The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

Vehicle Tracking based on Co-Learning Particle Filter

Weilong Ye, Huaping Liu, Fuchun Sun and Meng Gao

Abstract— **In this paper, we propose a co-learning particle filter approach for vehicle tracking, which is very important for intelligent vehicle. The proposal distribution of the particle filter is a combination of an extra support vector machine (SVM) detector and the motion prior. Previous works focusing on how to online update the detector or the observation likelihood using the tracking results. These approaches belong to "self-learning" fashion and easily tend to drift. The major difference between the proposed approach and previous works is that the SVM detector and the likelihood function can be mutually updated in a co-learning manner. By adopting the co-learning technology, the unlabelled samples which are generated during tracking are utilized to progressively modify the SVM detector and update the observation likelihood; therefore the resulting tracker is more robust and effectively avoids the drift problem. Finally, the performance of the proposed approach is evaluated using extensive real visual tracking examples.**

I. INTRODUCTION

As a class of important mobile robots, the intelligent vehicles have been received more and more interests since they can be used to reduce the number of traffic accidents and increase the driver comfort. Among many functionalities an intelligent vehicle must perform, vehicle detection and tracking play important roles[21][2]. In fact, the intelligent vehicle must be able to detect and track preceding vehicles on its path in order to perform autonomous driving. Different classes of sensors, such as camera, radar, and acoustic, have been considered for sensing in this application. Due to the increasingly powerful computers and the less-expensive high-performance video cameras that have become available in the past few years, the use of computer vision technology as a sensor in driver-assistance systems become more common and has led to increased performance. Vision sensors can provide rich information about the vehicle's surroundings and also have the advantage over active sensors of not causing intervenience interference.

As to the tracking algorithm, earlier results were based on the famous Kalman filtering, which can obtain optimal solution in the case of linear dynamics and Gaussian noise. Unfortunately, very few practical visual tracking problems belong to this case. For nonlinear or non-Gaussian problems, it is impossible to evaluate the probability distribution analytically and many algorithms have been proposed to approximate them. The particle filter, also known as sequential Monte Carlo[7], or Condensation[11], is the most popular approach which recursively constructs the posterior probability distribution function of the state space using Monte Carlo integration. Currently, the particle filter has been extensively used in the field of location[20], fault detection[8] and SLAM[3][19] for robots.

To enhance the visual tracking ability of particle filter, [17] incorporated Adaboost detector into the framework of particle filter. This approach is very promising since it merges the advantages of detector and tracker. However, it only employs fixed Adaboost detector to construct the proposal distribution. Such models are trained using only appearance data available before tracking begins, which in practice limits the range of appearances that are modelled, and ignores the large volume of information that becomes available during tracking. To solve this problem, [1] and [9] developed online boosting classifier that selects features to discriminate the object from the background. These "classification-based tracking" approaches are so promising that many scholars combined them with the popular particle filter. For example, [23] embedded the feature selection procedure into the particle filter with the aid of existed "background" particles. [14] proposed a cascaded particle filter with discriminative observers of different lifespan. However, we notice that the model is updated in a totally self-learning manner. That is to say, the classifier which is trained (or updated) in the previous frame is used in current frame to evaluate possible regions. Then we select the so-called "positive" or "negative" samples for updating the classifier. Note that the "positive" or "negative" samples are not manually labelled but labelled by the previously trained classifier (This is an important difference between tracking and detection problems). Since tracking may introduce error, the labels may be noisy. Therefore these supervised approaches usually tend to "drift" since the error may be accumulated during the learning and tracking process. In fact, in many tracking problems, the labelled samples are given by an extra detector which only works in the first frame and therefore the number of labelled samples is very small, while the unlabelled samples, which can be selected from any frame, is enormous and easy to get. If we wish to update the classifier online, we should not ignore the unlabelled samples. This motivates us to use the popular semi-supervised learning approach[26].

