Robust Tracking with Weighted Online Structured Learning

Rui Yao[1,2], Qinfeng Shi[2], Chunhua Shen[2], Yanning Zhang[1], and Anton van den Hengel[2]

[1] School of Computer Science, Northwestern Polytechnical University, China
[2] School of Computer Science, The University of Adelaide, Australia

Abstract. Robust visual tracking requires constant update of the target appearance model, but without losing track of previous appearance information. One of the difficulties with the online learning approach to this problem has been a lack of flexibility in the modelling of the inevitable variations in target and scene appearance over time. The traditional online learning approach to the problem treats each example equally, which leads to previous appearances being forgotten too quickly and a lack of emphasis on the most current observations. Through analysis of the visual tracking problem, we develop instead a novel weighted form of online risk which allows more subtlety in its representation. However, the traditional online learning framework does not accommodate this weighted form. We thus also propose a principled approach to weighted online learning using weighted reservoir sampling and provide a weighted regret bound as a theoretical guarantee of performance. The proposed novel online learning framework can handle examples with different importance weights for binary, multiclass, and even structured output labels in both linear and non-linear kernels. Applying the method to tracking results in an algorithm which is both efficient and accurate even in the presence of severe appearance changes. Experimental results show that the proposed tracker outperforms the current state-of-the-art.

1 Introduction

Visual tracking is an important topic in computer vision, and has a wide range of practical applications including video surveillance, perceptual user interfaces, and video indexing. Tracking generic objects in a dynamic environment poses additional challenges, including appearance changes due to illumination, rotation, and scaling; and problems of partial vs. full visibility. In all cases the fundamental problem is the variable nature of appearance, and the difficulties associated with using such a changeable feature to distinguish the target from backgrounds.

A wide variety of approaches have been developed for modelling appearance variations of targets within robust tracking. Many tracking algorithms attempt to build robust object representations in a particular feature space, and search for the image region which best corresponds to that representation. Such approaches include the superpixel tracker [1], the integral histogram [2], subspace learning

[3], visual tracking decomposition [4], interest points tracking [5], and sparse representation tracking [6, 7], amongst others.

An alternative approach has been to focus on learning, and particularly on identifying robust classifiers capable of distinguishing the target from backgrounds, which has been termed tracking-by-detection. This approach encompases boosting [8, 9], ensemble learning [10], multiple instance learning [11], discriminative metric learning [12], graph mode-based tracking [13], structured output tracking [14], and many more. One of the features of these approaches has been the ability to update the appearance model online and thus to adapt to appearance changes.

Despite their successes, existing online tracking-by-detection methods have two main issues. First, these trackers typically rely solely on an online classifier for adapting the object appearance model and thus completely discard past training data. This makes it almost impossible to recover once the appearance model begins to drift. Second, as tracking is clearly a time-varying process, more recent frames should thus have a greater influence on the current tracking process than older ones. Existing approaches to online learning of appearance models do not take this into account, but rather either include images in the model fully, or discard them totally.

Contribution Our contributions are: 1) We propose the first formulation of visual tracking as a weighted online learning problem. 2) We propose the first framework for weighted online learning. This is achieved using weighted reservoir sampling. We also provide a weighted online regret bound for the new method. 3) Using our weighted online learning framework, we propose a robust tracker with a time-weighted appearance model. The proposed tracker is capable of exploiting both structured outputs and kernels, and performs efficiently and accurately even in the presence of severe appearance drift. The weights associated with online samples equip the tracker with the ability to adapt quickly while retaining previous appearance information. The fact that the latest observation is given higher weight means that it has a higher probability of being maintained in the reservoir. This makes our method adapt quickly to newly observed data and also offers flexibility in the modelling of the frequent variations in target and scene appearance over time. On the other hand, unlike traditional online learning which sees the recent data only, our weighted online learning method trains on both the recent data and historical data through the use of weighted reservoir sampling. The combination of new and old information helps the tracker recover from drift, since the patterns that the tracker has seen in the past are not completely discarded. Overall, weighted online learning can update the object appearance model over time without losing track of previous appearances thus improving performance.

1.1 Related work

We now briefly review recent related tracking-by-detection methods. In [15], an off-line support vector machine (SVM) is used to distinguish the target vehicle

from the background, but it requires that all appearance variations be represented within the training data. Since the video frames are often obtained on the fly, features are obtained in an online fashion [9].For the same reason, online learning are brought into tracking. For example, Grabner *et al.* [9] proposed to update selected features incrementally using online AdaBoost. Babenko *et al.* [11] used online multiple instance boosting to overcome the problem of ambiguous labeling of training examples. Grabner *et al.* [8] combined the decision of a given prior and an online classier and formulated them in a semi-supervised fashion.

Structured learning has shown superior performance on a number of computer vision problems, including tracking. For example, in contrast to binary classification, Blaschko *et al.* [16] considered object localisation as a problem of predicting structured outputs: they modeled the problem as the prediction of the bounding box of objects located in images. The degree of bounding box overlap to the ground truth is maximised using a structured SVM [17]. Hare *et al.* [14] further applied structured learning to visual object tracking. The approach, which is termed "Struck", directly predicts the change in object location between frames using structured learning. Since Struck is closely related to our work, we now provide more details here.

