Playground Learning for Object Detection

Hugo Penedones and François Fleuret

Idiap Research Institute, Martigny, Switzerland

Abstract. We propose the general concept of learning in a "playground", which consists of building a strong prior from simple and controlled training data to help discriminative machine learning. Our main contribution is to introduce an operational procedure following that philosophy for the problem of object detection.
Under the assumption we can control the data collection process, we record a video sequence from which one can unsupervisedly build a prior model of the object's possible deformations. This information is then combined with noisy images generated from the same video, in order to train a binary classifier using standard machine learning techniques. The advantage of this new procedure is that the classifier does not need to infer the invariances to complex hidden-states, such as camera viewpoints and object poses. The result is a global appearance model, able to cope with out-of-plane rotations or other complex deformations seen in the training video sequence.
We demonstrate the increase of performance with respect to a standard sliding-window approach on difficult gray-scale images taken in a cluttered environment.

1 Introduction

Object detection in natural scenes is a core challenge in computer vision. There have been tremendous advances during the last decade, and applications such as face detection for digital cameras, pedestrian detection for car safety systems or vehicle detection for road surveillance are now handled automatically with performance sufficient for industrial applications.

However, raw performance remains far from humans' in terms of absolute error rate, and even state-of-the-art detection systems may produce false positives which can be trivially rejected by a human.

In this paper, we propose a novel strategy for machine learning in general and object detection in particular which breaks the learning into two steps. The first one is what we call the "playground". It uses a high-quality training signal produced in a controlled manner. Such data does not reflect the test distribution, but the absence of noise facilitates the discovery of its hidden structure, even with simple unsupervised techniques. From this first step, the algorithm produces a strong model used as a prior for the standard discriminative learning procedure of the second step.

In practice we use an unlabeled training video where the target object is visible in close up on a uniform background. During a first learning step, our

algorithm collects a family of sequences of pairs of location and scale, each likely to correspond to a fixed physical point on the object. Given any image, we can use these as "local referentials" to move the features extractors to compensate not only for changes in scale and location, but also for non-affine deformations observed in the training sequence. A second training step builds a linear classifier which takes these features as input and predicts if they correspond to the presence of the object. The training data for this step is made of artificially degraded images from the video, super-imposed over background images.

This approach can be seen as an extension of classical detection methods with a sliding window. Instead of moving "what to measure" in location and scale, an additional discrete pose parameter encodes the deformations appearing in the training video. It can also be seen as a generalized template matching in which instead of computing the distance to deformed examples, we deform the feature extraction and combine it with a discriminative predictor.

The assumptions underlying this approach are that (1) standard part-based tracking or detection methods can only be used reliably on controlled images of reasonable resolution, and (2) machine learning is necessary but requires a strong prior model to know where measurements should be done to normalize geometrical variability (location, scale, rotation in and out of the plan, etc.)

2 Related works

2.1 Monolithic object detection

The most straight-forward approach to object detection consists of scanning the space of locations and scales in the test image, and to repeatedly evaluate the response of a classifier trained on a population normalized in position and scale [1, 2]. Conditioning on the location and scale alleviates the requirement to learn such invariances from data.

The same approach has been extended to additional pose parameters such as the tilt in the image plane by training multiple detectors dedicated to homogeneous sub-populations [3]. In such a case, one has to cope with the reduction of the available training data, which is fragmented in as many dedicated sub-sets. However, in certain cases this issue can be handled by generating synthetically training images. Still, the number of classifiers to train increases with the resolution of the pose discretization, and for most of the additional degrees of freedom, there is no practical way to synthesize training data.

Recently, a natural generalization of the standard scanning process with a single classifier has been proposed for arbitrary pose parameters [4]. The core idea behind this work is to design features able to compensate for known changes of the additional degrees of freedom. While this new approach avoids the fragmentation of the training set into several sub-sets, it moves the burden to the manual design of the *pose-indexed features*, and requires a rich annotation of the training data. We can relate our technique to that work, and see our "playground" learning part as a way to learn pose-indexed features automatically from data, instead of defining them *a priori*.