Semi-supervised learning has received a lot of attentions over the past few years. The main motivation is that labelled samples are difficult to obtain, whereas unlabelled ones are easy. The task of semi-supervised learning algorithms is to utilize labelled samples in conjunction with their relationship to unlabelled data to design a classifier. Currently, different algorithms have been proposed for semi-supervised learning such as EM algorithm, co-training, tri-training, etc. For more details on semi-supervised learning, please see [26].

Weilong Ye, Huaping Liu and Fuchun Sun are with Department of Computer Science and Technology, Tsinghua University, State Key Laboratory of Intelligent Technology and Systems, TNLIST, Beijing, China. Meng Gao is with Shijiazhuang Railway Institute, Hebei, China.

Semi-supervised learning also finds extensive applications in robotics and automation fields, such as shadow detection[12] and visual guidance of mobile robots[15].

Though the semi-supervised learning achieves great successes, its application in tracking domain is still very rare. Recently, [22] utilized the co-training support vector machine (SVM) approach to design a semi-supervised tracker. In [10], a semi-supervised online boosting approach is used for tracking, which is a straightforward extension of the supervised online boosting approach[9]. [25] used co-training approach to update generative and discriminative model and incorporated this updating approach into the framework of particle filter. Both the works in [22] and [25] update the observation likelihood functions in a co-training manner.

In this paper, we propose a co-learning approach for vehicle tracking. We use an SVM detector and the motion prior to construct the proposal distribution of the particle filter. Differently from [22] and [25], at each instant, we use the current tracking result to update the detector, and use the current detection result to update the observation likelihood. Though in [22] and [25], the term "co-training" is used, in this paper, we prefer to use the term "co-learning" to describe our algorithm. The reason is that the original co-training approach is proposed for updating of two classifiers, while in this paper, only the SVM detector is in the form of classifier, and the observation likelihood is not in the form of classifier. To the best of the authors' knowledge, such a co-learning approach has not been proposed and the proposed approach can effectively avoid the drift phenomenon in adaptive tracking.

The remainder of this paper is organized as follows. In Section II, a brief introduction about particle filter is given. In Section III we describe the proposed co-learning particle filter. Section IV gives some experimental results. Finally, some conclusions are presented in Section V.

II. Brief review for particle filter

The task of tracking is to use the available measurement information to estimate the hidden state variables. Given the available observations $\mathbf{z}_{1:k-1} = \mathbf{z}_1, \mathbf{z}_2, \cdots, \mathbf{z}_{k-1}$ up to time instant $k-1$, the prediction stage utilizes the probabilistic system transition model $p(\mathbf{x}_k|\mathbf{x}_{k-1})$ to predict the posterior at time instant k as$p(\mathbf{x}_k|\mathbf{z}_{1:k-1}) = \int p(\mathbf{x}_k|\mathbf{x}_{k-1})p(\mathbf{x}_{k-1}|\mathbf{z}_{1:k-1})d\mathbf{x}_{k-1}$. At time instant k, the observation $\mathbf{z}_k$ is available, the state can be updated using $Bayes's$ rule $p(\mathbf{x}_k|\mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k-1})}{p(\mathbf{z}_k|\mathbf{z}_{1:k-1})}$, where $p(\mathbf{z}_k|\mathbf{x}_k)$ is described by the observation equation. To solve this problem, the particle filter approaches are proposed [7]. The kernel of particle filter is to recursively approximate the posterior distribution using a finite set of weighted samples. Each sample $\mathbf{x}_k^i$ represents one hypothetical state of the object, with a corresponding discrete sampling probability ω_k^i, which satisfies $\sum_{i=1}^N \omega_k^i = 1$. The posterior $p(\mathbf{x}_k|\mathbf{z}_{1:k})$ then can be approximated as $p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^N \omega_k^i \delta(\mathbf{x}_k - \mathbf{x}_k^i)$, where $\delta(\cdot)$ is Dirac function. Then the estimation of the state $\mathbf{x}_k$ can be obtained as $\hat{\mathbf{x}}_k = E_p[\mathbf{x}_k|\mathbf{z}_{1:k}] \approx \sum_{i=1}^N \omega_k^i \mathbf{x}_k^i$. The candidate samples $\{\mathbf{x}_k^i\}_{i=1,2,\cdots,N}$ are drawn from an importance distribution $q(\mathbf{x}_k|\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k})$ and the weight of the samples are $\omega_k^i = \omega_{k-1}^i \frac{p(\mathbf{z}_k|\mathbf{x}_k^i)p(\mathbf{x}_k^i|\mathbf{x}_{k-1}^i)}{q(\mathbf{x}_k|\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k})}$. The samples are re-sampled to generate an unweighed particle set according to their importance weights to avoid degeneracy. In the case of the bootstrap filter[7], $q(\mathbf{x}_k|\mathbf{x}_{1:k-1}, \mathbf{z}_{1:k}) = p(\mathbf{x}_k|\mathbf{x}_{k-1})$ and the weights become the observation likelihood $p(\mathbf{z}_k|\mathbf{x}_k)$.

