Efficient 3D Object Detection using Multiple Pose-Specific Classifiers

Michael Villamizar[1]
mvillami@iri.upc.edu
Helmut Grabner[2]
grabner@vision.ee.ethz.ch
Juan Andrade-Cetto[1]
cetto@iri.upc.edu
Alberto Sanfeliu[1]
sanfeliu@iri.upc.edu
Luc Van Gool[2,3]
vangool@vision.ee.ethz.ch
Francesc Moreno-Noguer[1]
fmoreno@iri.upc.edu

[1] Institut de Robòtica i Informàtica Industrial CSIC-UPC
Barcelona, Spain
[2] Computer Vision Laboratory
ETH Zurich
Switzerland
[3] ESAT - PSI / IBBT
K.U. Leuven
Belgium

Abstract

We propose an efficient method for object localization and 3D pose estimation. A two-step approach is used. In the first step, a pose estimator is evaluated in the input images in order to estimate potential object locations and poses. These candidates are then validated, in the second step, by the corresponding pose-specific classifier. The result is a detection approach that avoids the inherent and expensive cost of testing the complete set of specific classifiers over the entire image. A further speedup is achieved by feature sharing. Features are computed only once and are then used for evaluating the pose estimator and all specific classifiers. The proposed method has been validated on two public datasets for the problem of detecting of cars under several views. The results show that the proposed approach yields high detection rates while keeping efficiency.

1 Introduction

The problem of efficiently testing multiple specific classifiers has recently gained popularity for tackling the problem of detecting multiple object categories or specific objects seen from different viewpoints. In these problems, each object class, or object view, is commonly considered as a different topic represented by a distinct classifier. As a result, a large number of discriminative and specific classifiers are computed. Although these classifiers can be learned quite efficiently, testing each of them over an image is computationally expensive.

In this work we propose an *efficient strategy* for testing *multiple specific classifiers* for object detection. We study the problem more closely on the detection of cars from multiple views. This category includes challenges such as high inter-class variations, lighting changes, several car sizes or different aspect ratios of the bounding box. In order to address all these issues, we use a decoupled approach consisting of (i) a pose estimator and (ii) a

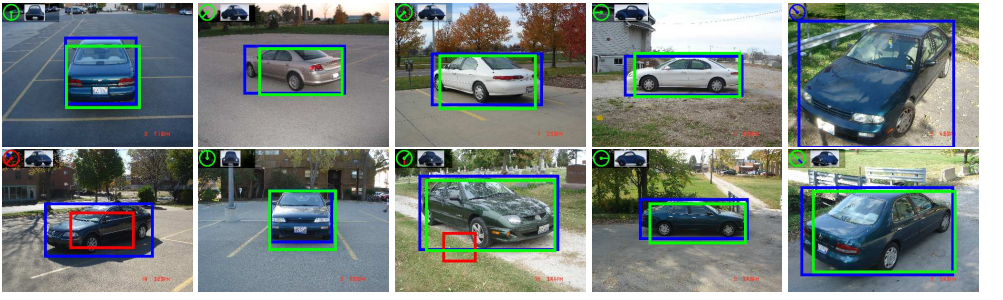

Figure 1: Efficient localization and pose estimation in the UIUC database [15]. Our approach allows localizing cars and estimating their pose despite large inter-class variations and in about 1 second. Correct detections are depicted by green rectangles, whereas false positives are indicated by red ones. The ground truth is shown by a blue rectangle. The circle and car toy located at top and left indicate the estimated viewpoint.

set of pose-specific classifiers. The estimator acts as filter and prevents of having to evaluate all the specific classifiers at every position. Furthermore, we use feature sharing for the estimator and all classifiers. Both these characteristics yield a remarkable efficiency to our approach while keeping high detection rates. Fig. 1 depicts some detection results and the corresponding estimated poses.

Related Work. Several strategies have been proposed in the past for an efficient object detection. Some of them mainly rely on (*i*) features that can be calculated very fast over images, and thus, increase the speed of the sliding window classifier that is evaluated at multiple scales and locations [12, 21, 24]. Other methods use specific cascaded classifier structures *e.g.* [17, 24] which allow rejecting background windows at early stages and hence, also reduce the computational effort. In other words, these approaches aim for (*ii*) reducing the search space during the detection phase. This is also achieved by means of branch and bound techniques [8, 9], using object priors [1] or splitting the process in two consecutive phases of object estimation and specific detection [11, 13, 14, 22]. Finally, other works (*iii*) have proposed to share features across object classes or views [20, 23, 25].

