Many-to-one Contour Matching for Describing and Discriminating Object Shape

Praveen Srinivasan, Qihui Zhu
GRASP Laboratory, University of Pennsylvania
3330 Walnut St., Philadelphia, PA - 19104
{psrin,qihuizhu}@seas.upenn.edu

Jianbo Shi
GRASP Laboratory, University of Pennsylvania
3330 Walnut St., Philadelphia, PA - 19104
jshi@cis.upenn.edu

Abstract

We present an object recognition system that locates an object, identifies its parts, and segments out its contours. A key distinction of our approach is that we use long, salient, bottom-up image contours to learn object shape, and to achieve object detection with the learned shape. Most learning methods rely on one-to-one matching of contours to a model. However, bottom-up image contours often fragment unpredictably. We resolve this difficulty by using many-to-one matching of image contours to a model.

To learn a descriptive object shape model, we combine bottom-up contours from a few representative images. The goal is to allow most of the contours in the training images to be many-to-one matched to the model. For detection, our challenges are inferring the object contours and part locations, in addition to object location. Because the locations of object parts and matches of contours are not annotated, they appear as latent variables during training. We use the latent SVM learning formulation to discriminatively tune the many-to-one matching score using the max-margin criterion. We evaluate on the challenging ETHZ shape categories dataset and outperform all existing methods.

1. Introduction and Related Work

Our approach is summarized in Fig. 1. Given positive and negative images for an object category, and object bounding boxes, we use a bottom-up grouping method ([18]) to obtain long, salient image contours. From these contours, we first learn a descriptive shape model from the positive images. We then discriminatively tune the model using additional negative images for good detection performance. Both the learning steps use many-to-one matching of image contours to a shape model as a key process.

Our contributions are: 1) the use of bottom-up image contours provides a valuable mid-level representation for recognition that can ultimately help bridge the gap between high-level semantic constructs and image features, 2) learning of object shape using image contours and 3) discriminative training of a contour-based object detector using the learned object shape for superior detection performance.

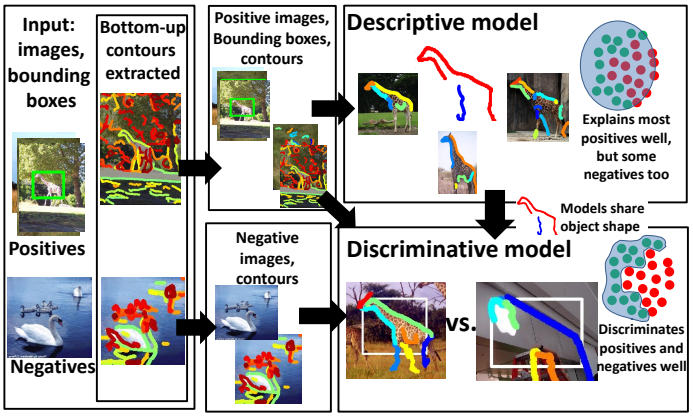

Figure 1. Overview of our proposed method. See text for details.

For many previous works, 2) and 3) are disjoint and unrelated. However, in our work they are tied together by the common underlying contour shape model.

There are three different areas related to our approach: image contour-based object detection, object shape learning, and discriminative object detection.

Image contour-based detection methods: There are several works that use bottom-up image contours for object detection; [5] is most related. They used voting of configurations of bottom-up contours followed by a refinement scheme to discover shape from positive examples. In [11], the authors extracted contours and used a particle filter method for recognizing and grouping contours given a hand-drawn model. However, both methods expect image contours to have consistent fragmentation across images, which is not always satisfied. The method of [19] used many-to-one (and also many-to-many) matching of image contours to a shape model for a detection, but required a hand-drawn model and hand-tuned detection parameters.

Object shape learning: In [3], the authors learned the outline shape of an object category from contours (found using a snake model), but were limited to learning from clutter-free "cartoon-images" and did not learn interior contours. There has been substantial other work in learning object shape, such as [9], [17], [10], and [15] but they do not leverage bottom-up image segmentations, leaving them at

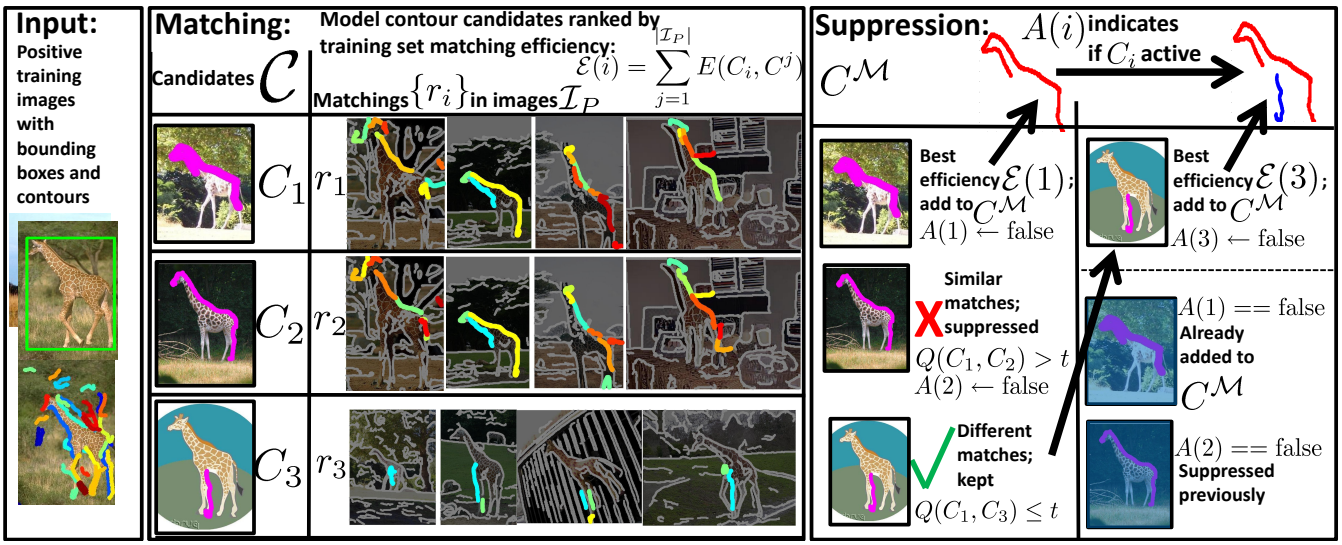

Figure 2. Model shape learning process, simplified example. Left, input consists of positive images w/ bounding boxes and contours (see Fig. 1). Middle: image contours are matched ($\{r_i\}$) against model candidates $\{C_i\}$ to compute training set matching efficiency $\mathcal{E}$. Right: model $C^{\mathcal{M}}$ constructed greedily using efficiency scores $\mathcal{E}$ (see Algorithm 1). Top candidate (according to $\mathcal{E}$) added to model and similar ones (Q) suppressed (A is active list; $A(i) \leftarrow$ false); process repeats. Contours shown in order of efficiency score for simplicity.

