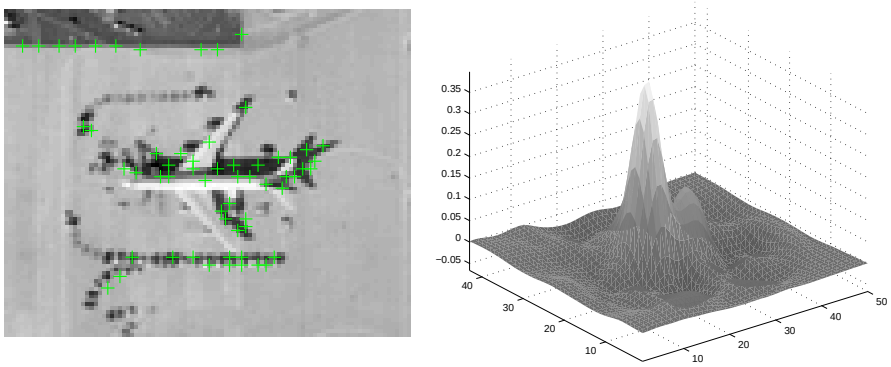


Figure 3: **Example for a voting map.** On the top you see the original image together with the detected keypoints marked by small crosses, on the bottom the rendered voting map. The voting map is rendered at half of the resolution of the original image. One can see that the voting map is also negative at some points, because of the non-density-like behavior of the voting functions.

Finally, we select the local maxima of the voting map as initial hypotheses. At this point one could be already finished. One could designate these local maxima above a certain threshold as successful detections and return them. But we want to go a step further and apply the already sketched verification stage. Therefore, we select for each initial hypotheses a subsets $\mathcal{I}$ of keypoints whose distance is lower than a specific threshold to the initial detection. Based on this subset we render the voting maps for the other parts of the object and process them as explained in the next section.

6.2 Hypothesis Verification / An Active Point Model

Having found the set $\mathcal{I}$ the voting maps for all other object parts are rendered based on this subset. To obtain higher accuracy the resolution of this voting maps are usually chosen higher than for the initial hypothesis. Then, we normalize the rendered voting maps such that they are probability densities. We set all negative entries to zero and normalize the sum over all entries to 1.

The goal of the verification process is to find a configuration $\mathbf{c}$, an orientation ϕ and a position v such that the log-likelihood is maximized

$$L(\mathbf{c}, \phi, v) = \log \left(p(\vec{\mathbf{c}}|\vec{v}, \phi) \prod_{j=0}^{M-1} p(\vec{c}_j \,|\, \mathcal{I}) \right)$$

We use an iterative approach for optimization. In the first step we assume that the current model $\mathbf{c}$ is fixed. Under the assumption that the mean shape $\mathbf{m}$ has a vanishing center of gravity $\mathbf{1}^\top \mathbf{m} = 0$, it is easy to compute the optimal v and ϕ analytically by Procrustes analysis (see e.g. [Mardia 98]),

$$\begin{aligned} v &= \tfrac{1}{M} \mathbf{1}^\top \mathbf{C}^{-1} \mathbf{c} \\ \phi &= \arg(\mathbf{c}^\top \mathbf{C}^{-1} \mathbf{m}) \end{aligned}$$

This is only possible due to our restriction to a M-dimensional complex Gaussian model, instead of the $2M$-dimensional real model. Afterwards, we make a gradient descent for $\mathbf{c} \rightarrow \mathbf{c} + \alpha \nabla L$, where α is a step width. These two procedures are repeated until a stop criterion is met. Such an optimization algorithm is typical for active-contour approaches that incorporate a prior information about the shape of the contour. In our case the shape prior is the mean $\mathbf{m}$ of the gaussian distribution and the covariance matrix $\mathbf{C}$ models the typical variation. In Algorithm 2 the optimization procedure is presented in pseudo code.

The question remains how to obtain good initializations of the parameters. As we already have an initial guess for the object part $\vec{c}_0$, it is obvious to use this for initialization. But still ϕ is arbitrary. To get a good initial estimate for ϕ we learn an additional distribution, namely $p(\phi|\mathbf{x})$. In this case the output vector is just a distribution over the angle ϕ in Fourier representation. It is assumed that the training samples are all given in normalized orientation, so we just have to learn a function which for all training samples returns a vector with its only entry at position $\phi = 0$.

As initial estimates we use the four highest local maxima of the estimate for $p(\phi|\mathcal{I})$. Finally, if we have obtained the optimal configuration $\vec{\mathbf{c}}^{\text{opt}}$ we compute

