PRISM: PRincipled Implicit Shape Model

Alain Lehmann[1]
lehmann@vision.ee.ethz.ch
Bastian Leibe[2]
leibe@umic.rwth-aachen.de
Luc van Gool[1]
vangool@vision.ee.ethz.ch

[1] Computer Vision Laboratory
ETH Zurich
Switzerland
[2] Mobile Multimedia Processing
UMIC Research Centre
RWTH Aachen University

Abstract

This paper addresses the problem of object detection by means of the Generalised Hough transform paradigm. The Implicit Shape Model (ISM) is a well-known approach based on this idea. It made this paradigm popular and has been adopted many times. Although the algorithm exhibits robust detection performance, its description, *i.e.* its probabilistic model, involves arguments which are unsatisfactory from a probabilistic standpoint. We propose a framework which overcomes these problems and gives a sound justification to the voting procedure. Furthermore, our framework allows for a formal understanding of the heuristic of soft-matching commonly used in visual vocabulary systems. We show that it is sufficient to use soft-matching during learning only and to perform fast nearest neighbour matching at recognition time (where speed is of prime importance). Our implementation is based on Gaussian Mixture Models (instead of kernel density estimators as with ISM) which lead to a fast gradient-based object detector.

1 Introduction

Most current state-of-the-art object detectors are based on either the sliding-window [6, 7, 8, 19, 25, 27] or the Hough transform [1, 14, 17, 18, 23] paradigm. The former considers all possible sub-windows of an image and a classifier decides whether they contain an object of interest or not. For reasons of efficiency, mostly linear classifiers are used (a fast non-linear approach has been presented recently [19]) and advanced search schemes have been designed to overcome exhaustive search [12].

The second aforementioned paradigm is inspired by the Generalised Hough Transform (GHT) [1]. GHT was introduced for finding shapes other than lines. Recent work has shown how the underlying idea can be extended to object category detection from local features [14]. It is this extended form which we refer to in this paper. Processing starts with local feature extraction and each feature subsequently casts probabilistic votes for possible object positions, where the hypothesis score is obtained as sum over all votes. Such bottom-up voting schemes seem to be more natural than an exhaustive sliding-window search. Indeed, object class detectors based on this idea, first introduced by ISM [14], have become increasingly popular and similar ideas have been adopted in various approaches [4, 10, 17, 18, 23]. However, we argue that the commonly used ISM formalism is unsatisfactory from a probabilistic point of view. In particular, the summation of feature likelihoods is explained by *marginalisation over features*. This argument implies that all features are possible realisations of a

© 2009. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.
BMVC 2009 doi:10.5244/C.23.64

single random variable and, thus, *only one* feature could really be observed. However, we do observe *all* features concurrently. Thus, the argument is not convincing. Nevertheless, ISM empirically demonstrates robust detection performance. Furthermore, the summation is crucial for the voting paradigm. Hence, the question is: "How else can this summation be justified?". In this paper, we propose a new framework which connects the sliding-window and the Hough paradigms, thereby giving a sound answer to this question.

The central aspect of our framework is to consider the sliding-window and the Hough paradigms as two sides of the same coin. This duality is then exploited to define the object score from a sliding-window point of view, while the actual evaluation follows the Hough paradigm. The core idea is to compute a (visual) footprint of an object that is compared to an object model. This footprint represents all features in a canonical reference frame. Contrary to classical histogram-based approaches [6, 12], we keep a feature-centric view inspired by the Hough-transform. As such, it is structurally similar to the Implicit Shape Model (ISM) [14], but overcomes various of its deficiencies (due to the sliding-window based reasoning) which we will discuss.

In Sec. 3, we derive a concrete object detection system from our framework. Our training procedure is closely related to the ideas of ISM, *i.e.*, the estimation of spatial occurrence distributions of visual words. We show how pure ISM can be modelled within our framework and we thus make direct comparison possible. However, we estimate these spatial occurrence distributions with Gaussian Mixture Models (GMMs), thereby avoiding the unfavourable scaling behaviour of the non-parametric kernel density estimators used in ISM. Hence, object localisation can be performed by means of efficient gradient-based search (similar to mean-shift in ISM). Such a local refinement strategy may be of particular interest if one already has prior knowledge about objects (*e.g.* from object tracking). We demonstrate that the semi-parametric model makes it possible to speed up the search with no, or only little loss of accuracy.

