Fast Online Learning and Detection of Natural Landmarks for Autonomous Aerial Robots

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

***Abstract*— We present a method for efficiently detecting natural landmarks that can handle scenes with highly repetitive patterns and targets progressively changing its appearance. At the core of our approach lies a Random Ferns classifier, that models the posterior probabilities of different views of the target using multiple and independent Ferns, each containing features at particular positions of the target. A Shannon entropy measure is used to pick the most informative locations of these features. This minimizes the number of Ferns while maximizing its discriminative power, allowing thus, for robust detections at low computational costs. In addition, after offline initialization, the new incoming detections are used to update the posterior probabilities on the fly, and adapt to changing appearances that can occur due to the presence of shadows or occluding objects. All these virtues, make the proposed detector appropriate for UAV navigation. Besides the synthetic experiments that will demonstrate the theoretical benefits of our formulation, we will show applications for detecting landing areas in regions with highly repetitive patterns, and specific objects under the presence of cast shadows or sudden camera motions.**

I. Introduction

Statistical methods for object detection and categorization have achieved a degree of maturity that makes them robust to several challenging conditions such as target scaling, 2D and 3D rotations, nonlinear deformations and lighting changes [6], [8], [10], [14], [21], [19], [26], [28], [30]. Yet, most of these approaches involve complex computations, preventing its use in systems requiring online video processing. This limitation is specially critical when designing perception algorithms for Unmanned Aerial Vehicle (UAVs), where both real time and robustness are mandatory characteristics.

Most current UAV perception algorithms use external markers placed along the environment or on the object of interest, which can be easily detected with RGB or infrared cameras. Tasks such target detection [12], [15], [18], navigation [9], [31] and landing [7], [24] can be easily simplified with the use of these markers. There are, however, situations where the deployment of markers is not practical or possible, especially when the vehicle operates in dynamically changing and outdoor scenarios.

In this paper, we propose an efficient algorithm for detecting the pose of natural landmarks on input video sequences without the need of using external markers. This is especially remarkable, as we consider scenes like the one shown in Fig. 1, where the target is a chunk of grass in which identifying distinctive interest points is not feasible, preventing thus the use of keypoint recognition methods [13],

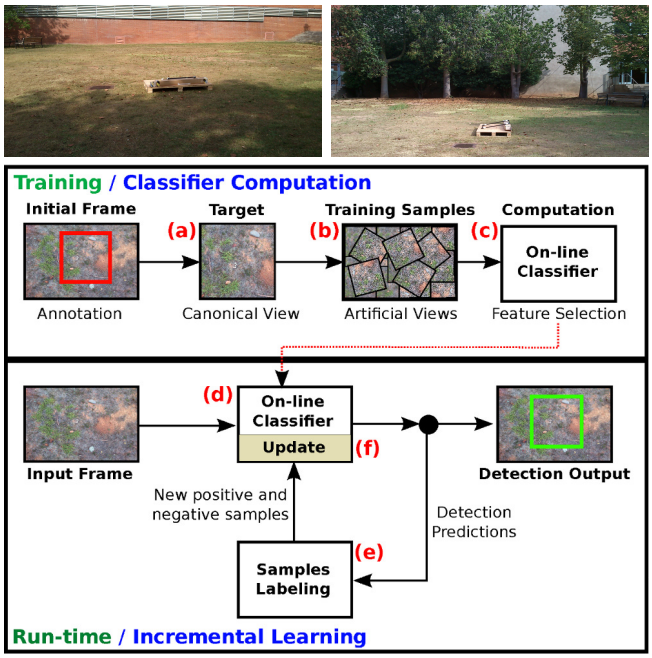

Fig. 1. **Detecting natural landmarks. Top:** Kind of outdoor scenario we consider. Some of the challenges our detector needs to address are light changes, shadows and repetitive textures. **Bottom:** Schematic of the approach we propose. It consists of two stages, an offline learning stage where a general model of the object's appearance is learned, and an online stage, where the object's model is continuously updated using input images.

[20]. In addition, our approach continuously updates the target model upon the arrival of new data, being able to adapt to dynamic situations where the its appearance may change. This is in contrast to the previous approaches, which learn object appearances from large training datasets, but once these models are learned, they are kept unchanged during the whole testing process.

As shown in Fig. 1, our approach consists of two main stages. Initially, a canonical sample of the target is provided by the user as a bounding box in the first frame of the sequence (Fig. 1(a)). Through synthetic warps based on shifts and planar rotations, new samples of the target are generated, each associated to a specific viewpoint (Fig. 1(b)). All these samples are used for training a classifier that models the posterior of each view (Fig. 1(c)). At the core of the classifier there are Ferns features [22] that given an input training sample, model its appearance by combining random sets of binary intensity differences. Yet, in contrast to [22] that

Fig. 2. **Failure of keypoint-based methods**. **Top:** Matching of SIFT features (red lines) between the visual target (small top-left image) and a sequence of image examples that includes lighting and viewpoint changes. **Bottom:** Output of the proposed detection approach (red circles) for the same target and sequence. Black circles indicate the ground truth and the rectangle shows the detection rates: true positives (TP), false positives (FP) and false negatives (FN).

randomly picks the location of each Fern feature within the image, we propose a strategy that uses an entropy reduction criterion for this purpose. This will let us to both minimize the number of Ferns to represent the target (making the algorithm more efficient), and maximize the discriminative power of the classifier. All this initial training is performed offline, in matter of minutes.

In the second stage (Fig. 1(d)) the classifier is evaluated at each input frame, and its detections are used to update the posterior probabilities, which still contain the information of the original target appearance that avoids drifting to false detections (Fig. 1(e)). This allows a non-supervised adaption of the classifier to progressive target changes.