risk for accidental alignments in clutter. In addition to detection, [5] also learned object shape using a combination of discovery of recurring configurations and refinement steps; however this again expects that configurations of bottom-up contours recur consistently.

Discriminative object detection: Topic-model approaches, such as [6], have been used to discover a low-dimensional representation for image regions as a set of responses to the topic model. This representation provided features for an SVM classifier. Local feature voting for hypothesis proposal was combined with intersection kernel SVM verification by [12]. However, these patch-based methods do not leverage the non-accidental nature of bottom-up segmentation. Boosting has been used to learn an object model shape made of contours extracted from training images, as in [13]. However, they used a chamfer matching cost for scoring placements of object parts in the image which often matches clutter. Some work ([14]) used a hand-drawn model of the object of interest, broke the model contours into fragments, and searched the image for fragment matches. Matches voted to yield object detections, which were followed by refinement and verification steps. In contrast, our method learns an object shape model automatically.

The histogram of oriented gradients (HOG, [2]) image feature was used in a discriminatively trained deformable part object model where part placements were scored by correlation of a learned part filter with the HOG features ([4] used). However, the support of a HOG feature in the image may overlap between the object and background (also noted in [9]), which can lead to unpredictable changes in the features.

Our paper first discusses the model shape learning, also explaining the use of many-to-one matching, and then explains the discriminative learning.

2. Learning a Descriptive Shape Model

We assume as input a set of positive images $\mathcal{I}_P$ containing the object $\mathcal{I}_P = \{I_1, ..., I_{|\mathcal{I}_P|}\}$.

Each image has associated with it one or more bounding boxes that indicate the rough location and size of the instances of the object in the image. Also, for each image I_j we use a bottom-up contour grouping process (we use the method of [18]) to obtain image contours:

$$C^j = \{C_1^j, ..., C_{|C^j|}^j\} \quad (1)$$

We restrict the contours to those that lie inside a bounding box. We define $\mathcal{C}$ as the union of these sets of contours for all the images. Instead of learning the object shape from scratch ([9]), we use $\mathcal{C}$ as a set of natural candidates for object model. The key insight is that in some of the images a significant portion of the object is clearly visible, resulting in a long, salient object contour. By combining several of these "lucky" contours, we can obtain a good shape model. We write the model contours as $C^{\mathcal{M}} = \{C_1^{\mathcal{M}}, ..., C_{|C^{\mathcal{M}}|}^{\mathcal{M}}\}$, and have two criteria for selecting these contours:

- Matching efficiency: each model contour in $C^{\mathcal{M}}$ should be able to match efficiently (defined below) to object contours C^j in a training image I^j
- Non-overlapping: each model contour in $C^{\mathcal{M}}$ should be matched to mostly different contours $\mathcal{C} = \{C^1 \cup C^2 \cup ... \cup C^{|\mathcal{I}_P|}\}$ in the training images

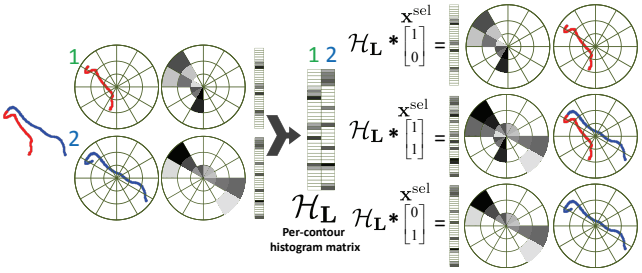

Figure 3. Encoding of contours for selection. Left, two contours, are shown with associated histograms (any histogram, grid or shape context, is possible), which are combined to form matrix $\mathcal{H}_{\mathbf{L}}$ (center). Right, different choices of selection vector $\mathbf{x}^{\text{sel}}$ lead to different histograms; $h_{\mathbf{L},\mathbf{x}^{\text{sel}}}$ is thus a linear function of $\mathbf{x}^{\text{sel}}$.

We explain score functions for both separately and then integrate them into a single score.

Matching efficiency score: The matching efficiency score evaluates the ability of a single model contour $C_i^{\mathcal{M}}$ to match well to many fragmented contours C^j in an image I_j: $E(C_i^{\mathcal{M}}, C^j)$. The key difficulty is that there is no one-to-one correspondence between image contours and the model. We define a many-to-one matching as:

- $\mathbf{L} \in \mathbb{R}^2$: a placement that describes the alignment of the image contours to the model contour
- $\mathbf{x}^{\text{sel}} \in \{0,1\}^{|\text{C}^\text{j}|}$: an indicator vector that defines which image contours are **selected** for matching to the model

The bounding boxes provided with the positive images allow us to infer rough alignment of the training images. We can use the bounding box center as $\mathbf{L}$ and additionally warp the contours according to the bounding box dimensions to roughly align the contours across all the training examples. Given the alignment from $\mathbf{L}$, we can define histogram shape features as a function of $\mathbf{x}^{\text{sel}}$. For a particular $\mathbf{x}^{\text{sel}}$, we can histogram the points in the selected contours to obtain a shape descriptor. The histogram counts the number of contour points that falls into each bin. Any binning pattern is possible, including grid histograms (which we use during shape learning) as well as log-polar histograms (like the shape context, which we use during discriminative learning/detection). Fig. 3 shows an example of histograms over the points in contours, and the flattening of these histograms into vectors.

