3

Bayesian Tracking by Online Co-Training and Sequential Evolutionary Importance Resampling

Lizuo Jin, Zhiguo Bian, Qinhan Xu and Zhengang Chen
School of Automation, Southeast University
China

1. Introduction

Object tracking is an indispensable ingredient of many machine vision applications such as intelligent visual surveillance, traffic monitoring, robot and vehicle navigation, human computer interactions, virtual and augmented realities, video compression and indexing, etc. and has drawn considerable attention from computer research communities in recent years. Actually, in the real world scenarios, that is a very challenging task due to the interference of noise, clutters, occlusions, illumination variations and dynamic changes of the object and the background appearance in the complex scene; a quite variety of tracking methods have been proposed to tackle these difficulties in decades (Yilmaz et al., 2006), which can be roughly divided into two categories: the deterministic method and the statistical methods.

The deterministic method performs tracking typically by seeking the local extreme of a matching function which measures the similarity between a template and a candidate image; the most widely used similarity measures include the sum of squared differences, the histogram intersection distance, the Kullback-Leibler divergence, the normalized cross correlation coefficient, and the Bhattacharyya coefficient. Some optimization techniques have been proposed to search the local extreme of the matching function such as the mean-shift method (Comaniciu et al., 2003) and the optical flow based method (Baker & Matthews, 2004). The drawback of these methods is if the matching function takes into account only the object and not the background, then it might not be able to correctly distinguish the object from the background and tracking might fail. More robust similarity measures are presented recently such as the posterior probability measure (Feng et al., 2008.) and the log likelihood ratio of features (Collins et al., 2005), which takes the background interference into account.

Recently, object tracking is treated as a binary classification problem, where the object have to be identified from the background with multiple image cues and better performance over the matching function based approaches such as the template-matching method (Lucas & Kanade, 1981), the view-based method (Black & Jepson, 1998), and the kernel-based method (Comaniciu et al., 2003), etc. was reported in literatures, where a discriminative model for separating the object from the background is trained offline and applied before tracking and termed tracking-by-detection method (Andriluka et al., 2008 ; Avidan 2004; Breitenstein et al., 2009 ; Choudhury et al., 2003; Leibe et al., 2008; Okuma et al., 2004; Wu & Nevatia, 2007). However, to formulate that as an object-background discrimination problem, two important factors need to be treated carefully: what features to choose and how to train the classifiers. Furthermore, since the object and background appearance may change greatly over time; online feature selection and classifier training are necessary to adapt the tracker to such variations. The appearance-based method deals with it by updating a holistic representation of object in a feature space with online learning, such as the incremental PCA (Wang et al., 2007; Ross et al., 2008), the incremental LDA (Lin et al., 2007; Li et al., 2008), the Expectation Maximization method (Jepson et al., 2003), etc. The online feature selection method chooses the most discriminative features that could distinguish the object from the background correctly rather than specifying features beforehand (Collins et al., 2005). The feature fusion method in (Yin et al., 2008) combines multiple features by weighting the likelihood maps with respect to their variance ratios. Additionally, other feature fusion methods for tracking such as the weighted averaging rule, the product rule, the maximum rule, the minimum rule and the dynamic weighting rule are compared in (Conaire et al., 2006). An ensemble of weak classifiers is trained with the offline boosting algorithm and updated online weakly in (Avidan, 2007) for identifying the pixels of object from that of background, and an improved version is proposed in (Grabner et al., 2006) with online boosting that endows the tracker the adaptability somehow. An ensemble of SVM classifiers for tracking is built in (Tian et al., 2007) by seeking the linear separating hyperplane which maximizes the margin between the positive and the negative sample in a kernel space, and that handles appearance changes somehow by heuristically updating a SVM queue online.

It is reported that the increment of the number of features used for tracking could benefit the performance of tracker (Isard & Blake, 1998; Wu & Huang, 2004); however, it depends on how the features are utilized. Online feature selection is advocated in (Collins et al., 2005; Grabner et al., 2006), but it is difficult to determine how many features should be chosen beforehand. Online feature fusion is proposed in (Yin et al., 2008) where the features are weighted and combined with respect to their variance ratios but the final combination is in fact not the most discriminative. Moreover, unlabelled data is valuable for classifier training, though they do not improve the performance always (Chapelle et al., 2006; Zhu & Goldberg, 2009). It is demonstrated recently that the performance of tracker could be significantly improved by training classifier on a labeled dataset augmented by unlabelled ones with the semi-supervised learning technologies; however, it depends on how to predict the labels on unlabeled data correctly. The self-training method updates the classifier online with its own predictions to adapt the tracker to the appearance changes in (Collins et al., 2005; Grabner et al., 2006), however incorrect predictions could deteriorate the tracker and even cause the tracking drifting, which is called tracking label jitter. Therefore, the unlabelled data need to be treated carefully when updating the classifier. The template is updated based on the geometric information heuristically in (Matthews et al., 2004) to make the tracker adaptive and avoid drifting. The semi-supervised online boosting method in (Grabner et al., 2008; Godec et al. 2009; Leistner et al. 2008) formulates the update process in a semi-supervised fashion as a combined decision of a given prior and an online classifier and can limit the drifting somehow while adaptive to appearance changes. An improved version is proposed in (Stalder et al., 2009) to adapt rapid appearance changes of the target object that may result in tracking label jitter, by optimizing a robust loss function based on a convex combination of a supervised and an unsupervised classifier. Recently online random forest is proposed for tracking in (Saffari et al. 2009; Godec et al. 2010), which outperform the online boosting based trackers in case of severe occlusions and large appearance changes. An online multi-classifier boosting algorithm is proposed in (Kim et al. 2010) for learning object multi-modal appearance models and tracking under rapid appearance changes. Tracking label jitter can also be handled by online multiple instance learning methods and in principle the classifier is still performing self-training (Babenko et al., 2009; Saffari et al., 2009). The co-training and the multi-view learning method in (Javed et al., 2005; Leistner et al., 2010; Liu et al. 2009; Tang et al., 2007; Yu et al. 2008) updates the classifiers online each other with the predictions generated on different features and can avoid tracking drifting somehow.

The statistical methods model the underlying dynamics of a tracking system in a state space. Assuming a linear Gaussian model of system dynamics, only one mode appears in the posterior probability density function (PDF); for example Kalman filter performs tracking by updating the mean and the covariance of the posterior PDF. However the dynamics of a practical tracking system is usually nonlinear and non-Gaussian; it is impossible to estimate the distribution analytically; therefore, many statistical approximation algorithms have been developed in recent years. Among them, particle filter, also called sequential Monte Carlo, is the most popular state estimation method, which constructs the posterior PDF recursively in a state space using Monte Carlo integration (Doucet et al. 2001; Cappe et al., 2007; Doucet & Johansen, 2010). It is developed for object tracking in computer vision research communities originally in (Isard & Black, 1998), also termed Condensation algorithm. During the tracking process, object state can be estimated recursively with particle filter, but over depletion of particles by sequential importance resampling (SIR) could cause tracking failure. It is very important to preserve particles diversity, which is measured usually by the effective sample size (Doucet et al. 2001). Various methods have been proposed to tackle this problem in recent years (Doucet et al. 2001; Cappe et al., 2007; Doucet & Johansen, 2010).

In this chapter we propose an adaptive object tracking method that integrates a particle filter with an online semi-supervised classifier to treat the appearance variations of the object and the background and occlusions. An online real AdaBoost algorithm is presented to train an ensemble of weak classifiers that endows the strong classifier faster convergence speed and higher classification accuracy. We further improve the classifiers by co-training operated on two groups of uncorrelated local image features in order to reduce tracking label jitter. To deal with the problem of particles depletion, an iterative importance resampling algorithm to maximize the effective sample size with evolutionary operations is proposed, which gives more accurate estimations than the SIR method, and we term it the sequential evolutionary importance resampling (SEIR) method. The final tracker combines online real AdaBoost, co-training and the SEIR particle filter all together. The experimental results on pedestrian and vehicles tracking in the real world scenarios demonstrate that our method is very robust to tracking objects undergoing large appearance changes and severe occlusions.

The remainder of this chapter is organized as follows: Section 2 introduces the local image features for representing object; Section 3 discusses the online real AdaBoost and co-training algorithm; Section 4 presents the SEIR particle filter; Section 5 gives the experimental results; and finally Section 6 concludes the chapter.

