Multiple Object Class Detection with a Generative Model

Krystian Mikolajczyk
University of Surrey
Guildford, UK
K.Mikolajczyk@surrey.ac.uk

Bastian Leibe
ETH
Zurich, Switzerland
leibe@vision.ee.ethz.ch

Bernt Schiele
TU-Darmstadt
Darmstadt, Germany
schiele@cs.tu-darmstadt.de

Abstract

In this paper we propose an approach capable of simultaneous recognition and localization of multiple object classes using a generative model. A novel hierarchical representation allows to represent individual images as well as various objects classes in a single, scale and rotation invariant model. The recognition method is based on a codebook representation where appearance clusters built from edge based features are shared among several object classes. A probabilistic model allows for reliable detection of various objects in the same image. The approach is highly efficient due to fast clustering and matching methods capable of dealing with millions of high dimensional features. The system shows excellent performance on several object categories over a wide range of scales, in-plane rotations, background clutter, and partial occlusions. The performance of the proposed multi-object class detection approach is competitive to state of the art approaches dedicated to a single object class recognition problem.

1. Introduction

Single object class detection has now reached some maturity in computer vision, and research interests have started to concentrate on more generic approaches capable of dealing with many object classes. However, the performance level of these methods is still far from the results obtained for single classes. Many of them can only classify images in presence/absence tasks without reliable object localization in challenging viewing conditions due to design of these approaches [4, 10] or other reasons *e.g*. noise from unsupervised training data [22], insufficient support from image measurements [8, 19], inefficient recognition technique to deal with large space of possible object sizes, poses, and positions, etc. Furthermore, performance of those multi-class approaches that can localize objects has not been reported for challenging data used to test single class detectors [8, 23, 24].

The purpose of this work is to design and investigate a novel approach based on local features and Bayesian decision rules. In this paper, we adopt the following design guidelines : fast and dense sampling of scale invariant features, effective object class representation, efficient and reliable training and recognition. We also require the approach to be capable of dealing simultaneously with many object classes while obtaining recognition results comparable with state of the art detectors. To fulfill these requirements we carefully design every crucial stage of the method while considering resent feature and clustering evaluations [16] as well as desired efficiency. For this we are pooling from many ideas reported in the literature [4, 6, 7, 8, 12, 22, 23, 25]. We use a generative approach rather than adapting one of the discriminative methods based on powerful machine learning techniques. Generative models are easier to extend to deal with multiple object classes and to update with new training data [7]. On the other hand the generalization properties often result in high recall but low recognition precision. However, we will demonstrate that a generative method is capable of competing with state of the art discriminative techniques [1, 5, 21] as well as with combinations of generative and discriminative methods [9].

Approaches based on feature detectors seem to dominate in multi-class recognition. Many approaches tend to use more and more local features applying simultaneously several detectors [22, 23], sampling features at edges [2, 15, 21] and approaching an extreme where interest regions are sampled from all image points at multiple scales [10]. This is due to the fact that insufficient support from features results in lower recognition performance as observed in [8]. We argue that dense sampling can be beneficial but it is sufficient to focus on image points where the signal changes. We therefore use scale invariant features densely but efficiently computed from edge points.

Many successful approaches are based on appearance clusters called visual vocabulary, codebook, or keywords constructed with different clustering methods *e.g*. agglomerative [1, 12], partitional k-means [22, 25] or other techniques [10]. K-means is often used for its low complex-

ity, thus allowing to deal with many features. However, we show that a combination of k-means and agglomerative clustering can benefit from the efficiency of the first one and results in the clustering quality of the second one. We therefore base our object representation on appearance clusters and a hierarchical structure of the data provided by partitional-agglomerative clustering.

There are several possibilities to represent object classes. A star shape model [11, 21] can be easily trained and evaluated in contrast to the constellation model [8] or complex graphical models [23]. It allows to use as many parts as required, since the complexity scales linearly. Moreover, this model is flexible enough to deal with large variations in object shape and appearance of rigid and articulated structures. Finally, it can be also extended to rotation invariance, which we show in this paper.

It has been demonstrated, that sharing features [24] and object parts [23] is a promising direction towards handling many classes without increase in the size of representation and complexity. We therefore follow this idea in our approach.

Our main contribution is an efficient object recognition system capable of recognizing and localizing simultaneously several object classes. We first propose a novel hierarchical representation capable of modeling several object classes as well as representing individual images. The representation is based on joint appearance-geometry probability distributions. The hierarchical data structure is efficiently constructed during learning and allows for efficient object detection, which we consider as an important contribution. The representation is fully invariant to in-plane rotations. Furthermore it allows to recover the rotation angle of the recognized objects. Finally, our probabilistic recognition strategy allows to deal with possible overlaps between objects as well as ambiguities arising between similar object classes.