We define a grid histogram of dimension d_m centered at $\mathbf{L}$ over the edge points of contours selected by $\mathbf{x}^{\text{sel}}$ as: $h_{\mathbf{L},\mathbf{x}^{\text{sel}}}$. An important property of histograms is that a histogram over the points of several contours is equivalent to summing the histograms computed for each contour individually, first noted in [19], and also depicted in Fig. 3. This means that histogram $h_{\mathbf{L},\mathbf{x}^{\text{sel}}}$ can be represented as a linear function of $\mathbf{x}^{\text{sel}}$ as shown in Fig. 3. We introduce the per-contour histogram matrix $\mathcal{H}_{\mathbf{L}}$ (Fig. 3) and write the histogram over selected-contours $h_{\mathbf{L},\mathbf{x}^{\text{sel}}}$ as a linear function of $\mathbf{x}^{\text{sel}}$:

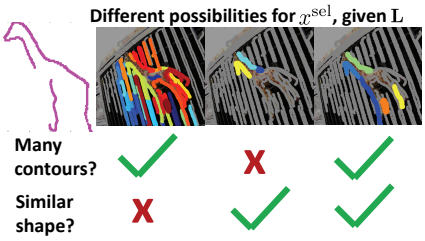

Figure 4. Examples of selection. For a given model shape (upper left; Giraffe) and placement $\mathbf{L}$, different selections $\mathbf{x}^{\text{sel}}$ of image contours are shown. Indicator $\mathbf{x}^{\text{sel}}$ enocdes which image contours are many-to-one matched. Matching score prefers selections and placement that select many contours which have similar shape to the model.

$$\mathcal{H}_{\mathbf{L}} \in \mathbb{R}^{d_m \times |\text{C}^\text{j}|} \quad h_{\mathbf{L},\mathbf{x}^{\text{sel}}} \iff \mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}} \tag{2}$$

The k-th column of $\mathcal{H}_{\mathbf{L}}$ is a histogram over the points in contour C_k^j. We can define a histogram over a single model contour $C_i^{\mathcal{M}}$ with a slight abuse of notation: $h_{C_i^{\mathcal{M}}}$. We define shape comparison features $K(\mathbf{L}, \mathbf{x}^{\text{sel}})$ between $h_{C_i^{\mathcal{M}}}$ and $\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}}$ that are a function of the placement and selection. We measure two types of features on the histograms: difference features $-|\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}} - h_{C_i^{\mathcal{M}}}|$ and intersection features $\min(\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}}, h_{C_i^{\mathcal{M}}})$:

$$K(\mathbf{L}, \mathbf{x}^{\text{sel}}) = \begin{bmatrix} -|\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}} - h_{C_i^{\mathcal{M}}}| \\ \min(\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}}, h_{C_i^{\mathcal{M}}}) \end{bmatrix} \tag{3}$$

Many-to-one matching optimization: Given a set of pre-defined weights on these features, $w^{\text{app}} \geq 0$, we can define the many-to-one matching score (larger is better) as: $w^{\text{app}\mathsf{T}} K(\mathbf{L}, \mathbf{x}^{\text{sel}})$. Fig. 4 summarizes the selection criteria. This score function has $2^{|C^j|}$ possible instantiations and is dramatically more complex than a score based solely on the placement $\mathbf{L}$. However, with this added complexity we can use a much larger-scale shape descriptor for cluttered images. We can now write the **matching efficiency** score E in terms of K, normalized by model contour length $l(C_i^M)$ (the length of a contour refers to its physical length, not simply the number of points in it):

$$E(C_i^{\mathcal{M}}, C^j) = \frac{1}{l(C_i^{\mathcal{M}})} \max_{\mathbf{L}\in\mathbb{R}^2, \mathbf{x}^{\text{sel}}\in\{0,1\}^{|\text{C}^\text{j}|}} w^{\text{app}\mathsf{T}} K(\mathbf{L}, \mathbf{x}^{\text{sel}})$$

An important question is how to perform the above maximization over $\mathbf{L}$ and $\mathbf{x}^{\text{sel}}$ for a given $C_i^{\mathcal{M}}$ and image contours C^j. For fixed $\mathbf{L}$ (provided by the center of the object bounding box in positive images), the resulting optimization problem is an integer linear program. Alternatively, we can relax $\mathbf{x}^{\text{sel}} \in [0,1]^{|\text{C}^\text{j}|}$, resulting in a linear program that can be solved efficiently via a linear program solver ([1]). With restriction of $w^{\text{app}} \geq 0$, this score function is concave and maximization is possible. We can write a linear

program that maximizes a relaxation of our score function $w^{\text{app}\mathsf{T}} K(\mathbf{L}, \mathbf{x}^{\text{sel}})$ using proxy variables m and o to represent the histogram difference $-|\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}} - h_{C_i^{\mathcal{M}}}|$ and intersection features $\min(\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}}, h_{C_i^{\mathcal{M}}})$ respectively:

$$\begin{aligned} \max_{\mathbf{x}^{\text{sel}} \in [0,1]^{|C^j|}} \quad & w^{\text{app}\mathsf{T}} \begin{bmatrix} m \\ o \end{bmatrix} \\ \text{s.t.} \quad & m \leq (h_{C_i^{\mathcal{M}}} - \mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}}), (\mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}} - h_{C_i^{\mathcal{M}}}) \\ & o \leq \mathcal{H}_{\mathbf{L}}\mathbf{x}^{\text{sel}}, h_{C_i^{\mathcal{M}}} \end{aligned}$$

We summarize the outputs of the matching process: training set matching efficiency $\mathcal{E}$ for each candidate in $\mathcal{C}$ (for $C_i \in \mathcal{C}$, sum of efficiencies over all training images; $\mathcal{E}(i) = \sum_{j=1}^{|\mathcal{I}_P|} E(C_i, C^j)$; see Fig. 2) and matching indicator vectors $\{r_1, ..., r_{|\mathcal{C}|}\}$, where each $r_i \in \{0,1\}^{|\mathcal{C}|}$ is an indicator vector. This vector tells us which contours in the images candidate $C_i \in \mathcal{C}$ was matched to.

In practice, we use the linear program to approximate $\mathcal{E}(i)$ for all candidates and prune those with low approximate score. We solve an integer program for the remaining candidates to get $\mathcal{E}(i)$ exactly, and discard those with $\mathcal{E}(i) < 0$ as they are unlikely to be good model contours.

