Learning-Based Object Tracking Using Boosted Features and Appearance-Adaptive Models

Bogdan Kwolek

Rzeszów University of Technology,
W. Pola 2, 35-959 Rzeszów, Poland
bkwolek@prz.rzeszow.pl

Abstract. This paper presents a learning-based algorithm for object tracking. During on-line learning we employ most informative and hard to classify examples, features maximizing individually the mutual information, stable object features within all past observations and features from the initial object template. The object undergoing tracking is discriminated by a boosted classifier built on regression stumps. We seek mode in the confidence map calculated by the strong classifier to sample new features. In a supplementing tracker based upon a particle filter we use a recursively updated mixture appearance model, which depicts stable structures in images seen so far, initial object appearance as well as two-frame variations. The update of slowly varying component is done using only pixels that are classified by the strong classifier as belonging to foreground. The estimates calculated by particle filter allow us to sample supplementary features for learning of the classifier. The performance of the algorithm is demonstrated on freely available test sequences. The resulting algorithm runs in real-time.

1 Introduction

Object tracking is a central theme in computer vision and has received considerable attention in the past two decades. The goal of tracking is to automatically find the same object in adjacent frames in a video sequence. To achieve a better quality of tracking many algorithms consider environment and utilize pixels from background [1][2][3]. To cope with changes of observable appearance many of them incrementally accommodate models to the changes of object or environment [4][5]. In such systems, Gaussian mixture models can be used to represent both foreground [6] and background [7].

Detecting and tracking of objects using their appearances play an important role in many applications such as vision based surveillance and human computer interaction [5][8][6]. A learning algorithm can improve the robustness if the observed appearance of a tracked object undergoes complex changes. A learning takes place in recently proposed algorithms built on classification methods such as support vector machines [1] or AdaBoost [2][3].

Obtaining a collection consisting of both positive and negative examples for on-line learning is complex task. The algorithm [9] starts with a small collection of manually labeled data and then generates supplementary examples by

applying co-training of two classifiers. To avoid hand labeling the use of motion detection in order to obtain the initial training set was proposed in [10].

In our approach, Gentle AdaBoost built on regression stumps combines several classifiers into an accurate one. An algorithm constructs on the fly a training set consisting of promising object and background features. It consists of representative object features from the initial template, the most stable object features seen so far, uniformly subsampled background features without repetition and features maximizing individually the mutual information. Such family of features can be poorly informative and therefore the set also consists of hard to classify examples that provide most new information during object tracking.

An on-line method using boosted features and adaptive appearance models is key contribution of this paper to learning based object tracking. This work's novelty consists in managing several kinds of features, namely describing stable object structures, characterizing two-frame variations and characteristic samples from the initial template to support a data-driven learning of weak classifiers within computationally feasible procedure based on Gentle AdaBoost. We also demonstrate how adaptive appearance models can be integrated with boosted features to improve the performance of tracking. The resulting algorithm considers the temporal coherence between images of object undergoing tracking.

The rest of the paper is organized as follows. In the next Section below we refer to learning in object tracking. In Section 3 we discus how regression stumps are utilized in Gentle AdaBoost. The components and details of learning based object tracking using boosted features are discussed in Section 4. The usage of adaptive appearance models in a particle filter is explained in Section 5. We demonstrate also how adaptive appearance models can be integrated with boosted features to improve the performance of tracking. We report and discuss experimental results in Section 6. We draw conclusions in the last Section.

2 Learning in object tracking

When learned off-line classifiers are employed the tracking can be realized trough detection of the target. Okuma *et al.* [11] propose an approach that uses a boosted detector operating on color distributions to construct a proposal distribution for the particle filter. Considering tracking as binary classification, Avidan [1] proposes a support vector based tracker built on the polynomial kernel. In such tracker with learning capabilities the score of support vector machine is maximized for every frame. A system built on the relevance vector machine which employs temporal fusion is described in work of Williams *et al.* [12].

In work [2] AdaBoost is used in algorithm termed as ensemble tracking to learn the classifier. The appearance model is updated by adding recent features. An approach presented in [13] employs image pairs and temporal dependencies into a learned similarity function instead of learning a classifier to differentiate the object from the background.