All these ingredients (markerless, efficient and adaptable) make our approach appropriate for Autonomous Aerial Robots applications. After showing the theoretical benefits of the method using synthetic data, we will describe several real experiments in which our classifier will be used to detect the position and orientation of natural landmarks in outdoor environments where a UAV is expected to operate.

II. Related Work

Marker-based Approaches

The standard approach for estimating the location of an aerial mobile robot or specific objects of its environment relies on visual markers introduced in the scene. These markers can be either perceived by RGB cameras [7], [24] or infrared sensors [9], [17]. Especially the latter, provide very accurate pose estimation results and at a high frame rate, allowing to design accurate control laws and perform complex tasks such as that of cooperative grasping and manipulation with multiple aerial vehicles [16].

There are situations, though, in which deploying these markers is not possible or convenient. For instance, the response of an infra-red sensor is easily washed out by sunlight in outdoors scenarios. In other circumstances, the target to be tracked or detected is chosen on the fly, and is not possible to place markers on it. In this kind of situations, passive and non-invasive techniques such as those based in vision alone, come into play.

Markerless Vision-based Approaches

Remind that our goal is to locate the position and the in-plane orientation of a given target in an input image or video sequence. One obvious solution for this is based on algorithms that first extract points of interest from the input and target images, represent them with a potentially complex descriptor [1], [14], and match them using a robust algorithm for outlier rejection [13], [20], [25]. Yet, in the situations we consider with natural landmarks, repetitive patterns (like grass) and lighting artifacts, extracting reliable and salient points of interest is not an easy task. Observe, for instance, the example in Fig. 2, where SIFT descriptors are used to match points of interest of the natural target seen from different viewpoints. Note that the percentage of matches is very large for the first images but it decreases significantly for the next ones because of lighting and viewpoint changes. Therefore, these point based algorithms have been mostly used in indoor applications with controlled light conditions. Indeed, we can find some recent works that under these constraints have been shown effective for UAV navigation [4], visual tracking [18] and target detection [27].

When single feature points are not reliable cues, one can model the appearance of the whole target object. This can be expressed as a classification problem, where each target orientation corresponds to a different class. There are potentially many algorithms which could be used for this task, and which have been shown to give excellent results in object detection applications [6], [8], [23], [26]. Yet, most of them have a high computational cost and require rigorous training procedures for being effectively implemented in aerial robots.

The approach we present here falls within this group of methods, but we alleviate the computational cost of our classifier using two strategies. First, we split the learning process in two stages, one offline and the other online. This helps to reduce the amount of information included in the model and thus, reduces its complexity. And second, we propose an optimization strategy based on entropy minimization, in which the number of features is minimized while retaining their discriminative power. The essence of this strategy is

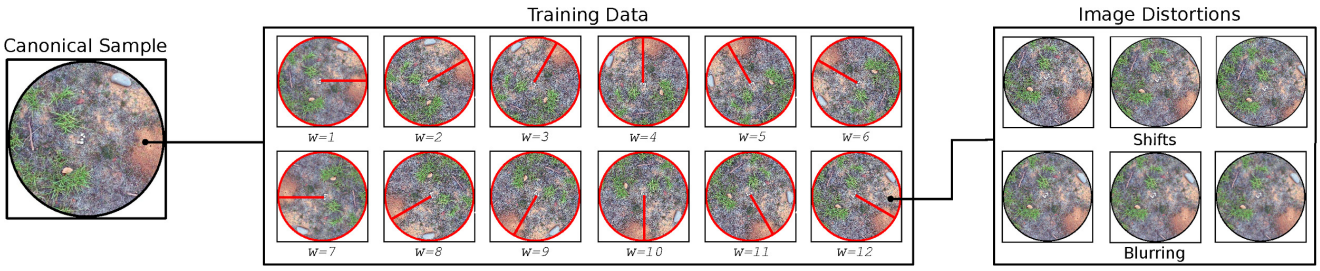

Fig. 3. **Synthetic training data**. The canonical sample (left) is synthetically warped to generate new training samples (middle). These samples are computed at different orientations and at different shift and blurring levels (right). The red circle and arrow indicate the target pose for each sample.

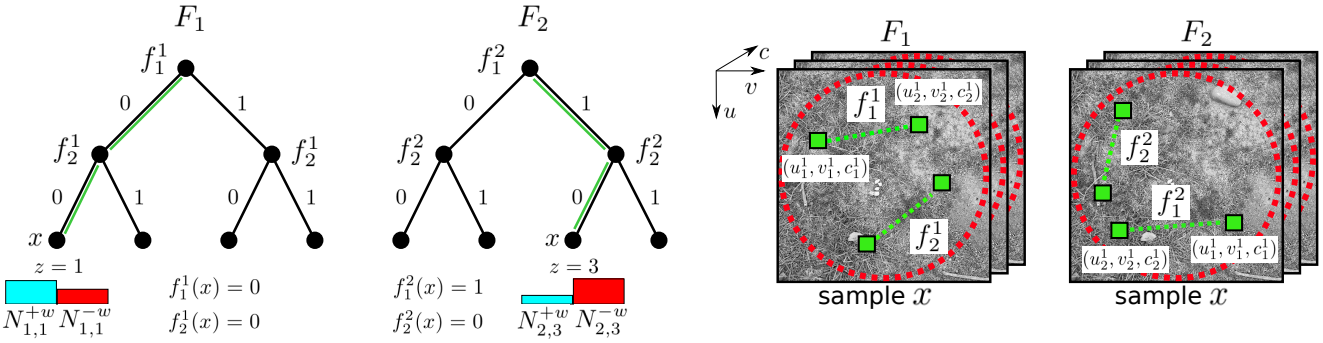

Fig. 4. **Fern-based classifier**. Computation of the classifier using $J = 2$ Ferns with $M = 2$ binary features. **Left:** Schematic representation of the Ferns structure using binary trees. At the bottom of the tree we plot the distributions which are updated for a training sample x. For instance, assuming the training sample x belongs to the positive class and that $F_1(x) = (00)_2 + 1 = 1$, the bin of the positive class in $z = 1$ would be increased in one unit. The same sample, would also increase in one unit the bin corresponding to $z = 3$ of F_2, as $F_2(x) = (10)_2 + 1 = 3$. **Right:** Example of how the Ferns are tested on an image sample x. Features are signed comparisons between image pixels. (u, v) denote the spatial coordinate, and c the color channel coordinate.

