Accurate Object Detection with Deformable Shape Models Learnt from Images

Vittorio Ferrari, Frédéric Jurie, Cordelia Schmid

Accurate Object Detection with Deformable Shape Models Learnt from Images

Vittorio Ferrari, Frederic Jurie, Cordelia Schmid
LEAR group - INRIA Grenoble
{vittorio.ferrari,frederic.jurie,cordelia.schmid}@inrialpes.fr

Abstract

We present an object class detection approach which fully integrates the complementary strengths offered by shape matchers. Like an object detector, it can learn class models directly from images, and localize novel instances in the presence of intra-class variations, clutter, and scale changes. Like a shape matcher, it finds the accurate boundaries of the objects, rather than just their bounding-boxes. This is made possible by 1) a novel technique for learning a shape model of an object class given images *of example instances; 2) the combination of Hough-style voting with a non-rigid point matching algorithm to localize the model in cluttered images. As demonstrated by an extensive evaluation, our method can localize object boundaries accurately, while needing no segmented examples for training (only bounding-boxes).*

1. Introduction

Object class detection is a central challenge of computer vision. Recent research has achieved impressive results, such as handling cluttered images [13], training from few examples [18], and exploiting both contour [15, 16] and appearance information [19]. However, most existing approaches localize objects up to a rectangular bounding-box.

In this work [1] , we want to go a step further and localize object *boundaries*. Our approach aims at bridging the gap between shape matching and object detection. Classic non-rigid shape matchers [2, 3] obtain accurate point correspondences, but take *point sets* as input. In contrast, we build a shape matcher with the input/output behavior of a modern object detector: it learns shape models *from images*, and automatically localizes them in cluttered images.

Our approach makes three contributions. First, we introduce a technique for learning the prototypical shape of an object class as well as a statistical model of intra-class deformations, given image windows containing training instances (figure 1a). The challenge is to determine which contour points belong to the *class boundaries*, while discarding background and details specific to individual instances (*e.g*. mug labels). Note how these typically form the large majority of points, yielding a poor signal-to-noise ratio. The task is further complicated by intra-class variability: the shape of the object boundary varies across instances.

Second, we localize the boundaries of novel class instances by employing a Hough-style voting scheme [13, 15, 16] to automatically initialize a non-rigid shape matcher [3]. This combination makes accurate, pointwise shape matching possible even in severely cluttered images, where the object boundaries cover only a small fraction of the contour points (figure 3a).

Third, we constrain the shape matcher [3] to only search over transformations compatible with the learned, class-specific deformation model. This ensures output shapes similar to class members, improves accuracy, and helps avoiding local minima.

These contributions result in a powerful system, capable of detecting novel class instances and localizing their boundaries in cluttered images, while training from objects annotated only with bounding-boxes.

2. Related works

Object localization. A few previous approaches go beyond bounding-box precision [1, 12, 13, 17, 18, 20], but most of them require segmented training objects. Todorovic and Ahuja [18] can learn when given only image labels, but the number of training images is strongly limited by the computational complexity of the learning stage. As [18], LOCUS [20] also learns from image labels, but it is based on entirely different techniques than our work. Besides, we demonstrate detections on more challenging images (small objects in large cluttered images; wide range of scales), and test our method as a full object detector, by evaluating detection rates and false-positives rates (also on images *not* containing the object). In contrast, LOCUS [20] focuses on segmentation accuracy and does not report these statistics. The work of Berg et al. [1] is related to ours in that they also cast object detection as a point-matching problem. As an important difference, it treats training images individually, without learning a shape or deformation model. Moreover, it requires hand-segmented training images. Although

[1]This research was supported by the EADS foundation, INRIA and CNRS. V. Ferrari was funded by a fellowship of the EADS foundation.

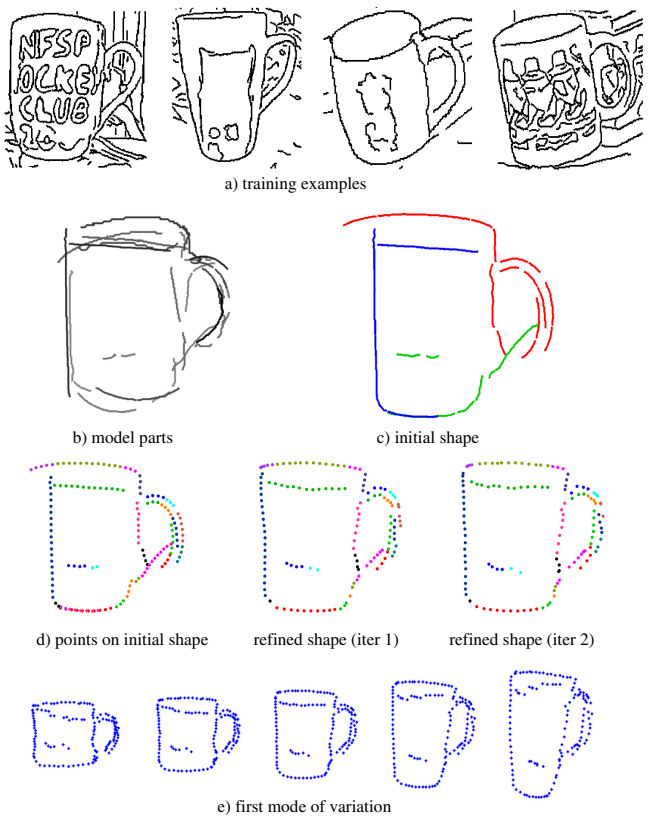

Figure 1. *a) 4 out of 24 training examples. b) Model parts. c) Selected occurrences. d) Shape refinement iterations. e) First mode of variation (mean shape in the middle).*

