Adaptive NormalHedge for robust visual tracking

Zhang, S., Zhou, H., Yao, H., Zhang, Y., Wang, K., & Zhang, J. (2015). Adaptive NormalHedge for robust visual tracking. Signal Processing, 110, 132-142. DOI: 10.1016/j.sigpro.2014.08.027

Published in:
Signal Processing

Document Version:
Peer reviewed version

Adaptive NormalHedge for Robust Visual Tracking

Shengping Zhang[a,*], Huiyu Zhou[b], Hongxun Yao[a], Yanhao Zhang[a], Kuanquan Wang[a], Jun Zhang[c]

[a]*School of Computer Science and Technology, Harbin Institute of Technology, China*
[b]*Institute of Electronics, Communications and Information Technology, Queen's University of Belfast, United Kingdom*
[c]*School of Computer Science and Information, Hefei University of Technology, China*

Abstract

In this paper, we propose a novel visual tracking framework, based on a decision-theoretic online learning algorithm namely NormalHedge. To make NormalHedge more robust against noise, we propose an adaptive NormalHedge algorithm, which exploits the historic information of each expert to perform more accurate prediction than the standard NormalHedge. Technically, we use a set of weighted experts to predict the state of the target to be tracked over time. The weight of each expert is online learned by pushing the cumulative regret of the learner towards that of the expert. Our simulation experiments demonstrate the effectiveness of the proposed adaptive NormalHedge, compared to the standard NormalHedge method. Furthermore, the experimental results of several challenging video sequences show that the proposed tracking method outperforms several state-of-the-art methods.

Keywords: Visual tracking, decision-theoretic online learning, particle filter, appearance changes

1. Introduction

Visual tracking is an overwhelming research topic in computer vision due to its wide applications such as intelligent video surveillance (Zhou et al., 2009a; Yu et al., 2013), video content analysis (Gao et al., 2011, 2012b) and human-robot interaction (Gao et al., 2012a; Yu et al., 2012). In the past few decades, a large number of visual tracking algorithms have been proposed (Zhou et al., 2008, 2009a,b; Zhang et al., 2012, 2013b,a), among which particle filter (Isard and Blake, 1996) and its variants (Pérez et al., 2002; Ross et al., 2007; Zhou et al., 2008) have attracted increasing interests because of their tractability and flexibility. Although they provide an appealing framework for visual tracking, the tracking performance becomes significantly poor when there are illumination changes, partial occlusion, pose changes and background clusters in complex scenes.

The main reason why particle filter based tracking methods tend to fail in complex scenes is that particle filter is in fact a generative framework which is very sensitive to model mismatches. Let $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$ be the hidden state and the available measurement of the tracked target at time t. The purpose of particle filter is to estimate the state $\boldsymbol{x}_t$ given all the available measurements $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_t\}$ up to time t. Particle filter based tracking method uses a set of weighted particles to approximate the posterior distribution of the state. In practice, the weight is computed as the observation likelihood defined by a measurement process $p(\boldsymbol{y}_t|\boldsymbol{x}_t)$ which models how the observation is generated from the hidden state. The tracking result is either the particle with the highest weight or the average weight of all the particles. The problem with such a generative framework is that it is fairly difficult to precisely determine the measurement process. Using a model which is different from the one of generating the observation may lead to poor tracking performance.

Decision-theoretic online learning (DTOL) is a research topic that allows us to study how to dynamically allocate resources among a number of experts. It was first introduced in Freund and Schapire (1997), where the Hedge algorithm was proposed to solve the learning problem. In contrast to traditional generative frameworks, Hedge algorithm uses a set of experts to explain the observations regardless of how the observations are generated. In other words, Hedge does not rely on the measurement process. The quality of the "explanation" is defined by a loss function, which depends only on the measurement rather than the hidden state. The resource assigned to each expert depends on the cumulative loss of this expert and a learning rate parameter. Although the Hedge algorithm is not sensitive to model mismatches, the best learning rate parameter cannot be obtained at all times, especially when the number of experts is large. That is to say, the Hedge algorithm cannot ensure the best predication in various applications. Recently, a novel and completely parameter-free algorithm for DTOL, called NormalHedge, was proposed in Chaudhuri et al. (2009) in order to overcome the drawback of the original Hedge algorithm. NormalHedge has a potential function for each expert and computes the expert weights based on the derivative of its potential. The NormalHedge algo-

*Corresponding author
Email address: `s.zhang@hit.edu.cn` (Shengping Zhang)

rithm is straightforward and easy to implement, which makes it suitable for real-time applications such as visual tracking. Chaudhuri et al. (2010) used NormalHedge to estimate hidden states in a continuous state space with noisy observations. Their simulation experiments, in the one-dimensional state space, showed that their algorithm vastly outperformed the standard Bayesian algorithm.

Chaudhuri *et al.*'s work indicates the possibility of applying NormalHedge to visual tracking. However, we observed that directly applying NormalHedge to practical visual tracking has several constraints. Firstly, it is unclear how to exploit historic information of each expert before making a decision at the current time. The algorithmic work of Chaudhuri et al. (2010) was only applied to an one-dimensional state space in which the true state remained stationary or changed at a constant velocity. In the meantime, a fixed percentage factor was used, which controls how much the historic information of each expert will be used to compute its cumulative regret. However, in practice, due to the difference among different experts, using a fixed percentage factor for all the experts is irrational and likely ends up with poor tracking performance. Secondly, determining the loss of each expert during visual tracking is challenging. In Chaudhuri et al. (2010), the negative sum of the scores for each expert was used. As a result, it is required to specify each candidate's position in the entire state space, which is time consuming. Finally, it is not clearly defined yet how the experts can be transferred from the current instance to the next one. In other words, the transition of the experts between two consecutive instances needs to be carefully re-visited.

In order to achieve robust visual tracking in complex scenes, in this paper, we propose a novel tracking framework based on an adaptive NormalHedge algorithm, which uses a set of weighted experts to predict the state of the target to be tracked. The major concept of the proposed method is an adaptive mechanism that determines how much historic information should be used in the estimation of the state at the current time. In particular, we compute the percentage factor of each expert according to its current loss and the learner's expected loss. This makes more sense than using the same percentage factor for all the experts (Chaudhuri et al., 2010). In addition, in order to characterize the loss of each expert, we define the loss of each expert as the negative similarity measure between the expert and the target template. Finally, we propagate the experts obtained at the previous time to the current time using a second-order autoregressive dynamic model. The contributions of this paper can be summarized as the following three-folds: **1)** We proposed an improved decision-theoretic online learning algorithm, which is more robust to noise. **2)** We proposed a tracking framework based on the proposed learning algorithm. **3)** Experiments on both simulation and video data validate the effectiveness of the proposed tracking method.

The rest of the paper is organized as follows. We first review the related work on visual tracking in section 2. Then we introduce the proposed adaptive NormalHedge algorithm in section 3. Section 4 shows the application of the proposed adaptive NormalHedge algorithm in visual tracking. Experimental results on simulation data and challenging video sequences are reported in section 5. Finally, we conclude this paper in section 6.

2. Related work

In the literature, a large number of tracking algorithms have been proposed, which focus on the development of effective appearance models and inference methods. In this section, in order to clarify the motivation of promoting the proposed method, we summarize various appearance models and inference methods reported in the literature.