similar to what is done in recent works with two-class classification problems [11], [29], [30]. Yet, these works are not applicable to our multiview-classification problem.

III. Approach

We next describe the main steps for building the classifier: generation of an initial set of synthetic samples, offline construction of the classifier, a new criteria for selecting the features, and finally, the online adaption of the algorithm.

A. Generating Synthetic Samples for Offline Training

We initially assume that we only have one single sample of the target we seek to detect. This canonical sample is provided by the user as a bounding box in the first frame of the video sequence. In order to obtain a more complete description of the target we synthetically generate new views of the canonical shape.

As it is typically done in aerial imagery, the depth of the target is assumed negligible compared to its distance w.r.t. the camera. We therefore consider the canonical target as being planar, and approximate the multiple views it can take as in-plane rotations. Note, however, that our approach is equally valid for non-planar objects. In that case, sample training images could be either generated with more sophisticated rendering tools or by simply acquiring real images of the target from each of the viewpoints.

For the purposes of this paper, and as shown in the example of Fig. 3, the canonical shape is rotated at $W = 12$ principal pose orientations, that will establish the classes of our classification problem. In addition, for each pose $w \in \{1, 2, .., W\}$ we further include 6 additional samples with random levels of shifting and blurring. This helps to model small deviations from the planar assumption, as well as the blurring produced by sudden motions of the camera. A final subset with a similar number of background samples (random patches chosen from background) per pose is also considered. We denote this whole initial training dataset as $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ where $x_i \in \mathcal{X}$ corresponds to a sample in the image space $\mathcal{X}$, N is the number of samples, and $y_i = \{+w_i, -w_i\}$ is the class label, indicating if the sample belongs to the pose w or background classes, respectively.

B. Building the Classifier

In order to perform online learning and object detection, we use Random Ferns (RFs) [22]. This classifier can be understood as an extreme and very fast implementation of a random forest [5] which combines multiple random decision trees. Furthermore, subsequent works have shown the RFs to be robust to over-fitting and that they can be progressively computed upon the arrival of new data [11], [29]. The most distinctive characteristic of RFs compared to the classical random forests is that the same test parameters are used in all nodes of the tree level [5], [22]. We show this in Fig. 4-left, where we can see two Ferns F, each one with two decision tests or binary features f.

More formally, the online classifier is built as an average of J Ferns in which each Fern F_j consists of a set of M binary features, $F_j = \{f_1^j, f_2^j, \ldots, f_M^j\}$, representing binary comparisons between pairs of pixel intensities. This can be written as

$$f(x) = \mathbb{I}(x(u_1, v_1, c_1) > x(u_2, v_2, c_2))$$

where x is the image sample, $x(u, v, c)$ indicates the image value at pixel coordinates (u, v) with color channel c, and $\mathbb{I}(e)$ is the indicator function:

$$\mathbb{I}(e) = \begin{cases} 1 & \text{if } e \text{ is true} \\ 0 & \text{if } e \text{ is false} \end{cases}$$

As we will describe in the following subsection, and in contrast to the original Ferns formulation [22], the location of these pairs of pixels is computed during the training stage according to a criterion of entropy minimization. Fig. 4-right shows a simple example of how two different Ferns with two features are evaluated in an image sample x. The combination of these binary features determines the Fern output, $F(x) = z$, where $z = (f_1, \ldots, f_M)_2 + 1$, is the co-occurrence of the features and corresponds to the Fern leaf where the sample x falls (See Fig. 4-left).

So far, we have discussed how a single Fern is evaluated on an image sample. Let us now explain how the classifier is built, from the response of J Ferns $\mathbf{F} = \{F_1, \ldots, F_J\}$. The response of the classifier, for an input sample image x can be written as

$$H(x) = \begin{cases} +1 & \text{if conf}(x \in \hat{w}) > \beta \\ -1 & \text{otherwise,} \end{cases}$$

where $\hat{w}$ is the estimated pose of the sample, $\text{conf}(x \in \hat{w})$ is the confidence of the classifier on predicting that x belongs to the class $\hat{w}$, and β is a confidence threshold whose default value is 0.5. Thus, if the output of the classifier for a sample x is $H(x) = +1$, the sample is assigned to the target (positive) class $\hat{w}$. Otherwise, it is assigned to the background (negative) class. The confidence of the classifier is defined according to the following posterior:

$$\text{conf}(x \in \hat{w}) = p(y = +\hat{w}|\mathbf{F}(x), \boldsymbol{\theta}), \tag{1}$$

where $\boldsymbol{\theta}$ are parameters of the classifier we will define below and $y = \{+w, -w\}$ denotes the class label.