ideas for training without segmentations are mentioned, the reported shape-matching results do use them.

Learning and matching shapes. Numerous methods appeared for learning and matching deformable shape models [4, 6, 9, 10]. Several approaches learn global modes of variation using PCA, following the seminal work of Cootes on Active Shape Models [4, 10]. Models of different nature have also been proposed, such as pairwise geometric relations between landmarks [6], or representing shapes as configurations of independently deforming triangles [9]. The main point is that all these techniques need *shapes* as input (*i.e*. sets of points on object outlines) [4, 9, 10], or, equivalently, clutter-free cartoon drawings [6]. Besides, only some methods automatically extract the necessary landmark points and their correspondences from the training shapes [6, 10], while others require them as additional input [4, 9]. In contrast, in this paper we automatically learn shapes, correspondences, and deformations from *images*.

Most existing shape matching techniques work in cluttered images only if initialized near the object to be found [3, 4, 5]. In our approach instead, this initialization is *automatic* (section 4.1). Two exceptions are [9], which has high computational complexity, and [7], against which we compare experimentally in section 5.

3. Learning a class-specific shape model

Given image windows with example instances (figure 1a), we learn the shape of the class and the principal intra-class deformation modes. To achieve this, we present a technique for discovering which contour points belong to the common class boundaries (*e.g*. the outline of the mug, as opposed to the varying labels), and for putting them in full point-to-point correspondence across the training examples. The technique is composed of four steps (figure 1b-d). First, we determine model parts as local contour features (subsection 3.1) frequently reoccurring with similar locations, scales, and shapes (subsection 3.2). Next, an initial shape is assembled by sampling a specific feature for each model part from the training examples (section 3.3). The shape is then matched back on the training images (subsection 3.4), thus producing different shape variations, all in point-to-point correspondence. These are used to refine the model shape and to learn intra-class deformations (subsection 3.5).

3.1. Local contour features

We employ the scale-invariant local contour features recently proposed by [8]. Edgels are found by the Berkeley edge detector [14], and then grouped into pairs of adjacent, approximately straight segments (figure 2a). Each such *PAS* feature P has a location (mean over the two segment centers), a scale (distance between the segment centers), a strength (mean edge detector confidence), and a descriptor invariant to translation and scale. The descriptor encodes the shape of the PAS, by the segments' orientations and lengths normalized by scale, and their relative location.

PAS features are particularly suited to our framework. First, they are robustly detected because they connect segments even across gaps between edgel-chains. Second, the descriptor records only properties of the two segments, without including other nearby edgels (as opposed to patch descriptors). This is valuable as non-boundary edgels just outside/inside the object would otherwise corrupt the descriptor of a PAS lying on the object boundary. Finally, since a correspondence between two PAS induces a translation and scale change, they can be easily used within a Hough-style voting scheme for object detection [13, 15, 16].

We construct a codebook by clustering all PAS inside the training bounding-boxes according to their descriptors (see [8] for details of the similarity measure between descriptors). For each cluster, we retain the centermost PAS, minimizing the sum of dissimilarities to all the others. The codebook is the collection of these centermost PAS, the *PAS types* (figure 2b). A codebook is useful for efficient matching, since all features similar (up to some threshold) to a type are considered in correspondence.

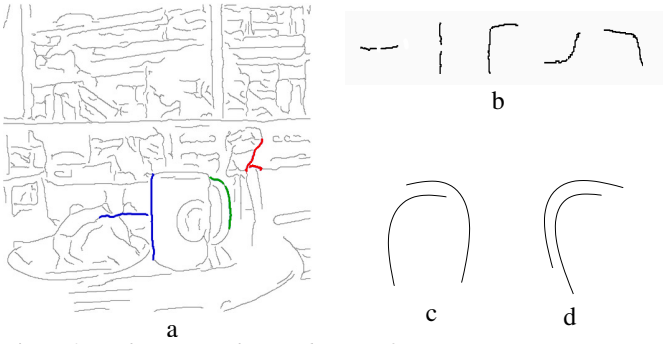

Figure 2. *a) three example PAS. b) some frequent PAS types. c) two model parts with high connectedness. d) two model parts with even higher connectedness (equivalent parts).*

3.2. Finding model parts

The key insight is that PAS belonging to the desired class boundaries will reoccur consistently across several training instances at similar locations and scales, and with similar shapes. In contrast, PAS not belonging to the common boundaries are not correlated across different examples.