Finally, Sec. 4 deals with the common practise of *soft-matching* for feature quantisation. We establish a formal understanding of this heuristic within our framework. As a result, we show that soft-matching can be moved entirely to the training stage, allowing for fast nearest neighbour matching during recognition.

2 The Framework

The task of object detection can be formulated as a search problem. Given a newly observed image I and a trained object model W, the goal is to find the best hypothesis $\lambda^* = \arg\max_{\lambda \in \Lambda} S(\lambda | I, W)$, where S is a score function and Λ is the search space of all potential object hypotheses. As the search space is rather large, finding the best-scoring hypothesis quickly is of great importance. The structure of S plays an important role when it comes to defining efficient search algorithms. We believe that *feature-centric* scores (as introduced in ISM [14]) offer a powerful approach. In such a framework, local features are matched to a visual vocabulary and cast votes for possible object positions (the score being the sum of votes). However, ISM's probabilistic model has various shortcomings and also does not allow for discriminative learning. We will now present a framework which overcomes these problems. We follow a sliding-window based reasoning and derive a feature-centric score from that. Fig. 1 supports the following derivations in graphical form. ISM's shortcomings as well as the connection between the sliding-window and the Hough paradigms will be discussed thereafter.

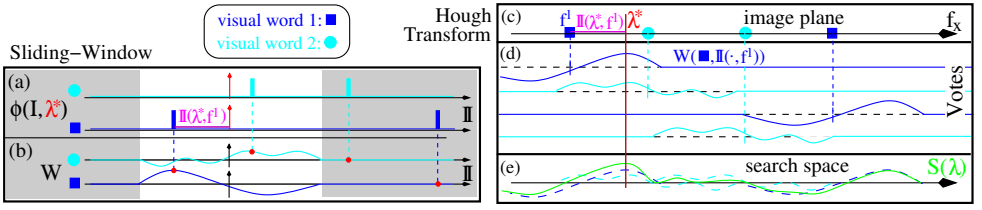

Figure 1: *PRISM: From the Sliding-Window (left) to the Hough Transform paradigm (right). 1D example with two visual words (blue rectangle, cyan circle). A footprint (a) gets extracted for a fixed hypothesis λ^* (red). The inner product with the weight function (b) results in a sum of point evaluations (red dots). Various features do* not *affect the score (shaded). Each extracted feature (c) casts a voting pattern (d) which, summed up, result in the final hypothesis score (e).*

2.1 Score Function

The central element of our framework is a footprint $\phi(\lambda, I)$ for a given object hypothesis λ extracted from the image I. This footprint represents all detected features in a canonical reference frame and, thus, describes an object independent of its location and scale in the image. Essentially, it crops out a sub-image which is the key idea of the sliding-window paradigm. Using a linear model, which is compulsory for the Hough transform, the hypothesis score is computed by the inner product

$$S(\lambda|I,W) = \langle \phi(\lambda, I), W \rangle \tag{1}$$

of the footprint and a weight function W, *i.e.*, the object model. Classical sliding-window detectors [6, 7, 12] represent all features in an object-centric coordinate frame. Often, the relative position is discretised which leads to a histogram representation [6] for ϕ and W. Contrary to previous work, we focus on the definition of the mapping function ϕ and avoid the discretisation. This allows us to switch (from the sliding-window standpoint) to a feature-driven Hough-transform view, which is central to our framework.

Image/Object representation. Before introducing the footprint we should specify the image representation and hypothesis parametrisation. For the image representation we extract a set of local features $\mathscr{F}$ [20]. Each feature is characterised by a descriptor, position, and scale, *i.e.*, $f = (f_c, f_x, f_y, f_s)$. Additionally, a feature weight $f_\omega > 0$ may be defined. In the remainder, we use Shape-Context [21] features (computed around Hessian-Laplace interest points [20]) and f_c is an index to the best-matching visual word in a learned vocabulary. As object parametrisation we use $\lambda = (\lambda_x, \lambda_y, \lambda_s)$, *i.e.*, the object's position and size, resp.