Some work has been done in the past to enable automatic labeling of training data. Robust automatic labeling is a highly desirable property in any learning

based tracking system. Levin *et al.* [9] propose the so called co-training approach which consists in starting with a small training set and increasing it by co-training of two classifiers, operating on different features. Nair and Clark [10] use the motion detection for constructing the initial training set and then the Winnow as a final classifier.

Ensemble methods such as boosting and bagging have demonstrated significant advantages in off-line settings. However little work has been done in exploring these methods in on-line settings. In [14], Oza and Russel propose on-line version of boosting which simulates the bootstrap process through updating each base model using multiple copies of each new example. The algorithm that is proposed in work [2] maintains a list of classifiers that are trained over time. During tracking it removes old classifiers, trains new classifiers using a confidence map generated by the strong classifier and then adds them to the ensemble. However, through removing the oldest classifiers this algorithm omits important information contained in the initial object template [15] as well it is not able to detect features being stable during tracking. The importance of such stable features during tracking has been highlighted by several authors, among others by [6]. In an algorithm described in [3] the selectors are updated when a new training sample is available. This operation needs considerable computations since the strong classifier contains 50 selectors and each can choose from 250 selectors. This in turn can even lead to slower boosting algorithm in comparison with an off-line algorithm applied to learn on-line. The average number of calculations per feature in this algorithm can be far larger than in off-line AdaBoost.

3 Boosting

Boosting originates from a machine learning model known as Probably Approximately Correct (PAC). Boosting algorithms combine simple decision rules into more complex ones. They aim at finding an accurate classifier consisting of many base classifiers, which are only moderately accurate. The boosting algorithm executes the base learning algorithm multiple times to achieve the desired classification performance. During iterations the weights are updated dynamically according to the errors in previous round of learning. The base learning algorithm takes into account a weight coupled with each training instance and attempts to find a learned hypothesis that minimizes the weighted classification error. The learning algorithm generates classification rules that are combined by the boosting algorithm into the final classification rule. In the first step a boosting algorithm constructs an initial distribution of weights over the training set. The weights are greater than zero, sum to one and constitute a distribution over the training set. Using the weighted training set the algorithm searches for a classification rule consisting in a selecting a base classifier that gives the least weighted error. The weights of the data that are misclassified by the selected base classifier are increased. This leads to selection of classifier that performs better on examples misclassified previously. Each weak classifier predicts the label of the data. In consequence, AdaBoost [16], which is the adaptive version of boosting

minimizes the following exponential loss function:

$$J(F) = E(e^{-yF(x)}), \tag{1}$$

where E denotes the expectation and the strong classifier $F(x)$ is a linear combination of T weak classifiers $f_i(x)$:

$$F(x) = \sum_{i=1}^{T} \alpha_i f_i(x), \tag{2}$$

with parameters α_i to balance the evidence from each feature. The set of decision rules $\{f_i\}_{i=1}^{T}$ and combining coefficients $\{\alpha_i\}_{i=1}^{T}$ are learned.

3.1 Gentle AdaBoost

We employ in our tracking algorithm a version of boosting called Gentle AdaBoost [17], because it requires fewer iterations to achieve similar classification performance in comparison with other methods. Given a set of training instances $\mathcal{X}$ and a corresponding weight distribution D the boosting algorithm calculates a weak hypothesis $f : \mathcal{X} \mapsto R$, where the sign of f determines the predicted label y of the instance $x \in \mathcal{X}$. The magnitude $|f(x)|$ expresses the confidence of the prediction. Suppose we have a current ensemble hypothesis $F(x) = \sum_{t=1}^{T} f_t(x)$ and seek better one $F + f$ by minimizing the following criterion:

$$J(F + f) = E[e^{-y[F(x)+f(x)]}], \tag{3}$$

where E denotes the expectation. Gentle AdaBoost minimizes this equation by employing adaptive Newton steps [17], which corresponds to minimizing at each step a weighted squared error. At each step m the current ensemble hypothesis F is updated as follows $F(x) \leftarrow F(x) + f_m$, where f_m is selected to minimize a second order Taylor approximation of the cost function. Replacing the weighted conditional expectation $E[y|x]$ in (3) with an empirical expectation over the training data leads to minimizing the weighted squared error:

$$J = \sum_{i=1}^{L} w_i (y_i - f_m(x_i))^2, \tag{4}$$

where $w_i = e^{-y_i F(x_i)}$ and the summation is over the training exemplars.