Overlapping candidate suppression: Two different candidate contours in $\mathcal{C}$, from the same or different images, may both correspond to the same part of the object. Additionally, the contour grouping method we use ([18]) produces overlapping contours within a single image, unlike most bottom-up grouping methods. So, we must be careful not to include overlapping contours in our model. We define for any two candidates $C_i, C_j \in \mathcal{C}$, $Q(C_i, C_j)$ as a measure of overlap in terms of the which contours matched to the candidates:

$$Q(C_i, C_j) = \frac{\sum_{k=1}^{|\mathcal{C}|} r_i(k) r_j(k) l(C_k)}{\sum_{k=1}^{|\mathcal{C}|} r_j(k) l(C_k)} \tag{4}$$

The numerator computes the sum of lengths of contours that both C_i, C_j matched, while the denominator computes the sum of lengths of contours that C_j matched. If Q is large, then C_i matched most of the contours (according to total length) that C_j matched. Q lies in the interval $[0, 1]$. If C_j matched no contours, Q is defined to be 1 for all C_i.

Model shape score function: Given score functions for matching efficiency and overlap, we can now define a joint score function that optimizes the total matching efficiency of the model contours subject to an overlap constraint $Q(C_i^{\mathcal{M}}, C_j^{\mathcal{M}}) \leq t$, where t is a pre-defined threshold on the contours:

$$\begin{aligned} \max_{C^{\mathcal{M}} | C^{\mathcal{M}} \subseteq \mathcal{C}} \quad & \sum_{i=1}^{|C^{\mathcal{M}}|} \sum_{j=1}^{|\mathcal{I}_P|} E(C_i^{\mathcal{M}}, C^j) \\ \text{s.t.} \quad & \forall k, l \;\; Q(C_k^{\mathcal{M}}, C_l^{\mathcal{M}}) \leq t \end{aligned} \tag{5}$$

Optimization over $C^{\mathcal{M}}$: Because the space of $C^{\mathcal{M}}$ includes all possible subsets of $\mathcal{C}$, there are an exponential number of states. We use a greedy approach which incrementally adds a contour candidate to the model according to training set matching efficiency and avoiding overlap, summarized in Algorithm 1. $C^{\mathcal{M}}$ is initialized to be empty ($\emptyset$), and all candidates are initially active ($A(i)$ is true). We add to $C^{\mathcal{M}}$ the active contour candidate with highest efficiency, and de-activate all conflicting candidates ($A(i) \leftarrow$ false, according to Q), repeating until no candidates remain. This process is guaranteed to improve the score in Eq. 5 at each step while obeying Q, since we only keep candidates with $\mathcal{E}(i) > 0$. Fig. 2 depicts the learning process, and Fig. 5 shows results on the ETHZ shape categories dataset.

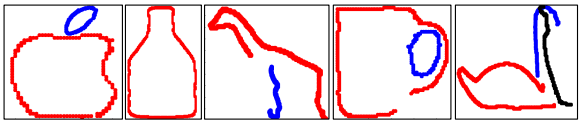

Figure 5. Models learned for Applelogos, Bottles, Giraffes, Mugs and Swans from the ETHZ shape categories dataset using half the images for each category. Contour colors indicate different candidates combined to form the model.

Algorithm 1: Model shape learning: optimizing Eq. 5

Input *candidates $\mathcal{C}$, match eff. scores $\mathcal{E}$, matches R*;
Output *Model shape $C^{\mathcal{M}}$; local maximum for Eq. 5*;
Initialization $C^{\mathcal{M}} \leftarrow \emptyset$, $A \leftarrow \{1\}^{|\mathcal{C}|}$ *(indicates whether candidate still active)*;
while *A has a true entry* **do**
  $C_i \leftarrow$ active candid. w/ best match eff. score $\mathcal{E}(i)$;
  $C^{\mathcal{M}} \leftarrow C^{\mathcal{M}} \bigcup \{C_i\}$, $A(i) \leftarrow$ false ;
  $\forall j$ s.t. $Q(C_i, C_j) > t$, $A(j) \leftarrow$ false ;
end

3. Discriminative Detector Learning

3.1. Object Detection as Many-to-one Matching

For object detection, we still perform many-to-one matching of image contours to a shape model (learned above), but with a matching score tuned for discrimination. Previously we assumed just one model part (single contour selection + part placement) because the bounding boxes provided the placement/warping. For detection, we extend to have $N + 1$ parts to better accomodate object deformation:

$$\text{Parts}: \; P_0, P_1, ..., P_N$$

Parts may deform relative to the root part, P_0, that represents the center of the object. In the model, parts $P_1, .., P_N$ are located at points of high curvature on the model shape contours: $\mathbf{L}'_1, ..., \mathbf{L}'_N$ (Fig. 6, panel "Model initialization"). We use the discrete curve evolution method of [8] (a contour simplification technique) to find these points $\mathbf{L}'_i$ from the model shape contours, and $\mathbf{L}'_0$ on the model is computed as simply the mean of $\mathbf{L}'_1, ..., \mathbf{L}'_N$ in the model. Part appearances for parts P_i, $i = 1, ..., N$ are represented with