III. Proposed particle filter

The proposed tracking approach uses a composite proposal distribution which can be represented as

$$q(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k) = \alpha q_{svm}(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{z}_k) + (1-\alpha)p(\mathbf{x}_k|\mathbf{x}_{k-1}) \quad (1)$$

where q_{svm}, which is dependent on the SVM detector, is a Gaussian distribution which will be discussed later. The parameter α can be set dynamically without affecting the convergence of the particle filter. When $\alpha = 0$, the proposed algorithm reduces to the conventional particle filter. By increasing α we place more importance on the SVM detections. In general cases, α can be set as 0.5.

The form of Eq.(1) is a little similar with the proposal distribution proposed in [17], except that in [17], an Adaboost detector used, while in this paper we use SVM detector. However, we point out that there exists an important difference between the work in [17] and ours. The novelty of our work is that the SVM detector and observation likelihood can be updated in a co-learning fashion, while in [17], both the Adaboost detector and likelihood function are fixed. Though recently there emerges a lot of approaches to online update the Adaboost detector during tracking period, most of them belong to the "self-learning" category, and to the best of our knowledge, there has no existing work in which the detector and the observation likelihood are mutually online learned. In the following we will introduce the implementations of SVM detector, observation likelihood and the co-learning approach.

A. SVM Detector

SVM is a popular machine learning algorithm for data classification due to its strong theoretical foundation and good generalization performance. Taking the sign of a linear discriminant function $f(\mathbf{s}) = \mathrm{w} \cdot \Phi(\mathbf{s}) + b$ learning from the training data $\{\mathbf{s}_i, y_i\}_{i=1}^M$, where y_i takes values in the set $\{-1, +1\}$, and M is the number of samples. SVM classifiers minimize the following objective function in feature space:

$$\min_{\mathrm{w},b,\xi_i} \frac{1}{2}\|\mathrm{w}\|^2 + C\sum_{i=1}^M \xi_i \quad (2)$$

subject to the constraints:

$$y_i(\mathrm{w} \cdot \Phi(\mathrm{s}) + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i \in 1, ..., M \quad (3)$$

where ξ_i is the slack variable, C is the tradeoff parameter between allowed error in the samples and the margin. By taking the Lagrangian of (2) and setting it to zero, we can express the original problem as the dual form

$$\min_{0 \leq \alpha \leq C} W = \frac{1}{2}\sum_{i,j=1}^M \alpha_i Q_{ij} \alpha_j - \sum_{i=1}^M \alpha_i + b\sum_{i=1}^M \alpha_i \quad (4)$$

where $Q_{ij} = y_i y_j \Phi(\mathbf{s}_i)\Phi(\mathbf{s}_j)$. The solution of dual parameters reduce to the Karush-Kuhn-Tucker (KKT) conditions:

$$g_i = \frac{\partial W}{\partial \alpha_i} = \sum_{i,j=1}^{M} Q_{ij}\alpha_j + y_i b - 1 \begin{cases} > 0 & \alpha_i = 0 \\ = 0 & 0 < \alpha_i < C \\ < 0 & \alpha_i = C \end{cases} \quad (5)$$

$$h = \frac{\partial W}{\partial b} = \sum_{i=1}^{M} y_i \alpha_i = 0 \quad (6)$$

Those samples with $g_i = 0$ are usually called support vectors, samples with $g_i < 0$ are called error vectors, the rest are called reserve vectors and exceed the margin ($g_i > 0$).