Invariant Space. It is important to define an invariant, which relates the position of a feature and the target object in an invariant manner. It is the building block from which the final detector inherits its invariance properties (*e.g.* shift- and scale-invariance). Possible mappings of the observables (λ, f) into a x-translation- and scale-invariant space are

$$\mathbb{I}_f(\lambda, f) = \left[\frac{\lambda_x - f_x}{f_s}, \log \frac{\lambda_s}{f_s} \right], \quad \mathbb{I}_\lambda(\lambda, f) = \left[\frac{f_x - \lambda_x}{\lambda_s}, \log \frac{f_s}{\lambda_s} \right]. \tag{2}$$

For brevity's sake, we dropped the y-coordinate as it is analogous to x. In this definition, $\mathbb{I}_f$ considers a feature-centric coordinate frame, while the $\mathbb{I}_\lambda$ opts for an object-centric one. Using the logarithm accounts for the multiplicative nature of the scale ratio and will be helpful later. The difference of object and feature location relates the two invariants *i.e.*,

$$\underbrace{\begin{pmatrix} f_s \mathbb{I}_{f,x} \\ \mathbb{I}_{f,s} \end{pmatrix}}_{\stackrel{def}{=} \mathbb{I}_f(f)} = \underbrace{\begin{pmatrix} \lambda_x - f_x \\ \log \lambda_s - \log f_s \end{pmatrix}}_{\lambda - f} = \underbrace{-\begin{pmatrix} \lambda_s \mathbb{I}_{\lambda,x} \\ \mathbb{I}_{\lambda,s} \end{pmatrix}}_{\stackrel{def}{=} \mathbb{I}_\lambda(\lambda)}, \tag{3}$$

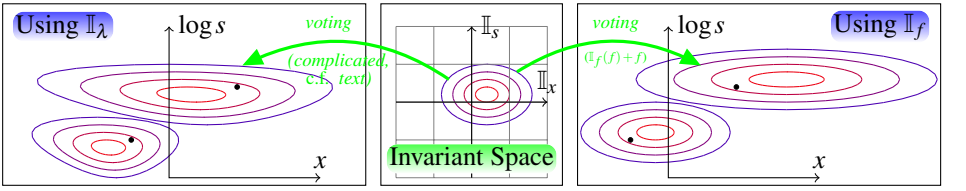

Figure 2: *Analysis of deformations caused during the Hough voting process. Approaches which use invariant $\mathbb{I}_\lambda$ (left, middle) or $\mathbb{I}_f$ (middle, right) are shown. The middle frame shows the contour lines of a single Gaussian of a learned mixture model. The left (right) frame depicts the hypothesis space along with the position of two features (dots). Given a feature, the Gaussian (in the invariant space) can be mapped into the hypothesis space. This is the act of voting. This mapping strongly depends on the chosen invariant. In case of $\mathbb{I}_f$, the resulting contours are still elliptic (*i.e.*, again a Gaussian), while $\mathbb{I}_\lambda$ leads to complicated, non-linear distortions.*

where the subscripts x and s refer to the first and second coordinate of the invariants, respectively. Despite the close relation, there is a subtle difference due to the dependency on f_s or λ_s, which suggests the preferable usage: predicting objects $\lambda \sim \mathbb{I}_f(f)+f$ given a feature f or sampling features $f \sim \mathbb{I}_\lambda(\lambda)+\lambda$ to verify an object hypothesis λ. The former approach is taken by the Generalised Hough Transform (*e.g.* [1, 14]), while the latter corresponds to a sliding window classifier scenario (*e.g.*, [6, 12]). The converse use is unfavourable, due to complicated couplings (*e.g.*, $f_x = \lambda_x - \lambda_s \exp(-\mathbb{I}_{f,s})\mathbb{I}_{f,x}$), but not impossible (see Fig. 2).

Footprint. Given an invariant $\mathbb{I}$, we define the joint mapping of an object-feature pair (λ, f) as a weighted Dirac delta function $f_\omega \delta_{f_c, \mathbb{I}(\lambda,f)}$ which is zero everywhere but at the 4D-point $[f_c, \mathbb{I}(\lambda, f)]$, and which integrates to f_ω. The object footprint is the superposition of all features,

$$\phi(\lambda, I) = \sum_{f \in \mathcal{F}} f_\omega \delta_{f_c, \mathbb{I}(\lambda,f)}. \tag{4}$$

In classical sliding-window systems, this 4D function is discretised and represented by a histogram. We avoid the discretisation and plug it directly into the score (Eq. (1)). Thus, the inner product of the two functions evaluates as

$$\langle \phi(\lambda, I), W \rangle = \sum_{f \in \mathcal{F}} f_\omega W(f_c, \mathbb{I}(\lambda, f)), \tag{5}$$

i.e., as point evaluations of the weight function. This form makes the connection to the feature-driven Hough-transform explicit. The summand $f_\omega W(f_c, \mathbb{I}(\cdot, f))$ represents the "vote" cast by each feature. Although no longer really visible, this formulation is *equivalent* to a standard sliding-window classifier (*e.g.*, [6]) when considering the object-centric invariant $\mathbb{I}_\lambda$ and a histogram for W. Note that we are however not restricted to this choice and even more general invariants than Eq. (2) are possible. They may account for *e.g.* variable aspect ratio, view-point changes, *etc*. A nice property of our framework is that the modelling of such geometric transformations is made explicit (*i.e.*, through the definition of $\mathbb{I}$) and decoupled from the rest of the system.