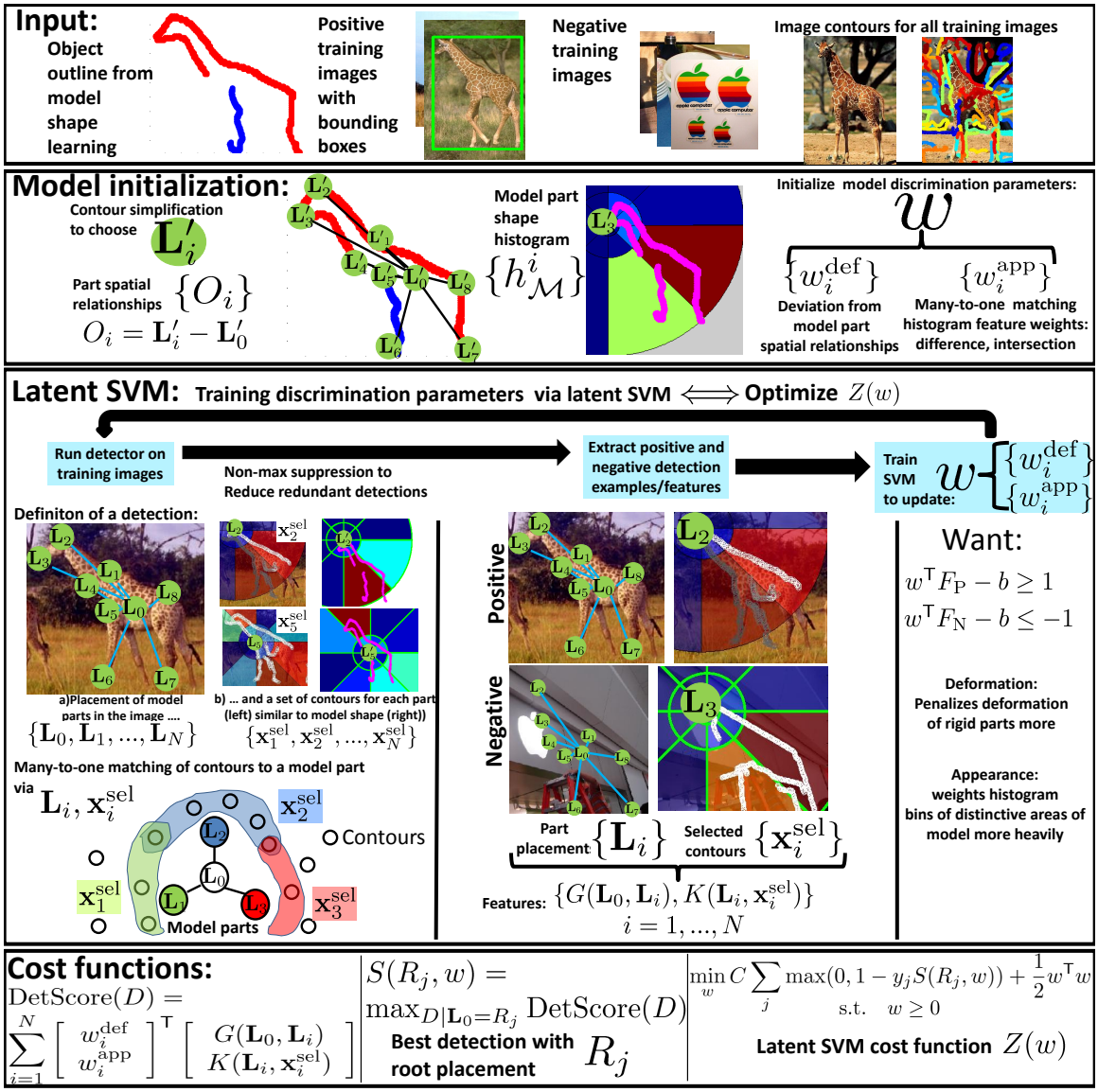

Figure 6. Overview of discriminative tuning of many-to-one matching score function."Input" panel: input to this step is the shape model automatically discovered previously, along with positive and negative training images and their image contours. "Model initialization": relative spatial relationships of parts w.r.t. to root part are computed along with shape histograms to represent part appearances. The many-to-one matching paramters (to be learned) are initialized. "Latent SVM": latent SVM training iterates between a) running the detector on images (left)/extracting positive, negative detection examples and features F_P and F_N, resp. (middle) and b) updating model parameters w, b (right). A detection (left) involves placements of the parts $\mathbf{L}_i$ and many-to-one matchings of contours to the parts $\mathbf{x}_i^{\text{sel}}$.

model part histograms $h^i_{\mathcal{M}}$ centered at $\mathbf{L}'_i$ computed over the model shape (P_0 has no appearance term, although our formulation can accomodate one); we use shape context histograms (Fig. 6, "Model initialization").

For an image I_j with contours C^j, a detection of consists of a many-to-one matching for each part: placements $\mathbf{L}_i$ for each part, and selected contours for matching to each part $P_i, i = 1, ..., N$, $\mathbf{x}_i^{\text{sel}}$ (with the exception of the root part, which only serves to spatially relate the other parts):

$$\begin{aligned} \mathbf{L}_i &: P_i \rightarrow \mathbb{R}^2 \\ \mathbf{x}_i^{\text{sel}} &: C^j \rightarrow \{0,1\}^{|C^j|} \end{aligned} \tag{6}$$

We define a detection as $D = \{\mathbf{L}_0, \mathbf{L}_1, \mathbf{x}_1^{\text{sel}}, \mathbf{L}_2, \mathbf{x}_2^{\text{sel}} ..., \mathbf{L}_N, \mathbf{x}_N^{\text{sel}}\}$. Fig. 6, "Latent SVM", left, also describes a detection.

Placement score: For each part P_i, we need to be able to score a placement $\mathbf{L}_i$ of the part in the image along with matching contours $\mathbf{x}_i^{\text{sel}}$. We use the same many-to-one shape matching features K, with a part-specific weight vector $w_i^{\text{app}} \geq 0$: $w_i^{\text{app}\,\mathsf{T}} K(\mathbf{L}_i, \mathbf{x}_i^{\text{sel}})$. The corresponding model part shape histogram is $h^i_{\mathcal{M}}$ (Fig. 6, "Model initialization", center).

Deformation score: For each part $i = 1, ..., N$ we use

a part offset $O_i = (O_i^x, O_i^y)$ that describes the expected spatial position of P_i, $\mathbf{L}_i$, relative to $\mathbf{L}_0$: $\mathbf{L}_0 + O_i$ (Fig. 6, "Model initialization"). O_i is computed as the difference between the locations of parts P_i and P_0 in the model: $\mathbf{L}'_i - \mathbf{L}'_0$. The deviation of a part P_i from its expected position relative to the root provides part deformation features G:

$$G(\mathbf{L}_0, \mathbf{L}_i) = \begin{bmatrix} -(\mathbf{L}_i^x - (\mathbf{L}_0^x + O_i^x))^2 \\ -(\mathbf{L}_i^y - (\mathbf{L}_0^y + O_i^y))^2 \end{bmatrix} \quad (7)$$

A set of parameters w_i^{def}, $i = 1, ..., N$ (to be learned, along with w_i^{app}) penalizes deviation of part P_i from its expected position relative to P_0. The overall score function for a particular detection D is:

$$\text{DetScore}(D) = \sum_{i=1}^{N} \begin{bmatrix} w_i^{\text{def}} \\ w_i^{\text{app}} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} G(\mathbf{L}_0, \mathbf{L}_i) \\ K(\mathbf{L}_i, \mathbf{x}_i^{\text{sel}}) \end{bmatrix} \quad (8)$$

In contrast to [4], our appearance term does not depend simply on the placement $\mathbf{L}_i$ of part P_i, but also on the contours chosen for matching, $\mathbf{x}_i^{\text{sel}}$.