2.1. Appearance modeling

Appearance modeling is to represent the target to be tracked using the information extracted from the image region outlined by the target. In the literature, many features have been used to represent the target such as color (Bradski, 1998), shape (Isard and Blake, 1996), texture (Shahrokni et al., 2005) and kinematic features (Castellini et al., 2011). For a number of tracking applications, color is an optimal choice because of its descriptive power and the fact that color information is readily accessible in the image. Color histogram has been commonly used in visual tracking (Pérez et al., 2002; Comaniciu et al., 2003) due to its simplicity, efficiency and robustness to rotation and scaling. However, color histogram cannot be used to model the spatial relationship between two pixels in an image. Adding spatial information into color histogram may improve its discriminative power. Along this line, color spatiograms (Birchfield and Sriram, 2005) and correlograms (Zhao and Tao, 2005) have been employed in visual tracking. These methods model the target's appearance from a global perspective. When the target is locally occluded, their representation ability will significantly degrade. In order to overcome this shortcoming, some part-based appearance models (Fieguth and Terzopoulos, 1997; Smith and Gatica-Perez, 2004; Adam et al., 2006) have been proposed. For example, in Adam et al. (2006), a target candidate was divided into multiple patches and each one was represented by a color histogram. The robustness of this research work to occlusions can be achieved by integrating the vote maps of multiple patches.

Appearance models mentioned above are usually fixed before the tracking begins, which makes it hard to deal with the target's appearance variations arising during the tracking. In Jepson et al. (2003), Jepson proposed a framework to adaptively learning the appearance model for visual tracking. The appearance model involved a mixture of stable image structures, learned over a long time sequence. This model can be adapted to slowly changing appearance. In Collins et al. (2005), an online feature selection method was proposed to select features that discriminate the target from its background. The feature selection was done online when new observations were available. The selected features adapted to the environmental changes and also obtained superior tracking results, even though the contrast between the target and its background was poor. In Ross et al. (2007), a tracking method was introduced, which incrementally learned a low-dimensional subspace representation, and efficiently adapted to the appearance changes of the target. When the target experiences significant changes of pose, scale and illumination, this system still reasonably tracked the target. Recently, sparse coding was widely used for appearance modeling. In Wang et al. (2012), each target candidate was divided into a set of patches and each patch was sparsely represented by a set of target templates and identity basis functions. The representation coefficients of all the patches were concatenated to form the final feature representation. In Jia et al. (2012), with a fixed spatial layout, local patches sampled from different templates were combined together as basis functions to code the patches sampled from each target candidate. The final feature representation was obtained using an alignment pooling operator which reserved the structural relationship between two local codes.

2.2. *Inference methods*

Given the appearance model of the target template, how can one infer the target's state in the current frame? Existing solutions can be roughly classified into two categories: direct optimization and probabilistic approximations. The first class takes a direct optimization approach, where iterative gradient based search (Zhao et al., 2007; Comaniciu et al., 2000) or linear program (Hager et al., 2004; Wu and Fan, 2009) are used to obtain the tracks. The advantage of this direct optimization is its efficiency with the implementation of modern nonlinear programs (Zhu et al., 1997). These approaches work well when certain assumptions hold or the computational resource is well constrained. However, the performance of direct optimization is not stable in complex scenes. For example, once the tracker fails, it cannot recover and then re-track the target. The second class takes a "hypothesis generation" and "observation verification" approach by fusing the probabilistic information. The representative methods include Kalman filter (Gutman and Velger, 1990) and particle filter (Isard and Blake, 1998; Pérez et al., 2002; Hess and Fern, 2009). Both Kalman filter and particle filter are based on a generative framework that relies on the used measurement process. However, in practice, it is difficult to precisely determine the measurement process. Therefore, they will achieve poor performance when a model different from the one of generating the observation is used.

In recent years, some novel tracking frameworks were proposed with impressive tracking performances. For example, tracking by detection methods (Grabner and Bischof, 2006) formulated visual tracking as a detection problem that detects whether the target appears or not within each candidate region. More sophisticated machine learning technologies such as semi-supervised learning (Grabner and Leistner, 2008) and multiple instance learning (Babenko et al., 2009) were also used in visual tracking. Very recently, novel learning methods (Yu et al., 2014c,b; Zhang et al., 2014; Yu et al., 2014a) attract increasing interests in computer vision. In particular, sparse coding based methods (Mei and Ling, 2009; Mei et al., 2011; Zhang et al., 2013b, 2012, 2013a) achieve desired performance compared to traditional methods. For example, in Mei and Ling (2009), a tracking method based on sparse representation was proposed, where each target candidate is sparsely represented by a set of target templates and identity basis. The representation coefficients were obtained using ℓ_1-minimization. The candidate with the smallest error when reconstructing it only using the target templates and the associated coefficients was chosen as the tracking result. This method is very time-consuming because ℓ_1-minimization was solved for each candidate.

3. Adaptive NormalHedge algorithm

In this section, we first describe the decision-theoretic online learning problem and the NormalHedge algorithm (Chaudhuri et al., 2009). Then we introduce in details the proposed adaptive NormalHedge algorithm. For convenience, Table 1 lists important notations used in the rest of this paper.

Table 1: Important notations used in this paper and their descriptions.

Notation	Description
ℓ_t^i	loss of expert i at time t
ℓ_t^A	expected loss under a distribution at time t
r_t^i	instantaneous regret of expert i at time t
R_t^i	cumulative regret of expert i in the first t times
$\boldsymbol{x}_t$	state of the tracked target at time t
$\boldsymbol{y}_t$	observation at time t
ω_t^i	weight of particle i at time t

3.1. *DTOL and NormalHedge algorithm*

Decision-theoretic framework for online learning (DTOL) is formulated in such a way: At time t, a learner has access to a set of N experts and maintains a distribution w_t^i over experts $1, 2, \ldots, N$. Each expert incurs a loss ℓ_t^i, and the learner's expected loss under this distribution is

$$\ell_t^A = \sum_{i=1}^{N} w_t^i \ell_t^i \tag{1}$$

The goal of the learner is to maintain a distribution over experts such that its cumulative loss over time is low, compared to the cumulative loss of the expert with the lowest cumulative loss. That is, the learner attempts to minimize its net loss

$$\sum_{\tau=1}^{t} \ell_\tau^A - \min_i \sum_{\tau=1}^{t} \ell_\tau^i$$

Starting with the seminal work of Littlestone and Warmuth (1994), the DTOL has been well-studied in the literature (Cesa-Bianchi et al., 1997; Freund and Schapire, 1997; Cesa-Bianchi and Lugosi, 2006). The classical solution to DTOL, the Hedge(β) algorithm (Freund and Schapire, 1997), updates the distribution using the multiplicative rule

$$w_{t+1}^i = w_t^i \cdot U_\beta(\ell_t^i)$$

where $U_\beta : [0, 1] \to [0, 1]$ is a function, parameterized by $\beta \in [0, 1]$, which satisfies

$$\beta^r \leq U_\beta(r) \leq 1 - (1 - \beta)r$$

Since it is very difficult to set the β parameter properly, in particular when the number of experts is large, the Hedge(β) algorithm cannot be directly applied in practice. In Chaudhuri et al. (2009), a parameter-free algorithm for DTOL was presented, named NormalHedge. The idea is to introduce a new notion of regret. At any time t, the learner's instantaneous regret to expert i is $r_t^i = \ell_t^A - \ell_t^i$, and its cumulative regret to expert i in the first t times is

$$R_t^i = \sum_{\tau=1}^{t} r_\tau^i = R_{t-1}^i + (\ell_t^A - \ell_t^i) \tag{2}$$

