Chapter
Robust Perception for Aerial Inspection: Adaptive and On-line Techniques

M. Villamizar, A. Sanfeliu

Abstract This chapter explains an adaptive on-line object detection and classification technique for robust perception due to varying scene conditions, for example partial cast shadows, change on the illumination conditions or changes in the angle of the object target view. This approach continuously updates the target model upon arrival of new data, being able to adapt to dynamic situations. The method uses an on-line learning technique that works on real-time and it is continuously updated in order to adapt to potential changes undergone by the target object. The method can run in real-time.

1 Introduction

For aerial robots, the recognition of objects and places plays an important role for diverse aerial tasks such as autonomous robot landing and localization. However, the detection of visual targets is a very challenging problem because the varying scene conditions such as light changes. This is particularly critical in outdoors where the environmental conditions change suddenly.

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 infra-red cameras. Tasks such target detection [7, 9, 10], navigation [5, 18] and landing [4, 13] 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.

Following, we propose an efficient algorithm for detecting the pose of natural landmarks on input video sequences without the need of using external markers [17]. This is especially remarkable, as there are 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 [8, 11]. In addition, the 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.

At the core of the 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 off-line 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.

As shown in Fig. 1, the 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 ro-

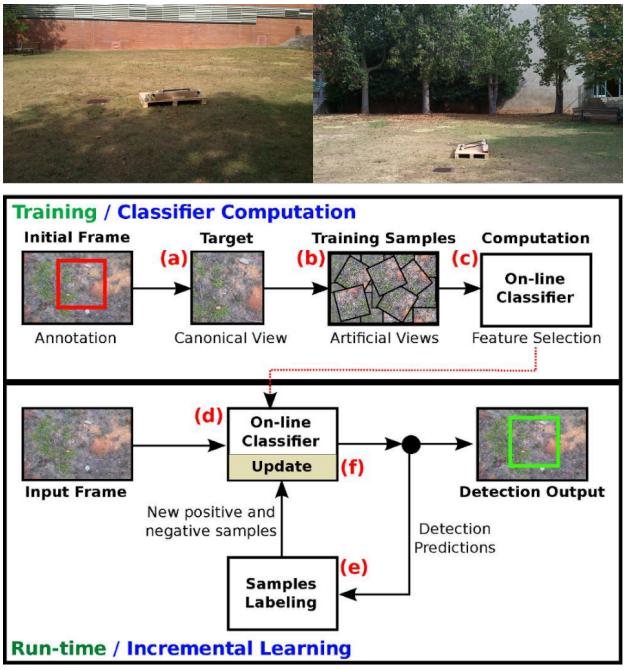

Fig. 1 Detecting natural landmarks. Top: Kind of outdoor scenario we consider. Some of the challenges the detector needs to address are light changes, shadows and repetitive textures. Bottom: Schematic of the approach. It consists of two stages, an off-line learning stage where a general model of the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

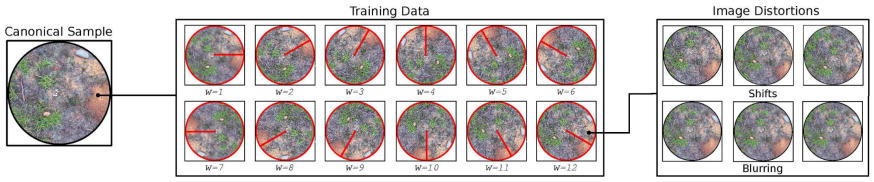

Fig. 2 Synthetic training data. The canonical sample (left) is synthetically warped to generate new training samples (middle). These samples are computed at different orientations and at different shift and blurring levels (right). The red circle and arrow indicate the target pose for each sample.

tations, 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)). The proposed classifier is a new version of Random Ferns [12] that uses an entropy reduction criterion to compute the most discriminative features. This classifier, dubbed Entropy-based Random Ferns (ERFs), allows to both minimize the number of Ferns to represent the target (making the algorithm more efficient), and to maximize the discriminative power. All this initial training is performed off-line, in matter of minutes.