This paper is organized as follows. Section 2 describes our recognition approach including feature detection, object representation, recognition and learning of the model. Section 3 gives the implementation details that lead to high efficiency of training and recognition. Finally, section 4 presents the evaluation results on challenging test data.

2. Object class representation and recognition

In this section we describe our recognition approach. We first present the features and the model used to represent the object classes. Next we explain the recognition and learning procedures of the model.

2.1. Features

We base our approach on scale invariant features densely sampled from edges. To extract features from images we apply the method proposed in [15] which combines the Canny edge detection with Laplacian-based automatic scale selection [13]. This approach provides position, scale and dominant orientation of a local structure in a neighborhood of edge points. The region is then described with 128 dimensional SIFT [14]. The descriptor represents local appearances of the interest regions. The descriptor is computed with respect to the dominant orientation, thus being rotation invariant. Finally, we reduce the number of dimensions to 40 with PCA. Besides the gain in efficiency, the dimensionality reduction also improves generalization properties of these features.

Rotation invariance. In [1, 7, 8, 11] the positions of features or parts are related to the reference frame or object center in Cartesian coordinates. This representation is convenient but restricted to one pose of the object. We propose to modify this model and express the positions of features in polar coordinates, in which the object center is related to local coordinates of features. The dominant orientation of the feature is used as a reference angle. Figure 1(a) illustrates the rotation invariant representation. There are 4 parameters used to represent the geometric information of a feature: d–distance to the object center, ϕ–angle between the dominant gradient orientation and the vector between object center and feature location, σ–scale (size), and θ–dominant gradient orientation. The first 3 parameters (d, ϕ, σ) are independent of the global rotation angle. Parameter θ is related to the global coordinates of the image. Given a query feature we can use its angle and scale as well as the corresponding geometric parameters of the object model to hypothesize the object center and scale as shown in figure 1(b). The fourth parameter θ is used to recover the rotation angle if required (cf. section 3.4). This representation allows for full rotation invariance using a three dimensional space of object position and scale, which reduces the search complexity.

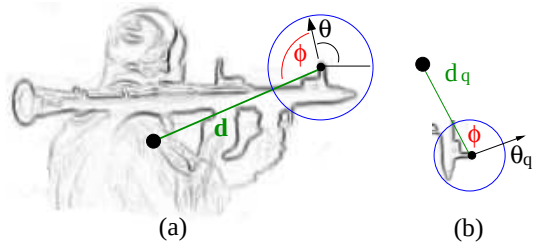
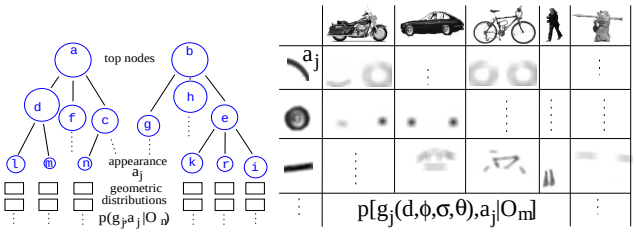


Figure 1. (a) Rotation invariant representation in polar coordinates with respect to the object center. (b) Query feature and hypothesized object center. $\theta_q - \theta$ is the rotation of the object.

2.2. Hierarchical codebook

Tree structure. We represent all features extracted from object classes in a single tree structure. The features are clustered with a method that produces a hierarchical tree of clusters. Figure 2(a) illustrates the tree representation. Appearance clusters a_j formed by similar features, are at the

first level of the tree (bottom nodes). Nodes at higher levels group the closest clusters together. Each node is therefore a hyperball in the feature space represented by a centroid and a radius. The centroid is a mean of all features within a ball and the radius is a distance from the center to the furthest feature in the cluster. The radii of nodes increase towards the top of the tree. Each of the appearance clusters has several geometric distributions $g_j(d, \phi, \sigma, \theta)$ corresponding to the object classes. The distributions contain information about the geometric relations between the object center and local appearance. Figure 2(b) shows appearance clusters and their distributions for objects. All object classes are therefore represented in one tree by a set of appearance clusters and geometric distributions.



(a) (b)

Figure 2. (a) Hierarchical structure. (b) Codebook representation. Appearance clusters (left column) and their geometric distributions for different object classes. For visualization the distributions are in 2D Cartesian coordinate system.