The goal of the learner is to minimize this cumulative regret R_t^i over all the experts i (in particular, the best expert), for any time t. The NormalHedge algorithm allows the weight to be updated for expert i as

$$w_{t+1}^i \propto \frac{[R_t^i]_+}{c_t} \exp\left(\frac{([R_t^i]_+)^2}{2c_t}\right) \tag{3}$$

where $[x]_+$ denotes $\max\{0, x\}$ and c_t is a scale parameter satisfying

$$\frac{1}{N} \sum_{i=1}^{N} \exp\left(\frac{([R_t^i]_+)^2}{2c_t}\right) = e \tag{4}$$

In order to make NormalHedge more robust in practical systems, Chaudhuri et al. (2010) introduced the NormalHedge algorithm. Let $0 < \alpha < 1$ be the percentage factor, the cumulative regret to expert i in the first t times is computed as

$$R_t^i = \sum_{\tau=1}^{t} r_\tau^i = \alpha R_{t-1}^i + (\ell_t^A - \ell_t^i) \tag{5}$$

Note that Eq. 2 is a special case of Eq. 5 when α is set to one.

3.2. Adaptive NormalHedge algorithm

The standard NormalHedge algorithm computes the cumulative regret of an expert at the current time by summing the learner's instantaneous regret to this expert and the cumulative regret to this expert at the previous time (see Eq. 2). Although further improvement as shown in Eq. 5 is effective as it allows the learner to easily recover from past mistakes, there is still a problem in properly setting the percentage factor. In Chaudhuri et al. (2010), the authors addressed this problem by running multiple copies of NormalHedge with multiple values of the percentage factor, and then choosing the output of the copy that performs the best in an online style. However, this solution is not optimal. First of all, the solution is very time-consuming when running multiple copies of NormalHedge at the same time. Secondly, setting the percentage factor with a fixed values for all the experts is questionable as there are both good and poor experts at each time. For good experts that perform well at the current time, it is reasonable to assume that they have provided correct prediction at the previous times. Therefore, we should set α to a large value. In contrast, for poor experts, we should set α to a small value.

In order to overcome the shortcoming of the standard NormalHedge algorithm, we propose an adaptive NormalHedge algorithm. The idea is the use of an adaptive mechanism to determine how much historic information of each expert should be used at the current time. We adaptively set α for each expert according to the learner's instantaneous regret to this expert as

$$\alpha = \begin{cases} 1 - \frac{1}{2}\exp -\gamma|\ell_t^A - \ell_t^i| & \text{if } \ell_t^A > \ell_t^i \\ \frac{1}{2}\exp -\gamma|\ell_t^A - \ell_t^i| & \text{else} \end{cases} \tag{6}$$

where γ is a constant that controls the shape of the exponential function. The motivation behind this mechanism is that, for a good expert whose current regret is less than the learner's expected regret, we increase the percentage of its previous cumulative regret when computing its current cumulative regret. On the contrary, for a poor expert, we reduce the percentage of its previous cumulative regret. The proposed adaptive NormalHedge is shown in Algorithm 1.

4. Visual tracking based on adaptive NormalHedge

Motivated by the latest progress of DTOL, especially the NormalHedge algorithm, we consider visual

Algorithm 1: Adaptive NormalHedge

```
1 initialize $R_t^i = 0$, $w_t^i = 1/N$;
2 for $t = 1, 2, \dots$ do
3     Each expert $i$ incurs loss $\ell_t^i$;
4     Compute learner's expected loss using Eq. 1;
5     Compute percentage factor using Eq. 6;
6     Update cumulative regrets using Eq. 5;
7     Find $c_t > 0$ satisfying Eq. 4;
8     Update weight using Eq. 3;
9 end
```

tracking as a DTOL problem and propose a solution to visual tracking based on the proposed adaptive NormalHedge algorithm presented in Section 3.2. The proposed tracking algorithm approximates the state posterior distribution by a set of weighted experts of the DTOL. The weight of each expert is computed by our proposed adaptive NormalHedge algorithm. The tracking result is obtained by the average weight of all the experts.

Let $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$ be the state and noisy observation at time t, respectively. Given all the available observations $\boldsymbol{y}_{1:t-1} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \dots, \boldsymbol{y}_{t-1}\}$ up to time $t-1$, we approximate the state posterior distribution $p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1})$ at time $t-1$ by a set of weighted experts $\{\boldsymbol{x}_{t-1}^i, w_{t-1}^i\}_{i=1\dots N}$ where N is the number of experts and w_{t-1}^i is the weight of the i-th expert. At the current time t, in order to obtain the approximation of $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$, the proposed tracking algorithm goes through three steps: experts prediction, experts weighting and experts resampling.

Experts prediction: Experts at time t are obtained from the last time $t-1$ by a second-order autoregressive dynamical model

$$\boldsymbol{x}_t \sim \mathcal{N}(g(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}), \Sigma_d), \tag{7}$$

where $\mathcal{N}(\mu, \Sigma)$ is the normal distribution with mean μ and covariance Σ, and

$$g(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}) = A\boldsymbol{x}_{t-1} + B\boldsymbol{x}_{t-2}, \tag{8}$$

where A and B define a constant acceleration model.

Experts weighting: For the i-th expert $\boldsymbol{x}_t^i$, a kernel-weighted histogram $\boldsymbol{q}(\boldsymbol{x}_t^i) = \{q_l(\boldsymbol{x}_t^i)\}_{l=1\dots L}$ can be computed from its corresponding candidate image where L is the number of histogram bins. In this work, we use the HSV histogram as reported in Pérez et al. (2002). The target template histogram is $\boldsymbol{q}^*$ that has been obtained in the first frame. We define the loss of the i-th expert as

$$\ell_t^i = -\exp\{-\lambda D^2[\boldsymbol{q}^*, \boldsymbol{q}(\boldsymbol{x}_t^i)]\}, \tag{9}$$

where $D[\boldsymbol{q}^*, \boldsymbol{q}(\boldsymbol{x}_t^i)]$ is a distance function derived from the Bhattacharyya similarity coefficient (Pérez et al., 2002) and defined as

$$D[\boldsymbol{q}^*, \boldsymbol{q}(\boldsymbol{x}_t^i)] = \left[1 - \sum_{l=1}^{L} \sqrt{q_l^* q_l(\boldsymbol{x}_t^i)}\right]^{\frac{1}{2}} \tag{10}$$