The estimated pose $\hat{w}$ is computed by evaluating the confidence function over all possible poses, and picking the one with maximum response, i.e.:

$$\hat{w} = \arg\max_{w} p(y = +w|\mathbf{F}(x), \boldsymbol{\theta}), \quad w = 1, \ldots, W$$

As said before, this posterior probability is computed by combining the posterior of the J Ferns:

$$p(y = +w|\mathbf{F}(x), \boldsymbol{\theta}) = \frac{1}{J} \sum_{j=1}^{J} p(y = +w|F_j(x) = z, \theta_{j,z,w}),$$

where z is the Fern output, and $\theta_{j,z,w}$ is the probability that a sample in the Fern j with output z belongs to the positive class with pose w. Since the posterior probabilities follow a Bernoulli distribution

$$p\left(y|F_j(x) = z, \theta_{j,z,w}\right) \sim \text{Ber}(y|\theta_{j,z,w}),$$

with we can write that

$$p\left(y = +w|F_j(x) = z, \theta_{j,z,w}\right) = \theta_{j,z,w}$$

The parameters of these distributions are computed during the training stage through a Maximum Likelihood Estimate (MLE) over the labeled set of synthetic samples $\mathcal{D}$ we have previously generated. That is,

$$\theta_{j,z,w} = \frac{N_{j,z}^{+w}}{N_{j,z}^{+w} + N_{j,z}^{-w}} \tag{2}$$

where $N_{j,z}^{+w}$ is the number of positive samples that fall into the leaf z of the Fern j. Similarly, $N_{j,z}^{-w}$ corresponds to the number of negative samples for the Fern j with output z. The reader is referred to Fig. 4-left for an illustrative example.

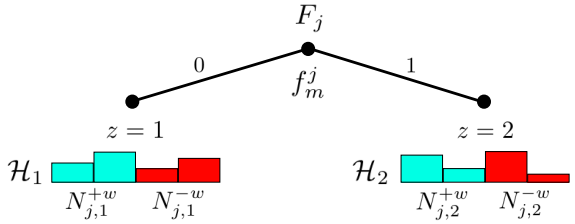

Fig. 5. **Feature selection process**. Example of the kind of distributions we consider in the Fern leaves, for a case where we have two classes or poses w. Each distribution $\mathcal{H}$ encodes the amount of positive and negative samples, for each of the classes.

C. Feature Selection

In all previous works that use RFs classifiers, the Ferns features, i.e, the pairs of pixels whose intensities are compared, are chosen at random [11], [22], [29]. In this paper we claim, and we will demonstrate it in the results, that a more principled approach for selecting those features can lead to increased levels of efficiency and robustness.

For this purpose we propose a methodology to choose the binary features that reduce the classification error over the training data $\mathcal{D}$. As an approach to this, we will seek for the features that minimize the Shannon Entropy $\mathcal{E}$, which gives a measure about the impurity of the tree (i.e, how peaked are the posterior distributions at each Fern), and about the uncertainty associated with the data [3], [26].

More specifically, each Fern F_j is independently computed from the rest of Ferns, and using a different and small random subset $\mathcal{S} \subseteq \mathcal{D}$ of the training data. Partitioning the training data will avoid potential overfitting errors during testing [3], [5]. Let us now assume we have a large and random pool of binary features, and we want to pick the best of them for a Fern F_j. At each node level m, we will choose the binary feature f_m that minimizes the entropy of the Fern $\mathcal{E}(F_j)$, computed as

$$\mathcal{E}(F_j) = \sum_{z=1}^{2^m} -\frac{N_{j,z}}{|S|} \mathcal{E}(\mathcal{H}_z), \quad \mathcal{E}(\mathcal{H}_z) = -\mathcal{H}_z \log \mathcal{H}_z,$$

where $N_{j,z}$ is the number of samples falling into the leaf z and $|S|$ is the size of the samples subset S. The variable $\mathcal{H}_z$ is the distribution of samples across poses w in the leaf z, and is represented trough a normalized histogram (See Fig. 5).

Once the feature f_m that minimizes $\mathcal{E}(F_j)$ is chosen, it is added to the set of features of F_j. This is repeated until a maximum number of features M (corresponding the the depth of the Fern) is reached. The pseudocode of the whole procedure for building the classifier is presented in Alg. 1.

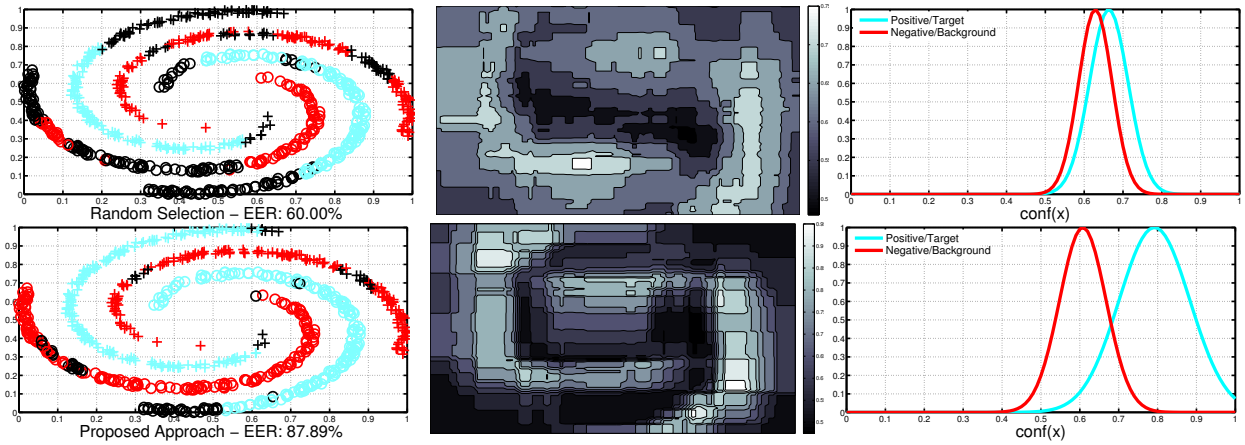

Fig. 6. **Random Ferns (Top) vs Entropy-based Ferns (Bottom)**. **Left:** The proposed approach is compared to standard RFs in a two-class synthetic problem. Cyan and red symbols correspond the the two main classes, positive and negative respectively, and the + and 0 are two additional values that each element of every class can take. Black symbols indicate misclassified samples. **Middle:** Maps showing the response of the classifiers on the dense sample space. **Right:** Distributions of positive and negative classes according to the confidence conf(x) of the classifiers, computed from Eq. 1.