2. Local image features for object representation

In order to identify pixels of the object from that of the background in a predicted candidate region at each image frame, two types of local image features, namely color features and texture features are employed to represent the appearance of the object and the background. The color features include the most wildly used color intensity RGB and gray level intensity Y. The texture features are the recently proposed local ternary pattern (LTP) (Tan & Triggs, 2010), which is in fact an improved version of local binary patterns (LBP) that is originally applied to texture classification (Ojala et al., 2002) and later extended to face recognition

(Ahonen et al., 2006). Nowadays the LBP pattern is one of the best performing texture descriptors. It has proven to be highly discriminative, invariant to monotonic gray level changes and computational efficient, make it much suitable for demanding image analysis tasks, and it has been used in various applications. Concretely LBP pattern is based on the difference between the central and the neighbor pixels in a mask to form a representation of texture pattern by coding a binary sequence, which is defined by

$$LBP_{P,r}(i,j) = \sum_{u=0}^{P\text{-}1} 2^u \cdot S(g_u - g_c) \tag{1}$$

where $S(x)$ is an indicating function, $S(x)=1$ if $x \ge 0$ else $S(x)=0$; r is the radius of a mask, P is the number of neighbor pixels in the mask, g_c is the gray value of the central pixel, g_u is the gray value of the neighbor pixels, u is the label of the neighbor pixels and (i,j) is the position of the central pixel. The most widely used masks and their neighbor pixels labeled by the black dots are shown in Figure 1.

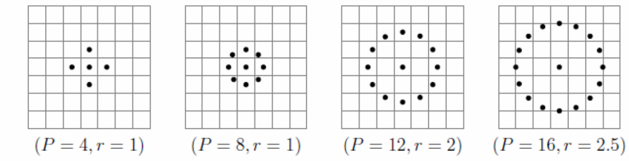
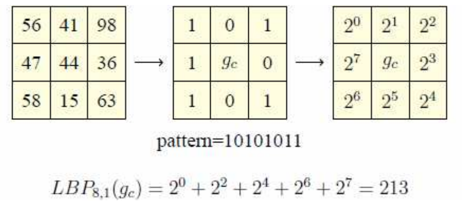


Fig. 1. The mask and the neighbor pixels

If set $P=8$, $r=1$ then we get the most widely used pattern LBP$_{8,1}$, named 1-radius 8-neighborhood pattern. Figure 2 shows how the pattern LBP$_{8,1}$ is calculated.



$$\begin{array}{|c|c|c|}\hline 56 & 41 & 98 \\ \hline 47 & 44 & 36 \\ \hline 58 & 15 & 63 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|}\hline 1 & 0 & 1 \\ \hline 1 & g_c & 0 \\ \hline 1 & 0 & 1 \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|}\hline 2^0 & 2^1 & 2^2 \\ \hline 2^7 & g_c & 2^3 \\ \hline 2^6 & 2^5 & 2^4 \\ \hline \end{array}$$

pattern=10101011

$$LBP_{8,1}(g_c) = 2^0 + 2^2 + 2^4 + 2^6 + 2^7 = 213$$

Fig. 2. Calculate the texture pattern LBP$_{8,1}$

One flaw of LBP pattern is that it cannot reflect the relationship between the central and the neighbor pixels in a mask completely, for example it cannot distinguish bright spots, dark spots and other small sized patterns; furthermore, it is very sensitive to noise. To deal with it, local ternary pattern (LTP) is developed recently (Tan & Triggs, 2010), which is defined by

$$LTP_{P,r}(i,j) = \sum_{u=0}^{P-1} 3^u \cdot \mathrm{S}'(g_u - g_c, t) \tag{2}$$

where $S'(x)$ is an indicating function with a threshold t specified beforehand.
To simplify the computation, the LTP pattern is divided into two parts: the upper pattern $S^u(x)$ and the lower pattern $S^l(x)$, that are defined respectively by

$$\mathrm{S}^u(x,t) = \begin{cases} 1 & x \ge t \\ 0 & x < t \end{cases}, \quad \mathrm{S}^l(x,t) = \begin{cases} 1 & x \le -t \\ 0 & x > -t \end{cases} \tag{3}$$

For each part, a similar operation as the calculation of LBP pattern is used to encode the pattern, and then they are combined into one unified representation. Figure 3 shows how the pattern $\mathrm{LTP}_{8,1}$ is calculated.

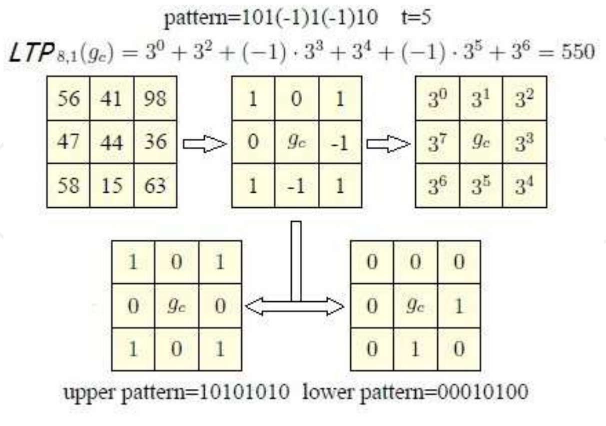

pattern=101(-1)1(-1)10 t=5
$LTP_{8,1}(g_c) = 3^0 + 3^2 + (-1)\cdot 3^3 + 3^4 + (-1)\cdot 3^5 + 3^6 = 550$
56 41 98
47 44 36
58 15 63
1 0 1
0 g_c -1
1 -1 1
3^0 3^1 3^2
3^7 g_c 3^3
3^6 3^5 3^4
1 0 1
0 g_c 0
1 0 1
0 0 0
0 g_c 1
0 1 0
upper pattern=10101010 lower pattern=00010100

Fig. 3. Calculate the texture pattern $\mathrm{LTP}_{8,1}$

The texture is represented by nine $\mathrm{LTP}_{8,1}$ patterns, which contains the in-channel texture feature R-R, B-B, G-G and the cross-channel texture feature R-G, R-B, G-R, G-B, B-R, and B-G. For example, when compute the R-G texture feature, around each pixel a 3x3 sized rectangle area is specified as the mask, the central pixel is taken from the red channel and the other eight neighbor pixels are taken from the green channel. Figure 4 shows the four color features and the nine texture features used in this chapter for representing the object and the background. It is obvious that the capability of each feature to discriminate the object from the background is different, and it may change greatly too when the object moves in the scene; we need to choose the best ones dynamically during tracking.

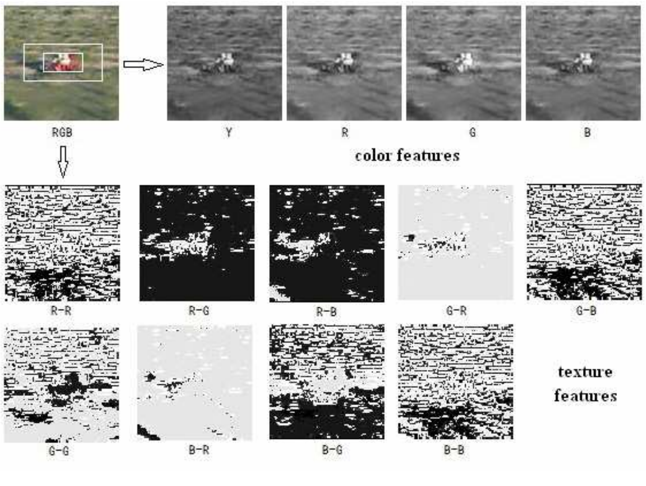


Fig. 4. The local image features

3. Locate object with online co-trained classifiers

3.1 Online real AdaBoost

To a practical tracking application, only a few training sample are available at each image frame, and the performance of the tracker depends heavily on the discriminative capability of the classifier. Boosting can improve the performance of weak learners by building a small number of weak classifiers iteratively and combining them into a strong one. AdaBoost is an adaptive boosting algorithm that combines weak classifiers together to adapt to the problem. Real AdaBoost deals with confidence-rated weak classifiers, mapping from a sample space to a real-valued prediction instead of a Boolean prediction. The real-valued weak classifiers have an advantage over the Boolean ones in discriminative capability. Although, the final classifier trained by discrete AdaBoost achieves a similar accuracy as that by real AdaBoost mostly, however, the former includes much more weak classifiers than the latter. Namely, real AdaBoost can achieve a higher accuracy if the number of weak classifiers is fixed beforehand (Schapire & Singer, 1999), exactly the case of online learning.

