Noname manuscript No.
(will be inserted by the editor)

Random Clustering Ferns for Multimodal Object Recognition

M. Villamizar · A. Garrell · A. Sanfeliu · F. Moreno-Noguer

Received: date / Accepted: date

Abstract We propose an efficient and robust method for the recognition of objects exhibiting multiple intra-class modes, where each one is associated to a particular object appearance. The proposed method, called Random Clustering Ferns (RCFs), combines synergically a single and real-time classifier, based on the boosted assembling of extremely-randomized trees (ferns), with an unsupervised and probabilistic approach in order to recognize efficiently object instances in images and discover simultaneously the most prominent appearance modes of the object through tree-structured visual words. In particular, we use Boosted Random Ferns (BRFs) and probabilistic Latent Semantic Analysis (pLSA) to obtain a discriminative and multimodal classifier that automatically clusters the response of its randomized trees in function of the visual object appearance.

The proposed method is validated extensively in synthetic and real experiments, showing that the method is capable of detecting objects with diverse and complex appearance distributions in real-time performance.

Keywords recognition · random trees · plsa · boosting

1 Introduction

Computer vision is becoming a very active research field, especially for the tasks of image understanding and object recognition in images and videos. There exist a vast amount of methods that are able to detect and identify objects in images with striking results, in spite of diverse factors that make difficult this problem such as lighting changes, scaling, cluttered backgrounds, object deformations, general 3D rotations, and intra-class variations [10,11,15,26,38].

Machine learning techniques such as support vector machines [8,11,26,47], boosting [1,16,38,39], and more recently convolutional neural networks [10,15,23] are widely used to compute robust and discriminative classifiers for object detection. However, most of these works are computationally expensive and unfeasible for real-time applications. The reasons lie in using complex algorithms, costly features, and large amounts of training data to compute the object classifiers. Additionally, the time taken to compute these classifiers is usually quite long, in the range of hours or even weeks.

Some methods that are able to compute and test object classifiers in a relatively short time are those based on binary decision trees, such as Random Forest [5,4,6,13,36] and Random Ferns [19,22,31]. These tree-structured classifiers have shown outstanding results in the past, especially in terms of efficiency and reliability. In particular, Random Ferns (RFs) have been used for learning and detecting object instances in real time since they are extremely-randomized trees containing sets of binary features (Boolean comparisons between two pixels values) computed at random [6,19,27,31].

Subsequently, Boosted Random Ferns (BRFs) were introduced to efficiently learn and detect object classes under intra-class appearance changes [35,39,44]. The BRFs improve the detection performance of RFs since the most discriminative ferns are selected via boosting to compute the object classifier. This is done by an iterative and adaptive process in which the ferns are trained using a distribution of misclassification weights over the training samples [34].

Although the BRFs classifier has shown remarkable results to detect objects with various intra-class appear-

M. Villamizar, A. Garrell, A. Sanfeliu and F. Moreno
Institut de Robòtica i Informàtica Industrial, CSIC-UPC.
E-mail: {mvillami,agarrell,sanfeliu,fmoreno}@iri.upc.edu

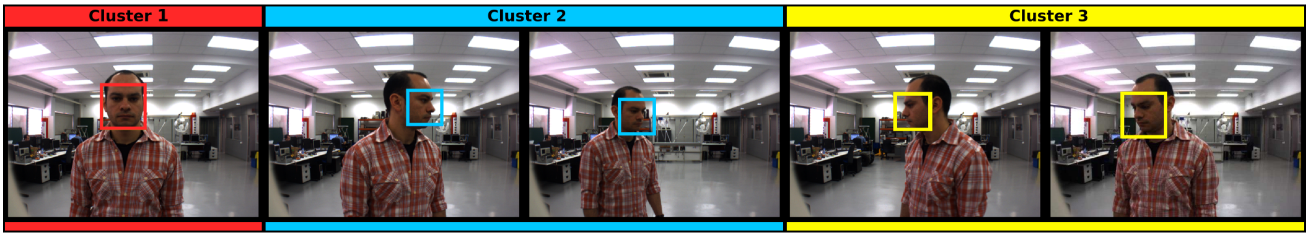

Fig. 1 Example images showing face detection and pose estimation using Random Clustering Ferns (RCFs). The output of RCFs are shown by squared boxes (face detections), whereas the color of each box indicates the face pose. The proposed method is able to simultaneously detect faces and distinguish three different intra-class appearances (face views).

ance modes (e.g, objects seen from different views), it is unable to distinguish these modes automatically. For this purpose, approaches based on the computation of independent and specialized classifiers have been proposed, where each one is devoted to a particular appearance cluster[1] [20,11,26,32,43]. Nevertheless, these methods increase the complexity and computational cost of the object detector because they require multiple classifiers to be computed and tested in the training and run-time steps, respectively. In addition, computing these classifiers using a supervised learning strategy carries a high annotation cost since each sample image (including the object) in the training set requires to be annotated with a specific object appearance mode (e.g, an object view). This annotation task is usually carried out by trained humans, being very time-consuming and costly.

In contrast to more complex and computationally expensive approaches, we present a straightforward and efficient method to recognize objects and discover their intra-class appearance modes (i.e, multimodal object recognition) using Random Clustering Ferns (RCFs). More precisely, RCFs integrate the Boosted Random Ferns (BRFs) classifier [39,44] with probabilistic Latent Semantic Analysis (pLSA) [3,17,37] in order to automatically cluster the different appearances of the object according to the responses of the random ferns over the training data. This particular combination allows detecting objects efficiently and distinguishing simultaneously different appearance modes of the object using a single classifier and without the need of human annotations during the training. This is shown in Fig. 1 where RCFs are applied for face detection. We see how the proposed method is able to detect faces and discriminate three appearance clusters (colored boxes), which correspond to different views of the face.

The proposed approach using weakly-supervised learning consists of three main stages, see Fig. 2. In the initial stage (*object tracking*), an online and fast classifier [14,40,41] is built progressively in order to detect and track the object through a sequence of images containing the object under different visual appearance (observe Fig. 2 (a,b)). This process is automatic and requires only the assignment of the object in the first frame using a bounding box (magenta box). The result of this step is a set of training samples (images) of the object with varying appearance (Fig. 2 (c)). In the second stage (*classifier*), the BRFs are used to compute a more robust object classifier using the set of training samples (see Fig. 2 (d)). Finally, in the third step (*clustering*), the training samples are clustered according to their visual appearance. This is done via pLSA over the response of the BRFs classifier (Fig. 2 (e)). The appearance clusters (latent variables) are found by seeking strong visual similarities among training samples. Fig. 2 (f) shows as example three clusters of training samples.

The method we present is an extension of the work proposed in [42] for multimodal object recognition. However, in this paper we describe the method in more depth and provide additional experiments that contribute to validate our contributions. This paper is also a further step of the approach proposed in [14,40] for learning and detecting objects using human-robot interactions. Indeed, this approach corresponds to the *tracking* stage in Fig. 2. In the present paper, we extend this approach with BRFs and pLSA so as to recognize multiple object appearances. This is particular convenient for robotics applications where knowing a specific object view allows to take actions. For example, for human-robot interaction is important to determine whether people look at the robot. This is possible using RCFs since they allow to detect human faces and estimate their view (refer to Fig. 2).

The remaining of this paper is organized as follows. Sec. 2 provides the related work and our contributions, while Sec. 3 describes the RCFs and their main components. In Sec. 4 the proposed method is validated trough a set of experiments conducted in synthetic and real scenarios. Finally, Sec. 5 concludes the paper with a summary of the achieved goals.

[1] We use interchangeably the terms cluster and mode to refer to a dense part of the object appearance distribution.

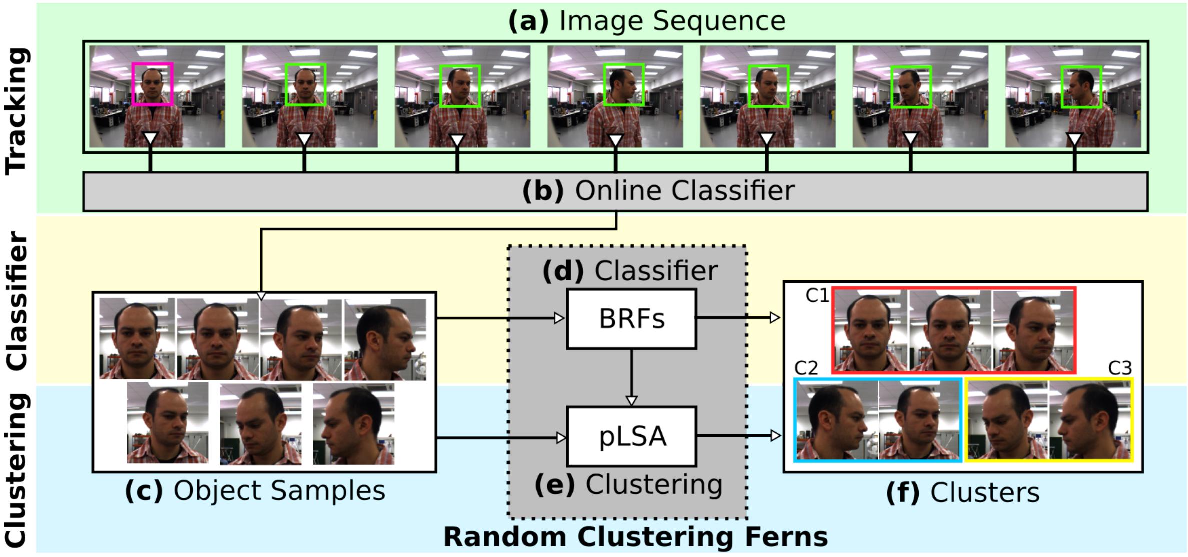

Fig. 2 Overall scheme of the proposed approach to compute object classifiers using weakly supervised learning. In this approach only the first frame is annotated manually. For clarity, this figure does not include background (negative) samples.

2 Related Work and Contributions

We next review the works that are more closely related to the proposed method and describe our contributions. Concretely, we focus on those works concerned with clustering data using random decision trees.