Inference for detection: The space of possible detections is exponential in the number of possible placements for each part and number of image contours. To cope, we create a regular grid of possible root part locations $\mathcal{R}$ in the image and only keep the highest scoring detection per root part location $R_j \in \mathcal{R}$, as in [4]. For each possible root location R_j, we fix $\mathbf{L}_0 = R_j$, and then maximize the detection score subject to this root constraint to obtain score $S(R_j, w)$:

$$S(R_j, w) = \max_{D|\mathbf{L}_0 = R_j} \text{DetScore}(D) \Longrightarrow$$
$$\max_{\{L_1,...,L_N, \mathbf{x}_1^{\text{sel}},...,\mathbf{x}_N^{\text{sel}}\}} \sum_{i=1}^{N} \begin{bmatrix} w_i^{\text{def}} \\ w_i^{\text{app}} \end{bmatrix}^{\mathsf{T}} \begin{bmatrix} G(R_j, \mathbf{L}_i) \\ K(\mathbf{L}_i, \mathbf{x}_i^{\text{sel}}) \end{bmatrix} \quad (9)$$

We note that $w_i^{\text{app}\mathsf{T}} K(\mathbf{L}_i, \mathbf{x}_i^{\text{sel}})$ does not depend on $\mathbf{L}_0$, and hence for each part P_i the maximization over $\mathbf{x}_i^{\text{sel}}$ can be pre-computed for each possible placement $\mathbf{L}_i$. In [4], this step corresponds to convolving the image with the filter associated with a part. In our case, we use the previously described linear programming relaxation to efficiently and accurately approximate the many-to-one matching score.

Given $w_i^{\text{app}\mathsf{T}} K(\mathbf{L}_i, \mathbf{x}_i^{\text{sel}})$ for each possible placement $\mathbf{L}_i$ of each part P_i, $\max_{D|\mathbf{L}_0 = R_j} \text{DetScore}(D)$ can be computed easily by picking the best part placement for each part P_i individually. For fixed $\mathbf{L}_0$, the scores for parts $P_1, ..., P_N$ are independent. The set of possible placements are sampled from points of high curvature along image contours, following the method of [8]. We take as a bounding box the bounding box of the part locations. A detection with center R_j can be labeled as a true or false positive (label $y_j = \pm 1$) according the overlap of its bounding box with a ground truth bounding box. Non-maximum suppression allows us to eliminate many redundant/overlapping detections, reducing the complexity of learning.

3.2. Latent SVM For Discriminative Detector Learning

Given detections centered at $R_j \in \mathcal{R}$ with labels $y_j = \pm 1$ from the training images, we learn discriminative model parameters $w = [w_1^{\text{def}\mathsf{T}} w_1^{\text{app}\mathsf{T}} \cdots w_N^{\text{def}\mathsf{T}} w_N^{\text{app}\mathsf{T}}]^{\mathsf{T}}$ to optimize detection performance.

Loss function: Following the standard SVM formulation, we can write a hinge-loss function associated with the above score function as:

$$Z(w) = C \sum_j \max(0, 1 - y_j(S(R_j, w) - b)) + \frac{1}{2} w^{\mathsf{T}} w$$
$$\text{s.t.} \quad w \geq 0 \quad (10)$$

where $\max(0, 1 - y_j(S(R_j, w) - b))$ is the hinge loss and $\frac{1}{2} w^{\mathsf{T}} w$ is the regularization term. The learned constant b is the usual SVM bias term. We require $w \geq 0$ so that many-to-one matching score remains concave and maximizable.

Latent SVM training: We adapt the "coordinate descent" method from [4] for minimizing Eq. 10. Beginning with an initial set of parameters w_0, b_0, we run the detection procedure on the training images. The following two steps are iterated as shown in Fig. 6 ("Latent SVM"):

- *Update model parameters w, b given argmax detection for each root placement R_j:* Given the detection results from the previous model parameters, we train an optimal model w, b using the the features from the argmax of Eq. 9 (also see in Fig. 6) for each root R_j. This is equivalent to traditional SVM using these features, but with the added constraint $w \geq 0$.
- *For each root placement R_j, update part placements $\mathbf{L}_i$ and $\mathbf{x}_i^{\text{sel}}$ for parts $1, ..., N$:* Given new model parameters w, b, we recompute the optimal selections and placements $\mathbf{x}_i^{\text{sel}}, \mathbf{L}_i$, $i = 1, .., N$ for each root placement R_j by running the detection procedure.

We iterate the two update steps until the average precision of the detection precision-recall curve stops increasing, up to a maximum number of iterations. As in [4], we use a cache of hard negative examples that is grown on each iteration to ensure that the update of w, b does not cycle from one iteration to the next.

Refinement and joint selection: We perform an additional placement refinement step for high scoring detections. The voting procedure is repeated with an enlarged/denser set of part placements. The placements are then fixed, and many-to-one matching of contours to model parts is performed *jointly* (as in [19]), enforcing that all parts match the same contours, also a linear program. An affine transformation is estimated using shape context correspondences between the model and the jointly selected contours.

This allows for better detection of deformed objects. After an additional round of voting refinement and joint selection using the affine transformation for better model/image alignment, a classifier is trained on the features from this last step for the final result.

