Learning an Alphabet of Shape and Appearance for Multi-Class Object Detection

Andreas Opelt · Axel Pinz · Andrew Zisserman

Received: 28 February 2007 / Accepted: 4 April 2008

Abstract We present a novel algorithmic approach to object categorization and detection that can learn category specific detectors, using Boosting, from a visual alphabet of shape and appearance. The alphabet itself is learnt incrementally during this process. The resulting representation consists of a set of category-specific descriptors—basic shape features are represented by boundary-fragments, and appearance is represented by patches—where each descriptor in combination with centroid vectors for possible object centroids (geometry) forms an alphabet entry. Our experimental results highlight several qualities of this novel representation. First, we demonstrate the power of purely shape-based representation with excellent categorization and detection results using a Boundary-Fragment-Model (BFM), and investigate the capabilities of such a model to handle changes in scale and viewpoint, as well as intra- and inter-class variability. Second, we show that incremental learning of a BFM for many categories leads to a sub-linear growth of visual alphabet entries by sharing of shape features, while this generalization over categories at the same time often improves categorization performance (over independently learning the categories). Finally, the combination of basic shape and appearance (boundary-fragments and patches) features can further improve results. Certain feature types are preferred by certain categories, and for some categories we achieve the lowest error rates that have been reported so far.

Keywords Generic object recognition · Object categorization · Category representation · Visual alphabet · Boosting

1 Introduction and Related Work

Object class recognition is a key issue in computer vision. Compared to the topic of recognizing previously learnt specific objects in unseen images (termed "specific object recognition", e.g. Ferrari et al. 2004; Sivic and Zisserman 2003; Nistér and Stewénius 2006) the task of object class recognition brings up additional difficulties. Models for object categories have to deal with the trade-off between modeling the intra-class variability and not confusing categories which have low inter-class variability.

There are different cues of information one could use from a training set of still images to learn models for object categories. Many approaches use appearance patches around salient points (e.g. Csurka et al. 2004; Fergus et al. 2003; Leibe et al. 2004; Ommer and Buhmann 2006) or patches using dense grid sampling on the training images (e.g. Deselaers et al. 2005; Epstein and Ullman 2005). But shape is also an important cue for object categorization, for instance humans do use shape by means of the objects silhouette to distinguish between categories even in early vision (Quinn et al. 2001). Using shape instead of appearance is not novel but is less explored for the task of categorization.

We present a novel approach to object categorization and detection (localization) that can *combine shape and appearance cues in a common visual alphabet*. This alphabet is the

A. Opelt · A. Pinz (✉)
Institute of Electrical Measurement and Measurement Signal Processing, Graz University of Technology, Graz, Austria
e-mail: axel.pinz@tugraz.at

A. Opelt
e-mail: andreas.opelt@gmail.com

A. Zisserman
Department of Engineering Science, University of Oxford, Oxford, UK
e-mail: az@robots.ox.ac.uk

basis for a codebook representation of object categories. It is a learnt selection of appearance parts or boundary-fragments from a corpus of training images. A particular instantiation of an object class in an image is then composed from codebook entries, possibly arising from different source images. However, the main focus of the paper is on representation and use of shape and geometry rather than appearance, because local appearance- (patch-) based categorization has already been extensively studied in previous research.

Examples of codebook usage include Agarwal et al. (2004), Vidal-Naquet and Ullman (2003), Leibe et al. (2004), Fergus et al. (2003, 2005), Crandall et al. (2005), Bar-Hillel et al. (2005). The methods differ on the details of the codebook, and cue of information used (shape or appearance patches), but more fundamentally they differ in how strictly the geometry of the configuration of parts constituting an object class is constrained. For example, Csurka et al. (2004), Sivic et al. (2005), Bar-Hillel et al. (2005) and Opelt et al. (2004) simply use a "bag of visual words" model (with no geometric relations between the parts at all), Agarwal et al. (2004), Amores et al. (2005), Marszalek and Schmid (2006), Wang et al. (2006), and Vidal-Naquet and Ullman (2003) use quite loose pairwise relations, whilst Fergus et al. (2003) have a strongly parametrized fully connected geometric model consisting of a joint Gaussian over the centroid position of *all* the parts. Reducing the connectivity has led to computationally simpler models, for instance the star model of Fergus et al. (2005), or the k-fan of Crandall et al. (2005). The approaches using no geometric relations are able to categorize images (as containing the object class), but generally do not provide location information (no detection), whereas the methods with even loose geometry are able to detect the object's location.

Our representation of alphabet entries with centroid votes is inspired by the method of Leibe et al. (2004), and Leibe and Schiele (2004), which has achieved the best detection performance to date on various object classes (e.g. cows, cars-rear (Caltech)). They use appearance patches as individual parts and their representation of the geometry is algorithmic—all parts vote on the object centroid as in a Generalized Hough transform (which can be considered a kind of implicit definition of a generative model, see Williams and Allan 2006). We extend this idea and add shape by means of fragments of the object's internal edges and external silhouette. The codebook consists then of *boundary-fragments* and appearance patches, with associated entries recording possible locations of the object's centroid.

1.1 Contributions and Background

The first key contribution of this paper is dedicated to specifically investigating the role of shape and geometry. We present a "Boundary-Fragment-Model" (BFM) (Opelt et al. 2006c) which is restricted to a codebook of boundary-fragments and does not represent appearance at all. The boundary represents the shape of many object classes quite naturally without requiring the appearance (e.g. texture) to be learnt and thus we can learn models using less training data to achieve good generalization. For certain categories (bottles, cups) where the surface markings are very variable, approaches relying on consistency of these appearances may fail or need a considerable amount of training data to succeed. Our BFM method, with its stress on boundary representation, is highly suitable for such objects. The intention is not to replace appearance fragments but to develop complementary features. As will be seen, in many cases the boundary alone performs as well as or better than the appearance and segmentation masks (mattes) used by other authors (e.g. Leibe et al. 2004; Vidal-Naquet and Ullman 2003)—the boundary is responsible for much of the success.

Others also used shape for object categorization. E.g. Kumar et al. (2004) used part outlines as shape in their application of pictorial structures (Felzenszwalb and Huttenlocher 2004); Fergus et al. (2004) used boundary curves between bitangent points in their extension of the constellation model; and, Jurie and Schmid (2004) detected circular arc features from boundary curves. However, in all these cases the boundary features are segmented independently in individual images. They are not flexibly selected to be discriminative over a training set, as they are here. Bernstein and Amit (2005) do use discriminative edge maps. However, theirs is only a very local representation of the boundary; in contrast we capture the global geometry of the object category. Recently, and independently, Shotton et al. (2005) presented a method quite related to the Boundary-Fragment-Model presented here. The principal differences are: the level of segmentation required in training (Shotton et al. 2005 requires more); the number of boundary fragments employed in each weak detector (a single fragment in Shotton et al. 2005, and a variable number here); and the method of localizing the detected centroid (grid in Shotton et al. 2005, mean shift here). Other methods using shape include e.g. Serre et al. (2005) who presented an approach which is biologically motivated. Based on oriented edges, they form complex features that allow small distortions in the image space but are still more selective than histogram based features. Without explicitly modeling the geometry a discriminative classifier yields good recognition performance. With slight variations on the method of Serre et al. (2005), Mutch and Lowe (2006) achieved even better results for multiple categories. Dalal and Triggs (2005) also use shape information in the form of grids of Histograms of Oriented Gradients (HOG). Studying influences of the binning of scale, orientation and position they yield excellent categorization by a SVM-based classifier.

The second key contribution of the paper concerns the *joint learning* of an alphabet that can be shared over many categories, with the possibility of adding further categories *incrementally* (Opelt et al. 2006b). With respect to shape variation and viewpoint variation, we also address how multiple aspects of one object category can be learnt with such a multi-class model. We build on the method of Torralba et al. (2004) who presented a joint multi-class Boosting approach. In their work 21 categories are jointly trained (including two aspects of cars). Torralba et al. build a strong classifier using GentleBoost from a number of weak classifiers which are shared between classes. Tackling the same problem, Tu (2005) also shows a joint training in his probabilistic Boosting tree. But in comparison to Torralba et al. Tu learns a strong classifier in each node of the classification tree. Other work on multi-class object detection includes the work of Fan (2005), Amit et al. (2004), Fei-Fei et al. (2004), Bart and Ullman (2005), Winn et al. (2005), and Shotton et al. (2006). Most closely related to our approach is the recent success by Mikolajczyk et al. (2006). A similar geometric model to our BFM is used to learn appearance clusters built from edge based features which can be shared amongst various object categories. In contrast to their method, ours mainly differs in the manner the codebook is learnt and also how the joint learning is performed. Approaches on the challenge of recognizing different aspects of one object category (e.g. cow-front and cow-side) were recently proposed by Seemann et al. (2006) and Thomas et al. (2006), both also based on the geometric model of Leibe et al. (2004). Seemann et al. (2006) use a 4-dimensional Hough Voting space (x, y, scale and aspect) in combination with a second stage of contour matching in the manner of Gavrila and Philomin (1999). This method works well on persons but seems rather restricted to this category, whereas we are proposing a method which generalizes over a set of different categories. Thomas et al. (2006) presented a combination of the geometric voting of Leibe et al. (2004) and the multi-view specific object recognition method by Ferrari et al. (2004). This method uses integrated codebooks over views to detect location and pose of objects in new test images. In contrast to their approach we present a method which uses the same algorithm for training multiple categories and multiple aspects based on shape instead of appearance patches.

Finally, our approach enables appearance and shape cues to be learnt in a unified representational model (Opelt et al. 2006d). This allows us to study how such a model benefits from the different visual cues with respect to various categories (e.g. patches might be good for spotted cats but not so suitable for motorbikes). Mixed/complementary feature types have been used previously (Fergus et al. 2004; Fergus et al. 2007; Opelt et al. 2006a; Zhang et al. 2005; Zhang et al. 2007), though, for the most part, these have been used for image classification rather than detection. For example, Opelt et al. (2006a) presented an algorithm which learns suitable category descriptors from a pool of different types of descriptors for appearance regions, and Zhang et al. (2005), used complementary descriptors (PCA-SIFT and shape context). Fergus et al. (2005) investigated detection with mixed types of features, which is closest to our work in terms of the used features (regions and edge boundaries), however their algorithm does not learn which features to use.

The paper is organized as follows: We start with an overview of our model and the required data for training, validation and testing in Sect. 2, and focus on the representation of shape by selection of boundary-fragments and learning of a visual alphabet of shape in Sect. 3. We continue with the learning of the Boundary-Fragment-Model (BFM) for a single object category and describe how this model is applied to detect instances of this category in test images in Sect. 4. Section 5 explains how the BFM can handle changes in scale and in-plane rotations, and explores its sensitivity to viewpoint changes. For these sections of the paper we use the category cows-side as a running example.

Multi-class joint and incremental learning is discussed in Sect. 6. In the experimental results Sect. 7, we discuss the role of shape-based detection using a single-category BFM (Sect. 7.1), present the development of a jointly/incrementally learnt visual alphabet for many categories (Sect. 7.2), and show that recognition rates can be improved by combining shape and appearance cues in Sect. 7.3. General merits and limitations of our approach as well as promising future research is discussed in Sect. 8.

2 Method and Data

Figure 1(a) gives an overview of our algorithm and the underlying model representations. We refer to this model as our “Unified Model” or UM approach in Sect. 7.3. It illustrates all the necessary steps of learning and detection for a single category (UIUC cars side). The slightly more complex case of multi-class joint and incremental learning is discussed later (see Fig. 15). In a similar manner to Leibe et al. (2004), we require the following data to train the model:

- A training image set with the object delineated by a bounding box.
- A validation image set with counter examples (the object is not present in these images), and further examples with the object’s centroid (but the bounding box is not necessary).

Training and validation images are required to be scale-normalized, which means that all instances of a category have to appear at roughly the same scale. Furthermore, sufficient spatial resolution is required to extract meaningful shape cues (Boundary-Fragments).

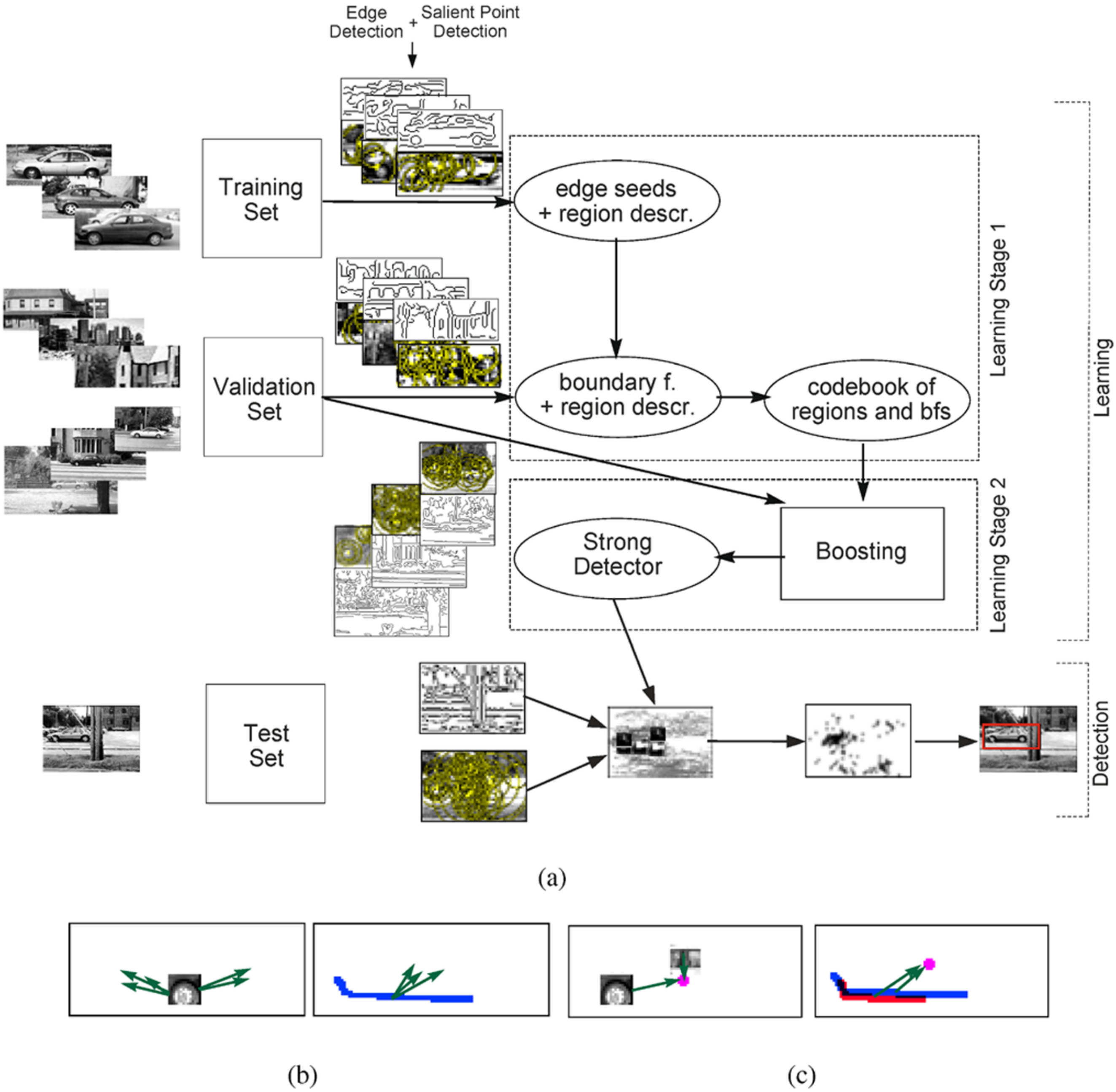


Fig. 1 Overview of our algorithm and the underlying model representations: (**a**) Learning and detection in our "Unified Model" (UM approach). (**b**) Two alphabet entries (one region, one BF). (**c**) Two weak detectors (one region-based, one BF-based)

Learning is performed in two stages. First, alphabet entries are added to a codebook depending on their significance: An alphabet entry can either be a Boundary-Fragment (BF—a piece of linked edges), or a patch (salient region and its descriptor). Each entry also casts at least one centroid vote, which is represented as a vector. Figure 1(b) shows a patch-based entry (wheel) with five vectors (front / rear wheel of slightly different sized cars) and a BF-based entry with three centroid votes (denoting the bottom of a car). An alphabet entry is considered significant if (i) it differs sufficiently from already existing entries, (ii) it discriminates the category well from counter examples, and (iii) it gives a precise estimate of the object centroid.

In the second learning stage, weak detectors are formed as pairs of two alphabet entries, and Boosting is used to select a strong detector that consists of many weak detectors. This process favors the selection of weak detectors that "fire" often on positive validation images (including a good centroid estimate), and not on the negative ones. Figure 1(c)

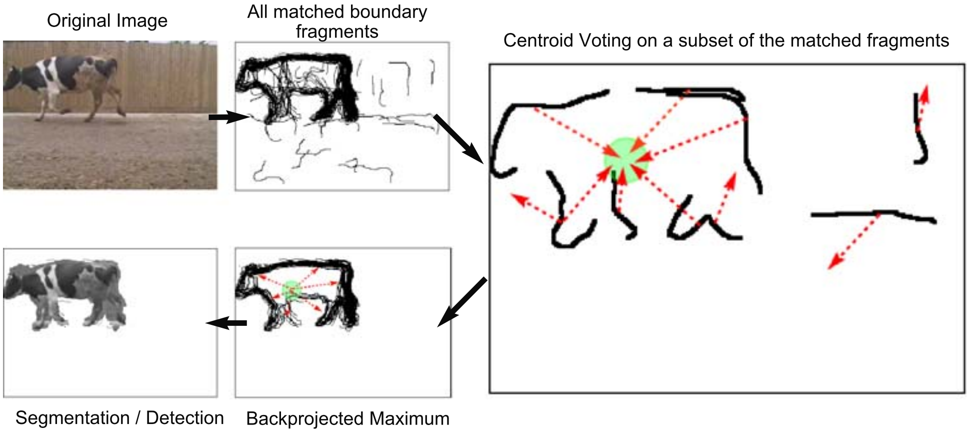

Original Image
All matched boundary fragments
Centroid Voting on a subset of the matched fragments
Segmentation / Detection
Backprojected Maximum

Fig. 2 Detection with the BFM: An overview of the steps involved in applying the Boundary-Fragment-Model detector. For clarity only a subset of the matched boundary fragments voting for the centroid are shown

shows a patch-based and a BF-based weak detector for the category cars-side.