In order to highlight the advantages of the Entropy-based approach for selecting features, compared to the standard random approach, we have performed the toy experiment summarized in Fig. 6. The problem consists in building a classifier for a two-class (red and cyan classes) problem, where each class element may take two possible values ("+" and "0"). The binary features in this example are axis-aligned split functions (2D decision stumps) with a random threshold. That is, given a sample x with coordinates $(u, v) \in [0, 1] \times [0, 1]$ we compute binary features as

$$f(x) = (p > \lambda) \ ,$$

where $p = \{u, v\}$ corresponds to one of the axis, and λ a random threshold in the interval $[0, 1]$. We train the classifiers with 20 Ferns and 7 such features per Fern.

The left-most column of Fig. 6 shows the response of both classifiers to new testing data, where black "0" or "+" symbols are misclassified samples. As expected, the classification results are consistently better when using the Entropy for selecting the features. This is also illustrated in the dense classification maps shown in Fig. 6-middle, where the response of the our classifier, clearly yields a more precise information about the spatial layout of each of the classes.

Another advantage of the proposed classifier is that it provides a greater separation between positive and negative classes than standard RFs, being thus more discriminative. This is shown in the right-most column of Fig. 6, where we plot the confidence value of Eq. 1 for each of the classes.

D. Online Learning

The offline training procedure described in the previous section can be done in about one minute (for $M \approx 3$ features and $J \approx 20$ trees). Then, at runtime, the resulting classifier is evaluated over the input data and it is continuously updated in order to adapt to potential changes undergone by the target object.

As shown in the approach overview in Fig. 1, during the online learning process, new detections are fed into the classifier to update the posterior probabilities. These samples are labeled as either positive, corresponding to the target, or negative, when they correspond to the background.

The labeling is done based on the confidence value about the input sample x computed using Eq. 1. If a sample x with pose w has a confidence value $\text{conf}(x) > \beta$, it is assigned to the positive class $+w$. Otherwise, the sample is considered negative $-w$. The parameter β is the threshold of the classifier and to reduce the risk of misclassification it is set to the Bayes error rate. Yet, since an incorrect labeling might lead to drifting problems and loss of the target, we make use of a more rigorous rejection criterion [2], and we set a confidence interval γ around β to indicate predictions with

Algorithm 1: Feature Selection & Building the Classifier

Input:
-J: Number of Ferns.
-M: Number of binary features.
-$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$: Training dataset consisting of N image samples $x \in \mathcal{X}$, where $y_i \in \{+w, -w\}$ is the label for the target and background classes with pose w, respectively.
Output: Visual target classifier $H(x)$.

for $j = 1; j \leq J$ **do**
  Sample at random a reduced set of images $\mathcal{S} \subseteq \mathcal{D}$ from the training data $\mathcal{D}$.
  for $m = 1; m \leq M$ **do**
    Compute a set of K random binary features.
    for $k = 1; k \leq K$ **do**
      Test feature f_k on the sample set $\mathcal{S}$.
      Compute the entropy of the current Fern j, $\mathcal{E}(F_j) = \sum_{z=1}^{2^m} -\frac{N_{j,z}}{|S|}\mathcal{E}(\mathcal{H}_z)$
    Select the feature f_m that minimizes $\mathcal{E}(F_j)$.
    Add feature f_m to the Fern $f_m \rightarrow F_j$.
  Assemble the computed Ferns $F_j \rightarrow \mathbf{F}$.

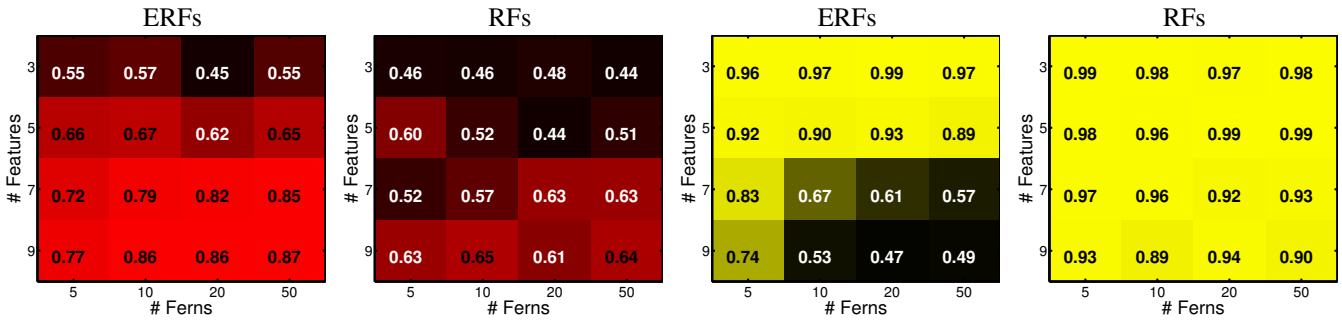

Fig. 7. **2D classification problem**. Evaluation of ERFs (first and third columns) against RFs (second and fourth columns). **Left:** Classification performance of both classifiers measured trough precision-recall rates. **Right:** Degree of overlapping between positive and negative class distributions.

ambiguous confidence values. Samples within this interval are not further considered in the updating process.