Among diverse works based on clustering trees [2, 24, 28], the most similar work to RCFs is the approach presented in [28] where Extremely Randomized Clustering Forests (ERC-Forests) are used for fast image classification and image search using support vector machines [29] and bags of visual words [7, 18, 30]. The efficiency of ERC-Forests lies in using the randomized forests as clustering trees in which each leaf of the tree corresponds to a particular cluster and visual word. This speeds up considerably the computation of visual words in comparison to more conventional quantization methods based on K-means [18] and complex descriptors like SIFT [25], especially in high-dimensional spaces.

In the same spirit, we use the leaves of the randomized trees as fast visual words but with two major differences. The first one is using ferns instead of random forest with the aim of reducing the complexity and computational cost of the classifier. The second difference is a shift in the concept of clustering. Rather than considering each leaf of a tree as a specific cluster, what leads to a large number of redundant clusters since multiple ferns are computed, we apply pLSA over the visual words (i.e, ferns leaves) to obtain more compact and significant clusters. This also can be seen as discovering strong visual similarities (intra-class appearance modes) from the co-occurrence of tree-structured visual words in the training samples. Fig. 3 shows a demo example. Note that the clusters have distinctive visual words distributions.

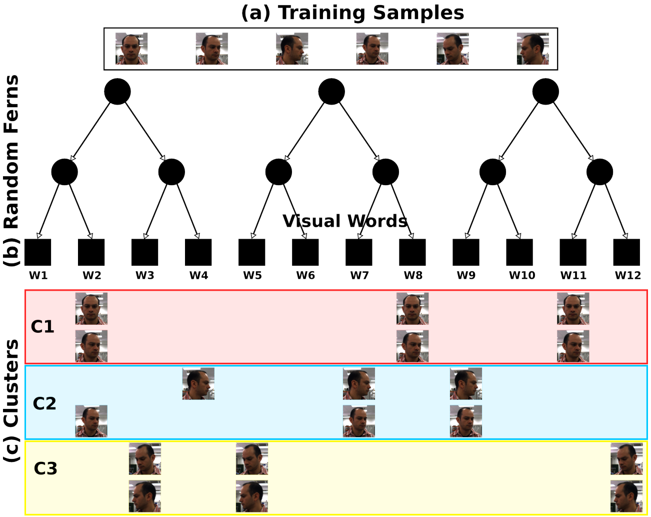

Fig. 3 Clustering of training samples (a) using three random ferns (b) in the BRFs classifier. The samples are grouped in three distinct clusters (c) using pLSA over the visual words.

This idea is similar to the method presented in [33] for video segmentation. In this method the clustering is done via Markov clustering algorithm on a graph connecting all the partitions given by the trees. Similar to RCFs, this work exploits the fact that randomized trees give overlapped partitions which can be merged subsequently to obtain more compact clusters (Fig. 3).

Other method related to RCFs is Cluster Boosted Tree (CBT), proposed in [46], and used for object detection from multiple views. Like RCFs, CBT does not use

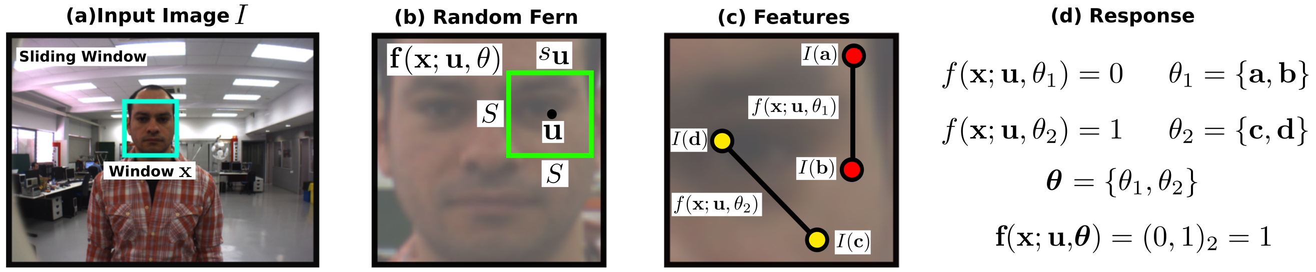
(a)Input Image I
Sliding Window
Window x
(b) Random Fern
$\mathbf{f}(\mathbf{x};\mathbf{u},\theta)$
$s_\mathbf{u}$
S
$\mathbf{u}$
S
(c) Features
$I(\mathrm{a})$
$f(\mathbf{x};\mathbf{u},\theta_1)$
$I(\mathrm{d})$
$I(\mathrm{b})$
$f(\mathbf{x};\mathbf{u},\theta_2)$
$I(\mathrm{c})$
(d) Response
$f(\mathbf{x};\mathbf{u},\theta_1)=0 \quad \theta_1=\{\mathbf{a},\mathbf{b}\}$
$f(\mathbf{x};\mathbf{u},\theta_2)=1 \quad \theta_2=\{\mathbf{c},\mathbf{d}\}$
$\boldsymbol{\theta}=\{\theta_1,\theta_2\}$
$\mathbf{f}(\mathbf{x};\mathbf{u},\boldsymbol{\theta})=(0,1)_2=1$

Fig. 4 Computation of a random fern in an input image. (a) The classifier is evaluated for every local window $\mathbf{x}$ inside the image I using a sliding window approach. (b) The fern $\mathbf{f}(\mathbf{x};\mathbf{u},\boldsymbol{\theta})$ is tested in the window $\mathbf{x}$ at position $\mathbf{u}$ and with features parameters $\boldsymbol{\theta}$. (c) Fern consisting of two binary features whose parameters θ_1 and θ_2 are chosen at random. (d) Response of the fern $\mathbf{f}(\mathbf{x};\mathbf{u},\boldsymbol{\theta})$ as a combination of its binary features.

predefined intra-class subcategorization based on domain knowledge but that it automatically clusters the samples through a boosted tree. Although this method uses a single tree and has shown very good results, CBT is computationally more expensive since the construction of the tree involves applying boosting at each tree node. Conversely, RCFs are computed in a few minutes because the multiple trees are built at random and combined afterwards using boosting. The clustering step is conducted once the BRFs classifier has been computed.

3 Random Clustering Ferns

In this section, we describe in detail the ingredients of the Random Clustering Ferns (RCFs). Fig. 2 shows a general scheme of RCFs and their main components. Firstly, we briefly introduce the computation of random ferns in Sec. 3.1 since they are at the core of the proposed method. Sec. 3.2 explains the tracking step using an online classifier to extract training samples. Sec. 3.3 comments the computation of the BRFs classifier with the training samples. Lastly, the clustering step through pLSA and BRFs is described in Sec. 3.4.

3.1 Random Ferns

Random Ferns (RFs) are a particular version of Random Forests which have shown great success for object detection in real-time applications, thanks to their simplicity and fast computation [19,22,31,40,41].

More formally, each fern $\mathbf{f}$ consists of a set of M local binary features f where each feature captures the difference between the values of two image pixels chosen at random. That is,

$$\mathbf{f}(\mathbf{x};\mathbf{u},\boldsymbol{\theta})=[f(\mathbf{x};\mathbf{u},\theta_1),\ldots,f(\mathbf{x};\mathbf{u},\theta_M)]\ , \tag{1}$$

where $\mathbf{x}$ is an image window in the input image I (see Fig. 4 (a)), and $\mathbf{u}$ and $\boldsymbol{\theta}$ are the fern parameters. The former defines the center of the subwindow $s_\mathbf{u}$, with size $S\times S$, where the fern is computed within the window $\mathbf{x}$ (observe Fig. 4 (b)). The second parameter $\boldsymbol{\theta}=\{\theta_1,\ldots,\theta_M\}$ corresponds, in turn, to the set of parameters of the binary features.

Each binary feature f is computed as

$$f(\mathbf{x};\mathbf{u},\theta)=\mathbb{I}(I(\mathbf{p})>I(\mathbf{p}')), \tag{2}$$

where $\theta=\{\mathbf{p},\mathbf{p}'\}$ are the random pixel positions in $s_\mathbf{u}$, $I(\mathbf{p})$ is the pixel value at position $\mathbf{p}$, and $\mathbb{I}(e)$ refers to the indicator function[2]. Fig. 4 (c) shows a demo example where a fern consisting of two binary features is computed in the subswindow $s_\mathbf{u}$. Note that each feature compares the image intensity values between two pixels selected randomly.

The response of the fern is a M-dimensional binary vector that is commonly represented by an integer value $z\in[0,\ldots,2^M-1]$. This response is calculated using the co-occurrence of the individual features outputs. This is illustrated in Fig. 4 (d) where the response of a fern with features outputs $[0,1]$ is calculated by $z=(01)_2=1$.

3.2 Object Tracking

The goal of this stage is to extract automatically a set of training samples which are used later to compute the BRFs classifier (see Fig. 2 (c)). These training samples forms the training data $\mathcal{D}$ which in turn consists of positive and negative images containing object instances and background patches, respectively.

To efficiently track the object in every frame, we compute an online classifier based on extremely randomized trees. Specifically, and in order to maintain consistency with other components of the presented method, we use Online Random Ferns (ORFs) which have shown very good in terms of efficiency and detection rates in the past [31,40,41].

[2] The indicator function $\mathbb{I}(e)=1$ if e is true, and 0 otherwise.