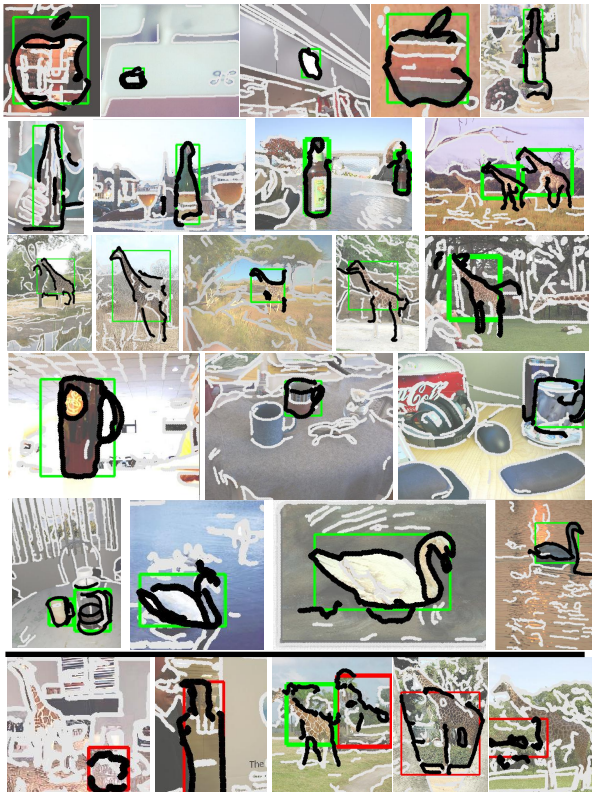

Figure 8. Some of our detection results on the ETHZ shape categories dataset. Each image shows segmented object contours and bounding boxes for one or more detections. Bottom row shows false positives for Applelogos, Bottles, Giraffes, Mugs and Swans (l-to-r); rest are true positives.

4. Experiments

We implemented our method in MATLAB, using additional software packages obtained from the authors of [18] and [1], and plan to release the code. We tested our method on the ETHZ shape categories dataset ([5]; freely available online), with five categories: Applelogos, Bottles, Giraffes, Mugs and Swans. We follow the train/test split described in [12]; for training for each category we used the first half of the images from that category as positive examples, and an equal number of negative images chosen equally from the remaining categories. Each category had 32 to 86 training images. Model shape learning was first performed, and a detector was trained using the latent SVM formulation. During shape learning, we used a 4-by-4 grid histogram with trilinear interpolation (orientation binning in addition to spatial binning; similar to [2]). Each image averaged 85 contours. Using the linear programming approximation to the matching efficiency score, these candidates were pruned to a pool of about 100 candidates, for which the exact efficiency score was computed using an integer linear program solver (also in GLPK; [1]). Histogram difference and intersection features were weighted uniformly as 1.2 and 1, resp. We used overlap threshold $t = 0.8$.

	Applelogos	Bottles	Giraffes	Mugs	Swans	**Mean**
Our method	0.845	0.916	**0.787**	**0.888**	**0.922**	**0.872**
Maji et al. [12]	0.869	0.724	0.742	0.806	0.716	0.771
Felz. et al. [4] code	**0.891**	**0.950**	0.608	0.721	0.391	0.712
Lu et al. [11]	0.844	0.641	0.617	0.643	0.798	0.709

Table 2. Comparison of **interpolated average precision (AP)** on the ETHZ shape categories dataset. Our method has the highest AP in 3 of the 5 categories, and the highest mean across categories.

During detection, images were searched at 6 different scales, 2 per octave. Each part had up to 200 different possible placements in the image; for each part/placement/scale tuple, a separate linear program was solved, taking a few minutes per image. Latent SVM parameters w were initialized uniformly as in model shape learning, and convergence took 3-7 iterations. After training the initial detector, learning was done for part placement refinement, affine transformation estimation and joint selection using high-scoring detections from voting (< 200 detections). All our results used 0.5 overlap score threshold for determining if a detection bounding box overlaps with a ground truth bounding box (PASCAL criterion). Each detector was tested on remaining 169 (Giraffe) to 223 (Swan) test images.

We compare our approach against the reported results from [12] and the method of [4] (using code from `http://people.cs.uchicago.edu/~pff/latent/`) with the same train/test split. We show the precision/recall (PR) curves (Fig. 7) as well as plots of false positives per image (FPPI) vs. detection rate (DR), and also the result of [11] (which tested on full dataset). Our method is comparable in Applelogos/Bottles and substantially outperforms on Mugs/Giraffes/Swans which have large deformation. Table 2 shows the interpolated average precision (AP; as used in the PASCAL VOC Challenge) for the methods. Our APs for the five categories are (0.845/0.916/0.787/0.888/0.922; mean: **0.872**), much better than the next best result at (mean: 0.771; [12]). Table 1 compares DR at 0.3/0.4 FPPI for several methods. Our detection rates at 0.3/0.4 FPPI of (0.95/0.95; 1/1; 0.872/0.896; 0.936/0.936; 1/1) and mean across categories of **0.952/0.956**, are a substantial improvement over the results of [12], 0.919/0.932 and [14], 0.930/0.952 (hand-drawn models). We also outperform methods using hand-drawn models ([19], [14], [11]). Fig. 8 shows detections/segmentations from our method. Both internal and external contours (e.g. mug handle/outline) are segmented out.

5. Conclusion

We presented an object recognition system that locates an object, identifies its parts, and segments out its contours. Model contour candidates are greedily added to create a shape model that can explain many contours in positive im-