Contribution. We propose an efficient method for the 3D object detection that integrates synergically the strategies reviewed above. More specifically, our method computes fast Random Ferns (RFs) [12] over local histograms of oriented gradients. These features are simple comparisons whose binary outputs encode objects appearance. These Random Ferns are shared among objects and used for the computation of the two-step detection approach, that is, the pose estimator and the set of pose-specific classifiers. Unlike other previous works that use Hough-based approaches as object classifiers [3, 4, 11], we use a novel Hough-RFs for building an efficient and robust 3D pose estimator. This estimator uses the Hough transform to learn and map the local appearances of objects (encoded by RFs) into probabilistic votes for the object center. This methodology overcomes to previous works which compute rough estimators or predict the object size at first [13, 22].

The resulting method is able to learn and detect objects in a straightforward and efficient manner. In particular, the estimator and specific classifiers can be learned in a couple of minutes, while the object detection is performed in about 1 second, using a non-optimized code based on Matlab. In addition, this efficiency is accompanied with a high detection rate, comparable and even better than existing approaches.

2 Overview of the Method

The main ingredients of our approach are (i) a shared feature representation and (ii) an object pose estimator that limits the search space for (iii) object pose specific classifiers. Fig. 2 depicts an overview of the method, which we describe in detail in the following sections.

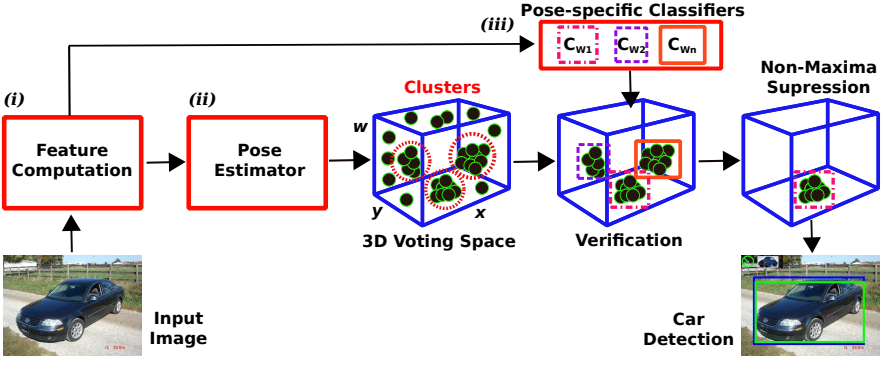

Figure 2: Overview of the proposed approach. To detect the 3D pose, given an input image we initially compute a set of shared RFs (Feature Computation). We then apply the pose estimator to generate several object/pose hypotheses which are verified by the pose-specific classifiers. Non-maximal potential detections are finally filtered out.

Since features are shared among classifiers, their computation is performed in an initial step that is pose independent. This allows an efficient computation of both the pose estimator and the classifiers. For the pose estimation step, each feature is evaluated over the entire image, and casts probabilistic votes for the object/pose center. This yields a set of potential hypotheses (clusters within the voting space), which are then validated according to a set of specific classifiers. Finally, multiple detections are removed using non-maxima suppression.

3 Feature Computation: Random Ferns

The first key element of our approach are the kind of features we use: the Random Ferns. They consist of sets of binary features resulting from simple comparisons on the intensity domain [7, 13]. Yet, and drawing inspiration from [23], we compute RFs over local histograms of oriented gradients (HOG), that is, our binary features are simple comparisons between two bins of HOG. The co-occurrence of all feature outputs encodes different image appearances that are used for building the estimator and each one of the classifiers. More formally, each Random Fern F captures the co-occurrence of r binary features, whose outputs determine the Fern observation z. Therefore, each Fern maps the image appearances to a $K = 2^r$-dimensional space, $F : x \rightarrow z$ where x is an image sample and $z = 1, 2, ...K$.

In addition, in order to gain in efficiency we share the same RFs among different classes. This was already proposed in [23], although as we will show in the results section, this previous work does not scale properly for a large number of classifiers since every classifier is independently tested.