Algorithm 2 Optimization Procedure

1: Initialize $\mathbf{c}^{(0)}$
2: **repeat**
3: Let $v = \frac{1}{M}\mathbf{1}^\top \mathbf{C}^{-1}\mathbf{c}^{(i)}$ and $\phi = \arg(\mathbf{m}^\top \mathbf{C}^{-1}\mathbf{c}^{(i)})$
4: Let $\mathbf{k} = \left[\frac{\nabla p(\mathbf{c}_0^{(i)}|\mathcal{I})}{p(\mathbf{c}_0^{(i)}|\mathcal{I})}, \ldots, \frac{\nabla p(\mathbf{c}_{Q-1}^{(i)}|\mathcal{I})}{p(\mathbf{c}_{Q-1}^{(i)}|\mathcal{I})}\right]$
5: Update $\mathbf{c}^{(i+1)} = \mathbf{c}^{(i)} + \alpha(\mathbf{k} - \mathbf{C}^{-1}(e^{\mathbf{i}\phi}\mathbf{m} + v\mathbf{1} - \mathbf{c}^{(i)}))$
6: Increment $i = i + 1$.
7: **until** $\|\mathbf{c}^{(i+1)} - \mathbf{c}^{(i)}\| <$ threshold

the probability $p(\vec{\mathbf{c}}^{\text{opt}} \,|\, \mathcal{I})$ and select the highest of the results of the different initializations. We call this $p(\vec{\mathbf{c}}^{\text{opt}} \,|\, \mathcal{I})$ estimate the detection confidence for $\mathcal{I}$. As a final step, it is decided via thresholding of the detection confidence whether the object is present or not.

7 Experiments

For the actual implementation of the proposed approach we used *Matlab* and *C++*. The time-consuming tasks, including the computation of the features, the kernel matrix and the voting maps are implemented in *C++* using the *mex*-interface of *Matlab*. The development and experiments took place on a *P4 2.8Ghz*, so all timings reported below are achieved on this machine.

Experiments on two different databases are presented. The the detection of planes on aerial photographs and the detection of pollen grains in microscopical images. There are several parameters that can be optimized to the considered data. The parameters for the feature computation are: the cutoff frequency $l_{\max}$ for the Fourier transform, the number of bins R in angular direction a, the number of bins D in radius direction d and the maximal radius $d_{\max}$ in pixel units. For the voting scheme, there is the number E of triangular shaped envelope functions that are used to synthesize the voting function and the radial range $w_{\max}$ over which they are distributed. The parameters for the different databases are shown in Table 1. They were chosen manually by trial and error on small training sets.

Another important parameter is the number of keypoints. For the training phase we adjusted the threshold such that the number of keypoints stay below approximately 500 in order to keep the detection times in a reasonable range. During the detection process the threshold is usually reduced so that also keypoints are

found in regions with very low contrast.

7.1 Comparison to Reference Methods

We proposed a new type of local feature descriptor together with a new parametric voting scheme. To examine the performance of our features we have chosen to consider additionally the so called GLOH features that are currently known to be one of the best local descriptors. On the other hand, to compare the proposed voting scheme, we will consider also a non-parametric voting that is usually applied for GHT-based object detectors.

7.1.1 GLOH Features

The GLOH (Gradient Location and Orientation Histogram) features [Mikolajczyk 05] are a further development of the SIFT features [Lowe 04]. They were built to increase its robustness and distinctiveness. Compared to SIFT, the histogram is computed for 17 location and 16 orientation bins in a log-polar location grid. PCA is used to reduce the dimension to 128. The GLOH features will be used to compare our features introduced above with the state-of-the-art.

To obtain rotation invariance the GLOH features are steered at the main gradient direction that is estimated from the local neighborhood. If the main gradient direction is not unique the feature is computed for several directions.

7.1.2 Non-Parametric Voting

Traditionally, the GHT-based [Ballard 81] object detection systems like the ISM [Leibe 04] or others [Mikolajczyk 06, Teynor 07] rely on a non-parametric voting scheme. That is, the conditional density $p(\vec{c} \mid \mathbf{X}) = p(\vec{c} \mid (\vec{z}, \mathbf{x}))$ is estimated in a non-parametric manner (see e.g. [Duda 73] for non-parametric density estimation in the context of pattern recognition). To make the computation feasible most

Database	Feature				Voting	
	$l_{\max}$	R	D	$d_{\max}$	E	$w_{\max}$
Pollen	8	8	6	15	7	40
Planes	6	6	4	12	7	40

Table 1: Parameters for the different databases

approaches compute beforehand a so called codebook of local appearance. The entries of this codebook can be imagined as those patterns $\mathbf{x}$ that contain the main support of the density. Those patterns are usually chosen to be cluster centers that are obtained e.g. by agglomerative or k-means clustering. How to obtain optimal codebooks is still an open problem. Depending on the type of clustering, it involves the tuning of a lot of parameters and several design choices have to be made. We want to circumvent this to avoid any pitfalls by using a traditional kernel density estimator with a Gaussian kernel. Though this is quite time consuming but it is probably the most simple and most canonical way.

Assume that the set of training samples $\mathcal{T} = \{(\vec{z}^k, \mathbf{X}^k) \mid k = 0, \ldots, N-1\}$ are given in a translation normalized form, that is $(\vec{z}^k, \mathbf{X}^k) = (\vec{z}^k, (\vec{0}, \mathbf{x}^k))$. Then the density estimate looks as follows

$$p(\vec{c} \mid (\vec{0}, \mathbf{x})) = \frac{1}{Z(\mathbf{x})} \sum_{k=0}^{N-1} \exp\left(-\frac{\|\vec{c} - \vec{z}_k\|^2}{\sigma^2_{\text{spat}}} - \frac{\|\mathbf{x} - \mathbf{x}^k\|^2}{\sigma^2_{\text{feat}}} \right), \tag{7}$$

where σ^2_{spat} and σ^2_{feat} are fixed width parameters. The term $Z(\mathbf{x})$ is a normalization factor that ensures that $p(\vec{c} \mid \mathbf{X})$ is a conditional probability density. It is proportional to

$$Z(\mathbf{x}) \propto \sum_{k=0}^{N-1} \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}^k\|^2}{\sigma^2_{\text{feat}}} \right)$$