Given the recent success of Histogram of Oriented Gradient (HOG) feature in object detection [6], we adopt it in our classifier design. As shown in Fig.1(a), to incorporate spatial information into HOG, we use a 2×2 cell array to form the block. For each cell, the 9-bin histogram of the gradient magnitude at each orientation is computed. The concatenation of the HOG for 4 cells within one block forms a 36-dimensional vector, as shown in Fig.1(c). More details can be found in [6].

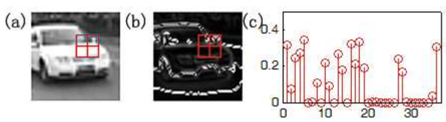

Fig. 1. HOG representation: (a) A block with 4 cells; (b) The gradient map; (c) The HOG of a block.

B. Observation Likelihood

Currently there are many observation likelihoods have been developed for detection and tracking. In this paper, to show the ability of co-learning, we use the most conventional feature: RGB histogram, since it achieves robustness against non-rigidity, rotation and partial occlusion[18][16]. It should be noted that any other appearance features can be easily incorporated into this framework. In our experiments, the histograms are typically calculated in the RGB space using $8 \times 8 \times 8 = 512$ bins.

The color-similarity measure is based on the similarity between the color histogram of a reference region and that of the image region in frame k represented by a sample $\mathbf{x}_k^i$. To estimate the proper weight for this sample during the measurement update step, we need the observation model $p_c(\mathbf{z}_k|\mathbf{x}_k = \mathbf{x}_k^i)$. This model can be obtained by the following equation

$$p_c(\mathbf{z}_k|\mathbf{x}_k = \mathbf{x}_k^i) \propto \exp\{-\lambda_c D^2(\mathbf{q}^*, \mathbf{q}_k(\mathbf{x}_k^i))\} \quad (7)$$

where $\lambda_c = 20$ in our experiments and $\mathbf{q}^*$ and $\mathbf{q}_k(\mathbf{x}_k^i)$ are the fixed reference color histogram and the color histogram extracted from the region defined by $\mathbf{x}_k^i$, respectively. The distance measure $D(\cdot,\cdot)$ is derived from the Bhattacharyya similarity coefficient and is defined as

$$D(\mathbf{q}^*, \mathbf{q}_k(\mathbf{x}_k^i)) = \{1 - \sum_{n=1}^{512} \sqrt{q^*(n) q_k(n; \mathbf{x}_k^i)}\}^{1/2} \quad (8)$$

More details can be found in [18] and [16]. Note that in [18] the reference histogram $\mathbf{q}^*$ is fixed while in [16] $\mathbf{q}^*$ can be updated using the tracking results. As we analyze in above, the updating approach proposed in [16] belongs to the fashion of "self-learning" and therefore easily tends to drift. In the following subsection the co-learning technology will be proposed to solve this problem.

C. Co-Learning Particle Filter

The original motivation to develop co-training came from the fact that labelled data is scarce, whereas unlabelled data is usually plenty and cheap to obtain. In conventional co-training algorithms[4][12], two classifiers are trained using two different feature sets on the initial labelled data. Then each classifier is deployed on the unlabelled data, and at each round, it chooses the example which it can label most confidently from each class, and adds it to the pool of labelled examples. This is carried out iteratively until a fixed number of rounds, or until all the originally unlabelled data is labelled. The main drawback of the original co-training[4] is the assumption of conditional independence, which requires the two feature sets be statistically independent. In most real world cases, this assumption is not likely to hold. Recently [13] demonstrated that even two closely related classifiers could be co-trained effectively.

The idea behind co-training is the following. If the two classifiers are trained using conditionally independent feature sets, when one classifier labels an example, it is seen as a random training example by the other classifier. In this case, the other classifier benefits from this added example. In this way, different views of the target concept may help achieve better combined classification accuracy, even though individual classifier accuracy may be much weaker.

