BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **16**, No 2
Sofia • 2016 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2016-0030

Object Tracking Based on Online Semi-Supervised SVM and Adaptive-Fused Feature

Ruxi Xiang[1,2,3], *Xifang Zhu*[1,2,3], *Feng Wu*[1,2,3], *Qinquan Xu*[1,2,3], *Jianwei Li*[4]

[1]*College of Optoelectronic Engineering, Changzhou Institute of Technology, 213002 China*
[2]*Changzhou Institute of Modern Optical Technology, 213002 China*
[3]*Changzhou Key Laboratory of Optoelectronic Materials and Devices, 213002 China*
[4]*Key Laboratory of Optoelectronic Technology and Systems of Ministry of Education, Chongqing University, Chongqing, 400044 China*
Email: zhuxfcz@yeah.net

Abstract: *In order to improve the performance of tracking, we propose a new online tracking method based on classification and adaptive fused feature. We first label a few positive and negative samples, train the classifier by the online SSSM* (*Semi-Supervised Support Vector Machine*) *learning and these labelled samples, and then locate the position of the object from the next frame according to the trained classifier. In order to adapt more of the new samples, we need to update the classifier by finding new samples with high confident value obtained by the trained classifier and add them into the online SSSM. Finally we also update the object model by the online incremental PCA* (*Principal Component Analysis*) *because of background clutter, heavy occlusion and complicated object appearance changes. Compared with the basic mean shift tracking and the ensemble tracking method, experimental results show that our tracking method is able to effectively handle heavy occlusion and background clutter in some challenge videos including some thermal videos.*

Keywords: *Visual tracking, SSSM, feature fusion, incremental PCA，online.*

1. Introduction

Recently, tracking problem formulated as a binary classification has been a promising direction used in some visual applications such as video surveillance and military field [1, 2]. The key idea of this kind of tracking method is to obtain an effective classifier from training samples which are labelled by online or offline method. In spite of speeding up the matching process by offline method, the classifier trained by offline method can't deal efficiently with changes in appearance or complex background, while the classifier, trained by the online method, on the other hand, can solve these problems effectively [3].

Many methods based on classifier have been proposed for visual tracking [1, 3-6]. S. Avidan [4], introduced the support vector tracker, combined the trained classifier by offline method with optical-flow feature to track the object. S. Avidan proposed the ensemble tracker which trained some weak classifiers by Adaboost method to obtain stronger classifier [5]. However, the ensemble tracker used the classification results with high confidence to update the classifier itself. Unfortunately, the classification error is gets more prominent over time and the tracking error increases as well. To further reduce the tracking error, co-tracking based on semi-supervised support vector machines [3] was proposed, with the underlying assumption that each feature is different and independent but completely independent features are difficult to obtain in reality. Co-tracking method makes use of two independent features to train the corresponding classifiers, produces the confidence maps for a new unlabelled sample using the trained classifiers, and then combines a final confidence map with the produced confidence maps. However, co-tracking method neglects the contribution of the each feature taking into consideration the fusion of two maps which were produced by the trained classifiers. To take full advantage of each feature, we combine all features to form an efficient feature which is helpful for distinguishing the object from the background by the distance between the candidate object and the model which is adaptively updated by online increment PCA [7].

Motivated by self-training and the fusion features idea [8], we present a new tracking method based on semi-supervised support vector machine and adaptive fused feature. We first label a few positive and negative samples, train the classifier by the online SSSM [9] learning and these labelled samples, and then locate the position of the object from the next frame according to the trained classifier. In order to adapt more of the new samples, we need to update the classifier by finding new samples with high confident value obtained by the trained classifier and add them into the online SSSM. Finally we also update the object model by the online incremental PCA because of background clutter, heavy occlusion and complicated object appearance changes. Compared with the basic mean shift tracking and the ensemble tracking method, experimental results show that our tracking method is able to effectively handle heavy occlusion and background clutter in some challenge videos including some thermal video.

The rest of this paper is organized as follows. Section 2 briefly reviews the online support vector machine and the self-training method. Section 3 describes the features fused by the new similarity MSSBRS and model object updated by incremental PCA. Our method is described in Section 4. Experimental results and comparisons are shown in Section 5. Finally, the conclusions are made in Section 6.

2. Background

2.1. Online support vector machine

Support Vector Machine (SVM) which is also a powerful learning tool for classification and regression samples easily solves small and high dimensional samples. When the number of the samples is very large the computation of SVM

will take considerable time. To reduce the computational time of training samples, Gert improves the basic SVM by incremental and decreasing leaning method [9].