4 The Pose Estimator

Based on the response of the RFs on an input image, the pose estimator will provide image regions with a high probability of object/pose. For that, we will need to map from the feature

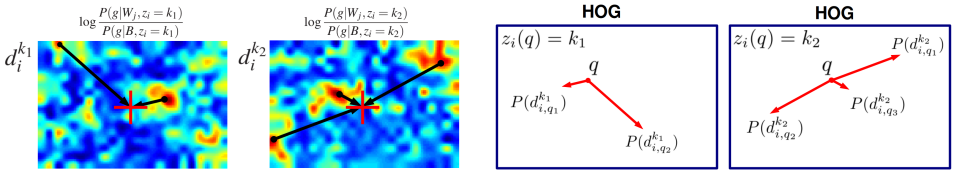

Figure 3: The Hough-RFs Estimator. **Left:** The computation of the estimator is carried out by selecting the most discriminative appearance locations against the background category B. Each Fern output $(z_i = k)$ describes a specific image appearance that has associated a list of distances d_i^k indicating where this appearance has occurred over training samples. **Right:** In runtime, each Fern is evaluated in every image location q to cast probabilistic votes for diverse image locations according to its output. The result is a voting space where its maximum values correspond to possible object instances.

domain of the RFs to spatial image locations. This is achieved by means of what we call *Hough-RFs*.

4.1 Hough-RFs

In the spirit of Hough-Forests [9, 10], our Hough-RFs encodes the local appearance captured by RFs and casts probabilistic votes about the possible location of object poses. Specifically, for every Fern F_i each output $(z_i = k)$ represents a specific image appearance that has associated a list of displacements $\{d_i^k\}$ where that appearance has occurred in images. These displacements are measured from the image center and are extracted during the learning phase as those ones with higher occurrence over training samples.

Training. The computation of the Hough-RFs is carried out by evaluating a fixed set of m RFs for every image location of training samples and selecting the most discriminative object appearances and their displacements. This is done for each object view W_j and an additional background class B. Assuming probability independence among Ferns [12], we can define the estimator E_{W_j} as:

$$E_{W_j}(x) = \log \frac{\prod_{i=1}^{m} P(g, z_i | W_j, F_i)}{\prod_{i=1}^{m} P(g, z_i | B, F_i)} = \sum_{i=1}^{m} \log \frac{P(g, z_i | W_j, F_i)}{P(g, z_i | B, F_i)}, \tag{1}$$

where x represents the possible location of the object (image center for training) and g denotes the image locations where the Fern F_i is calculated. These locations are always measured from the image center of the object pose j.

The aim is to compute the estimator in order to maximize the ratio of probabilities between the object view and background classes (Eq. (1)) with the objective of selecting the most important image appearances and their locations for the current pose. This is done by selecting the most discriminative locations against the background samples. Fig. 3(Left) shows a simple example where discriminative locations for Fern outputs k_1 and k_2 are chosen. These locations form the lists of displacements $\{d_i^{k_1}\}$ and $\{d_i^{k_2}\}$ which are used to cast probabilistic votes in runtime. These votes are weighted according to their occurrences over training images,

$$P(d_{i,q_n}^k) = \log \frac{P(g = q_n, z_i = k | W_j, F_i)}{P(g = q_n, z_i = k | B, F_i)}, \quad k = 1, 2, ..., K \quad i = 1, 2, ..., m \tag{2}$$

where d_{i,q_n}^k is a displacement in the list $\{d_i^k\}$.

Testing. Once the estimator has been constructed for every object pose, it is evaluated in runtime as follows: given an input image, a HOG is computed over the whole image for then to test the m RFs. For each image location q (in the HOG space), each Fern F_i casts votes for different image locations according to its observation $z_i(q)$ and its voting list $\{d_i^k\}$. This voting procedure is illustrated in Fig. 3(Right). The result of evaluating all RFs is a 3D voting space where their maximum values correspond to object/pose candidates.

4.2 Efficient Pose Estimation

As it was exposed in the previous section, the cost of the estimator depends on the number of poses given that each RF must cast votes for the different views. In order to speed up the process, similar to the work of [2], we propose to evaluate the estimator in two consecutive steps. For each Fern F_i, the first step predicts the most likely object pose according to its observation. The second step cast votes only for the estimated pose. In this way, the cost of evaluating the estimator for multiple poses is reduced considerably.