Algorithm 2: Visual tracking based on Adaptive NormalHedge

```
  Input: $N$ (number of experts), $A$ and $B$ (dynamics
         functions), $\mathcal{A} := \{\boldsymbol{x}_1^1, \dots, \boldsymbol{x}_1^N\}$ with
         $\boldsymbol{x}_1^i = \boldsymbol{x}_0^*$; $R_0^i := 0$, $w_0^i := 1/N$, $\forall i$
1 for $t = 1, 2, \dots$ do
2     Obtain losses $\ell_t^i$ for each expert $i$ and compute
      learner's expected losses: $\ell_t^A = \sum_{i=1}^N w_{t-1}^i \ell_t^i$;
3     Compute percentage factor using Eq. 6 ;
4     Update cumulative regrets using Eq. 5;
5     Find $c_t > 0$ satisfying Eq. 4;
6     Compute weight of each expert using Eq. 3;
7     Estimate current tracking result using Eq. 11;
8     Resample $N$ experts $\boldsymbol{x}_t^i$ with replacement from
      current expert set according to probabilities
      $w_t^i$;
9     Predict experts using Eq. 7;
10 end
```

The motivation behind the definition of the loss function is that the smaller the loss, the larger the similarity between the target template and the expert candidate. After obtaining loss ℓ_t^i for each expert i, we compute the learner's expected loss using Eq. 1. Afterwards, we compute the percentage factor α by Eq. 6. Finally, we update regrets for each expert by Eq. 5 and weight each expert using Eq. 3. The tracking result $\hat{\boldsymbol{x}}_t$ at time t is estimated as the average of all the experts as

$$\hat{\boldsymbol{x}}_t = \sum_{i=1}^{N} w_t^i \boldsymbol{x}_t^i \tag{11}$$

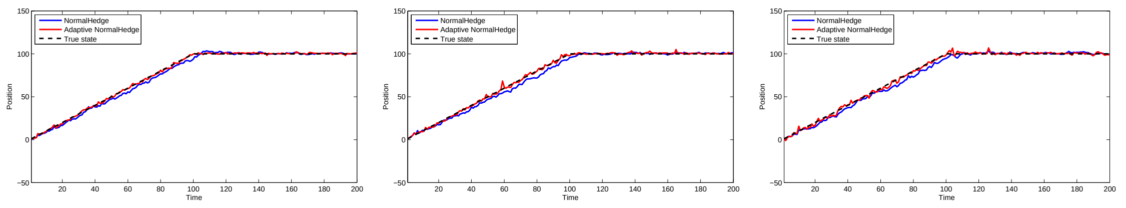
Experts resampling: When experts were weighted by Eq. 3, the experts with high weights perform well in the approximation of the state posterior distribution, and they may hold high confidence in the next approximation as well. In contrast, for those experts with low weights, they will make less contributions to the next approximation. Therefore, we propose to resample all experts based on their weights. We associate a number of offspring N_t^i with each expert $\boldsymbol{x}_t^i$ in such a way that $(N_t^1, N_t^2, \dots, N_t^N)$ follow a multinomial distribution with a parameter vector $(N, \boldsymbol{w}_t)$ where $\boldsymbol{w}_t = [w_t^1, w_t^2, \dots, w_t^N]$. After resampling, each offspring is associated with a weight of $1/N$.

The tracking algorithm starts with a set of experts with the initial state $\boldsymbol{x}_0^*$ obtained by manually labeled in the first frame, and assign each expert with the same weight. During each iteration, the tracking algorithm repeats the three steps shown above. The detailed tracking procedure is summarized in Algorithm 2.

5. Experimental results

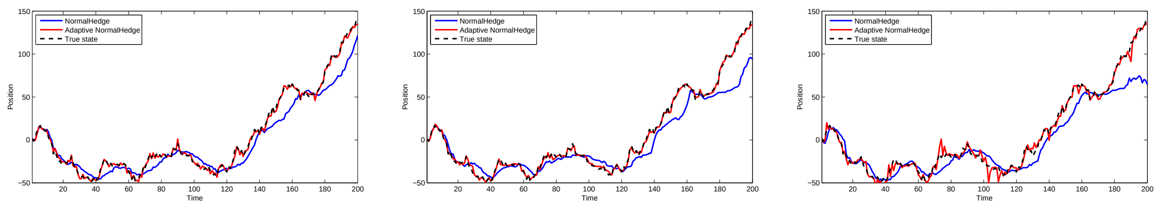
5.1. Simulation experiments

In order to evaluate the performance of the proposed adaptive Normalhedge algorithm, compared with the



(a) $\rho = 0$ (b) $\rho = 0.1$ (c) $\rho = 0.2$

Figure 1: Predicted states in the first simulation experiment with ρ being 0, 0.1 and 0.2.



(a) $\rho = 0$ (b) $\rho = 0.1$ (c) $\rho = 0.2$

Figure 2: Predicted states in the second simulation experiment with ρ being 0, 0.1 and 0.2.

standard Normalhedge, we conducted two simulation experiments. Our experiment setup is similar to that reported in Chaudhuri et al. (2010), which considered the task of tracking an object in an one-dimensional state space. At time t, the true state is the position z_t in the interval $[-500, 500]$. The measurements correspond to a 1000-dimensional vector $\mathbf{M}(t) = [M(-500, t), M(-499, t), \ldots, M(499, t), M(500, t)]$ for locations in the grid $G = \{-500, -499, \ldots, 499, 500\}$, generated by an additive noise process $M(x, t) = H(x, z_t) + n_t(x)$, where $H(x, z_t) = 1$ if $|x - z_t| \leq W$ and 0 otherwise. The additive noise $n_t(x)$ is randomly generated using the mixture distribution $(1 - \rho) \cdot \mathcal{N}(0, \sigma_o^2) + \rho \cdot \mathcal{N}(0, (10\sigma_o)^2)$ where σ_o represents how noisy the measurements are relative to the signal and ρ represents the fraction of the outliers. In these two simulations, we fix $\sigma = 1.0$ and then change ρ. In the experiments, we also have $W = 50$ and the total number of time steps for tracking is $T = 200$.

Table 2: Simulation results of the first experiment. The RMSE between the true state and the predicted state over $T = 200$ time steps for our adaptive NormalHedge and the standard NormalHedge. The RMSE is computed over 100 independent simulations.

ρ	Normalhedge	Adaptive NormalHedge
0.00	15.80 ± 2.7	2.30 ± 0.24
0.01	16.10 ± 5.1	2.40 ± 0.36
0.02	16.40 ± 3.4	4.60 ± 0.88

In the first simulation, the true state changes with the velocity equivalent to 1 when $t \leq 100$ and then remains stationary. Fig. 1 shows the true states (Black) and the states predicted by our adaptive normalhedge (Red) and the standard normalhedge (Blue) for three different values of ρ in one independent simulation. Table 2 shows the average and standard deviation of the Root-Mean-Squared-Error (RMSE) between the true states and the predicted ones. The RMSE is computed over 100 independent simulations. In the second simulation experiment, we randomly generate the true state sequence. Similar to the first simulation experiment, we show the predicted states and the corresponding RMSE in Fig. 2 and Table 3 respectively.

Table 3: Simulation results of the second experiment. The RMSE between the true state and the predicted state over $T = 200$ time steps for our adaptive NormalHedge and the standard NormalHedge. The RMSE is computed over 100 independent simulations.