Assume that training data $(\boldsymbol{x}, \boldsymbol{y})$ with $\boldsymbol{x} \in R^n$ and $\boldsymbol{y} \in \{-1, 1\}$ is obtained from several frames, the SVM classifier is then represented as $f(\boldsymbol{x}) = \boldsymbol{a}^{\mathrm{T}}\phi(\boldsymbol{x}) + b$, in which $\boldsymbol{a}$ is a weight vector, b is a constant, and $\phi(\boldsymbol{x})$ is a map function from the input space to feature space, N is the number of the training samples, and the classifier can be learned as

$$\min_{a}\left(\frac{1}{2}\|\boldsymbol{a}\|^2 + \omega\sum\nolimits_{i=1}^{N}\boldsymbol{\lambda}_i\right), \quad \text{s.t. } \boldsymbol{y}_i(\boldsymbol{a}\phi(\boldsymbol{x}) + b) \geq 1 - \boldsymbol{\lambda}_i. \tag{1}$$

Here $\lambda > 0$ is a slack parameter, and ω is a penalty factor which represents the degree of punishment to the error. To further simplify the formula (1) can be rewritten as

$$\min_{0 \leq a_i \leq \omega} W = \frac{1}{2}\sum\nolimits_{i,j}\alpha_i Q_{i,j}\alpha_j - \sum\nolimits_i \alpha_i + b\sum\nolimits_i y_i\alpha_i, \tag{2}$$

where $\boldsymbol{Q}_{i,j} = \boldsymbol{y}_i\boldsymbol{y}_j k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, and $k(,)$ is a kernel function including Gaussian RBF $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2h^2}\right)$ in which h is a variance of Gaussian RBF or polynomial kernel function $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j + 1)^d$ in which d is a value of degree. In this paper, we adopt the Gaussian RBF as our kernel function. The solution of the dual parameters $\{a, b\}$ can be defined by minimizing (2). These solutions need to satisfy the KKT (Karush-Kuhn-Tucker) conditions expressed by (1) and (2) because it is a necessary condition for a solution in nonlinear to be optimal:

$$\boldsymbol{f}_i = \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{\alpha}_i} = \sum\nolimits_j \boldsymbol{Q}_{i,j}\boldsymbol{\alpha}_j + \boldsymbol{y}_i b - 1 = \boldsymbol{y}_i f(\boldsymbol{x}_i) - 1, \tag{3}$$

$$\frac{\partial \boldsymbol{W}}{\partial b} = \sum\nolimits_i \boldsymbol{y}_i\boldsymbol{\alpha}_i = 0. \tag{4}$$

Based on the partial derivatives f_i, the training samples can be classified into three categories, the samples on the margin are usually called support vectors ($f_i = 0$), while samples exceeding the margin are error vector ($f_i < 0$), the remaining samples, within the margin, are called reserve vectors ($f_i > 0$).

In [9], incremental or decreasing SVM learning is a procedure that adds or removes samples from one vector of SVM according to the value f_c that is a partial derivative value adding the training yields margin each time. In the incremental procedure, new samples with $f_c > 0$ are directly added to the reserve vector set due to the fact that they don't affect the solution. On the contrary, all other new samples are added to the set of either margin vector or error vector according to the value of f_c and a_c. that is a coefficient being incremental. In the decreasing procedure, the samples are removed according to the value f_c and a_c, No matter how the samples are added or removed, they should satisfy the KKT conditions.

2.2. Self-training method

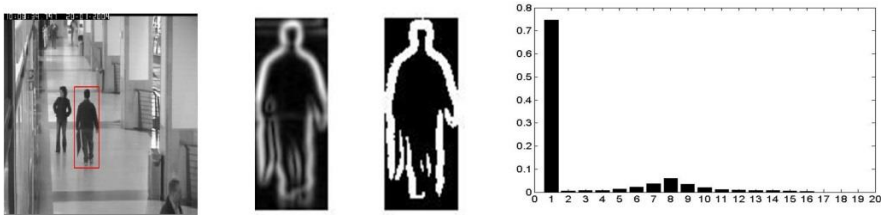
Self-training, also called self-teaching or bootstrapping, is a classical training algorithm in semi supervised leaning [10]. In self-training frame, a classifier is first trained with only a few labelled samples, and then unlabelled samples are classified by the classifier. Typically unlabelled samples with the highest confidence are added to the training set, and the classifier is retrained.

3. Fusing feature and updating model

3.1. Feature fusion

3.1.1. HOGE

HOGE (Histogram of Orientation Gradient Edge) is an effective representation of object appearance because it is insensitive to the change of illumination. We first pre-process the object by the guided filtering method [11], compute the edge (Fig.1b) of [12], and then reserve some pixels with greater than double the mean value. The orientation of the gradient edge of the object is quantified using N_h bins. In this paper, N_h is set 16.



(a) (b) (c) (d)

Fig. 1. Process of HOGE: an original image (a); an edge image (b); the gradient orientation corresponding to the edge image (c); and a HOGE with 16 bins (d)

3.1.2. Fusing feature

During the tracking, each feature is different in contribution of the different frames, the more the contribution is, the bigger the weight is. On the contrary, the less the contribution is, the smaller the weight is, so it is extremely important and crucial to compute the weight of each feature.

Let fe_i and $\mathrm{fe}_{\mathrm{fu}}$ represent the *i*-th feature and fusion feature of the object, respectively, the multiple features are fused by

$$\mathrm{fe}_{\mathrm{fu}} = \sum_{i=1}^{N_s} \eta_i \mathrm{fe}_i, \ \text{s.t.}, \ \sum_{i=1}^{N_s} \eta_i = 1, \tag{5}$$

where η_i denotes the coefficient of the *i-th* feature, which is the normalized value of the similarity measure between the *i*-th feature and the object model. In this paper, the adopted similarity measure is the MSSBRS introduced in Section 3.2 that is more discriminative than the common measure, such as the Bhattacharyya similarity and the BRS. The selected features are different in visible video and thermal video, for example, the selected features are grayscale and HOGE in thermal video and two features are selected from color and HOGE in visible vide .