2.2 Part-based detection

Another general strategy consists of first detecting multiple pieces which have been observed to be likely on the object during training, and to combine these various cues into a final detection decision.

This has been applied with success either by using invariant signatures with an exhaustive catalog of points matched in test under strong geometrical constraints [5] or by using less invariant signatures such as cross-correlation with a clustering of the space of signatures, and a less strict geometrical constraint [6–8].

The boundary with monolithic methods described in the previous section is blurred by approaches which construct large catalogs of patches, which play the role of generic local features in a parsing-like process [9].

Also, a generative model was proposed for the global object appearance as patchwork of parts [10]. Each part is modeled under an assumption of conditional independence given its location, and the part locations are modeled jointly as a Gaussian distribution. This approach relates strongly to ours since it relies on matching the parts jointly. However, we do not model analytically the joint locations of parts but use the empirical distribution observed in the training sequence. Also, we rely on discriminative approach to distinguish between the negative and the positive local part appearances, instead of modeling the distribution of the latter.

Table 1. Notation

T	Number of frames in the training video.
Q	Number of physical points tracked in the training sequence.
$\tilde{\mathbb{R}}$	Set of real numbers extended with the value $n.a.$
σ	Thresholding extended to the value $n.a.$
Θ	Set of triplets (x, y, s).
$\mathcal{C}$	Set of configurations of Q local referentials
c_t	Configuration of the local referentials recorded in training frame t.
$f(u, r, t)$	Pose-indexed predictor evaluated on image u, location and scale r and reference training frame t.
$g(u, r)$	Pose-indexed predictor evaluated on image u at location and scale r.
$r * c$	Configuration of points c moved to the reference frame defined by r.
$\psi_a(u, c)$	Single feature evaluated on image u, for the configuration of local referentials c.
$\Psi(u, r, t)$	Pose-indexed feature evaluated on image u, for location and scale r and reference training-frame t.
D	Dimension of the pose-indexed feature vector Ψ.
Φ	Trained classifier.
M	Number of stumps in the classifier Φ.

3 Playground Trained Detector

Given an image u, a sliding-window detection method computes a set of alarms

$$\Big\{ r \in \mathbb{R}^3_+ \text{ s.t. } g(u,r) \geq \rho \Big\},$$

where $r = (x, y, s)$ should be interpreted as a location and a scale, and $g(u, r)$ is the response of a two-class classifier in image u at that location and scale. This can usually be seen as extracting the sub-image defined by r and feeding it to a classifier. But it can also be seen equivalently – and is often implemented – as measuring features at locations translated and scaled according to r, without the actual creation of an intermediate sub-image in the process.

Our method is a direct extension of that approach with

$$g(u,r) = \max_t f(u,r,t)$$

where t is an additional discrete pose parameter which, as r, modulates the locations where the features will be computed. As in [4], this is a parameterization of the deformation of the feature extraction process itself, beyond scale and translation.

In our novel approach, instead of defining by hand this additional pose parameter, or more abstractly how f is parametrized by t, we exploit an unlabeled training video of the object and use the frame index t itself as a discrete pose. A positive value of $f(u, r, t)$ should be interpreted as

The object is visible in image u, at location and scale r, with the same pose as the one it has in the frame t of the training video.

This new parameter t can be seen as a discretization of the object pose component not captured by the scale and location. If the video shows an object on a turntable, with only one angle varying, than t will be that angle. If the video shows a deformable object with multiple degrees of freedom such as a hand, the parameter t will be a coarse discretization of the corresponding multi-dimensional space. Because we do not build any analytical model besides scale and location, we have an implicit requirement that the video is a reasonable sampling of the deformations and rotations (in and out of the plan) met during test.

We construct f in two steps. During the first step, we track points on the surface of the object in the training video and record their locations in every frame, as depicted in Figure 1 and described in § 3.1. In the second step, we train a two-class classifier Φ with AdaBoost, which takes as input a vector $\Psi(u, r, t)$ of measurements done at the locations recorded during training. The training set for that step is generated from the video, as described in § 3.4 and § 4.3. During detection, the algorithm tries all the deformations seen during training at all possible locations and scales and computes the response for the corresponding feature vector, as depicted in Figure 2 and described in § 3.2 and § 3.4.