In this paper, we slightly modify the co-training approach to the so-called "co-learning" approach since we update SVM detector and the reference color histogram template mutually. Before discussing the implementation of the co-learning, we should separately introduce how to update the SVM detector and the reference color histogram template.

First, we consider the online updating of SVM detector. Currently there have many online SVM approaches[5][24]. The online SVM classifier need to train incrementally on new data. In this paper, we use $\alpha - ISVM$ proposed in [24] to update the SVM detector. This algorithm fully utilizes the properties of support vector set, and accumulates the distribution knowledge of the sample space through the adjustable parameters. Note that in this updating the so-called new data is not the detection results produced by the SVM detector itself, but extracted from the tracking results. For notational simplicity, we denote the SVM detector at instant k as $\mathscr{C}_k$.

Then we discuss the updating of the reference template. We denote the reference template at instant k as $\mathscr{S}_k$. We can set the initial $\mathscr{S}_0 = \mathbf{q}^*$ and use

$$\mathscr{S}_k = \gamma \mathscr{S}_{k-1} + (1-\gamma)\mathscr{N}_k \quad (9)$$

to online update the reference template. In (9), γ is the updating parameter which can be set by designer. If γ is set to 1 then the reference template will keep fixed and therefore γ can be used to control the updating speed. $\mathscr{N}_k$ is the innovative information obtained at current instant k. In some popular updating approach such as [16], $\mathscr{N}_k$ is the RGB histogram extracted from the current tracking box. This is a "self-learning" approach and easily tends to drift. In our approach, $\mathscr{N}_k$ is extracted from the detection results which is given by the SVM detector $\mathscr{C}_{k-1}$. If there are more than one detection results, we choose the most confident one to use.

It should be noted that co-learning is just a learning approach, but not a classification approach. Therefore how to integrate the two classifiers is another problem. In [22] and [25], some heuristical integration approaches are used for tracking. In this paper, we use one detector and one RGB template for co-learning, where detector is used to construct the proposal distribution and RGB template is used to evaluate the likelihood of particles. Obviously the detector and the RGB template play different roles and work in different states; therefore no extra fusion approach is needed.

Finally, the co-learning module is integrated into the framework of particle filter and forms the co-learning particle filter, which is summarized in Algorithm 1. At each time instant, we maintain a set of weighted particle, an SVM detector, and a reference color histogram template. All of them will be updated during the tracking period.

Algorithm 1 `Co-learning Particle Filter Tracking Algorithm`

Given: $\{\mathbf{x}_{k-1}^{(n)}, \omega_{k-1}^{(n)}\}_{n=1}^N$, SVM classifier $\mathscr{C}_{k-1}$, RGB Template $\mathscr{S}_{k-1}$
Output: $\{\mathbf{x}_{k}^{(n)}, \omega_{k}^{(n)}\}_{n=1}^N$, SVM classifier $\mathscr{C}_{k}$, RGB Template $\mathscr{S}_k$

Object Tracking
Re-sampling:
– Resample $\{\mathbf{x}_{k-1}^{(n)}, \omega_{k-1}^{(n)}\}_{n=1}^N$ to get $\{\bar{\mathbf{x}}_{k-1}^{(n)}, 1/N\}_{n=1}^N$
Prediction:
– For $n = 1, 2, \cdots, N$, draw predicted particles $\mathbf{x}_k^{(n)}$ from the proposal distribution

$$\alpha q_{svm}(\mathbf{x}_k|\mathbf{x}_{k-1} = \bar{\mathbf{x}}_{k-1}^n, \mathbf{z}_k) + (1-\alpha)p(\mathbf{x}_k|\mathbf{x}_{k-1} = \bar{\mathbf{x}}_{k-1}^n) \quad (10)$$

Calculate the weights:
– For $n = 1, 2, \cdots, N$, calculate the weights of the predicted particles

$$\omega_k^{(n)} \propto \exp\{-\lambda_c D^2(\mathscr{S}_{k-1}, \mathbf{q}_k(\mathbf{x}_k^i))\} \quad (11)$$