ρ	Normalhedge	Adaptive NormalHedge
0.00	3.00 ± 0.25	1.60 ± 0.42
0.01	3.30 ± 0.37	2.60 ± 0.59
0.02	3.60 ± 0.62	3.50 ± 0.67

In the first simulation experiment, the true states involve a simple motion (move with a constant velocity or keep stationary). Although both the standard Normalhedge and the proposed adaptive Noramlhedge perform well (as shown in Fig. 1), the proposed adaptive Noramlhedge is slightly better than the standard Normalhedge (as shown in Table 2). As shown in Fig 2 and Table 3, however, when the true states are generated randomly, the standard Normalhedge cannot accurately predict the true states anymore and this results in large RMSE. In contrast, the proposed adaptive Normalhedge still performs well. These two simulation experiments prove that our adaptive Normalhedge is effective when a complex motion is engaged.

5.2. *Experiments on video sequences*

To demonstrate the effectiveness of the proposed tracking framework, we select several baseline trackers for the comparisons, including the fragment based tracker (Frag) (Adam et al., 2006), the incremental visual tracker (IVT) (Ross et al., 2007), the multiple instance learning tracker (MIL) (Babenko et al., 2011), the online AdaBoost tracker (OAB) (Grabner and Bischof, 2006), the ℓ_1 norm minimization based tracker (L1) (Mei and Ling, 2009), the minimum error bounded L1 tracker (BL1) (Mei et al., 2011), the online local sparse representation based tracker (OLSR) (Wang et al., 2012), and the structural local sparse appearance based tracker (SLSA) (Jia et al., 2012) using eight publicly available test sequences[1]. These sequences were recorded either outdoors or indoors. The challenges raised from these sequences contains occlusion, illumination, pose changes and so on and so forth. In the implementation of our tracker, the expert is a four dimensional variable consisting of target center coordinates and size. We set $\gamma = 2$, $N = 600$, $\Sigma_d = diag(5, 5, 0.5, 0.5)$, $\lambda = 20$, $A = 2$ and $B = -1$ for all the test sequences.

To quantitatively assess the performance of the selected trackers, we choose two evaluation criteria. The first one is *tracking success rate*, which computes the percentage of correctly tracked frames over the entire sequence. To evaluate whether the target is correctly tracked or not in a frame, we adopt the PASCAL score (Everingham et al., 2010), which can be computed as $\frac{area(\mathcal{R}^* \cap \mathcal{R}_{gt})}{area(\mathcal{R}^* \cup \mathcal{R}_{gt})}$ where $\mathcal{R}^*$ is the bounding box obtained by a tracker, $\mathcal{R}_{gt}$ is the corresponding ground truth bounding box, and $area(\mathcal{R})$ is the area of the bounding box $\mathcal{R}$. The target is correctly tracked in a frame if the score is larger than 0.5. The second one is *center position error*, referring to the distance between the center position of the tracking results and the ground truth. At time t, let (x_t, y_t) be the center position of the tracking result, $(\hat{x}_t, \hat{y}_t$ be the center position of the ground truth, the center position error at time t can be computed as $\sqrt{(x_t - \hat{x}_t)^2 + (y_t - \hat{y}_t)^2}$. To evaluate the overall performance of a tracker, the average center position error over the entire sequence is used.

5.2.1. *Qualitative Comparison*

Occlusion is extremely challenging for visual tracking as the appearance of the tracked target will be changed in this case. In order to test whether or not our proposed method is robust to occlusion, we conduct experiments on two challenging sequences. The first sequence is the *woman* sequence where a woman is partially occluded by a car. The second sequence is the *face* sequence where the face of a woman is partially occluded by a book. Some representative tracking results of these two sequences are shown on the first and the second rows of Fig. 3. As we can see, our tracker is capable of accurately tracking the target shown on all the representative frames. However, other trackers either drift to the background or fully lose the target.

Different from partial occlusion, illumination changes will affect the global appearance of a target. We choose two sequences *david_outdoor* and *david_indoor* to evaluate the performance of the proposed tracker in handling illumination changes. These two sequences were recorded in outdoor and indoor environments respectively, and therefore undergo different illumination changes. Some representative tracking results are shown on the third and fourth rows of Fig. 3. As we can see, our tracker correctly tracks the target on all the representative frames and other trackers somehow fail in the tests.

The fifth and sixth rows of Fig. 3 show some representative tracking results of the *sylv* and *bird* sequences in which the tracked target undergoes pose changes. When the toy in the *sylv* sequence has small pose changes, the proposed tracker successfully tracks the target in all the representative frames. However, when the bird in the *bird* sequence has large pose changes, the proposed tracker also drifts to the background. Compared with the other trackers, the proposed tracker still has much better performance. The remaining two sequences are the *CAVIAR* and *singer* sequences which come up with the challenges of background cluster and scale changes. The representative tracking results of these two sequences are shown on the last two rows of Fig. 3. The proposed tracker also outperforms the other trackers.

5.2.2. *Quantitative Comparison*

The PASCAL score plots of the tested trackers on eight test sequences are shown in Fig. 4. As we can see, our tracker almost achieves the highest PASCAL scores on all the test sequences. The tracking success rates and average central position errors are shown in Tables 4 and 5, respectively. The proposed tracker achieves the highest tracking success rates from all the sequences. In term of the average central position errors, the proposed tracking method achieves the lowest errors on seven sequences and the second lowest error on the remaining one. These results indicate the overall performance of the proposed tracker on all the sequences significantly outperforms the other trackers.

Note that the latest baseline trackers (L1, BL1, OLSA and SLSA) are based on the particle filter framework. As explained in Section 1, these methods are sensitive to model mismatches. When challenges such as occlusions and illumination changes occur, these trackers tend to fail. In contrast, the proposed tracking method is based on the adaptive NormalHedge algorithm which is an explanation frame-

[1]The *woman* and *face* sequences are from http://www.cs.technion.ac.il/~amita/fragtrack/fragtrack.htm. The *singer* sequence is from http://cv.snu.ac.kr/research/~vtd/index.html. The *david_outdoor*, *david_indoor* and *sylv* sequences are from http://www.cs.toronto.edu/~dross/ivt/. The *bird* sequence is from http://ice.dlut.edu.cn/lu/Project/iccv_spt_webpage/iccv_spt.htm. The *CAVIAR* sequence is from http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/.