Because f depends on one parameter which accounts for non-affine changes in pose, we will refer to it as a *pose-indexed classifier* in the rest of the paper.

3.1 Learning the set of object deformations

We are given a video showing the object in close-up, with virtually no motion blur, no illumination changes, no occlusions, and in front of a uniform background. Our goal is to use this data to automatically create a model of the deformations of the object as a function of the video frame.

To tackle this problem, we take a very simple approach: we track points of interest over the sequence and store their locations at each frame. The result is a series of Q *chains*. Ideally, each chain corresponds to the successive locations and scales of a fixed physical point on the object. More details are provided in § 4.2.

We define $\Theta = \mathbb{R}^3_+$ and interpret it as the set of local referentials (defined by a location and a scale) $((x, y), s)$. Let $\mathcal{C} = (\Theta \cup \{n.a.\})^Q$ be the set of configurations of Q such triplets, each one possibly not available, in which case the corresponding component is $n.a.$ Let $c_t \in \mathcal{C},\ t = 1, \ldots, T$ be the locations and scales of the points recorded in the frame t of the training sequence. They constitute the only information kept from this first training step. In particular we discard the point of interest descriptors.

A certain point has the position $n.a.$ in the frames where it was not yet detected, and again in the frames where it is not detected anymore. Hence, in any frame t, some of the Q points are not visible, and the corresponding components of c_t are equal to $n.a.$

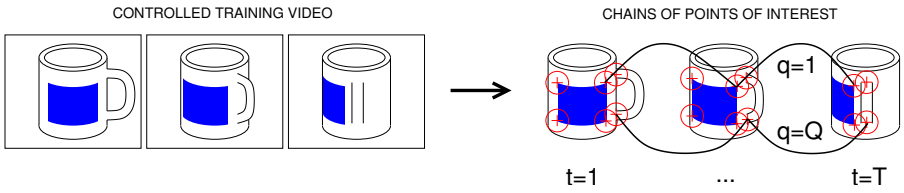

Fig. 1. The first step of the learning consists of matching points of interest in successive frames to generate chains of locations through the video. We define $c_t^q = (x_t^q, y_t^q, s_t^q)$ the location and scale of the point of interest q in frame t. This tracking produces a configuration c_t of Q local referentials in every frame. Note that a point location can also take the value $n.a.$ if the point has not been detected yet or could not be matched anymore.

Given $r = (x, y, s) \in \Theta$ which stands for a reference frame, and $c \in \mathcal{C}$ a configuration of point locations, let $r * c$ denote the configuration scaled and translated according to r, as depicted in Figure 2. The original reference frame is $(0, 0, 1)$, hence $\forall c \in \mathcal{C},\ (0, 0, 1) * c = c$.

We will use this operator to move the configurations c_t at any location and scale in an image to compute the feature responses, as described in § 3.2.

3.2 Pose-indexed features

As described in § 3.4, the predictor $f(u, r, t)$ is a mapping Φ composed with the vector $\Psi(u, r, t)$. Practically, each component of Ψ is attached to one of the Q local referentials, and we measure it for the value of that referential as recorded in c_t.

We first define image features whose responses depend on both an input image and Q local referentials. To cope with the *n.a.* points, let $\widetilde{\mathbb{R}} = \mathbb{R} \cup \{n.a.\}$ be the set of real values extended with the "not available" value, and let

$$\psi_a(u, c) \in \widetilde{\mathbb{R}}$$

be the value of a feature parametrized by a and computed in image u for the configuration of points $c \in \mathcal{C}$.

From the definition of ψ, we define a large pose-indexed feature vector

$$\Psi(u, r, t) = (\psi_{a_1}(u, r * c_t), \ldots, \psi_{a_D}(u, r * c_t)) \tag{1}$$

where the a_d have been generated at random before training, see § 4.3 for details.