The first step of our algorithm is to align the training bounding-boxes by transforming them into a zero-centered rectangle with unit height and width equal to the geometric mean of the training aspect-ratios (*i.e*. width over height). In addition to removing translation and scale differences, this effectively cancels out shape variations due to different aspect-ratios (*e.g*. tall mugs versus coffee cups).

The core of the algorithm maintains a separate voting space for each PAS type, where it accumulates votes from all training PAS. Each PAS votes for the existence of a part of the class boundary with shape, location, and size like its own. More precisely, a PAS is soft-assigned to all codebook types within a dissimilarity threshold, and casts a vote in each corresponding accumulator space. Votes are weighted proportionally to the PAS' edge strength, and inversely proportionally to the shape dissimilarity to the codebook type.

The output of the algorithm are the local maxima in location and scale of the accumulator spaces. Each maximum yields a *model part*, which has a specific location and size relative to the canonical bounding-box, and a specific shape (the codebook type corresponding to the accumulator space where the maximum was found). Moreover, the value of the local maximum provides a measure of the confidence that the part really belongs to the class boundaries (part *strength*).

This procedure benefits from adopting PAS as basic shape elements (as opposed to individual edgels): it is highly unlikely that a significant number of unrelated PAS will accidentally have similar locations, scales, and shapes. Hence, recurring PAS stemming from the desired class boundaries will tend to form peaks in the accumulator spaces, whereas background clutter and details of individual training instances won't.

The proposed algorithm has two important properties. First, it sees all training data *at once*, and therefore reliably selects parts and robustly estimates their locations/scales/shapes. This is more stable than matching pairs of training instances and then recombines their output a posteriori. Second, its complexity is *linear* in the number of training instances, so it can learn from large training sets efficiently.

3.3. Assembling the initial model shape

The learned model parts already capture the shape of the object class quite well (figure 1b). The outer boundary of the mug and the handle hole are included, whereas the label is largely excluded. However, this doesn't look like the outlines of an average class member yet. There are often multiple stokes along what should be a single line. Adjacent parts don't fit well together in terms of their relative locations and sizes, resulting in short, discontinuous lines. This is because individual parts are learnt *independently*, without trying to assemble them into a proper whole shape. In the following we describe how to build such a shape, made of single-stroked, long continuous lines.

Let us notice that each model part occurs several times on the training examples. These occurrences present roughly similar, yet different alternatives for the part's location, size, and shape. Hence, we can assemble several variants of the overall shape by selecting different occurrences for each part. The main idea is to select occurrences so as to form larger aggregates of connected occurrences stemming from only a few training images. The intuition being that occurrences from the same training image fit together naturally.

A training PAS o is an occurrence of model part P if they are sufficiently similar according to a confidence measure based on their shapes, locations, and scales. We denote an occurrence with $P \rightarrow o$, and its confidence with $\mathrm{conf}(P \rightarrow o) \in [0, 1]$.

Recall that a model part P has two segments P_1, P_2. Let the equivalence between two model segments be

$$\mathrm{eq}(P_i, Q_j) = \sum_{\{s | P_i \rightarrow s, Q_j \rightarrow s\}} \mathrm{conf}(P_i \rightarrow s) + \mathrm{conf}(Q_j \rightarrow s) \tag{1}$$

with $i, j \in 1, 2$ and s any training segment on which both P_i, Q_j occur. Two model segments have high equivalence if they frequently occur on the same training segments. Lastly, we define the connectedness between two model parts P, Q as

$$\begin{aligned} \mathrm{conn}(P, Q) \quad = \quad & \max(\mathrm{eq}(P_1, Q_1) + \mathrm{eq}(P_2, Q_2), \\ & \mathrm{eq}(P_1, Q_2) + \mathrm{eq}(P_2, Q_1)) \end{aligned} \tag{2}$$

i.e. the combined equivalence of their segments (for the best of the two possible segment matchings). Two parts have high connectedness if their occurrences frequently

share a segment (figure 2c+d). Moreover, two parts can even share *both* their segments. In this case, their connectedness is even higher, indicating they explain the same portion of the class boundaries. Equivalent model segments are the origin of the multiple strokes in figure 1b. Equation (2) measures connectedness/equivalence by smoothly integrating evidence over the whole training set.

The occurrence selection task can now be formulated precisely as follows. Find the assignment $\mathcal{A}(P) = o$ of occurrences to parts that maximizes the objective function (exactly one occurrence is assigned to each part):

$$\sum_{P} \text{conf}\left(P \rightarrow \mathcal{A}(P)\right) - \lambda_{img} N_{img} \qquad (3)$$
$$+\lambda_{conn} \sum_{P,Q} \text{conn}(P,Q) \cdot \mathbf{1}_{img}\left(\mathcal{A}(P), \mathcal{A}(Q)\right)$$

where the indicator function $\mathbf{1}_{img}$ takes value 1 if two occurrences stem from the same image, and 0 otherwise. N_{img} is the number of images contributing an occurrence to $\mathcal{A}$, and $\lambda_{img}, \lambda_{conn}$ are predefined weights. The first term of (3) prefers high confidence occurrences. The second term discourages scattering occurrences across many training images, while the third term favors assigning connected parts to occurrences from the same training image. Overall, the function encourage the formation of aggregates of good confidence *and* properly connected occurrences, which typically fit well together. Moreover, the last two terms push equivalent model segments to be assigned to the same training segments, hence suppressing multiple strokes.