Table 4: The tracking successful rates of the evaluated trackers on eight test sequences. The highest tracking successful rate for each sequence is marked in red bold.

	woman	singer	face	david_outdoor	david_indoor	CAVIAR	sylv	bird
Frag	0.38	0.25	0.85	0.25	0.21	0.35	0.91	0.34
IVT	0.14	0.27	0.79	0.32	0.91	**1.00**	0.76	0.17
MIL	0.25	0.25	0.82	0.20	0.19	0.39	0.91	0.51
OAB	0.31	0.27	0.75	0.14	0.11	0.36	0.80	0.90
L1	0.17	0.27	0.81	0.66	0.48	0.38	0.48	0.45
BL1	0.13	0.22	0.58	0.62	0.24	0.89	0.48	0.40
OLSR	0.14	0.98	0.98	0.59	0.23	0.40	0.90	0.65
SLSA	0.97	0.66	0.25	0.79	0.71	0.40	0.98	0.58
Our	**1.00**	**1.00**	**1.00**	**0.95**	**1.00**	**1.00**	**1.00**	**1.00**

Table 5: The average center position errors of the evaluated trackers on eight test sequences. The lowest average center position error for each sequence is marked in red bold.

	woman	singer	face	david_outdoor	david_indoor	CAVIAR	sylv	bird
Frag	92.26	14.15	9.26	79.17	42.18	57.96	5.89	50.24
IVT	146.67	7.42	12.30	89.20	**3.61**	3.06	28.12	99.37
MIL	117.70	18.59	14.20	55.35	25.85	57.47	7.74	21.54
OAB	105.42	18.37	14.82	67.21	21.34	23.01	13.38	12.13
L1	94.37	61.26	23.37	24.49	25.23	58.09	46.81	51.11
BL1	88.42	78.08	45.80	39.72	68.09	7.17	42.89	36.72
OLSR	140.83	2.59	4.91	28.99	116.28	61.78	7.67	15.27
SLSA	2.49	4.44	53.61	14.14	15.91	45.75	6.73	20.42
Our	**1.76**	**2.24**	**3.35**	**6.31**	4.56	**2.02**	**4.29**	**7.73**

work and has the capability to handle complex challenges. In addition, the adaptive mechanism of the proposed method also makes our tracker easy to recover from past failure, which further improves our tracking performance.

We also compare the computational complexity of the proposed tracking method against that of several state-of-the-art methods. For a fair comparison, we select several trackers including IVT, L1, BL1, OLSA and SLSA which have recently been developed and implemented in Matlab. In addition, since our tracker has a similar framework to the particle filter based tracker (PF), we also choose a PF tracker as a baseline model. The average tracking speeds (frames/second) are shown in Table 6. It is evident that the average speed of our tracker is very close to PF and slightly faster than the IVT tracker and significantly faster than L1, BL1, OLSA and SLSA trackers. The most time consuming step is to compute the similarity between the candidate and the target template. In addition, IVT tracker uses a fast online manner to update the parameter subspace, leading to fast tracking speeds. However, L1, BL1, OLSA and SLSA trackers are all based on sparse coding, resulting in a complex optimization process and an extremely slow tracking speed.

6. Conclusion and future works

In this paper, we proposed a robust tracking framework, based on an adaptive NormalHedge algorithm. The main contribution of this paper has two-folds: Firstly, we improved the standard NormalHedge algorithm and proposed an adaptive NormalHedge algorithm, which overcomes the disadvantages of the fixed percentage factor used in the standard NormalHedge. Our proposed adaptive NormalHedge algorithm is simple yet effective as it adaptively determines the percentage factor of each expert according to the learner's instantaneous regret to the expert's expectation. Secondly, directly using the standard NormalHedge algorithm in visual tracking takes too much computational complexity. We proposed the corresponding strategies to handle these two problems. Simulation results showed the effectiveness of using the proposed adaptive NormalHedge algorithm in the hidden state estimation with noise observations. Experiments on several video sequences also show the proposed tracking algorithm based on the adaptive NormalHedge is more robust than several state-of-the-art trackers. Although the performance of the proposed tracking method is superior to most of the state of the art trackers, there are still some room for further improvement. In the future work, we will make efforts to implement an effective loss function that can be used in multiple target tracking.

7. Acknowledgement

The authors would like to thank the anonymous reviewers and the editors for their insightful comments. This work was supported in part by the Natural Science Foundation of China (No. 61300111, No. 61133003 and No. 61173086), the China Postdoctoral Science Foundation (No. 2014M550192), the Research Fund for the Doctoral Program of Higher Education of China (No. 20132302120084), the Fundamental Research Funds for the Central Universities, the Natural Scientific Research Innovation Foundation in Harbin Institute of Technology (HIT. NSRIF. 2014137) and the Science Research Foundation in Harbin Institute of Technology at Weihai (HIT (WH) 201309). H. Zhou's work was partially supported by the CSIT project funded by UK EPSRC under grant EP/H049606/1.

References

Adam, A., Rivlin, E., Shimshoni, I., 2006. Robust fragments-based tracking using the integral histogram. Proc. 2006 IEEE Conf. Computer Vision and Pattern Recognition 1, 798–805.

Babenko, B., Yang, M., Belongie, S., 2009. Visual tracking with online multiple instance learning. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 983–990.

Babenko, B., Yang, M., Belongie, S., 2011. Robust object tracking with online multiple instance learning. IEEE Transactions on Pattern Analysis and Machine Intelligencen 33 (8), 1619–1632.

Birchfield, S., Sriram, R., 2005. Spatiograms versus histograms for region-based tracking. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 2, 1158–1163.

Bradski, G., 1998. Computer vision face tracking as a component of a perceptual user interface. Intelligence Technology Journal 2, 1–15.

Castellini, C., Tommasi, T., Noceti, N., Odone, F., Caputo, B., 2011. Using object affordances to improve object recognition. IEEE Transactions on Autonomous Mental Development 3 (3), 207–215.

Cesa-Bianchi, N., Freund, Y., Helmbold, D., Haussler, D., Schapire, R., Warmuth, M., 1997. How to use expert advice. Journal of the ACM 44 (3).

Cesa-Bianchi, N., Lugosi, G., 2006. Prediction, Learning and Games. Cambridge University Press.

Chaudhuri, K., Freund, Y., D.Hsu, 2010. An online learning-based framework for tracking. Twenty-Sixth Conference on Uncertainty in Artificial Intelligence.

Chaudhuri, K., Freund, Y., Hsu, D., 2009. A parameter-free hedging algorithm. In: Proc. Neural Information Processing Systems (NIPS'09). pp. 1–15.

Collins, R., Liu, Y., Leordeanu, M., 2005. On-line selection of discriminative tracking features. IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (10), 1631–1643.

Comaniciu, D., Ramesh, V., Meer, P., 2000. Real-time tracking of non-rigid objects using mean shift. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 2, 142–149.

Comaniciu, D., Ramesh, V., Meer, P., 2003. Kernel-based object tracking. IEEE Trans. Pattern Analysis and Machine Intelligence 25 (5), 564–577.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., Zisserman, A., 2010. The pascal visual object classes (VOC) challenge. International Journal of Computer Vision 88 (2), 303–338.

Fieguth, P., Terzopoulos, D., 1997. Color based tracking of heads and other mobile objects at video frame rates. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 21–27.

Freund, Y., Schapire, R. E., 1997. A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences 55 (1), 119–139.

Gao, X., Wang, B., Tao, D., Li, X., 2011. A relay level set method for automatic image segmentation. IEEE Transactions on Systems, Man and Cybernetics Part B: Cybernetics 41 (2), 518–525.

Gao, X., Wang, B., Tao, D., Li, X., 2012a. Face sketch-photo synthesis and retrieval using sparse representation. IEEE Transactions on Circuits Systems for Video technology 22 (8), 1213–1226.

Gao, Y., Wang, M., Ji, R., Zha, Z.-J., Shen, J., 2012b. k-partite graph reinforcement and its application in multimedia information retrieval. Information Sciences 194, 224–239.

Grabner, H., Bischof, H., 2006. On-line boosting and vision. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 1, 260–267.

Grabner, H., Leistner, C., 2008. Semi-supervised on-line boosting for robust tracking. Proceedings of the 10th European Conference on Computer Vision, 234–247.