3.2. MSSBRS

The BRD [13] (Bin-Ratio Dissimilarity) is a new similarity measure which considers the ratios between bin values of histograms, and The BRD is successful in the scene of classification. Nevertheless, it is prone to neglecting spatial information of the object. To improve the performance of distinguishing objects, we have introduced a modified BRD named the SBRS [14] which is better than the similarity Bhattacharyya and the BRS. We introduce a Modified Similarity SBRS (MSSBRS) which considers not only the spatial structure of the object but also the relation of the bins of the histogram of between the candidate object and the reference object.

Assume q_{ref} with second-order spatiogram [15] to represent the reference object, which is defined as $\boldsymbol{q}_{\text{ref}}(j) = [\boldsymbol{h}_j, \boldsymbol{m}_j, \boldsymbol{c}_j]$, where $\boldsymbol{h}_j$ is the j-th bin of the histogram, and $\boldsymbol{m}_j$, $\boldsymbol{c}_j$ denote the mean vector and covariance of the coordinates of the pixels corresponding to the j-th bin respectively. Meanwhile, the histogram of the edge needs to satisfy $\sum_{j=1}^{M} \boldsymbol{h}_j^2 = 1$, where M is the number of the bins. Assume $\boldsymbol{p}_{\text{cal}}$ with second-order spatiogram, which defines as $\boldsymbol{q}_{\text{cal}}(j) = [\boldsymbol{h}_j^{'}, \boldsymbol{m}_j^{'}, \boldsymbol{c}_j^{'}]$ to represent the candidate object. So the similarity $\rho(\boldsymbol{q}_{\text{ref}}, \boldsymbol{p}_{\text{cal}})$ between reference spatiogram $\boldsymbol{q}_{\text{ref}}$ and candidate spatiogram $\boldsymbol{p}_{\text{cal}}$ is defined as

$$\rho(\boldsymbol{q}_{\text{ref}}, \boldsymbol{p}_{\text{cal}}) = \sum_{j=1}^{M} \boldsymbol{\beta}_j d_j(\boldsymbol{h}_j, \boldsymbol{h}_j^{'}), \tag{6}$$

where $\boldsymbol{\beta}_j$ is the weight of the j-th bin similarity, if the spatiogram is a 0-th order, $\boldsymbol{\beta}_j$ is set to 1, and if it is a second-order spatiogram, $\boldsymbol{\beta}_j$ can be defined as $\boldsymbol{\beta}_j = a_0 N(\boldsymbol{m}_j; \boldsymbol{m}_j^{'}, k(\boldsymbol{c}_j + \boldsymbol{c}_j^{'}))$, in which the factor a_0 ensures the similarity measure that satisfies the condition $0 \leq \rho \leq 1$ and $\rho(\boldsymbol{h}, \boldsymbol{h}) = 1$; $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a Gaussian function with a mean vector $\boldsymbol{\mu}$ and a covariance matrix $\boldsymbol{\Sigma}$ of the coordinates of the pixels. The bigger the control factor k is, the smoother the weight $\boldsymbol{\beta}_j$ is; $d_j(\boldsymbol{h}_j, \boldsymbol{h}_j^{'})$ is a bin ratio similarity measure that improved the bin ratio dissimilarity [14], which is defined as

$$d_j(\boldsymbol{h}_j, \boldsymbol{h}_j^{'}) = \frac{\| \boldsymbol{h} + \boldsymbol{h}^{'} \|_2^2 \dfrac{\boldsymbol{h}_j \boldsymbol{h}_j^{'}}{(\boldsymbol{h}_j + \boldsymbol{h}_j^{'})^2}}{N_{\text{e}}}, \tag{7}$$

where N_{e} is the number of bins, and each pair of bins has at least one non-zero value in the two histograms $\boldsymbol{h}_j$, $\boldsymbol{h}_j^{'}$, and $\|.\|_2$ is a $\boldsymbol{L_2}$ norm.

Given the reference object is represented as a set of B histograms $\boldsymbol{q} = [\boldsymbol{q}_1, \boldsymbol{q}_2, ..., \boldsymbol{q}_B]$. Meanwhile, and the candidate object selected as the same size as the reference object is represented as a set of B histograms $\boldsymbol{p} = [\boldsymbol{p}_1, \boldsymbol{p}_2, ..., \boldsymbol{p}_B]$, where $\boldsymbol{q}_b$ and $\boldsymbol{p}_b$ are the *spatiogram* of the b-th *fragment* respectively. In this paper, the fragments of the object are consisted of the fragments with three levels under different scales, and three levels of each scales are made up of the top level, the

middle level and the bottom, in which the top level is the image with equally divided into 16 rectangle fragments, and the second level is the image with equally divided into 4 rectangle fragments, and the bottom level is the original image. So all the fragments of each scale are concatenated as $F=[F_1, F_b, ..., F_{21}]$; $\boldsymbol{q}$ is a reference histogram set corresponding fragments, and $\boldsymbol{p}$ is a candidate histogram set. The similarity measure between $\boldsymbol{q}$ and $\boldsymbol{p}$ is expressed as

$$\rho(\boldsymbol{p},\boldsymbol{q})=\sum_{i=1}^{B}\lambda_b\rho_b \quad \text{s.t.,}\ \sum_{b=1}^{B}\lambda_b=1, \tag{8}$$

where λ_b represents the normalized weight corresponding to the *b*-th spatiogram, which is defined as $\lambda_b=\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{d(\boldsymbol{p}_b,\boldsymbol{q}_b)}{2\sigma^2}}$ in which σ controls the normalized weight of the λ_b. The similarity measure of the *b*-th spatiogram between $\boldsymbol{q}$ and $\boldsymbol{p}$, ρ_b is expressed as

$$\rho_b=\sum_{j=1}^{M}\beta_j^b d_j^b(\boldsymbol{h}_j^b,\boldsymbol{h}_j^{'b}). \tag{9}$$