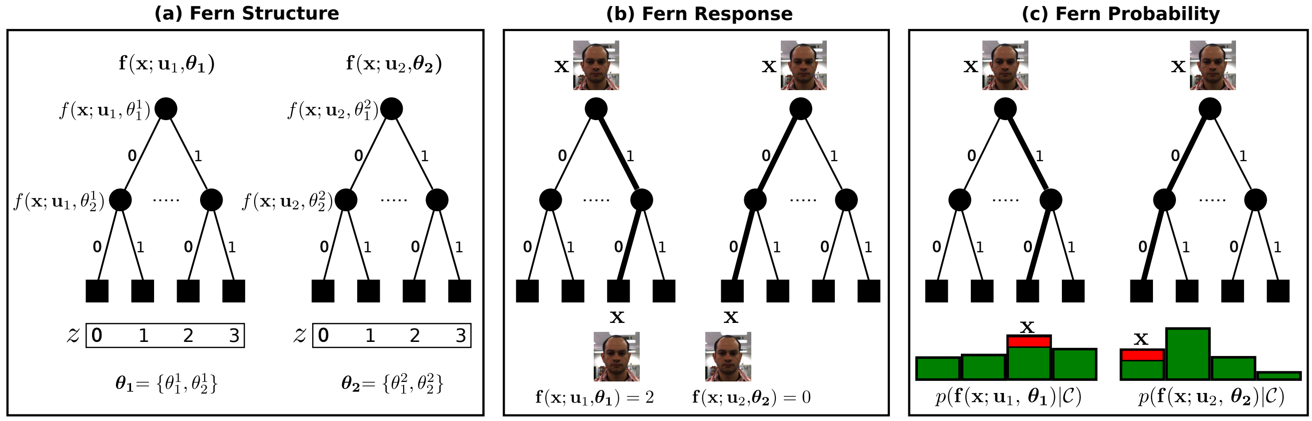
(a) Fern Structure
$\mathbf{f}(\mathbf{x};\mathbf{u}_1,\boldsymbol{\theta}_1)$
$\mathbf{f}(\mathbf{x};\mathbf{u}_2,\boldsymbol{\theta}_2)$
$f(\mathbf{x};\mathbf{u}_1,\theta_1^1)$
$f(\mathbf{x};\mathbf{u}_1,\theta_2^1)$
$f(\mathbf{x};\mathbf{u}_2,\theta_1^2)$
$f(\mathbf{x};\mathbf{u}_2,\theta_2^2)$
0
1
z 0 1 2 3
$\boldsymbol{\theta}_1 = \{\theta_1^1, \theta_2^1\}$
$\boldsymbol{\theta}_2 = \{\theta_1^2, \theta_2^2\}$
(b) Fern Response
x
$\mathbf{f}(\mathbf{x};\mathbf{u}_1,\boldsymbol{\theta}_1)=2$
$\mathbf{f}(\mathbf{x};\mathbf{u}_2,\boldsymbol{\theta}_2)=0$
(c) Fern Probability
$p(\mathbf{f}(\mathbf{x};\mathbf{u}_1,\boldsymbol{\theta}_1)|\mathcal{C})$
$p(\mathbf{f}(\mathbf{x};\mathbf{u}_2,\boldsymbol{\theta}_2)|\mathcal{C})$

Fig. 5 Online random ferns for object tracking. (a) Tree-structured representation of two ferns having two binary features. (b) Ferns responses on an input sample $\mathbf{x}$. (c) Updating ferns probabilities with the sample $\mathbf{x}$ (assuming this sample is positive).

The procedure to compute this online classifier is as follows: firstly, the human user provides one object annotation in the first frame of the image sequence (see magenta box in Fig. 2 (a)) to initialize the classifier with a set of image patches extracted around the annotated object using small deformations like image shifts and scale changes. Additionally, random image patches from the background are also collected to feed the classifier with negative samples.

Subsequently, the classifier is incrementally computed through the image sequence using a self-learning approach, in which detection and updating steps are carried out jointly, both to extract training samples and update the object classifier. Further in detail, for a given input image I, the classifier is tested at every image window $\mathbf{x} \subset I$ using a sliding window strategy (see Fig. 4 (a)). This procedure is also applied for multiple image resolutions so as to deal with scale changes. The result of this detection step is a set of potential object hypotheses (image windows) that corresponds to the training images. Then, these images are labeled automatically as positive or negative samples in accordance to the classifier confidence, and used to update the online classifier. In this way, the classifier is continuously improved with new input images while it detects object instances under varying environmental and imaging conditions.

More precisely, the online classifier $T(\mathbf{x})$ for object tracking is defined by

$$T(\mathbf{x}) = \begin{cases} +1 \text{ if } \mathrm{conf}_T(\mathbf{x}) > \beta_T \\ -1 \text{ otherwise,} \end{cases} \tag{3}$$

where $\mathbf{x} \in \mathrm{I\!R}^2$ refers to the image window, β_T is the classifier threshold (set at 0.5 by default), and $\mathrm{conf}_T(\mathbf{x})$ is the confidence of the online classifier for the sample $\mathbf{x}$. Therefore, if the output of the classifier is $T(\mathbf{x}) = +1$, the sample $\mathbf{x}$ is assigned to the positive or object class $\mathcal{C}$. Otherwise, this sample is considered as negative and assigned to the background class $\mathcal{B}$.

The confidence of the online classifier comprises the probabilities of a series of R random ferns,

$$\mathrm{conf}_T(\mathbf{x}) = \frac{1}{R} \sum_{r=1}^{R} \frac{p(\mathbf{f}|\mathcal{C})}{p(\mathbf{f}|\mathcal{C}) + p(\mathbf{f}|\mathcal{B})}, \tag{4}$$

where $p(\mathbf{f}|\{\mathcal{C},\mathcal{B}\}) = p\left(\mathbf{f}(\mathbf{x};\mathbf{u}_r,\boldsymbol{\theta}_r) = z|\{\mathcal{C},\mathcal{B}\}\right)$ are the class-conditional probabilities of the fern $\mathbf{f}(\mathbf{x};\mathbf{u}_r,\boldsymbol{\theta}_r)$ with response z, and $\frac{1}{R}$ is a normalization factor to establish the classifier confidence in the range $[0,1]$. The fern parameters $\mathbf{u}_r \in \{u_1,\ldots,u_L\}$ and $\boldsymbol{\theta}_r \in \{\theta_1,\ldots,\theta_P\}$ are selected at random, being $\{u_1,\ldots,u_L\}$ the set of all possible locations in the window $\mathbf{x}$, and $\{\theta_1,\ldots,\theta_P\}$ a small pool of features parameters that is used to reduce the computational cost [40,41]

In Fig. 5 (a,b), we show an example of the structure and responses of two ferns on an input sample $\mathbf{x}$. Contrary to Fig. 4, here, the structure of ferns are represented as binary decision trees. Observe that the fern response z depends of the binary features outputs. The co-occurrence of these features determines the fern response z, that is associated to the tree leaf where the sample falls.

The online learning of the classifier $T(\mathbf{x})$ is carried out by updating the ferns probabilities with new input samples (potential object hypotheses). For each input sample $\mathbf{x}$, the classifier first determines its class label according to its confidence on this sample (refer to Eq. 3). Subsequently, the class-conditional probabilities, $p\left(\mathbf{f}(\mathbf{x};\mathbf{u}_r,\boldsymbol{\theta}_r)|\mathcal{C}\right)$ and $p\left(\mathbf{f}(\mathbf{x};\mathbf{u}_r,\boldsymbol{\theta}_r)|\mathcal{B}\right)$, are updated according to whether the sample belongs to either the positive or negative class. To this end, we make use of 2^M-dimensional histograms to represent these fern probabilities (empirical distributions). For example, if the sample is considered positive, the z-th bin of the histogram $p\left(\mathbf{f}(\mathbf{x};\mathbf{u}_r,\boldsymbol{\theta}_r)|\mathcal{C}\right)$ is increased in one unit. This is illustrated in Fig. 5 (c), where the input sample is used to update the fern distributions.

3.3 The Object Classifier

Once the training data, consisting of positive and negative image samples, has been obtained from the sequence of images (see Fig. 2 (c)), the next stage of the proposed method is the computation of an efficient and discriminative classifier which recognizes objects with multiple intra-class modes (Fig. 2 (d)). For this purpose, we resort to Boosted Random Ferns (BRFs) since they have demonstrated to be able to detect objects robustly and with low computational cost [44,39].

The reason that motivates the use of the BRFs classifier instead of the ORFs classifier, presented in the previous section for object tracking, lies in BRFs are more discriminative than ORFs whilst maintain the fast computation of ferns [39]. Particularly, BRFs uses a boosting algorithm in order to find automatically the fern parameters $\mathbf{u}_r \in \{\mathbf{u}_1, \ldots, \mathbf{u}_L\}$ and $\boldsymbol{\theta}_r \in \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p\}$ that best discriminate the object class (positive) from the background one (negative) in an iterative learning scheme using the training data. This differs largely of ORFs that selects all ferns parameters at random without considering the set of training samples.

More formally, and similar to ORFs, the BRFs classifier $H(\mathbf{x})$ for object detection is computed by

$$H(\mathbf{x}) = \begin{cases} +1 \text{ if } \mathrm{conf}_H(\mathbf{x}) > \beta_H \\ -1 \text{ otherwise}, \end{cases} \tag{5}$$

where $\mathbf{x}$ is the input sample, β_H is a threshold that determines the classifier tolerance, and $\mathrm{conf}_H(\mathbf{x})$ is the confidence of the classifier on the sample $\mathbf{x}$. Once again, if the output of the classifier is $H(\mathbf{x}) = 1$, the sample $\mathbf{x}$ is considered as a positive (object) sample. Otherwise, this sample belongs to the negative or background class.

As it is standard in boosting, we define the confidence of the BRFs classifier using a combination of R weak learners h_r,

$$\mathrm{conf}_H(\mathbf{x}) = \sum_{r=1}^{R} h_r(\mathbf{x}), \tag{6}$$

where each weak learner is computed as:

$$h_r(\mathbf{x}) = \frac{1}{2} \log \frac{p(\mathbf{f}(\mathbf{x}; \mathbf{u}_r, \boldsymbol{\theta}_r)|\mathcal{C}) + \epsilon}{p(\mathbf{f}(\mathbf{x}; \mathbf{u}_r, \boldsymbol{\theta}_r)|\mathcal{B}) + \epsilon}, \tag{7}$$

being $p(\mathbf{f}(\mathbf{x}; \mathbf{u}_r, \boldsymbol{\theta}_r)|\{\mathcal{C}, \mathcal{B}\})$ the class-conditional probabilities of fern $\mathbf{f}(\mathbf{x}; \mathbf{u}_r, \boldsymbol{\theta}_r)$, and ϵ a small constant.

In order to obtain the values of $\mathbf{u}_r$ and $\boldsymbol{\theta}_r$ for each weak learner h_r that most discriminate the object class from the background, BRFs use Real AdaBoost [34] to iterate over all image locations $\mathbf{u}_l = \{\mathbf{u}_1, \ldots, \mathbf{u}_L\}$ and the small set of features parameters $\boldsymbol{\theta}_p = \{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_P\}$ to find those values where the score of h_r gets larger. To this end, BRFs use a distribution w_r of misclassification weights over the training samples that is updated at each boosting iteration r according to the performance of the weak learners on the training data [34,39].