Building the tree. To build the tree we use a metric (Euclidean distance) to group the appearance clusters that lie in a hyperball of a given radius regardless of the object class or part they belong to. To build the tree structure we apply agglomerative clustering. This bottom-up method starts with the number of clusters equal to the number of features and merges the two closest clusters at each iteration. We record the indices of merged clusters and the similarity distance at which the clusters are merged at each iteration. The procedure continues until the last two clusters are merged. The resulting clustering trace is then used to construct the tree. The only parameter to provide is the size of the bottom nodes (radius of appearance clusters) and the number of tree levels. The radii for intermediate levels are equally distributed between the bottom node radius and the radius of the top node. These radii are only indicators for maximum cluster size at a given level. If there are no clusters between two levels that can be merged within the prescribed radius, the level is not created. Therefore, the actual number of levels and the number of nodes depends on the distribution of data points (features).

Discussion. This tree representation has several advantages. The appearance clusters are shared within one image as well as among different classes and object parts. Despite massive amounts of features extracted from images the representation is compact. The tree can be used to represent features extracted from individual images as well as to represent all object classes. The difference is that a single image representation contains only one geometric distribution per cluster, and the object class representation contains as many geometric distribution as there are categories in the model. This representation can be also seen as a metric tree similar to those used in fast data structure algorithms [17] thus allowing for efficient search when matching features to the appearance clusters (cf. section 3.3). Finally, one can also interpret it as a tree of classifiers. A classifier (node) gives a positive response if a query feature lies within the radius of the node, negative otherwise.

2.3. Recognition

The approach relies on robustly estimated probability distributions of the model parameters. We apply Bayesian rules and use a formulation similar to [8]. Given features F detected in a query image, appearance clusters A, and geometric distributions G we make a decision :

$$\frac{p(O_m|G,A)}{p(B|G,A)} = \frac{p(G,A|O_m)p(O_m)}{p(G,A|B)p(B)} \qquad (1)$$

$p(O_m)$ and $p(B)$ are priors of an object and background. In the following we explain only the object related terms but the background terms are estimated in a similar way. The likelihood $p(G, A|O_m)$ is given by

$$p(G,A|O_m) = \prod_i \sum_j p(g_j, a_j|f_i, O_m)p(f_i|O_m) \qquad (2)$$

where $p(f_i|O_m)$ is a feature probability and $p(g_j, a_j|O_m)$ is a joint appearance-geometry distribution of each cluster. To simplify the notation we use g_j instead of $g_j(d, \phi, \sigma)$. Each feature likelihood is then modeled by a mixture of distributions $p(g_j, a_j|f_i, O_m)$ from appearance clusters a_j which match to query feature f_i.

To find initial hypotheses for object locations we search for local maxima of the likelihood function given by equation 1. To improve detection precision overlapping bounding boxes of detected objects are frequently removed. However, we cannot follow this strategy since we want to deal with multiple objects which can partially occlude each other, that is the hypotheses can overlap. We allow several objects to be simultaneously present at similar positions and scales. However, we impose a condition that each feature can contribute only to one hypothesis, since it is very unlikely that similar parts belonging to different objects are visible at the same position in the image. We therefore remove from the weaker hypothesis H, all $p(g_j, a_j|f_i, O_n)$ that contribute to both overlapping hypotheses:

$$p(G,A|O_m) = p(G_{H\setminus S}, A_{H\setminus S}|O_m) \qquad (3)$$

where $S \subset H$ is a set of appearance-geometry likelihoods that also contribute to another stronger hypothesis. The weaker hypothesis can be still accepted if it has enough support from other features.

A problem occurs when there are similar objects in the model, which share many appearance clusters, and have similar location distribution *e.g.* bicycles and motorbikes. The probabilities $p(g_j, a_j|bicycle)$ and $p(g_j, a_j|motorbike)$ are comparable in shared clusters. The likelihoods for these classes are often dominated by shared clusters *e.g.* wheels on different vehicles. Therefore we cannot rely on comparing the scores. To deal with such cases we use an average confusion factor between every pair of objects estimated during learning :

$$conf(O_m, O_n) = \frac{1}{|A|} \sum_j \frac{p(g_j, a_j|O_m)}{p(g_j, a_j|O_n)} \qquad (4)$$

where n and m are switched during summing such that $p(g_j, a_j|O_m) < p(g_j, a_j|O_n)$. This factor is approaching 1 if two classes have similar distributions. If the overlap between two hypotheses as well as the confusion prior are significant, we remove from both hypotheses all the contributions that come from the shared clusters and compare the remaining score to decide which object is present.