Having learnt such a model for one category, Fig. 2 illustrates how detection works in more detail using a model for cows-side. A previously unseen test image is processed in the same manner as the training and validation images (edge extraction and BF formation or patch extraction, and region description), but the detection procedure can handle a certain range of scale variation (a factor of $0.5, \dots, 2$ of the normalized training scale). We illustrate detection using just shape information (a Boundary-Fragment-Model BFM) on a cows-side image. BFs from the codebook are matched to the edge representation of the image. Matching BFs which are learnt as a weak detector vote for one or more object centroids in a Hough voting space. Using Mean-Shift-Mode-Estimation (Comaniciu and Meer 2002) this voting space can be searched for maxima. The values of these modes are taken as evidence in the detection of an object. If the evidence is above a certain threshold the boundary fragments that voted for this maximum are backprojected in the test image which results in a localization of the object and even a rough segmentation.

2.1 Datasets

In our experiments on single-class BFM and on the combination of BFs with patches, we use a variety of datasets for evaluation of our own algorithms, as well as for comparison with the results of related work. The relevant references to these data are given for each of our experiments in Sect. 7. We had to set up our own multi-class dataset for our experiments on joint and incremental learning of BFM models of many categories as we need multiple aspects of the categories in order to evaluate sharing of shape information over viewpoints (e.g. bikes from frontal, side, and rear views).

Our multi-class dataset consists of a combination of categories from well known datasets (e.g. Caltech, GRAZ-02) and some new categories acquired from the Internet. The dataset contains various object categories, some with multiple aspects of one category (e.g. cow front and cow side), and others with specific aspects of similar categories (e.g. cow side and horse side). Each of the categories contains a different number of training, validation and test images. Table 1 lists the 17 categories, the data sources, and gives the exact numbers of training, validation, and test images. Note that we also include categories which are well suited for shape based object detection, like bottles. Figure 3 illustrates the complexity of the different categories by showing some example images of this new multi-class dataset.[1]

3 Learning a Visual Alphabet of Shape

The Boundary Fragment category Model is built from weak detectors over a set of boundary fragments selected discriminatively for a particular category. In this section we describe how boundary fragments are represented and learnt. This involves two stages: first, proposing suitable fragments from a training image set, and second, assessing the fragments suitability for a category using a validation image set.

A suitable candidate boundary fragment is required to (i) match edge chains often in the positive images but not in the negative, *and* (ii) have a good localization of the centroid in the positive images. These requirements are illustrated in Fig. 4. The idea of using validation images for discriminative learning is motivated by Sali and Ullman (1999). However, in their work they only consider requirement (i), the learning of class-discriminate parts, but not the second requirement which is a geometric relation. In the following we first explain how to score a boundary fragment according to how well it satisfies these two requirements, and then how this score is used to select candidate fragments from the training images.

[1] The complete multiclass dataset is available at http://www.emt.tugraz.at/~pinz/data/multiclass.

Table 1 Our multi-class dataset: The table lists the 17 categories, the number of training, validation and test images, and the source of the data

C	Name	Train	Val	Test	Source
1	Plane	50	50	400	Caltech (Fergus et al. 2003)
2	CarRear	50	50	400	Caltech (Fergus et al. 2003)
3	Motorbike	50	50	400	Caltech (Fergus et al. 2003)
4	Face	50	50	217	Caltech (Fergus et al. 2003)
5	BikeSide	45	45	53	Graz02 (Opelt et al. 2006a)
6	BikeRear	15	15	16	Graz02 (Opelt et al. 2006a)
7	BikeFront	10	10	12	Graz02 (Opelt et al. 2006a)
8	Cars2-3Rear	17	17	18	Graz02 (Opelt et al. 2006a)
9	CarsFront	20	20	20	Graz02 (Opelt et al. 2006a)
10	Bottles	24	30	64	ImgGoogle (Opelt et al. 2006c)
11	CowSide	20	25	65	(Magee and Boyle 2002)
12	HorseSide	30	25	96	ImgGoogle[a]
13	HorseFront	22	22	23	ImgGoogle
14	CowFront	17	17	17	ImgGoogle
15	Person	19	20	19	Graz02 (Opelt et al. 2006a)
16	Mug	15	15	15	ImgGoogle
17	Cup	16	15	16	ImgGoogle

[a] http://www.msri.org/people/members/eranb/

The cost $C(\gamma_i)$ for each candidate boundary fragment γ_i is a product of two factors:

(1) $c_{match}(\gamma_i)$: the matching cost of the fragment to the edge chains in the validation images using a Chamfer distance (Borgefors 1988; Breu et al. 1995), see (1). This is described in more detail below.

(2) $c_{loc}(\gamma_i)$: the distance (in pixels) between the true object centroid and the centroid predicted by the boundary fragment γ_i averaged over all the positive validation images

with $C(\gamma_i) = c_{match}(\gamma_i) c_{loc}(\gamma_i)$. The matching cost is computed as

$$c_{match}(\gamma_i) = \frac{\sum_{i=1}^{L^+} distance(\gamma_i, P_{v_i})/L^+}{\sum_{i=1}^{L^-} distance(\gamma_i, N_{v_i})/L^-} \qquad (1)$$

where L^- denotes the number of negative validation images N_{v_i} and L^+ the number of positive validation images P_{v_i}, and $distance(\gamma_i, I_{v_i})$ is the distance to the best matching edge chain in image I_{v_i}:

$$distance(\gamma_i, I_{v_i}) = \frac{1}{|\gamma_i|} \min_{\gamma_i \subset I_{v_i}} \sum_{t \in \gamma_i} DT_{I_{v_i}}(t) \qquad (2)$$

where $DT_{I_{v_i}}$ is the distance transform, which calculates the Euclidean distance from a point t on γ_i to the closest edge pixel in I_{v_i}. The Chamfer distance (Borgefors 1988) is implemented using 8 orientation planes with an overlap of 5 degrees. The orientation of the edges is averaged over a length of 7 pixels by orthogonal regression. The best match is found as the minimum distance by searching all possible positions and orientations of γ_i in I_{v_i}. Because of background clutter the best match is often located on highly textured background clutter, i.e. it is not correct. To solve this problem we use the $N_{match} = 10$ best matches (with respect to (2)), and from these we take the one with the best centroid prediction. Remember that the training and validation images are scale normalized.

3.1 Implementation Details

Linked edges are obtained for each image in the training and in the validation set using a Canny edge detector with hysteresis (we use the Canny edge detector with $\sigma = 1$, hysteresis thresholding $th_1 = \max(GI) * 0.2$ and $th_2 = \max(GI) * 0.1$ where GI denotes the gradient magnitude image, and an edge linking with $\min_{edgelength} = 10$ pixels to reduce clutter). Training images provide the candidate boundary fragments γ_i by selecting random starting points on the edge map of each image. Then at each such point we grow a boundary fragment along the contour. The ordering of the linked edges in the bounding box is obtained by starting with the left upper edge point, following the edge to its endpoint and then proceeding with the next unseen edge closest to that point. We are aware that smarter possibilities of building such edge graphs exist (e.g. Ferrari et al. 2006). However this straightforward method works well enough for our purpose. Growing is performed from a certain fragment

Fig. 3 Example images for each of the 17 categories of our multi-class dataset

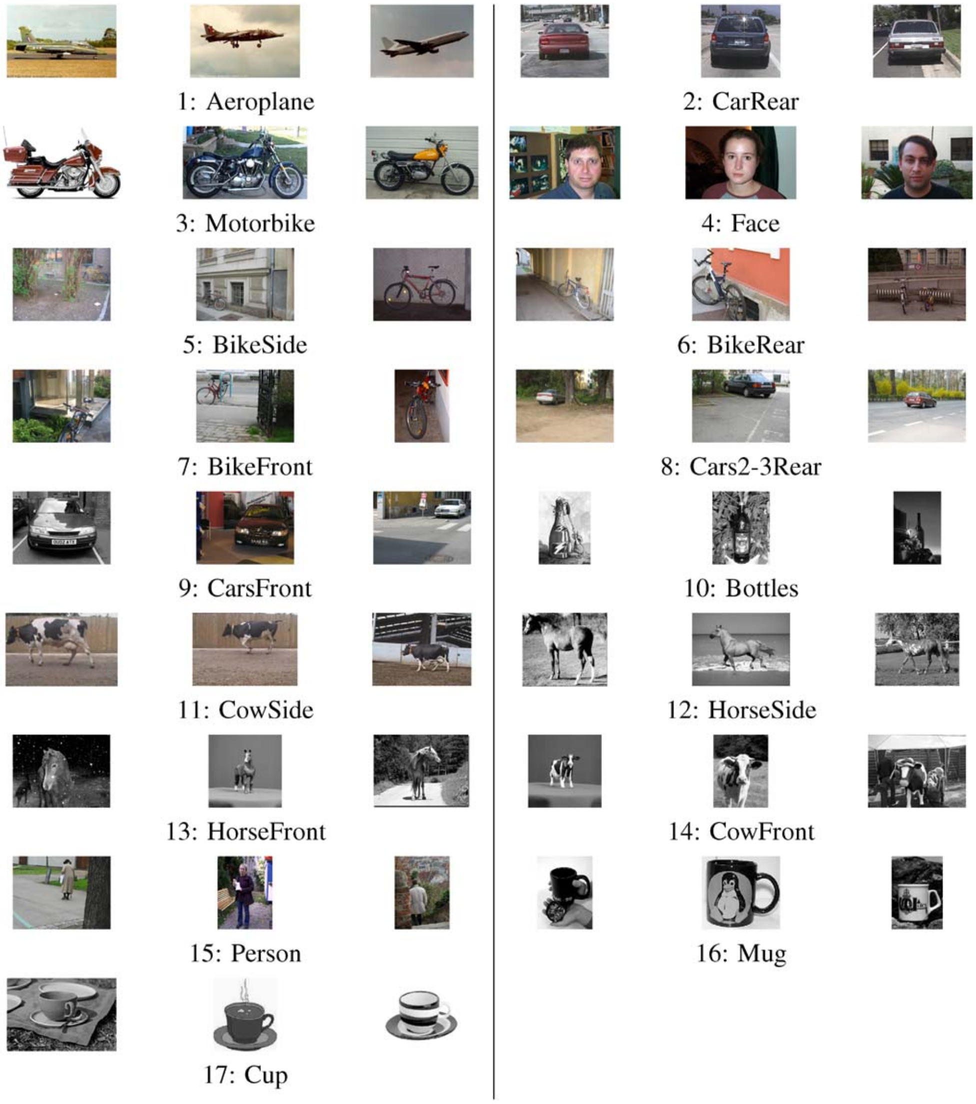
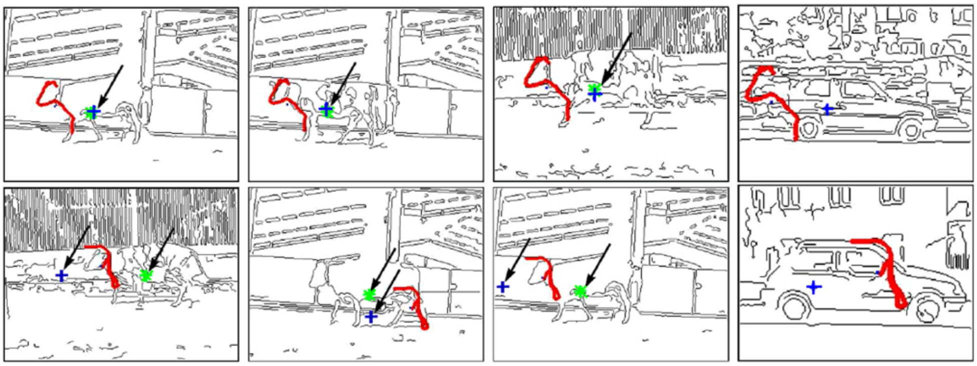


Fig. 4 Two Boundary-Fragments are validated. The fragment in the *top row* provides good centroid localization in the positive validation images, whereas the fragment in the *bottom row* does not. The *last column* shows poor localization on counter examples



starting length L_{start} in steps of L_{step} pixels until a maximum length L_{stop} is reached.[2] At each step candidates are optimized over the validation set by calculating matching costs. Figure 5 illustrates that process on one training image.

Using this procedure we obtain an alphabet of boundary fragments each having the geometric information to vote for an object centroid.

[2] We used $L_{start} = 20$, $L_{step} = 30$ in both directions, and $L_{stop} = 520$ pixels.

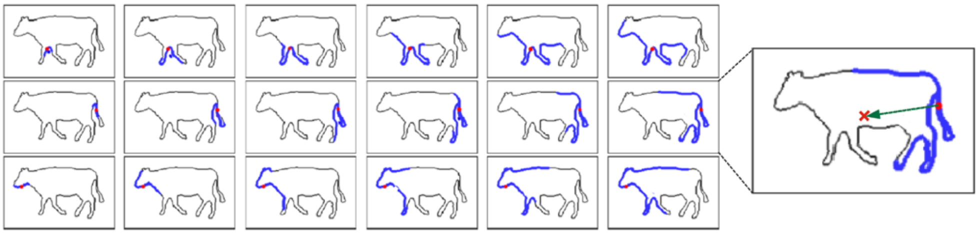


Fig. 5 Growing of candidate boundary fragments on one training image of the category cows-side starting from three different random points. In *each row* a different random starting point is used (*red dot*). On the *right* a zoomed out image is shown where the *blue edge* denotes the BF candidate and the *green dotted arrow* shows the geometric information for this BF related to the objects centroid (*red cross*)

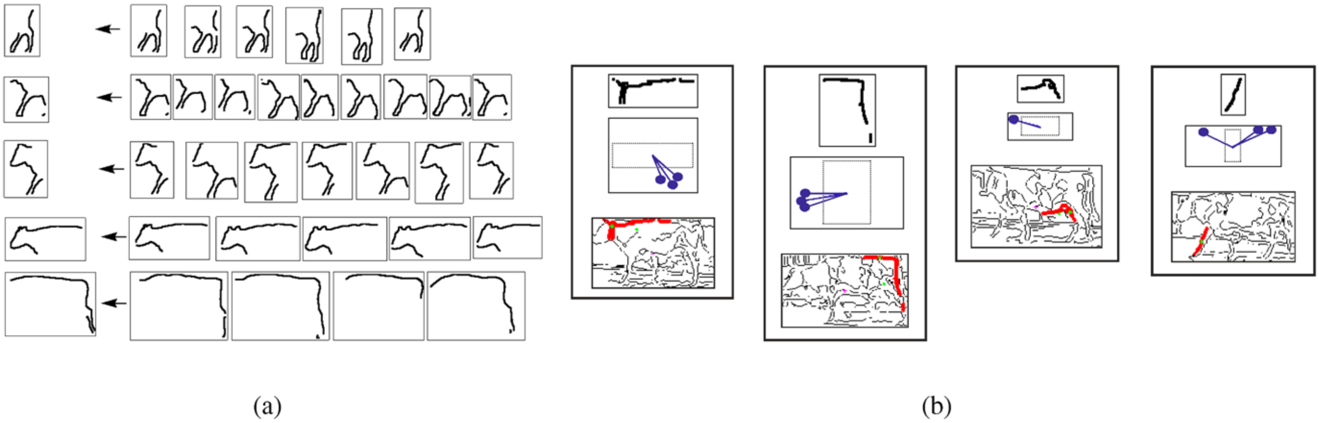


Fig. 6 (**a**) The clustering where alphabet entries on the left are selected as representatives of clusters on the right which are obtained by agglomerative clustering. (**b**) Alphabet entries learnt for the category cow-side. On *top of each entry* the boundary fragment is shown. The *second row* illustrates the centroid vectors. In the *bottom row* we show the training image where this boundary fragment was extracted from

To reduce redundancy in the codebook the resulting boundary fragment set is merged using agglomerative clustering on medoids. The distance function is $distance(\gamma_i, \gamma_j)$ (where I_{v_i} in (2) is replaced by the binary image of fragment γ_j) and we cluster with a threshold of $th_{cl} = 0.2$. Figure 6(a) shows some examples of the resulting clusters for the categories cows side, and Fig. 6(b) shows examples of the learnt alphabet entries overlaid on images. Note that each alphabet entry can have one or more centroid vectors. This optimized alphabet forms the basis for the next stage in learning the one-class BFM.

4 Learning a BFM for (Single) Category Detection

In this section we describe the Boundary-Fragment-Model (BFM) for single object category detection. We build on the alphabet of optimized boundary fragments (each carrying additional geometric information for predicting the object centroid). The BFM can be seen as a combination of these fragments so that their aggregated estimates determine the object centroid and increase the matching precision. One could use a single boundary fragment in the same way as single regions are used in Leibe et al. (2004). However, boundary fragments are not so discriminative and often (even with the use of various orientation planes) match in the background on highly complex images. To overcome this difficulty we use a combination of several (k) such fragments (for example distributed around the actual object boundary) which are more characteristic for an object category. In the following we will generally set $k = 2$, as the computational complexity increases dramatically with values of $k > 2$.