Gutman, P., Velger, M., 1990. Tracking targets using adaptive kalman filtering. IEEE Transactions on Aerospace and Electronic Systems 26 (5), 691–699.

Hager, G., Dewan, M., Stewart, C., 2004. Multiple kernel tracking with SSD. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 1, 790–797.

Hess, R., Fern, A., 2009. Discriminatively trained particle filters for complex multi-object tracking. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1–8.

Isard, M., Blake, A., 1996. Contour tracking by stochastic propagation of conditional density. Proceedings of the 4th European Conference on Computer Vision, 343–356.

Isard, M., Blake, A., 1998. Condensation - conditional density propagation for visual tracking. International Journal of Computer Vision 29 (1), 5–28.

Jepson, A., Fleet, D., EI-Maraghi, T., 2003. Robust online appearance models for visual tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (10), 1296–1311.

Jia, X., Lu, H., Yang, M., 2012. Visual tracking via adaptive structural local sparse appearance model. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1–8.

Littlestone, N., Warmuth, M., 1994. The weighted majority algorithm. Information and Computation 108 (2).

Mei, X., Ling, H., 2009. Robust visual tracking using L1 minimization. Proceedings of the 12th International Conference on Computer Vision, 1436–1443.

Mei, X., Ling, H., Wu, Y., Blasch, E., Bai, L., 2011. Minimum error bounded efficient L1 tracker with occlusion detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1257–1264.

Pérez, P., Hue, C., Vermaak, J., Gangnet, M., 2002. Color-based probabilistic tracking. Proceedings of European Conference on Computer Vision, 661–675.

Ross, D., Lim, J., Lin, R., Yang, M., 2007. Incremental learning for robust visual tracking. International Journal of Computer Vision 77 (8), 125–141.

Shahrokni, A., Drummond, T., Fua, P., 2005. Fast texture-based tracking and delineation using texture entropy. Proceedings of International Conference on Computer Vision 2, 1154–1160.

Smith, K., Gatica-Perez, D., 2004. Order matters: A distributed sampling method for multi-object tracking. Proceedings of the British Machine Vision Conference, 25–32.

Wang, Q., Chen, F., Xu, W., Yang, M., 2012. Online discriminative object tracking with local sparse representation. Proceedings of IEEE Workshop on the Applications of Computer Vision, 425–432.

Wu, Y., Fan, J., 2009. Contextual flow. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition 1, 33–40.

Yu, J., Rui, Y., Chen, B., 2014a. Exploiting click constraints and multi-view features for image re-ranking. IEEE Transactions on Multimedia 16 (1), 159–168.

Yu, J., Rui, Y., Tang, Y., Tao, D., 2014b. High-order distance-based multiview stochastic learning in image classification. IEEE Transactions on Cybernetics10.1109/TCYB.2014.2307862.

Yu, J., Rui, Y., Tao, D., 2014c. Click prediction for web image reranking using multimodal sparse coding. IEEE Transactions on Image Processing 23 (5), 2019–2032.

Yu, J., Tao, D., Rui, Y., Cheng, J., 2012. On combining multiple features for cartoon character retrieval and clip synthesis. IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 42 (5), 1413–1427.

Yu, J., Tao, D., Rui, Y., Cheng, J., 2013. Pairwise constraints based multiview features fusion for scene classification. Pattern Recognition 46 (2), 483–496.

Zhang, S., Yao, H., Sun, X., Liu, S., 2012. Robust visual tracking using an effective appearance model based on sparse coding. ACM Transactions on Intelligent Systems and Technology 3 (3), 1–18.

Zhang, S., Yao, H., Sun, X., Lu, X., 2013a. Sparse coding based visual tracking: Review and experimental comparison. Pattern Recognition 46 (7), 1772–1788.

Zhang, S., Yao, H., Sun, X., Wang, K., Zhang, J., Lu, X., Zhang, Y., 2014. Action recognition based on overcomplete independent components analysis. Information Sciences10.1016/j.ins.2013.12.052.

Zhang, S., Yao, H., Zhou, H., Sun, X., Liu, S., 2013b. Robust visual tracking based on online learning sparse representation. Neurocomputing 100, 31–40.

Zhao, Q., Brennan, S., Tao, H., 2007. Differential emd tracking. Proceedings of IEEE Conference on Computer Vision, 1–8.

Zhao, Q., Tao, H., 2005. Object tracking using color correlogram. Proceedings of IEEE Workshop Performance Evaluation of Tracking and Surveillance, 263–270.

Zhou, H., Taj, M., Cavallaro, A., 2008. Target detection and tracking with heterogeneous sensors. IEEE Journal of Selected Topics in Signal Processing 2 (4), 503–513.

Zhou, H., Wallace, A., Green, P., 2009a. Efficient tracking and egomotion recovery using gait analysis. Signal Processing 89 (12), 2367–2384.

Zhou, H., Yuan, Y., Shi, C., 2009b. Object tracking using sift features and mean shift. Computer Vision and Image Understanding 113 (3), 345–352.

Zhu, C., Byrd, R., Lu, P., Nocedal, J., 1997. Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization. ACM Transaction Mathematical Software 23, 550–560.