2.4. Learning

The model parameters are joint probability distributions $p(g_j, a_j|O_m)$ for individual clusters. In the training stage we separate them in two terms $p(g_j|a_j, O_m)p(a_j|O_m)$. The first term is a geometric distribution of an appearance cluster a_j for a given object O_m and the second term is a likelihood that the appearance a_j represents object O_m. To estimate the model $p(a_j|O_m)$, $p(a_j|B)$, $p(g_j|a_j, O_m)$, and $p(g_j|a_j, B)$ we first extract features F from all labeled training examples. We then build the appearance clusters and match the features back to the cluster centers. Matches are considered only within a threshold β. Each feature that matches to a_j contributes to the probability estimate for a_j and to its geometric distribution g_j at the position, scale and orientation given by the feature. The similarity between appearance cluster a_j and feature f_i is evaluated as $p(a_j|f_i) = \frac{1}{Z_1} \exp(-\frac{\|a_j - f_i\|^2}{\beta})$. The probability for a given appearance cluster a_j is therefore :

$$p(a_j|O_m) = \sum_i p(a_j|f_i, O_m)p(f_i|O_m) \qquad (5)$$

Z_1 is chosen such that $\sum_j p(a_j|O_m) = 1$. For comparison, in a threshold based matching $p(a_j|f_i)$ would be binary and $p(a_j|O_m)$ would be a ratio of the number of features that match to cluster a_j to the total number of matches. The geometric distribution $p(g_j|a_j, O_m)$ is estimated simultaneously with $p(a_j|O_m)$ for each appearance cluster and all classes :

$$p(g_j|a_j, O_m) = \frac{1}{Z_2} \sum_i p(g_j|f_i, a_j, O_m)p(f_i|O_m) \qquad (6)$$

Z_2 is chosen such that $\sum_j p(g_j|a_j, O_m) = 1$. Background probabilities $p(a_j|B)$ are estimated in a similar way, by matching background features to the appearance clusters. We assume that $p(g_j|a_j, B)$ is uniform, which is true for sufficiently large training data. Finally, each appearance cluster has a set of $p(a_j|O_m)$ values and distributions $p(g_j|a_j, O_m)$ for all objects O_m as well as for the background $p(a_j|B)$.

3. Efficient implementation

One of the advantages of this approach is its efficiency. In the following we give the implementation details for crucial stages of our learning and recognition system. We first describe the feature detector, clustering method and fast matching technique and then we give the details of the training and recognition stage.

3.1. Feature detector

To efficiently compute features we build on recent advances from [14] and [15]. Given an input image we first build a scale-space pyramid with a Gaussian kernel and sampling interval of 1.2. Next, for each scale level, we compute gradient magnitude and phase, as well as the Laplacian. Gradient magnitude and phase are used to compute Canny edges. For each edge point we estimate a characteristic scale based on the Laplacian extrema in a way similar to [15]. To do so we search for the values at the corresponding positions and scale levels in the Laplacian pyramid. Given the point position and the scale level of the Laplacian extremum we identify a region of interest at the corresponding position in the gradient-orientation pyramids. Following [14] we first estimate all the dominant orientations within the region, and compute a 128 dimensional SIFT descriptor for each of the orientations. This increases the robustness for circular structures with many dominant orientations i.e., wheels of cars, motorbikes, bicycles etc. All those operations are fast since we only collect samples from the precomputed pyramids. Finally, the dimensionality of descriptors is reduced with PCA to 40.

3.2. Clustering

To build the hierarchical structure of appearance clusters we use a combined top-down-bottom-up clustering method. The top-down strategy is efficient in partitioning the space but sensitive to outliers. Bottom-up method provides better clustering solution but it is computationally expensive. To overcome the complexity problems we apply k-means to initially divide the feature space into few partitions C. The bottom-up approach based on average link agglomerative clustering is then used to obtain compact feature clusters

within each partition. It is important to notice that standard average link is not applicable to large data sets due to its $O(N^2 \log(N))$ time and $O(N^2)$ space complexity. We use reciprocal nearest neighbor (RNN) search [12] to reduce the complexity to $O(N^2)$ time and $O(N)$ space. RNN average link is therefore applied to features in each k-means partition with a similarity threshold, which determines the maximum cluster size. This threshold is empirically set to half the size of the appearance clusters (codebook). We then collect the resulting clusters Q from all C partitions and perform agglomerative clustering on their centroids until the last two clusters are merged. Using a smaller initial cluster size prevents from dividing appearance clusters by the boundaries of C partitions. The combination top-down-bottom-up effectively reduces the complexity to $O(NC + N_1^2 + \cdots + N_C^2 + Q^2) \ll O(N^2 \log(N))$ where $N_1 + \cdots + N_C = N$. The run time is an order of magnitude shorter than for standard k-means (if k set to the same number of clusters).