To this point the optimization procedure has chosen boundary fragments independently. We now use AdaBoost to find combinations of boundary fragments that fit well on many positive validation images. Generally Boosting is used to form a strong classifier from a weighted combination of weak classifiers (see Freund and Schapire 1997). However, we aim at *detecting* the objects in a new test image and not just to classify the image. Hence, we use a standard Boosting framework which is adapted to learn detection rather than classification. This learning method chooses boundary fragments which model the whole distribution of the training data (whereas the method of the previous section can score fragments highly if they have low costs on only a subset of

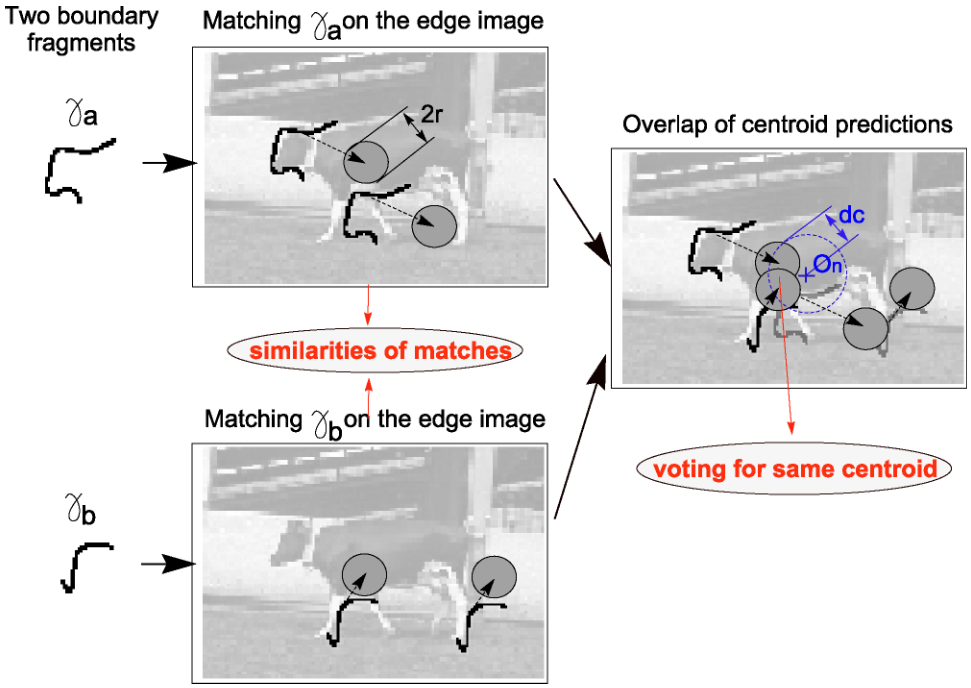

Two boundary fragments
γ_a
γ_b
Matching γ_a on the edge image
2r
similarities of matches
Matching γ_b on the edge image
Overlap of centroid predictions
dc
On
voting for same centroid

Fig. 7 Weak detector: The combination of boundary fragments to form a weak detector h_i. It fires on an image if the k boundary fragments (γ_a and γ_b) match image edge chains, the fragments agree in their centroid estimates (within an uncertainty of $2r$), and, in the case of positive images, the centroid estimate agrees with the true object centroid (O_n) within a distance of d_c

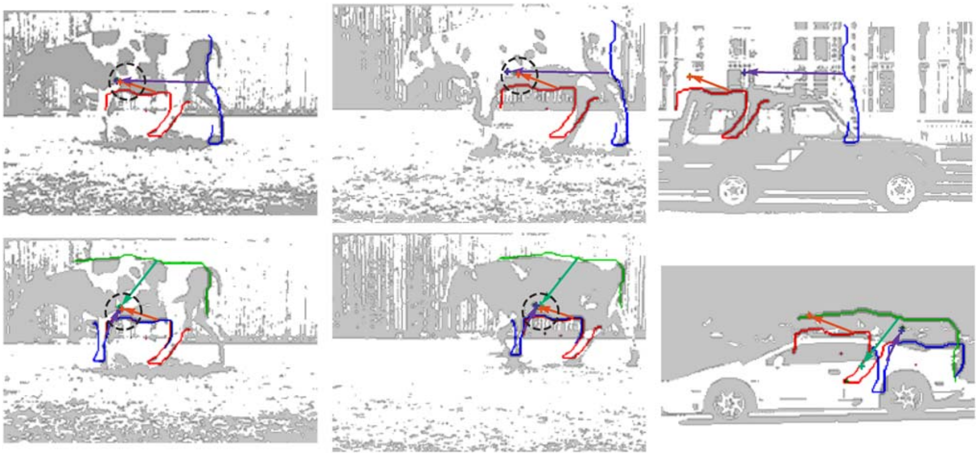


Fig. 8 Matching weak detectors to the validation set: The *top row* shows a weak detector with $k = 2$, that fires on two positive validation image because of highly compact center votes close enough to the true object center (*black circle*). In the *last column* a negative validation image is shown. There the same weak detector does not fire (votings do not concur). *Bottom row*: the same as the top with $k = 3$

the validation images) and give good predictions for the objects centroid on many images.

4.1 Building Weak Detectors as Pairs of Boundary-Fragments

We start with the idea of a weak classifier which is composed of k (typically 2) boundary fragments from the discriminative codebook learnt earlier. This could be selections of boundary fragments which match edge chains in the image and agree with their centroid estimates. However, we want to learn weak detectors as we aim for detection rather than classification. A weak detector h_i should fire ($h_i(I) = 1$) on an image I if (i) the k boundary fragments match image edge chains, (ii) the centroid estimates concur, *and*, (iii) in the case of positive images, the centroid estimate agrees with the true object centroid. Figure 7 illustrates these requirements for a weak detector with a positive detection in an image (with $k = 2$ and the boundary fragments named γ_a and γ_b), and Fig. 8 shows examples of firing and not firing.

The classification output $h_i(I)$ of detector h_i on an image I is defined as:

$$h_i(I) = \begin{cases} 1 & \text{if } D(h_i, I) < th_{h_i}, \\ 0 & \text{otherwise} \end{cases}$$

with th_{h_i} the learnt threshold of each detector (see Sect. 4.2), and where the distance $D(h_i, I)$ of h_i (consisting of k boundary fragments γ_{i_j}) to an image I is defined as:

$$D(h_i, I) = \frac{1}{m_s^2} \cdot \sum_{j=1}^{k} distance(\gamma_{i_j}, I). \quad (3)$$

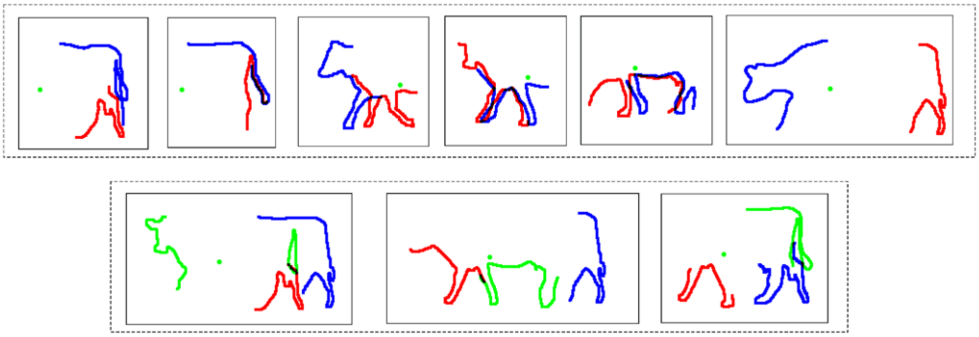


Fig. 9 Examples of weak detectors: Selected for the learnt strong detector. The *top row* shows examples for $k = 2$, and the *bottom row* for $k = 3$

1) Perform edge detection.
2) *Evaluate strong detector*:
   - For each weak detector h_i match its boundary fragments and record the compactness of their votes (m_s) for the centroid. Calculate $D(h_i, I_T)$.
   - If $D(h_i, I_T) \leq th_{h_i}$ vote in Hough space with weak detector w_{h_i} for object centroid c.
   - Use mean-shift-mode estimation (kernel radius $R = 8$ pixels) to obtain scores for the strong detector on possible object locations.
   - If the mode-value is above a threshold t_{det} declare a detection at position x_n.
   - Calculate a confidence $conf(x_n|W(x_n))$ using (5).
3) Back-project the hypotheses (the boundary fragments) that voted for these modes.
4) The back-projection of step 3 is used to segment the object.

Fig. 10 The BFM algorithm for detection and segmentation in a test image

The $distance(\gamma_{i_j}, I)$ is defined in (2) and m_s is explained below. Any weak detector where the centroid estimate misses the true object centroid by more than d_c (in our case 15 pixels), is rejected.

As shown in column 2 of Fig. 7 each fragment also estimates a centroid by a circular uncertainty window. Here the radius of the window is r. The compactness of the centroid estimate is measured by m_s (shown in the third column of Fig. 7). $m_s = k$ if the circular uncertainty regions overlap, and otherwise a penalty of $m_s = 0.5$ is allocated. These decision parameters are rather strict but experimental evaluation showed better results than for a smooth decision region (e.g. m_s as a function of the center distances). Note, to keep the search for weak detectors tractable, the number of used codebook entries (before clustering) is restricted (in out experiments we use the 300 entries with lowest costs).

4.2 Learning a Strong Detector

Having defined a weak detector consisting of k boundary fragments and a threshold th_{h_i}, we now explain how we learn this threshold and form a strong detector H out of T weak detectors h_i using AdaBoost. First we calculate the distances $D(h_i, I_j)$ of all combinations of our boundary fragments (using k elements for one combination) on all (positive and negative) images of our validation set $I_1, \ldots, I_v$. Then in each iteration $1, \ldots, T$ we search for the weak detector that obtains the best detection result on the current image weighting. This selects weak detectors which generally (depending on the weighting) "fire" often on positive validation images (classify them as correct and estimate a centroid closer than d_c to the true object centroid, see Fig. 7) and not on the negative ones. Figure 9 shows examples of learnt weak detectors that finally form the strong detector. Each of these weak detectors also has a weight w_{h_i} and a threshold th_{h_i}. The output of a strong detector on a whole test image is generally:

$$H(I) = \text{sign}\left(\sum_{i=1}^{T} h_i(I) \cdot w_{h_i}\right). \tag{4}$$

However we relax this condition such that we introduce a threshold t_{det} instead of the $sign$ function. Thus an object is detected in the image I if $H(I) > t_{det}$ and no evidence for the occurrence of an object if $H(I) \leq t_{det}$. As we train a detector this summation over the whole image would be unsuitable. Hence, Mean-Shift-Mode estimation over a probabilistic voting space is used.

4.3 Detection and Segmentation Procedure

The detection algorithm is summarized in Fig. 10, and Fig. 11 gives example qualitative results. First the edges are

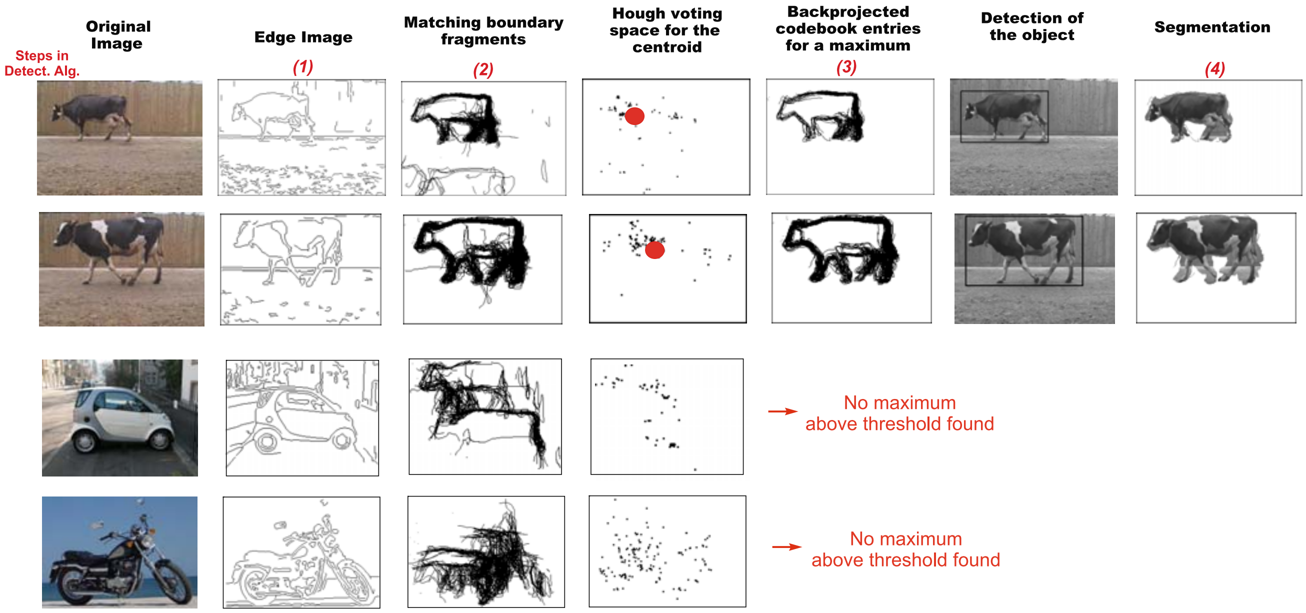

Original Image
Edge Image
Matching boundary fragments
Hough voting space for the centroid
Backprojected codebook entries for a maximum
Detection of the object
Segmentation
Steps in Detect. Alg.
(1)
(2)
(3)
(4)
No maximum above threshold found
No maximum above threshold found

Fig. 11 Examples of processing test images with the BFM detector

detected, then the boundary fragments of the weak detectors are matched to this edge image (step 2). In order to detect (one or more) instances of the object (instead of classifying the whole image) each weak detector h_i votes with a weight w_{h_i} in a Hough voting space. Votes are then accumulated in the following probabilistic manner: for all candidate points x_n found by the strong detector in the test image I_T we sum up the (probabilistic) votings of the weak detectors h_i in a 2D Hough voting space which gives us the probabilistic confidence:

$$conf(x_n) = \sum_i^T p(c, h_i) = \sum_i^T p(h_i)p(c|h_i) \qquad (5)$$

where $p(h_i) = \frac{1}{\sum_{q=1}^{M} score(h_q, I_T)} \cdot score(h_i, I_T)$ describes the pdf of the effective matching of the weak detector with $score(h_i, I_T) = 1/D(h_i, I_T)$ (see (3)) and M being the number of weak detectors matching in an image. The second term of this vote is the confidence we have in each specific weak detector and is computed as:

$$p(c|h_i) = \frac{\#fires_{correct}}{\#fires_{total}} \qquad (6)$$

where $\#fires_{correct}$ is the number of positive and $\#fires_{total}$ is the number of positive and negative validation images the weak detector fires on. Finally our confidence of an object appearing at position x_n is computed by using a Mean-Shift algorithm (Comaniciu and Meer 2002) (circular window $W(x_n)$) in the Hough voting space defined as: $conf(x_n|W(x_n)) = \sum_{X_j \in W(x_n)} conf(X_j)$.

The segmentation is obtained by back-projection of the boundary fragments of weak detectors which contributed to that center to a binary pixel map. Typically, the contour of the object is over-represented by these fragments. We obtain a closed contour of the object, and additional, spurious contours (shown in Fig. 11, step 3). Short segments (< 30 pixels) are deleted, the contour is filled (using Matlab's 'filled area' in regionprops), and the final segmentation matte is obtained by a morphological opening, which removes thin structures (votes from outliers that are connected to the object). Finally, each of the objects obtained by this procedure is represented by the bounding box of the segmentation matte. We postpone giving quantitative recognition results until the experiments of Sect. 7.

5 Extending the BFM for Recognition under Scaling and Rotation

The BFM has only a limited tolerance to scale change and rotation in the test images. We describe here how these limitations are overcome.

5.1 Search over Scale

We search over a set of scales to achieve scale invariant recognition in testing. Two possibilities have been implemented and experimentally evaluated: In the first method, a scaled codebook representation is used for each scale. Correspondingly, we normalize the parameters of the detection

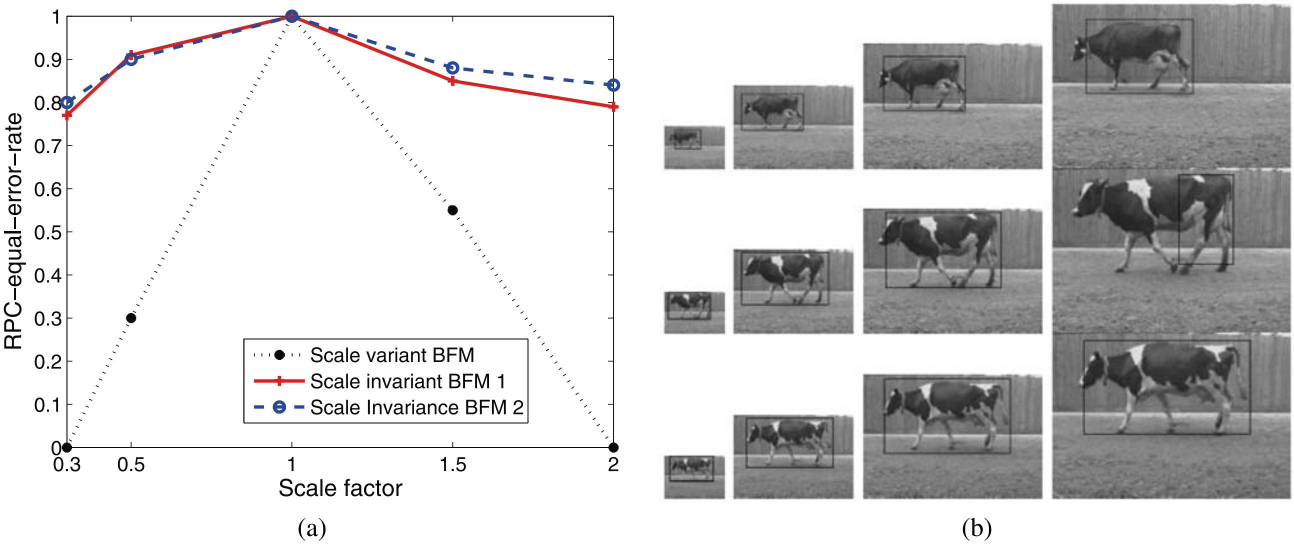


Fig. 12 (**a**) Scale invariance: RPC-equal-error rate on the cow dataset (Magee and Boyle 2002) depending on scale changes. In the experiments for each scale we used the same test images, but resized them to the selected scale. (**b**) Example of detection at different scales: The *second column* shows the object at the size of the training images (scale = 1.0). The scale varies from left to right as 0.5, 1.0, 1.5, 2.0. The *second row* of the last column shows a false detection (because the detected bounding box is too small)

algorithm (Fig. 10) with respect to scale, for example the radius for centroid estimation, in the obvious way. The Mean-Shift modes are then aggregated over the set of scales, and the maxima explored as in the single scale case.