The labeled samples that pass the confidence test are then used to recompute the prior probabilities $\theta_{j,z,w}$ of Eq. 2, and update the classifier. For instance, let us assume that a sample x is labeled as $+w_i$, and that it activates the output z of the fern F_j, i.e, $F_j(x) = z$. We will then update the classifier by adding one unit to the i-th bin of the histogram of $N^{+w}_{j,z}$. This is repeated for all ferns. With these new distributions, we can recompute the priors $\theta_{j,z,w}$, and thus, update the classifier.

IV. Experiments

We next evaluate the proposed method, dubbed ERFs (for Entropy-based Random Ferns), using both synthetic data and real experiments of detection of natural landmarks.

A. 2D Classification Problem

This experiment has already been presented in the previous section to evaluate different characteristics of the ERFs and compare them against a classifier whose Ferns are computed completely at random (RFs). In Fig. 6 we have already shown some qualitative results that visually demonstrate the advantages of our approach. We next present a more in-depth analysis of two approaches.

We first analyze the amounts of Ferns and features used to compute both types of classifiers. Fig. 7-(two leftmost plots) represents the classification performance of these approaches through the Equal Error Rate (EER) over precision and recall scores. Note that the classification rates grow with the size of the classifier, and that ERFs consistently obtain higher classification rates than RFs, even when for smaller amounts of features and Ferns.

Another advantage of the proposed classifier is that it yields larger degrees of separability between the positive and negative classes. We already qualitatively observed this in Fig. 6. In the new Fig. 7-(two rightmost plots) we numerically demonstrate this using the Bhattacharyya coefficient, that measures the amount of overlap between distributions. We clearly see that ERFs provide lower coefficients than the classifier computed at random (RFs). This is critical for online learning and detection, as it reduces the misclassification error and possible drifting problems.

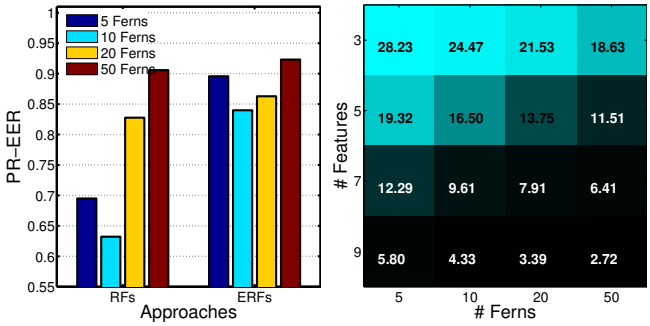

Fig. 8. **Detection of ground patches**. **Left:** Detection rates according to the number of Ferns used to compute the classifier. **Right:** Speed of the classifier (frames per second) for different amounts of features and Ferns.

B. Detection of Ground Patches

We next use the ERFs classifier to detect specific patches on the ground, in a field containing a mixture of grass and soil. While this is a very useful task for detecting landing areas for UAVs it is extremely challenging, due to the presence of many similar patterns, and the lack of salient and recognizable visual marks. Fig. 12-(top, middle) shows a few sample images.

Like in the previous experiment, we again compare the ERFs and RFs. To this end, we evaluate the classifiers in a video sequence containing 150 images of a ground field, that suffers from several artifacts, such as sudden camera motions, and light and scale changes (see Fig. 12-top). In this experiment, we considered 9 features per Fern. The detection performance rate of both methods are presented in Fig. 8-left, where we detail the PR-EER (Equal Error Rate over the Precision-Recall curve) values for classifiers trained with different numbers Ferns. Note again that the ERFs classifier yields better results and is less sensitive to the number of Ferns, thus allowing for more efficient evaluations. This is verified in Fig. 8-right where we provide the computation time of the classifiers in frames per second. Some sample images with the outputs of the ERFs (red circles) and the RFs (green ones) are depicted in Fig. 12-top. Observe that the ERFs are able to accurately detect the visual target, even when it is difficult for the human eye.

Fig. 12-middle shows another experiment of recognizing ground landmarks. This experiment contains 64 images where the target appears at multiple locations and under

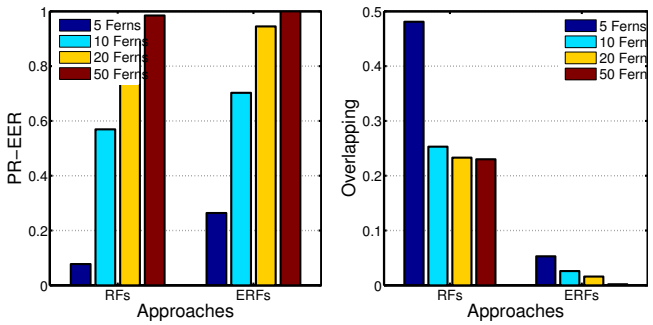

Fig. 9. **Landmark detection performance.** ERFs are evaluated and compared to RFs using different number of Ferns (left), and according to the degree of overlapping between the target and background classes (right).