Although function (3) cannot be optimized exactly, as the space of all possible assignments is huge, in practice the following approximation algorithm brings satisfactory results. We start by assigning the part which has the single most confident occurrence. Next, we iteratively consider the part most connected to those assigned so far, and assign it to the occurrence which maximizes (3).

Figure 1c shows an example of the selected occurrences. The shape is composed of three blocks, each from a different training image. Within each block, segments fit well together and form continuous lines. Nevertheless, there are still discontinuities between blocks, and some redundant strokes still remain (lower half of handle).

3.4. Model shape refinement

We can improve the model shape by treating it as a deformable point set and *matching it back* onto the training images. For this, the image edgels are now treated *individually*, no longer grouped into PAS features. We iterate over three steps:

1. *Backmatching.* The model shape is matched back to each training image, using an extension of the non-rigid robust point matcher by Chui and Rangarajan [3]. This powerful algorithm estimates a Thin-Plate Spline (TPS) transformation and at the same time rejects image points not corresponding to any model point (see subsection 4.2 for details). We initialize the algorithm by transforming the model shape so that its bounding-box perfectly aligns with the training bounding-box. This strong initialization makes the matcher very likely to succeed.

2. *Mean shape.* The shapes generated by backmatching are in *full point-to-point correspondence*, because they are all smooth deformations of the same initial shape. Hence, we apply Procustes analysis [4] to align them w.r.t. to translation, scale, and orientation. The model shape is now updated by setting it to the mean of the aligned shapes. The combined effects of backmatching and computing the mean shape are very beneficial (figure 1d-middle). Model segments move, bend, and stretch in order to form smooth, connected lines, and to recover the shape of the class well on a *global* scale (*e.g.* topmost and leftmost segments in figure 1d-middle). The reason for these improvements is that backmatching manages to deform the initial shape onto the class boundaries of the training images, thus providing a set of natural, well formed shapes. The averaging step then integrates them all into a generic-looking shape, and smoothes out occasional inaccuracies of the individual backmatches.

3. *Remove redundant points.* As another effect, the previous steps tend to crush multiple strokes (and other clutter points) onto the correct class boundaries. This results in redundant points, roughly coincident with other segments. We remove them by deleting any point lying very close to points from a stronger part. If a significant proportion of points ($> 10\%$) are removed, the procedure iterates to point 1. Otherwise, it is completed.

As shown in figure 1d-right, the running example improves further during the second (and final) iteration (*e.g.* handle). It now has a clean overall shape, and includes no background clutter and very little interior clutter. Notice how the fine scale structure of the double handle arc is correctly recovered.

3.5. Learning shape deformations

The previous subsection matches the model shape to each training image, and thus provides examples of the variations within the object class we want to learn. Since these examples are in full point-to-point correspondence, we can learn a compact model of the intra-class variations using the statistical shape analysis technique by Cootes [4].

The idea is to consider each example shape as a point in a $2p$-D space (with p the number of points on each shape), and model their distribution with Principal Component Analysis (PCA). The eigenvectors returned by PCA represent modes of variation, and the associated eigenvalues λ_i their importance (how much the example shapes deform along them, figure 1e). By keeping only the n largest eigenvectors $E_{1:n}$

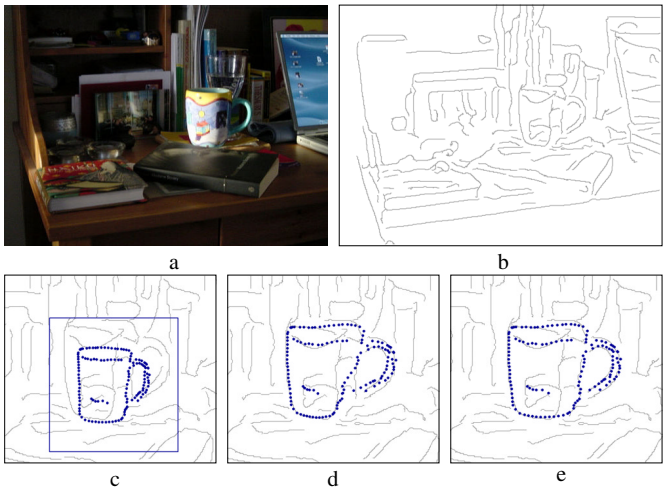

Figure 3. *a) A challenging test image and its edgemap b). The object covers only about* 6% *of the image surface, and only about 1 edgel in 17 belongs to its boundaries. c) Initialization with a local maximum in Hough space. d) Output shape with unconstrained TPS-RPM. e) Output shape with shape-constrained TPS-RPM.*

representing 95% of the total variance, we can approximate the region in which the training examples live by $\bar{x} + E_{1:n}b$, where $\bar{x}$ is the mean shape, b is a vector representing shapes in the subspace spanned by $E_{1:n}$, and b's i^{th} component is bound by $\pm 3\sqrt{\lambda_i}$. This defines the *valid region* of the shape space, containing shapes similar to the example ones. Typically, $n < 15$ eigenvectors are sufficient (compared to $2p \simeq 200$).