Non-Contributing Features. Objects have a finite extent. Thus, the weight function W has a compact support and is zero outside that range. As a consequence, many (far away) features do not contribute to the score, *i.e.*, $W(f_c, \mathbb{I}(\lambda, f)) = 0$. In practice, identifying and excluding such features reduces runtime significantly. Doing that properly is *not* a detail, but an important aspect of an algorithm. As it strongly depends on the representation of W, we postpone further discussions to Sec. 3.

Sliding Window	Hough Transform
for $\lambda \in \Lambda$	**for** $f \in \mathscr{F}$
for $f \in \{\bar{W}(\lambda, f) \neq 0\}$	**for** $\lambda \in \{\bar{W}(\lambda, f) \neq 0\}$
$S(\lambda)$+=$\bar{W}(\lambda, f)$	$S(\lambda)$+=$\bar{W}(\lambda, f)$
end	**end**
end	**end**

Figure 3: *Pseudo-code for sliding-window and Hough transform. We use the shortcut* $\bar{W}(\lambda, f) := W(f_c, \mathbb{I}(\lambda, f))$. *Sliding-window detectors process all hypotheses sequentially. For each hypothesis, the entire score function is evaluated* at once *which involves finding contributing features. Contrarily, the Hough transform is feature driven and processes all objects* in parallel*: each feature adds its non-zero contributions to the corresponding objects,* i.e.*, an accumulation of voting patterns in the search space.*

2.2 Discussion

Bottom-up? Sliding-Window vs. Hough-Transform. Often, sliding-window detectors are considered fundamentally different from Hough-transform approaches. The former reasons top-down, while the latter is assumed to work bottom-up. An attempt to relate the two concepts has also been investigated by Williams *et al*. [28], but their approach relies on a Taylor approximation. In our framework, the duality of the two approaches emerges naturally, as can be seen from Eq. (5). The left hand side represents the object-centric view taken by sliding-window detectors. There, the complete footprint ϕ (usually a histogram) is extracted and compared to the model. The right hand side expresses the feature-centric formulation of Hough-transform based systems. Note that both views are *equivalent*. The actual difference is more of an *algorithmic* nature, *i.e*., how the score is evaluated for all possible object hypotheses. This difference can be explained by an interchange of loops as illustrated in Fig. 3. Thus, in our opinion, both approaches follow the same top-down reasoning and only feature extraction (as pre-processing) should be seen as a bottom-up process.

Implicit Shape Model (ISM) [14]. The score function of the Implicit Shape Model (ISM) [14] (without the Minimum Description Length (MDL) post-processing) is defined as a probability density over the whole search space, *i.e*.

$$p(\lambda|I) = \sum_{f \in \mathscr{F}} p(\lambda|f)p(f|I). \tag{6}$$

As mentioned in the introduction, this formulation bears various problems which we will discuss now. We argue that marginalisation is not justified. It implies that all features f are possible realisations of a *single* random variable. Hence, only *one* feature could actually be observed. However, we do observe *all* features concurrently. Thus, each feature should be modelled with a separate random variable, *i.e*., $p(\lambda, f_1, f_2, \ldots f_n)$, as done correctly in other probabilistic setups [26]. For tractability, features are assumed to be independent which then leads to the factorisation $p(\lambda)\prod_i p(f_i|\lambda)$, *i.e*., a multiplicative combination instead of the additive one of ISM. The summation is however crucial for the voting concept. Moreover, as pointed out by Kittler *et al*. [11], additive combination of evidence is more robust than multiplicative ones, which is in line with the empirically observed robust detection performance of ISM. Thus, the question is then how to still justify the summation. A second problem is due to the normalisation constraint of the density $p(\lambda|I)$. This (global) normalisation implies that every feature affects the score of every hypothesis. Unless contextual information is explicitly modelled (which is not the case), such long-range interactions are not justified.