3.3. Updating the model by incremental PCA

During the tracking, the tracking results with fixed object model are prone to producing a drift due to the fact that object or background appearance often changes. It is necessary to update online the object model in order to alleviate drift. PCA has been successful in face recognition and object tracking [7]. However, when the size of the object is very large, the complexity of these algorithms is very high. The main reason is that PCA computes the feature values and the feature vectors of training images. In this section, we introduce to update the object model by PCA which is similar to the method in [7].

Given $\mathbf{ch}_1=[\mathrm{hi}_1, \mathrm{hi}_2, \cdots, \mathrm{hi}_{n_{in}}]$ and $\mathbf{ch}_2=[\mathrm{hi}_{n+1}, \mathrm{hi}_{n+2}, \cdots, \mathrm{hi}_{n_{\mathrm{in}}+m_{\mathrm{o}}}]$, where hi_i is an observation histogram with d bins, and m_{o}, n_{in} is the number of the training histograms $\mathbf{ch_1}$ and $\mathbf{ch_2}$, respectively; $\overrightarrow{\mathbf{ch}_1}$, $\sum_1$ is the mean vector and the scatter matrix S_{CH1} of $\mathbf{ch_1}$ respectively, and $\overrightarrow{\mathbf{ch}_2}$, $\sum_2$ is the mean vector and the Scatter matrix $\boldsymbol{S}_{\mathbf{ch}_2}$ of $\mathbf{ch_2}$ respectively; $\mathbf{ch}=[\mathbf{ch}_1\ \mathbf{ch}_2]$ is their concatenation, $\boldsymbol{S}_{\mathrm{CH}}$ is the scatter matrix of CH, which is expressed as

$$\boldsymbol{S}_{\mathrm{CH}}=\boldsymbol{S}_{\mathbf{ch}_1}+\boldsymbol{S}_{\mathbf{ch}_2}+\frac{n_{\mathrm{in}}m_{\mathrm{o}}}{n_{\mathrm{in}}+m_{\mathrm{o}}}(\overline{\mathbf{ch}_2}-\overline{\mathbf{ch}_1})(\overline{\mathbf{ch}_2}-\overline{\mathbf{ch}_1})^{\mathrm{T}}. \tag{10}$$

Its detailed proof can refer to [7]. After the decomposition of the incremental PCA, we select the corresponding vectors according to the first five big eigen values, and then we make them assemble the feature set. The reference histogram is approximated as

$$\boldsymbol{H}_b\approx\sum_{k=1}^{n_u}\boldsymbol{V}(:,k,b)^{\mathrm{T}}\,\mathrm{cih}(k,b)\boldsymbol{V}(:,k,b)+\bar{h}_b, \tag{11}$$

where $\boldsymbol{V}(:, k, b)$ in (11) is the *k*-th eigen vector of the *b*-th fragment in the scatter matrix $\mathbf{S}_{\mathrm{ch}}$; $\bar{\boldsymbol{h}}_b$ is the mean histogram of the *b*-th fragment; cih(*k*, *b*) is the *k*-th eigen-

values of the b-th fragment of the object; n_u is the number of the selected feature histogram (In this paper, we set n_u 5. In that we consider the time and space complexity and the performance of the tracking method. If $n_u > 5$, the time and space complexity is high, hence it affects the speed of the tracking method, on the contrary, the performance of the tracking method is affected). To further adaptive the change of the object, we also consider the initial histogram, therefore, the new object model is expressed as

$$\boldsymbol{H}_{0,b} = \gamma \boldsymbol{H}_{\text{init},b} + (1-\gamma)\boldsymbol{H}_{1,b}, \tag{12}$$

where γ is a learning rate， which is computed by the similarity between the current and the initial object. (We set from 0.6 up to 0.8 due to the fact that the appearance changes are prone to neglecting if is bigger than 0.8, on the contrary, the appearance changes are excessively considered if is smaller than 0.6.)

4. Object tracking based on online semi-SVM and adaptive fused feature

Self-training has been a widely used method in semi-supervised learning, in which a classifier is firstly trained with a small amount of labelled samples, and then the unlabelled samples are classified by the classifier. If the confident values of these unlabelled samples are high, we need to add these unlabelled samples into the training set as the label samples and update the classifier by retraining these label samples. Our method combines the incremental SVM with self-learning to track the object. To improve the drifting problem resulted from the self-learning, we need to update the object mode on line.

The larger the number the more it leads to degradation in tracking results, while the smaller also lead to unreliable results since the number of the labelled samples is important for training. In semi-supervised leaning, only a few samples are labelled. In this paper, we find that our method can perform well with only 15 labelled positive samples and 60 negative labelled samples. The positive samples are a histogram set including some feature histograms of the objects and the negative sample is also a histogram set of including some feature histograms of the areas which do not overlap with the objects. After the initialization, we will locate the object and update the classifier.