More specifically, boosting evaluates each fern with parameters $\mathbf{u}_l$ and $\boldsymbol{\theta}_p$ over the whole training set to compute the class-conditional probabilities with

$$p(\mathbf{f}(\mathbf{x}; \mathbf{u}_l, \boldsymbol{\theta}_p) = z|\mathcal{C}) = \sum_{i:\mathbf{f}(\mathbf{x}_i;\mathbf{u}_l,\boldsymbol{\theta}_p)=z \ \wedge \ y_i=+1} w_r(\mathbf{x}_i) \tag{8}$$

$$p(\mathbf{f}(\mathbf{x}; \mathbf{u}_l, \boldsymbol{\theta}_p) = z|\mathcal{B}) = \sum_{i:\mathbf{f}(\mathbf{x}_i;\mathbf{u}_l,\boldsymbol{\theta}_p)=z \ \wedge \ y_i=-1} w_r(\mathbf{x}_i) \tag{9}$$

where $i = 1, 2, \ldots, N$, being N the number of training samples, $y_i = \{+1, -1\}$ denotes the object and background class labels for the window sample $\mathbf{x}_i \in \mathcal{D}$, and $w_r(\mathbf{x}_i)$ is the sample weight at round r.

Next, the Bhattacharyya coefficient Q is used to select the fern parameters $\mathbf{u}_r$ and $\boldsymbol{\theta}_r$ as those values $\mathbf{u}_l$ and $\boldsymbol{\theta}_p$ that minimize the training error. This can be formulated as:

$$Q(\mathbf{u}_l, \boldsymbol{\theta}_p) = 2 \sum_{z=0}^{2^M-1} \sqrt{p(z|\mathcal{C}) \ p(z|\mathcal{B})} \ , \tag{10}$$

$$(\mathbf{u}_r, \boldsymbol{\theta}_r) = \arg\min_{\mathbf{u}_l, \boldsymbol{\theta}_p} Q(\mathbf{u}_l, \boldsymbol{\theta}_p). \tag{11}$$

where $p(z|\{\mathcal{C}, \mathcal{B}\})$ stands for $p(\mathbf{f}(\mathbf{x}; \mathbf{u}_l, \boldsymbol{\theta}_p) = z|\{\mathcal{C}, \mathcal{B}\})$.

Finally, at the end of the iteration, this weak learner is used to update the weights associated to the image window samples

$$w_{r+1}(\mathbf{x}_i) = \frac{w_r(\mathbf{x}_i) \ \exp(-y_i h_r(\mathbf{x}_i))}{\sum_{j=1}^{N} w_r(\mathbf{x}_j) \ \exp(-y_j h_r(\mathbf{x}_j))} \tag{12}$$

This updating rule increases the weight for the incorrectly classified samples and decreases the weight for the correctly classified samples, with the purpose of focusing the next weak learner $r + 1$ on the misclassified samples.

Finally, once the BRFs classifier has been computed during the training phase, this classifier can be evaluated in run time to detect object instances in images. Similar to tracking stage, the classifier is tested densely over each input image using a sliding window strategy, and tested for multiple image scales to cope with scale changes.

To illustrate the performance of the classifier in a clear and simple manner, we show in Fig. 6 (a,b) the

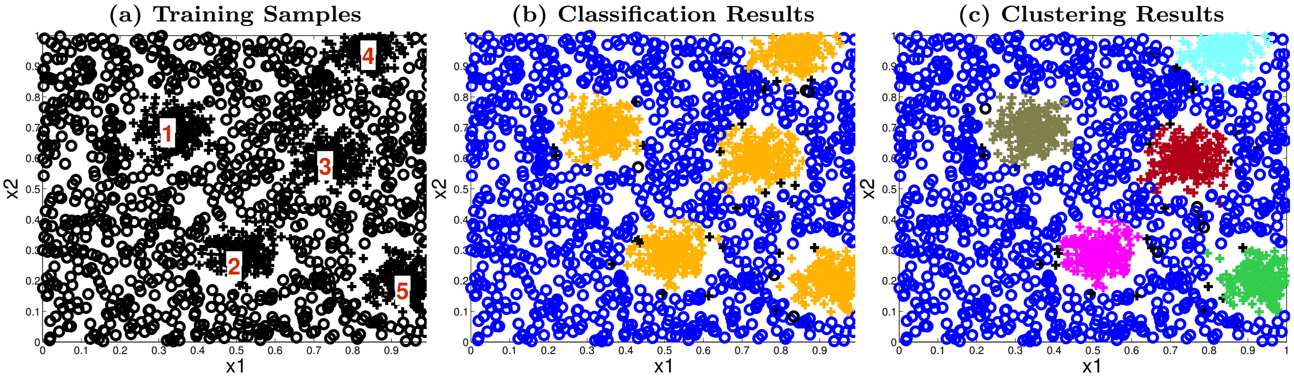

Fig. 6 RCFs for a 2D classification problem. (a) Positive (crosses) and negative (circles) samples used to compute the BRFs classifier (1000 positive and negative samples). (b) Classification results provided by BRFs. Yellow samples refer to the positive class, whereas blue ones are negative samples. Misclassified samples are shown in black. (c) Output of the proposed RCFs. Each color indicates a different cluster of the positive class distribution.

classification results of BRFs for a 2D classification problem consisting of 1000 positive and negative samples. Positive samples are indicated by crosses, whereas the negative samples are represented by circles. Fig. 6 (a) shows the complex class distributions. Note that while the positive samples are distributed in five intra-class clusters, the negative samples are spread at random over the entire 2D feature space.

For this particular problem, instead of the comparisons between two pixels values (see Eq. 2), we use 2D decision stumps (axis-aligned split functions) as binary features, where each decision stump f maps a given sample $\mathbf{x} \in [0,1] \times [0,1]$ to a Boolean label, $f(x) = \mathbb{I}(x_j > \tau)$, where j corresponds to a specific (horizontal or vertical) coordinate of $\mathbf{x}$, and τ is a random threshold in the interval $[0,1]$.

Fig. 6 (b) shows the output of BRFs for discerning the positive class from the negative one. Samples in yellow color make reference to samples correctly labeled as positive samples. Similarly, blue samples denote negative samples. Black samples correspond to samples wrongly classified by BRFs. Observe that these misclassified samples are a small portion of the whole set of samples. Contrary, BRFs are able to classify correctly most samples with high accuracy and precision rates (rates of 98%) in spite of the complexity present in this problem like having multiple positive clusters and a narrow separation between classes.

3.4 Clustering

In the past section, we have seen that BRFs are able to compute a robust classifier that can cope with various distribution clusters and with non-linearly separable class distributions. However, this classifier can not distinguish among these clusters in order to detect the object and recognize a particular object appearance. For the example shown in Fig. 1, BRFs focus only on detecting faces under multiple views, but not on discerning among them.

With the aim of finding important internal structures of the object class without human supervision or computing multiple classifiers, we propose to use probabilistic Latent Semantic Analysis (pLSA) to discretize the overall appearance of the object into multiple clusters of samples with a strong feature similarity. Actually, this corresponds to the third stage of the proposed method, see Fig. 2 (e), where the output of the BRFs classifier is clustered in function of the object appearance. This is exemplified in Fig. 3.

Specifically, pLSA is a generative model from the statistical text literature that allows discovering latent or hidden topics from a corpus containing co-occurrences between documents and words [3,17,37].

In this work, we follow the same idea, but instead of documents and words we use images and tree-structured visual words, respectively, to find automatically the most relevant clusters of the object appearance (topics). The pLSA algorithm is suitable for this problem because it provides a statistical model that allows represent an object sample $\mathbf{x}_i$ as a mixture of K topics,

$$p(w_j|\mathbf{x}_i) = \sum_{k=1}^{K} p(\mathcal{T}_k|\mathbf{x}_i)p(w_j|\mathcal{T}_k), \qquad (13)$$

where the number of topics K is defined a priori, $p(\mathcal{T}_k|\mathbf{x}_i)$ is the probability of topic $\mathcal{T}_k$ occurring in the sample $\mathbf{x}_i$, and $p(w_j|\mathcal{T}_k)$ is the probability of the visual word w_j occurring in the topic $\mathcal{T}_k$ [17,37].

For our case, we use the object samples extracted by the online classifier, and define that each fern leaf j corresponds to a particular visual word w_j since it encodes

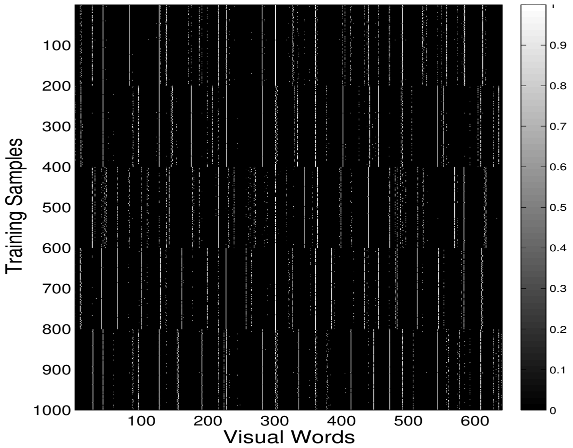

Fig. 7 Binary corpus including the co-occurrence between training samples and visual words. White values represent that a visual word occurs in a given training sample.

a specific visual appearance determined by the configuration of binary features. This is shown in Fig. 3.

The parameters of the model, $p(\mathcal{T}_k|\mathbf{x}_i)$ and $p(w_j|\mathcal{T}_k)$, are estimated via likelihood estimation. This is a non-convex optimization that is commonly solved using the Expectation-Maximization (EM) algorithm over the log-likelihood of the objective function L,

$$\log L = \sum_{i=1}^{N}\sum_{j=1}^{W} n(\mathbf{x}_i, w_j) \log \sum_{k=1}^{K} p(w_j|\mathcal{T}_k)p(\mathcal{T}_k|\mathbf{x}_i), \quad (14)$$

where N is the number of training samples, W is the amount of visual words (compute by $W = R \times 2^M$), and $n(\mathbf{x}_i, w_j)$ is the number of times that the visual word w_j appears in the document $\mathbf{x}_i$.