As a second technique we improve the first one by using the idea of Leibe and Schiele (2004). Instead of several discrete 2D Hough voting spaces, the votes are now collected in a 3D Hough voting space (scale as the third axis) and then a balloon-mean-shift mode estimation is performed, which finds the modes and their corresponding scales. Leibe and Schiele (2004) use appearance patches with a characteristic scale (at discrete scale levels predefined by the scale-space) which eases the voting procedure. We use again scaled versions of our codebook with a certain step size ($factor_{sc}$), perform detection at each level and then vote in the 3D space. The advantage of this method compared to the previous one is a better theoretical foundation and a more reliable detection of objects of the same category with different scales in the same image.

Figure 12(a) shows the RPC-equal-error rate on the cow-side category depending on artificially generated scale changes with no scale invariance and the two methods proposed here (BFM 1 for re-scaled codebook and BFM 2 for the 3D Hough voting space). The drop in the detection rate is because of multiple false positive detections of the object or insufficient overlap of the bounding boxes. However, the more complicated second method does not gain much in performance. Figure 12(b) shows results on detections on various cows at different scales.

However, one problem is the general issue of re-scaling contours without losing information or getting artifacts. We use standard morphological techniques (bridging, then for small scales dilation, finalized by a skeleton operation), which works for scale ranges of 0.5–2.0 quite reliably, but does fail for bigger scale changes.

5.2 In-Plane Rotation

The BFM is invariant to small (less than 20 degrees, but depending on the number of orientation planes) rotations in plane due to the orientation planes used in the Chamfer-matching. This is a consequence of the nature of our matching procedure. For many categories the rotation invariance up to this degree may be sufficient (e.g. cars, cows) because they have a favored orientation where other occurrences are quite unnatural. For complete in-plane rotation invariant recognition we can use rotated versions of the codebook (see Fig. 25 second column for an example). Different Hough voting spaces for each rotation are obtained and then the maximum over the possible rotations is selected. However, the built in invariance means that only about 15 bins of different rotations are required.

5.3 Different Aspects and Small (Out of Plane) Viewpoint Changes

For natural objects (e.g. cows) the perceived boundary is the visual rim. The position of the visual rim on the object will vary with pose but the shape of the associated boundary fragment will be valid over a range of poses. The BFM implicitly couples fragments via the centroid, and so is not as flexible as, say, a “bag of” features model where feature

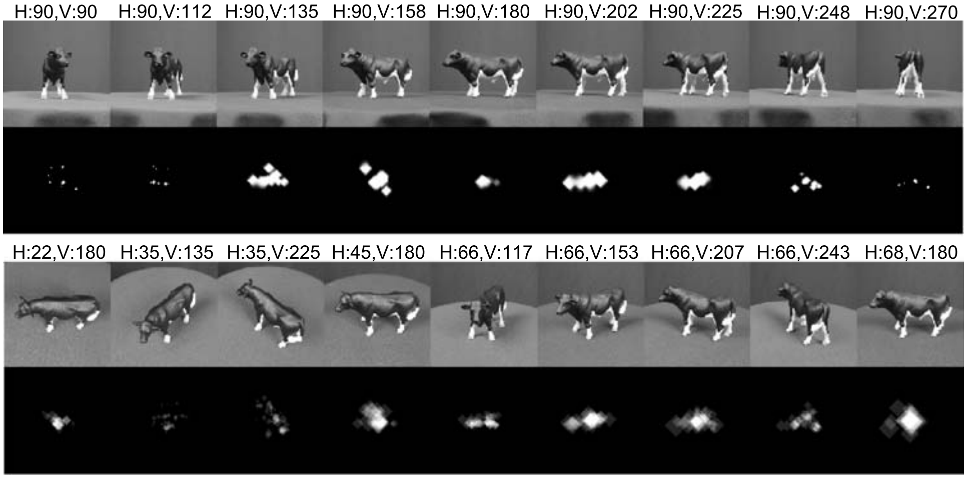


Fig. 13 Robustness to changes in viewpoint: The robustness of the BFM to viewpoint changes to rotations about a vertical axis (V) and/or about a horizontal (H) axis. Degrees of viewpoint angles are stated above the images. The circular centroid vote gets blurred to an ellipse corresponding to the viewpoint rotation. Up to a certain degree of rotation the centroid is still prominently observable

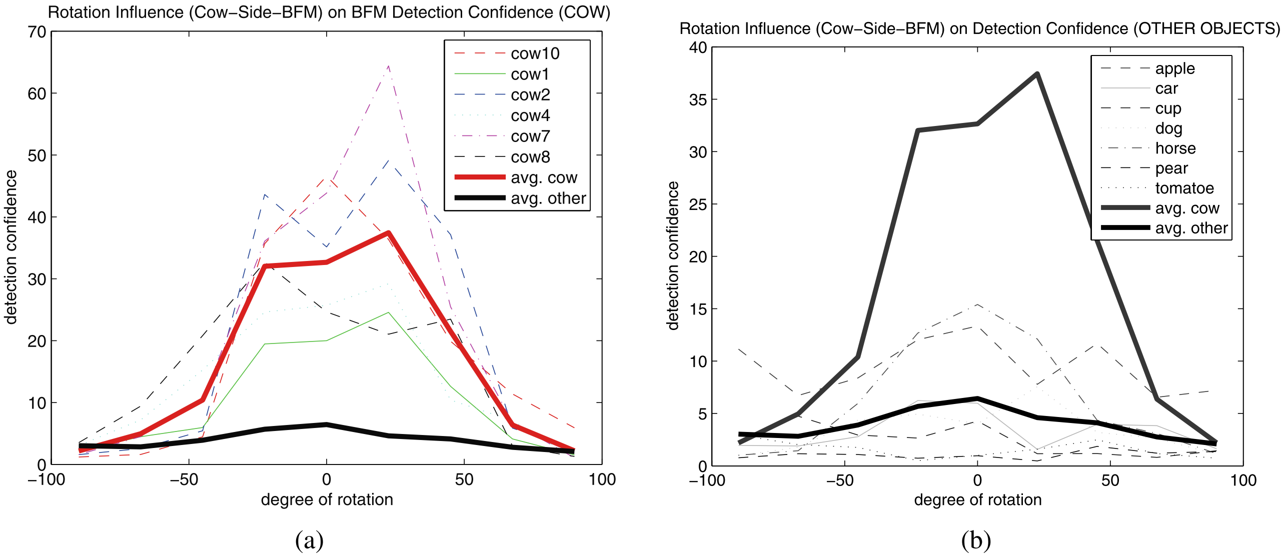


Fig. 14 The detection confidence with a change in viewpoint: The BFM learnt for cow-side is used. In (**a**) the *thick black solid line* shows the average confidence with this model on other objects but cows and the *red thick solid line* shows the average confidence on 6 different cows. (**b**) Shows the same model tested on cows rotated about a vertical axis, and also on images of other categories similarly rotated

position is not constrained. We investigate qualitatively the tolerance of the model to viewpoint change. The evaluation is carried out on the ETH-80 dataset. This is a toy dataset, but is useful here for illustration because it contains image sets of various instances of categories at controlled viewpoints. More realistic experiments on viewpoint change are given in Sect. 7.

We carry out the following experiment: a BFM model is learnt from instances of the cow category in side views. The model is then used to detect cows in test images which vary in two ways: (i) they contain cows (seven *different* object instances) over varying viewpoints—object rotation about a vertical and horizontal axis (see Fig. 13); (ii) they contain instances of other categories (horses, apples, cars ...), again over varying viewpoints.

Figure 13 shows the resulting Hough votes on the centroid, averaged over the seven cows with viewpoint changes about a vertical axis (second row) or changes about the vertical and/or the horizontal axis (last row). It can be seen that the BFM model is robust to significant viewpoint changes with the mode still clearly defined (though elongated). Figure 14 summarizes the change in the detection response averaged over the different cows or other objects under rotation about a vertical axis (as in the top row of Fig. 13). Figure 14(a) shows the detection confidences of various cow instances rotated about a vertical axis compared to an average of other objects. Note that the cow detection response is above that of other non-cow category objects. The side-trained BFM can still discriminate object class based on detection responses with rotations up to 45 degrees in both directions. In summary: the BFM trained on one visual aspect can correctly detect the object class over a wide range of viewpoints, with little confusion with other object classes.

Similar results are obtained for BFM detectors learnt for other object categories (e.g. horses), whilst for some categories with greater invariance to viewpoint (e.g. bottles) the response is even more stable. These results allow us to cut down the bi-infinite space of different viewpoints to a few category relevant aspects. These aspects allow the object to be categorized and also to predict its viewpoint.

Generally, we currently treat several viewpoints of an object category as different categories, and the following section on multi-class detection gives examples how such cases are treated.

6 The BFM for Multiple Categories

The previous sections have shown that, using our basic single category BFM, shape can be a strong cue for categorization. This idea is now enhanced towards the learning and detection of many categories. Here it is necessary to develop algorithms with a sublinear growing effort with the numbers of categories instead of learning a full separate model for each category. Thus, our multi-class BFM is based on a novel joint learning algorithm which is a variation on that of Torralba et al. (2004), where weak classifiers are shared between classes. The principal differences are that our algorithm allows *incremental* as well as joint learning, and we learn a regressor of the object location (which is a direct implementation of a *detector*) rather than the *classification* of an image window, and detection by scanning over the whole image as it is done in Torralba et al. (2004). A less significant difference is that we use AdaBoost (Freund and Schapire 1997) instead of GentleBoost (Friedman et al. 1998). The main benefits of the approach, over individual learning of category detectors, are: (i) that we need less training data when sharing across categories; and (ii) that we are able to add new categories incrementally making use of already acquired knowledge. This approach leads to a universal visual alphabet of shape that is shared between many categories.

Figure 15 gives an overview of the two cases of joint learning many categories and incremental learning of a new category. Considering the first case, we have a training and a validation set for each category and a set of background images in the validation set. We proceed with every training image of every category, extract boundary-fragment candidates around edge seeds and calculate costs on the corresponding validation set. If the costs are below a certain threshold we add the boundary-fragment with its geometric information (centroid vectors) to the alphabet. However, now we proceed with the same boundary-fragment and evaluate it also on all the other validation sets of the other categories. This results in alphabet entries which have costs for all categories specifying their ability of sharing. This incrementally built alphabet is then used as a basis for joint-boost to learn a strong detector for each category which shares weak detectors from a collection of weak detectors. The second case of learning a newly added category incrementally is based on the existing (previously learnt) knowledge of an alphabet and a collection of weak detectors. We add a new category with its training images and validation images and then learn a strong detector in a two stage process. First existing alphabet entries and the weak detectors are evaluated on the new validation set and if they can be shared they are added to the strong detector for the new category. These are weak detectors which discriminate the category from the background. Then in the second stage we learn new weak detectors which are used to discriminate the incrementally added category from the other categories.

6.1 A Universal Alphabet of Shape

Building the alphabet of shape for many categories is based on the process for the one-class BFM (see Sect. 3). However, we also search over other categories to see if a boundary fragment can be shared. The search algorithm is outlined in Fig. 16. Note that in step 4 we have to distinguish between the following three different cases (which are illustrated in Fig. 17):

- The boundary fragment matches on many positive validation images of another category and gives a roughly correct prediction of the object centroid. In this case we just update the alphabet entry with the new costs for this category and sharing is possible.
- The boundary fragment matches well on many positive validation images, but the prediction of the object centroid is not correct, though often the predictions for each match are consistent with each other. In this case we add a new centroid vector to the alphabet entry. We are still able to share the boundary fragments but not the geometric information.
- The third obvious case is where the boundary fragment matches arbitrarily in validation images of a category in which case high costs emerge and sharing is not possible.

6.2 Incremental/Joint-Boosting

The algorithm can operate in two modes: either joint learning (as in Torralba et al. 2004); or incremental learning. In both cases our aim is a reduction in the total number of weak detectors required compared to independently learning each class. For C classes this gain can be measured by $\sum_{i=1}^{C} T_{c_i} - T_s$ (as suggested in Torralba et al. 2004) where T_{c_i} is the number of weak detectors required for each class

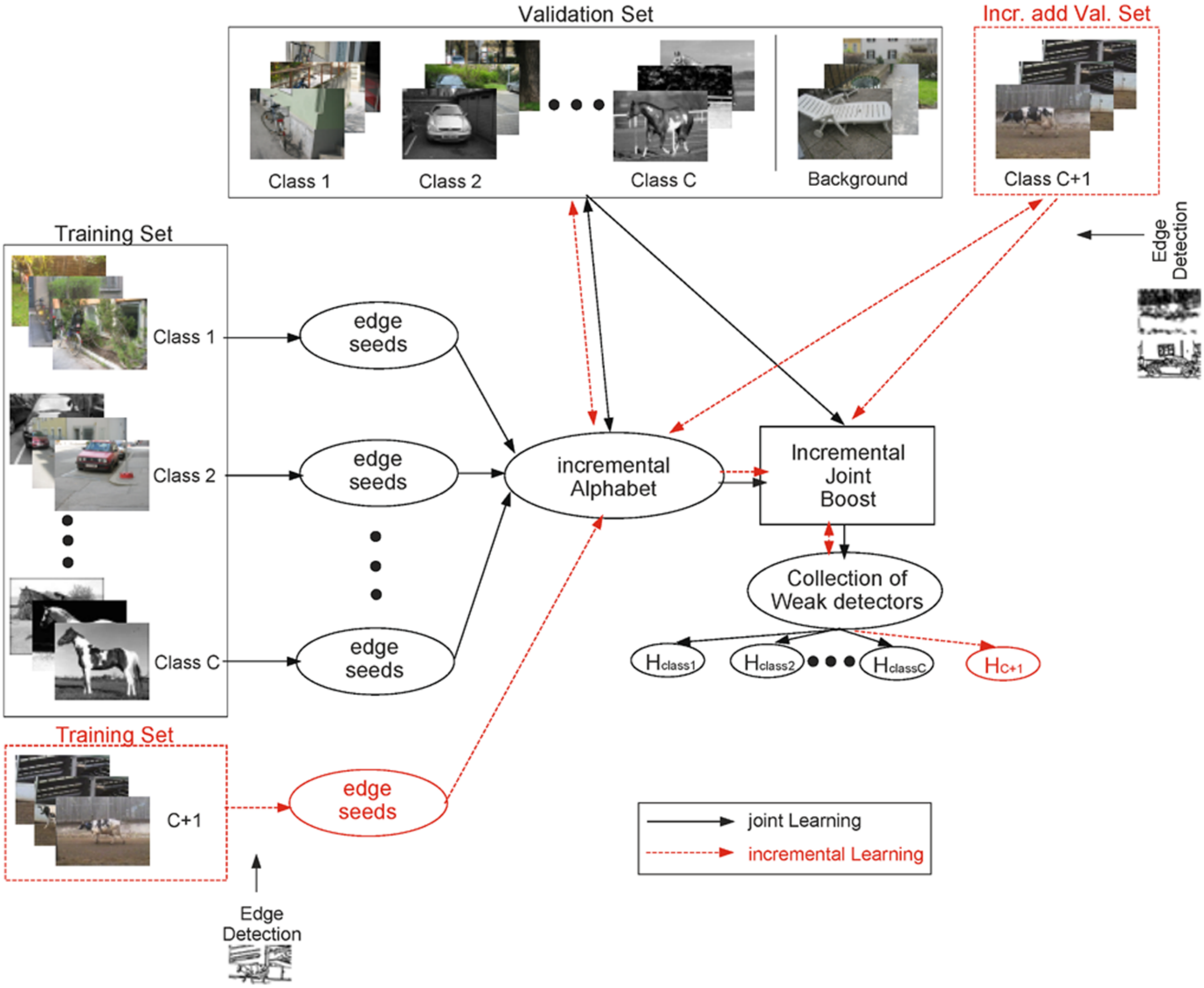


Fig. 15 An overview of the procedure to learn a multi-class Boundary-Fragment-Model for jointly learning many categories (*black, solid lines*) or adding a new category incrementally (*red, dotted lines*)

```
For each class $C_i$
        For each training image of $C_i$
                For each random edge seed

1) Grow candidate boundary fragment around random starting point (edge seed).
2) Evaluate the boundary fragment at each growth step on the validation set of the category. Calculate costs.
3) If the fragments costs are above a certain threshold discard this fragment, otherwise go on with step 4.
4) Evaluate the boundary fragment on the validation sets of the other categories (3 cases, see Fig. 17).
5) Add this fragment with costs on all categories and the geometric information to the alphabet.
```

Fig. 16 The algorithm to build the alphabet of shape for many categories

trained separately (to achieve a certain error on the validation set) and T_s is the number of weak detectors required when sharing is used. In the separate training case this sum is $O(C)$, whereas in the sharing case it should grow sub-linearly with the number of classes. The algorithm optimizes an error rate E_n over all classes.