3.3. Fast matching

The approach uses large numbers of local features. To maintain a reasonable run time we make use of the hierarchical structure and efficiently match features to cluster centers. This data structure is a metric tree, also called ball tree where the metric obeys the triangular inequality [17]. It is therefore straightforward to use the hierarchy obtained from our bottom-up agglomerative clustering. Moreover, this method optimizes a global criterion for constructing a compact ball tree representation [18]. Instead of matching each query image feature separately to the appearance clusters, as it is done in many previous approaches [12, 22, 23], we represent both the query image and the model as tree structures. The matching of two trees is then performed by first computing the Euclidean distance between the centroids of top nodes, see figure 3. If the distance between two nodes is smaller than the sum of their radii (nodes intersect) then the first node is compared with all the children of the intersecting node. The search is continued down to the bottom nodes of both trees with a simple recursive method which verifies if two nodes intersect. The distance between bottom nodes is compared with a similarity threshold.

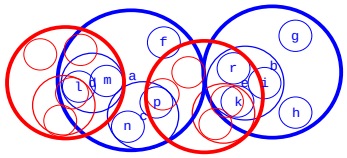


Figure 3. (a) Matching two tree structures. The cluster center indices in blue can be compared with the tree structure in figure 2(a).

This matching method allows for a significant speedup. The matching is approximately 200 times faster compared to exhaustive search. Moreover, the matching solution is identical to the one obtained with exhaustive search unlike in approximations based on kd-tree [3].

3.4. Training and recognition

Training. Given a set of roughly aligned training images with associated class labels we extract features and compute the appearance clusters for the hierarchical tree. Note that only positive training data is used to obtain the appearance clusters. Estimating model parameters (c.f. equations 5 and 6) consists of matching features to the appearance clusters. We therefore match the model tree to itself, and compute the object related distributions. Such obtained distributions are more reliable than just using appearance cluster members, since we can vary similarity threshold β to obtain more matches per cluster. Moreover, a larger training set than the one used for clustering can be used to estimate the distributions (not used here). The background related terms are computed in a similar way by matching features from background images to the appearance clusters.

Recognition. Given a query image we detect features and build a tree structure using the clustering method. We then match the query tree to the model tree to compute a 3D likelihood function (cf. equations 1 and 2) for each object class in the model. This is done by scaling and rotating the geometric distribution $g_j(d, \phi, \sigma)$ with the scale and dominant orientation of each query feature that matches to cluster a_j. All 3D geometric distributions of matched clusters are then integrated into the likelihood function. Finally, we search for local maxima using a 3D Gaussian kernel and solve for overlapping hypotheses and ambiguities (cf. equations 3 and 4).

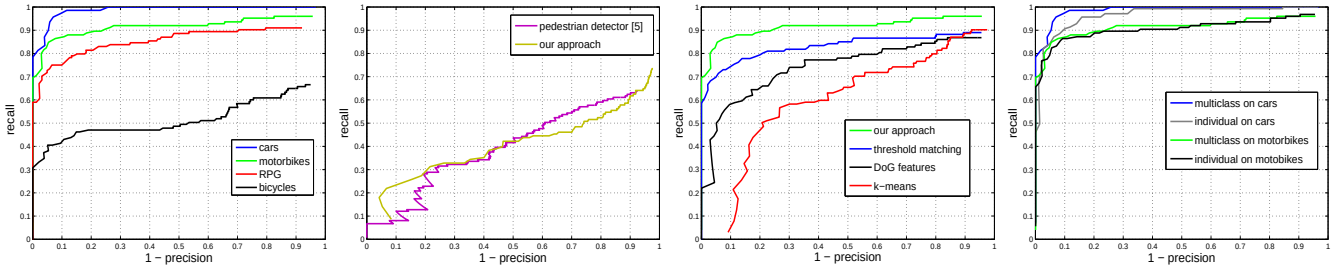
Rotation recovery. Given a hypothesis and the features that contributed to the hypothesis we compare their dominant orientations θ_q with the orientations collected in the fourth dimension of geometric distributions (cf. section 2.1) and compute a 1D histogram of rotation angles. The maximum bin in this histogram indicates the orientation of the hypothesis.

4. Experimental results

In this section we present experimental results for our method. We first explain the experimental setup, then show the results on different test data and compare with other methods.

4.1. Experimental setup

Training and test data. We evaluate the performance of the recognition system on 4 object classes from a street scenario, namely pedestrians, cars, motor-bikes and bicycles. We introduce one more challenging category which is an RPG shooter (rocket propelled grenade launcher). All training and test data except RPG come from the Pascal collec-



(a) (b) (c) (d)

Figure 4. Performance evaluation. (a) Precision-recall for cars, motorbikes, RPG, and bicycles. (b) Results for pedestrians (c) Comparison of different methods on motorbikes. (d) Performance for multi-class vs. individual detectors.