With such a definition, we expect $\Psi(u, r, t)$ to be *stationary* [4]: If we fixed $r = (x, y, s)$ and t, and collected a very large number of test images with the object truly at location and scale r and with the same pose as in training frame t, the distribution of $\Psi(u, r, t)$ estimated on these images should not depend on r or t.

Of course, this ideal property can not be achieved exactly, at least because the *n.a.*'s give a strong information about t. Still, given the construction of ψ and Ψ, stationarity is achieved as much as possible.

The dimension D corresponds to the total number of features we will ever consider during training. The classifier we describe in 3.4 uses a very limited subset of components, which are the only ones actually computed during test.

3.3 Feature Extractors

In our actual implementation, we use very simple features, which compute the absolute difference between the average gray-scales of two rectangles. More formally, the parameter a has value in $\{1, \ldots, Q\} \times \mathbb{R}^8$ and defines both the index of a point q and the shapes of the two rectangles. The value $\psi_a(u, c)$ is the difference of gray-scales estimated in u over the two rectangles attached to the location and scale of the point q in c. Hence for different c, the rectangles translate and scale consistently with the point q and remain roughly at the same physical location on the object. Also, ψ takes the value *n.a.* if that point has the location *n.a.*

3.4 Pose-indexed classifier

At the core of our strategy is a pose-indexed classifier f, a two-class predictor which is able to test if a certain image contains the object in a certain pose. This classifier takes as parameters an image u, a location and scale r, and a

training frame index t. The value $f(u, r, t)$ will be positive if there is in u, at r, the object with a pose similar to the pose it had in the training image t, and will be negative otherwise.

We use a standard AdaBoost procedure to train a classifier Φ. It is a linear combination of stumps, each parametrized by a threshold ρ and the index d of a component of the feature vector $\Psi(u, r, t)$ defined in Equation (1) in § 3.2. Combining Φ with the pose-indexed feature vector Ψ results in the pose-indexed classifier f.

To handle the *n.a.* values, let first define $\sigma(z, \rho)$ which is a standard thresholding function extended to the *n.a.* value:

$$\sigma(z, \rho) = \begin{cases} 0 \text{ if } z = n.a. \\ -1 \text{ if } z < \rho \\ 1 \text{ if } z \geq \rho \end{cases}$$

Given an image u, a location and scale $r = (x, y, s)$ and an index t, our pose-indexed predictor f has the form

$$f(u, r, t) = \Phi(\Psi(u, r, t)) \tag{2}$$

$$= \sum_{m=1}^{M} \omega_m \, \sigma(\Psi_{d_m}(u, r, t), \rho_m) \tag{3}$$

where the parameters of Φ, namely the stump weights ω_m, the indexes of the components d_m, and the thresholds ρ_m on their responses, are all chosen during training. Note that the total number of stumps M is small compared to the dimensionality D of $\Psi(u, r, t)$.

Such a classifier mixes stumps, each looking at a particular local referential. By definition, the threshold function σ silences stumps looking at points which are not available in the reference training frame t.

The detection is similar to the standard parsing of locations and scales $r = (x, y, s)$, but adds a fourth loop visiting the values of t.

To generate the training dataset we segment the object in every frame t of the original video, and generate a synthetic image u by copying the segmented pixels into a cluttered image at a location and scale r picked at random, as depicted in Figures 3. More details are given in § 4.1.

By repeating this process several times for every frame of the training sequence we can build a series of positive samples. We can do the same without copying the object in the cluttered image to produce negative ones. We generate a total of N training samples

$$(\Psi(u_n, r_n, t_n),\, l_n) \in \widetilde{\mathbb{R}}^D \times \{0, 1\}, \quad n = 1, \ldots, N \tag{4}$$

where u_n is an image constructed as described above, r_n a location and scale, t_n the index of the frame used to produce u_n and l_n is the label in $\{0, 1\}$. Note that the parameter r_n is discretized consistently with the scene scanning procedure, and that the parameters r_n and t_n are unrelated to the image in the case of negative samples.