Unlike offline training, which uses all samples to train one weak classifier at the same time, the online version has a fixed-length classifier and uses only one sample on the entire stage. For each weak classifier on the online stage, there is a feature selector and each maintains its own information on features and updates it when a new sample is availab1e. The selector chooses the best feature with respect to the cumulated classification error; then updates the weights according to the output of the corresponding weak classifier and passes the sample to the next selector. The weights updating process boosts the online stage and enables the succeeding weak classifiers to focus on difficult sample. When the cumulated error is beyond a specified threshold, the weights updating steps are skipped. Online boosting is proposed originally in (Oza & Rusell, 2001) and later improved in (Grabner et al., 2006).

In this chapter, real AdaBoost is applied to select a group of optimal weak classifiers by minimizing the cumulated classification error on the available training sample at each image frame. In this way, the final strong classifier could adapt to appearance changes of the object and background somehow. A remarkable feature of the proposed method is that the optimal weak classifiers are chosen by real AdaBoost, which improves the classification performance significantly compared with that by discrete AdaBoost. The pseudo code of the proposed online real AdaBoost method is shown as Algorithm 1.

3.2 Online co-training

Co-training is a typical semi-supervised learning method originally introduced in (Blum & Mitchell, 1998) that allows starting with only a few labeled data to train classifiers initially and then apply more unlabeled data to improve accuracy of classification. The basic idea is that the features describing the data are redundant and could be split into different groups, each of which is sufficient for correct classification. One classifier is trained from one group of features, and the resulted two classifiers go through parallelly more unlabeled data, label them and add a part of the unlabeled ones having the most confident predictions to the labeled dataset; namely the classifier trains each other on the unlabeled data.

In this chapter we train two online ensemble classifiers parallelly by co-training, one from the group of color features and one from the group of texture features introduced in section 2, and each ensemble classifier is trained by online real AdaBoost. To predict the unlabeled data more reliably, a voting scheme for classifier combination is applied; logically, it is based on the degree of predicting agreement between the two ensemble classifiers. The final fused classifier is applied to locate the target object from each image frame. The pseudo code of the proposed online co-training method is shown as Algorithm 2.

Algorithm 1 : online real AdaBoost

```
Input:   training sample $(x,y) \in \chi \times \{-1,1\}$ ,
                     where $\chi$ is partitioned into several disjoint blocks $X_1, X_2, \cdots, X_k$ .
         weights $\lambda_{n,m}^{corr}$ , $\lambda_{n,m}^{wrong}$ (initialized with one)
Output:  strong classifier $h^{strong}$ (initialized randomly)

Initialize the importance weight $\lambda$ of the incoming sample with one.
for $n = 1,2,\cdots,N$ do // update all feature selector $h_n^{sel}$
  for $m = 1,2,\cdots,M$ do // one feature selector maintains $M$ features individually
    Initialize the weight of the sample $w_{n,m}(x,y) = w_{n,m}(x,y) + \lambda$ ,
            where $x \in X_j$ , $j \in \{1,2,\cdots,k\}$ .
  Update the weak classifier $h_{n,m}^{weak} = \frac{1}{2}\ln\left(\left(w_{n,m}(x,+1)+\varepsilon\right)/\left(w_{n,m}(x,-1)+\varepsilon\right)\right)$,
            where $\varepsilon$ is a small positive constant.
  Estimate the classification error $e_{n,m} = \lambda_{n,m}^{wrong} / \left(\lambda_{n,m}^{wrong} + \lambda_{n,m}^{corr}\right)$,
            if $h_{n,m}^{weak} \cdot y \geq 0$ then $\lambda_{n,m}^{corr} = \lambda_{n,m}^{corr} + \lambda\left|h_{n,m}^{weak}\right|$ ,
            else $\lambda_{n,m}^{wrong} = \lambda_{n,m}^{wrong} + \lambda\left|h_{n,m}^{weak}\right|$ end if.
  end for // feature selector
  Select the weak classifier having the lowest error $h_n^{sel} = h_{n,m^+}^{weak}$ , $m^+ = \arg\min_m\left(e_{n,m}\right)$ .
  if $e_{n,m^+} = 0$ or $e_{n,m^+} > 0.5$ then exit end if.
  Calculate the voting weight $\alpha_n = \ln\left(\left(1-e_{n,m^+}\right)/e_{n,m^+}\right)$ .
  Update the importance weight $\lambda$ ,
            if $h_{n,m}^{weak} \cdot y \geq 0$ then $\lambda = \lambda / \left(2\left(1-e_{n,m^+}\right)\right)$ ,
            else $\lambda = \lambda / \left(2e_{n,m^+}\right)$ end if.
  Replace the worst weak classifier $h_{n,m^-}^{weak}$ , $m^- = \arg\max_m\left(e_{n,m}\right)$ with a new one,
  set $\lambda_{n,m^-}^{corr} = 1$ , $\lambda_{n,m^-}^{wrong} = 1$ .
end for // update all feature selectors

The final classifier is $h^{strong} = sign\left(conf(x)\right)$ , the confidence score is $conf(x) = \sum_{n=1}^{N} \alpha_n h_n^{sel}(x)$ .
```

Algorithm 2 : online co-training

At the first image frame:

1. Locate the target object manually or automatically, and generate the labeled data.
2. Train two strong classifiers by online real AdaBoost shown in Table 1,
   One classifier from one group of features on the labeled data respectively.

At each newly coming image frame:

1. Apply the two strong classifiers pixel by pixel in a predicted candidate image region,
   generate two confidence score maps respectively.
2. Combine the two confidence score maps into one by voting:
   the label is $h^{fused} = sign\left(\min\left(conf_i\left(x\right)\right)\right)$,
   the confidence score is $conf^{fused}\left(x\right) = \max\left(\left|conf_i\left(x\right)\right|\right) \cdot h^{fused}\left(x\right)$.
3. Locate the target object from the image frame based on the combined confidence map.
4. Add a part of the pixels having the most confident scores to the labeled dataset.
5. Retrain the two strong classifiers on the updated labeled data respectively.

3.3 Locate the target object

Giving the two strong classifiers, to a newly incoming image frame, we need to determine the location of the target object within it. To do this, each classifier is operated pixel by pixel in a predicted candidate region at that image frame individually, thus yields two confidence score maps from the two groups of local features; and then they are combined into one. Figure 5 illustrates how the combined confidence score map is generated. In Figure 5, (a) is the predicted candidate region containing the target object; (b) and (c) are the confidence score maps from the texture and color features respectively; (d) and (e) are the confidence score maps of (b) and (c) after applying the morphological operation OPENING respectively; (f) is the final combined confidence score map. It is obvious that each feature contributes the map differently; it has fewer clutters than either one before combining, which means it can reduce tracking label jitter and may lead to more stable tracking.

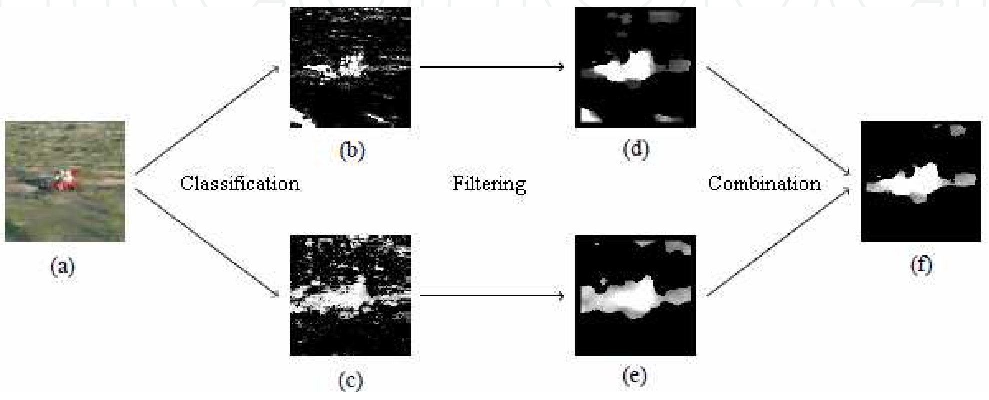


Fig. 5. Combine the confidence score maps into one

The location of the target object can be conjured from the combined confidence score map. Since local features are utilized, the confidence score can somehow measure the contribution of the pixel to the existence of the target object; the largest blob having highest accumulated confidence score locates the object most confidently. We check all the blobs within the predicted candidate region and choose the most confident blob as the target object and the center of the ellipse bounding the blob most closely is treated as object location. The whole locating procedure is visualized in figure 6, where, (a) is the predicted candidate region containing the target object; (b) is the confidence score map from the color features; (c) is the confidence score map from the texture features; (d) is the combined confidence score map; (e) is the result of (d) after thresholding; (f) shows the ellipse that bounding the blob most closely, of which the center indicates the object location.