To discover the latent topics, pLSA requires an input table (corpus in the text analysis literature) of size $N \times W$ containing the co-occurrence of training samples and the bag of visual words. Fig. 7 shows this table for a 2D problem example in which the positive samples have been ordered by cluster in order to distinguish visually strong patterns in the corpus. We can differentiate five different patterns corresponding to the clusters. Notice also that the table has binary values since $n(\mathbf{x}_i, w_j) \in \{1, 0\}$ and that indicates the presence or not of a particular visual word w_j in the sample $\mathbf{x}_i$. This is similar to the example exposed in Fig. 3.

Once the parameters $p(\mathcal{T}_k|\mathbf{x}_i)$ and $p(w_j|\mathcal{T}_k)$ have been estimated during the training step, we use $p(\mathcal{T}_k|\mathbf{x}_i)$ to discover the intra-class mode (object appearance) of a test sample $\mathbf{x}$ in run time. This can be written as:

$$t = \arg\max_{\mathcal{T}_k} \quad p(\mathcal{T}_k|\mathbf{x}), \quad k = 1, 2, ..., K \quad (15)$$

where $t \in \{1, \ldots, K\}$ is the index indicating the intra-class mode associated to a topic $\mathcal{T}_t$. For this paper, the test samples are the image windows $\mathbf{x}$ provided by the BRFs classifier for object detection. For example, Fig. 1 depicts these image windows through bounding boxes, whereas the output of pLSA clustering is represented by a color code (each color represent an object appearance). Here, we can see how these two methods (BRFs and pLSA) are appropriately combined to compute the proposed RCFs for multimodal object recognition.

With respect to the 2D classification problem presented before, Fig. 6 (c) shows the output of the clustering stage over the 2D training samples. Observe that the positive samples are clustered in K=5 different clusters, indicated again through different colors, and that each one keeps strong feature correlation in the 2D space.

4 Experiments

In this paper, we evaluate qualitatively and quantitatively the proposed RCFs in various synthetic and real experiments in which their main contributions are highlighted.

4.1 Synthetic Experiments - 2D Classification Problem

The proposed method has been evaluated in synthetic experiments in order to observe more clearly the performance of the RCFs and their main constituents (BRFs and pLSA). Fig. 8 shows, for instance, the output of the proposed approach on a scenario generated at random in which two class distributions (positive and negative) with high complexity are considered. For this experiment, the RCFs are computed using $K = 5$ intra-class clusters (topics).

We observe in Fig. 8 (a) that the proposed method achieves correctly classify most samples while discovers multiple intra-class modes (indicated by means of clusters with different colors). The method only produces a small number of misclassified samples (black samples). This result is also shown in the precision-recall curve, see Fig. 8 (b), where RCFs obtain a high Equal Error Rate (EER)[3]. This proves that the proposed method, and especially the BRFs classifier, is able to robustly classify positive samples, under a complex and multimodal distribution, from a negative class which contains a large number of negative samples spread over the whole 2D feature space. Furthermore, the method also increases the separability between classes while reduces the risk of misclassification. This is seen in Fig. 8 (c)

[3] The Equal Error Rate (EER) is the point in the precision-recall curve where precision=recall.

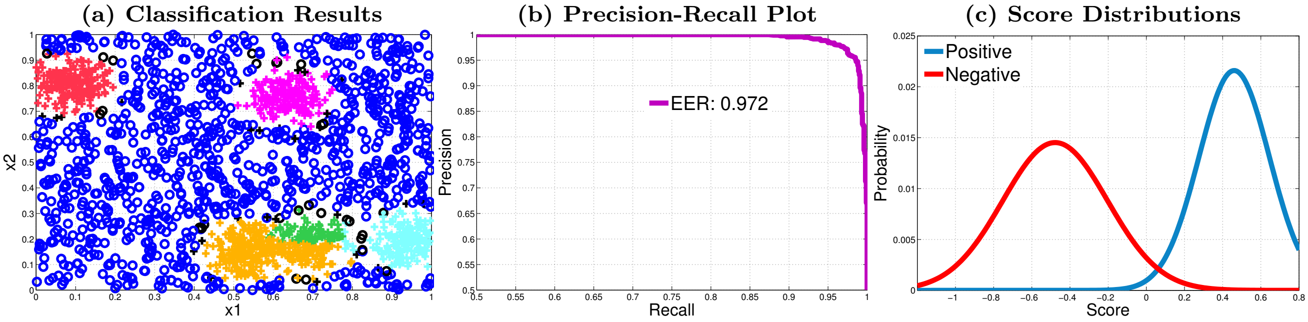

Fig. 8 2D classification performance of RCFs. (a) Output of the proposed method to classify sets of positive and negative samples. In this 2D problem, we indicate positive samples through crosses, whereas negative samples are represented by circles. Samples in black color represent to misclassified samples. (b) Classification plot using recall-precision curve and Equal Error Rate (EER). (c) Class score distributions for the positive and negative classes.

2D Classification Performance														
	RFs			RCFs										
	# Clusters (K)			# Clusters (K)			# Ferns (R)				# Features (M)			
	3	5	10	3	5	10	5	10	20	50	1	3	5	7
EER(%)	90.6	92.4	84.4	96.9	97.4	96.0	95.4	96.6	96.9	97.2	83.1	96.1	96.9	97.2
Distance(%)	59.0	68.8	46.5	83.1	90.6	73.2	77.2	80.4	83.1	88.8	41.2	73.0	83.1	88.4

Table 1 2D classification results for the RCFs and RFs classifiers in function of the numbers of clusters (K), ferns (R) and binary features (M). The RCFs outperform to the RFs classifier, especially as the amount of clusters gets larger.

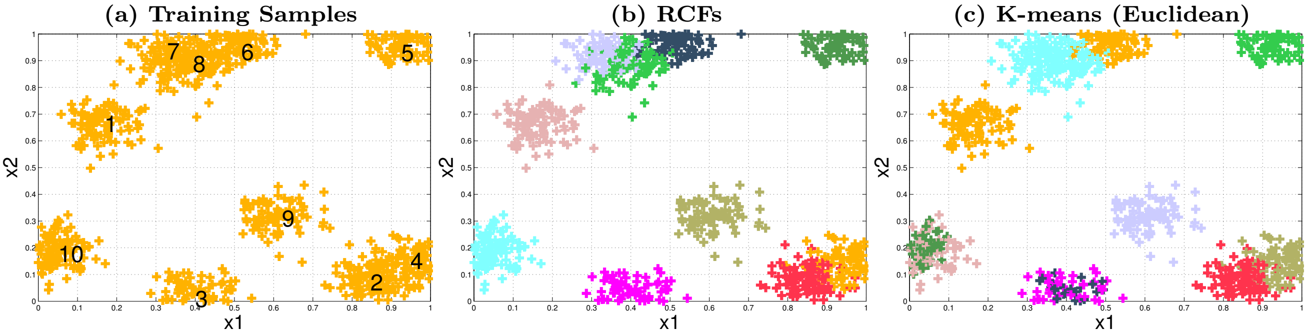

Fig. 9 Synthetic clustering results. (a) Positive training class consisting of 10 intra-class modes. (b) Clustering results provided by RCFs, where each color is associated to a particular cluster. (c) Output of the K-means algorithm using Euclidean distance.

where we plot the score distributions[4] for the positive and negative classes. Notice that RCFs largely separates these two classes.

Quantitatively speaking, Table 1 provides the classification results of RCFs in terms of the numbers of ferns, binary features, and intra-class clusters. We report the average classification rates of RCFs over 10 runs in order to consider the randomness of the BRFs classifier and the 2D scenario. Each scenario is generated at random with multiple sample clusters (K). We see that RCFs obtain high classification scores (EER) and large distances between classes when the amounts of features (M) and ferns (R) get larger. Here, we use the Hellinger distance[5] to measure the separability between Gaussian distributions. The default parameters for this experiment are $K = 5$, $R = 20$ and $M = 5$.

Table 1 also shows a comparison, in function of the number of clusters, of RCFs against the RFs classifier [31,40]. We see that RCFs attain the best performance rates since the classifier uses boosting to select the most discriminative ferns. This improvement is more noticeable as the number of clusters increases.

With regards to the clustering performance, Fig. 9 shows the results of the RCFs in comparison with the K-means algorithm [29]. Positive samples are shown

[4] The score distribution (Gaussian function) is calculated using the confidences of the BRFs for all class samples.

[5] The squared Hellinger distance for two distributions P and Q is defined as: $H^2(P,Q) = 1 - \sqrt{k_1/k_2}\exp(-0.25k_3/k_2)$, with $k_1 = 2\sigma_P\sigma_Q$, $k_2 = \sigma_P^2 + \sigma_Q^2$, and $k_3 = (\mu_P - \mu_Q)^2$.

Clustering Results												
	BRFs+pLSA				**K-means (Euclidean)**				**BRFs+K-means**			
D	2	5	10	20	2	5	10	20	2	5	10	20
K=3	0.097	0.001	0.033	0.000	0.147	0.100	0.133	0.067	0.177	0.036	0.086	0.113
K=5	0.240	0.022	0.019	0.020	0.180	0.139	0.163	0.201	0.304	0.127	0.114	0.173
K=10	0.514	0.144	0.092	0.096	0.367	0.143	0.159	0.116	0.548	0.251	0.207	0.102

Table 2 Average clustering values for RCFs, K-means, and BRFs+K-means.