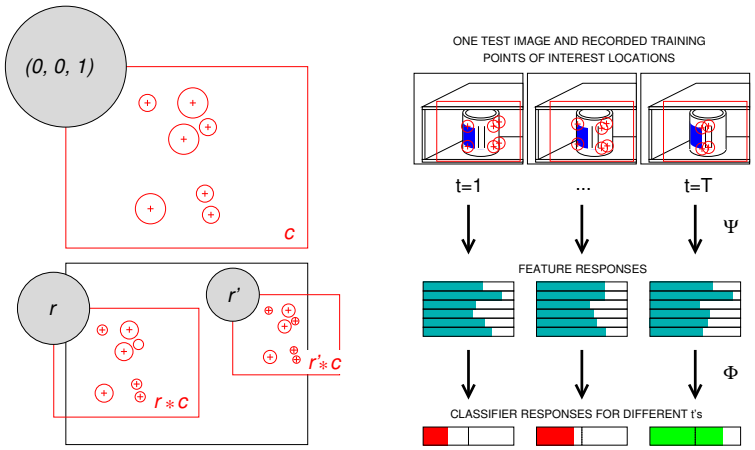

Fig. 2. We define the $*$ operator to move and scale configurations of local referentials (left figure). Given a configuration of local referentials $c \in \mathcal{C}$ and a reference frame $r = (x, y, s) \in \Theta$, then $r * c$ stands for the local referentials c translated and scaled according to r, given that the reference frame $(0, 0, 1)$ corresponds to no translation and no change of scale. The top-left image shows the local referentials c as they could be detected in a training image, whose frame is shown in red. The bottom-left picture shows this same local referentials as would be translated and scaled to two reference frames r and r' in a test image. The right figure illustrates the detection in a test image, for a given location and scale (x, y, s). The algorithm tests all Q configurations of c_t recorded in the training sequence and measures for each the corresponding feature vector $\Psi(u, r, t)$ in the test image. Given this feature vector, the classifier Φ produces a response, that should be positive when an object is truly at that location (x, y, s) with that pose t. See § 3.4 for details.

We then use a standard Adaboost procedure to pick the parameters of the classifier of Equation (3). At every Boosting step we sample several d_m uniformly and compute the optimal ρ_m, and chose the optimal ω_m for the chosen stump. See § 4.3 for experimental details.

4 Experiments

The main experimental result we present here is a reduction of the number of false positives when using the pose-indexed approach with respect to a standard sliding-window method. All the experiments were done on Linux computers, using the OpenCV library for the SURF key-point detection and descriptors [11], and the Boost serialization C++ library for object persistence.

We will distribute the C++ source code, and the data required to reproduce our experiment, under open licenses.

4.1 Data

We measure the performance of our method for the detection of three different objects in gray-scale images of resolution 800×600 pixels, shot with a Canon PowerShot A710-IS, in a cluttered environment, with variations in scale, rotations in and out of the image plane, and strong illumination changes. See Figure 4 for examples. We label every one of these test images with a square bounding box around the target.

The training videos were shot with the same camera in front of a uniform background, see Figure 4, top row. The resolution is 640×480 pixels at 15 frames per second, and we shot a total of between $1,000$ and $1,200$ frames for each object.

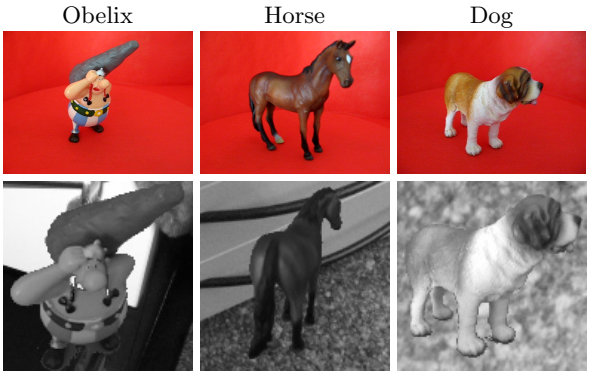

Fig. 3. The training videos (top) are shot in front of a uniform red background. The resolution is 640×480 pixels at 15 frames per second. We segment the frames to generate synthetic training images (bottom).