The actual implementation is quite simple. Assume we have given the set of local appearance patches $\mathcal{X}$. To render the voting map we have to do the following. For each $\mathbf{X} = (\vec{z}, \mathbf{x}) \in \mathcal{X}$ we cast N votes at the positions $\vec{z} + \vec{z}^k$ for $k = 0, \ldots, N-1$. Each vote is weighted by $\frac{1}{Z(\mathbf{x})} \exp(-\frac{\|\mathbf{x}-\mathbf{x}^k\|^2}{\sigma^2_{\text{feat}}})$. The spatial form of the vote is a Gaussian. To avoid to render for each vote a Gaussian into the voting map, we just accumulate a single pixel by the appropriate weight and convole the voting map later with a Gaussian when all votes have been casted.

The above presented non-parametric voting scheme is only translation equivariant. The rotation equivariance is usually incorporated by a normalization approach (see e.g. [Mikolajczyk 06] or [Teynor 07]). We do not want to show this here in detail. The approach is only modified slightly. The voting directions z^k are measured relative to the gradient main directions around the training points and votes are then casted relative to the gradient main direction of the observed keypoints. If the gradient main direction is not unique the samples are duplicated for all local direction maxima that are above 85% of the global one.

7.1.3 Evaluation

For evaluation and comparison we consider Precision/Recall graphs and the equal error rate (EER). The equal error rate is the error rate obtained when the number of objects that are not detected is the same as the number of wrong detections. We consider an object to be detected if a detection is less than half the object size away from the object's center. If there are multiple detections in this range with different detection confidences we choose the detection with the maximal confidence.

7.2 Aircraft Detection

To demonstrate the effectiveness of our approach, we let the system to detect planes on aerial photographs of airports. We obtained 20 aerial images from *Google Earth*[2] of the airports of Frankfurt, Munich, London and New York. The altitude of the images is approx. 700ft. Each image is scaled to the of size of 1200×1000. A plane has then an average size of about 50 pixels. All images together contain 208 airplanes. The images show heavy clutter and there are many possible candidates for false positive detections. Additionally, the lighting conditions change such that the system has to cope with many kinds of different shadows. And finally, there are planes of different sizes, the system is confronted with scale changes up to a factor of two. To cope with such scale changes we applied the system at four different scales, where the scale is sampled logarithmically by a subsampling factor of 1.2, that is, the smallest image is by a factor of 1.2^4 smaller than the original. To get the initial hypotheses the four voting maps from the different scales are stacked together in a 3D voting stack and the local maxima in this 3D stack serve as detection hypotheses.

Figure 4 shows the images that were used for training. They are chosen such that the typical variations for the different kinds of shadows can be learned. We selected three object parts: the region where the wings touch the fuselage, that is the 'center' of the object and the two tips of the wings. The center is used to gather the initial hypotheses.

In Figure 6 a detection example is shown to give an impression of the complexity of the images and of the performance of the system. The computation time heavily depends on the complexity of the scene. For the given example, that is one of medium complexity, our algorithm needs about 10 seconds. Most of the time is spend for the verification stage. In Figure 5 some typical examples of

[2]Google Earth, A 3D Interface to the Planet, *http://earth.google.com*

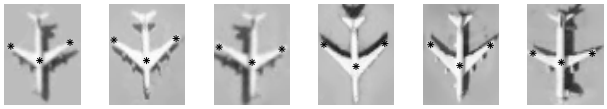


Figure 4: **Training Images.** The six images that were used for training. We let the plane consist of three parts: the center and the two tips of the wings. They are marked by black stars.

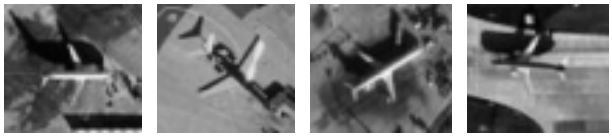


Figure 5: **Planes that were not detected.**

missing detections are shown. Their small size and the heavy shadows seem to be responsible for the misses.

In Figure 7(a) we show a PR-graph that shows the performance of our detection system with and without verification by the active point model. The voting function is based on a GIM-kernel with an exponential basis kernel. It is obvious that the verification method improves the results dramatically. With verification we obtain a EER of 30.3%. Without verification we are only able to get a EER of 41.8%. The overall number of detected aircrafts (80%) is not derogated.

For further comparison we want to leave out the verification stage and directly work with initial hypotheses that are obtained by the voting map for the center of the aircraft. This allows us to figure out more clearly the underlying circumstances. And secondly, in literature there is no comparable validation approach that works in a rotation invariant manner (to the author's knowledge). Although the ISM [Leibe 04] uses segmentation masks to obtain even better confidence scores, it does not work in a rotation invariant manner and a generalization is not straight forward.