Get the expected results: $\hat{\mathbf{x}}_k = \sum_{n=1}^N \omega_k^{(n)} \mathbf{x}_k^{(n)}$
Online Updating
Use the detection results of SVM detector to extract the RGB feature and use it to update $\mathscr{S}_{k-1}$ to $\mathscr{S}_k$
Use the tracking results of $\hat{\mathbf{x}}_k$ to extract the HoG feature and use it to update $\mathscr{C}_{k-1}$ to $\mathscr{C}_k$

Fig. 2. Data collection platform. Both images were captured in Tsinghua University campus.

IV. Experimental results

The developed algorithm has been tested in a number of different situations. In this section, we will give the descriptions of the data collection and experimental results.

A. Data Collection

In the experiments, we use CCD camera to collect data on practical roads. The camera is mounted on two independent platforms (see Fig.2). The left in Fig.2 is THMR-V, which is the intelligent vehicle developed by our laboratory. The right one in Fig.2 is developed for Shijiazhuang Railway Institute. The collected images are in 320×240 resolution. We collected a lot of data under different weather conditions and different scenarios. In the following we will give some experimental results.

B. Results

To verify the effectiveness of the proposed approach, we compare it with other three algorithms, of which implementations are detailed as follows:

(1)"**PF**": This approach is a conventional implementation of particle filter. That is to say, we set the parameter α in (1) to be zero and therefore no detector is incorporated into the tracker. In addition, the reference histogram is never updated. This approach is little similar to the approach in [18].

(2)"**PF-U**": This approach is the same as **PF** except that the reference histogram is updated using the current tracking results. The update parameter γ in (9) is set as 0.9, and the innovative term $\mathscr{N}_k$ in (9) is extracted from the tracking results. This approach is little similar to the approach in [16].

(3)"**SVM-PF**": This approach is the same as the proposed approach except that the SVM detector and the reference histogram are never updated. This approach is little similar to the approach in [17].

Finally, for a fair comparison, in our proposed approach, the updating parameter γ is also set to 0.9.

For all of the experiments, the state of the particle filter is defined as $\mathbf{x}_k = [x_k, y_k, s_k]$, where x_k, y_k indicate the location of the object, s_k the scale. The dynamics of the objects are assumed to be a random walking model, which can be represented as $\mathbf{x}_k = \mathbf{x}_{k-1} + \mathbf{v}_k$, where $\mathbf{v}_k$ is a multivariate zero-mean Gaussian random variable. Its variances are set by $[\sigma_x, \sigma_y, \sigma_s] = [5, 5, 0.1]$. For each particle filter, we assign 100 samples.

For the design of initial SVM detector, we collected 450 positive samples including different vehicles and 800 negative samples for training. The obtained SVM detector has 306 support vectors. For **SVM-PF**, this SVM detector will be used during the whole tracking period, while in our approach, this SVM detector can be updated during the tracking period.

In the first scenario, we try to track a oncoming car which faces to us. The car is first detected at a long distance, during the tracking period, the scale of it will become larger and larger. Due to the small scale in the initial frame, there has very few features can be extracted from it. Therefore, the conventional tracker easily loses it. In Figs.3-4 we give some representative examples. It is obvious that **PF** which uses fixed reference histogram rapidly loses the target. **PF-U**, which is equipped with updating ability, performs a little better but still not satisfactory since it only locks a small part of the target at Frame 47. **SVM-PF** is also not satisfactory since it uses fixed detector and reference histogram. Our approach, of which results are shown in the right columns, shows excellent performances during the tracking period. This is not difficult to understand since the detector and the reference histogram are updated each other. Though only a few features can be extracted from the initial frames, the co-learning approach can make the tracker robust to the change of the scale.