Each frame from those videos is processed independently to segment the object in it. Because the background is flat, we first run a Canny edge detector to estimate roughly the location of the object. This first estimate is used to learn two RGB color histograms corresponding respectively to the object and the background. We then threshold the posterior probability for every pixel to belong to the object, and keep the largest resulting connected component as the object's silhouette.

This segmentation is used to generate three positive images from each frame, for a total $\simeq 3,500$ positive samples, and we also generate $200,000$ negative images by simply cropping sub-images in cluttered scenes, as described in § 3.4.

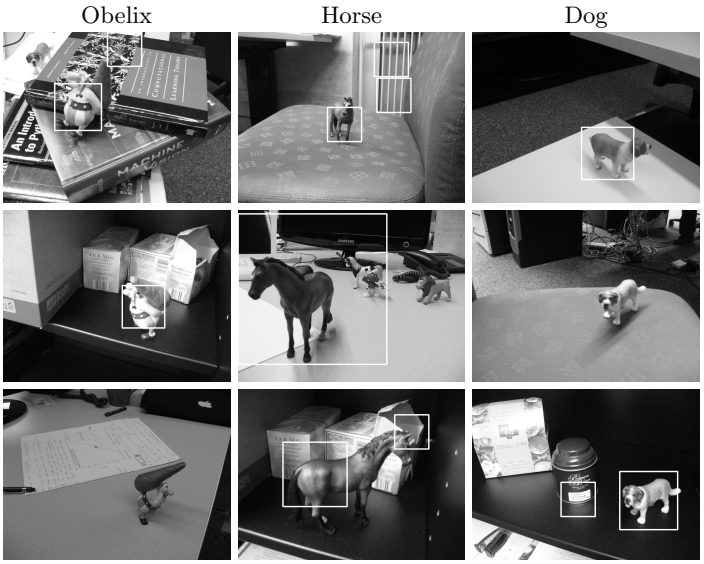

Fig. 4. We use gray-scale test images of resolution 800×600 pixels. Each column corresponds to one object, which is visible in every image. The environment is cluttered, with strong changes in pose and illumination of the target. This is a random selection of actual results in detection, after clustering alarms that overlap and keeping only the one with the highest response. A threshold is put to keep about 70% of true alarms.

4.2 Point of interest tracking

To record the configurations c_t of locations of points of interest, we first run the SURF key-point detector in each frame. Then, we go through the video and maintain a list of growing "chains" of points of interest.

Precisely, we start with one chain per point detected in the initial frame of the video. Then, in each frame, for every detected point of interest we consider all chains whose extremity is closer than a fixed threshold both spatially and in appearance according to the SURF descriptor. If several such chains exist, we take the one with the most similar extremity and add the point to it. If none exists, the point is used to start a new chain.

We keep only chains longer than 64 frames, which amount to $Q \simeq 400$ chains per object. Among the ones we keep, the median chain length was around 77 frames, 90% of the chains were longer than 65 frames, and 10% longer than 115.

4.3 Pose-indexed classifier

The total number of stumps combined in the classifier Φ is $M = 750$, and the optimization of the stump parameters d_m is done by looking at 400 candidate values at every Boosting step. So the dimension of Ψ is $D = 300,000$. These

parameter were not optimized for test performance but roughly chosen as a good trade-off between training time and loss reduction.

The sampling of the a_d parameters of the Ψ feature vector is done by taking the point indexes uniformly between 1 and Q, and the rectangle corners uniformly in an area that contains both the disc defined by the SURF location and scale, and an additional margin to account for discretization in the test scanning of the scene. Also, since we sub-sample the parameter t during test for computational reason, we increase this area accordingly.

We artificially add one local referential in all the training frames. Its location is forced at a fixed location and its scale is large enough to cover all the object. That allows the classifier to exploit static features. However they are sampled with probability $1/(Q+1)$ during training.