3.2 Regression stumps based weak learner

As weak learners we employ regression stumps of the following form:

$$f_m(x) = a\delta(x^{(k)} > \theta) + b \tag{5}$$

where $x^{(k)}$ denotes the k-th coordinate of K dimensional feature vector x, δ is the Kronecker delta function, θ is a threshold, and a, b are regression parameters.

Such binary regression stumps were employed in [18][19]. To minimize function (4) we should determine in each iteration m four parameters of the regression stump (5), namely a, b, θ and k. First, we calculate parameters a and b with respect to each possible threshold $\theta_i^{(k)} = x_i^{(k)}$, i.e. for $i = 1, 2, ..., L$ and $k = 1, 2, ..., K$:

$$b_i^{(k)} = \frac{\sum_{j=1}^{L} w_j y_j \delta(x_j^{(k)} \leq x_i^{(k)})}{\sum_{j=1}^{L} w_j \delta(x_j^{(k)} \leq x_i^{(k)})} \qquad a_i^{(k)} = \frac{\sum_{j=1}^{L} w_j y_j \delta(x_j^{(k)} > x_i^{(k)})}{\sum_{j=1}^{L} w_j \delta(x_j^{(k)} > x_i^{(k)})} - b_i^{(k)}. \quad (6)$$

Then, we determine error according to the following formula:

$$e_i^{(k)} = \sum_{j=1}^{L} w_j (y_j - a_i^{(k)} \delta(x_j^{(k)} > x_i^{(k)}) + b_i^{(k)})^2. \quad (7)$$

Next, for each dimension k we seek for thresholds $\theta^{(k)} = x_{\check{i}^{(k)}}^{(k)}$, which minimize the error function given by (7). This can be expressed in the following manner:

$$\check{i}^{(k)} = \arg \max_{i=1,2,...,L} \{e_i^{(k)}\}. \quad (8)$$

In the final step of selecting the best regression stump we determine the coordinate $\check{k}$ for which the error function (7) takes minimal value:

$$\check{k} = \arg \max_{k=1,2,...,K} \{e_{\check{i}^{(k)}}^{(k)}\}. \quad (9)$$

To speed up the selecting θ the computations were conducted using K sorted vectors x. In order to decrease the number of summations during fitting the regression stumps we utilized the cumulative sums of w_j and $w_j y_j$.

4 Learning-based object tracking using boosted features

The most informative and hard to classify examples are in vicinity of the decision boundary between background and target. In our approach, an on-line AdaBoost focuses on such hard examples that provide more new information than easy ones. Such examples cause the base learner to concentrate on unseen examples. The updated on-line training set consists of also most stable object features seen so far, uniformly subsampled background features without repetition and features maximizing individually the mutual information. In this context, the major difference of our work from relevant research is that weak classifiers are not trained from the same data sets, which are acquired within rectangles covering the object and the surrounding background, but only a small portion of the newly available training sets. It is major difference between our learning based tracking algorithm and algorithms relying on linear adaptation or learning, where the update of the object model is done via all newly extracted pixels.

An on-line learning algorithm does not need all the training data processed so far to calculate a current hypothesis, rather it process data as it become available

without the need for storage, through reusing previously learned weak classifier to learn new classifier. In our approach we initially train the classifier on pixels that were labeled in advance and then apply the classifier in each frame to extract the object of interest. An unsupervised learning is done using labeled pixels by the classifier, pixels depicting initial object appearance as well as stable object structures within all past observations. The object and background pixels are extracted using center-surround approach in which an internal rectangle covers the object, while a larger surrounding rectangle represents the background. The weak learner that was described in subsection 3.2 is used in on-line training.