4.1. Locating the object and updating the classifier

Given a self-learning classifier and an unlabelled frame, we need to compute the location of the object within the frame. Specifically, the location of the unlabelled frame is classified by the classifier using the sliding window technology, and then the confidence map is generated. One common method of searching the maximum value of the confidence map is a gradient ascent algorithm which is known at the prior location of the object. However, this method suffers from the spatial constraint. To simplify the procedure, we only compute the global maximum value in the confidence map.

To adapt to the changes of the object or background, it is vital to update the classifier and the object model on line. The main problem is how to select online samples comprising of positive samples and negative samples, that is, we determine whether or not to add the current sample to training set and update the classifier. In this paper, we determine whether or not to add the current sample to the feature set according to its value of the global maximum in the confidence map, if the value is higher than the given threshold (threshold is selected manually and its value is 0.3 because of considering effective and accuracy of the classifier), the current sample is added and then the classifier is updated. The negative samples are selected according to whether or not adding positive sample. When the current object is added, we select the k highest peaks which do not overlap with the object in the confidence map in order to select the most important negative samples. However, the number of the samples can become very large in time, moreover, they make our method of studying the changes of the object or the background work much slower. To improve the tracking velocity, we maintain the number of training samples N_i (we set N_i = 15 in our experiments because of considering effective and accuracy of the classifier). If the number of the trained samples is larger than N_i, the old sample will be removed from the SVM by decreasing method.

4.2. Pseudo-code of our method

In this section, the detailed procedure of the presented algorithm will be given as the following:

Initialization:

For i=1:n_{in}

Step 1. Acquire the i-th labelled object either manually or by other object tracking method

Step 2. Compute the histogram of the intensity and the HOGE

Step 3. Fuse feature histogram through the Equation (5)

Step 4. Acquire a positive sample and desired negative samples

End

Train the classifier using incremental/decremental SVM

L=1;

Online self-tracking with adaptive update model

For i=n_{in}+1: n_{in}+N_{fr}

Step 1. Build a confidence map making use of the trained classifier

Step 2. Locate the object employing the maximum value of the confidence map

Step 3. If maximum value is bigger than threshold1 and similarity between the current object

(a) the current object is added to the positive sample set

(b) the images corresponding to the first four minimum value of the confidence map are added to the negative sample set

L=L+1

(c) update the SSSVM classifier

(d) the current object is added to the feature set

Else
Track the object using the basic particle filter
End
Step 4. If $L>20$
Remove old samples from the SVM by the decreasing SVM
Update the object model by (12) and retrain the classifier.
$L=1$;
End
End

5. Experimental results and discussion

To evaluate the performance of the proposed method, we perform our method in some visible and thermal video sequences. The visible sequences are selected from two vehicle sequences and the CAVIAR database[1], and the thermal sequences are selected from OTCBVS Benchmark Dataset[2]. In our experiments, two selected features are intensity histogram and HOGE, and then it fuses them by MSSBRS to represent the tracked object and local background. Of course, the other feature, such as optical flow can be easily selected. The selected features are intensity and HOGE for thermal sequences, and two features from *r*, *g*, *b*, HOGE are selected according to the method in [8] for visible sequences. The scale which may produce the maximum score was selected when scale changes are bound to ± 10% in each frame.

All the experiments are carried out with a core 4 Duo 2.9 GHz processor with 2 GB RAM under Matlab R2010b. To demonstrate the performance of our method, we compare our tracker with the ensemble tracking and the basic mean-shift tracking in some invisible and thermal sequences.

5.1. Results in visible sequences

The first experiment can be seen on a video sequence of a white car crossing the crossroads under heavy fog and heavy wind. The sequence is 150 frames long and the size of each image is 576×768 pixels. The sequence is challenging because the car can be easily affected by the heavy fog. Obviously, it is difficult to track the car accurately. In spite of this, our method is still able to track the car throughout the entire sequence. The ensemble tracker [5] is able to track the car for the first 100 frames, however the following frame is clear drift. The main reason is that the ensemble tracking method suffers from the fog, where our method can solve this problem. Fig. 2 shows the results of the ensemble tracker and our method, which contain the tracking results and the corresponding confidence map, and it can see that our confidence maps have a clear peak at the object center (Fig. 3), so it leads to more stable tracking.

[1] **http://i21www.ira.uka.de/image_sequences/**
http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/
[2] **http://www.cse.ohio-state.edu/OTCBVS-BENCH/bench.html**