In subsection 4.3, we exploit this deformation model to constrain the matching of the model to novel test images. Notice that previous works on these deformation models require at least the example shapes as input [10], and many also need the point-to-point correspondences [4]. In contrast, we automatically learn shapes, correspondences, and deformations given just *images*.

4. Object detection

In this section we describe how to accurately localize novel object instances in a test image. We first obtain rough estimates for their location and scale based on a Hough-style voting scheme (subsection 4.1). These estimates are then used to initialize the non-rigid shape matcher [3] (subsection 4.2). This combination enables shape matching in cluttered images, and hence to localize object boundaries. In subsection 4.3, we constrain the matcher to explore only the region of shape space spanned by the training examples, thereby ensuring that output shapes are similar to class members.

4.1. Hough voting

In subsection 3.2 we represented the shape of the class as a set of PAS parts, each with a specific shape, location, size, and strength. Here we match these parts to PAS from the test image, based on their shape descriptors. Since a pair of matched PAS induces a translation and scale transformation, we let each match vote for the presence of an object instance at a particular image location (object center) and scale (in the same spirit as [13, 15, 16]). Votes are weighed by the shape similarity between the model part and test PAS, the edge strength of the PAS, and the strength of the part. Local maxima in the voting space define rough estimates of the location and scale of candidate object instances.

4.2. Shape Matching by TPS-RPM

For each candidate location l and scale s we obtain a point set V by centering the model shape on l and rescaling it to s, and a set X which contains all image edge points within a larger rectangle of scale $1.8s$ (figure 3c). Given this initialization, we want to put V in correspondence with the subset of points of X lying on the object boundary. We estimate the associated non-rigid transformation, and reject image points not corresponding to any model point, with the Thin-Plate Spline Robust Point Matching algorithm (TPS-RPM [3]).

TPS-RPM matches the two point sets $V = \{v_a\}_{a=1..K}$ and $X = \{x_i\}_{i=1..N}$ by applying a non-rigid TPS mapping $\{d, w\}$ to V (d is the affine component, and w the non-rigid warp). It estimates both the correspondence $M = \{m_{ai}\}$ between V and X, and the mapping $\{d, w\}$ that minimize an objective function of 1) the distance between TPS-mapped points and their corresponding points of X, 2) the regularization terms for the affine and warp components of the TPS. In addition to its inner $K \times N$ part, M has an extra row and an extra column which allow to reject points as unmatched.

Since neither the correspondence nor the mapping are known beforehand, TPS-RPM iteratively alternates between updating M, while keeping $\{d, w\}$ fixed, and updating the mapping with M fixed. M is a continuous-valued soft-assign matrix, allowing the algorithm to evolve through a continuous correspondence space, instead of jumping around in the space of binary matrices (hard correspondence). It is updated by setting m_{ai} as a function of the distance between x_i and v_a mapped by the TPS. The update of the mapping fits a TPS $\{d, w\}$ between V and the current estimate of the corresponding points

$$Y = \{y_a = \sum_{i=1}^{N} m_{ai} x_i\} \tag{4}$$

This minimizes the proximity of TPS-mapped points to y_a under the influence of the regularization terms, which penalize local warpings w and deviations of d from the identity.

The optimization procedure is embedded in a deterministic annealing framework by introducting a temperature parameter T, which decreases at each iteration. This gradually makes M less fuzzy, progressively approaching a hard correspondence matrix. At the same time, the regularization terms get less weight. Hence, the TPS is rigid in the beginning, and gets more and more deformable as the iterations

continue. These two phenomena enable TPS-RPM to find a good solution even when given a rather poor initialization. At first, when the correspondence uncertainty is high, each y_a essentially averages over a wide area of X around the TPS-mapped point (equation (4)), and the TPS is constrained to near-rigid transformations. As the temperature decreases, M looks less and less far, and pays increasing attention to the differences between matching options from X. Since the uncertainty diminishes, it's safe to let the TPS looser, freer to fit the details of X more accurately.

We have extended TPS-RPM by adding two terms to the objective function: the orientation difference between corresponding points (minimize), and the edge strength of matched image points (maximize). In our experiments, these extra terms made TPS-RPM more accurate and stable, *i.e*. it succeeds even when initialized farther away from the best location and scale.

4.3. Constrained shape matching

TPS-RPM treats all shapes according to the same *generic* TPS deformation model, simply preferring smoother transformations. This might result in shapes unlike any of the training examples. Here, we extend TPS-RPM with the *class-specific* deformation model learned in subsection 3.5. We constrain the optimization to explore only the valid region of the shape space, containing shapes plausible for the class (defined by $\bar{x}, E_{1:n}, \lambda_i$ from subsection 3.5).