tion [20]. We use 600 pedestrian training images, which is a subset of the data used in [5]. The test set consists of 84 images with 149 pedestrians. Note that 20 images in this set contain only upper-bodies, which cannot be detected by the current implementation of our approach due to the object center falling outside the image, thus outside the limits of the likelihood function. Nonetheless, we use this set in order to be directly comparable with the performance of [5] for which the results are reported in [20]. The multi-scale car test data of 108 images with 139 cars was used in [1, 9]. We used 50 side views of cars for training. The 115 motorbike test images were previously used in [9]. For training we use 150 motorbike examples from 101 Caltech set. The bicycle test data [20] contains 113 images with 123 bicycles. 100 side views of bicycles were used for training. A dummy RPG was used to prepare the training images and some of the test images. The training set contains 104 images of various RPG-shooter poses in 5 different cloths on a uniform background. The test data contains 40 images collected from feature movies, TV news as well as taken from real street scenes. Finally, 600 background images are taken from the set in [5] for training.

Evaluation criteria. We adopt the evaluation criteria used in Pascal challenge [20]. A detection is considered correct, if the area of overlap between the predicted bounding box B_p and ground truth bounding box B_{gt} exceeds 50% by the formula $\frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})} > 0.5$. Thus, our results are directly comparable with [5, 9, 20].

4.2. Run time

One codebook representation is trained for all five categories. There are 1004 positive training images in total and $2.5 \cdot 10^6$ edge based features. The tree structure contains $8.2 \cdot 10^4$ nodes with 7 top nodes and $5.5 \cdot 10^4$ bottom nodes (appearance clusters). The entire training procedure including clustering and model parameters estimation takes 8 hours. 95% of that time is spent on building the tree structure. For comparison, training a single class model from $2 \cdot 10^5$ features takes less than 1 hour on a P4 2GHz. Feature extraction in a VGA image takes less than 1 second. Clustering, matching and computing the likelihood takes 1-10 seconds, depending on the number of features in the image. This favorably compares to other methods *e.g.* in [8] training a model from $1.2 \cdot 10^4$ takes 24-36 hours and the detection time is 10-15 second. In [21] training from 50 images takes 1-4 hours and the detection time is 10s per image.

4.3. Recognition results

Figure 4(a) shows precision-recall curves for cars, motorbikes, bicycles, and RPG object classes. The best recognition score is obtained for cars. 94.7% EER (equal error rate) performance compares favorably to 44% [1] and 87.8% in [9]. Initial recall for our method with 0 false positives is 78%. The motorbike test data is more challenging and contains many out of plane rotations as well as occlusions. Our EER performance for this data is 88% with initial recall of 70% compared to 81% EER and 40% initial recall in [9]. The performance for bicycles is lower since the test images contain many different viewpoints and partial occlusion. EER score is 49%. However the detector trained on side views demonstrates some tolerance to viewpoint changes (cf. figure 5) due to the rotation invariance of the model. The score for RPG class is 81%. The confusion factor between pedestrians and RPGs is surprisingly low, but some false positives occur on upper bodies of pedestrians. Finally the lowest performance of 45% EER is on pedestrian images. However a state of the art pedestrian detector from [5] obtains a similar EER score but lower recall in high precision range. 2416 of positive and 12180 of negative examples were used to train this detector compared to our 600 positive and 600 negative. Both curves are displayed in figure 4(b).

In figure 4(c) we compared the overall performance of the system on the motorbike test data with different methods applied at various stages of our approach. The top performance of 89% is obtained by the method proposed in the paper. The score drops to 72% if a DoG feature detector from [14] is applied instead of our edge based detector. We also modified the original approach to threshold based matching (see section 2.4). The resulting EER performance is 8% lower. Finally we changed the clustering approach

to k-means to obtain the appearance clusters. This method was tested without the hierarchical tree. The matching was approximately 150 times slower and the EER performance dropped by 28%. These results clearly show that a careful selection of methods at each stage of the approach can bring a significant improvement of the performance.

In figure 4(d) we compare the performance of the multi-class detector with the results obtained with detectors trained for individual classes. The recognition rate of individual car and motorbike detectors is slightly lower than for the complete model which agrees with the observations in [23, 24]. No significant difference in performance was observed for other classes. However, we observe that the recall is higher for our new multi-class detector but the confusions with other classes lower the score. We also observe that the number of appearance clusters grows sub-linearly with increasing number of object classes as more clusters are shared between different categories. The multi-class model uses approximately 75% of the number of clusters in all individual detectors with the same maximum size of appearance clusters. This indicates that the method can scale to a large number of classes since the complexity reduces compared with individual detectors. Another consequence of sharing clusters is that the required amount of the training data is reduced. In particular, wheel based vehicles seem to benefit from the training data of each other. Additional experiments have to be carried out to quantify this.