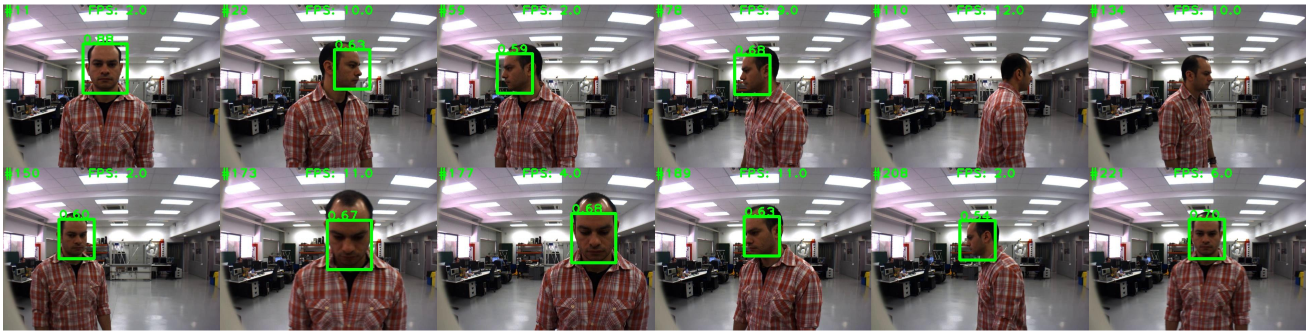

Fig. 10 Example images showing the output of the ORFs classifier for object tracking (Sec. 3.2). Green boxes indicate the output of the online classifier. Small numbers beside boxes correspond to the confidence of the classifier (Eq. 4).

only for the purpose of clarity. The left figure corresponds to the positive training samples belonging to a 20-dimensional feature space. In this figure, we only plot the first two feature dimensions ($\mathbf{x}^1$,$\mathbf{x}^2$). Fig. 9 (b) plots the clustering output of the proposed method (BRFs+pLSA), whereas the right figure shows the results of K-means using Euclidean distance in the feature space. We see that our method yields good clustering results (each cluster has a singular color), in contrast to the K-means algorithm which produces some incorrect clustering labels (observed through the confusion of colors). This is particularly visible for clusters 3 and 10, where K-means assigns multiple labels to each cluster. By contrast, our method produces one-color distribution clusters, which results in a better clustering performance, observe Fig. 9 (b).

Finally, Table 2 shows quantitative results of the clustering performance of the proposed method. More specifically, this table reports the average confusion values among clustering labels (using ground truth labels) for different numbers of clusters (K) and feature space dimensions (D). Here, we use as general measure of clustering error, the entropy function over the confusion matrix using the ground-truth labels vs. the estimated labels. For example, if the method performs clustering correctly, the confusion matrix becomes into the identity matrix and the entropy score is zero. Contrary, if the quality of the clustering is poor, the entropy function yields much larger values. From the table, it is seen that our method (BRFs+pLSA) obtain lower scores than the K-means algorithm, especially whether we consider large feature spaces ($D > 5$). This is due mainly to the curse of dimensionality that affects largely to clustering algorithms like K-means [29].

Table 2 also includes an approach combining BRFs and K-means using the Hamming distance. Similar to pLSA, the K-means algorithm is applied here to the response of the BRFs classifier (ferns leaves) to perform clustering. This contrasts with previous experiments where K-means is applied to the feature space. The goal of this particular experiment is to show that pLSA is convenient for the proposed method and that provides better clustering results. Again, observe that RCFs produce lower confusion values than the BRFs+K-means approach.

4.2 Real Experiments - Face and Object Detection

In this section, the proposed approach has been used to perform multimodal object recognition in diverse real experiments concerning to the detection of faces and objects from multiple views. Particularly, this corresponds to a classification problem involving multiple intra-classes where each one is associated to a specific view of the object or face. Refer to Fig. 1.

Multi-view Face Detection. For face detection, the method uses two image sequences of human faces from the dataset introduced in [40]. Both sequences have more than 200 images acquired by a mobile robot while it interacted with humans in indoor scenarios. For training, we use the the first sequence, whereas the second one is used for validation.

Fig. 10 shows example images of the output of the tracking step on the training sequence. Observe how

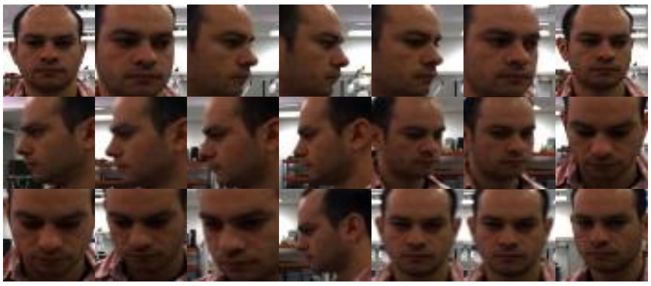

Fig. 11 Positive training samples acquired during the tracking step by the ORFs classifier.

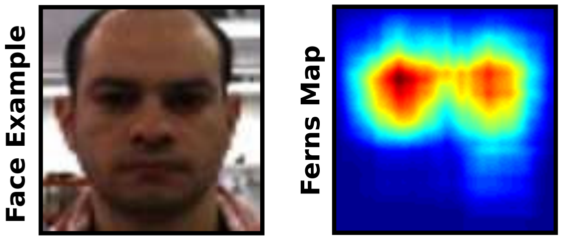

Fig. 12 Spatial distribution of ferns selected by boosting to assemble the BRFs classifier.

the ORFs classifier is capable of detecting most faces despite the out-the-plane rotations. This tracking step is automatic and it is only necessary to annotating the first face in the image sequence using a bounding box[6]. This is achieved thanks to ORFs that perform tracking by detection. That is, the classifier is tested at every frame in order to discover human faces. Subsequently, the detection predictions obtained in this frame are used to update and refine the online classifier. Note also that the classifier runs in real time (about 10 FPS) using a C++ code[7].

With respect to the parameters used to compute the ORFs classifier, we use $R = 500$ ferns, $M = 10$ binary features per fern, and a squared subwindow $s_{\mathbf{u}}$ of size $S = 10$. For this experiment, all images are in RGB color space and normalized to a size of 30×30 pixels. We set to $P = 10$ the size of the pool of fern parameters.

Once the tracking step has been conducted on the training sequence, the result is a set of positive (faces) and negative (background) image samples that are used to compute the BRFs classifier. Some positive samples are shown in Fig. 11. Looking at these images we see the large intra-class variability of faces. The visual appearance of faces changes depending on the viewpoint.

Using the training samples, the BRFs classifier is computed by selecting the most relevant ferns towards classification. Precisely, the boosting algorithm selects the feature parameters $\boldsymbol{\theta}_r$ and the fern location $\mathbf{u}_r$ that best discriminates the faces from background samples. The spatial layout of the selected ferns is visualized in Fig. 12, where reddish areas indicate high density of ferns. Clearly the classifier focuses on regions around the eyes since they have shown to be discriminative features for face detection [45].

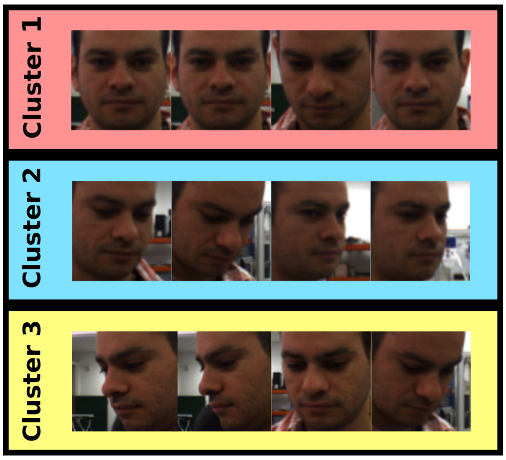

Fig. 13 Sample images showing three different face appearance clusters. Observe that samples belonging to the same cluster share visual similarities.

With regard to the parameters of BRFs, we keep the same values that ORFs, with the exception of using $M = 7$ binary features and $R = 300$ ferns. To compute the classifier, we use about 150 positive and negative image samples. Fig. 11 shows some positive samples.

The result of the clustering step via pLSA is provided in Fig. 13. This figure shows some example images corresponding to $K = 3$ different intra-class appearances modes (clusters) found by the proposed method during the training phase. We see that these samples share similar visual features and that RCFs are able to cluster these samples using the output of a tree-structured classifier.

In Fig. 14 are shown some detection results of RCFs on the test image sequence. Note that RCFs are capable of detecting most faces at the same time that they can estimate the face pose. This is indicated in the images through colored boxes. This experiment reveals that the proposed method using a single classifier can be used for pose estimation using the co-occurrence of tree-structured visual words. However, we observe in the figure a few mistakes (e.g, false positives) due mainly to detecting other faces outside the training set. Note that the goal here is to detect a particular face in which the classifier was trained (training image sequence) and not in the general problem of face detection. RCFs run in this experiment at 2 FPS using a Matlab code.

[6] Albeit it is possible to use human assistance during the learning to improve the visual skills of the classifier [40]

[7] The code is available at http://www.iri.upc.edu/people/mvillami/code.html

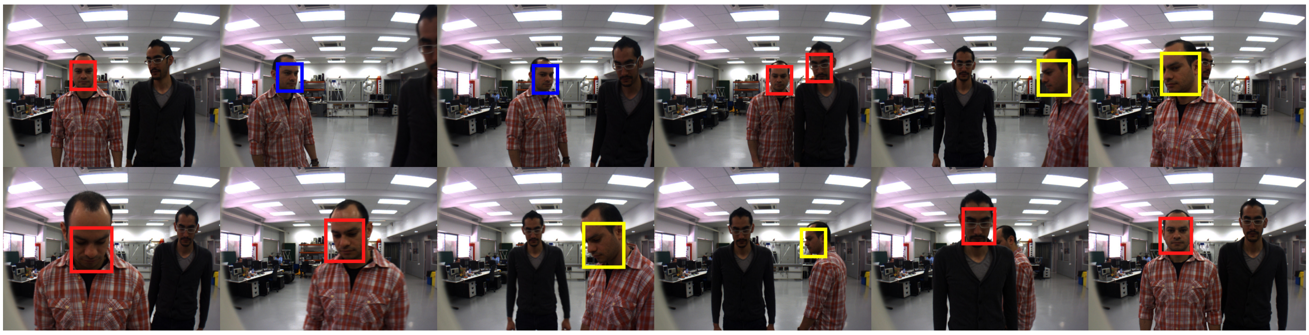

Fig. 14 Face detection results provided by the proposed approach. The output of the RCFs is indicated by boxes, whereas the face pose is shown through different colors.