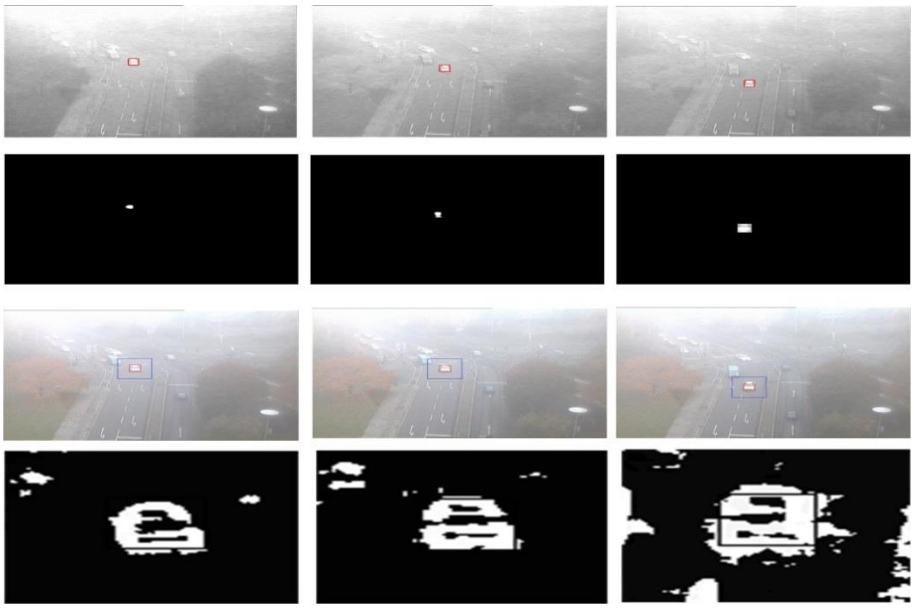


Fig. 2. Comparison of our method and the ensemble tracker method. The first two rows are the tracking results and the corresponding confidence maps of the proposed method, and later two rows are the tracking results and the corresponding confidence maps of the ensemble tracker

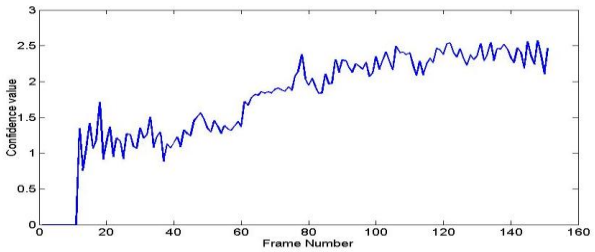


Fig. 3. Confidence value of the car under the heavy fog

5.2. Results in thermal sequences

Our method can track not only the object in visible sequences but also the object in thermal sequences. Here we show results of two experiments[3]; one is the boat in the sea and the other is the pedestrian across the big tree. The only modification is the feature space that is used to represent the object because the thermal object has few efficient cues, so the efficient features are selected as the intensity and the HOGE.

In the first thermal sequence, a boat at sea, which is over 800 frames is tracked, because the background of the sequence suffers from the clutter and it is represented using the mixture histogram which is fused between the intensity histogram with 16 bins and the HOGE with 16 bins. The tracking results of our method and the basic mean shift tracking method -are shown in Fig. 4. We observe that our method is able to track robustly the boat because we adopt the adaptive

[3] **http://www.cse.ohio-state.edu/OTCBVS-BENCH/bench.html** and **http://www.cs.technion.ac.il/~idol/**

object model, while the basic mean shift tracking method [16] is slight drift due to the fact that the thermal object does not provide enough information for tracking. Meanwhile, we also observe that the confidence map has a clear peak and helps the tracking.

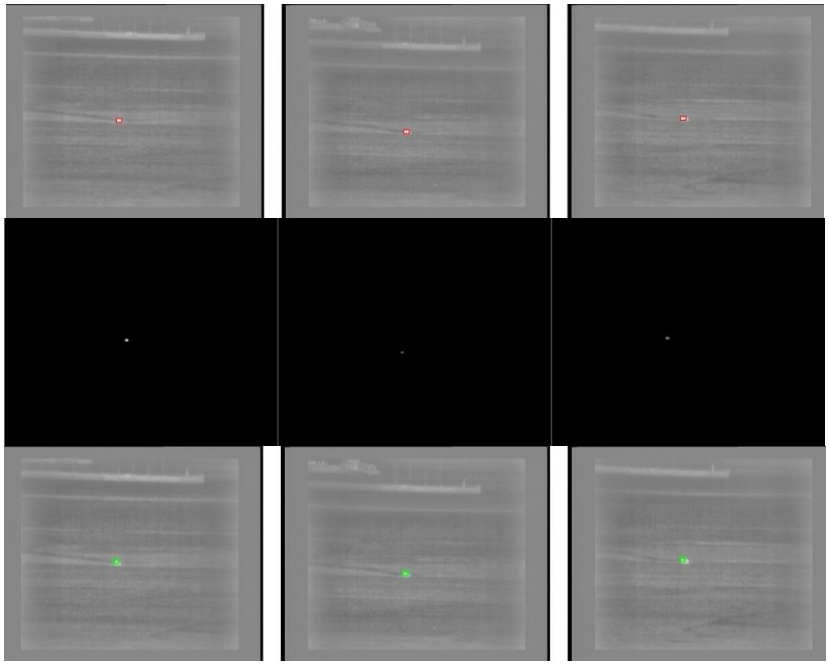
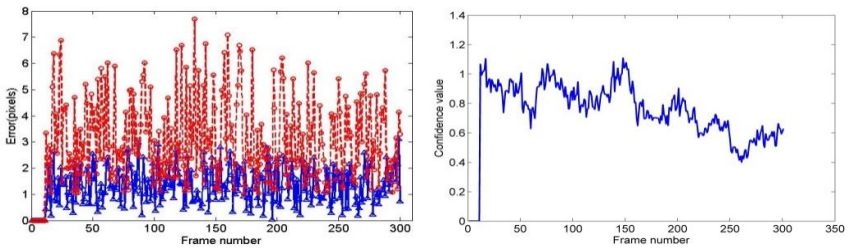