Before starting of the tracking the foreground and background pixels are extracted using center-surround approach. The initial object template is constructed on the basis of the internal rectangle covering the object of interest. A number of representative pixels that are sampled from the object of interest are then utilized during tracking. Such pixel collection holds information about initial object appearance and prevents from model drift. A strong classifier is used to label the pixels as either belonging to the object of interest or background. On the basis of the distribution indicated by weights we sample from the current frame a set of foreground pixels that are hardest to classify. Using a histogram holding information about colors of all pixels seen so far in the object rectangle we extract in each frame a set of the most stable pixels and add it to the set representing the current frame. Through such stable pixels the algorithm considers the temporal coherence between images of object undergoing tracking. The background is represented by pixels laying in close to decision boundary as well as collection of uniformly sampled pixels both from the current and previous frame. In order to avoid the weakness of the random sampling we additionally pick features maximizing individually the mutual information to forecast the class. Given N_s samples with the M binary features $X_1, ..., X_M$, and the target classification variable Y, our goal is to select G features $X_{v(1)}, ..., X_{v(G)}$, which accurately characterize Y. The selected features individually maximize the mutual information $I(Y; X_{v(l)}) = H(Y) - H(Y|X_{v(l)})$, where $H()$ is the entropy. During tracking a simple procedure is responsible for removing the pixels belonging to previous frame and inserting the pixels from the new frame as well as maintaining proportions between the mentioned above ingredients of the training vector at possibly the same level. The length of the list containing training pixels is constant.

During boosting iterations the weights that are employed by weak learner are calculated as follows:

$$w \leftarrow w \exp(-y\, f_m) \tag{10}$$

The total score produced by AdaBoost is normalized through soft identity function to range between -1 and 1 in the following manner:

$$s = \tanh(F(x)) = \tanh(\sum_{m=1}^{T} f_m(x)) \tag{11}$$

Such a normalized score can be used as a measure of prediction confidence [20]. The face location during tracking is computed by CamShift [21] acting on the

likelihood images. Since our tracking algorithm should spend small number of CPU cycles, we use similar color cues to those employed in original implementation of CamShift, i.e. RG or HS color components.

5 Adaptive models for particle filtering

Low-order parametric models of the image motion of pixels laying within a template can be utilized to predict the movement in the image plane [22]. This means that by comparing the gray level values of the corresponding pixels within region undergoing tracking, it is possible to obtain the transformation (giving shear, dilation and rotation) and translation of the template in the current image [23]. Therefore, such models allow us to establish temporal correspondences of the target region. They make region-based tracking an effective complement to tracking that is based on classifier distinguishing between foreground and background pixels. In a particle filter the usage of change in transformation and translation $\Delta\omega_{t+1}$ arising from changes in image intensities within the template can lead to reduction of the extent of noise ν_{t+1} in the motion model. It can take the form [6]: $\omega_{t+1} = \hat{\omega}_t + \Delta\omega_{t+1} + \nu_{t+1}$.

5.1 Adaptive velocity model

Let $I_{x,t}$ denote the brightness value at the location (x_1, x_2) in an image $\mathcal{I}$ that was acquired in time t. Let $\mathcal{R}$ be a set of J image locations $\{x^{(j)} \mid j = 1, 2, ..., J\}$ defining a template. $Y_t(\mathcal{R}) = \{I_{x,t}^{(j)} \mid j = 1, 2, ..., J\}$ is a vector of the brightness values at locations $x^{(j)}$ in the template. We assume that the transformations of the template can be modeled by a parametric motion model $g(x; \omega_t)$, where x denotes an image location and $\omega_t = \{\omega_t^{(1)}, \omega_t^{(2)}, ..., \omega_t^{(l)}\}$ denotes a set of l parameters. The image variations of planar objects that undergo orthographic projection can be described by a six-parameter affine motion models [22]:

$$g(x; \omega) = \begin{bmatrix} a & d \\ c & e \end{bmatrix} x + \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = Ax + u, \tag{12}$$

where $\omega = (a, c, d, e, u_1, u_2)^T$. With these assumptions, the tracking of the object in time t can be achieved by computing ω_{t+1} such that $Y_{t+1}(g(\mathcal{R}; \omega_{t+1})) = \hat{Y}_t(\mathcal{R})$, where the template $\hat{Y}_t(\mathcal{R})$ is in pose determined by the estimated state.