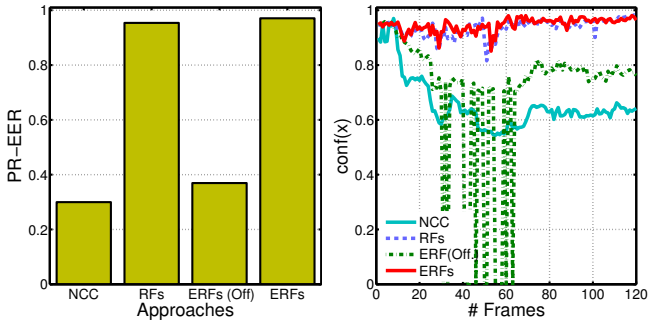

Fig. 10. **Detection of 3D objects**. ERFs are assessed to learn and detect 3D objects. **Left:** Detection rates. **Right:** Output of the classifier conf(x).

various rotations. In this experiment, the classifiers are trained with $W = 16$ in-plane possible orientations. The detection rates of both the ERFs and the RFs are shown in Fig. 9-left. Again, the ERFs provide better results. In addition, if we analyze the degree of overlapping between the target and background classes through the Bhattacharyya coefficient (Fig. 9-right), we see that ERFs provide much higher separation of classes, and therefore, much higher confidence values in its detection. Observe in Fig. 12-middle a few sample results where both the position and orientation of the target are correctly estimated. Indeed, the proposed method yields a detection rate over 95% (PR-EER) and an orientation accuracy of 93%.

C. 3D Object Detection

We have also tested our approach in objects that do not satisfy the assumption we made of having a depth which is negligible compared to its distance to the camera. Fig. 12-bottom shows a few samples of a 120 frames sequence of a bench seen from different viewpoints and scales.

In this case we have included in the analysis a template matching approach based on Normalized Cross Correlation (NCC), widely used for detecting specific objects. The recognition results of all methods are summarized in Fig. 10-left. Observe that the performance of NCC is quite poor. This is because a plain NCC template matching can not adapt the appearance changes produced different viewpoints. The same limitation would suffer our approach without the online adaption, shown in the figure as ERFs (Off). This behavior is also reflected in Fig. 10-right that plots the confidence conf(x) of each classifier along the sequence. ERFs (Off.)

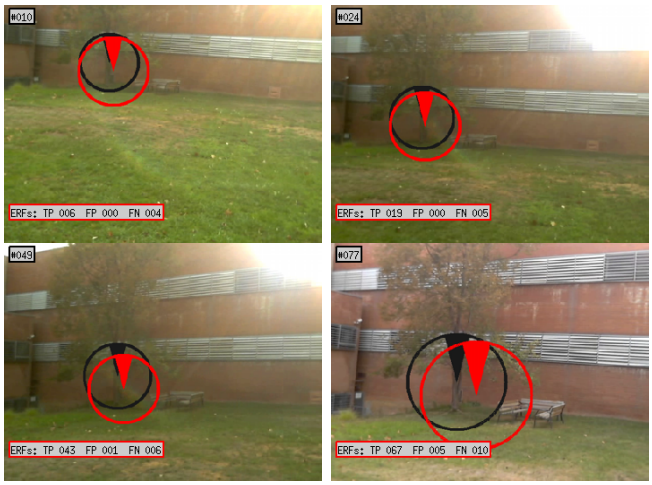

Fig. 11. **3D target detection.** Our approach is able to learn and detect 3D targets in outdoor environments. Red circle indicates the classifier output, whereas black one is the ground truth or visual target.

and NCC give very high scores for the first frames, but these values rapidly fall when the viewpoint changes. On the other hand, the online approaches continue updating the classifiers with new incoming samples and maintain high recognition scores.

The circles in Fig. 12-bottom, represent the detection results of the ERFs (red), NCC (cyan) and ground truth (black), for a few sample frames. Note that the ERFs are able to effectively handle viewpoint change.

Finally, Fig. 2 (which we already introduced in Sect. II) and Fig. 11 show additional examples where our classifier is used to detect 3D objects, in this case specific trees, which can help for performing UAV navigation and obstacle avoidance tasks. Our ERFs classifier is able to learn these visual landmarks on the fly and to detect them despite illumination variations, self-occlusions, viewpoints changes and repetitive textures. As we already showed in Fig. 2-top, methods based on feature-point descriptors like SIFT [14] would not succeed in theis kind of scenarios, as they would seldom find good matches.

V. Conclusions

We have proposed an efficient and robust vision-based approach for learning and detecting natural targets in outdoor environments. Applications like UAV landing area detection of obstacle avoidance benefit from this outcome, and contrasts with the most of recent detection approaches which typically rely on artificial markers spread over the scene. Our solution includes an online classifier that is used to learn the model of the target on the fly and is able to update that model online with the new incoming observations. In addition, we have proposed a theoretically grounded strategy based on Entropy minimization to guarantee a high discriminative power of the classifier while keeping its efficiency. The advantages of our method are demonstrated by thorough testing on both synthetic data and real scenes with natural targets.