4.4 Baseline

The baseline is the same algorithm as the pose-indexed one described above, except that the features can only use the artificially added point. Hence, the value of $\Psi(u, r, t)$ does not depend anymore on t, and this set-up is a standard sliding-window approach. All parameters such as D and M and the scene scanning procedure remain identical.

4.5 Results

To compute true-positive and false-positive rates, we have to define a "hit criterion". Let $\mathcal{S}(x, y, s)$ denote the square of size s centered on (x, y) and let $||A||$ denote the surface of A. Then, we will say that the predicted alarm (x, y, s) is a correct hit for a target at (x', y', s') if

$$\frac{||\mathcal{S}(x,y,s)\,\Delta\,\mathcal{S}(x',y',s')||}{||\mathcal{S}(x,y,s)\,\cap\,\mathcal{S}(x',y',s')||} \leq 0.25,$$

where $A \Delta B$ is the symmetric difference $A \cup B \setminus A \cap B$.

Since we are interested in comparing the relative performance of our pose-indexed approach with respect to a standard sliding-window one, we do not post-process the detections. The error rates in Table 2 and Figure 5 correspond to the raw outputs of the classifiers.

The pose-indexed classifier performs substantially better at any true positive rate on two objects, and as well as the baseline on one of the objects. The selected pose-indexed features accumulate on informative parts of the objects, as shown on Figure 6. However, a weakness of the current implementation is that they will hardly be sampled in flat regions, due to the absence of points of interest in the vicinity.

After investigation, we noticed that the weights ω_m of the pose-indexed stumps are greater than those of the static ones (median value of 0.51 versus 0.08). This shows that pose-indexed features are very efficient on the population on which they are active.

Training the pose-indexed classifier takes about half the time of the static one (15min vs. 30min), because many features are *n.a.* and do not have to be computed. Since we use the static detector with a very conservative threshold as a pre-processing for the pose-indexed one, the detection takes only 3 to 5 times more time. This gap could be further reduced by using a coarse-to-fine search strategy.

5 Conclusion

We proposed a new machine learning approach to object detection: learning should be done in part in a "playground", a simplified situation where limited prior works well, so that a complex model can be constructed prior to a classical machine-learning procedure.

The complex multi-dimensional latent pose parameter can be broken into multiple components which can be estimated independently only if local cues are reliable, hence if the signal is of good quality. However such conditions are not met in the "real world". On uncontrolled test data, the only hope is to use a global appearance model able to exploit jointly many cues collected over the full object. Only in such case the law of large numbers can neutralize the remaining randomness, not captured in the model.

This approach is a strict generalization of classical "sliding window" detectors. Indeed, if the training video was showing the object at all locations and scales, the loop through t alone would result in a detector visiting locations and scales with a single classifier.

The possible extensions are numerous, from adapting the Boosting to this new context, to enriching the local poses of the points of interest with information such as skewing or illumination, or improving the modeling of the point of interest joint location distribution. The latter would allow to extrapolate from the limited set of poses appearing in the video to other viewpoints.

Table 2. False-positive error rate at several true-positive rates for our approach and a standard sliding-window baseline. This is the raw performance of the classifiers, without post-processing.

	Obelix		Horse		Dog	
	FP		FP		FP	
TP	baseline	pose-indexed	baseline	pose-indexed	baseline	pose-indexed
95%	0.94%	0.53%	1.29%	1.06%	0.83%	0.88%
90%	0.53%	0.31%	0.95%	0.76%	0.46%	0.41%
80%	0.17%	0.05%	0.48%	0.20%	0.15%	0.13%