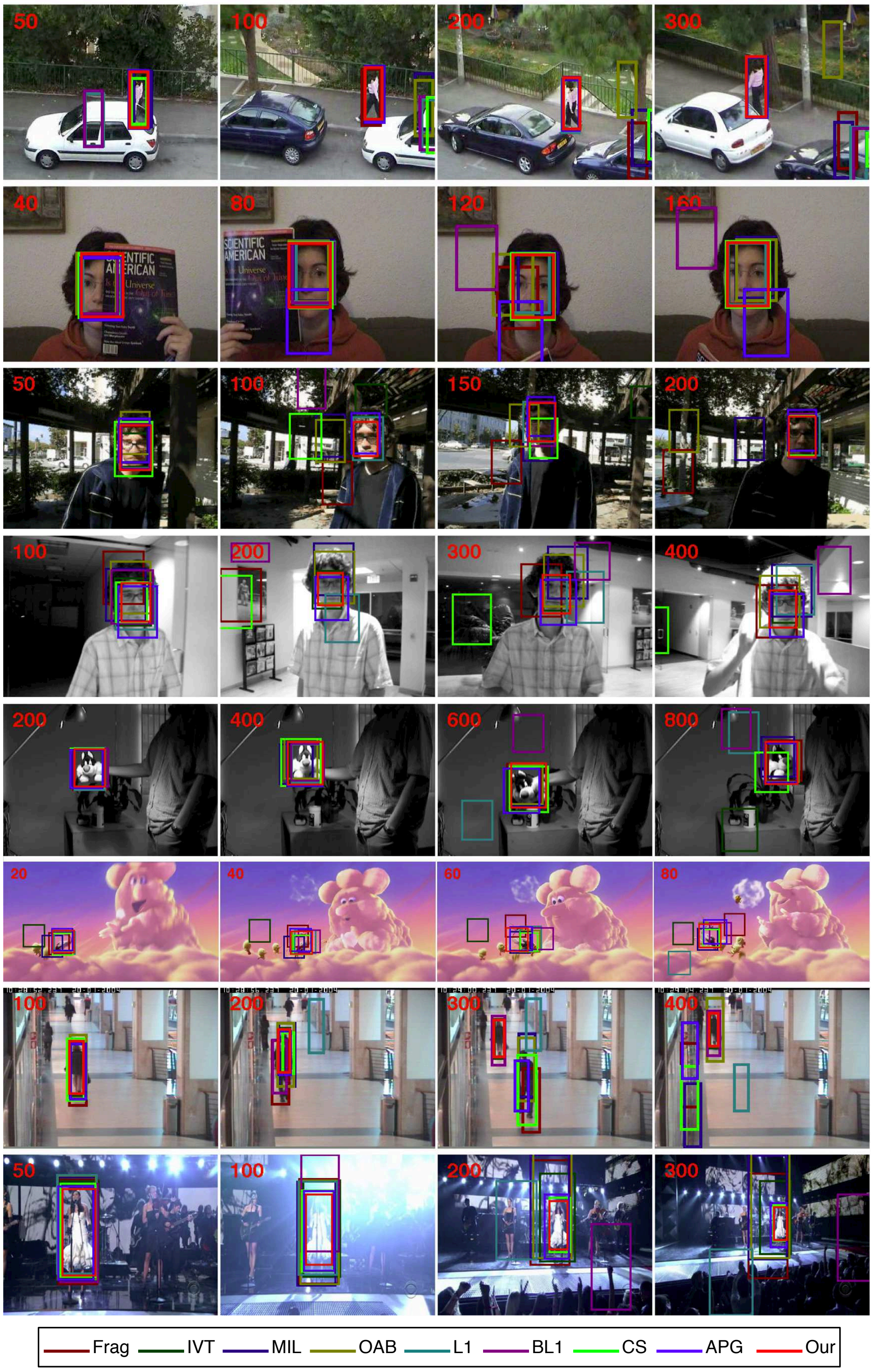
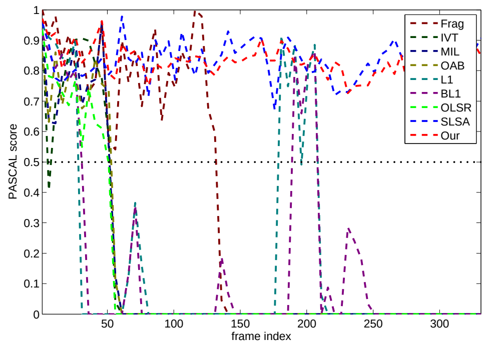


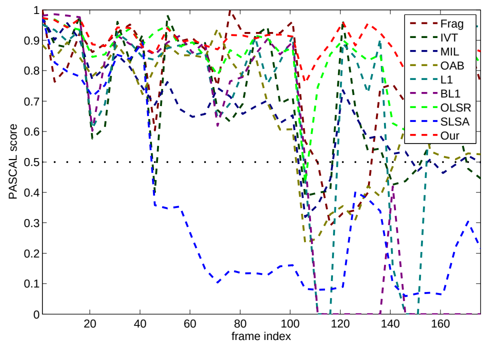
Figure 3: Sample representative tracking results of the evaluated trackers on eight test sequences.

Table 6: The average tracking speeds (frames/second) comparison between our tracker and several state-of-the-art trackers.

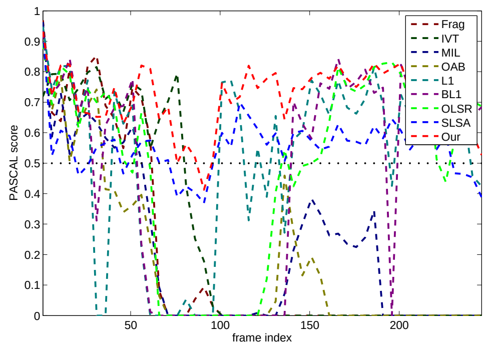
Tracker	PF	IVT	L1	BL1	OLSA	SLSA	Our
Speed	12.44	11.15	0.18	0.51	0.47	1.93	11.76



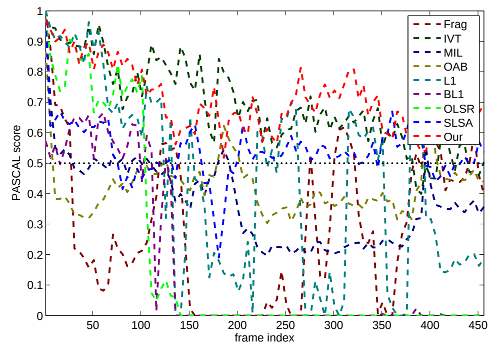
(a) *woman*



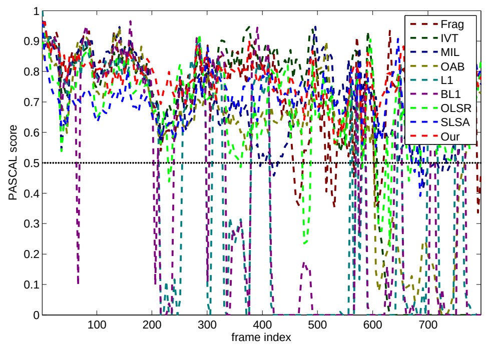
(b) *face*



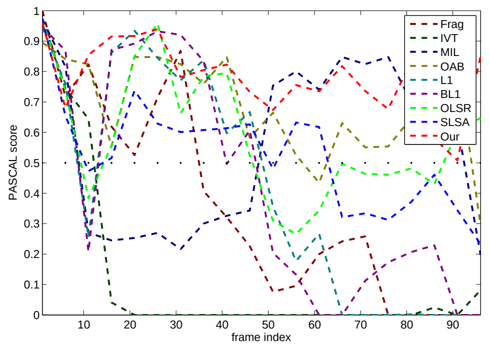
(c) *david_outdoor*



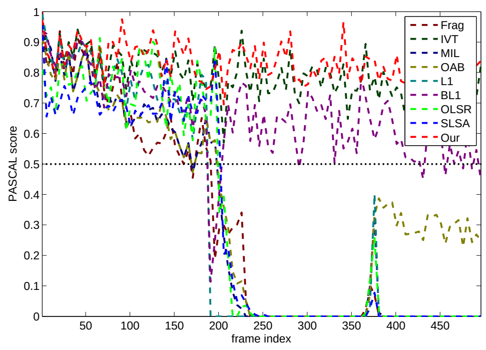
(d) *david_indoor*



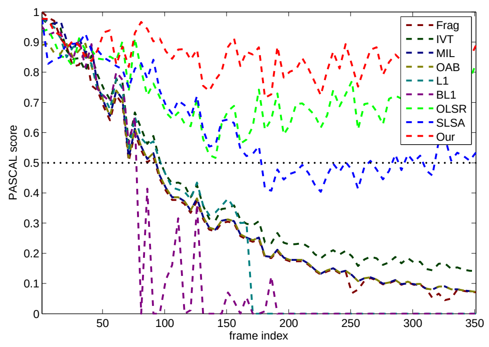
(e) *sylv*



(f) *bird*



(g) *CAVIAR*



(h) *singer*

Figure 4: The PASCAL score plots of the evaluated trackers on eight test sequences.