The most likely object pose W_*^i for a Fern F_i tested in location q is calculated by

$$W_*^i = \arg\max_j P(z_i(q) = k | W = W_j), \quad j = 1,2,...,J \tag{3}$$

where W is the object pose variable and J is the total number of poses. Rewriting the Eq. 1 the estimator can be defined as:

$$E(x) = \sum_{i=1}^{m} \log \frac{P(\{d_{i,q}^k\}, z_i(q) = k | W_*^i, F_i)}{P(\{d_{i,q}^k\}, z_i(q) = k | B, F_i)}, \tag{4}$$

being q every location in the image.

Search Space Reduction. In order to reduce the possible locations where a time-consuming pose-specific classifier (see Sec. 5) has to be evaluated, we look for the most remarkable hypotheses. This is done by filtering the estimator output, $E(x) > \beta_e$, being β_e a sensitivity parameter. The choice of this parameter is, however, a trade-off between speed of the approach and an increment of false negatives. For instance, if $\beta_e = 0$ each pose-specific classifier is tested on every image position. In this case, the object is not missed but it implies a high computational cost given that all classifiers are tested. By contrast, for increasing values of β_e we speed up the detection phase but with the risk of filtering likely object locations. The estimator in this case reduces the search space and may yield false negatives (missed objects). The effects of this parameter are evidenced in more detail in Sec. 6.

5 The Pose-specific Classifier

Each one of the pose-specific classifiers is built independently using a boosting combination of RFs [22]. Hereby, a classifier is a set of weak classifiers, where each one of them is based on a Fern selected from the common pool of RFs. This pool is constructed at random and is shared by all classifiers in order to reduce the cost of calculating a large number of pose-specific features and to reuse features for constructing different weak classifiers.

The specific classifier $H_{W_j}(x)$ is built in order to find the Ferns F_i and locations g_i that most discriminate the positive class from the background. The positive class corresponds to a collection of image samples extracted from the specific object view W_j, whereas the background images B are used for extracting negative samples. The classifier computation is done by means of the Real AdaBoost algorithm [16], that iteratively assembles weak classifiers and adapts their weighting values with the aim of focusing all its effort on the hard samples, which have been incorrectly classified by previous weak classifiers.

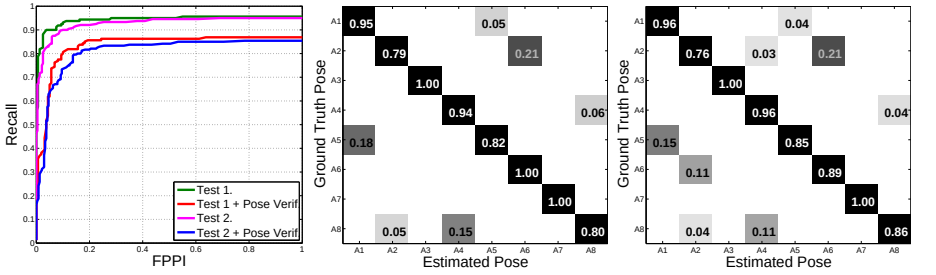

Figure 4: UIUC Dataset. Car detection and pose estimation. **Left:** ROC curves using different evaluation approaches. **Center, Right:** Confusion matrices for Test 1 and Test 2.

The boosted classifier is then defined as:

$$H_{W_j}(x) = \sum_{t=1}^{T} h^{(t)}_{W_j}(x) > \beta_{W_j} \,, \tag{5}$$

where β_{W_j} is its threshold and $h^{(t)}_{W_j}$ is a weak classifier computed by

$$h^{(t)}_{W_j}(x) = \frac{1}{2} \log \frac{P(F_t|W_j, g_t, z_t = k) + \varepsilon}{P(F_t|B, g_t, z_t = k) + \varepsilon} \,, \quad k = 1, .., K \,, \tag{6}$$

where F_t is the selected RF that is evaluated at fixed location g_t (measured from the image center), and the parameter ε is a smoothing factor. At each boosting iteration t, the probabilities $P(F_t|W_j, g_t, z_t)$ and $P(F_t|B, g_t, z_t)$ are computed using a distribution of weights D over the training samples. This is done as follows,

$$P(F_t|W_j, g_t, z_t = k) = \sum_{\substack{i: z_t(x_i) = k \\ y_i = +1}} D_t(x_i), \quad P(F_t|B, g_t, z_t = k) = \sum_{\substack{i: z_t(x_i) = k \\ y_i = -1}} D_t(x_i) \tag{7}$$

being x_i and $i = 1, 2, .., N$ the set of training samples. To select the most discriminative weak classifier at each iteration we use, as other previous works, the Bhattacharyya distance. In this way, the weak classifier $h^{(t)}_{W_j}$ that minimizes this distance is chosen.