At each iteration of TPS-RPM we project the current shape estimate Y (equation (4)) inside the valid region, just before fitting the TPS. This amounts to:
1) align Y on $\bar{x}$ w.r.t. to translation/rotation/scale
2) project Y on the subspace spanned by $E_{1:n}$:
$b = E^{-1}(Y - \bar{x})$, $b_{(n+1):2p} = 0$
3) bound the first n components of b by $\pm 3\sqrt{\lambda_i}$
4) transform b back into the original space: $Y^c = \bar{x} + Eb$
5) apply to Y^c the inverse of the transformation used in 1)

The assignment $Y \leftarrow Y^c$ imposes *hard constraints* on the shape space. While this guarantees output shapes similar to class members, it might sometimes be *too* restrictive. To match a novel instance accurately, it could be necessary to move a little along some dimensions of the shape space not recorded in the deformation model. The training data cannot be assumed to present all possible intra-class variations. Hence, we also propose a *soft constrained* variant, where Y is *attracted* by the valid region, with a force that diminishes with the temperature: $Y \leftarrow Y + \frac{T}{T_{init}}(Y^c - Y)$. This causes TPS-RPM to start fully constrained, and then, as temperature decreases and M looks for correspondences closer to the current estimates, later iterations are allowed to apply small deformations beyond the valid region (typically along dimensions not in $E_{1:n}$). As a result, output shapes fit the image data more accurately, while still resembling class members. Notice how this behavior is fully in the spirit of TPS-RPM, which also lets the TPS more and more free as T decreases.

The proposed extension to TPS-RPM reaches deep into its heart as it *alters the search* through the transformation and correspondence spaces. Beside improving accuracy, it can take TPS-RPM out of a local minima far from the correct solution, thus avoiding a gross failure.

4.4. Detections

Every local maximum in Hough space leads to a shape matched to the test image, a *detection*. Each detection is scored based on four terms: 1) average proximity of the TPS-mapped points to their corresponding image points; 2) deviation of the affine component of the TPS from the identity; 3) amount of non-rigid warp of the TPS; 4) number of model points matched with good confidence (max m_{ai} values). If two detections overlap substantially, we remove the lower scored one. Notice that the method can detect multiple instances of the same class in an image.

5. Results and conclusions

We present an extensive evaluation over six diverse object classes from two existing datasets. The first one is the *ETHZ Shape Classes* [7], which contains a total of 255 images divided among apple logos (40), bottles (48), giraffes (87), mugs (48), and swans (32). It's highly challenging as objects appear in a wide range of scales, they exhibit considerable intra-class variation, and many images are extensively cluttered. The second dataset is the *INRIA Horses* [11], consisting of 170 images containing horses, and 170 without horses. Horses appear at different scales, and against cluttered backgrounds.

Models from real images. Experiments are conducted in 5-fold cross-validation. For each class of the ETHZ dataset we learn 5 different models by sampling 5 subsets of half of the class images at random. The test set for a model then consists of *all* other images in the dataset. This includes about 200 negative images, hence supporting accurate estimation of the false-positive rate. For the INRIA dataset, we sample 5 subsets of 50 horse images for training, and use all other images for testing (*i.e*. 120 positive and 170 negative).

We report object detection performance as the detection-rate at the moderate rate of 0.4 false-positives per image (FPPI), averaged over the 5 trials (table 1, second row). In order to compare to [7], we adopt their criterion: a detection is counted as correct if its bounding-box overlaps more than 20% with the ground-truth one, and vice-versa. As the table shows, our method performs well [2] on all classes but

[2] Using the somewhat stricter PASCAL Challenge criterion (bounding-box intersection over union $>$ 50%) lowers detection rates by only 0%/1.6%/3.6%/4.9% for apple logos/bottles/mugs/swans. This indicates that bounding-boxes are accurate. For horses and giraffes the decrease is more significant (18.1%, 14.1%), because the legs of the animal are harder to detect.

	Applelogos	Bottles	Giraffes	Mugs	Swans	Horses
Hough alone:	35.9 (7.5)	71.1 (4.6)	56.8 (9.7)	51.4 (4.8)	63.3 (8.1)	85.8 (1.6)
full system:	83.2 (1.7)	83.2 (7.5)	58.6 (14.6)	83.6 (8.6)	75.4 (13.4)	84.8 (2.6)
accuracy:	1.5 (0.2)	2.4 (0.3)	3.5 (0.6)	3.1 (0.7)	3.0 (0.2)	5.4 (0.6)
Ferrari et al. [7]:	72.7 / 56.8	90.9 / 89.1	68.1 / 62.6	81.8 / 68.2	93.9 / 75.8	-
our system:	86.4 / 84.1	92.7 / 90.9	70.3 / 65.9	83.4 / 80.3	93.9 / 90.9	-
accuracy:	2.2	2.9	3.9	4.0	3.2	-