Our framework overcomes these issues by reasoning in a sliding-window manner. Each hypothesis is scored independently, *i.e*., there is no global normalisation, and the compact support of W ensures that the features have only local influence. The summation of feature contributions results from the definition of the footprint and the linear model (which both are

valid modelling choices). Last but not least, notice that there is no restriction on the function W. Thus, we are free to choose any suitable representation and learning algorithm. In particular, discriminative learning (*e.g.* with SVMs, *c.f.* [13]) may be used which was not possible in the ISM. An extension to ISM which allows for discriminative learning has also been presented in [18]. However, our framework is more general in two ways. Their discriminative learning is restricted to a per-visual-word re-weighting (*i.e.*, all spatial positions encounter the same re-weighting) and the final weights remain positive. Moreover, they also adopt the questionable marginalisation argument. In our framework, the only restriction is that the features contribute linearly which is inherent in the Hough transform paradigm. However, this is not a huge burden since most (fast) detectors rely on linear models [6, 7, 12, 14].

Multiple Objects. Detection of multiple objects is another point where, in our opinion, the sliding-window reasoning (underlying our framework) is cleaner. Eq. (6) (*i.e.*, ISM without MDL) answers the question "*Where* is the object?" and completely ignores the important questions "*Is* there an object?"[1] or "How many objects are there?". Sliding-window reasoning on the other hand answers for each position independently if it contains an object or not, and postpones model selection (*i.e.* "How many objects are there?") to later components in the object detection pipeline. Thus, sliding-window allows for multiple detections, but it does not account for already explained evidence. Hence, it tends to re-detect the same object. A possible way to resolve this problem is to limit each feature to support only one single object. In this work, we consider a greedy strategy: once the best object is found we eliminate all features which explain this object and restart the search for more objects. This procedure automatically suppresses the score of nearby hypotheses and relates to the usual non-maximum suppression post-processing. Clearly, the MDL based optimisation strategy implemented in ISM [14] is more powerful as all hypotheses compete for evidence. This is an advantage in situations with strongly overlapping objects. However, MDL is applied as post-processing and only considers hypotheses found by greedy detection.

3 Gaussian Mixture Model & Gradient Search

So far, we have presented an abstract framework. A concrete system is obtained by specifying the geometric invariant and the representation of the weight function W. In this paper we will use $\mathbb{I}_f$ and a Gaussian Mixture Model (GMM) representation, which allows for efficient gradient-based search. Concerning learning we follow a generative approach inspired by ISM [14]. For each visual word c, we learn a generative model $W(c,\mathbb{I}(\lambda,f)) = p_c(\mathbb{I}(\lambda,f))$, *i.e.*, the occurrence distribution in ISM terminology. Learning is performed by mapping all features from the training objects into the invariant space, where the distribution can be estimated. ISM considers non-parametric kernel density estimators for this. Their strong dependence on the training data (in terms of storage requirement and evaluation time) is a clear downside. Instead, we consider GMMs[2], which overcome this limitation. Search can be performed using local gradient-based search methods [3], analogue to mean-shift in ISM. The mixture model is learned using an Expectation-Maximisation (EM) implementation[3], which automatically adapts the number of mixture components using a split/merge heuristic.

Voting. In this setup, the feature centric-invariant $\mathbb{I}_f$ has two advantages. The reason is the simple relation $\lambda = \mathbb{I}_f(f) + f$, which leads to just a translation and scaling given an observed feature f. Hence, each mixture component in the invariant space can be mapped

[1] Assuming that an object is present may be reasonable in a tracking context, but not for object detection.
[2] Using $\log \lambda_s/f_s$ is important here, as the positivity constraint of $\lambda_s/f_s > 0$ could not be satisfied with GMM.
[3] http://www.npt.nuwc.navy.mil/Csf/software.html

into the hypothesis space by adapting mean and covariance of each Gaussian (*c.f.* Fig. 2). This is the actual *voting*. Note that such voting is not possible with $\mathbb{I}_\lambda$, because its non-linear distortion breaks the algebraic structure of a Gaussian. Moreover, also the computation of the (score-)gradient (with respect to λ) is simpler than with $\mathbb{I}_\lambda$. A final remark concerns a connection to ISM. In [15], a scale-adaptive version of mean-shift is introduced. This scale-adaptation actually accounts for the above-mentioned scaling. Thus, ISM implicitly operates in the invariant space (which is not defined in ISM [15]) instead of the hypothesis space.