(a) *Joint learning:* involves for each iteration searching for the weak detector for a subset $S_n \in \mathcal{C}$ that has the lowest accumulated error E_n on all classes S_n. Subsets might be e.g. $S_1 = \{c_2\}$ or $S_3 = \{c_1, c_2, c_4\}$. A weak detector only fits for a category if ϵ^{c_i} on this category c_i is below 0.5 (and is rejected otherwise), where ϵ^{c_i} is the training error

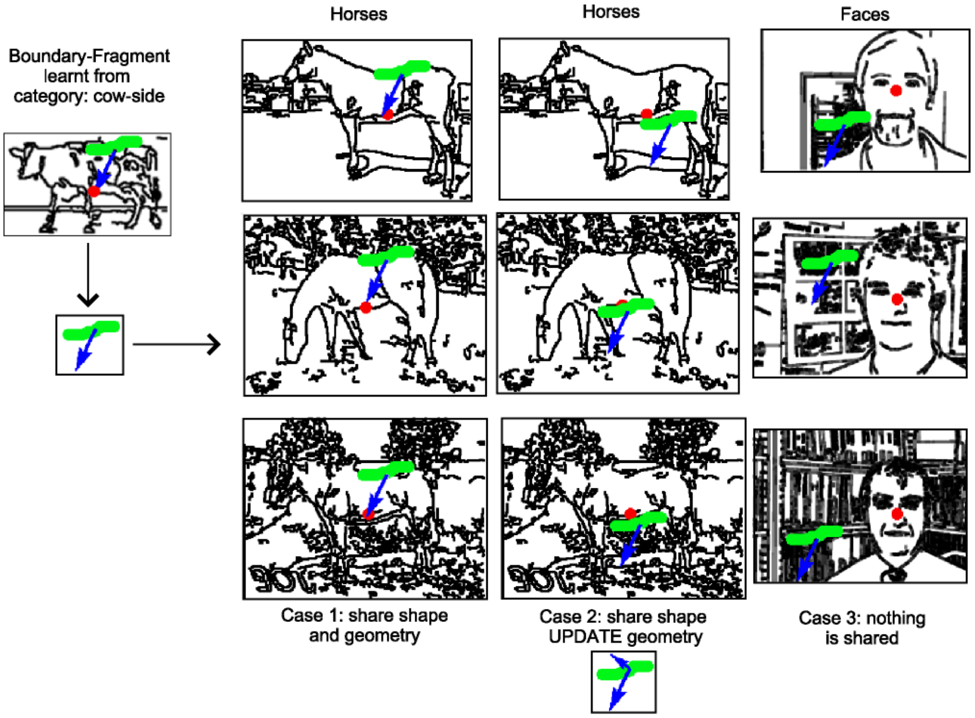

Boundary-Fragment learnt from category: cow-side
Horses
Horses
Faces
Case 1: share shape and geometry
Case 2: share shape UPDATE geometry
Case 3: nothing is shared

Fig. 17 Illustrates the three cases that can occur when alphabet entries are evaluated on the validation sets of other categories

of the category c_i. E_n is the sum of all class specific errors ϵ^{c_i} if $c_i \in S_n$ and a penalty error ϵ_p (0.6 in our implementation) otherwise. Searching for a minimum of E_n over a set of subsets S_n guides the learning towards sharing weak detectors over several categories. We give a brief example of that behavior: imagine we learn three categories, c_1, c_2 and c_3. There is one weak detector with $\epsilon^{c_1} = 0.1$ but this weak detector does not fit any other category ($\epsilon^{c_2} > 0.5$ and $\epsilon^{c_3} > 0.5$). Another weak detector can be found with $\epsilon^{c_1} = 0.2$, $\epsilon^{c_2} = 0.4$ and $\epsilon^{c_3} = 0.4$. In this case the algorithm would select the second weak detector as its accumulated error of $E_n = 1.0$ is smaller than the error of the first weak detector of $E_n = 1.3$ (note that for each category not shared ϵ_p is added). This makes the measure E_n useful to find detectors that are suitable for both distinguishing a class from the background, *and* for distinguishing a class from other classes. Clearly, the degree of sharing is influenced by the parameter ϵ_p, and this enables us to control the degree of sharing in this algorithm (a larger ϵ_p encourages sharing). Instead of exploring all $2^C - 1$ possible subsets S_n of the jointly trained classes $\mathcal{C}$, we employ the maximally greedy strategy from Torralba et al. (2004). This starts with the first class that achieves alone the lowest error on the validation set, and then incrementally adds the next class with the lowest training error. The combination which achieves the best overall detection performance over all classes is then selected. Torralba et al. (2004) showed that this approximation does not reduce the performance much.

(b) *Incremental learning:* implements the following idea: suppose our model was jointly trained on a set of categories $\mathcal{C}_L = \{c_1, c_2, c_3\}$. Hence the "knowledge" learnt is contained in a set of three strong detectors $H_L = \{H_1, H_2, H_3\}$ which are composed from a set of weak detectors h_L. The number of these weak detectors depends on the degree of sharing and is defined as $T_s \leq \sum_{i=1}^{C} T_{c_i}$ ($C = 3$ here). Now we want to use this existing information to learn a detector for a new class c_{new} (or classes) incrementally. To achieve this, one can search already learnt weak detectors h_L to see whether they are also suitable ($\epsilon^{c_{new}} < 0.5$) for the new class. If so, these existing weak detectors are also used to form a detector for the new category and only a reduced number of new weak detectors have to be learnt using the joint learning procedure.

Note that joint and incremental training reduces to standard Boosting if there is only one category.

(c) *Weak detectors:* are formed from pairs of fragments. The possible combinations of k fragments define the feature pool (the size of this set is the binomial coefficient of k and the number of alphabet entries). This means for each sharing of each iteration we must search over all these possibilities to find our best weak detector. We can reduce the size of this feature pool by using only combinations of boundary fragments which can be shared over the same categories as candidates for weak detectors. E.g. it does not make much sense to test a weak detector which is combined from a boundary fragment representing a horses leg and one that represents a bicycle wheel if the boundary horses leg never matches in the bike images.

(d) *Details of the algorithm:* The algorithm is summarized in Fig. 18. We train on C different classes where

```
Input: Validation images $(I_1, \ell_1^0), \ldots, (I_N, \ell_N^C)$,
$\ell_i^c \in \{\mathcal{C}, -1\}$, $N = N_{bg} + \sum_{i=1}^{C} N_{c_i}$.
Initialization: Set the weight matrices $w_i^c$:
$w_i^c = \begin{cases} \frac{1}{2N_{c_i}} & \text{if } \ell_i = c. \\ \frac{1}{2(N_{bg}+\sum_{i=1,c_i \neq \ell_i}^{C} N_{c_i})} & \text{else} \end{cases}$
Learn incrementally:
For $c_i = 1 : C$
    For $h_L(I, S_n) \in H_L(I, c)$
        if $\epsilon^{c_i} < 0.5$: $h_L = h_L(I, S_n \cap c_i)$, update $w_i^c$, $t = t + 1$
$T_{c_i} = T_{c_i} + 1$
For $t = 1, \ldots, T_{max}$
1. For $n = 1, 2, .., \frac{C(C+1)}{2}$
    a) Find the best weak detector $h_t(I, S_n)$ w.r.t. the weights $w_i^{S_n}$.
    b) Evaluate error:
        $E_n = \begin{cases} \sum_c^C \epsilon^c & \text{if } \epsilon^c < \frac{1}{2}, \forall c \in S_n \\ C & \text{else} \end{cases}$
        with $\epsilon^c = \begin{cases} \frac{\sum_{i=1}^{N} w_i^c \cdot (\frac{1}{2}(\ell_i^c - h_t(I_i, S_n))^2)}{\sum_{i=1}^{N} w_i^c} & \text{if } \ell_i \in S_n, \\ \epsilon_p & \text{otherwise.} \end{cases}$
2. Get best sharing by selecting: $n = argmin_n E_n$ and pick corresponding $h_t$, $S_n$
3. Update additive model and weights:
    $H(I, c) = H(I, c) + \alpha_t h_t(I, S_n)$
    $w_i^c \leftarrow w_i^c \cdot \alpha^{\ell_i^c h_t(I_i, c)}$
    with $\alpha_t = \frac{1}{2}\log(\frac{1-\epsilon^c}{\epsilon^c})$, and $\epsilon^c = p$ for $c \notin S_n$
4. Update $T_{c_i}$, and if $T_{c_i} \geq T \ \forall c_i \rightarrow STOP$
```

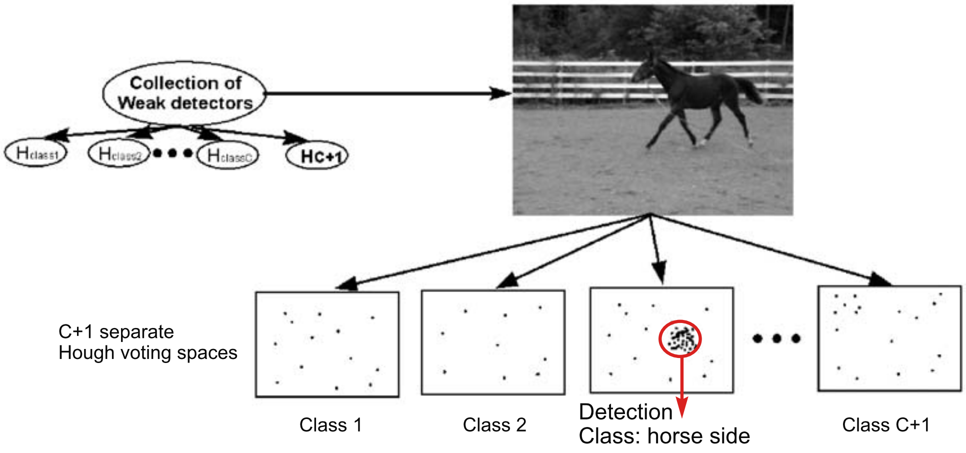
Fig. 18 Incremental/joint Adaboost learning algorithm

each class c_i consists of N_{c_i} validation images, and a set of N_{bg} background validation images (which are shared for all classes and are labeled ℓ_i^0). The total number of validation images for all classes and background is denoted by N. The weights are initialized for each class separately. This results in a weight vector w_i^c of length N for each class c_i, normalized with respect to the varying number of positive validation images N_{c_i}. In each iteration a weak detector for a subset S_n is learnt. To encourage the algorithm to focus also on the categories which were not included in S_n we vary the weights of these categories slightly for the next iteration ($\epsilon^c = p, \forall c \notin S_n$, with $p = 0.47$ in our implementation). Note that we use a fixed number of weak detectors, T, per category rather than train until the validation error is below some threshold (as is done by Torralba et al. 2004). In Torralba et al. (2004) the authors use weak classifiers on subwindows of the training images and can so easily track the training error at each iteration. However, we are learning weak detectors and to achieve a proper evaluation of the training error in each iteration for each category we would have to perform the whole Hough voting detection procedure. This slows down the training a lot and our experimental results show that a fixed number of weak detectors per category gives excellent results. Still for further improvement one could include this full Hough evaluation, and thereby save some effort for certain categories where fewer weak detectors are sufficient to form a suitable model.

6.3 Detection in the Multi-Class Case

Learning the strong detectors results in a collection of weak detectors which are shared among the categories. If one wants to detect objects in a new test image, then one could simply follow the procedure of the single class BFM detection algorithm (Fig. 10) for each class independently. However, it is straightforward to extend the procedure to include multiple categories. Each of the weak detectors from the collection of weak detectors is applied to the test image. We have separate voting spaces for each category. If the weak detector matches it votes in the corresponding Hough voting spaces for those categories that share this specific weak detector. After testing all the weak detectors we perform Mean-Shift-Mode estimation on all voting spaces. Modes above a certain threshold are treated as detection of the category this voting space belongs to (see Fig. 19). The resulting bounding box and rough segmentation is obtained as in the one-class BFM. Note that this procedure can handle multiple detections of a specific category (more than one mode above

Fig. 19 Shows how detection is performed using the multi-class BFM



threshold in this categorie's voting space), as well as detections of several categories in one image (significant modes in several voting spaces). Detection time is linear in the number of categories ($O(C)$).

Table 2 Comparison of the BFM detector to other published results on cows

Method	Caputo et al. (2004)	Leibe et al. (2004)	Our approach
RPC-equal-err	2.9%	0.0%	0.0%

7 Experimental Results

Our experiments are structured into three subsections. First, the features of our basic BFM for single categories are evaluated and compared with related work on common datasets. These results in Sect. 7.1 show that shape alone is a strong cue for category detection, and that our BFM performs comparably or even better than other state of the art categorization approaches which are based on appearance patches or on shape features. Next, in Sect. 7.2 we investigate the multi-class BFM on our novel multi-class dataset. The emphasis of our experiments is to analyze the visual shape alphabet (which grows sublinearly with the number of categories), to compare incremental and joint learning, and to compare with detection results of related work on a number of individual categories from the multi-class dataset. Finally, in Sect. 7.3 we return to the unified approach which is outlined in Fig. 1. We combine boundary-fragments and appearance patches in a unified framework and show that this combination of diverse cues improves detection rates as compared to our BFM as well as to related patch- and shape-based approaches.

Unless stated otherwise, our experiments were performed with the following parameter settings (details about the parameters can be found throughout the paper): Canny $\min_{edgelength} = 10$; growing of candidate boundary fragments $L_{start} = 20$, $L_{step} = 30$ and $L_{stop} = 520$; chamfer matching with 8 orientation planes, 5 degrees overlap, orientation averaged over 7 pixels, and $N_{match} = 10$ best matches of a boundary fragment; agglomerative clustering threshold $th_{cl} = 0.2$; centroid uncertainty $r = 10$ and centroid estimate tolerance $d_c = 15$; $k = 2$ boundary fragments form a weak detector; $T = 200$ weak detectors form a strong detector; detection threshold for a strong detector $t_{det} = 8$.

We consider a detection as correct (true positive) if the bounding box predicted by the algorithm BB_{pred} has at least 50% area of overlap a_o with the ground truth bounding box BB_{gt}. As suggested by Everingham et al. (2005) this ratio of area of overlap is defined by $a_o = \frac{area(BB_{pred} \bigcap BB_{gt})}{area(BB_{pred} \bigcup BB_{gt})}$.

7.1 Experimental Results for Single-Category BFM

(a) *Cows:* First we give quantitative results on the cow dataset. We used 20 training images (validation set: 25 positive 25 negative) and tested on 80 unseen images half belonging to the category cows and half to counter examples (cars and motorbikes). Note that we provided contours as supervision for this dataset as was done in Leibe et al. (2004). In Table 2 we compare our results to those reported by Leibe et al. (2004) and Caputo et al. (2004) (Images are from the same test set—though the authors do not specify which ones they used). We perform as well as the result in Leibe et al. (2004), clearly demonstrating that in some cases just the contour is sufficient for an excellent detection performance. Figure 20 shows example segmentations of detected objects. The segmentation uses the back-projected outline of the object to delineate a foreground region.

Kumar et al. (2004) also give an RPC curve for cow detection with an ROC-equal-error rate of 10% (though they use different test images). Note, that our detector can identify multiple instances in an image, as shown in Fig. 21.

Table 3 Comparison of the BFM detector to other published results on the Caltech dataset. The first two columns give the actual object detection error reported in RPC-equal-error (BFM-D) and the remaining columns the categorization of the images (BFM-C) given by the ROC-equal error rates

Cat.	BFM-D	Leibe et al. (2004)	BFM-C	Fergus et al. (2003)	Opelt et al. (2004)	Sivic et al. (2005)	Amores et al. (2005)	Bar-Hillel et al. (2005)	Fergus et al. (2005)	Thureson and Carlsson (2004)	Zhang et al. (2005)	Caputo et al. (2004)
Cars-rear	**2.25**	6.1	**0.05**	9.7	8.9	21.4	3.1	2.3	0.7	9.8	–	2.2
Airplane	**7.4**	–	**2.6**	7.0	11.1	3.4	4.5	10.3	4.7	17.1	5.6	–
Motorbikes	**4.4**	6.0	**3.2**	6.7	7.8	15.4	5.0	6.7	6.2	6.8	5.0	–
Faces	**3.6**	–	**1.9**	3.6	6.5	5.3	10.5	7.9	17.0	16.9	0.3	7.6

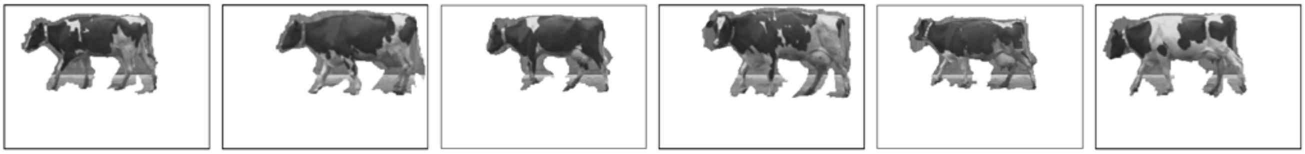


Fig. 20 *Segmentation of cows:* Example segmentations obtained using the BFM cow detector

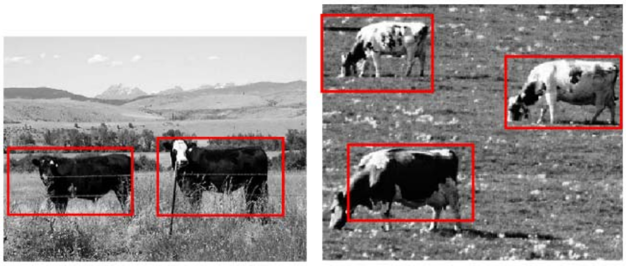


Fig. 21 Detecting multiple objects in one image

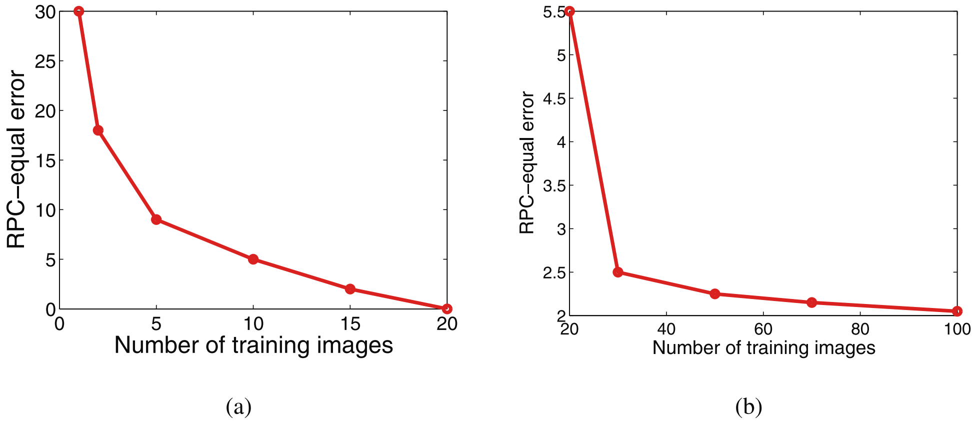
(b) *Variation in performance with number of training images:* The results on the cow dataset reported above have been achieved using 20 training images. Figure 22(a) shows how the number of training images influences the performance of the BFM detector. Even with five images our model achieves detection results of better than 10% RPC-equal-error rate. The performance saturates at twenty in this case, but this number is dependent on the degree of within class variation (e.g. see Fig. 22(b) for the category Cars-Rear) and the amount of supervision.