In the second stage (Fig. 1(d)) the ERFs 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.

Next, we describe the main steps for building the classifier: generation of an initial set of synthetic samples, off-line construction of the classifier, a new criteria for selecting the features and finally, the on-line adaption of the algorithm.

2 Generating synthetic samples for off-line training

Initially it is assumed only one single sample of the target to be detected. 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, synthetically are generated 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. It is therefore considered the canonical target as being planar, and the approximation of multiple views can done through in-plane rotations. Note, however, that the method 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. 2, 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 $w \in \{1,2,..,W\}$

Table 1 Symbols used in the development of the Random Fern based classifier for adaptive perception.

Definition	Symbol
Classifier classes $w \in 1,2,..W$ (pose orientation)	w
Estimated pose of w	$\hat{w}$
Class label $y_i = \{+w_i, -w_i\}$ (if belongs to a classifier class or background)	y_i
Sample $x_i \in \mathscr{X}$ in image space $\mathscr{X}$	x_i
Initial training dataset $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^{N}$	$\mathscr{D}$
Image $x()$ value at pixel coordinates (u,v), with color channel c	$x(u,v,c)$
Binary features	f_i^j
Fern $F_j = \{f_1^j, f_2^j, \ldots, f_M^j\}$ consisting of a set of M binary features	F_j
Fern output, $F(x) = z = (f_1, \ldots, f_M)_2 + 1$	z
Indicator function	$\mathbb{I}(e)$
Classifier response for sample x	$H(x)$
Classifier confidence	$conf(\cdot)$
Classifier parameters	$\boldsymbol{\theta}$
Probability a sample in Fern j belongs to positive class with pose w at output z	$\theta_{j,z,w}$
Shannon Entropy of Fern F_j	$\mathscr{E}(F_j)$
Distribution of samples across poses w in the leaf z	$\mathscr{H}_z$

there are further included 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. Let us denote this whole initial training dataset as $\mathscr{D} = \{(x_i, y_i)\}_{i=1}^{N}$ where $x_i \in \mathscr{X}$ corresponds to a sample in the image space $\mathscr{X}$, 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.

3 Building the classifier

In order to perform on-line learning and object detection, it is used Random Ferns (RFs) [12, 15]. This classifier can be understood as an extreme and very fast implementation of a random forest [3] 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 [6, 16]. 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 [3, 12]. It is shown this in Fig. 3-left, where there can seen two Ferns F, each one with two decision tests or binary features f.

More formally, the on-line 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

$$f(x) = \mathbb{I}(x(u_1, v_1, c_1) > x(u_2, v_2, c_2)) \tag{1}$$

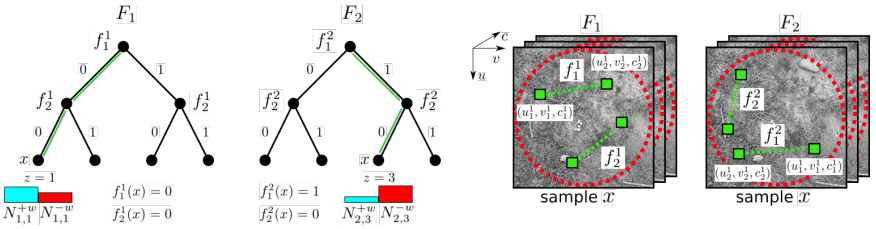

Fig. 3 Fern-based classifier. Computation of the classifier using $J = 2$ Ferns with $M = 2$ binary features. Left: Schematic representation of the Ferns structure using binary trees. At the bottom of the tree it is plotted the distributions which are updated for a training sample x. For instance, assuming 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.

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:

$$\mathbb{I}(e) = \begin{cases} 1 & \text{if } e \text{ is true} \\ 0 & \text{if } e \text{ is false} \end{cases} \tag{2}$$

As it will described in the following section, and in contrast to the original Ferns formulation [12], the location of these pairs of pixels is computed during the training stage according to a criterion of entropy minimization. Fig. 3-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. 3-left).