Given a set $S = \{\omega_t^{(n)}, \pi_t^{(n)}) \mid n = 1, ..., N\}$ of weighted particles, which approximate the posterior distribution $p(\omega_t \mid Y_{1:t})$, the maximum aposteriori estimate (MAP) of the state is calculated according to the following formula:

$$\hat{\omega}_t = \arg\max_{\omega_t} p\left(\omega_t \mid Y_{1:t}\right) \approx \arg\max_{\omega_t} \pi_t^{(n)} \tag{13}$$

The motion parameters in time $t + 1$ take values according to:

$$\omega_{t+1} = \hat{\omega}_t + A_{t+1}[\hat{Y}_t(\mathcal{R}) - Y_{t+1}(g(\mathcal{R}; \hat{\omega}_t))]. \tag{14}$$

This equation can be expressed as follows: $\Delta\omega_{t+1} = A_{t+1}\Delta y_{t+1}$. Given N measurements we can estimate matrix A_{t+1} from matrices consisting of adjoined vectors $\Delta\omega_{t+1}$ and Δy_{t+1} [23]:

$$\Delta \mathrm{M}_t = [\hat{\omega}_t^{(1)} - \omega_t^{(1)}, ..., \hat{\omega}_t^{(N)} - \omega_t^{(N)}] \tag{15}$$

$$\Delta \mathrm{Y}_t = [\hat{Y}_t^{(1)} - Y_t^{(1)}, ..., \hat{Y}_t^{(N)} - Y_t^{(N)}]. \tag{16}$$

Using the least squares (LS) method we can find the solution for A_{t+1} [23]:

$$A_{t+1} = (\Delta \mathrm{M}_t \Delta \mathrm{Y}_t^T)(\Delta \mathrm{Y}_t \Delta \mathrm{Y}_t^T)^{-1}. \tag{17}$$

Singular value decomposition of $\Delta \mathrm{Y}_t$ yields: $\Delta \mathrm{Y}_t = UWV^T$. Taking q largest diagonal elements of W the solution for A_{t+1} is as follows: $A_{t+1} = \Delta \mathrm{M}_t V_q W_q^{-1} U_q^T$. The value of q depends on the number of diagonal elements of W, which are below a predefined threshold value.

In the particle filter [24] we utilize the following motion model:

$$\omega_{t+1} = \hat{\omega}_t + \Delta\omega_{t+1} + \nu_{t+1}, \tag{18}$$

where ν_{t+1} is zero mean Gaussian i.i.d. noise, independent of state and with covariance matrix Q which specifies the extent of noise.

When individual measurements carry more or less weight, the individual rows of $\Delta\omega = A\Delta y$ can be multiplied by a diagonal matrix with weighting factors. If the diagonal matrix is the identity matrix we obtain the original solution. In our approach such row weighting is used to emphasize or de-emphasize image patches according to number of background pixels they contain.

5.2 Appearance modeling using adaptive models

Our intensity-based appearance model consists of three components, namely, the W-component expressing the two-frame variations, the S-component characterizing the stable structure within all previous observations and F component representing a fixed initial template. The model $A_t = \{W_t, S_t, F_t\}$ represents thus the appearances existing in all observations up to time $t-1$. It is a mixture of Gaussians [5] with centers $\{\mu_{i,t} \,|\, i = w, s, f\}$, their corresponding variances $\{\sigma_{i,t}^2 \,|\, i = w, s, f\}$ and mixing probabilities $\{m_{i,t} \,|\, i = w, s, f\}$.

The update of the current appearance model A_t to A_{t+1} is done using the Expectation Maximization (EM) algorithm. For a template $\hat{Y}(\mathcal{R}, t)$ corresponding to the estimated state we evaluate the posterior contribution probabilities as follows:

$$o_{i,t}^{(j)} = \frac{m_{i,t}^{(j)}}{\sqrt{2\pi\sigma_{i,t}^2}} \exp\left[-\frac{\hat{I}_{x,t}^{(j)} - \mu_{i,t}^{(j)}}{2\sigma_{i,t}^2}\right] \tag{19}$$

where $i = w, s, f$ and $j = 1, 2, ..., J$. If the considered pixel belongs to background, the posterior contribution probabilities are calculated using $\hat{I}_{x,1}^{(j)}$:

$$o_{i,t}^{(j)} = \frac{m_{i,t}^{(j)}}{\sqrt{2\pi\sigma_{i,t}^2}} \exp\left[-\frac{\hat{I}_{x,1}^{(j)} - \mu_{i,t}^{(j)}}{2\sigma_{i,t}^2}\right]. \tag{20}$$