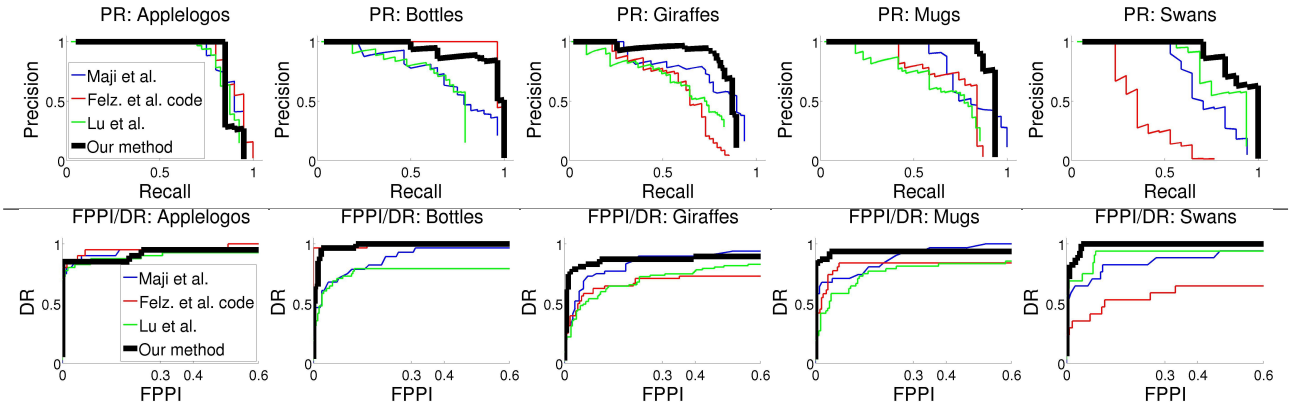

Figure 7. Precision vs. recall (PR; top row) and false positives per image vs. detection rate (FPPI/DR; bottom row) curves for our method, Maji et al. [12], code of Felz. et al. [4] and Lu et al. [11] on the ETHZ shape categories dataset. We can see that our method is comparable or significantly outperforms the other methods, particularly for those categories with significant deformation such as giraffes and swans.

	Applelogos	Bottles	Giraffes	Mugs	Swans	**Mean**
Our method	**0.95** / **0.95**	**1** / **1**	0.872 / **0.896**	**0.936** / 0.936	**1** / **1**	**0.952** / **0.956**
Maji et al. [12]	**0.95** / **0.95**	0.929 / 0.964	**0.896** / **0.896**	**0.936** / **0.967**	0.882 / 0.882	0.919 / 0.932
Felz. et al. [4] code	**0.95** / **0.95**	**1** / **1**	0.729 / 0.729	0.839 / 0.839	0.588 / 0.647	0.821 / 0.833
Gu et al. [7]	0.906 / -	0.948 / -	0.798 / -	0.832 / -	0.868 / -	0.871 / -
Ferrari et al. [5]	0.777 / 0.832	0.798 / 0.816	0.399 / 0.445	0.751 / 0.8	0.632 / 0.705	0.671 / 0.72
Stark et al. [16] 0.2 overlap	- / -	- / 0.944	- / 0.917	- / 0.845	- / 0.888	- / 0.899
Fritz et al. [6], 0.2 overlap	- / 0.899	- / 0.768	- / 0.905	- / 0.827	- / 0.754	- / 0.768
Lu et al. [11], hand-drawn	0.9 / 0.9	0.792 / 0.792	0.734 / 0.77	0.813 / 0.833	0.938 / 0.938	0.836 / 0.851
Ravishankar et al. [14], hand-drawn	0.955 / 0.977	0.909 / 0.927	0.912 / 0.934	0.937 / 0.953	0.939 / 0.969	0.930 / 0.952
Zhu et al. [19], hand-drawn	0.800 / 0.800	0.929 / 0.929	0.681 / 0.681	0.645 / 0.742	0.824 / 0.824	0.776 / 0.795

Table 1. Comparison of **detection rates for 0.3/0.4 FPPI** on ETHZ Shape Classes. Our method is tied for or leads all methods that do not use hand-drawings in 8 of 10 different detection rates for individual categories, and in both detection rates averaged across all categories. Unless otherwise noted, results were computed with 0.5 overlap score threshold and not using hand-drawn models.

ages via many-to-one matching. Given this shape model and negative images, the many-to-one matching score is tuned discriminatively to improve detection. Additional refinement ensures accurate placement of model parts and correct matching of image contours to model. Unlike previous work, our shape learning and discriminative training steps share both a shape model and many-to-one matching of contours to the model.

References

[1] GLPK software. http://www.gnu.org/software/glpk/. 3, 7

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2, 7

[3] G. Elidan, G. Heitz, and D. Koller. Learning object shape: From drawings to images. In *CVPR*, 2006. 1

[4] P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *CVPR*, 2008. 2, 6, 7, 8

[5] V. Ferrari, F. Jurie, and C. Schmid. From images to shape models for object detection. In *PAMI*, 2009. 1, 2, 7, 8

[6] M. Fritz and B. Schiele. Decomposition, discovery and detection of visual categories using topic models. In *CVPR*. IEEE Computer Society, 2008. 2, 8

[7] C. Gu, J. Lim, P. Arbelaez, and J. Malik. Recognition using regions. In *CVPR*, 2009. 8

[8] L. J. Latecki and R. Lakamper. Polygon evolution by vertex deletion. In *SCALE-SPACE*, 1999. 4, 6

[9] Y. J. Lee and K. Grauman. Shape discovery from unlabeled image collections. In *CVPR*, 2009. 1, 2

[10] B. Leibe, A. Leonardis, and B. Schiele. An implicit shape model for combined object categorization and segmentation. In *Toward Category-Level Object Recognition*, 2006. 1

[11] C. Lu, L. J. Latecki, N. Adluru, H. Ling, and X. Yang. Shape guided contour grouping with particle filters. In *ICCV*, 2009. 1, 7, 8

[12] S. Maji and J. Malik. A max-margin hough tranform for object detection. In *CVPR*, 2009. 2, 7, 8

[13] A. Opelt, A. Pinz, and A. Zisserman. A boundary-fragment-model for object detection. In *ECCV*, 2006. 2

[14] S. Ravishankar, A. Jain, and A. Mittal. Multi-stage contour based detection of deformable objects. In *ECCV*, 2008. 2, 7, 8

[15] J. Shotton, A. Blake, and R. Cipolla. Multiscale categorical object recognition using contour fragments. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(7):1270–1281, 2008. 1

[16] M. Stark, M. Goesele, and B. Schiele. A shape-based object class model for knowledge transfer. In *ICCV*, 2009. 8

[17] J. M. Winn and N. Jojic. Locus: Learning object classes with unsupervised segmentation. In *ICCV*, 2005. 1

[18] Q. Zhu, G. Song, and J. Shi. Untangling cycles for contour grouping. In *ICCV*, 2007. 1, 2, 4, 7

[19] Q. Zhu, L. Wang, Y. Wu, and J. Shi. Contour context selection for object detection: A set-to-set contour matching approach. In *ECCV (2)*, pages 774–787, 2008. 1, 3, 6, 7, 8