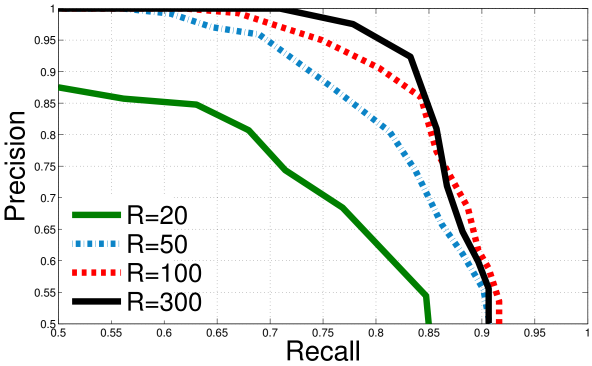

Fig. 15 Performance of RCFs in function of the amount of ferns used to compute the BRFs classifier.

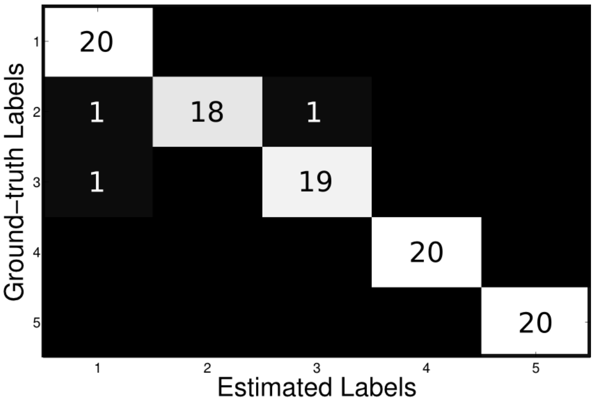

Fig. 16 Confusion matrix for automatic face clustering.

Fig. 15 plots the performance of RCFs, based on the BRFs classifier, in accordance to the number of ferns R. We see that the classification rates, using the recall-precision plot, increase as the quantity of ferns is larger. Nevertheless, this improvement is small after 300 ferns. This saturation behavior is commonly observed in BRFs for other recognition problems [39].

Automatic Face Clustering. To evaluate more deeply the clustering performance of RCFs, we propose an experiment for automatic face clustering in which a set of images with human faces is grouped according to people identity. This experiment counts with $N = 100$ images from the Caltech face dataset [12]. These images contain 5 different people (20 images per person) in different illumination conditions.

Multimodal Performance Rates				
	Binary Features			
	$M = 3$	$M = 5$	$M = 7$	$M = 9$
# Words (W)	2.400	9.600	38.400	153.600
Classification and Clustering Rates				
EER	1.0	1.0	1.0	1.0
Entropy	0.49	0.37	0.29	0.20
Training Times				
BRFs [sec.]	15.08	15.75	17.94	22.63
pLSA [min.]	0.26	1.50	7.40	30.27

Table 3 Clustering and classification performance of RCFs in terms of the numbers of features M and visual words W.

For the computation of RCFs, instead of using the tracking step, we use the ground truth of the face dataset to extract the training samples. Also, we use $R = 300$ ferns and $M = 7$ features. This results in $W = 38.400$ visual words and in a corpus of size 100×38.400, being most values empty (zeros)[8]. See Fig. 7 for an example.

The confusion matrix using $K = 5$ intra-class modes is observed in Fig. 16. Brighter diagonal values show that the method performs clustering remarkably[9]. Most of estimated labels (people identity) match with the ground-truth labels provided by the database. Some example for this problem are seen in Fig. 17. Each color refers to an intra-class mode or person identity. We see that the proposed approach computes a classifier for face detection and arranges the image set in function of people identity.

The average clustering performance of RCFs in terms of the number of visual words is observed in Table **??** This is done computing the BRFs classifier for different

[8] For this problem, only 300 visual words are activated out of 38.400 words, each one corresponding to a fern output.

[9] Since the pLSA clustering is automatic, the confusion matrix is not necessarily diagonal. However, here the labels provided by pLSA have been sorted for display purposes.

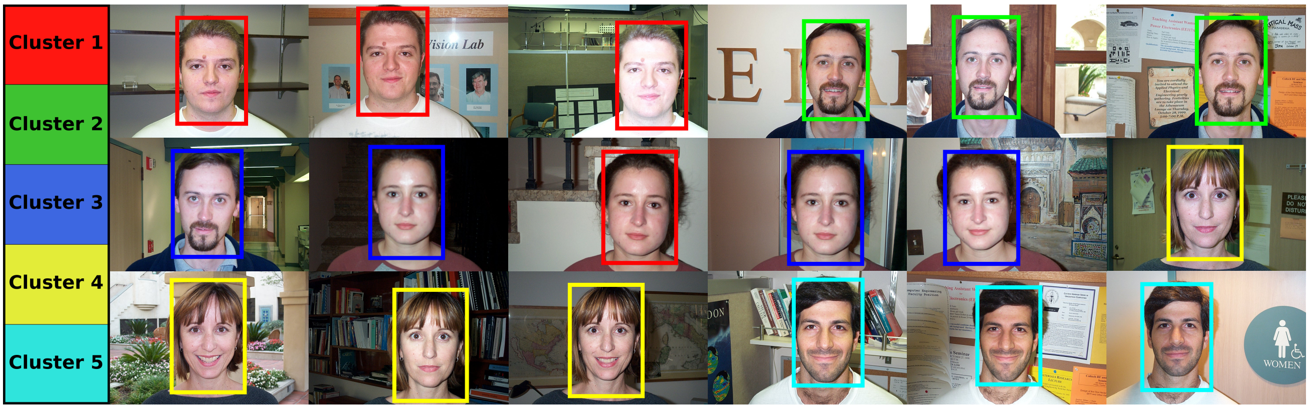

Fig. 17 Clustering results of RCFs on the Caltech face dataset [12]. The proposed method discovers latent intra-class appearance clusters which correspond to people identity.

Clustering Performance Comparison			
	K-means	BRFs+K-means	BRFs+pLSA
Entropy	0.79	0.35	0.29

Table 4 Entropy values for different clustering strategies: K-means, BRFs+K-means and RCFs (BRFs+pLSA).

values of binary features M, which results in different numbers of ferns leaves. The number of ferns is fixed to $R = 300$. Similarly to the synthetic experiments, the performance of RCFs is measured using the entropy-based error on the confusion matrix since this matrix is not frequently diagonal. Note that the clustering error is decreased along to the fern depth (number of binary features) and visual words. Using larger amounts of visual words, RCFs yield low clustering errors. Fig. 16 shows, for instance, the clustering results for $M = 7$ features, where just three samples were wrongly classified.

Face detection rates are also reported in Table 3. RCFs attain perfect classification in all cases (1.0 EER), showing that the BRFs classifier is able to learn and detect multiple and different faces. The table also indicates the training times of RCFs. We see that the computation of the BRFs classifier is done relatively fast, in decaseconds, whereas the clustering via pLSA is done in the order of minutes. This is especially critical for larger sizes of visual words, which are in turn need to obtain good clustering results. However, once RFCs are computed during the learning step, the proposed multimodal detector runs in about two frames per second using Matlab and MEX files[10].

Similar to the synthetic experiments, we compare the clustering performance of RCFs against K-means and BRFs+K-means (see Table 4). The former performs clustering using the pixels intensities in images while the latter is carried out using the visual words provided by BRFs. Observe that RCFs (BRFs+pLSA) achieve the lowest entropy value, showing that performing clustering on visual words yields better results than on pixel data.

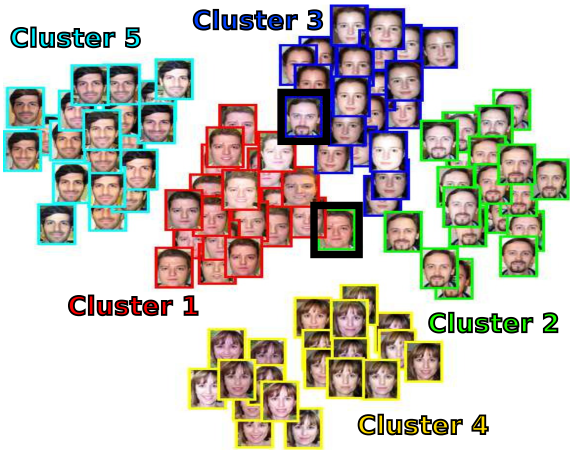

Fig. 18 RCFs clustering results using tSNE [9] visualization.

Fig. 18 shows the clustering results of RCFs using the algorithm proposed in [9] for visualization (tSNE). This algorithm performs dimensionality reduction via PCA. Here, the visual words contained in the $N \times W$ corpus are reduced to 2D points, each representing a training image sample. The colors in the figure indicate the intra-class cluster labels provided by RCFs. Notice that most of the estimations are correct and agree with the dataset labels (people identity). Only two samples are misclassified, indicated in the figure by black boxes.

Multi-view Object Detection. The proposed method is also used for object recognition. In this case, for detecting a mug and a Rubik cube seen from multiple views. To this end, we use the BoBot dataset [21] that contains various image sequences with different objects. Each sequence has more than 600 images. For this ex-

[10] The code for RCFs is available at http://www.iri.upc.edu/people/mvillami/code.html.