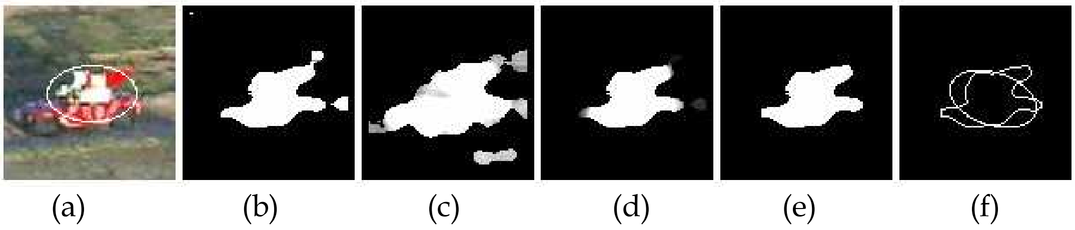


Fig. 6. Locate the target object from the combined confidence score map

Stable tracking depends on the discriminative ability of the strong classifiers trained by co-training, in fact which is heavily affected by the correctness of the sequentially predicted sample added to the labeled dataset. We must check the accuracy of classification carefully; however it is difficult to be evaluated online due to lacking ground-truth data. Fortunately, by checking the amount of object pixels within and background pixels outside of the object blob, we can guess that somewhat, though it is ad hoc. Usually the most confident blob is treated as the object; when the accumulated confidence score of pixels within it beyond a specified threshold, the pixels are reckoned as positive sample, and added to the labeled dataset, otherwise no one is selected; that happens mostly as occlusions appear. Meanwhile, the pixels near the boundary of the candidate region are treated as negative sample and added to the labeled dataset at any time to adapt the tracker to the varying background.

4. Predict object location with particle filter

4.1 Sequential importance resampling

To a practical object tracking system, we can describe the dynamics of the target object by

$$X_k = X_{k-1} + \delta_k \tag{4}$$

where X_k and X_{k-1} are the location of the target object at time k and $k-1$ respectively, δ_k is a random variable which subjects to the transition distribution $p\left(X_k \middle| X_{k-1}\right)$.

The posterior PDF is approximated recursively by a weighted sample, involving two steps mainly: prediction and update. Given the observations $Y_{1:k\text{-}1} = \left\{Y_1, \ldots, Y_{k\text{-}1}\right\}$ up to time $k-1$, at the prediction step, the transition distribution $p\left(X_k \middle| X_{k-1}\right)$ is applied to predict the posterior PDF at time k, termed the prior as well,

$$p\left(X_k \middle| Y_{1:k-1}\right) = \int p(X_k \mid X_{k-1}) p\left(X_{k-1} \mid Y_{1:k-1}\right) dX_{k-1} \tag{5}$$

At time k as the observation Y_k is available, applying the Bayes rule, the posterior PDF is

$$p\left(X_k\middle|Y_{1:k}\right)=\frac{p(Y_k\mid X_k)p\left(X_k\mid Y_{1:k-1}\right)}{p(Y_k\mid Y_{1:k-1})} \tag{6}$$

The posterior PDF is approximated by N sample X_k^i drawn from the proposal distribution $q\left(X_k\mid X_{1:k\text{-}1},Y_{1:k}\right)$ with the importance weight v_k^i,

$$v_k^i=v_{k-1}^i\frac{p(Y_k\mid X_k^i)p\left(X_k^i\mid X_{k-1}^i\right)}{q(X_k\mid X_{1:k-1},Y_{1:k})} \tag{7}$$

In the case of sequential importance resampling (SIR), $q\left(X_k\mid X_{1:k\text{-}1},Y_{1:k}\right)=p\left(X_k\mid X_{1:k\text{-}1}\right)$, the importance weight becomes the observation likelihood $p\left(Y_k\middle|X_k\right)$.

To the SIR particle filter, if it fails to generate new values for the states from the latest observations, only a few particles will have significant importance weights, the variance of weights will increase continuously and eventually cause tracking failure, which is termed particle degeneration. So it is very important to move particles towards the regions of high likelihood. This problem arises when the likelihood is too peaked or lies in the tail of the prior. Various approaches have been proposed to tackle this problem such as the auxiliary particle filter (APF), the unscented particle filter (UPF), the kernel particle filter (KPF), the regularized particle filter (RPF), to name a few (Doucet et al. 2001; Cappe et al., 2007).

4.2 Sequential evolutionary importance resampling

To keep the diversity of particles, we enhance importance resampling in this chapter with four evolutionary operations: copy, crossover, mutation and selection.

1. COPY: generate N new particles X_k^{l1} by duplicating all the existing N particles,

$$X_k^{l1}=X_k^i \tag{8}$$

2. CROSSOVER: generate N new particles X_k^{l2} based on N pair of particles X_k^i and X_k^j chosen randomly from the existing N particles according to the probability $p\left(Y_k\middle|X_k^i\right)$ and $1-p\left(Y_k\middle|X_k^j\right)$,

$$X_k^{l2}=X_k^j+\mu\cdot(X_k^i-X_k^j) \tag{9}$$

where μ is a random number which subjects the standard uniform distribution.

3. MUTATION: generate N new particles X_k^{l3} by disturbing the existing N particles,

$$X_k^{l3}=X_k^i+\lambda\cdot\nu \tag{10}$$

where ν is a random number which subjects the standard normal distribution, and λ is a constant which controls the amplitude of interference.

4. SELECTION: resample N particles from the resulted $3N$ new particles $\left\{X_k^{l1},X_k^{l2},X_k^{l3}\right\}$ according to their importance weights.

In fact, the operation CROSSOVER and MUTATION can increase the amount of distinctive particles, the operation COPY and SELECTION can keep the particles staying in the regions of high likelihood. The combination of the four operations can make particles move towards the regions of high likelihood so as to overcome the degeneration of particles.

The effective sample size (ESS) is a function of the coefficient of the variation of importance weights, measuring the efficiency of an importance sampling algorithm, which is defined by

$$N_{eff} = 1 \Big/ \sum_{i=1}^{n} \left(\tilde{v}_k^i \right)^2 \tag{11}$$

where $\tilde{v}_k^i$ is the normalized importance weight, which is defined by

$$\tilde{v}_k^i = v_k^i \Big/ \sum_{i=1}^{n} \left(v_k^i \right) \tag{12}$$

When ESS is small, it means there is a risk of degeneration; importance resampling should be conducted to augment the particles by applying COPY, CROSSOVER, MUTATION and SELECTION until ESS becomes large enough. The pseudo code of the proposed sequential evolutionary importance resampling method is detailed as Algorithm 3.