In the present work all pose-specific classifiers are learned using the same parameters, that is, 300 weak classifiers and 10 shared RFs. Since they are learned independently for extracting the most relevant features for each pose, the resulting classifiers are very discriminative for each pose and focus on the most relevant object parts.

6 Experimental Results

We validated using two public and recent datasets: (i) The UIUC dataset [15], and (ii) EPFL dataset [13]. Both datasets contain car instances under multiple views, scales and harsh image conditions such as light changes.

6.1 UIUC car Dataset

Dataset. This dataset has multiple views of 10 cars in outdoor settings. For each car, images under 8 different angles, 2 camera heights and 3 distances are available. We train and test our detector using two different sets of images, Test 1 and Test 2, as done in other state-of-the-art approaches[10, 15]. Test 1 is made by 320 car images without the largest distance, whereas Test 2 contains the entire set of 480 images. For both tests, the first 5 cars of each set are used for training and the rest are used for testing.

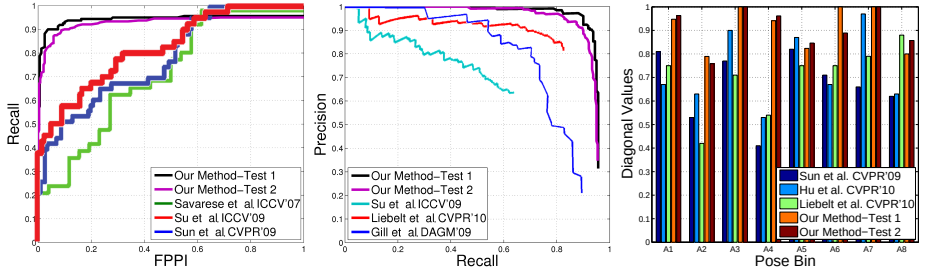

Figure 5: UIUC Dataset. Comparison against state of the art. **Left:** ROC curves for our method and some recent works. **Center:** The comparison is done using the Recall-Precision plots. **Right:** Comparison in terms of the diagonal values of the confusion matrix.

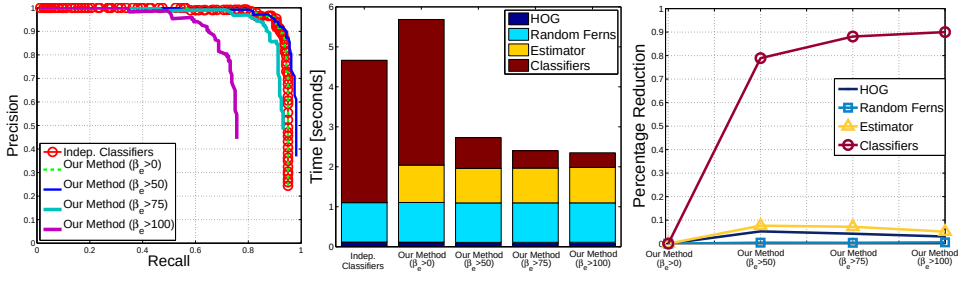

Figure 6: UIUC Dataset. Detection and efficiency in terms of the sensitivity parameter β_e. **Left:** Detection rates. **Center:** Detection times. **Right:** Computational reduction of the proposed approach.

Results. The detection performance of our approach is shown in the ROC curves of Fig. 4(Left). We depict the results of just detection (Test 1,2) and detection plus pose estimation (Test 1,2 + Pose Verif). Note that for images with the largest distance (Test 2) the detection rates are slightly reduced. Fig. 4(Center,Right) show the confusion matrix both for Test 1 and Test 2. Observe that only a small fraction of the detections are incorrect, and usually correspond to confusions of the true with the symmetric pose.

Comparison. The ROC curves of Fig. 5(Left) and the Recall-Precision plots of Fig. 5(Center) compare our approach with state of the art [5, 6, 10, 15, 18, 19]. In both cases, our method outperforms the detection rates of other approaches. Fig. 5(Right) compares the methods in terms of pose classification. Note again that the proposed method yields better results. A few sample results are shown in Fig. 1.