Fig. 4. Results of our algorithm and the basic mean shift algorithm on a thermal video of a boat. The top row is the results of our method, the middle row is the confidence map of our method, and the bottom row is the results of the basic mean shift algorithm. The frame number is 33,283, and 734 from the left to right respectively. Note that the confidence maps are smoother and have a clear peak at the object centre

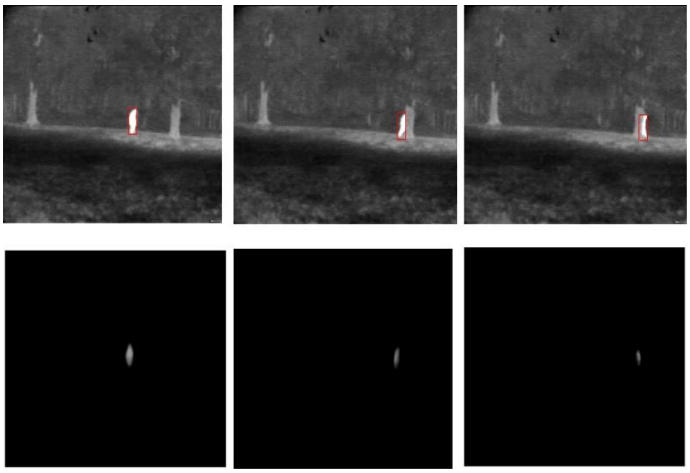
Fig. 5a shows the results of our method compared with the basic mean shift method on the boat sequence. As can be seen from the Fig. 5a, our method can track more stable and more consistent than the basic mean shift method. Furthermore, it is close to the ground truth. Fig 5b shows the confidence value of the tracked object in the first 300 frames.



(a) (b)

Fig. 5. The tracking error about our method and the basic mean shift method of the first 300 frames in the boat sequence, Note that solid blue is the tracking error of our method and dashed red is the tracking error of the basic mean shift algorithm (a); the confidence value of the first 300 frames (b)

So far, we only take into account the partial occlusions of the object. With respect to the complete occlusions, we can track the object using the basic particle filter tracker [17]. Specifically, as long as the confidence value is higher than the given threshold, the tracking position preserve unchanged until the next value is lower than the threshold, and then we begin with tracking the object via the basic particle filter tracker. In our experiments, we find that the object is occluded when the confidence value is less than 0.5, namely, we may set the threshold to 0.5. While tracking object, the confidence value is firstly computed by the trained classifier and then judge if it is above the threshold (that is to say, the object reappears if the threshold value is high, as is shown Fig. 6c. On the contrary ,the basic particle filter tracker will stop performing it if the confidence value of the current frame is above the threshold ,and then our method begin to track the object from the next frame . When the target is occluded, our method can accurately track the object than other tracking method, as is shown in Fig. 6. Fig. 7 shows the confidence value of all complete 151 frames, and finds that the confidence value between the frame 138 and the frame 143 is very low.



(a) (b) (c)

Fig. 6. Tracking results and the confidence map corresponding the frame number 93,136, and 144 of our tracker. Note that the confidence maps are smoother and have a clear peak at the object centre

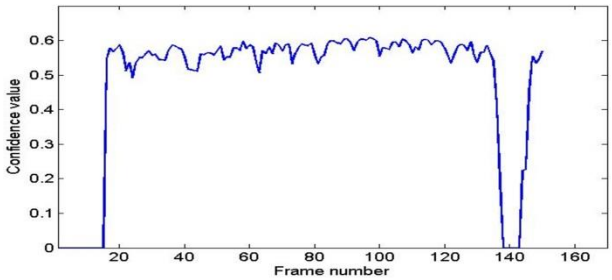


Fig. 7. The confidence value of all complete 151 frames. Note that the confidence value between the frame 138 and the frame 143 is very low

Fig. 8 shows the tracking results of additional sequences applying proposed our method.