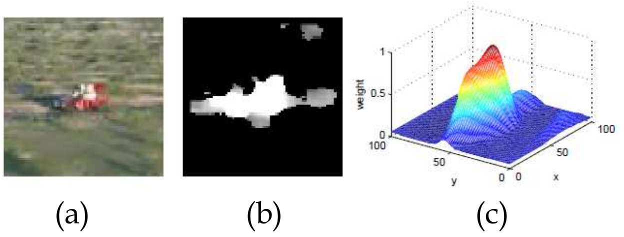
Algorithm 3: sequential evolutionary importance resampling

```
1: Initialization, $k = 1$
   Sample $X_1^i \sim p(X_1)$, $i = 1, \cdots, N$.
   Evaluate the importance weight $v_1^i = p\left(Y_1 \mid X_1^i\right)$.
   Resample $\left\{X_1^i, v_1^i\right\}$ to generate the $N$ equally weighted particles $\left\{X_1^i, \frac{1}{N}\right\}$.

2: Importance sampling, $k > 1$
   Sample $\tilde{X}_k^i \sim p\left(X_k \mid X_{k-1}^i\right)$, $i = 1, \cdots, N$.
   Evaluate the importance weight $v_k^i = p\left(Y_k \mid \tilde{X}_k^i\right)$.
   Normalize the importance weight to obtain $\tilde{v}_k^i = v_k^i \Big/ \sum_{i=1}^{n} \left(v_k^i\right)$.

3: Evolutionary importance resampling
  Compute the effective sample size $N_{eff}$, and set the loop counter $T = 0$.
  While ( $N_{eff}$ or $T$ is not large enough)
     Generate $N$ new particles $X_k^{l1}$ by COPY, $l1 = 1, \cdots, N$,
        evaluate the importance weight $v_k^{l1} = p\left(Y_k \mid X_k^{l1}\right)$.
     Generate $N$ new particles $X_k^{l2}$ by CROSSOVER, $l2 = 1, \cdots, N$,
        evaluate the importance weight $v_k^{l2} = p\left(Y_k \mid X_k^{l2}\right)$.
     Generate $N$ new particles $X_k^{l3}$ by MUTATION, $l3 = 1, \cdots, N$,
        evaluate the importance weight $v_k^{l3} = p\left(Y_k \mid X_k^{l3}\right)$.
     Normalize the importance weight of the $3N$ particles $\left\{X_k^{l1}, X_k^{l2}, X_k^{l3}\right\}$.
     Resample $N$ particles $X_k^i$ by SELECTION from the $3N$ particles $\left\{X_k^{l1}, X_k^{l2}, X_k^{l3}\right\}$.
     Normalize the importance weight of the selected $N$ particles to obtain $\tilde{v}_k^i$.
     Compute the effective sample size $N_{eff}$ and set $T = T + 1$.
  end while
  Resample $\left\{X_k^i, \tilde{v}_k^i\right\}$ to generate the $N$ equally weighted particles $\left\{X_k^i, \frac{1}{N}\right\}$.
  Set $k = k + 1$ and go to step 2.
```

4.3 Predict the location of the target object

When performing tracking, a particle is a candidate object state, actually corresponding to a possible location of the target object; of which the importance weight is proportional to the observation likelihood, that in fact indicates the possibility whether the object appears there; Theoretically, the likelihood must be proportional to the accumulated confidence scores of pixels classification around the position; practically, it is computed by adding the confidence scores together in a specified window at that position in this chapter.

Figure 7 exemplifies the distribution of accumulated confidence scores in a predicted region, where, (a) is the predicted image region containing the target object; (b) is the combined confidence score map yield by the co-trained classifier; (c) is the distribution of accumulated confidence scores. It is obvious in (c) that there is one mode holding a very high likelihood value, which may result in an accurate prediction on object location.



(a) (b) (c)

Fig. 7. The distribution of accumulated confidence scores

To track an object in complex scene, actually, pixels classification error is inevitable, means clutters must appear at the predicted object region. To reduce the interference of clutters, we use a gating function to penalize pixels by weighting; concretely, pixels near the predicted position get a higher weight, otherwise gets a lower weight. In this chapter, we choose a Gaussian $g(X;\mu,\sigma)$ as the gating function, let the distribution of accumulated confidence scores is $l(X)$, then the observation likelihood is defined by

$$p(Y\,|\,X) = l(X)\cdot g(X;\mu,\sigma) \tag{13}$$

5. Experimental results

This section presents two tracking examples on practical video sequences that illustrate the benefits of combining online feature selection and co-training techniques. Specifically, these benefits are the enhanced ability to track objects undergoing large appearance variations, the ability to adapt to the changing background and illumination conditions, the ability to treat severe occlusions, and the ability to avoid distraction by similar objects by emphasizing automatically certain features that are distinctive to the target object.

5.1 Results on pixels classification

Figure 8 shows the comparison result on pixels classification by the co-trained and the self-trained classifier, where, (a) is the predicted candidate image region containing the target object; (b) and (c) are the confidence score map respectively from the texture and the color features yield by the co-trained classifier; (d) is the combined confidence score map from (b) and (c); (e) is the confidence score map from the mixed texture and color features yield by the self-trained classifier; (f) is the result of (e) after applying the morphological operation OPENING. It is obvious that the score map yield by the co-trained classifier has much fewer clutters than the one by the self-trained classifier, which may lead to more stable tracking.

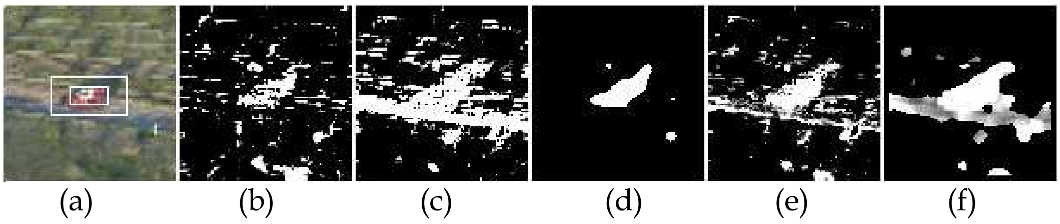


Fig. 8. Pixels classification by the co-trained and the self-trained classifier

5.2 Results on tracking objects in complex scene

The proposed method is applied to track vehicles and pedestrians in complex scene. The test video data consists of a vehicle sequence from PETS 2005 dataset (Collins et al., 2005) and a pedestrian sequence captured by us. Especially, the vehicle sequence involves significant appearance variations resulting from the change of object size, environmental lighting and distractions by similar objects in the scene. Besides, in the pedestrian sequence, the object is occasionally occluded severely by background clutters for several times.

Figure 9 (a) and (b) shows the results on tracking a vehicle and a pedestrian respectively, where the red box indicates the location of the target object yield by the proposed tracker. The experimental results demonstrate that the proposed method can enhance the ability to effectively track objects undergoing large appearance variations due to size change, and to adapt the background and illumination change, severe occlusions, and even the distractions by similar objects.

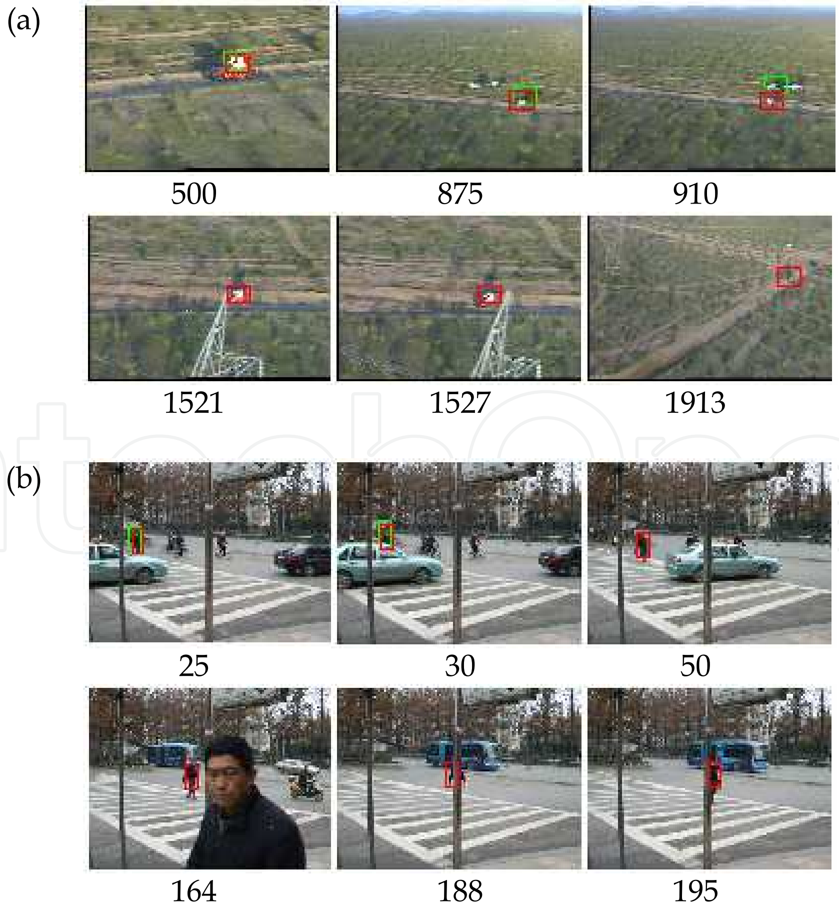


Fig. 9. Results on tracking vehicle and pedestrian

To check why it can do these, the results on pixels classification yield by the co-trained classifier in the predicted candidate regions on the selected frames from the two sequences are visualized in figure 10 and 11 respectively, where, (a) is the predicted candidate image region, (b) is the combined confidence score map. As can be seen, the combined map in (b) has very few clutters; based on that the target object can be located very accurately, in fact, the white bounding ellipse in (a) predict the object location correctly.