(c) *Caltech datasets:* From the widely used Caltech datasets we performed experiments on the categories Cars-Rear, Airplanes, Motorbikes and Faces. Table 3 shows our results compared with other state of the art approaches on the same test images as reported in Fergus et al. (2003).

First we give the detection results (BFM-D) and compare them to the best (as far as we know) results on detection by Leibe et al. (2004) (scale changes are handled as described in Sect. 5.1). We achieve superior results—even though we only require the bounding boxes in the training images.

For the classification results (BFM-C) an image is classified, in the manner of Fergus et al. (2003), if it contains the object, but localization by a bounding box is not considered. Compared to recently published results on this data we again achieve the best results. Note that the amount of supervision varies over the methods where e.g. Thureson and Carlsson (2004) use labels and bounding boxes (as we do); Amores et al. (2005), Bar-Hillel et al. (2005), Caputo et al. (2004), Fergus et al. (2003), Opelt et al. (2004) use just the object labels; and Sivic et al. (2005) uses no supervision. It should be pointed out that we use just 50 training images, and 50 positive as well as 50 negative validation images for each category, which is less than the other approaches use. Figure 22(b) shows the error rate depending on the number of training images (again, the same number of positive and negative validation images are used). However, it is known that the Caltech images are now not sufficiently demanding, so we consider some more challenging datasets in Sect. 7.2.

(d) *Horses and cow/horse discrimination:* To address the topic of how well our method performs on categories that consist of objects that have a similar boundary shape we attempt to detect and discriminate horses and cows. We use the horse data from http://www.msri.org/people/members/eranb/ to be comparable to others. In the following we compare three models. In each case they are learnt on 20 training images of the category and a validation set of 25 positive and 25 negative images that is different for each model. The first model for cows (cow-BFM) is learnt using no horses in the negative validation set (13 cars, 12 motorbikes). The second model for horses (horse1-BFM) is learnt using also cows in the negative validation set (8 cars, 10 cows, 7 motorbikes). Finally we train a model (horse2-BFM) which uses just cow



(a)

(b)

Fig. 22 (**a**) The RPC-equal-error rate depending on the number of training images for the cow dataset. (**b**) The error depending on the number of training images (for Cars-Rear)

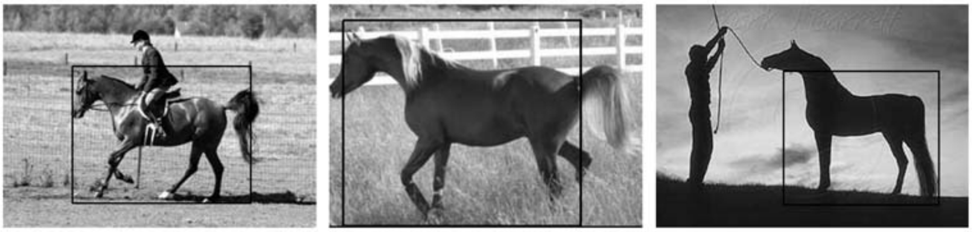


Fig. 23 Example of BFM detections for horses

Table 4 Confusing cows and horses: The first 3 rows show the failures made by the three different models (FP = false positive, FN = false negative, M = multiple detection). The last row shows the RPC-equal-error rate for each model

	Cow-BFM	Horse1-BFM	Horse2-BFM
FP	0	3	0
FN	0	13	12
M	0	1	2
RPC-eq.	0%	23%	19%

Table 5 Results on Weizman horse dataset: This table shows how the performance of the BFM increases if more supervision is used. BFM performs slightly worse than the method of Shotton et al. (2005), but a direct comparison of these two methods is hard (see text for further discussion)

Approach	RPC-equal-error
BFM (bounding box)	18.7
BFM (pre-segmented)	10.8
Shotton et al. (2005)	7.9

images as negative validation images (25 cows). We now apply all three models on the same test set, containing 40 images of cows and 40 images of horses. Figure 23 shows example detection results and Fig. 24 shows some segmentations. Table 4 shows the failures and the RPC-equal error rate of each of these three models on this test set. The cow model is very strong (no failures) because of the low intra-class variability of this category it needs no knowledge of another object class even if its boundary shape is similar. Horse1-BFM is a weaker model (this is a consequence of greater variations of the horses in the training and test images). The model horse2-BFM obviously gains from the cows in the negative validation images, as it does not have any false positive detections. Overall this means our models are good at discriminating classes of similar boundary shapes, but need either more data or more consistent training objects.

For quantitative comparison we trained on 20 horse images, with 30 horses for validation and 30 background images from the Caltech dataset. Tests were performed on 277 other horse images (approx. scale normalized) and 277 Caltech background images as in Shotton et al. (2005). We trained once using just the bounding boxes as supervision and in a second test we used pre-segmented training objects. The results are summarized in Table 5. The results reported by Shotton et al. (2005) used similar conditions. These results also point out that a direct comparison of methods sometimes is quite hard. Shotton et al. use only 10 segmented images, we use 20. But we extract boundary fragments only from 20 training images, using just centroid information from the remaining 30 positive validation images, whereas Shotton et al. extract contour information from all 50 training images. This second point probably explains the slightly better results for their method. Horse shapes show significant intra-class variability so that more shape training data are beneficial.

(e) *Bottles:* To show the advantage of an approach relying on the shape of an object category we set up a new dataset of bottle images. This consists of 118 images collected us-

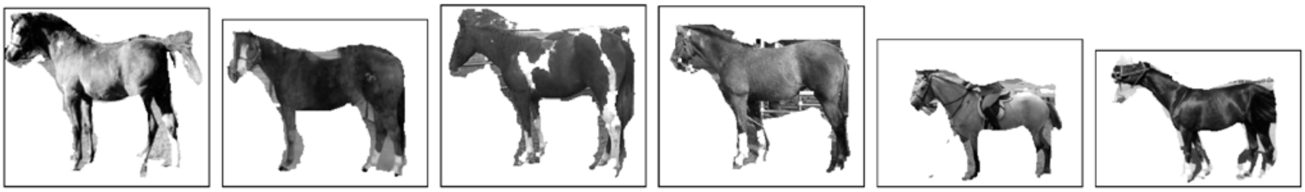


Fig. 24 Segmentation of horses: Example segmentations obtained using the BFM detector on horses

Fig. 25 Example of BFM detections for bottles: The *first row* shows the bounding box of the detection and the *second row* shows the back-projected boundary fragments for these detections. Note the in-plane rotation in the *second column*

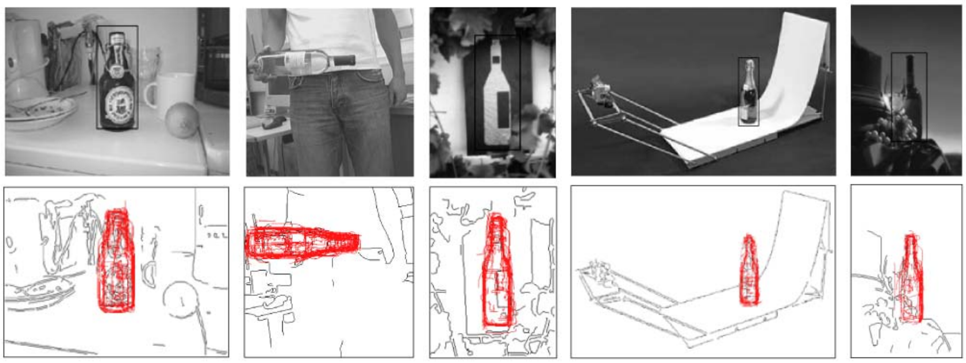
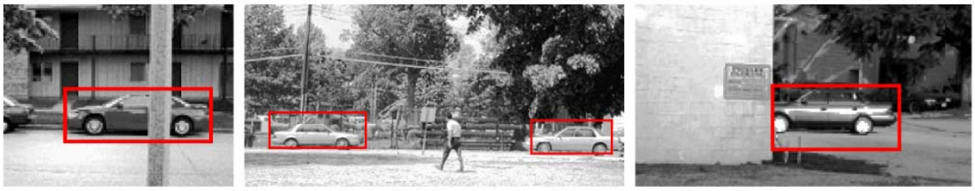


Fig. 26 Examples of BFM detections on the UIUC car dataset



ing Google Image Search. Negative images are provided by the Caltech background image set. We separated the images in the test/training/validation-set (64/24/30) and added the same proportion of negative images in each case. We achieve an RPC-equal error rate of 9%. Figure 25 shows some detection examples.

(f) *UIUC car dataset:* This dataset consists of 549 training and 170 test images. We only used a subset of the available training data (50 training images) and a random selection of 40 test images as a validation set (+ 40 negative training images, no positive training images could be used for validation as they are too simple for validation purpose). On this dataset we again show that our model can learn by only providing the bounding box of the object—though there is a performance penalty: on these low resolution images this noisy training data contains a high number of edges on the background surrounding the training object. It was necessary to use slightly different parameters for this set because of this low image resolution of the training images ($\sigma = 0.2$, $t_{det} = 1$, seeds grown from 20 pixels in steps of 10 to 240 pixels). Table 6 shows our results compared to those reported by Agarwal et al. (2004), Leibe et al. (2004) and Fergus et al. (2003). Note that the results of Leibe et al. (2004) were achieved using pre-segmented cars for training. Shotton et al. (2005) also used some (10) pre-segmented training images. Additionally Leibe et al. (2004) added a final verification procedure that improved their performance after Hough voting. We would also benefit from a similar verification procedure. However, this category points out a drawback of our method based on boundary fragments, namely the need of a sufficient resolution of the training objects.

Table 6 Comparison of the BFM detector to others on the UIUC car database

Method	RPC-equal-err
Agarwal et al. (2004)	21.0%
Fergus et al. (2003)	11.5%
Leibe et al. (2004)	9.0%
Leibe et al. (2004) + Verif.	2.5%
Amores et al. (2005)	10.0%
Shotton et al. (2005)	7.2%
BFM approach	15.0%

7.2 Learning BFMs for Many Categories on the Multi-Class Dataset

(a) *The alphabet:* When we train on 17 categories each of the alphabet entries is on average shared over approximately 5

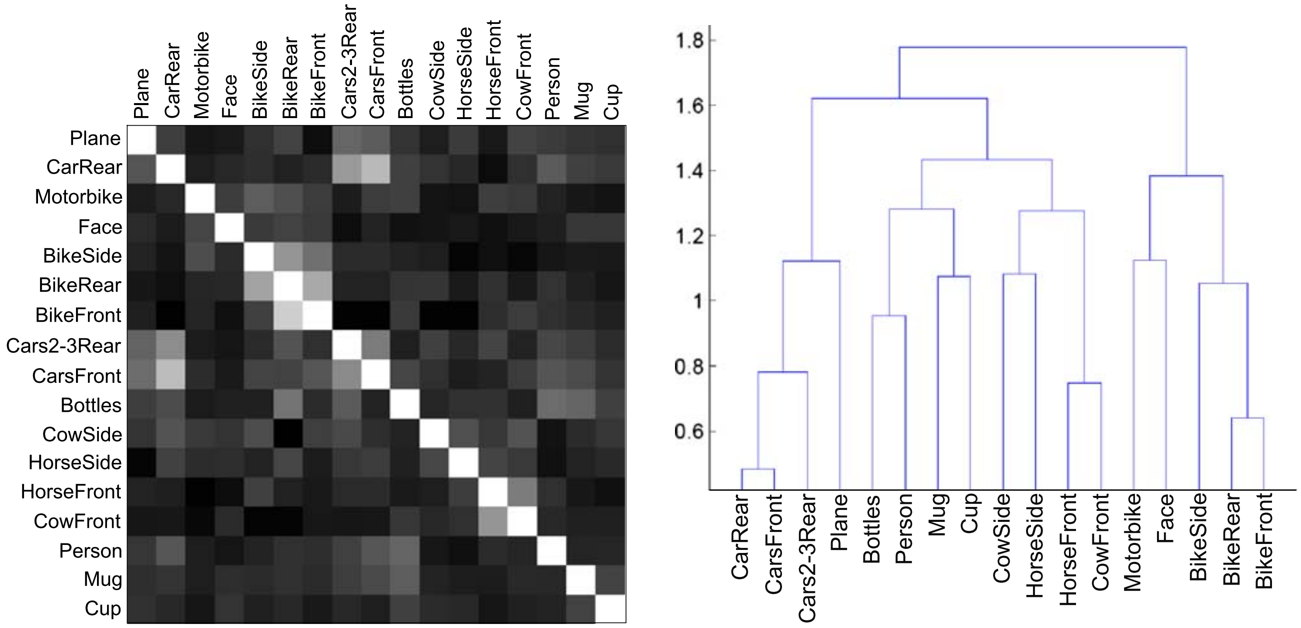


Fig. 27 *Left*: similarity matrix of the alphabet entries of the different categories. *Right*: a dendrogram generated from this similarity matrix

categories. The sublinear growth of the number of alphabet entries with an increasing number of categories can be seen in Fig. 29(a). Further, the alphabet can be used to take a first glance at class similarities. Figures 27(a) and (b) show the alphabet similarities using a similarity matrix and a dendrogram illustration. The correlations visible in the similarity matrix are due to alphabet entries that can be shared over categories. The matrix is calculated by using the alphabet entries for each category specified by the row, and measuring their matching costs when matching them to the training images of the categories. Each column shows the values of the performance on a different category. We further use the row vectors as description vectors, normalize them, and perform agglomerative clustering. This results in the shown dendrogram which directly visualizes category similarities. The dendrogram for the 17 categories shows some intuitive similarities (e.g. for the CarRear and CarFront classes).

Figure 28 shows some examples of the learnt alphabet for these 17 categories. Note how the more basic shapes have more centroid vectors as they occur in different categories at different positions.

(b) *Incremental learning:* Here we investigate our incremental learning at the alphabet level, and on the number of weak detectors used. We compare its sharing abilities to independent and joint learning. A new category can be learnt incrementally, as soon as one or more categories have already been learnt. This saves the effort of a complete retraining procedure, but only the new category will be able to share weak detectors with previously learnt categories, not the other way round. However, with an increasing number of already learnt categories the pool of learnt weak detectors will enlarge and give a good basis to select shareable weak detectors for the new unfamiliar category. We thus can expect a sublinearly growing number of weak detectors when adding categories incrementally. The more similar the categories the more that can be shared. This can be confirmed by a simple experiment where the category HorseSide is incrementally learnt, based on the previous knowledge of an already learnt category CowSide, resulting in 18 shared weak detectors. In comparison, the joint learning shares a total of 32 detectors (CowSide also benefits from HorseSide features). For the 17 categories incremental learning shows its advantage at the alphabet level. We observe (see Fig. 29(a)) that the alphabet requires only 779 entries (worst case approximately 1700 for our choice of the threshold th_K, giving roughly a set of 100 boundary fragments per category).

Figure 29(a) shows the increase in the number of shared weak detectors, as new categories are added incrementally, one category at a time. Assuming we do learn 100 weak detectors per category the number of the worst case (1700) can be reduced to 1116 by incremental learning. Learning all categories jointly reduces the number of used weak detectors even further to 623. However, a major advantage of the incremental approach is the significantly reduced computational complexity. Whilst joint learning with I validation images requires $O(2^C I)$ steps for each weak detector, incremental learning has a complexity of only $O(|h_L|I)$ for those weak classifiers (from already learnt weak classifiers) that can be shared (here $|h_L|$ is the number of already learnt weak detectors, and C is the total number of classes).