Then we will show the robustness to the influences produced by shadows. In scenario 2, a preceding car which is passing the region under a bridge is tracked. The bridge brings large shadows and strongly changes the color histogram of the car. From Figs.5-6 we can see that all performances of **PF**, **PF-U** and **SVM-PF** are strongly influenced by the shadows of the bridge, while our approach is more robust. After the car having passed the region under bridge, **PF** makes a little recover since the color histogram of the car comes back. However, since **PF-U** updates the reference histogram using the tracking results, the reference histogram is wrongly updated and therefore **PF-U** performs worse (see the second column of Fig.6). This is an obvious disadvantage of the "self-learning".

In Figs.7-8 we give another scenario which also admits large shadow and observe similar results. More extensive experimental results are omitted due to the space.

Another key point should be indicated is that the selection of parameter γ in **PF-U** is very important. If γ is too small, the updating speed will be fast and the reference histogram easily tends to drift. After our extensive tests, the value $\gamma = 0.9$ is relatively suitable for our applications. This represents a moderate updating. On the other hand, our approach is not so sensitive to the parameter γ, since the updating is determined by the detection results, but not the tracker itself. In fact, we make some other tests such as $\gamma = 0.8, 0.7, 0.6$ and 0.5 and observe similar tracking results. Therefore the proposed co-learning approach is rather robust to the updating parameter.

V. Conclusions

In this paper, we propose a co-learning particle filter. The major novelty is that the SVM detector and the likelihood function can be mutually updated in a co-learning manner. By adopting the co-learning technology, the unlabelled samples are utilized to progressively modify the SVM detector and update the reference color histogram template; therefore the resulting tracker effectively avoids the drift problem.

VI. Acknowledgements

This work was jointly supported by the National Science Fund for Distinguished Young Scholars (Grant No. 60625304), the National Key Project for Basic Research of China (Grants No. G2007CB311003, 2009CB724002), and National High-tech Research and Development Program (2007AA04Z232).

References

[1] S. Avidan, Ensemble tracking, *PAMI*, vol.29, no.2, 2007, pp.261-271
[2] C. R. Baker, J. M. Dolan, Traffic interaction in the urban challenge: Putting boss on its best behavior, *IROS*, 2008, pp.1752-1758
[3] J. L. Blanco, J. A. Fernandez, J. Gonzalez, An entropy-based measurement of certainty in Rao-Balckwelized particle filter mapping, *IROS*, 2006, pp.3550-3555
[4] A. Blum, T. Mitchell, Combining labeled and unlabeled data with co-training, *ACCLT*, 1998, pp.92-100
[5] G. Cauwenberghs, T. Poggio, Incremental and decremental support vector machine learning, *NIPS*, 2000, pp.409-415
[6] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, *CVPR*, 2005, pp.886-893
[7] A. Doucet, N. de Freitas, N. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, New York, 2001
[8] Z. Duan, Z. Cai, J. Yu, Adaptive particle filter for unknown fault detection of wheeled mobile robots, *IROS*, 2006, pp.1312-1315
[9] H. Grabner, and H. Bischof, On-line boosting and vision, *CVPR*, 2006, pp.260-267
[10] H. Grabner, C. Leistner, and H. Bischof, Semi-supervised online boosting for robust tracking, *ECCV*, 2008, pp.234-247
[11] M. Isard, A. Blake, Condensation - conditional desity propagation for visual tracking, *IJCV*, vol.29, no.1, 1998, pp.5-28
[12] A. J. Joshi, N. Papanikolopoulos, Learning of moving cast shadows for dynamic enviroments, *ICRA*, 2008, pp.987-992
[13] A. Levin, P. Viola, Y. Freund, Unsupervised improvement of visual detectors using co-training. *ICCV*, 2003, pp.626-633
[14] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade, Tracking in low frame rate video: A cascade particle filter with discriminative observers of difierent lifespans, *CVPR*, 2007, pp.1-8
[15] M. Mahdaviani, N. Freitas, B. Fraser, and F. Hamze, Fast computational methods for visually guided robots, *ICRA*, 2005, pp.138-143
[16] K. Nummiaro, E. K. Meierb, L. V. Gool, An adaptive color-based particle filter, *IVC*, vol.21, 2003, pp.99-110
[17] K. Okuma, A. Taleghani, N. Freitas, J. J. Little, and D. G. Lowe, A boosted particle filter: Multitarget detection and tracking, *ECCV*, 2004, pp.1-12
[18] P. Perez, C. Hue, J. Vermaak, M. Gangnet, Color-based probabilistic tracking, *ECCV*, 2002, pp.661-675
[19] K. Qian, X. Ma, X. Dai, Simultaneous robot localization and person tracking using Rao-Blackwellised particle filters with multi-modal sensors, *IROS*, 2008, pp.3452-3457
[20] D. Schulz, D. Fox, Bayesian color estimation for adaptive vision-based robot localization, *IROS*, 2004, pp.1884-1889
[21] Z. Sun, G. Bebis, R. Miller, On-road vehicle detection: A review, *PAMI*, vol.28, no.5, 2006, pp.694-711