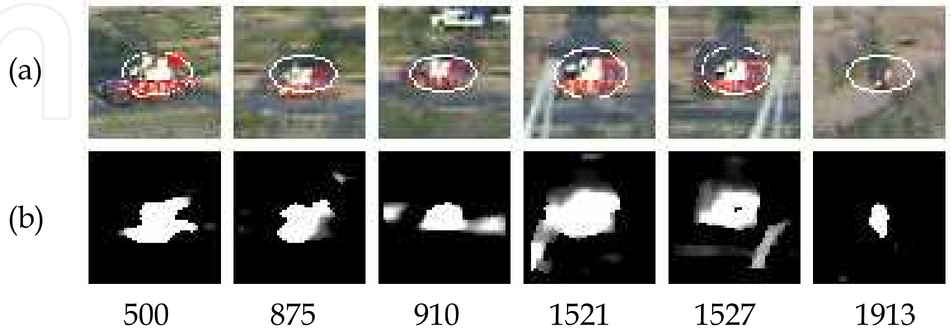


Fig. 10. Pixels classification by the co-trained classifier on the vehicle sequence

5.3 Results on importance resampling

Figure 12 shows the comparison results on vehicle tracking with state estimation by particle filter using different importance resampling method, where, the object locating results are shown in (a); the effective sample size and maximum loop number yield by the sequential evolutionary importance resampling (SEIR) particle filter are shown respectively in (b) and (c). The results demonstrate that the SEIR particle filter yields more accurate state estimation than the SIR particle filter. It is obvious in (a) that the red box (SEIR) encloses the object more closely than the green box (SIR) and the latter loses tracking at frame 1600, which may due to the inaccurate state estimation. The figure in (b) shows that the SEIR particle filter (red line) can maintain high efficiency of importance resampling while the SIR particle filter (green line) is much lower, where in total 300 particles are used. Furthermore within finite times, only about three loops averagely, the expected efficiency can be reached by the SEIR particle filter, means it is computationally very efficient.

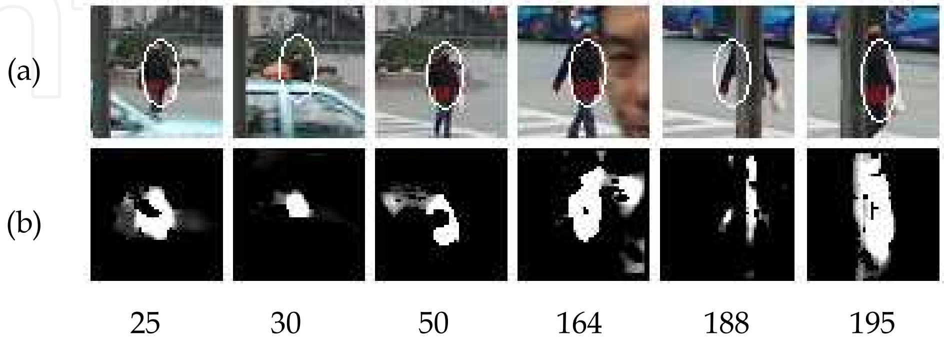


Fig. 11. Pixels classification by the co-trained classifier on the pedestrian sequence

Figure 13 shows the variation on scatter of particles when conducting the evolutionary importance resampling at some frame. In figure (a) to (d), the effective sample size is 62, 186,

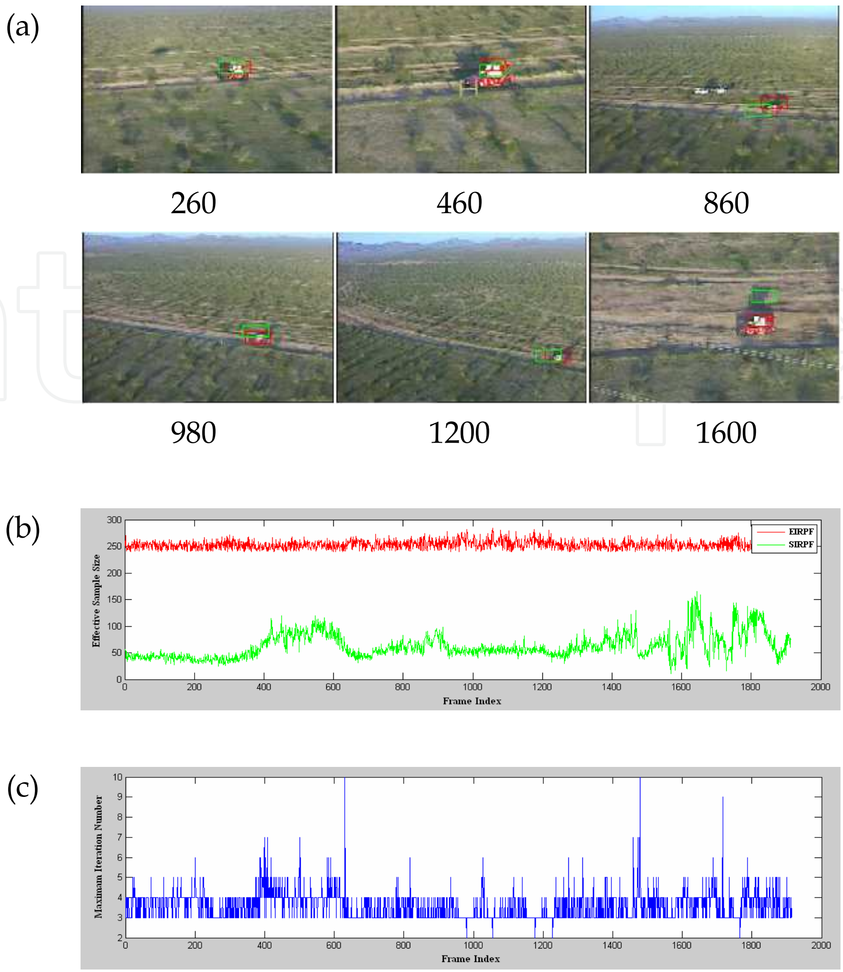


Fig. 12. Results on vehicle tracking with state estimation by particle filter

236 and 259 respectively. It is obvious that the particles move quickly towards the center of the target object and cover it more closely, meanwhile the effective sample size increases accordingly; and that may lead to more accurate state estimation than the SIR particle filter. Figure 14 shows the scatter of particles after importance resampling at some frame, where, (a) is the predicted candidate image region containing the target object; (b) is the combined confidence score map yield by the co-trained classifier; (c) shows the distribution of particles after SEIR; (d) shows the distribution of particles after SIR. It is obvious that though there are some clutters in (b), yield by inaccurate pixels classification, through conducting SEIR, particles move towards the true location of the target object, which is demonstrated by the number and the spread of the mode appearing in the distributions of particles. There are fewer modes in (c) than in (d) and the spread of mode is much lower in (c) than that in (d).