Contributing Features & Candidate Generation. In this setup, the idea of contributing features translates to mixture components which contribute significantly to the score. We assume that Gaussians contribute significantly within two times the standard deviation. Thus, we keep a *coarse* 3*D*-(location,scale) grid over the hypothesis space. Each grid cell keeps a list of contributing mixture components. Note that each such cell corresponds to a set of object hypotheses. We adopt the idea of computing score bounds [12] (on such sets) to initialise the local search. We consider a crude estimate by summing the peak contribution (*i.e.*, the value at the mean) of each Gaussian in a given cell. Local maxima of this bound serve as starting points for the gradient-based, local refinement [3].

Algorithm. Our algorithm keeps a list of candidates that is populated as described above. After gradient-based refinement, the best candidate is reported as detection, and candidates with a score below a threshold are removed from the list. Features with a significant contribution to the detected object are eliminated from the system (*i.e.*, the corresponding Gaussians are removed from the cells and the bound is updated). Then, the algorithm restarts to find more objects. Keeping a list of candidates helps saving computation in subsequent iterations. The reason is that feature removal only changes the score function *locally*. Hence, the refinement of most candidates is likely to converge quickly.

Modified GMM. Contrary to the ISM [14], we are not forced to use a density $p_c(\mathbb{I})$. We exploit this freedom and consider a simple re-scaling, *i.e.* $W(c,\mathbb{I}) = p_c(\mathbb{I})/(\max_{\mathbb{I}} p_c(\mathbb{I}))$, which shows significant performance improvements. The aim is that the maximal weight (of every visual word) is approximately one. An intuitive explanation is as follows. The maximal weight of multi-modal occurrence distributions tends to be smaller than the one of rather peaked distributions. Thus, depending on which visual words are activated, the object score may be lower, which is undesirable. Learning good re-weightings is also possible [18].

3.1 Experiments

Datasets. We evaluate our system on two benchmark datasets consisting of motorbikes [9] and side views of pedestrians [14]. The former includes 153 training and 115 test images, whereas the latter consists of 426 training and 205 test images. In order to compare our results to the published baseline, we use the same evaluation criterion as in [9, 14]. A detection is accepted as correct if its bounding-box overlap with the ground-truth annotation is above 50%; multiple detections on the same object are counted as false positives. Performance is measured using precision-recall curves which is the common measure for these datasets.

GMM vs. modified GMM. The probabilistic argument given in [16] defines the sum of densities as the objective function to hypothesise object positions. This roughly corresponds to the GMM without modification. Fig. 4(center) illustrates the effect of using the modified GMM. The results are reported along with the baseline performance from [9, 14, 16]. The difference between the two objective functions is actually fairly small, *i.e.* only a rescaling of each density function by a factor. Looking at the effect of this little change on the performance, an astonishing impact can be observed. Namely, the modified GMM based objective