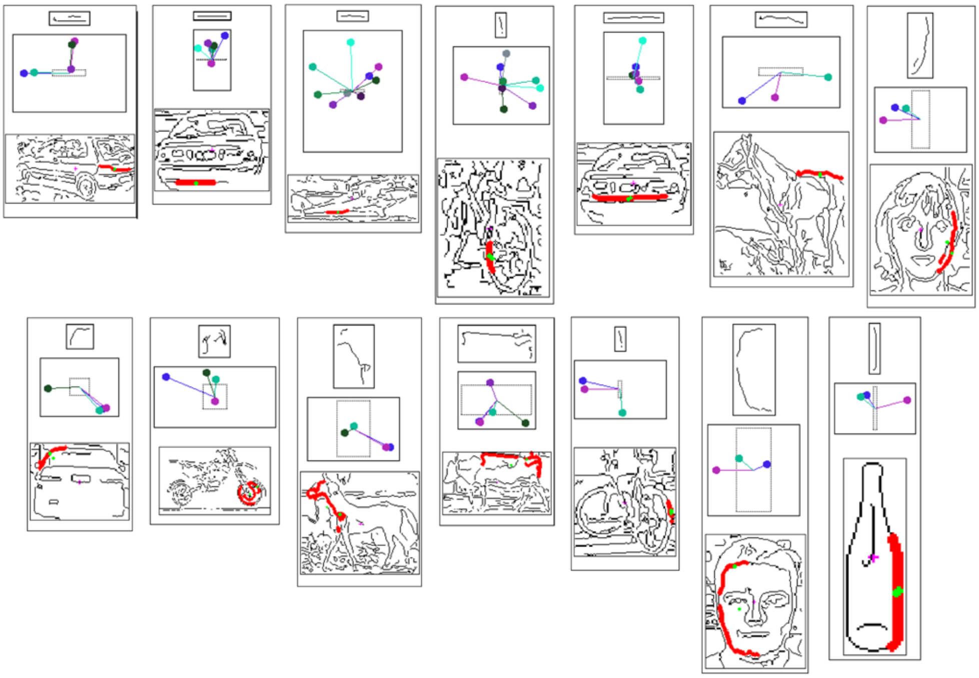


Fig. 28 Examples of the alphabet entries learnt for the multi-class dataset. The *top of each* entry shows the boundary fragment (shape), the *middle* shows the centroid vectors and the *bottom* shows the image this alphabet entry originates from

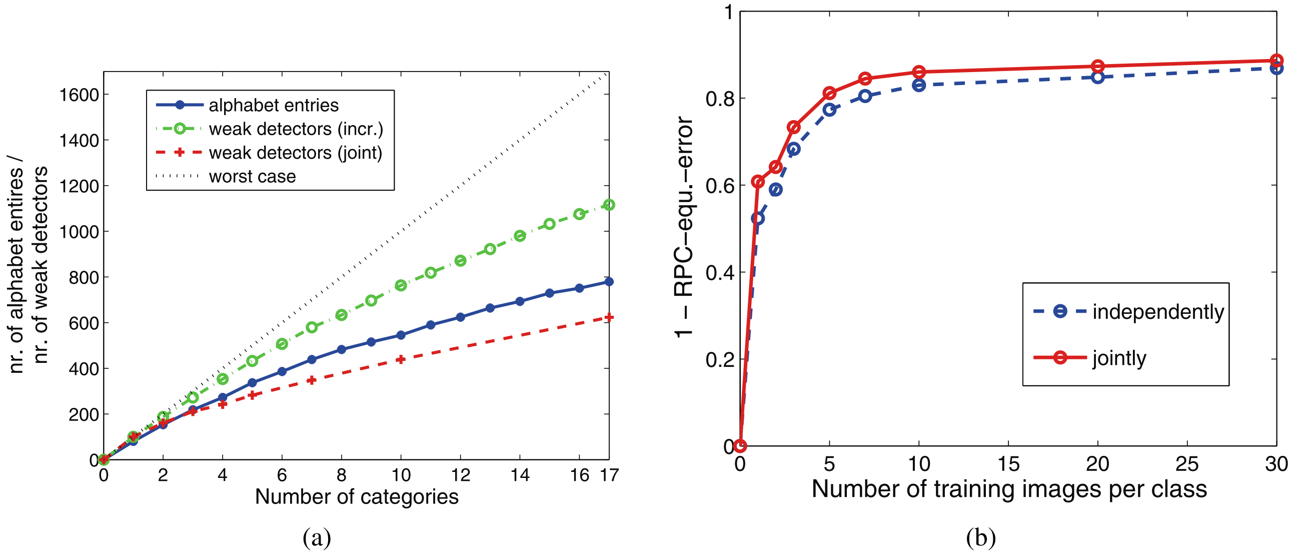


Fig. 29 (**a**) The increase in the number of alphabet entries w.r.t. the number of classes, and increase of the number of weak detectors when adding new classes incrementally or training a set of classes jointly. The values are compared to the worst case (linear growth, *dotted line*). For weak detectors the worst case is training independent and given by ($\sum_{i=1}^{C} T_{c_i}$), and for the alphabet we approximate the worst case by assuming an addition of 100 boundary fragments per category. Classes are taken sequentially (Planes(1), CarRear(2), Motorbike(3), . . .). Note the sublinear growth. (**b**) Error averaged for 6 categories (Planes, CarRear, Motorbike, Face, BikeSide and HorseSide) either learnt independently or jointly with a varying number of training images per category. Note, a large value indicates a smaller error

One could use the information from the dendrogram from Fig. 27(b) to find out the optimal order of the classes for the incremental learning, but this is future work.

(c) *Joint learning:* Here we investigate joint learning for a varying number of classes. First we learn detectors for different aspects of cows, namely the categories CowSide and CowFront independently, and then compare this performance with joint learning. For CowSide the RPC-equal-error is 0% for both cases. For CowFront the error is reduced from 18% (independent learning) to 12% (joint learning). At the same time the number of learnt weak detectors is reduced from 200 to 171. We have carried out a similar compari-

Table 7 Recognition results. In the first row we compare categories to previously published results. We distinguish between detection D (RPC-eq.-err.) and classification C (ROC-eq.err.). Then we compare our model, either trained by the independent method (I) or by the joint (J) method, and tested on the class test set $\mathcal{T}$ or the multiclass test set $\mathcal{M}$. On the multiclass set we count the best detection in an image (over all classes) as the object category. The abbreviations are: B = Bike, H = Horse, Mb = Motorbike, F = Front, R = Rear, S = Side, 23 = two thirds Rear view

Class	Plane	CarR	Mb	Face	B-S	B-R	B-F	Car23	CarF	Bottle	CowS	H-S	H-F	CowF	Pers.	Mug	Cup
Ref.	6.3 (Fergus et al. 2003),C	6.1 (Leibe et al. 2004),D	7.6 (Shotton et al. 2005),D	6.0 (Shotton et al. 2005),D							0.0 (Leibe et al. 2004),D						
I,$\mathcal{T}$	7.4	2.3	4.4	3.6	28.0	25.0	41.7	12.5	10.0	9.0	0.0	8.2	13.8	18.0	47.4	6.7	18.8
J,$\mathcal{T}$	7.4	3.2	3.9	3.7	22.4	20.8	31.3	12.5	7.6	10.7	0.0	7.8	11.5	12.0	42.0	6.7	12.5
I,$\mathcal{M}$	1.1	7.0	6.2	1.4	10.3	7.7	8.5	5.2	7.6	7.1	1.6	10.0	8.2	9.5	29.1	5.1	8.0
J,$\mathcal{M}$	1.5	4.3	4.5	1.6	8.9	5.9	7.7	3.8	8.5	6.1	1.3	11.0	4.7	6.8	27.7	5.8	8.3

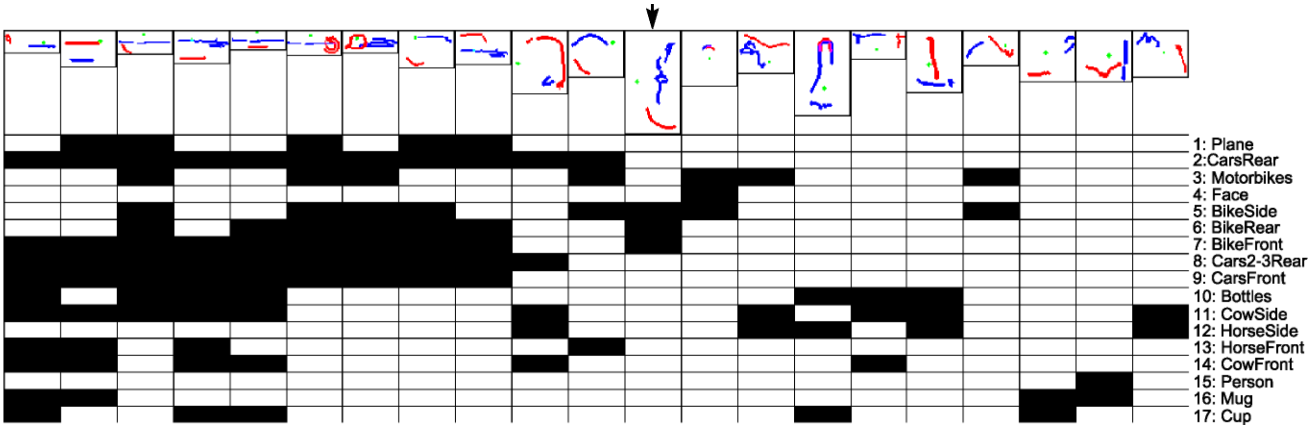


Fig. 30 Examples of weak detectors that have been learnt for the whole dataset (resized to the same width for this illustration). The *black rectangles* indicate which classes share a detector. Rather basic structures are shared over many classes (e.g. *column 2*). Similar classes (e.g. *rows 5, 6, 7*) share more specific weak detectors (e.g. *column 12*, indicated by the arrow, where parts of the bike's wheel are shared)

son for horses which again shows the same behavior. This is due to the reuse of some information gathered from the side aspect images to detect instances from the front. Information that is shared here is e.g. legs, or parts of the head. This is precisely what the algorithm should achieve—fewer weak detectors with the same or a superior performance. The joint algorithm has the opportunity of selecting and *sharing* a weak detector that can separate both classes from the background. This only has to be done once. On the other hand, the independent learning does not have this opportunity, and so has to find such a weak detector for each class.

In Fig. 29(b) we show that joint learning can achieve better performance with less training data as a result of sharing information over several categories (we use 6 categories in this specific experiment).

Finally we focus on many categories, and compare independent learning performance to that achieved by learning jointly. Table 7 shows the detection results on the single category's test set (category images and background images), denoted by $\mathcal{T}$ and on the multiclass test set ($\mathcal{M}$) for both cases. It also gives comparisons to some other methods that used this data in the single category case where we used the same test data. The joint learning procedure does not significantly reduce the detection error (although we gain more than we loose), but we gain in requiring just 623 weak detectors instead of the straightforward 1700 (i.e. 100 times the number of classes for independent learning). Errors are more often because of false positives than false negatives. We are superior or similar in our performance compared to state-of-the-art approaches (note that classification is easier than detection) as shown in Table 7. Looking at the multiclass case (I, $\mathcal{M}$, and J, $\mathcal{M}$, in error per image), we obtain comparable error rates for independent and joint learning.

Figure 30 shows examples of weak detectors learnt in this experiment, and their sharing over various categories. Finally, in Fig. 31 we illustrate some qualitative detection results on various categories of this dataset.

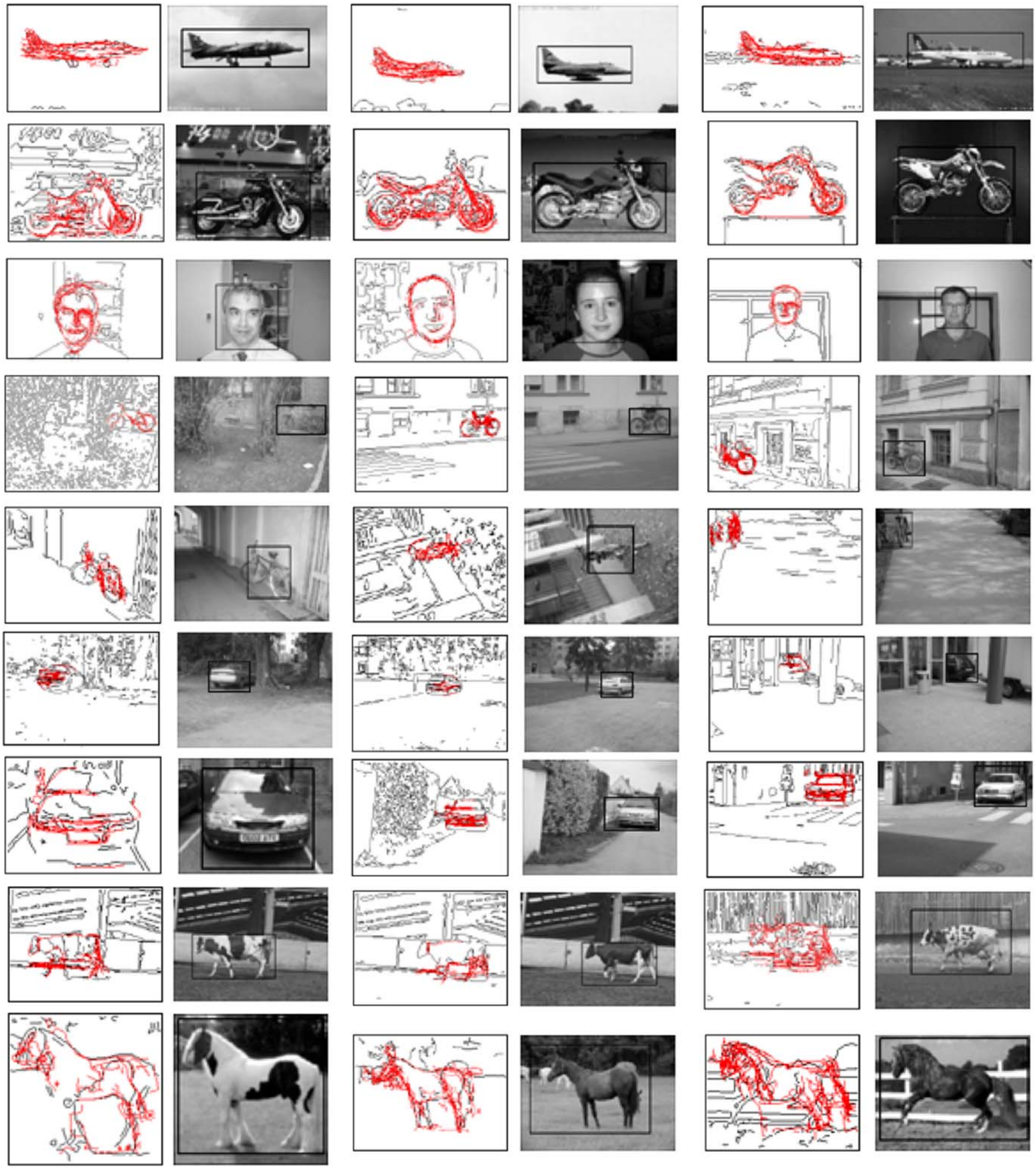


Fig. 31 Examples of detection results using the multi-class detector. On the *left* we show the matched weak-detectors in red and on the *right* the resulting detection result as a bounding box

7.3 Combination of Shape And Appearance Cues

In the overview of our approach, Fig. 1(a) shows that we can use the very same procedure to learn such diverse features as boundary fragments and patches in a unified framework (which we call the "Unified Model", or UM approach below). At detection, this UM can use one combined strong detector and one common Hough voting space. It can be

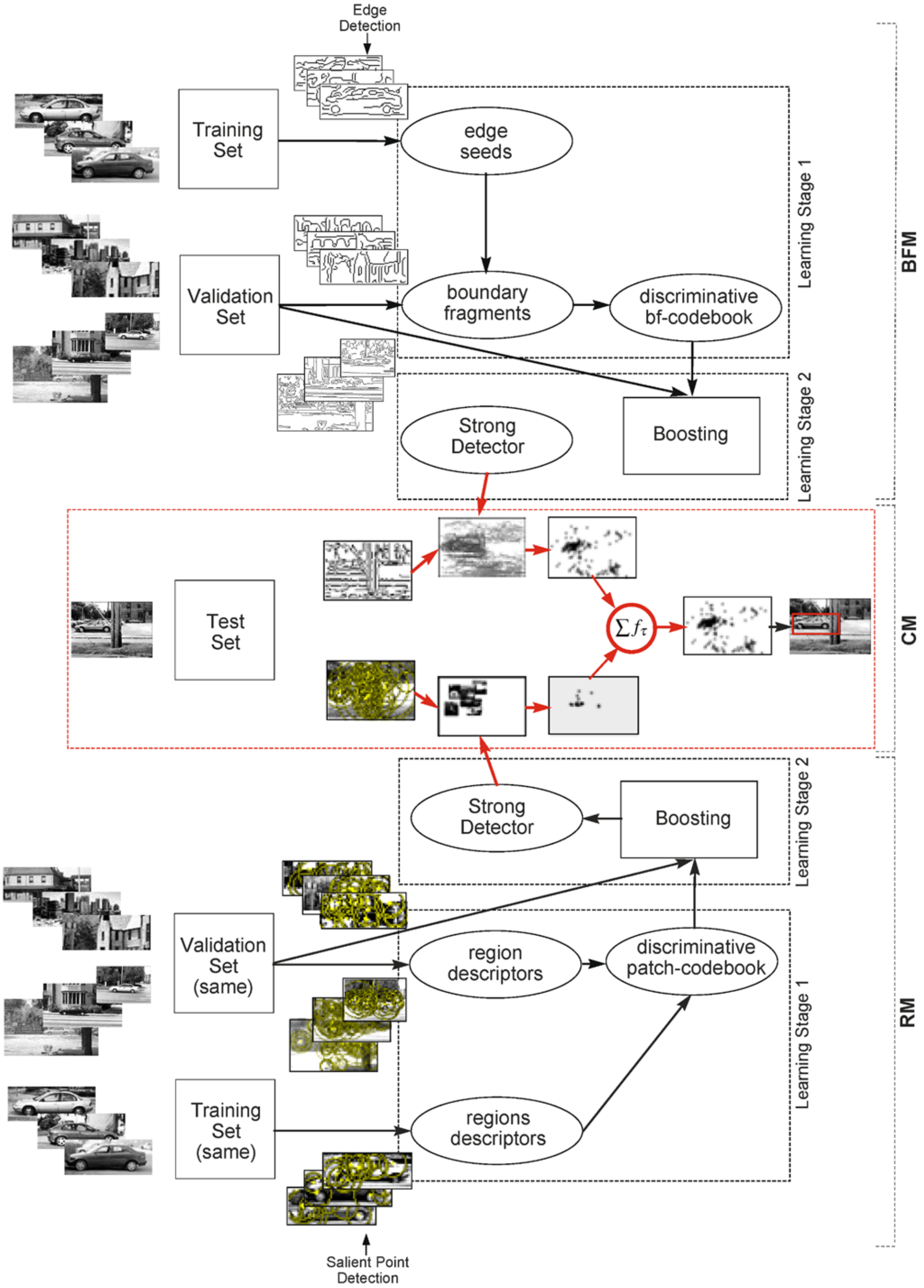


Fig. 32 Intermediate steps towards a unified model—the "Combination of Models" (CM) approach: This CM method is a combination of the BFM (shown on the top) and the "Region Model" (RM) (illustrated on the *bottom*)

expected, that the UM approach will lead to a very flexible selection of class-specific features during learning of the class-specific strong detectors.