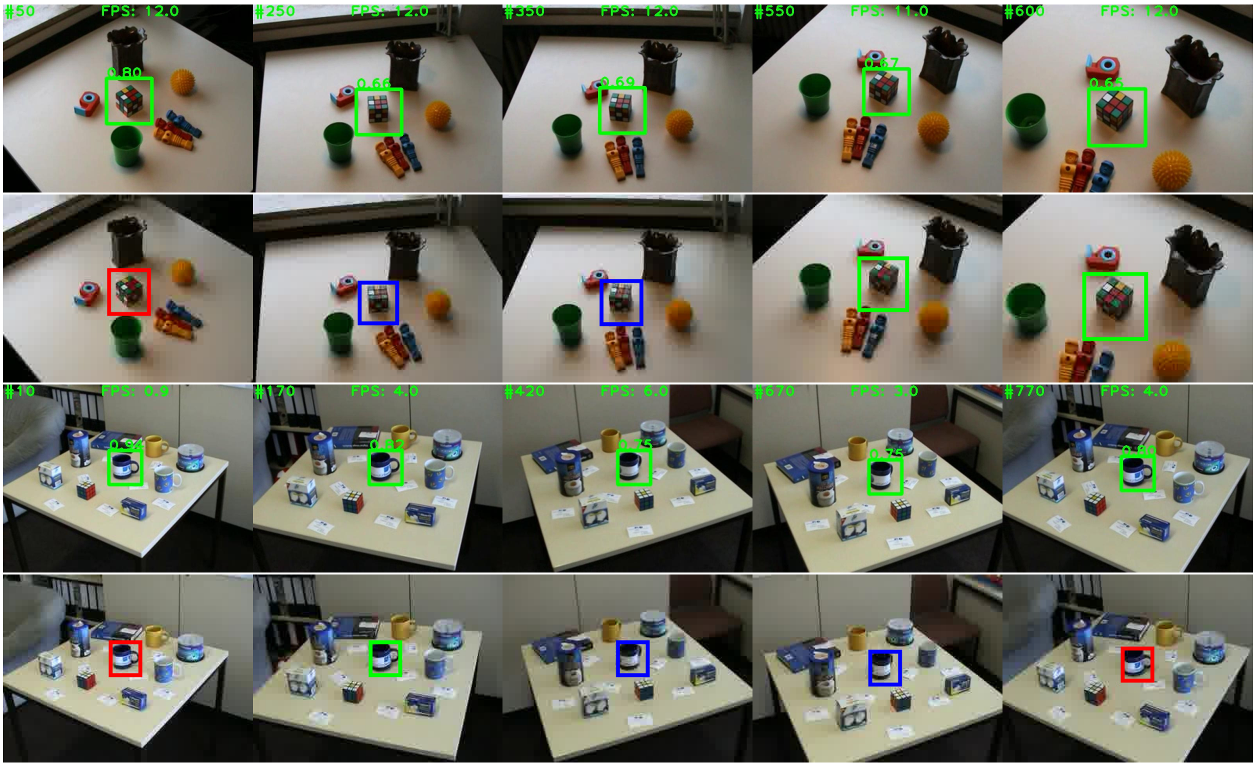

Fig. 19 Multimodal object recognition results provided by RCFs for two image sequences of the BoBot dataset [21]. First and third rows depict the response of the tracking step for a Rubik cube and a mug, respectively. Second and fourth rows show the output of RCFs using $K = 3$ latent appearance clusters.

periment, we use $K = 3$ appearance clusters and the same parameters values than previous experiments.

Fig. 19 shows the response of RCFs for the addressed objects, represented by bounding boxes in images. The first and third rows correspond to the output of the tracking step on the Rubik cube and mug sequences, respectively. Once again, the ORFs classifier can detect the objects pretty well using weakly human annotation. However, this method is not able to distinguish different object appearance modes. By contrast, the second step of the proposed method, the RCFs, allows to detect the objects and discretize the overall object appearance into different subsets with high feature correlation. The second and fourth rows, in the same figure, show some example images with the output of RCFs. Observe that the method discovers $K = 3$ latent appearance modes.

5 Conclusions and Future Work

In this paper, we have introduced Random Clustering Ferns (RCFs) for the learning and detection of objects with multiple intra-class appearance modes. The proposed method performs simultaneously object detection and clustering using Boosted Random Ferns (BRFs) and probabilistic Latent Semantic Analysis (pLSA). This is particularly convenient for detecting objects seen from different viewpoints since RCFs allow to localize the object in images and recognize a specific object appearance. Unlike other works devoted to this task, using various pose-specific detectors, RCFs use a single and discriminative classifier based on random ferns. This proposed approach also contrasts with more complex methods in the state of the art which require of thousands of training samples and of hours, or even days, for computing reliable classifiers (e.g, deep neural networks). Conversely, our method can be computed in a few minutes. The efficiency of RCFs lies in using fast features (random ferns) to compute an object classifier via Boosting, and visual words to perform clustering using pLSA. The method has been validated in diverse synthetic and real experiments where remarkable results have been obtained. In addition, the proposed RCFs have been combined with a tracking step in order to reduce the human supervision and extract automatically training samples.

In spite of the good results achieved for multimodal object recognition, we consider that the presented work can be improved, especially in terms of efficiency and accuracy. For example, computing more semantic visual features in order to reduce the high dimensionality of the visual words. We have seen during the experiments that using large visual words reduces the clustering error, but this is at the expense of an increased computational cost because the clustering via pLSA is time-consuming. Other future line of research is to use incremental pLSA algorithms to perform online learning and clustering. This would of great interest for interactive and robotics applications.

6 Acknowledgments

This work has been partially funded by the Spanish Ministry of Economy and Competitiveness under projects ERA-Net Chistera project ViSen PCIN-2013-047, RobInstruct TIN2014-58178-R, ROBOT-INT-COOP DPI2013-42458-P, and by the EU project AEROARMS H2020-ICT-2014-1-644271

References

1. K. Ali and K. Saenko. Confidence-rated multiple instance boosting for object detection. In *CVPR*, 2014.
2. H. Blockeel, L. De Raedt, and J. Ramon. Top-down induction of clustering trees. In *ICML*, pages 55–63, 1998.
3. A. Bosch, A. Zisserman, and X. Muñoz. Scene classification via plsa. In *ECCV*. 2006.
4. A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *ICCV*, pages 1–8, 2007.
5. Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
6. Antonio Criminisi, Jamie Shotton, and Ender Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7(2–3):81–227, 2012.
7. G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Proc. ECCV Workshop Statistical Learning in Computer Vision*, pages 59–74, 2004.
8. N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893, 2005.
9. L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(85):2579–2605, 2008.
10. D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014.
11. P. F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 32(9):1627–1645, 2010.
12. R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.
13. J. Gall, A. Yao, N. Razavi, L. Van Gool, and V. Lempitsky. Hough forests for object detection, tracking, and action recognition. *PAMI*, 33(11):2188–2202, 2011.
14. A. Garrell, M. Villamizar, F. Moreno-Noguer, and A. Sanfeliu. Proactive behavior of an autonomous mobile robot for human-assisted learning. In *RO-MAN*, 2013.
15. R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.
16. D. Hall and P. Perona. From categories to individuals in real time–a unified boosting approach. In *CVPR*, 2014.
17. T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2):177–196, 2001.
18. F. Jurie and B. Triggs. Creating efficient codebooks for visual recognition. In *ICCV*, pages 604–610, 2005.
19. Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking–learning–detection. *PAMI*, 34(7):1409–1422, 2012.
20. T.K. Kim and R. Cipolla. Mcboost: Multiple classifier boosting for perceptual co-clustering of images and visual features. In *NIPS*, pages 841–848, 2009.
21. Dominik A. Klein, Dirk Schulz, Simone Frintrop, and Armin B. Cremers. Adaptive real–time video–tracking for arbitrary objects. In *IROS*, 2010.
22. E. Krupka, A. Vinnikov, B. Klein, A.B. Hillel, D. Freedman, and S. Stachniak. Discriminative ferns ensemble for hand pose recognition. In *CVPR*, pages 3670–3677, 2014.
23. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
24. B. Liu, Y. Xia, and P.S. Yu. Clustering through decision tree construction. In *Proc. Ninth ACM Int. Conf. Information and Knowledge Management*, pages 20–29, 2000.
25. D.G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.
26. T. Malisiewicz, A. Gupta, and A.A. Efros. Ensemble of exemplar-svms for object detection and beyond. In *ICCV*, pages 89–96, 2011.
27. R. Marée, P. Geurts, J. Piater, and L. Wehenkel. Random subwindows for robust image classification. In *CVPR*, pages 34–40, 2005.
28. F. Moosmann, E. Nowak, and F. Jurie. Randomized clustering forests for image classification. *PAMI*, 30(9):1632–1646, 2008.
29. K. P. Murphy. *Machine learning: a probabilistic perspective*. 2012.
30. D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *CVPR*, pages 2161–2168, 2006.
31. M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *PAMI*, 32(3):448–461, 2010.
32. M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *CVPR*, pages 778–785, 2009.
33. F. Perbet, B. Stenger, and A. Maki. Random forest clustering and application to video segmentation. In *BMVC*, pages 1–10, 2009.
34. R.E. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, 1999.
35. P. Sharma and R. Nevatia. Multi class boosted random ferns for adapting a generic object detector to a specific video. In *WACV*, pages 745–752, 2014.
36. J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*, pages 1–8, 2008.
37. J. Sivic, B. Russell, A. Efros, A. Zisserman, and W.T. Freeman. Discovering objects and their location in images. In *ICCV*, 2005.
38. A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing visual features for multiclass and multiview object detection. *PAMI*, 29(5):854–869, 2007.

39. M. Villamizar, J. Andrade-Cetto, A. Sanfeliu, and F. Moreno-Noguer. Bootstrapping boosted random ferns for discriminative and efficient object classification. *Pattern Recognition*, 45(9):3141–3153, 2012.
40. M. Villamizar, A. Garrell, A. Sanfeliu, and F. Moreno-Noguer. Online human-assisted learning using random ferns. In *ICPR*, pages 2821–2824, 2012.
41. M. Villamizar, A. Garrell, A. Sanfeliu, and F. Moreno-Noguer. Modeling robot's world with minimal effort. In *ICRA*, 2015.
42. M. Villamizar, A. Garrell, A. Sanfeliu, and F. Moreno-Noguer. Multimodal object recognition using random clustering trees. In *IBPRIA*, 2015.
43. M. Villamizar, H. Grabner, J. Andrade-Cetto, A. Sanfeliu, L. Van Gool, and F. Moreno-Noguer. Efficient 3d object detection using multiple pose-specific classifiers. In *BMVC*, 2011.
44. M. Villamizar, F. Moreno-Noguer, J. Andrade-Cetto, and A. Sanfeliu. Efficient rotation invariant object detection using boosted random ferns. In *CVPR*, pages 1038–1045, 2010.
45. P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, pages l–511, 2001.
46. B. Wu and R. Nevatia. Cluster boosted tree classifier for multi-view, multi-pose object detection. In *ICCV*, pages 1–8, 2007.
47. J. Yan, Z. Lei, L. Wen, and S. Z. Li. The fastest deformable part model for object detection. In *CVPR*, pages 2497–2504, 2014.