Figure 5 shows detection examples on different categories. The results are displayed for recognition threshold set to EER on the pedestrian database. Note the difficulty of the recognition task in presence of occlusion on motorbike examples and RPG, various object scales for pedestrians and cars, different viewpoints for bicycles, multiple object classes occluding each other and in-plane rotations. Bicycle examples demonstrate the deficiencies of the star shape model. The object center is hypothesized based on strong cues from one wheel only. Thus, the star model allows to deal with occlusion but not with strong changes in the structure of the object.

5. Conclusions

We have presented an approach capable of detecting multiple object classes simultaneously in images using a single hierarchical codebook representation for all object classes. The performance is comparable with state of the art discriminative approaches based on powerful machine learning techniques and trained on large image sets of individual classes. We proposed an efficient method for building object class representation and for recognition. The complexity of the model, training as well as recognition run times are improved compared to many other methods. We studied the influence of various detector parameters and demonstrated that careful selection of feature detector, clustering method, and probabilistic model are equally important and can lead to significant improvements.

In the short term it would be interesting to test other features *e.g*. from [16]. Smaller sets of initial features would result in further increase of efficiency and allow to learn large numbers of object classes. Another experiment is to use a fixed tree model trained on several classes and add a new class to the model by estimating the geometric distributions, without using appearance clusters from the new class. This would demonstrate the scalability to large number of object classes.

Acknowledgments

This work has been funded by the EU project CoSy (IST-2002-004250).

References

[1] S. Agarwal, A. Awan, and D. Roth, Learning to detect objects in images via a sparse, part-based representation. *PAMI*, 26(11):1475–1490, 2004.

[2] T. Berg, A. Berg, and J. Malik, Shape Matching and Object Recognition using Low Distortion Correspondence. In *CVPR*, pages 26–33, 2005.

[3] J. Beis and D. Lowe, Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces. In *CVPR*, pages 1000–1006, 1997.

[4] G. Csurka, C.R. Dance, L. Fan, J. Willamowski, C. Bray. Visual Categorization with Bags of Keypoints In *ECCV* Workshop, 2004.

[5] N. Dalal and B. Triggs Histograms of Oriented Gradients for Human Detection. In *CVPR*, pages 886–893, 2005.

[6] G. Dorko and C. Schmid. Selection of scale-invariant parts for object class recognition. In *ICCV*, pages 634–640, 2003.

[7] L. Fei-Fei, R. Fergus, P. Perona, A Bayesian Approach to Unsupervised One-Shot Learning of Object Categories. In *ICCV*, pages 1134–1140, 2003.

[8] R. Fergus, P. Perona, and A.Zisserman, Object class recognition by unsupervised scale-invariant learning. In *CVPR*, pages 264–271, 2003.

[9] M. Fritz, B. Leibe, B. Caputo, and B. Schiele. Integrating Representative and Discriminant Models for Object Category Detection. In *ICCV*, pages 1363–1370, 2005.

[10] F. Jurie and B. Triggs. Creating Efficient Codebooks for Visual Recognition. In *ICCV*, pages 604–610, 2005.

[11] B. Leibe, E. Seeman, and B. Schiele. Pedestrian Detection in Crowded Scenes. In *CVPR*, pages 878–885, 2005.

[12] B. Leibe, A. Leonardis, and B. Schiele. Robust Object Detection by Interleaving Categorization and Segmentation. Submitted to *IJCV*, 2006.

[13] T. Lindeberg. Feature detection with automatic scale selection. In *IJCV*, 30(2):79–116, 1998.

[14] D. Lowe, Distinctive image features from scale-invariant keypoints. *IJCV*, 2(60):91–110, 2004.

[15] K. Mikolajczyk, A. Zisserman, and C. Schmid. Shape recognition with edge-based features. In *BMVC*, pages 779–788, 2003.