To be able to study the effects of shape, appearance, and their combination, we have also implemented two intermediate steps on the way to this UM framework: a "Region Model" and a "Combination Model". Figure 32 shows at the top the BFM for single categories (as explained in detail in Sects. 3 and 4). The BFM learns a strong detector, which at runtime (processing of a test image) casts votes into a BFM Hough voting space. At the bottom of Fig. 32, the very same situation is shown for a purely region-based model ("Region Model" RM). In particular, we extract interest points using the Harris-Laplace combined with the Hessian-Laplace (from http://www.robots.ox.ac.uk/~vgg/software). But our implementation is flexible and could use other techniques, e.g. Affine-Harris, as well. The regions are scale normalized and described by a SIFT descriptor (Lowe 1999). As for the BFM, we first learn an alphabet of appearance, that includes regions which appear in many positive training images, do

Table 8 Comparison of Boundary-Fragment-Model BFM, Region Model RM, Combination of Models CM, and Unified Model UM approach with other state-of-the-art approaches on several categories from the Caltech database and the UIUC cars-side. Note that we report detection results here in terms of RPC-equal-error rates

Cat.	BFM	RM	CM	UM	Leibe et al. (2004)	Shotton et al. (2005)
Cars-side (UIUC)	15.0	10.5	6.2	7.0	9.0	7.2
Cars-rear (Caltech)	2.3	2.9	0.0	0.5	6.1	
Airplanes (Caltech)	7.4	22.5	4.2	13.4	–	–
Motorbikes (Caltech)	4.4	4.0	2.0	3.7	6.0	7.6
Faces (Caltech)	3.6	2.4	1.0	3.2	–	6.0

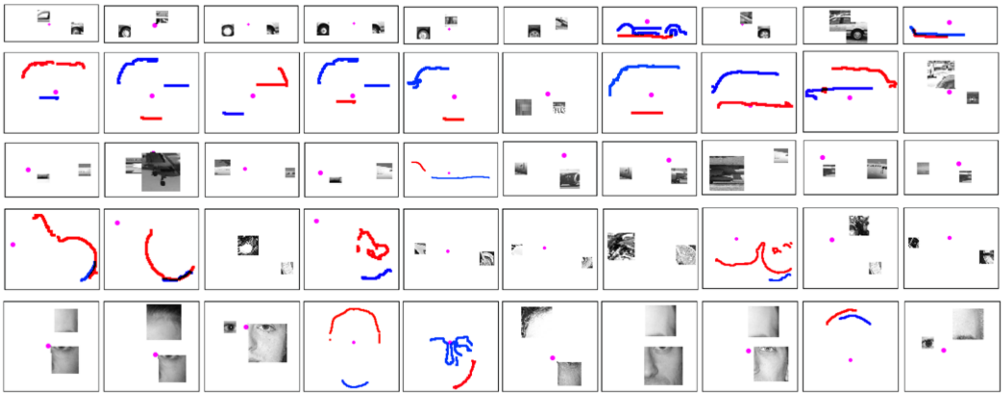


Fig. 33 The first ten weak detectors learnt in the UM for the categories: Cars-side (UIUC), Cars-rear, Airplanes, Motorbikes and Faces (Caltech)

not appear in counter examples, and provide good centroid votes on the positive training images. Next, we combine $k = 2$ alphabet entries to form weak detectors, which provide good centroid votes on the validation images, and finally learn a strong detector from the pool of weak detectors using Boosting. At runtime, the RM detector casts votes into a separate RM Hough voting space. In the middle of Fig. 32, we present the straightforward solution to combining BFM and RM by a simple linear combination of the two Hough voting spaces (with weights f_τ). We term this straightforward combination method CM ("Combination of Models").

Our experiments compare BFM, RM, CM, and UM on a number of test data sets, and provide also a comparison with related work, where available. Table 8 shows the resulting error rates and Fig. 33 shows examples of the learnt weak detectors for each category. In general identical weights f_τ for the different models in the Combined Models approach (CM) are a reasonable choice. More detailed investigation of the weight parameter shows e.g. for motorbikes that the lowest error rate of 1.3% can be achieved at $\mathbf{f} = [0.3, 0.7]$. However, tuning of this parameter requires human supervision as there is often no error on the validation set (which could serve as possibility for automatic tuning) and is thus not always useful. This a disadvantage of the CM compared to the Unified Model (UM).

The CM algorithm is robust to the combination of a reliable with an unreliable model (i.e. one that achieves poor detection results). This is because the method of searching modes by Mean-Shift mode estimation in a Hough space is robust against the addition of a random distribution (the votes of the poor model) and thus the correct modes from the reliable model do not get too distracted by the addition of this second Hough voting space. For the UM algorithm we would expect it to achieve at least the minimum of the error rate that the separate models (RM and BFM) for each feature type achieve. This is generally true. However, in the case of airplanes the UM model achieves poor results. More detailed investigation shows that this is caused by over-fitting on the validation set, whereas restricting the model to only one feature type is sufficient to prevent over-fitting in this case.

8 Conclusion and Discussion

We have presented a unified approach for object category detection which combines shape, appearance and geometry. Starting from our one-class Boundary-Fragment-Model (BFM), where we showed high performance on common datasets, we proposed a multi-class BFM. On the basis of that model we presented algorithms which can learn various categories jointly or can add new categories incrementally.

A clearly sublinear growth in the number of weak detectors with the increasing number of categories gives evidence that shape is well shared amongst different classes. Results on our multi-class dataset show excellent detection performance and little confusion between classes. Finally, the Unified Model (UM) combines shape and appearance, so that we are now able to learn an alphabet of shape and appearance that can be shared over many categories. Experimental results show that selection of features (preference for shape over appearance or vice-versa) is category specific. Furthermore, it is obvious, that object categorization and detection clearly benefits from models like ours, which allows a flexible integration of diverse visual cues.

These results for the combination of boundary fragments with appearance patches show the way for future categorization research. As the field is rapidly developing towards multi-class detection of many categories, models which use just one type of information will certainly lack in their discriminative power. Future work should try to integrate more cues, like color, texture and segments.

There remain several hard problems for future research work. The required amount of supervision should be dramatically reduced to make the learning of such category models more practical. What can be done to learn centroid votes without providing centroids in training and validation data? Treatment of scale is much harder for boundary fragments than for patches (where affine covariant detectors and affine invariant descriptors have reached a certain maturity).

Acknowledgements This work was supported by the Austrian Science Fund FWF, project S9103, Pascal Network of Excellence and EC Project CLASS.

References

Agarwal, S., Awan, A., & Roth, D. (2004). Learning to detect objects in images via a sparse, part-based representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(11), 1475–1490.

Amit, Y., German, D., & Fan, X. (2004). A coarse-to-fine strategy for multi-class shape detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*, 1606–1621.

Amores, J., Sebe, N., & Radeva, P. (2005). Fast spatial pattern discovery integrating boosting with constellations of contextual descriptors. In *Proceedings of the CVPR* (Vol. 2, pp. 769–774), CA, USA, June 2005.

Bar-Hillel, A., Hertz, T., & Weinshall, D. (2005). Object class recognition by boosting a part-based model. In *Proceedings of the CVPR* (Vol. 1, pp. 702–709), June 2005.

Bart, E., & Ullman, S. (2005). Cross-generalization:learning novel classes from a single example by feature replacement. In *Proceedings of the CVPR* (Vol. 1, pp. 672–679).

Bernstein, E. J., & Amit, Y. (2005). Part-based statistical models for object classification and detection. In *Proceedings of the CVPR* (Vol. 2, pp. 734–740).

Borgefors, G. (1988). Hierarchical chamfer matching: a parametric edge matching algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *10*(6), 849–865.

Breu, H., Gil, J., Kirkpatrick, D., & Werman, M. (1995). Linear time Euclidean distance transform algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(5), 529–533.

Caputo, B., Wallraven, C., & Nilsback, M. E. (2004). Object categorization via local kernels. In *Proceedings of the ICPR* (Vol. 2, pp. 132–135).

Comaniciu, D., & Meer, P. (2002). Mean shift: a robust approach towards feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *24*(5), 603–619.

Crandall, D., Felzenszwalb, P., & Huttenlocher, D. (2005). Spatial priors for part-based recognition using statistical models. In *Proceedings of the CVPR* (pp. 10–17).

Csurka, G., Bray, C., Dance, C., & Fan, L. (2004). Visual categorization with bags of keypoints. In *ECCV'04: workshop on statistical learning in computer vision* (pp. 59–74).

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proceedings of the CVPR* (Vol. 1, pp. 886–893).

Deselaers, T., Keysers, D., & Ney, H. (2005). Discriminative training for object recognition using images patches. In *Proceedings of the CVPR* (Vol. 2, pp. 157–162).

Epstein, B., & Ullman, S. (2005). Feature hierarchies for object classification. In *Proceedings of the ICCV* (Vol. 1, pp. 220–227).

Everingham, M., Zisserman, A., Williams, C., Van Gool, L., Allan, M., Bishop, C., Chapelle, O., Dalal, N., Deselaers, T., Dorko, G., Duffner, S., Eichhorn, J., Farquhar, J., Fritz, M., Garcia, C., Griffiths, T., Jurie, F., Keysers, D., Koskela, M., Laaksonen, J., Larlus, D., Leibe, B., Meng, H., Ney, H., Schiele, B., Schmid, C., Seemann, E., Shawe-Taylor, J., Storkey, A., Szedmak, S., Triggs, B., Ulusoy, I., Viitaniemi, V., & Zhang, J. (2005). The 2005 pascal visual object classes challenge. In *Lecture notes in artificial intelligence*. *Selected proceedings of the first PASCAL challenges workshop*. Berlin: Springer.

Fan, X. (2005). Efficient multiclass object detection by a hierarchy of classifiers. In *Proceedings of the CVPR* (Vol. 1, pp. 716–723).

Fei-Fei, L., Fergus, R., & Perona, P. (2004). Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In *Proceedings of the CVPR workshop on generative-model based vision*.

Felzenszwalb, P., & Huttenlocher, D. (2004). Pictorial structures for object recognition. *International Journal of Computer Vision*, *61*(1), 55–79.

Fergus, R., Perona, P., & Zisserman, A. (2003). Object class recognition by unsupervised scale-invariant learning. In *Proceedings of the CVPR* (pp. 264–271).

Fergus, R., Perona, P., & Zisserman, A. (2004). A visual category filter for Google images. In *Proceedings of the ECCV* (pp. 242–256).

Fergus, R., Perona, P., & Zisserman, A. (2005). A sparse object category model for efficient learning and exhaustive recognition. In *Proceedings of the CVPR* (Vol. 1, pp. 380–387).

Fergus, R., Perona, P., & Zisserman, A. (2007). Weakly supervised scale-invariant learning of models for visual recognition. *International Journal of Computer Vision*, *71*(3), 273–303.

Ferrari, V., Tuytelaars, T., & Van Gool, L. (2004). Simultaneous object recognition and segmentation by image exploration. In *Proceedings of the ECCV* (pp. 40–54).

Ferrari, V., Tuytelaars, T., & Van Gool, L. (2006). Object detection by contour segment networks. In *Proceedings of the ECCV* (Vol. 3, pp. 14–28).

Freund, Y., & Schapire, R. (1997). A decision theoretic generalisation of online learning. *Computer and System Sciences*, *55*(1), 119–139.

Friedman, J., Hastie, T., & Tibshirani, R. (1998). *Additive logistic regression: a statistical view of boosting* (Technical report). Stanford University, Department of Statistics, California 94305.

Gavrila, D. M., & Philomin, V. (1999). Real-time object detection for smart vehicles. In *Proceedings of the ICCV* (pp. 87–93).

Jurie, F., & Schmid, C. (2004). Scale-invariant shape features for recognition of object categories. In *Proceedings of conference on vision and pattern recognition* (pp. 90–96).

Kumar, M. P., Torr, P. H. S., & Zisserman, A. (2004). Extending pictural structures for object recognition. In *Proceedings of the BMVC*.

Leibe, B., Leonardis, A., & Schiele, B. (2004). Combined object categorization and segmentation with an implicit shape model. In *ECCV'04: workshop on statistical learning in computer vision* (pp. 17–32), May 2004.

Leibe, B., & Schiele, B. (2004). Scale-invariant object categorization using a scale-adaptive means-shift search. In *DAGM'04* (pp. 145–153), August 2004.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of the ICCV* (pp. 1150–1157).

Magee, D., & Boyle, R. (2002). Detection of lameness using resampling condensation and multi-steam cyclic hidden Markov models. *Image and Vision Computing*, *20*(8), 581–594.

Marszalek, M., & Schmid, C. (2006). Spatial weighting for bag-of-features. In *Proceedings of the CVPR*.

Mikolajczyk, K., Leibe, B., & Schiele, B. (2006). Multiple object class detection with a generative model. In *Proceedings of the CVPR*.

Mutch, J., & Lowe, D. (2006). Multiclass object recognition with sparse, localized features. In *Proceedings of the CVPR*.

Nistér, D., & Stewénius, H. (2006). Scalable recognition with a vocabulary tree. In *Proceedings of the CVPR*.

Ommer, B., & Buhmann, J. M. (2006). Learning compositional categorization models. In *Proceedings of the ECCV* (Vol. 3, pp. 316–329).

Opelt, A., Fussenegger, M., Pinz, A., & Auer, P. (2004). Weak hypotheses and boosting for generic object detection and recognition. In *Proceedings of the ECCV* (pp. 71–84).

Opelt, A., Fussenegger, M., Pinz, A., & Auer, P. (2006a). Generic object recognition with boosting. *Pattern Analysis and Machine Intelligence*, *28*(3).

Opelt, A., Pinz, A., & Zisserman, A. (2006b). Incremental learning of object detectors using a visual shape alphabet. In *Proceedings of the CVPR* (Vol. 1, pp. 3–10), June 2006.

Opelt, A., Pinz, A., & Zisserman, A. (2006c). A boundary-fragment-model for object detection. In *Proceedings of the ECCV* (Vol. 2, pp. 575–588), May 2006.

Opelt, A., Pinz, A., & Zisserman, A. (2006d). Fusing shape and appearance information for category detection. In *Proceedings of the BMVC* (Vol. 1, pp. 117–126), September 2006.

Quinn, P. C., Eimas, P. D., & Tarr, M. J. (2001). Perceptual categorization of cat and dog silhouettes by 3-to-4 month old infants. *Journal of Experimental Child Psychology*, *79*(1), 78–94.

Sali, E., & Ullman, S. (1999). Combining class-specific fragments for object classification. In *Proceedings of the BMVC* (Vol. 1, pp. 203–213).

Seemann, E., Leibe, B., & Schiele, B. (2006). Multi-aspect detection of articulated objects. In *Proceedings of the CVPR*.

Serre, T., Wolf, L., & Poggio, T. (2005). A new biologically motivated framework for robust object recognition. In *Proceedings of the CVPR*.

Shotton, J., Blake, A., & Cipolla, R. (2005). Contour-based learning for object detection. In *Proceedings of the ICCV* (Vol. 1, pp. 503–510).

Shotton, J., Winn, J., Rother, C., & Criminisi, A. (2006). TextonBoost: Joint appearance, shape and context modeling fdor multi-class object recognition and segmentation. In *Proceedings of the ECCV* (Vol. 1, pp. 1–15), May 2006.

Sivic, J., & Zisserman, A. (2003). Video Google: a text retrieval approach to object matching in videos. In *Proceedings of the ICCV*.

Sivic, J., Russell, B., Efros, A., Zisserman, A., & Freeman, W. (2005). Discovering objects and their location in images. In *Proceedings of the ICCV*.

Thomas, A., Ferrari, V., Leibe, B., Tuytelaars, T., Schiele, B., & Van-Gool, L. (2006). Towards multi-view object detection. In *Proceedings of the CVPR*.

Thureson, J., & Carlsson, S. (2004). Appearance based qualitative image description for object class recognition. In *Proceedings of the ECCV* (pp. 518–529).

Torralba, A., Murphy, K. P., & Freeman, W. T. (2004). Sharing features: efficient boosting procedures for multiclass object detection. In *Proceedings of the CVPR*.

Tu, Z. (2005). Probabilistic boosting-tree: learning discriminative models for classification, recognition, and clustering. In *Proceedings of the CVPR* (pp. 1589–1596).

Vidal-Naquet, M., & Ullman, S. (2003). Object recognition with informative features and linear classification. In *Proceedings of the ICCV* (Vol. 1, pp. 281–288).

Wang, G., Zhang, Y., & FeiFei, L. (2006). Using dependent regions for object categorization in a generative framework. In *Proceedings of the CVPR*.

Williams, C. K. I., & Allan, M. (2006). *On a connection between object localization with a generative template of features and pose-space prediction methods* (Technical Report EDI-INF-RR-0719). School of Informatics, University of Edinburgh.

Winn, J., Criminisi, A., & Minka, T. (2005). Object categorization by learning universal visual dictionary. In *Proceedings of the ICCV* (pp. 1800–1807).

Zhang, W., Yu, B., Zelinsky, G. J., & Samaras, D. (2005). Object class recognition using multiple layer boosting with heterogenous features. In *Proceedings of the CVPR* (pp. 66–73).

Zhang, J., Marszalek, M., Lazebnik, S., & Schmid, C. (2007). Local features and kernels for classification of texture and object categories: a comprehensive study. *International Journal of Computer Vision*, *73*(2), 213–238.