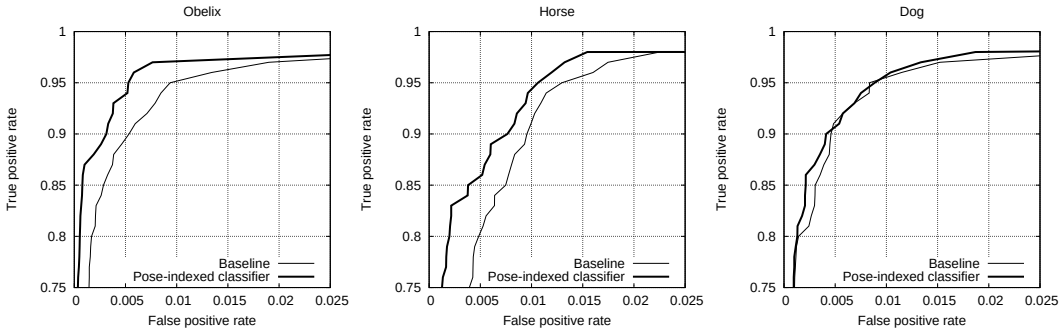

Fig. 5. These ROCs depict the performance of our approach compared to a standard sliding-window baseline. These results correspond to the raw performance of the classifiers, without post-processing the scene to remove isolated alarms and consolidate clusters.

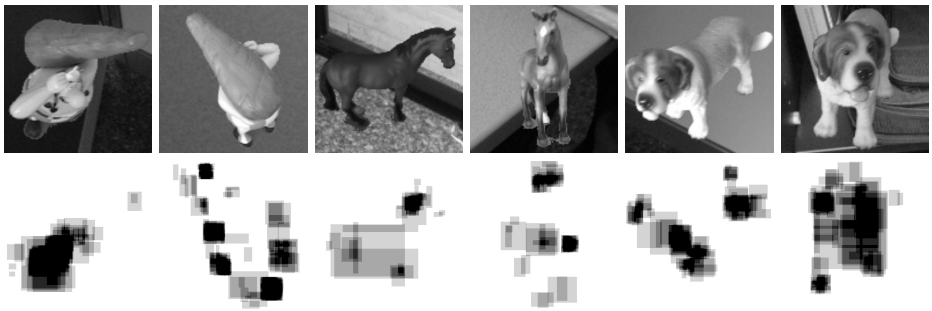

Fig. 6. This figure shows training images (top row) with a picture displaying the concentration of pose-indexed features under each of them for the corresponding t (bottom row). The shades of gray correspond to the number of rectangles at every point. As expected features accumulate on informative part of the object.

References

1. Viola, P., Jones, M.J.: Robust real-time face detection. International Journal of Computer Vision **57** (2004) 137–154
2. LeCun, Y., Huang, F., Bottou, L.: Learning methods for generic object recognition with invariance to pose and lighting. In: Conference on Computer Vision and Pattern Recognition, IEEE Press (2004)
3. Jones, M., Viola, P.: Fast multi-view face detection. Technical Report TR2003-96, Mitsubishi Electric Research Laboratories (2003)
4. Fleuret, F., Geman, D.: Stationary features and cat detection. Journal of Machine Learning Research (JMLR) **9** (2008) 2549–2578
5. Lowe, D.G.: Object recognition from local scale-invariant features. In: International Conference on Computer Vision. (1999) 1150–1157

6. Li, F., Fergus, R., Perona, P.: A Bayesian approach to unsupervised one-shot learning of object categories. In: International Conference on Computer Vision. Volume 2. (2003) 1134
7. Schneiderman, H., Kanade, T.: Object detection using the statistics of parts. International Journal of Computer Vision **56** (2004) 151–177
8. Leibe, B., Seemann, E., Schiele, B.: Pedestrian detection in crowded scenes. In: Conference on Computer Vision and Pattern Recognition. Volume 1. (2005) 878–885
9. Lampert, C.H., Blaschko, M.B., Hofmann, T.: Beyond sliding windows: Object localization by efficient subwindow search. In: Conference on Computer Vision and Pattern Recognition. (2008) 1–8
10. Amit, Y., Trouvé, A.: POP: Patchwork of parts models for object recognition. International Journal of Computer Vision **75** (2007) 267–282
11. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: SURF: Speeded up robust features. Computer Vision and Image Understanding **10** (2008) 346–359