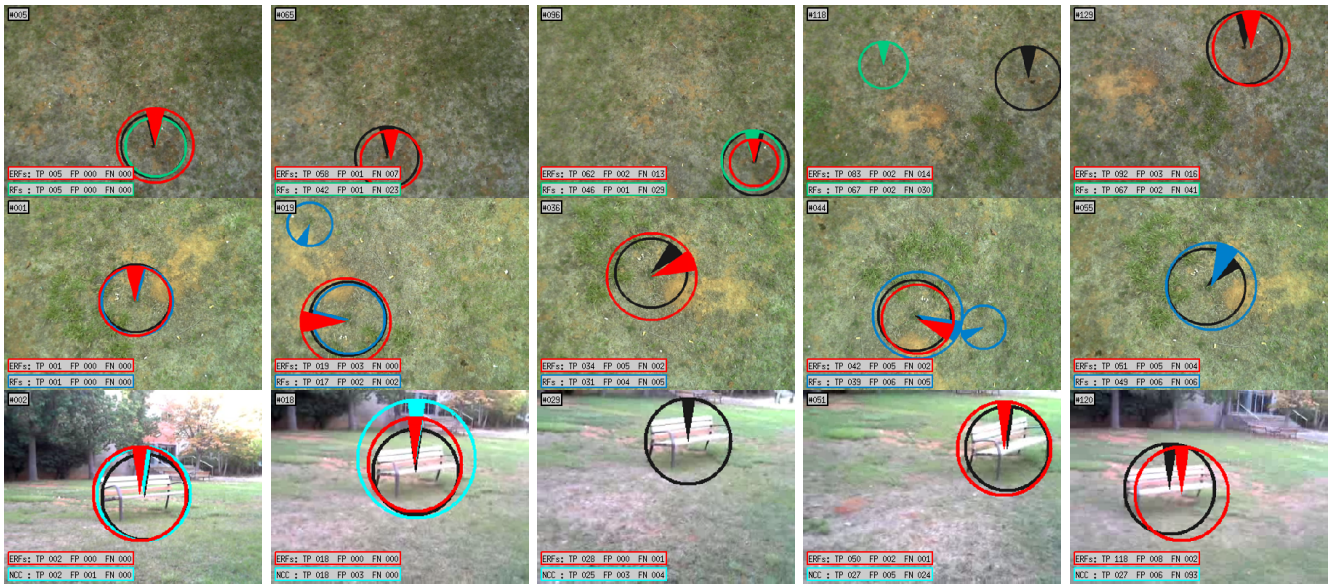

Fig. 12. **Visual target detection**. Output of the proposed approach (red circles) for detecting ground targets (top, middle) and 3D objects (bottom). Black circles denote the location of the targets, whereas the rectangle shows the detection rates: true positives (TP), false positives (FP) and false negatives (FN).

VI. Acknowledgments

This work was partially supported by the projects PAU+ DPI2011-27510, RobTaskCoop DPI2010-17112, ERA-Net Chistera project ViSen PCIN-2013-047, and by the EU project ARCAS FP7-ICT-2011-28761.

References

[1] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded up robust features. In *ECCV*, 2006.

[2] C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[3] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[4] A. Cesetti, E. Frontoni, A. Mancini, P. Zingaretti, and S. Longhi. A vision-based guidance system for uav navitagion and safe landing using natural landmarks. *Journal of intelligent and robotic systems*, 57(1):233–258, 2010.

[5] A. Criminisi, J. Shotton, and E. 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, 2011.

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

[7] Y. Fan, S. Haiqing, and W. Hong. A vision-based algorithm for landing unmanned aerial vehicles. In *International Conference on Computer Science and Software Engineering*, pages 993–996, 2008.

[8] 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.

[9] G. Flores, S. Zhou, R. Lozano, and P. Castillo. A vision and gps-based real-time trajectory planning for mav in unknown urban environments. In *ICUAS*, 2013.

[10] S. Hinterstoisser, V. Lepetit, P. Fua, and N. Navab. Dominant orientation templates for real-time detection of texture-less objects. In *CVPR*, 2010.

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

[12] J. Kim and D.H. Shim. A vision-based target tracking control system of a quadrotor by using a tablet computer. In *ICUAS*, 2013.

[13] V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *PAMI*, 28(9):1465–1479, 2006.

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

[15] A. Masselli, S. Yang, K.E. Wenzel, and A. Zell. A cross-platform comparison of visual marker based approaches for autonomous flight of quadrocopters. In *ICUAS*, 2013.

[16] D. Mellinger, M. Shomin, N. Michael, and V. Kumar. Cooperative grasping and transport using multiple quadrotors. *Distributed autonomous robotic systems*, pages 545–558, 2013.

[17] N. Michael, D. Mellinger, Q. Lindsey, and V. Kumar. The grasp multiple micro-uav testbed. *Robotics and Automation Magazine*, 17(3):56–65, 2013.

[18] I.F. Mondragon, P. Campoy, J.F. Correa, and L. Mejias. Visual model feature tracking for uav control. In *IEEE International Symposium on Intelligent Signal Processing*, pages 1–6, 2007.

[19] F Moreno-Noguer, J Andrade-Cetto, and A Sanfeliu. Fusion of color and shape for object tracking under varying illumination. In *Pattern Recognition and Image Analysis*, pages 580–588, 2003.

[20] F. Moreno-Noguer, V. Lepetit, and P. Fua. Pose priors for simultaneously solving alignment and correspondence. In *ECCV*, 2008.

[21] F. Moreno-Noguer, A. Sanfeliu, and D. Samaras. Integration of deformable contours and a multiple hypotheses fisher color model for robust tracking in varying illuminant environments. *Image and Vision Computing*, 25(3):285–296, 2007.

[22] M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *PAMI*, 32(3):448–461, 2010.

[23] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *CVPR*, 2009.

[24] J.L. Sanchez-Lopez, S. Saripalli, P. Campoy, J. Pestana, and C. Fu. Toward visual autonomous ship board landing of a vtol uav. In *ICUAS*, 2013.

[25] E. Serradell, M. Ozuysal, V. Lepetit, P. Fua, and F. Moreno-Noguer. Combining geometric and appearance priors for robust homography estimation. In *ECCV*, 2010.

[26] J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *CVPR*, 2008.

[27] A. Symington, S. Waharte, S. Julier, and N. Trigoni. Probabilistic target detection by camera-equipped uavs. In *ICRA*, 2010.

[28] 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.

[29] M. Villamizar, A. Garrell, A. Sanfeliu, and F. Moreno-Noguer. Online human-assisted learning using random ferns. In *ICPR*, 2012.

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

[31] S. Yang, S.A. Scherer, and A. Zell. An onboard monocular vision system for autonomous takeoff, hovering and landing of a micro aerial vehicle. *Journal of intelligent and robotic systems*, 69(1–4):499–515, 2013.