In the next experiment we compare the GIM-kernels for different kind of non-linearities as given in equation (2). Figure 7(b) shows the results. It is astonishing that already a linear kernel works quite well. Note, that a kernelized regression with a linear kernel is equivalent to a traditional linear regression scheme, i.e. the mapping from the local features onto the voting function is a linear one. The quadratic kernel improves the performance but cannot compete with the exponential kernel. Actually, the approximation of the exponential kernel is as good as the

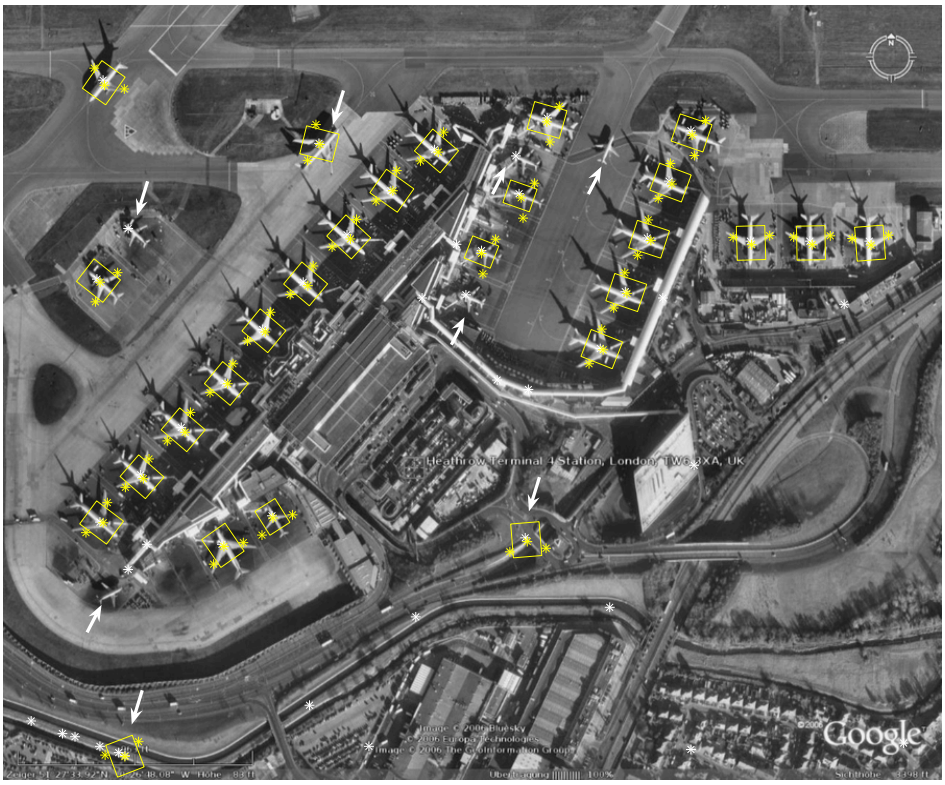
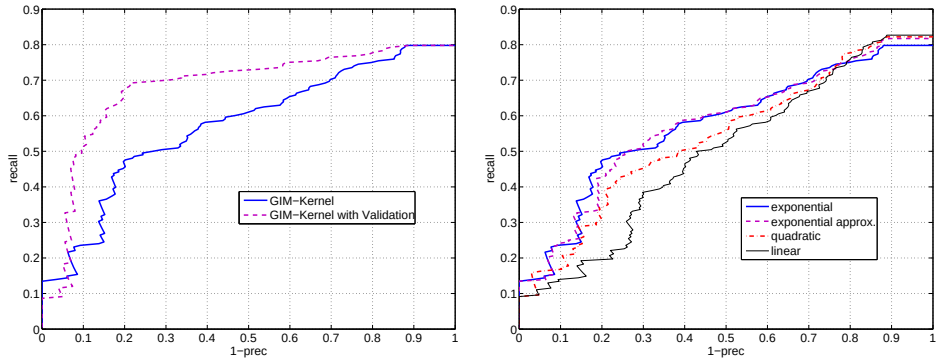


Figure 6: **Recognition examples for the plane database**. The yellow rectangles and dots indicate the detected configuration. The white dots give the position of the initial hypotheses. The white arrows show two false positive, five not detected false negatives and one example for a wrong determination of the orientation of the plane.



(a) PR-Graph with and without validation (b) PR-Graph of for different GIM-kernels

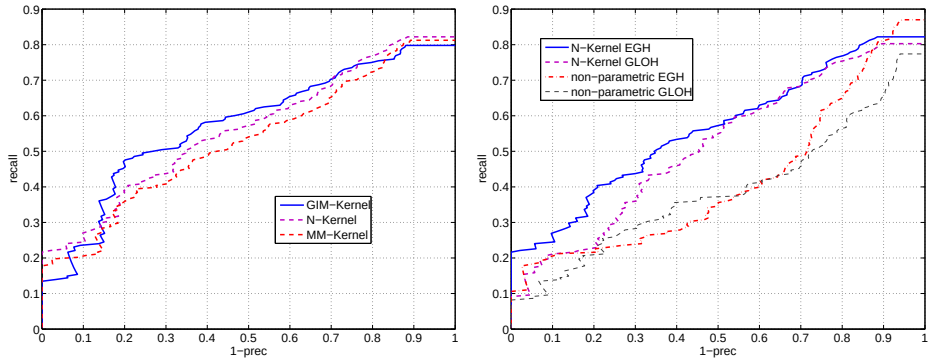
Figure 7: Evaluation of our method

exponential itself. That is quite astonishing. The underlying feature space of this approximative kernel is just the direct sum of feature space of the linear and the quadratic kernel. Thus, it seems that the feature spaces alone are worse than the combination of both with a proper weighting. The value for λ was chosen by a manual tuning on a small training set.