Table 1. *Experimental results. The top 3 rows cover the experiments based on models learnt from real images. Entries of the first two rows report the average detection-rate at 0.4 FPPI, and its standard-deviation (in brackets). The 'accuracy' row refers to the accuracy of the produced shapes averaged over all detections at 0.4 FPPI (lower values are better, see main text). The bottom 3 rows cover experiments based on hand-drawings. Rows 4 and 5 show detection-rates at 0.4 and 0.3 FPPI (before and after '/' respectively). The accuracy of the shapes matched by our system is given in the last row.*

giraffes, mainly due to the difficulty of building shape models from their extremely noisy edge maps. For comparison, the first row of table 1 reports the detection performance obtained by the Hough voting stage alone (subsection 4.1), without the shape matcher on top. The full system performs considerably better, showing the benefit of treating object detection fully as a shape matching task, rather than simply matching local features. Moreover, the shape matching stage also makes it possible to localize complete object boundaries (figure 4). Notice that the standard-deviations in table 1 reflect variations in the randomized training *and* test sets for different models. When the test set is fixed for all models to include all images in the dataset, the standard deviations diminish by about half, reaching low values. This suggests the proposed shape learning technique is stable with respect to the choice of training images.

In addition to the above evaluation we also measure how accurately the output shapes delineate the true object boundaries (ground-truth annotations). For this we use the symmetric Chamfer distance, normalized by the ground-truth diagonal, and averaged over correct detections and trials. The system brings a convincing performance also in this respect, with low errors around 3% (see third and sixth rows of the table and the examples in figure 4).

Models from hand-drawings. Our system can directly input one hand-drawing as model of the class shape, instead of learning it from images. In this case, the modeling stage simply decomposes the model into PAS. Object detection then uses these PAS for the Hough voting step, and the hand-drawing itself for the shape matching step. This allows a comparison to [7] using their exact setup, with a single hand-drawing per class and *all* 255 images of the ETHZ Shape Classes as test set. Our method performs better than [7] on all 5 classes (fifth row of the table). Moreover, our approach offers three additional advantages over [7]: it can train from real images, it supports branching and self-intersecting models, and it is more robust as it does not need the test image to contain long chains of contour segments around the object.

Interestingly, hand-drawings lead to better results than when learning models from images. This can be explained by the fact that hand-drawings are essentially the prototype shapes the system tries to learn.

Conclusions. The experiments confirm that the presented approach can learn class-specific shape models from images annotated with bounding-boxes, and then localize the boundaries of novel class instances in the presence of extensive clutter, scale changes, and intra-class variability. In addition, it is also very effective when given hand-drawings as models. By supporting both images and hand-drawings as training data, our approach bridges the gap between shape matching and object detection.

References

[1] A. Berg, T. Berg and J. Malik, *Shape Matching and Object Recognition using Low Distortion Correspondence*, CVPR, 2005.

[2] S. Belongie and J. Malik, *Shape Matching and Object Recognition using Shape Contexts*, PAMI, 2002.

[3] H. Chui and A. Rangarajan, *A new point matching algorithm for non-rigid registration*, CVIU, 2003.

[4] T. Cootes, *An introduction to Active Shape Models*, 2000.

[5] D. Cremers, C. Schnorr, and J. Weickert, *Diffusion-Snakes: Combining Statistical Shape Knowledge and Image Information in a Variational Framework*, Workshop on Variational and Levelset Methods, 2001.

[6] G. Elidan, G. Heitz, D. Koller, *Learning Object Shape: From Drawings to Images*, CVPR, 2006.

[7] V. Ferrari, T. Tuytelaars, and L. Van Gool, *Object Detection with Contour Segment Networks*, ECCV, 2006.

[8] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, *Groups of Adjacent Contour Segments for Object Detection*, PAMI, 2007 (to appear).

[9] P. Felzenswalb, *Representation and Detection of Deformable Shapes*, PAMI, 2004.

[10] A. Hill and C. Taylor, *A Method of Non-Rigid Correspondence for Automatic Landmark Identification*, BMVC, 1996.

[11] F. Jurie and C. Schmid *Scale-invariant Shape Features for Recognition of Object Categories*, CVPR, 2004.

[12] P. Kumar, P. Torr, and A. Zisserman, *OBJ CUT*, CVPR, 2005.

[13] B. Leibe and B. Schiele, *Scale-Invariant Object Categorization using a Scale-Adaptive Mean-Shift Search*, DAGM, 2004.

[14] D. Martin, C. Fowlkes and J. Malik, *Learning to detect natural image boundaries using local brightness, color, and texture cues*, PAMI, 2004.

[15] A. Opelt, A. Pinz, and A. Zisserman, *A Boundary-Fragment-Model for Object Detection*, ECCV, 2006.

[16] J. Shotton, A. Blake, and R. Cipolla, *Contour-Based Learning for Object Detection*, ICCV, 2005.