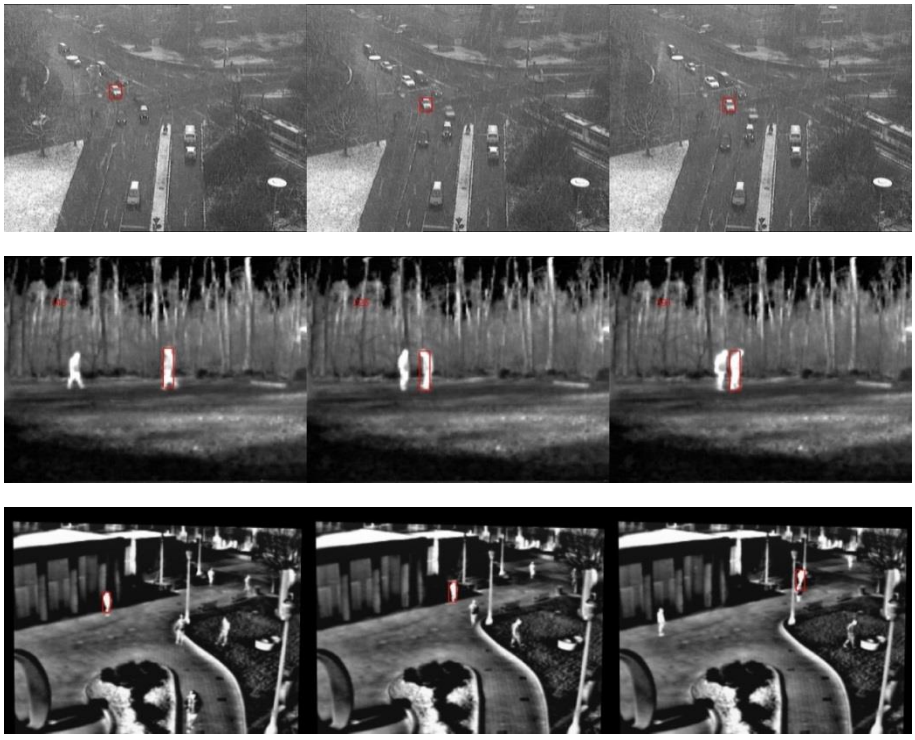


Fig. 8. Tracking results of our method under the visible sequence and the infrared sequences. The first row shows the results of our method on frames 114,211 and 226 in car under the heavy snow. The second row shows the results of our method on frames 143,225, and 391 from the OTCBVS sequence. The third row shows the results of our method on frames 22,124, and 259 from the OTCBVS sequence. Note that our method can robustly track the object

6. Conclusion

We treat object tracking as a binary classification and propose a robust tracking algorithm based on online semi-SVM with the self-training framework and adaptive fused features whose fused coefficients are computed by new similarity MSSBRS. The object model is updated using the incremental PCA. The proposed method only labels a small amount data to train the classifier during the initialization, and then the tracked object is classified according to the confidence value, the classifier is updated online by the self-training framework and new samples with high confident value. With regards to the complete occlusion, we use the basic particle filter algorithm to solve it. However, the object is represented by a fixed object scale, the proposed method is not able to deal with large changes in object scale. In future, we would like to make the proposed method adapt to these changes.

***Acknowledgments*:** This work was supported by the Research Foundation of Changzhou Institute of Technology (Grant No YN1208), the Natural Science Foundation of China (Grant No 61475027), Collaborative innovation fund of Jiangsu province (XYN1408) and the Research Foundation of Changzhou Institute of Modern Optical Technology (Grant No CZGY005).

References

1. Lu, Z., L. J. P. Van Der Maaten. Preserving Structure in Model-Free Tracking. – IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. **36**, 2014, No 4, pp. 756-769.
2. Wu, Y., M. Pei, M. Yang, J. Yuan, Y. Jia. Robust Discriminative Tracking via Landmark-Based Label Propagation. – IEEE Transactions on Image Processing, Vol. **24**, 2015, No 5, pp.1510-1523.
3. Tang, F., S. Brennan, Q. Zhao, H. Tao et al. Co-Tracking Using Semi-Supervised Support Sector Machines. – In: Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2007, pp. 1-8.
4. Avidan, S. Support Vector Tracking. – IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. **26**, 2004, No 8, pp. 1064-1072.
5. Avidan, S. Ensemble Tracking. – IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. **29**, 2007, No 2, pp. 261-271.
6. Grabner, C. L. A. H. H. Semi-Supervised Online Boosting for Robust Tracking. – In: Proc. of European Conference on Computer Vision, 2008, pp. 234-247.
7. Ross, D. A., J. Lim, R. S. Lin, M. H. Yang. Incremental Learning for Robust Visual Tracking. – International Journal of Computer Vision, Vol. **77**, 2008, pp.125-141.
8. Wang, J. Q., Y. Yagi. Integrating Color and Shape-Texture Features for Adaptive Real-Time Object Tracking. – IEEE Transactions on Image Processing, Vol. **17**, 2008, No 2, pp. 235-240.
9. Cauwenberghs, G., T. Poggio. Incremental and Decremental Support Vector Machine Learning. – In: Proc. of Neural Information Processing Systems, 2000, pp. 409-415.
10. Zhu, X. Semi-Supervised Learning Literature Survey. Computer Sciences Technical Report, 2006.
11. He, K., J. Sun, X. Tang. Guided Image Filter. – IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. **35**, 2013, No 6, pp. 397-1409.
12. Geusebroek, J. M., R. Van Den Boomgaard, A. W. M. Smeulders et.al. Color Invariance. – IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. **23**, 2001, No 12, pp. 1338-1350.
13. Xie, N., H. Ling, W. Hu et. al. Use Bin-Ratio Information for Category and Scene Classification. – In: Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, pp. 2313-2319.
14. Xiang, R., J. Li, X. Wang, F. Youjia. Probabilistic Tracking Method Based on the Spatial Bin-Ratio Information. – International Journal of Optomechatronics, Vol. **5**, 2011, No 4, pp. 360-377.
15. Birchfield, T., S. Rangarajan. Spatial Histograms for Region-Based Tracking. – ETRI Journal, Vol. **29**, 2007, pp. 697-699.
16. Comaniciu, D., V. Ramesh, P. Meer. Kernel-Based Object Tracking. – IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. **25**, 2003, No 5, pp. 564-577.
17. Gordon, N. J., D. J. Salmond, A. Smith. Novel-Approach to Nonlinear Non-Gaussian Bayesian State Estimation. – In: IEE Proc. of Radar and Signal Processing. Vol. **140**, 1993, pp. 107-113.