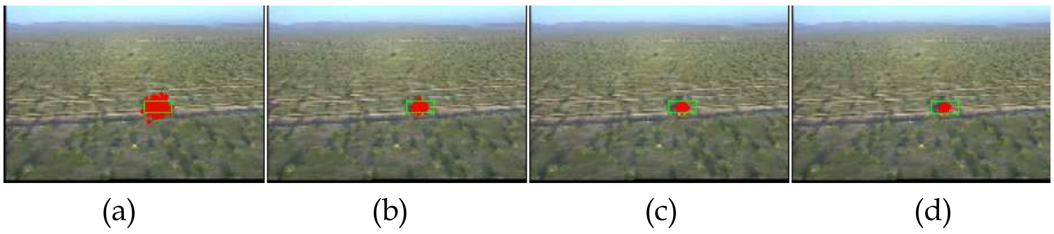


Fig. 13. Scatter of particles when conducting importance resampling

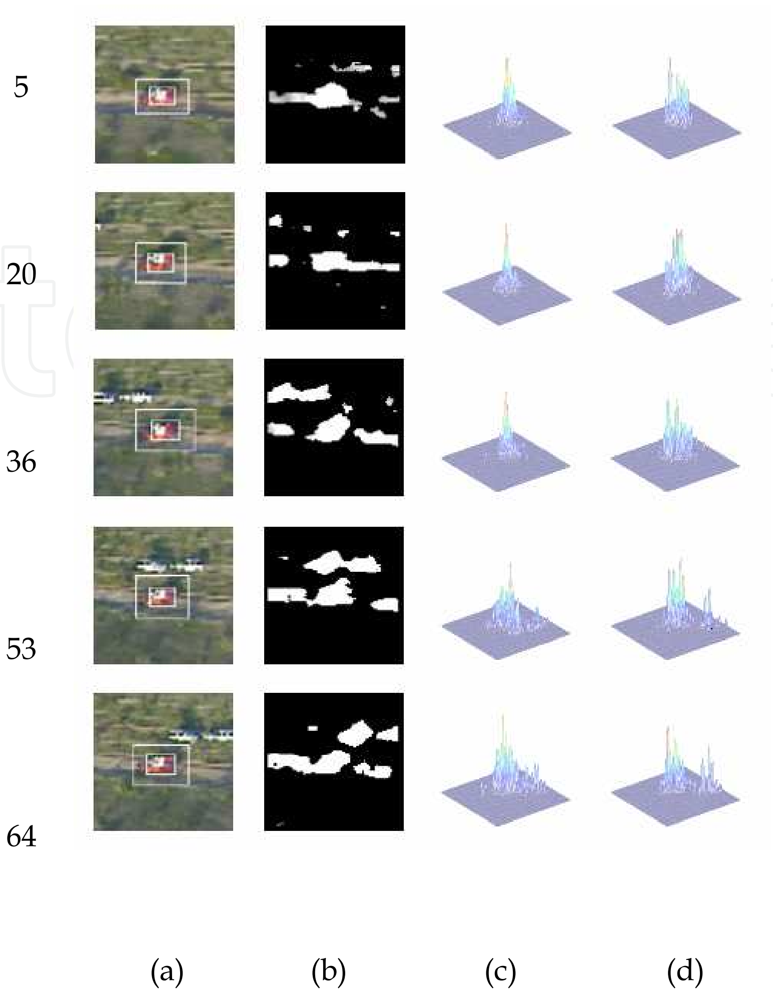


Fig. 14. Scatter of particles after importance resampling

6. Conclusions

Object tracking is still a challenging and open task in computer vision research, especially when dealing with large appearance variations and severe occlusions. In this chapter, an adaptive object tracking method is proposed, which integrates online semi-supervised classification and particle filter efficiently. Thanks to the feature selection capability achieved by online real AdaBoost and co-training, two classifiers are trained on two groups of features complementarily; the tracker consistently provides accurate classification of the object and the background for stable tracking, even under severe situations.

In fact the efficiency of the proposed method comes from three aspects. First of all, the real AdaBoost approach constructs ensemble classifiers with a soft decision scheme to adapt to aspects, occlusions and size variations of the object by online learning. Second, classifier co-training on two uncorrelated groups of features further improves the classification accuracy on the object from the background. Finally, the particle filter with sequential evolutionary importance resampling can adapt the nonlinearity of object dynamics and integrates the confidence scores updated online to predict object state more accurately even when it is occluded by background clutters completely.

7. References

Ahonen, T.; Hadid, A. & Pietikainen, M. (2006). Face Description with Local Binary Patterns: Application to Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 28, No. 12, (Dec. 2006) pp. 2037-2041.

Andriluka, M.; Roth, S. & Schiele, B. (2008). People tracking by detection and people detection by tracking, *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, Anchorage, Alaska, USA, Jun. 2008.

Avidan, S.; Support vector tracking (2004). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 8, (Aug. 2004) pp. 1064-1072.

Avidan, S.; Ensemble Tracking (2007). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 29, No. 2, (Feb. 2007) pp. 261-271.

Babenko, B.; Yang, M-H & Belongie, S. (2009). Visual tracking with online multiple instance learning, *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, Miami, FL, USA, Jun. 2009.

Baker, S. & Matthews, I. (2004). Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, Vol. 56, No.3, (Mar. 2004) pp.221-255.

Black, M.J. & Jepson, A.D. (1998). EigenTracking: Robust Matching and Tracking of Articulated Objects Using a View-Based Representation. *International Journal of Computer Vision*, Vol. 26, No.1, (Jan. 1998) pp. 63-84.

Blum, A. & Mitchell, T. (1998). Combining labeled and unlabeled data with co-training, *Proceedings of the 11th Annual Conference on Computational Learning Theory*, pp. 92-100, Madison, Wisconsin, USA, Jul. 1998.

Breitenstein, M.D.; Reichlin, F., Leibe, B., Koller-Meier, E. & Gool, L.V. (2009). Robust tracking-by-detection using a detector confidence particle filter, *Proceedings of the 12th IEEE International Conference on Computer Vision*, pp. 1515-1522, Kyoto, Japan, Sep. 2009.

Cappe, O.; Godsill, J.S. & Moulines, E. (2007). An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo. *Proceeding of IEEE*, Vol. 95, No. 5, (May 2007) pp. 899-924.

Chapelle, O.; Scholkopf, B. & Zien, A. (2006). *Semi-Supervised Learning*. The MIT Press, ISBN: 978-0-262-03358-9, London, England.

Choudhury, R.; Schmid, C. & Mikolajczyk, K. (2003). Face detection and tracking in a video by propagating detection probabilities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 10, (Oct. 2003) pp. 1215-1228.

Collins, R.T.; Liu, Y. & Leordeanu, M. (2005). Online Selection of Discriminative Tracking Features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 10, (Oct. 2003) pp. 1631-1643.

Collins, R.T.; Zhou, X. & Teh, S.K. (2005). An Open Source Tracking Testbed and Evaluation Web Site, *Proceedings of the IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS 2005)*, pp. 1-8, Breckenridge, USA, Jan. 2005.

Comaniciu, D.; Ramesh, V. & Meer, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 5, (May 2003) pp. 234-240.

Conaire, C.O.; O'Connor, N.E., Cooke E. & Smeaton, A.F. (2006). Comparison of Fusion Methods for Thermo-Visual Surveillance Tracking, *Proceedings of the 9th IEEE International Conference on Information Fusion*, pp.1-7, Florence, Italy, Jul. 2006.

Doucet, A.; Freitas, N.D. & Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*. Springer, ISBN: 978-0-387-95146-1.

Doucet, A. & Johansen, A. M. (2010). A Tutorial on Particle Filtering and Smoothing: Fifteen years later, in *Oxford Handbook of Nonlinear Filtering*, D. Crisan, B. Rozovsky (Ed.), Oxford University Press, London UK.

Feng, Z.; Lu, N. & Jiang, P. (2008). Posterior probability measure for image matching. *Pattern Recognition*, Vol. 41, No. 7, (Jul. 2008) pp. 2422-2433.

Godec, M., Grabner, H., Leistner, C. & H. Bischof. (2009). Speeding up Semi-Supervised On-line Boosting for Tracking, *Proceedings of the 33rd annual Workshop of the Austrian Association for Pattern Recognition*, pp. 1-8, Stainz, Austria, May 2009.

Godec, M., Leistner, C., Saffari, A. & Bischof, H. (2010). On-line Random Naive Bayes for Tracking, *Proceedings of the IEEE 20th International Conference on Pattern Recognition*, pp. 1-4, Istanbul, Turkey, Aug. 2010.

Grabner, H.; Grabner, M. & Bischof, H. (2006). Real-time tracking via on-line boosting, *Proceedings of the 17th British Machine Vision Conference*, pp. 47-56, Edinburgh UK, Sep. 2006.

Grabner, H.; Leistner, C. & Bischof, H. (2008). Semi-supervised On-Line Boosting for Robust Tracking, *Proceedings of the 10th European Conference on Computer Vision*, pp. 234-247, Marseille, France, Oct. 2008.

Hieu, N. & Arnold, S. (2006). Robust Tracking Using Foreground-Background Texture Discrimination. *International Journal of Computer Vision*, Vol. 69, No. 3, (Jun. 2006) pp. 277-293.

Isard, M.; & Blake, A. (1998). CONDENSATION: Conditional Density Propagation for Visual Tracking. *International Journal of Computer Vision*, Vol. 29, No.1, (Jan. 1998) pp. 5-28.