This prevents the slowly varying component from updating by background pixels. The posterior contribution probabilities (with $\sum_i o_{i,t}^{(j)} = 1$) are utilized in updating the mixing probabilities in the following manner:

$$m_{i,t+1}^{(j)} = \gamma o_{i,t}^{(j)} + (1-\gamma) m_{i,t}^{(j)} \quad | \; i = w, s, f, \tag{21}$$

where γ is accommodation factor. Then, the first and the second-moment images are determined as follows:

$$\begin{aligned} M_{1,t+1}^{(j)} &= (1-\gamma) M_{1,t}^{(j)} + \gamma o_{s,t}^{(j)} \hat{I}_{x,t}^{(j)} \\ M_{2,t+1}^{(j)} &= (1-\gamma) M_{2,t}^{(j)} + \gamma o_{s,t}^{(j)} (\hat{I}_{x,t}^{(j)})^2. \end{aligned} \tag{22}$$

In the last step the mixture centers and the variances are calculated as follows:

$$\begin{aligned} &\mu_{s,t+1}^{(j)} = \tfrac{M_{1,t+1}^{(j)}}{m_{s,t+1}^{(j)}}, \;\; \sigma_{s,t+1}^{(j)} = \sqrt{\tfrac{M_{2,t+1}^{(j)}}{m_{s,t+1}^{(j)}} - (\mu_{s,t+1}^{(j)})^2} \\ &\mu_{w,t+1}^{(j)} = \hat{I}_{x,t}^{(j)}, \quad \sigma_{w,t+1}^{(j)} = \sigma_{w,1}^{(j)} \\ &\mu_{f,t+1}^{(j)} = \mu_{t,1}^{(j)}, \quad \sigma_{f,t+1}^{(j)} = \sigma_{f,1}^{(j)}. \end{aligned} \tag{23}$$

When the considered pixel belongs to background, the mixture center in the component expressing two-frame variations is updated according to:

$$\mu_{w,t+1}^{(j)} = \hat{I}_{x,l}^{(j)}, \tag{24}$$

where index l refers to last non-background pixel.

In order to initialize the model A_1 the initial moment images are set using the following formulas: $M_{1,1} = m_{s,1} I(\mathcal{R}, t_0)$ and $M_{2,1} = m_{s,1}(\sigma_{s,1}^2 + I(\mathcal{R}, t_0)^2)$. The observation likelihood is calculated according to the following equation:

$$p(Y_t \,|\, \omega_t) = \prod_{j=1}^{J} \sum_{i=w,s,f} \frac{m_{i,t}^{(j)}}{\sqrt{2\pi\sigma_{i,t}^2}} \exp\left[-\frac{I_{x,t}^{(j)} - \mu_{i,t}^{(j)}}{2\sigma_{i,t}^2} \right] \tag{25}$$

Underlying AdaBoost-based tracking algorithms do not take into account of temporal information (except [13]) as they rely on learned binary classifiers that discriminate the target and the background. In our algorithm the data-driven binary classifier learns on-line using features from the initial object template, stable object features within all past observations, features maximizing individually the mutual information, most informative and hard to classify examples, and the features that are sampled from the object rectangle estimated by particle filter. In the particle filter we use a recursively updated mixture appearance model, which depicts stable structures in images seen so far, initial object appearance as well as two-frame variations. The update of slowly varying component is done using only pixels that are classified by the strong classifier as belonging to foreground. In pairwise comparison of object images we employ only non-background pixels and in case of background we use the last foreground pixels. Our probabilistic models differ from those proposed in [6] in that we adapt models using information about background. The outcome of the strong classifier is used to construct a Gaussian proposal distribution, which guides particles towards most likely locations of the object of interest.

6 Experiments

The tests were done on a sequence[1] of images 288 high and 384 pixels wide. In this sequence a tracked pedestrian crosses zones in varying illumination conditions. In tracking experiments with this sequence and a particle filter built only on adaptive appearance models and configured to run with 100 particles, some pixels of the object rectangle are updated by background pixels (for example in frames #1000 and #1200). Despite this undesirable effect the object model can adapt to pedestrian's side view. However, the update of the model by background pixels leads to considerable jitter of ROI and in consequence the track is lost in frame #1226.