So far, there 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 $\underline{\mathrm{F}} = \{F_1, \ldots, F_J\}$. The response of the classifier, for an input sample image x can be written as

$$H(x) = \begin{cases} +1 \text{ if } \mathrm{conf}(x \in \hat{w}) > \beta \\ -1 \text{ otherwise,} \end{cases} \tag{3}$$

where $\hat{w}$ is the estimated pose of the sample, $\mathrm{conf}(x \in \hat{w})$ 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:

$$\mathrm{conf}(x \in \hat{w}) = p(y = +\hat{w} | \mathbf{F}(x), \boldsymbol{\theta}), \tag{4}$$

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.:

$$\hat{w} = \arg\max_{w} p(y = +w|\mathbf{F}(x), \boldsymbol{\theta}), \quad w = 1, \ldots, W \tag{5}$$

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

$$p(y = +w|\mathbf{F}(x), \boldsymbol{\theta}) = \frac{1}{J} \sum_{j=1}^{J} p(y = +w|F_j(x) = z, \theta_{j,z,w}), \tag{6}$$

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

$$p\left(y|F_j(x) = z, \theta_{j,z,w}\right) \sim \text{Ber}(y|\theta_{j,z,w}), \tag{7}$$

with we can write that

$$p\left(y = +w|F_j(x) = z, \theta_{j,z,w}\right) = \theta_{j,z,w} \tag{8}$$

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

$$\theta_{j,z,w} = \frac{N_{j,z}^{+w}}{N_{j,z}^{+w} + N_{j,z}^{-w}} \tag{9}$$

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. 3-left for an illustrative example.

4 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 [6, 12, 16]. In this work, it is proposed Entropy-based Random Ferns (ERFs) to select the most relevant and discriminative binary features, resulting in a classifier with increased levels of efficiency and robustness.

For this purpose, it is used a methodology to choose the binary features that reduce the classification error over the training data $\mathscr{D}$. As an approach to this, it will

be looked for the features that minimize the Shannon Entropy $\mathscr{E}$, 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 [2, 14].

Specifically, each Fern F_j is independently computed from the rest of Ferns, and using a different and small random subset $\mathscr{S} \subseteq \mathscr{D}$ of the training data. Partitioning the training data will avoid potential over-fitting errors during testing [2, 3]. Let us now assume 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, it will be chosen the binary feature f_m that minimizes the entropy of the Fern $\mathscr{E}(F_j)$, computed as

$$\mathscr{E}(F_j) = \sum_{z=1}^{2^m} -\frac{N_{j,z}}{|S|}\mathscr{E}(\mathscr{H}_z), \quad \mathscr{E}(\mathscr{H}_z) = -\mathscr{H}_z \log \mathscr{H}_z, \tag{10}$$

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 $\mathscr{H}_z$ is the distribution of samples across poses w in the leaf z, and is represented trough a normalized histogram.

Once the feature f_m that minimizes $\mathscr{E}(F_j)$ 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.

5 On-line learning

The off-line 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 on-line 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. 4. 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, it can be used of a more rigorous rejection criterion [1], by means of a confidence interval γ around β to indicate predictions with 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. 9, 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, $F_j(x) = z$. It will be then updated the classifier by adding one unit to

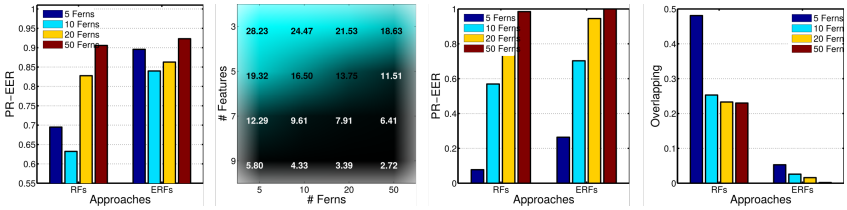

Fig. 4 Detection of ground patches. Left: Detection rates according to the number of Ferns used to compute the classifier. Center-left: Speed of the classifier (frames per second) for different amounts of features and Ferns. Center-right: ERFs and RFs comparison using different number of Ferns. Right: Degree of overlapping between the target and background classes.

the i-th bin of the histogram of $N_{j,z}^{+w}$. This is repeated for all ferns. With these new distributions, there are recomputed the priors $\theta_{j,z,w}$, and thus, updated the classifier.