Speedup. Fig. 6(Center), depicts the detection times of our approach for different values of the sensitive parameter β_e. We also show an additional method that would test all the pose-specific classifiers (*Indep. Classifiers*). It can be seen that the efficiency of our method is increased for larger magnitudes of β_e (see Fig. 6(Right)). However, this is at expense of a reduction in the recall rate, because the estimator misses correct object hypotheses. This can be observed at Fig. 6(Left).

6.2 EPFL car Dataset

Dataset. This dataset contains cars under multiple views, light changes and varying backgrounds. The set of images is formed by images collected from 20 specific cars rotating on a platform. The first 10 cars are used for learning the estimator and each one of the pose-

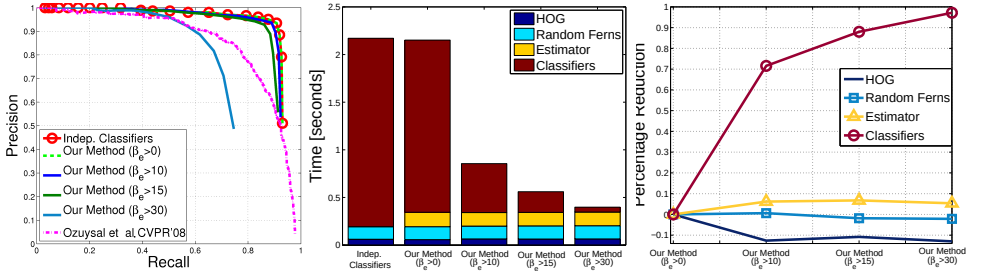

Figure 7: EPFL Dataset. Detection and efficiency. **Left:** Recall-Precision plots of the proposed method and the state-of-the-art work of [13] . **Center:** Detection times for several values of the parameter β_e . **Right:** Computational reduction of the proposed approach.

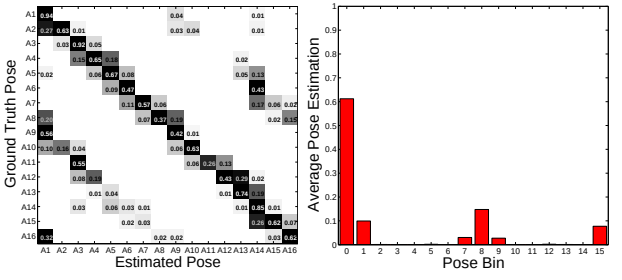

Figure 8: EPFL Dataset. Car pose estimation. **Left:** Confusion matrix. Incorrect pose estimations occur mainly at opposite views because of their strong similarities. **Right:** Distribution of error for each pose bin.

specific classifiers. The remaining 10 cars are used for testing [13]. For this dataset, 32 pose-specific classifiers corresponding to 16 views and 2 different aspect ratios have been learned.

Results. Detection rates for this dataset using several values of β_e are shown in Fig. 7(Left). Also the performance curve reported by [13] is depicted for comparison purpose. We see that our method consistently outperforms this work. On the other hand, Fig. 7(Center) plots the detection times, and shows the efficiency of our method by reducing the time of evaluating the set of classifiers. This efficiency is also evidenced in Fig. 7(Right), where the percentage of computational reduction across the values of β_e are shown. We see that the main computational reduction of our method lies in the evaluation of pose-specific classifiers.

Viewpoint Estimation. To measure the viewpoint estimation accuracy of our approach on this dataset we build again the confusion matrix (Fig. 8(Left)). We can see that most estimations are correct, showing a diagonal line. This performance is similar to the results reported in [13], where most of incorrect estimations appear on symmetric points of view. This is because there is a strong similarity among these views. This issue is represented in Fig. 8(Right), where the distribution of error among pose bins is shown. Most of pose estimations are correct (i.e., they belong to bin 0), but a few of them appear at opposite and adjacent pose bins. Fig. 9 shows some sample detections on this database.

7 Conclusions

This work has presented an efficient strategy for testing multiple classifiers by means of a decoupled approach consisting of a pose estimator and a set of pose-specific classifiers.