In a comparison of the results generated by our on-line learning-based algorithm and an adaptive algorithm, where all pixels laying inside the object rectangle are utilized in an linear adaptation of the model, we observed that our algorithm performs significantly better. In particular, we compared the probability images, which illustrate the potential of algorithms in extraction of the target. The confidence maps generated by the learning-based algorithm picks better the person's shape over time. In frames that were generated by learning-based algorithm the jitter of rectangular ROI is smaller and it is located near the true location of the target in most frames. Despite similar distribution of background color with the foreground color, the number of background pixels with high confidence in the rectangle surrounding the object is relatively small. The mentioned effect has been achieved using only ten rounds of boosting in on-line learning.

Figure 1 shows the behavior of learning-based tracker using boosted features and appearance-adaptive models. It has been initialized and configured in the same manner as the algorithm based on adaptive appearance models. Because the appearance models are updated using only object pixels, the algorithm performs far better than algorithm built on only adaptive appearance models, especially in case of rotations of the pedestrian. The estimates calculated by particle filter were employed to sample additional features for learning of the classifier. Generally speaking, the 2-frame affine tracker can be expected to posses problems with targets that are nor deforming in a roughly affine manner, as well as with small objects. In such a situation the learning based algorithm can support the tracking. The algorithms have different failure modes and complement each other during tracking.

Our algorithm is about 2.2 times slower than the algorithm built on adaptive appearance models. It was implemented in C/C++ and runs with 320×240 images at about 10 fps on 2.4 GHz Pentium IV. It can be easily extended to run with other features, for example integral images or orientation histograms. A modification consisting in a replace of the CamShift by a particle filter operating on the confidence maps is also straightforward.

[1] Downloaded from site at: http://groups.inf.ed.ac.uk/vision/CAVIAR/

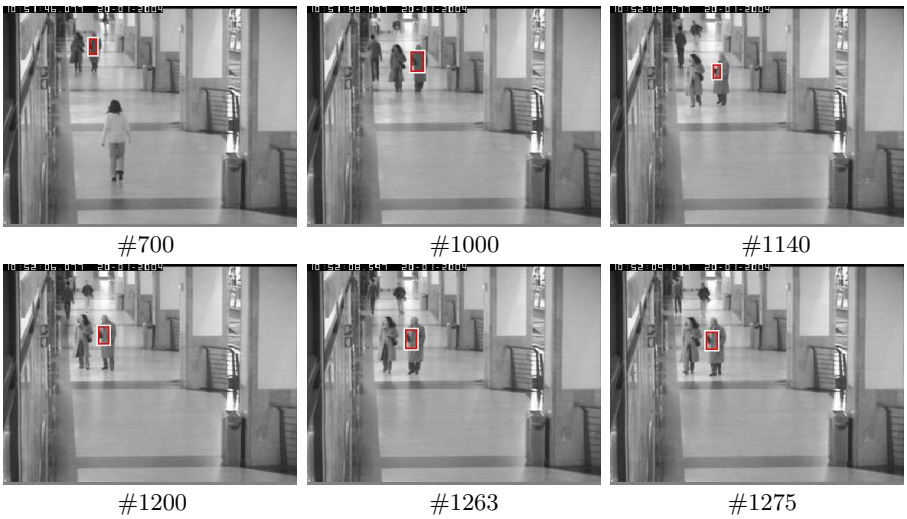

Fig. 1. Pedestrian tracking using learning and adaptive appearance models

.

7 Conclusions

We have presented an approach for on-line learning during tracking. The major difference of our work from relevant research is that weak classifiers are not trained from the same data but only a portion of newly available pixels. During learning we employ stable object features seen so far, features maximizing individually the mutual information, examples that are in vicinity of the decision boundary between background and target, and uniformly subsampled background features. To avoid drift the on-line training is conducted using pixels of the object template. In a supplementing tracker based on a particle filter we use a recursively updated mixture appearance model, which depicts stable structures in images seen so far, initial object appearance as well as two-frame variations. We accommodate the slowly varying component using only pixels that are classified by the strong classifier as belonging to object. The estimates calculated by particle filter are employed to sample learning features. The two algorithms have different failure modes and complement each other during tracking.

Acknowledgment

This work has been supported by Polish Ministry of Education and Science (MNSzW) within the projects 3 T11C 057 30 and N206 019 31/2664.