Fig. 3. Scenario 1, Frame 25. From left to right: **PF**, **PF-U**, **SVM-PF**, **Proposed approach**.

Fig. 4. Scenario 1, Frame 47. From left to right: **PF**, **PF-U**, **SVM-PF**, **Proposed approach**.

Fig. 5. Scenario 2, Frame 143. From left to right: **PF**, **PF-U**, **SVM-PF**, **Proposed approach**.

Fig. 6. Scenario 2, Frame 180. From left to right: **PF**, **PF-U**, **SVM-PF**, **Proposed approach**.

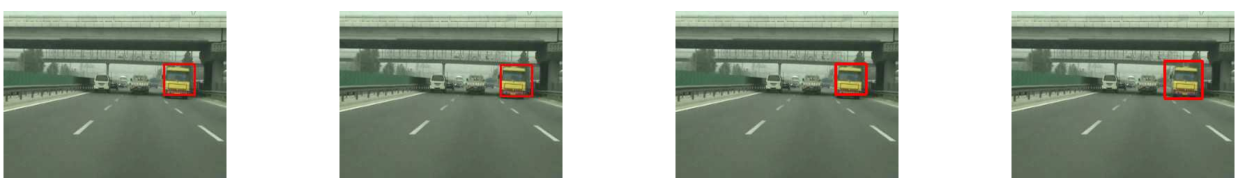

Fig. 7. Scenario 3, Frame 70. From left to right: **PF**, **PF-U**, **SVM-PF**, **Proposed approach**.

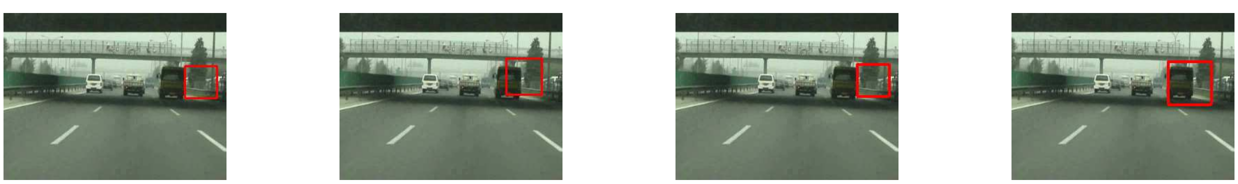

Fig. 8. Scenario 3, Frame 120. From left to right: **PF**, **PF-U**, **SVM-PF**, **Proposed approach**.

[22] F. Tang, S. Brennan, Q. Zhao, and H. Tao, Co-tracking using semi-supervised support vector machines, *ICCV*, 2007, pp.1-8

[23] J. Wang, X. Chen, and W. Gao, Online selecting discriminative tracking features using particle filter, *CVPR*, 2005, pp.1037-1042

[24] R. Xiao, J. Wang, Z. Sun, F. Zhang, An apporach to incremental SVM learning algorithm, *J. of Nanjing University*, vol.38, no.2, 2002, pp.152-157

[25] Q. Yu, T. B. Dinh, G. Medioni, Online tracking and reacquisition using co-trained generative and discriminative trackers, *ECCV*, 2008, pp.1-14

[26] X. Zhu, Semi-supervised learning literature survey, *Technical Report*, Computer Sciences, University of Wisconsin-Madison, 2005