6 Experiments

Four different experiments were done to show the robustness of the method applied to different type of images. The first one is the detection of ground patches that can occur when the UAV tries to find the landing area. The second one is for the detection of a specific object, a bench in a park, when the UAV is flying. The third one is the detection of a specific pipe feature when the UAV is following the pipe, and can be used for example, for the detection of the initial reference point in a pipe to be inspected. Finally the fourth one is the line tracking in a pipe and the detection when the line is lost.

6.1 Detection of ground patches

Let us use ERFs 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. 6-(top, middle) shows a few sample images.

Following, let us compare the detection performance of ERFs and RFs. To this end, let us 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. 6-top). In this experiment, it is considered 9 features per Fern. The detection performance rate of both methods are presented in Fig. 4-left, where we detail the PR-EER (Equal Error Rate over the Precision-Recall curve) values for classifiers trained with different numbers Ferns. Note that the ERFs classifier yields better results and is less sensitive to the number of Ferns,

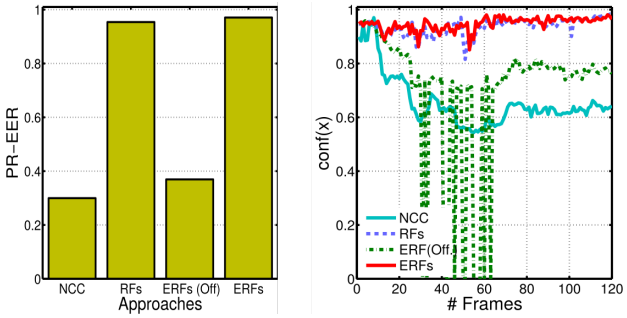

Fig. 5 Detection of 3D objects. ERFs are assessed to learn and detect 3D objects. Left: Detection rates. Right: Output of the classifier conf(x).

thus allowing for more efficient evaluations. This is verified in Fig. 4-center-left where there are 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. 6-top. Observe that the ERFs are able to accurately detect the visual target, even when it is difficult for the human eye.

Fig. 6-middle shows another experiment of recognizing ground landmarks. This experiment contains 64 images where the target appears at multiple locations and under 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. 4-Center-right. Again, the ERFs provide better results. In addition, if it is analyzed the degree of overlapping between the target and background classes through the Bhattacharyya coefficient (Fig. 4-right), we see that ERFs provide much higher separation of classes, and therefore, much higher confidence values in its detection. Observe in Fig. 6-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%.

6.2 3D object detection

There have been also tested our approach in objects that do not satisfy the assumption of having a depth which is negligible compared to its distance to the camera. Fig. 6-bottom shows a few samples of a 120 frames sequence of a bench seen from different viewpoints and scales.

In this case there have been 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. 5-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 on-line adaption,