Isard, M. & Blake, A. (1998). ICONDENSATION: Unifying Low-Level and High-Level Tracking in a Stochastic Framework, *Proceedings of the 6th European Conference on Computer Vision*, pp.893-908, Freiburg, Germany, Jun. 1998.

Javed, O.; Ali, S. & Shah, M. (2005). Online Detection and Classification of Moving Objects Using Progressively Improving Detectors, *Proceedings of the IEEE 2005 Conference on Computer Vision and Pattern Recognition*, pp. 1-8, San Diego, Jun. 2005.

Jepson, A.; Fleet, D. & El-Maraghi, T. (2003). Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25, No. 10, (Oct. 2003) pp.1296-1311.

Kim, T-K; Woodley, T., Stenger, B. & Cipolla, R. (2010). Online Multiple Classifier Boosting for Object Tracking, Proceedings of the 2010 IEEE Conference on computer Vision and Pattern Recognition, pp. 1-6, San Francisco, CA, USA, Jun. 2010.

Leibe, B.; Schindler, K., Cornelis, N. & Gool, L.V. (2008). Coupled object detection and tracking from static cameras and moving vehicles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 30, No. 10, (Oct. 2008) pp. 1683-1698.

Leistner, C., Godec, M., Saffari, A. & Bischof, H. (2010). On-line Multi-View Forests for Tracking, *Proceedings of the 32nd annual symposium of The German Association for Pattern Recognition*, pp.1-10, Darmstadt, Germany, Sep. 2010.

Leistner, C.; Grabner, H. & Bischof, H. (2008). Semi-Supervised Boosting using Visual Similarity Learning, *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1-8, Anchorage, Alaska, USA, Jun. 2008.

Li, G.; Liang, D., Huang, Q., Jiang, S. & Gao, W. (2008). Object tracking using incremental 2D-LDA learning and Bayes inference, *Proceedings of the IEEE 15th International Conference on Image Processing*, pp. 1568-1571, San Diego, California, USA, Oct. 2008.

Lin, R-S; Yang, M-H & Levinson, S.E. (2004). Object Tracking Using Incremental Fisher Discriminant Analysis, *Proceedings of the IEEE 17th International Conference on Pattern Recognition*, pp. 757-760, Cambridge, UK, Aug. 2004.

Liu, R.; Cheng, J. & Lu, H. (2009). A robust boosting tracker with minimum error bound in a co-training framework, *Proceedings of the 12th IEEE International Conference on Computer Vision*, pp. 1459-1466, Kyoto, Japan, Sep. 2009.

Lucas B. & Kanade T. (1981). An iterative image registration technique with an application to stereo vision, *Proceedings of the 7th international joint conference on Artificial intelligence*, pp.674-679, Vancouver, British Columbia, Aug. 1981.

Matthews, I.; Ishikawa, T. & Baker, S. (2004). The template update problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No.6, (Jun. 2004) pp. 810-815.

Mallapragada, P.K.; Jin, R., Jain, A.K. & Liu, Y. (2009). SemiBoost: Boosting for Semi-Supervised Learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 31, No. 11, (Nov. 2009) pp. 2000-2014.

Ojala, T.; Pietikainen, M. & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 7, (Jul. 2002) pp. 971-987.

Okuma, K.; Taleghani, A., Freitas, N.D., Little, J., & Lowe, D. (2004). A boosted particle filter: Multitarget detection and tracking, *Proceedings of the 8th European Conference on* Computer Vision, pp. 28-39, Prague, Czech, May 2004.

Oza, N. & Rusell, S. (2001). Online bagging and boosting, *Proceedings of the 8th International Workshop on Artificial Intelligence and Statistics*, pp.105-112, Key West, FL, USA, Jan. 2001.

Ross, D.A.; Lim, J., Lin, R-S & Yang, M-H (2008). Incremental Learning for Robust Visual Tracking. *International Journal of Computer Vision*, Vol. 77, No. 3, (Mar. 2008) pp. 125-141.

Saffari, A.; Leistner, C., Santner, J., Godec, M. & Bischof, H. (2009). On-line random forests, *Proceedings of the 12th IEEE International Conference on Computer Vision*, pp. 1-8, Kyoto, Japan, Sep. 2009.

Schapire, R.E. & Singer, Y. (1999). Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning*, Vol. 37, No. 3, (Dec. 1999) pp. 297-336.

Stalder, S.; Grabner, H. & Gool, L.V. (2009). Beyond Semi-Supervised Tracking: Tracking Should Be as Simple as Detection, but not Simpler than Recognition, *Proceedings of the 12th IEEE International Conference on Computer Vision*, pp. 1409-1416, Kyoto, Japan, Sep. 2009.

Tan, X. & Triggs, B. (2010). Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE Transactions on Image Processing*, Vol. 19, No. 6, (Jun. 2010) pp. 1635-1650.

Tang F.; Brennan S., Zhao Q. & Tao H. (2007). Co-tracking using semi-supervised support vector machines, *Proceedings of the 11th IEEE International Conference on Computer Vision*, pp. 1-8, Rio de Janeiro, Brazil, Oct. 2007.

Tian, M.; Zhang, W. & Liu, F. (2007). On-Line Ensemble SVM for Robust Object Tracking, *Proceedings of the 8th Asian Conference on Computer Vision*, pp. 355-364, Tokyo Japan, Nov. 2007.

Wang, T.; Gu, I.Y.H. & Shi, P. (2007). Object Tracking Using Incremental 2D-PCA Learning and ML Estimation, *Proceedings of the IEEE 32nd International Conference on Acoustics, Speech, and Signal Processing*, pp. 933-936, Honolulu, Hawaii, USA, Apr. 2007.

Wu, B. & Nevatia, R. (2007). Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. *International Journal of Computer Vision*, Vol. 75, No. 2, (Feb. 2007) pp. 247-266.

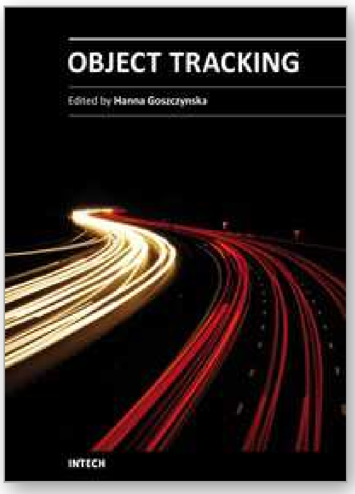
Wu, Y. & Huang, T. S. (2004). Robust Visual Tracking by Integrating Multiple Cues Based on Co-Inference Learning. *International Journal of Computer Vision*, Vol. 58, No. 1, (Jan. 2004) pp. 55-71.

Yilmaz, A.; Javed, O. & Shah, M. (2006). Object Tracking: A Survey, *ACM Computing Surveys,* Vol. 38, No. 4, (Dec. 2006) pp. 1-45.

Yin, Z.; Porikli, F. & Collins, R.T. (2008). Likelihood Map Fusion for Visual Object Tracking, *Proceedings of the IEEE 2008 Workshop on Application of Computer Vision*, pp. 1-7, Copper Mountain, USA, Jan. 2008.

Yu, Q.; Dinh, T.B. & Medioni, G. (2008). Online Tracking and Reacquisition Using Co-trained Generative and Discriminative Trackers, *Proceedings of the 10th European Conference on Computer Vision*, pp. 678-691, Marseille, France, Oct. 2008.

Zhu, X. & Goldberg, A.B. (2009). *Introduction to Semi-Supervised Learning*. Morgan & Claypool Publishers, ISBN: 978-1-598-29548-1.



Object Tracking
Edited by Dr. Hanna Goszczynska

ISBN 978-953-307-360-6
Hard cover, 284 pages
Publisher InTech
Published online 28, February, 2011
Published in print edition February, 2011

Object tracking consists in estimation of trajectory of moving objects in the sequence of images. Automation of the computer object tracking is a difficult task. Dynamics of multiple parameters changes representing features and motion of the objects, and temporary partial or full occlusion of the tracked objects have to be considered. This monograph presents the development of object tracking algorithms, methods and systems. Both, state of the art of object tracking methods and also the new trends in research are described in this book. Fourteen chapters are split into two sections. Section 1 presents new theoretical ideas whereas Section 2 presents real-life applications. Despite the variety of topics contained in this monograph it constitutes a consisted knowledge in the field of computer object tracking. The intention of editor was to follow up the very quick progress in the developing of methods as well as extension of the application.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821