In another experiment we compared the different kind of equivariant kernels. We always used an exponential scalar basis kernel. In Figure 8(a) we show the Precision/Recall graph for the GIM-kernel, the Normalization-kernel and the Maximum-Matching-kernel. The GIM-kernel performs best, closely followed by the N-kernel and the MM-kernel. It is astonishing that the MM-kernel is a little bit worse than the N-kernel. One might expect that a matching is more reliable than a normalization of the patches. But overall, the differences are not very distinct.

Finally, we make a comparison of the proposed parametric voting with the non-parametric voting scheme and secondly, a comparison of the GLOH features with our EGH features. We used a N-kernel for the parametric voting scheme, because it is the only kernel that works efficiently with the GLOH features. The results are concluded in Figure 8(b). We can observe a clear difference between the non-parametric approach in comparison to the parametric approach. The parametric approach works definitely better than the non-parametric approach. In Figure 9 we show two exemplary voting maps for the non-parametric and parametric approach. The non-parametric approach has obvious problems in regions with lots of spurious keypoints. The reasons for that are difficult to determine, we figured out three main issues.

Firstly, the parametric representation of the voting function allows to handle



(a) PR-graph for different equivariant Kernels.

(b) PR-graph for non-parametric versus parametric approach

Figure 8: Comparison to reference methods

radial and angular deformation independently, because it is expanded in polar coordinates. This means that the smoothing parameters can be tuned independently in radial and angular direction. We found that this relationship is an important factor for the performance of the parametric approach. For the non-parametric case it is only possible to incorporate the smoothing efficiently in an isotropic manner (controlled by σ_{spat} in equation (7)).

For the non-parametric approach the parameter σ_{feat} controlls the trade-off between specificity for the searched class of objects and robustness against intra-class variations. With growing σ_{feat} we become robust against variations of the object but also induce lots of false positive detections. We found that this trade-off has a major impact on the performance of the non-parametric approach. It seems that the parametric approach can handle this trade-off much better. There are two parameters that play a similar role for the parametric approach. In the case of a exponential basis kernel, the λ parameter, and secondly during the training stage one can apply a kernelized ridge regression (also known as regularized regression), where the regularization parameter plays a similar role. We found that the choice for both are very easy and robust. For our tasks we never found a regularized regression to be superior to a non-regularized. For the λ parameter we found that once it was adapted to the type of feature it was very robust and application independent.

We already mentioned that our parametric voting algorithm can also produce negative contributions, that is, it cannot be interpreted as a real probability density anymore. In fact, this 'fault' of our approach helps to avoid false positive

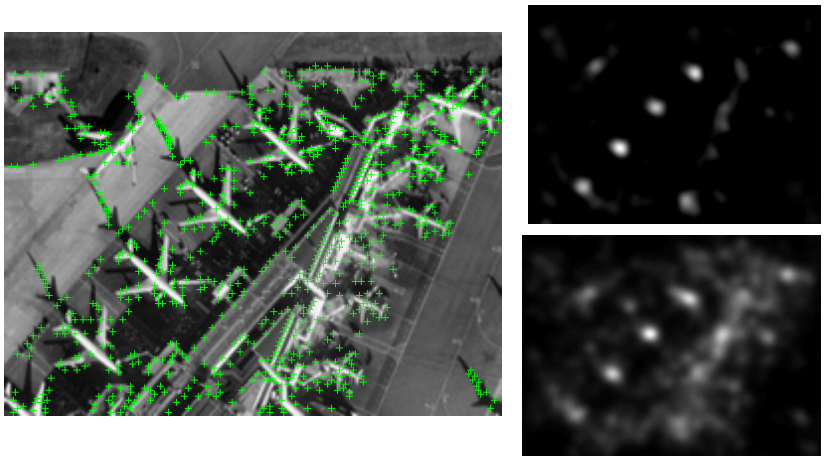


Figure 9: **Voting Map Comparison.** On the left you see the original image together with the detected keypoint, on the upper right the voting map produced by the parametric approach, on the lower right the voting map produced by non-parametric approach.

detections in regions that contain lots of spurious keypoints. Due to the negative voting contributions it can happen that votings destructively interfere. Only if the contributions show a kind of coherence the votes show constructive interference and thus strong responses.

Finally, consider in Figure 8(a) the performance differences for the GLOH and EGH features. It seems that our features perform slightly better but not significantly. And even the comparison is not really fair because the parameters of our features were specifically tuned for the considered dataset, while the parameters of the GLOH features were fixed from the beginning (except the overall radius, it was chosen the same for both features).

7.3 Pollen Detection

Analysis techniques for data acquired by microscopy typically demand for a rotation and translation invariant treatment. The microscopical images of particles like cells or pollen have usually no predetermined orientation; the positions of the particles are unknown and even the number of particles is not known a-priori. We want to demonstrate the effectiveness of our system with a pollen detection task. Applications of such a system are manifold. Palynology, the study and analysis of pollen, is an interesting topic with very diverse applications like in forensics.