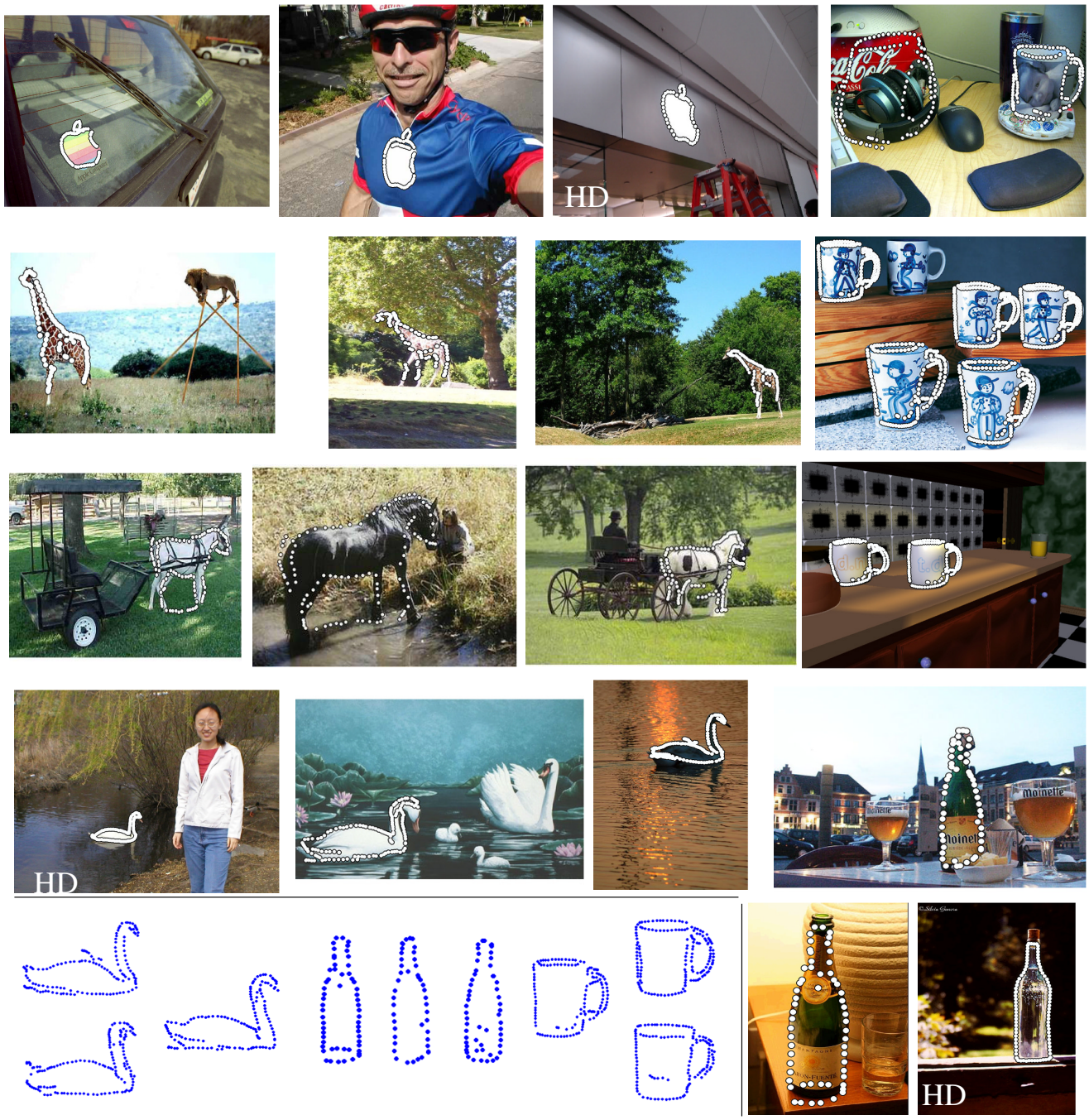

Figure 4. *Example detections. Results are based on models automatically learnt from images, expect those marked with 'HD', which use a hand-drawing from [7] as model. The rightmost case of the top row includes a typical false positive, and the image below it illustrates multiple detected instances (one missed). In the first two images of the third row, the horizontal line below the horses' legs is part of the model and represents the ground. Interestingly, the ground line systematically reoccurs over the training images for that model and gets learned along with the horse. Bottom row: models produced by our approach from different random image subsets. They represent prototypical shapes for their respective class. Similar shapes are produced from different training images, indicating the stability of our learning technique.*

[17] J. Shotton, J. Winn, C. Rother, and A. Criminisi, *TextonBoost: Joint Appearance, Shape and Context Modeling for Multi-Class Object Recognition and Segmentation*, ECCV, 2006.

[18] S. Todorovic and N. Ahuja, *Extracting Subimages of an Unknown Category from a Set of Images*, CVPR, 2006.

[19] A. Torralba, K. Murphy, and W. Freeman, *Sharing Features: efficient boosting procedures for multiclass object detection*, CVPR, 2004.

[20] J. Winn and N. Jojic, *LOCUS: Learning Object Classes with Unsupervised Segmentation*, ICCV, 2005.