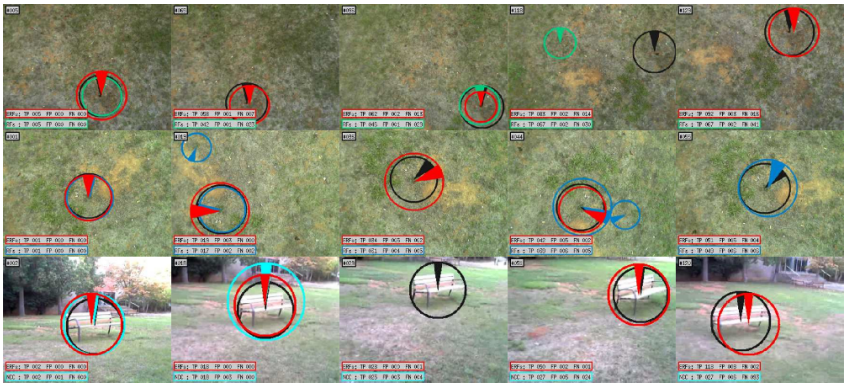

Fig. 6 Visual target detection. Output of the proposed approach (red circles) for detecting ground targets (top, middle) and 3D objects (bottom). Black circles denote the location of the targets, whereas the rectangle shows the detection rates: true positives (TP), false positives (FP) and false negatives (FN).

shown in the figure as ERFs (Off). This behavior is also reflected in Fig. 5-right that plots the confidence conf(x) of each classifier along the sequence. ERFs (Off.) and NCC give very high scores for the first frames, but these values rapidly fall when the viewpoint changes. On the other hand, the on-line approaches continue updating the classifiers with new incoming samples and maintain high recognition scores.

The circles in Fig. 6-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. 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.

6.3 Pipe feature detection

We have also applied our technique in the detection of a pipe feature, for example a pipe welding. In this case, first the approach learn the pattern to be detected in the following image frames of the video sequence. This pattern can be given in advance or it is automatically detected and captured by the vision system. In our case, the pattern is giving by the operator. Fig. 7-top left shows the first image frame where the pipe feature is captured (inside of a red rectangle). The next frames in Fig. 7-top and Fig. 7-bottom show several situations where the pipe feature does not appear or other pipe features are present, but are different from the learned pattern. As it can be seen, the system only detects the pipe welding when it appears in the image frame and in the rest of the image frames the system does not detect anything. The system was tested in a large sequence of pipe images.

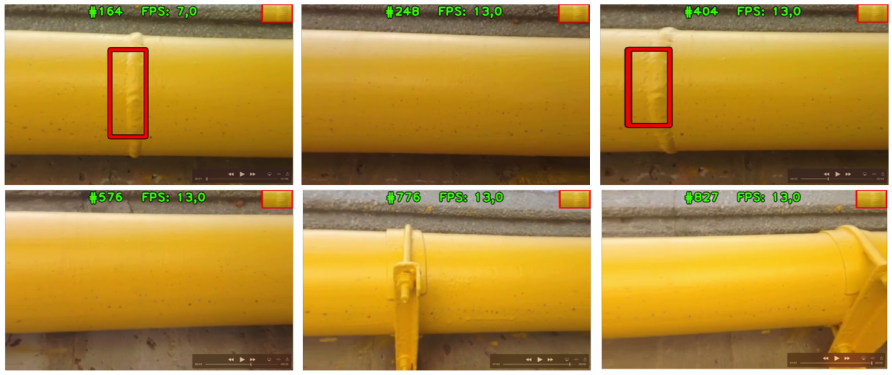

Fig. 7 Detection of a welding in a pipe

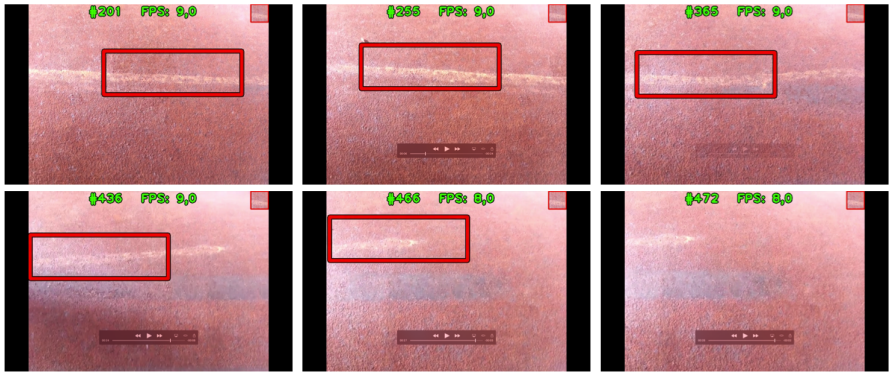

Fig. 8 Detection of a line in a pipe

6.4 Line tracking in a pipe

Finally, we have also tested our approach in following a white line in a pipe, where the color and pipe features are changing at every frame of the video. Fig. 8-top and Fig. 8-bottom show some of the frames of the video.