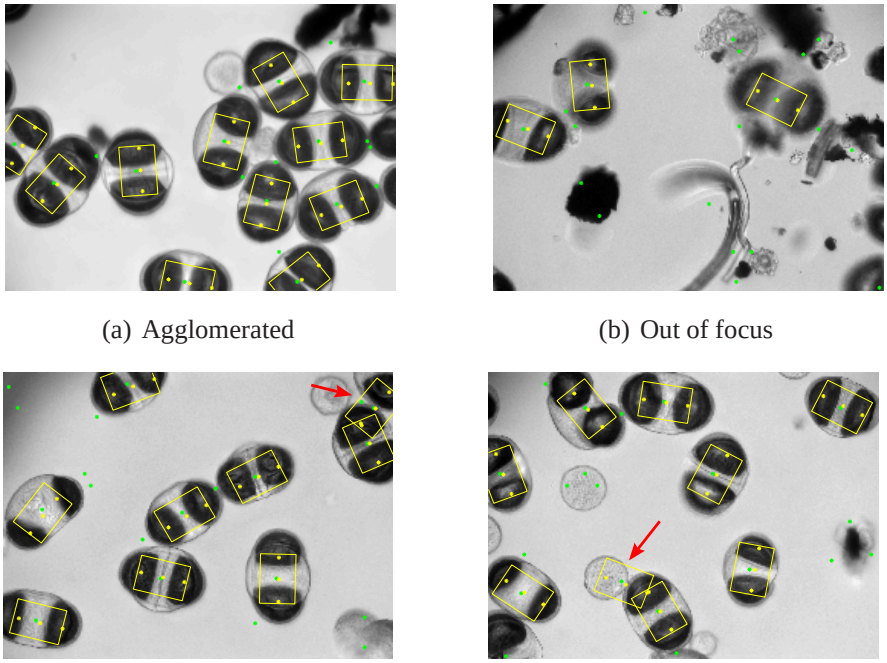
Pollen-forcasts for allergological relevant pollen species is also an important issue. Most of the pollen species are very easily detectable by a simple Hough Transform because they have a round, circle-like structure. But a high amount of fossil pollen and also pollen of today's conifers have not such an easy detectable structure.

The data used in this experiment was recorded with an optical microscope in conjunction with the OMNIBUSS project [Scharring 06]. Originally the samples are of size $1392 \times 1040 \times 70$. For recognition the sharpest of the 70 image layers is selected and this layer is downscaled to a size of 256×191, which is more than enough for the detection task. The pollen's size is between a fifth up to a tenth of the image size. To get an impression, have a look at the four images at Figure 11. The used training examples are shown in Figure 10. We selected three parts to represent the pollen. The center and the two characteristic black dots. The center is used to obtain the initial votes. 549 images were selected for testing. Nearly half of the images contain dust and dirt and no pollen at all. The others from one up to approximately 10 pollen. In total the images contain 751 pollen. We labeled them manually by determining just the position of the pollen. The orientation of the pollen is not verified in the experiments.

In Figure 12(a) we compare four different types of approaches. Our parametric kernel-based methods based on the GIM-kernel, N-kernel and MM-kernel and the non-parametric approach. For all four methods we only report the results for our EGH features. The results for the GLOH features show qualitatively the same behavior. When comparing the kernel-based approaches one can observe that the GIM-kernel again performs best, while this time the MM-kernel is better than the N-kernel. It seems that the normalization is not so reliable for this task. In contrast to the experiences from the aircraft detection, this time, the non-parametric approach works much better and is competitive with the parametric approach. It works as good as the MM-kernel.

Why does the non-parametric approach perform so badly for the aircraft detection and on the other hand show comparable results for the pollen task? It is difficult to answer this question. One major difference between the two databases

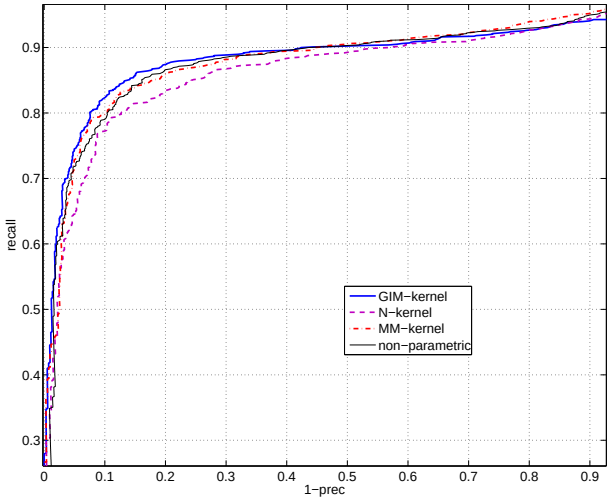
Figure 10: **Pollen training images** The white dots indicate the location of the object parts.



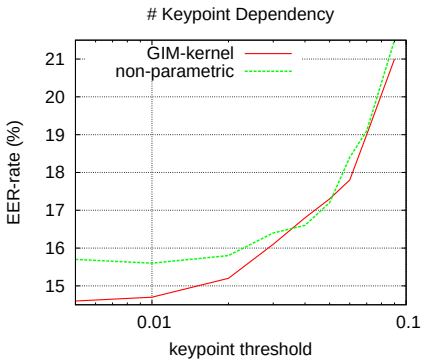
(a) Agglomerated (b) Out of focus

(c) False Positive (d) False Positive

Figure 11: **Recognition examples for the pollen database.** The yellow rectangles and dots indicate the detected configuration. The green dots give the position of the 20 initial hypotheses. Images 11(a) and 11(b) show two difficult examples that are perfectly solved. Images 11(c) and 11(d) show two typical false positive detection. If two pollen are very near, their parts may be confused.