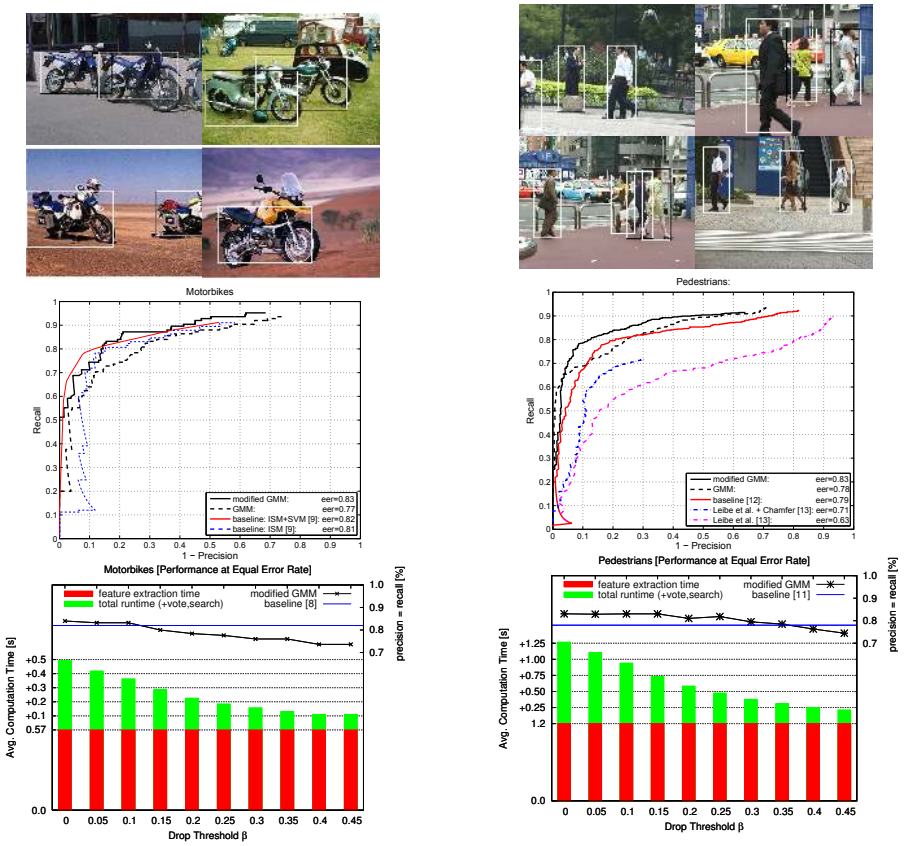

Figure 4: *Quantitative evaluation on two benchmark datasets of motorbikes (left) and pedestrians (right). (top): sample images of the datasets along with detections. (middle): performance compared to published baseline; (bottom): detection time and recognition performance with our reduced model. The detection time can be reduced significantly at the cost of only a small decrease in accuracy.*

function clearly outperforms the density based objective function and yields comparable performance to (motorbikes) or even improves (pedestrians) on the baseline. It has to be stressed at this point that, compared to the baseline systems, our approach does *not* need to perform a final MDL or SVM verification of the hypotheses.

Runtime. The last experiment measures the runtime of our system (measured on a 3GHz Intel Core 2). For this experiment we use an external Hessian-Laplace feature extractor from [20], which takes about $0.6s$ on a typical motorbike image (340×255) and $1.2s$ on a pedestrian image (480×360). This can be reduced dramatically by changing to similar SURF features [2] that allows for GPU-implementations [5] which run at about 100Hz. Thus, feature extraction is negligible compared to the rest of our algorithm. Moreover, the structure of our parametric model allows for a simple heuristic to improve detection time. Let us denote the peak contribution of mixture component i by γ_i. Then, we drop all mixture components whose peaks are smaller than β percent of the maximal peak, *i.e.*, $\gamma_i < \beta \max_k \gamma_k$. The average detection time, as well as the achieved performance at equal error rate is reported in Fig. 4(bottom) for both datasets. We see that for values up to $\beta = 0.1$ and 0.15, respectively, the computation time (ignoring feature extraction) can be reduced by about a third without decrease in accuracy. This yields a detection time of about $0.35s$ for a motorbike image. If

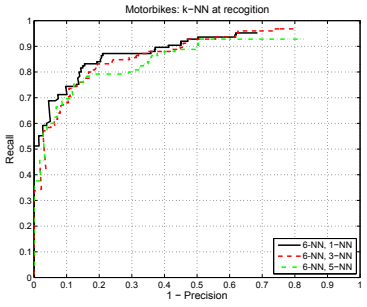

Figure 5: *The consequence of applying additional soft-matching during recognition. During learning, the soft-matching is fixed to 6-NN matching while we vary between 1 to 5 NN during recognition. The performance clearly drops when deviating form the hard-NN matching rule.*

runtime is of prime importance, we can also sacrifice some accuracy in favour of speed. On the pedestrians, for example, we can increase the threshold to $\beta = 0.35$, which yields the same equal error rate as the baseline. Doing so, the detection time decreases from initially $1.25s$ to only $0.25s$. This simple, but very effective heuristic exploits the structure of our parametric model and would not be easily realisable in a non-parametric framework [14].

4 Soft-Matching

Soft-matching is a heuristic where a feature activates multiple codewords from the visual vocabulary, instead of just the best-matching one. As has been mentioned by other authors [8, 14, 22, 24], this increases robustness against noise and quantisation effects. Unfortunately, the downside of soft-matching is an increased computational cost.

Soft-Matching during Recognition? Although runtime is of prime importance during recognition, soft-matching is usually still applied. We claim that this is *unnecessary* which can be seen as follows. Soft-matching can be understood as a blurry activation of codewords, instead of a sharp peak at the best match. In our framework, this corresponds to blurring the Dirac functions (Eq. (4)). Thus, using soft-matching corresponds to a blurred footprint $B\phi$ where B denotes the blurring operator. It can be expressed as a matrix because the visual words are discrete. Consequently, soft-matching can be avoided during recognition by incorporating it completely into the training, *i.e.*, $\langle B\phi, W\rangle = \langle \phi, B^T W\rangle$. Thus, instead of blurring the footprint, we blur the model, which can be done off-line. In other words, stronger blurring during learning and NN-matching during recognition. Hence, the system benefits from soft-matching's robustness without having the computational overhead during recognition. In Fig. 5 we see the result of using $\{1,3,5\}$-NN during detection in addition to $6-$NN during learning. Additional soft-matching clearly degrades the performance. In other words, applying soft-matching twice leads to an over-smoothing which causes under-fitting.