References

1. Avidan, S.: Support vector tracking. In: Int. Conf. on Comp. Vision and Pattern Rec., Hawaii (2001) 184–191

2. Avidan, S.: Ensemble tracking. In: Int. Conf. on Comp. Vision and Pattern Rec. (2005) 494–501, vol. 2
3. Grabner, H., Grabner, M., Bischof, H.: On-line boosting and vision. In: Int. Conf. on Comp. Vision and Pattern Rec. (2006) 260–267, vol. 1
4. Han, B., Comaniciu, D., Zhu, Y., Davis, L.: Incremental density approximation and kernel-based bayesian filtering for object tracking. In: Int. Conf. on Comp. Vision and Pattern Rec., Washington, DC (2004) 638–644
5. Jepson, A.D., Fleet, D.J., El-Maraghi, T.: Robust on-line appearance models for visual tracking. In: Int. Conf. on Comp. Vision and Pattern Rec. (2001) 415–422
6. Zhou, S.K., Chellappa, R., Moghaddam, B.: Appearance tracking using adaptive models in a particle filter. In: Proc. Asian Conf. on Comp. Vision. (2004)
7. Grimson, W.E.L., Stauffer, C.: Adaptive background mixture models for real-time tracking. In: IEEE Int. Conf. on Comp. Vision and Pattern Rec. (1999) 22–29
8. Viola, P., Jones, M., Snow, D.: Detecting pedestrians using patterns of motion and appearance. In: Proc. Int. Conf. on Comp. Vision. (2003) 734–741, vol. 2
9. Levin, A., Viola, P., Freund, Y.: Unsupervised improvement of visual detectors using co-training. In: Proc. Int. Conf. on Comp. Vision. (2004) 626–633, vol. 1
10. Nair, V., Clark, J.J.: An unsupervised, online learning framework for moving object detection. In: Int. Conf. on Comp. Vision and Pattern Rec. (2004) 317–324, vol. 2
11. Okuma, K., Teleghani, A., Freitas, N.D., Little, J., Lowe, D.G.: A boosted particle filter: Multitarget detection and tracking. In: Proc. 8th European Conf. on Comp. Vision, Prague, Czech Republic (2004) 29–39
12. Williams, O., Blake, A., Cipolla, R.: A sparse probabilistic learning algorithm for real-time tracking. In: Int. Conf. on Comp. Vision, Nice, France (2003) 353–360
13. Zhou, S.K., Shao, J., Georgescu, B., Comaniciu, D.: Boostmotion: Boosting a discriminative similarity function for motion estimation. In: Proc. of Int. Conf. on Comp. Vision and Pattern Rec., New York (2006) 1761–1768, vol. 2
14. Oza, N.C., Russell, S.: Online bagging and boosting. In: 8th Int. Workshop on Artificial Intelligence and Statistics, Morgan Kauffman (2001) 105–112
15. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. IEEE Trans. on Pattern Analysis and Machine Intelligence **26** (2004) 810–815
16. Freund, Y., Schapire, R.: Experiments with a new boosting algorithm. In: Proc. of Int. Conf. on Machine Learning, San Francisco, Morgan Kauffman (1996) 148–156
17. Friedman, J., Hastie, T., Tibshirani, R.: Additive logistic regression: a statistical view of boosting. Annals of Statistics **38** (2000) 337–374
18. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Int. Conf. on Comp. Vision and Pattern Rec. (2001) 511–518, vol. 1
19. Torralba, A., Murphy, K., Freeman, W.: Sharing features: efficient boosting procedures for multiclass object detection. In: Int. Conf. on Comp. Vision and Pattern Rec. (2004) 762–769, vol. 2
20. Schapiere, R., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: A new explanation for the effectiveness of voting methods. The Annals of Statistics **26** (1998) 1651–1686
21. Bradski, G.R.: Computer vision face tracking as a component of a perceptual user interface. In: Proc. IEEE Workshop on Appl. of Comp. Vision. (1998) 214–219
22. Hager, G.D., Belhumeur, P.N.: Efficient region tracking with parametric models of geometry and illumination. IEEE Trans. on Pattern Analysis and Machine Intelligence **20** (1998) 1025–1039
23. Horn, B.K.P.: Robot Vision. The MIT Press (1986)
24. Isard, M., Blake, A.: Condensation – conditional density propagation for visual tracking. Int. J. of Computer Vision **29** (1998) 5–28