(a) Comparison for the Pollen database.



(b) Influence of the saliency threshold.

Figure 12: Evaluation on the Pollen database

is that the airport images show much more possible candidates for false positive detections. For example, compare Figure 9 with one of the images in Figure 11. The airport images contain much more spurious keypoints that do not belong to the objects of interest. The aircrafts often do not show the strongest local variations in comparison to the clutter. Contrarily, the conifer pollen are often the most prominent objects in the images, they show the strongest local variations and the most salient keypoints are located on the pollen itself. Thus, we can suspect that the parametric approach can cope much better with a high number of spurious keypoints and, hence, is more robust to false positives. But how to verify this conjecture? If we assume that the most salient keypoint are detected on the pollen itself it should be possible by choosing a high saliency threshold that most of the detected keypoints are on the pollen. Then, the results for the non-parametric approach should get comparable to the parametric approach, because there are only a very low number of spurious keypoints. On the other hand if the saliency threshold is very low and the number of spurious keypoints is large, the advantage of the parametric approach should become obvious. So, we made experiments for various saliency thresholds. We recorded the EER-rate for the parametric approach with a GIM-kernel and the non-parametric approach, the results are reported in Figure 12(b). In fact, our conjecture is confirmed. For a high number of keypoints the parametric approach shows one percent less error than the non-parametric one. On the other hand, for a low number of keypoints, the differences get negligible.

8 Conclusion

In this article we presented a rotation invariant object detection concept. The system is well motivated in a probabilistic framework. The invariance demands are gently introduced into the framework by the idea of using matrix-valued kernels. The experiments have shown that the system works for very different problems and can achieve competitive results.

Although the running times for the presented tasks are acceptable, they restrict the system to small training set sizes. The training set size has a direct influence on the number of support vectors and hence on the run-time. Thus, the system is yet not able to model too large variations within one object class while working in reasonable time. Here is space for further improvements.

The number of keypoints has even the same impact on the run-time as the support vectors. We have shown that the number of keypoints has a high influence on the performance of the system. Probably the best would be to designate each

pixel in the image to be a keypoint, which is unfortunately much too expensive. A possible way out would be to construct object specific key point detectors.

9 A Complex Gaussian Density

Let $\mathbf{c} \in V$ be a M-dimensional vector in a complex valued vector space. Further let $\mathbf{m} \in V$ a mean and $\mathbf{C} \in L(V)$ a positive definite covariance matrix, then

$$p(\mathbf{c}) = \frac{1}{(2\pi)^M |\det(\mathbf{C})|} \exp\left(-\frac{1}{2}(\mathbf{c}-\mathbf{m})^\top \mathbf{C}^{-1}(\mathbf{c}-\mathbf{m})\right)$$

defines a probability density on V. Any one-dimensional marginal $p(\mathbf{c}^\top \mathbf{w} = w)$ in direction $\mathbf{w} \in V$ is of the form

$$p(w) = \frac{1}{2\pi k} \exp\left(-\frac{1}{2k}\,|w-m|^2\right),$$

where $k \in \mathbb{R}$ is positive and real and $m \in \mathbb{C}$. This easy to see, because any marginal is again a complex Gaussian density and a one-dimensional complex Gaussian has by definition a positive definite $k \in \mathbb{R}$ variance. Hence, any one-dimensional complex Gaussian has a spherical invariant (isotropic) variance in the complex plane.

References

[Abe 01] ABE, N., TAKEUCHI, J., AND WARMUTH, M. "Polynomial Learnability of Stoachastic Rules with Respect to the KL-Divergence and Quadratic Distance". *IEICE Trans. Inf. and Syst.*, Vol. 3, pp. 299–316, 2001.

[Aguado 02] AGUADO, A., MONTIEL, E., AND NIXON, M. "Invariant Characterization of the Hough Transform for Pose Estimation of Arbitrary Shapes". *Pattern Recognition*, Vol. 35, pp. 1083–1097, 2002.

[Ballard 81] BALLARD, D. "Generalizing the Hough transform to Detect Arbitrary Shapes". *Pattern Recognition*, Vol. 13, No. 2, pp. 111–122, 1981.

[Barczak 05] BARCZAK, A. "Towards an Efficient Implementation of a Rotation Invariant Detector Using Haar-like Features". In: *Proceedings of the IVCNZ'05, Dunedin, New Zealand*, pp. 31–36, 2005.

[Crandall 06] CRANDALL, D. AND HUTTENLOCHER, D. "Weakly Supervised Learning of Part-Based Spatial Models for Visual Object Recognition". In: Leonardis, A., Bischof, H., and Pinz, A., Eds., *Proceedings of ECCV*, LNCS, Springer, 2006.

[Duda 73] DUDA, R. AND HART, P. *Pattern Classification and Scene Analysis*. John Wiley & Sons, 1973.

[Felzenszwalb 05] FELZENSZWALB, P. AND HUTTENLOCHER, D. "Pictorial Structues for Object Recognition". *Intl. Journal of Computer Vision*, Vol. 61, No. 1, pp. 55–79, 2005.