A similar argument can be given for *spatial pyramid* histograms as used in [12]. There, B is the mapping from the *flat* histogram to the multi-layer pyramid representation. $B^T W$ is the back-transform which compresses the learned model back into a (lower-dimensional) flat histogram. Thus, the pyramid is *not* needed during detection which, again, reduces algorithmic complexity and runtime.

Benefits. Moving soft-matching entirely to the learning stage has clear advantages. On the one hand, NN-matching during recognition is much simpler and faster. In the above example, changing from $k=5$ to 1 incurs a speedup of approximately 4. On the other hand, stronger soft-matching during learning causes more occurrences which makes density estimation more stable. Contrary to non-parametric density estimators [14], our GMM compresses the increased number of occurrences. Hence, the overhead of soft-matching during recognition is very low. In summary, soft-matching should only be applied during learning where it causes model regularisation which is controlled by the degree of soft-matching.

5 Conclusion

We introduced a hybrid framework for sliding-window and Hough-transform based object detection which combines the advantage of both approaches. Our framework overcomes various issues of ISM and gives a sound explanation for the *voting* paradigm. Although we adopt ISM's generative training procedure, the framework is generic and other training methods are possible, in particular discriminative ones. Moreover, our *invariant space* concept nicely decouples the modelling of geometric transformation of objects, which will be helpful in multi-view setups. Contrary to usual sliding-window detectors, we are not restricted to histogram representations and demonstrate a Gaussian mixture model which allows for efficient gradient-based search. Finally, we formalise the idea of soft-matching/spatial-pyramid-histograms within our framework. We show that their increased complexity can be avoided during detection without missing their positive effect.

Acknowledgements

The authors wish to thank the Swiss National Fund (SNF) for support through the CASTOR project (200021-118106).

References

[1] D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. van Gool. Surf: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[3] M. Á. Carreira Perpiñán. Mode-finding for mixtures of gaussian distributions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1318–1323, 2000.

[4] O. Chum and A. Zisserman. An exemplar model for learning object classes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[5] N. Cornelis and L. Van Gool. Fast scale invariant feature detection and matching on programmable graphics hardware. In *Computer Vision and Pattern Recognition (CVPR) Workshop*, June 2008.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[7] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[8] V. Ferrari, F. Jurie, and C. Schmid. Accurate object detection with deformable shape models learnt from images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[9] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating representative and discriminant models for object category detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2005.

[10] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[11] J. Kittler, M. Hatef, R. Duin, and J. Matas. On combining classifiers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):226–239, 1998.

[12] C. H. Lampert, M. B. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[13] A. Lehmann, B. Leibe, and L. van Gool. Feature-centric efficient subwindow search. In *Proceedings of the IEEE International Conference on Computer Vision*, 2009.

[14] B. Leibe, A. Leonardis, and B. Schiele. Robust object detection by interleaving categorization and segmentation. *International Journal of Computer Vision*, 77(1-3):259–289, 2008.

[15] B. Leibe and B. Schiele. Scale-invariant object categorization using a scale-adaptive mean-shift search. In *Proceedings of the DAGM Symposium*, 2004.

[16] B. Leibe, E. Seemann, and B. Schiele. Pedestrian detection in crowded scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

[17] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3D feature maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[18] S. Maji and J. Malik. Object detection using a max-margin hough transform. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[19] S. Maji, A. C. Berg, and J. Malik. Classification using intersection kernel support vector machines is efficient. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[20] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.

[21] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, 2005.

[22] B. Ommer and J. M. Buhmann. Learning the compositional nature of visual objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[23] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *Proceedings of the European Conference on Computer Vision*, 2006.

[24] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[25] H. Schneiderman and T. Kanade. Object detection using the statistics of parts. *International Journal of Computer Vision*, 56(3):151–177, 2004.

[26] E. B. Sudderth, A. Torralba, W. T. Freeman, and A. S. Willsky. Learning hierarchical models of scenes, objects, and parts. In *Proceedings of the IEEE International Conference on Computer Vision*, 2005.

[27] P. A. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, 2004.

[28] C. K. I. Williams and M. Allan. On a connection between object localization with a generative template of features and pose-space prediction methods. Technical Report 0719, University of Edinburgh, 2006.