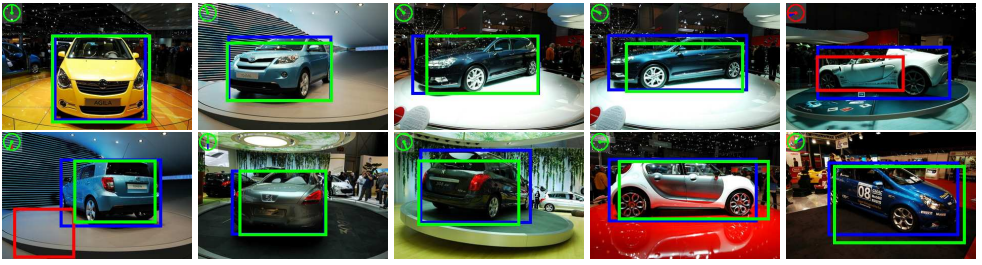

Figure 9: EPFL Dataset. Sample Results. Please see Fig. 1 for the interpretation of the results.

This method has reported high detection rates and efficiency for the problem of detecting car under multiple views. The estimator filters out image locations to yield potential candidates where specific classifiers are then evaluated.

To increase efficiency, it has been proposed to compute all the specific classifiers and the estimator using a reduced set of features (Random Ferns). This allows to reduce the cost of evaluating a large number of features and to decompose the detection process into two stages. The first one tests the RFs over the input image, whereas the second one calculates the estimator and the corresponding specific classifier. The benefit lies in the feature computation is independent of the number of classifiers.

Acknowledgement

The first stages of this work were done during the research visit of first author to K.U. Leuven and ETH Zurich. This visit was funded by the Spanish Ministry of Education. This work was also supported by the Spanish Ministry of Science and Innovation under Projects RobTaskCoop (DPI2010-17112), PAU (DPI2008-06022), and MIPRCV (Consolider - Ingenio 2010 CSD2007-00018), and the EU CEEDS Project FP7-ICT-2009-5-95682.

References

[1] B. Alexe, T. Deselaers, and V. Ferrari. What is an object? In *CVPR*, 2010.

[2] K. Ali, F. Fleuret, D. Hasler, and P. Fua. Joint learning of pose estimators and features for object detection. In *ICCV*, 2009.

[3] O. Barinova, V. Lempitsky, and P. Kohli. On detection of multiple object instances using hough transform. In *CVPR*, 2010.

[4] J. Gall and V. Lempitsky. Class-specific hough forests for object detection. In *CVPR*, 2009.

[5] G. S. Gill and M. D. Levine. Multi-view object detection based on spatial consistency in a low dimensional space. In *DAGM*, 2009.

[6] W. Z. Hu and S. Zhu. Learning a probabilistic model mixing 3d and 2d primitives for view invariant object category recognition. In *CVPR*, 2010.

[7] Z. Kalal, J. Matas, and K. Mikolajczyk. P - n learning: Bootstrapping binary classifiers by structural constraints. In *CVPR*, 2010.

[8] C. Lampert, M. Blaschko, and T. Hofmann. Beyond sliding windows: Object localization by efficient subwindow search. In *CVPR*, 2008.

[9] A. Lehmann, B. Leibe, and L. Van Gool. Fast prism: Branch and bound hough transform for object class detection. In *IJCV*, 2010.

[10] J. Liebelt and C. Schmid. Multi-view object class detection with a 3d geometric model. In *CVPR*, 2010.

[11] S. Maji and J. Malik. Object detection using a max-margin hough transform. In *CVPR*, 2009.

[12] M. Ozuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code, . In *CVPR*, 2007.

[13] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization, . In *CVPR*, 2008.

[14] H. A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *CVPR*, 1998.

[15] S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *ICCV*, 2007.

[16] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. Machine Learning, 1999.

[17] J. Šochman and J. Matas. Waldboost - learning for time constrained sequential detection. In *CVPR*, 2005.

[18] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *ICCV*, 2009.

[19] M. Sun, H. Su, S. Savarese, and L. Fei-Fei. A multi-view probabilistic model for 3d object classes. In *CVPR*, 2009.

[20] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *PAMI*, 2007.

[21] O. Tuzel, F. Porikli, and P. Meer. Human detection via classification on riemannian manifolds. In *CVPR*, 2007.

[22] M. Villamizar, F. Moreno-Noguer, J. Andrade-Cetto, and A. Sanfeliu. Efficient rotation invariant object detection using boosted random ferns, . In *CVPR*, 2010.

[23] M. Villamizar, F. Moreno-Noguer, J. Andrade-Cetto, and A. Sanfeliu. Shared random ferns for efficient detection of multiple categories, . In *ICPR*, 2010.

[24] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001.

[25] B. Wu and R. Nevatia. Cluster boosted tree classifier for multi-view, multi-pose object detection. In *ICCV*, 2007.