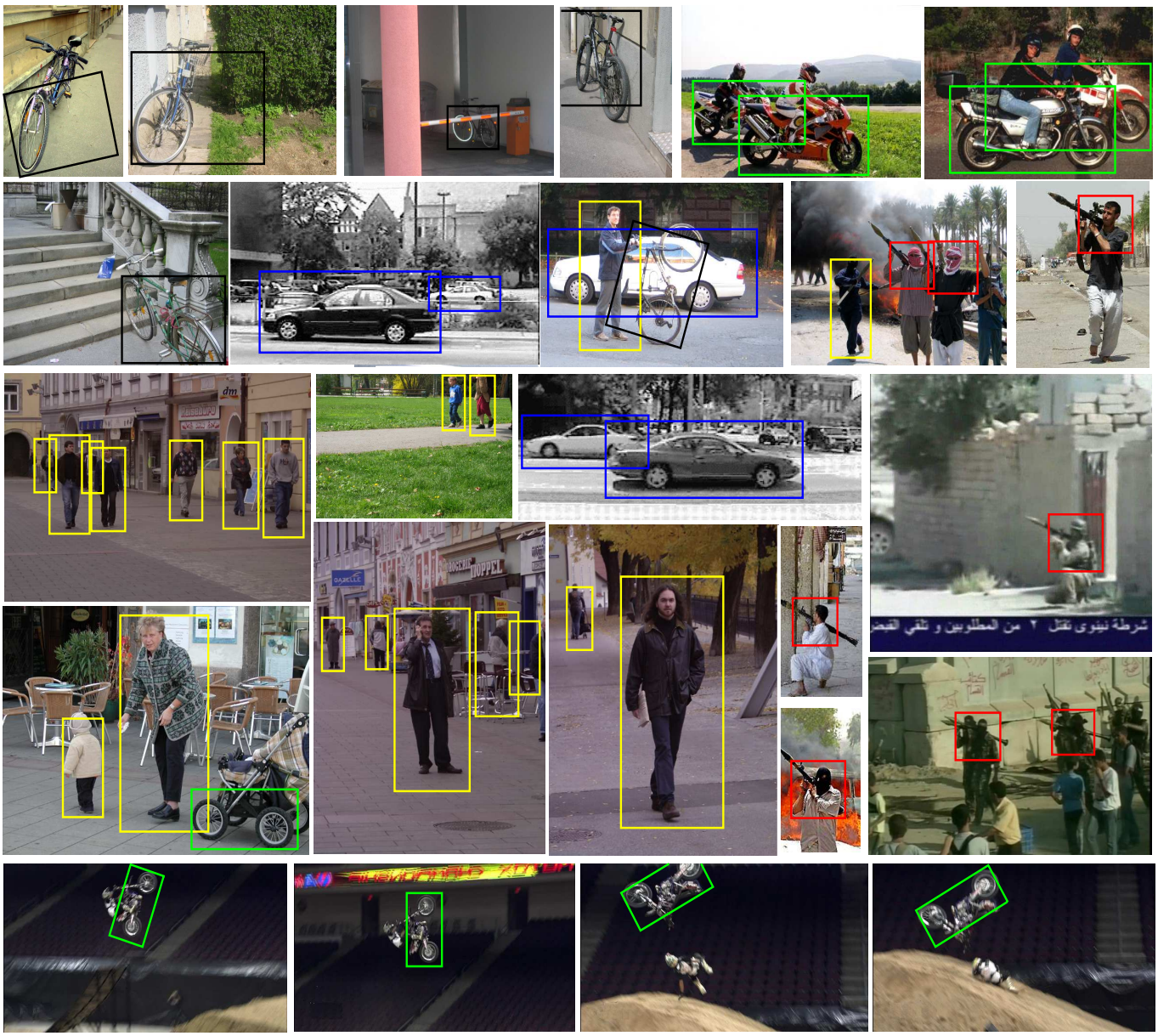


Figure 5. Recognition examples of motorbikes, cars, bicycles, pedestrians, and RPG. Different objects have different colors of bounding boxes.

[16] K. Mikolajczyk, B. Leibe, and B. Schiele. Local features for object class recognition. *ICCV*, pages 1792–1799, 2005.

[17] A.W. Moore, The Anchors Hierarchy: Using the Triangle Inequality to Survive High Dimensional Data. In *UAI*, AAAI Press, pages 397–405, 2000.

[18] S.M. Omohundro, Five balltree construction algorithms. Technical Report TR-89-063, Berkeley, 1989.

[19] A. Opelt, M. Fussenegger, A. Pinz, and P. Auer. Weak hypotheses and boosting for generic object detection and recognition. In *ECCV*, pages 71–84, 2004.

[20] M. Everingham *et al.* The 2005 PASCAL Visual Object Classes Challenge. In *PASCAL Challenge Workshop*, LNAI, 2006.

[21] J. Shotton, A. Blake and R. Cipolla, Contour-Based Learning for Object Detection. In *ICCV*, pages 503–510, 2005.

[22] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman, Discovering object categories in image collections. In *ICCV*, pages 370–377 , 2005.

[23] E. Sudderth, A. Torralba, W. T. Freeman, and A. Wilsky. Learning Hierarchical Models of Scenes, Objects, and Parts. In *ICCV*, pages 1331–1338, 2005.

[24] A. Torralba, K. Murphy, and W. Freeman. Sharing features: efficient boosting procedures for multiclass object detection. In *CVPR*, pages 762–769, 2004.

[25] M. Weber, M. Welling, and P. Perona, Unsupervised Learning of Models for Recognition. In *ECCV*, pages 628-0641, 2000.