The detection process works very well although the line have slight changes, but it stop when the line is lost.

7 Conclusions

This chapter explains an adaptive and on-line technique for robust perception due to varying scene conditions, for example partial cast shadows, change on the illu-

mination conditions or changes in the angle of the object target view. This approach continuously updates the target model upon arrival of new data, being able to adapt to dynamic situations. The core of the technique lies in a Random Fern classifier that models the posterior probability of different views of the target using multiple and independent Ferns, each containing features at particular positions of the target object. The technique uses a Shannon entropy measure to pick up the most informative locations of these features, minimizing the number of Ferns while maximizing its discriminative power. During the on-line learning the method works on real-time and it is continuously updated in order to adapt to potential changes undergone by the target object. The method is demonstrated in four different experiments, the first one detecting specific patches on the ground; the second one, detecting specific objects in the aerial scene, for example a bench in a park; and the third and four ones detecting features and lines in a pipe.

References

1. C. M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.
2. Leo Breiman. Random forests. *ML*, 45(1):5–32, 2001.
3. A. Criminisi, J. Shotton, and E. Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *FTCGV*, 7(2–3):81–227, 2011.
4. Y. Fan, S. Haiqing, and W. Hong. A vision-based algorithm for landing unmanned aerial vehicles. In *ICCSSE*, pages 993–996, 2008.
5. 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*, pages 1150–1155, 2013.
6. Z. Kalal, J. Matas, and K. Mikolajczyk. P-N learning: Bootstrapping binary classifiers by structural constraints. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 49–56, 2010.
7. J. Kim and D.H. Shim. A vision-based target tracking control system of a quadrotor by using a tablet computer. In *ICUAS*, pages 1165–1172, 2013.
8. V. Lepetit and P. Fua. Keypoint recognition using randomized trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, 2006.
9. 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*, pages 685–693, 2013.
10. I.F. Mondragon, P. Campoy, J.F. Correa, and L. Mejias. Visual model feature tracking for uav control. In *WIPS*, pages 1–6, 2007.
11. F. Moreno-Noguer, V. Lepetit, and P. Fua. Pose priors for simultaneously solving alignment and correspondence. In *Proc. of the IEEE Europ. Conf. on Computer Vision (ECCV)*, volume 2, pages 405–418, 2008.
12. M. Ozuysal, M. Calonder, V. Lepetit, and P. Fua. Fast keypoint recognition using random ferns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(3):448–461, 2010.
13. 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*, pages 779–788, 2013.
14. J. Shotton, M. Johnson, and R. Cipolla. Semantic texton forests for image categorization and segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.
15. M. Villamizar, J. Andrade-Cetto, A. Sanfeliu, and F. Moreno-Noguer. Boosted random ferns for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

16. M. Villamizar, A. Garrell, A. Sanfeliu, and F. Moreno-Noguer. Online human-assisted learning using random ferns. In *Proc. International Conference on Pattern Recognition (ICPR)*, pages 2821–2824, 2012.
17. M. Villamizar, A. Sanfeliu, and F. Moreno-Noguer. Fast online learning and detection of natural landmarks for autonomous aerial robots. In *Proc. of the IEEE Int. Conf. on Robotics & Automation (ICRA)*, pages 4996–5003, 2014.
18. S. Yang, S.A. Scherer, and A. Zell. An onboard monocular vision system for autonomous takeoff, hovering and landing of a micro aerial vehicle. *JIRS*, 69(1–4):499–515, 2013.