[Fergus 03] FERGUS, R., PERONA, P., AND ZISSERMAN, A. "Object class recognition by unsupervised scale-invariant learning". In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 264–271, IEEE Computer Society, 2003.

[Fergus 05] FERGUS, R., PERONA, P., AND ZISSERMAN, A. "A Sparse Object Category Model for Efficient Learning and Exhaustive Recognition". In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society, 2005.

[Fischler 73] FISCHLER, M. AND ELSCHLAGER, R. "The Representation and Matching of Pictorial Structures". *IEEE Transactions on Computers*, Vol. C-22, No.1, pp. 67–91, 1973.

[Jurie 04] JURIE, F. AND SCHMID, C. "Scale-invariant shape features for recognition of object categories". In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 90–96, IEEE Computer Society, 2004.

[Leibe 04] LEIBE, B., LEONARDIS, A., AND SCHIELE, B. "Combined Object Categorization and Segmentation with an Im-

plicit Shape Model". In: Pajdla, T. and Matas, J., Eds., *Proceedings of the ECCV'04 Workshop on Statistical Learning in Computer Vision, Prague*, LNCS, Springer, 2004.

[Leung 98] LEUNG, T., BURL, M., AND PERONA, P. "Probabilistic affine invariants for recognition". 1998.

[Lindeberg 98] LINDEBERG, T. "Feature detection with automatic scale selection". *International Journal of Computer Vision*, Vol. 30, No. 2, pp. 77–116, 1998.

[Lowe 04] LOWE, D. "Distinct Image Features from Scale-Invariant Keypoints". *International Journal of Computer Vision*, Vol. 60, pp. 91–110, 2004.

[Mardia 98] MARDIA, K. AND I.L.DRYDEN. *Statistical Shape Analysis*. Wiley, Chichester, 1998.

[Micchelli 05] MICCHELLI, C. AND PONTIL, M. "On Learning Vector-Valued Functions". *Neural Computation*, Vol. 17, pp. 177–204, 2005.

[Mikolajczyk 05] MIKOLAJCZYK, K. AND SCHMID, C. "A Performance Evaluation of Local Descriptors". *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 27, No. 10, pp. 1615–1630, 2005.

[Mikolajczyk 06] MIKOLAJCZYK, K., LEIBE, B., AND SCHIELE, B. "Multiple Object Class Detection with a Generative Model". In: *Proceedings of the CVPR*, pp. 26 – 36, IEEE Computer Society, 2006.

[Nicholson 01] NICHOLSON, W. AND GLAESER, R. "Review: Automatic Particle Detection in Electron Microscopy". *Journal of Structural Biology*, Vol. 133, pp. 90–101, 2001.

[Nicholson 04] NICHOLSON, W. "Object Detection by Correlation Coeffcients Using Azimuthally Averaged Reference Projections". *IEEE Trans. Biomed. Eng.*, Vol. 51, No. 11, 2004.

[Perona 95] PERONA, P. "Deformable Kernels for Early Vision". *IEEE Trans. Pattern Anal. Machine Intell.*, Vol. 17, No. 5, pp. 488 – 499, 1995.

[Reisert 07] REISERT, M. AND BURKHARDT, H. "Learning Equivariant Functions with Matrix Valued Kernels". *J. Mach. Learn. Res.*, Vol. 8, pp. 385–408, 2007.

[Scharring 06] SCHARRING, S., BRANDENBURG, A., BREITFUSS, G., BURKHARDT, H., DUNKHORST, W., V. EHR, M., FRATZ, M., GIEL, D., HEIMANN, U., KOCH, W., LÄDDING, H., MÜLLER, W., RONNEBERGER, O., SCHULTZ, E., SULZ, G., AND WANG, Q. *Biophotonics: Visions for Better Health Care*, Chap. Online Monitoring of Airborne Allergenic Particles (OMNIBUSS), pp. 31–88. Wiley-VCH, june 2006.

[Schmid 96] SCHMID, C. AND MOHR, R. "Combining greyvalue invariants with local constraints for object recognition". *cvpr*, Vol. 00, p. 872, 1996.

[Scholkopf 02] SCHÖLKOPF, B. AND SMOLA, A. J. *Learning with Kernels*. The MIT Press, 2002.

[Teynor 07] TEYNOR, A. AND BURKHARDT, H. "Localization of Visual Object Class Instances". In: *Proceedings of the IAPR Workshop on Machine Vision Applications (MVA2007)*, 2007.

[Torralba 04] TORRALBA, A., MURPHY, K., AND FREEMAN, W. "Sharing Features: Effcient Boosting Procedures for Multiclass Object Recognition". In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 762–769, IEEE Computer Society, 2004.

[Viola 01] VIOLA, P. AND JONES, M. "Rapid Object Detection using a Boosted Cascade of Simple Features". In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE Computer Society, 2001.

[Yokono 04] YOKONO, J. AND POGGIO, T. "Rotation Invariant Object Recognition from One Training Example". Tech. Rep. 238/AI Memo 2004-010, Massachusetts Institute of Technology, Cambridge, MA, 2004.

[Zhu 03] ZHU, Y., CARRAGHER, B., MOUCHE, F., AND POTTER, C. "Automatic Particle Detection through Effcient Hough Transform". *IEEE Trans. Med. Imag.*, Vol. 22, No. 7, pp. 1053–1062, 2003.