Struck: Structured output tracking with kernels. In [14] Hare *et al.* used a Structured SVM [17] to learn a function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$, such that, given the t-th frame $\mathbf{x}_t \in \mathcal{X}$, and the bounding box of the object B_{t-1} in the $(t-1)$-th frame, the bounding box B_t in the t-th frame can be obtained via

$$B_t = B_{t-1} + \mathbf{y}_t^*, \quad \mathbf{y}_t^* = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} f(\mathbf{x}_t, \mathbf{y}). \tag{1}$$

Here $B = (c, l, w, h)$, where c, l are the column and row coordinates of the upper-left corner, and w, h are the width and height. The offset is $\mathbf{y} = (\Delta c, \Delta l, \Delta w, \Delta h)$. Hare *et al.* set Δw and Δh to always be 0 and $f(\mathbf{x}, \mathbf{y}) = \langle \mathbf{w}, \Psi(\mathbf{x}, \mathbf{y}) \rangle$ for the feature map $\Psi(\mathbf{x}, \mathbf{y})$.

The discriminative function f is learned via structured SVM [17] by minimising

$$\begin{aligned} &\min_{\mathbf{w}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{t=1}^{T}\xi_t \\ &\text{s.t. } \forall t : \xi_t \geq 0 \ \ ; \forall t, \mathbf{y} \neq \mathbf{y}_t : \langle \mathbf{w}, \Psi(\mathbf{x}_t, \mathbf{y}_t) \rangle - \langle \mathbf{w}, \Psi(\mathbf{x}_t, \mathbf{y}) \rangle \geq \Delta(\mathbf{y}_t, \mathbf{y}) - \xi_t, \end{aligned} \tag{2}$$

where the label cost is

$$\Delta(\mathbf{y}_t, \mathbf{y}) = 1 - \frac{Area((B_{t-1} + \mathbf{y}_t) \bigcap (B_{t-1} + \mathbf{y}))}{Area((B_{t-1} + \mathbf{y}_t) \bigcup (B_{t-1} + \mathbf{y}))}, \tag{3}$$

which was introduced in [16] to measure the VOC bounding box overlap ratio. In practice, $\mathbf{y}$'s are uniformly sampled in the vicinity of the $\mathbf{y}_t$ in [14]. To enable kernels, Hare *et al.* updated the dual variables online following the work of [16, 18, 19] and achieved excellent results. Note that prior to the work of [14, 16],

supervised learning in detection and tracking often required large volumes of manually labeled training examples, which are not easy to obtain in many cases. Also in tracking, when it is considered as binary classification, the boundary is not well defined between a target image patch and a non-target image patch. Using structured learning to optimise a label cost like the VOC bounding box overlap ratio, the manual labeling requirement is significantly reduced and the ambiguity between target and non-target images patches is removed[14].

However, tracking is a time-varying process. It is not appropriate to weight each frame uniformly, as this can cause tracking failure when visual drifts occurs. It is this problem which motivates our approach.

2 Weighted Online Learning

In this section, we propose a novel framework for online learning with weights on examples. We first re-visit the standard approach to online learning as applied to visual tracking, and then propose a weighted online risk that is better suited to the requirements of the problem. The novel formulation of risk we propose cannot be directly optimised by traditional online learning, however. We thus propose an approach based on reservoir sampling for both primal and dual variables. A weighted regret bound is provided as a performance guarantee.

In online learning, at each iteration, the online algorithm has visibility of one observation $\mathbf{z}$, or at most a small selection of recent data (a.k.a. mini-batch). Here $\mathbf{z} = (\mathbf{x}, y)$ for a binary or multiclass problem, and $\mathbf{z} = (\mathbf{x}, \mathbf{y})$ for a multilabel or a structured problem. In our visual tracking context (also in [14]), $\mathbf{z} = (\mathbf{x}, \mathbf{y}, \mathbf{y}')$ where $\mathbf{x}$ is an image frame, $\mathbf{y}$ is the ground truth offset of the target bounding box in the frame, and $\mathbf{y}'$ is a generated offset $\mathbf{y}' \neq \mathbf{y}$. The goal of online learning is to learn a predictor that predicts as well as batch learning (which has visibility over all data).

Unweighted Risk The regularised unweighted (empirical) risk for batch learning is

$$C \sum_{i=1}^{m} \ell(\mathbf{w}, \mathbf{z}_i) + \Omega(\mathbf{w}),$$

where $\Omega(\mathbf{w})$ is the regulariser (*e.g.* $\Omega(\mathbf{w}) = \| \mathbf{w} \|^2/2$) that penalises overly complicated models, and C is a scalar. Here $\ell(\cdot, \cdot)$ is a loss function such as hinge loss and $\mathbf{w}$ is the primal parameter.

Online learning algorithms use one datum or a very small number of recent data to learn at each iteration by definition. Thus at iteration i, the online algorithm minimises the regularised risk

$$J(i, \mathbf{w}) = C\ell(\mathbf{w}, \mathbf{z}_i) + \Omega(\mathbf{w}).$$

The *regret* Q of an online algorithm measures the prediction difference between the online predictor and the batch learning predictor using the same loss,

$$Q(m, \mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} \ell(\mathbf{w}_i, \mathbf{z}_i) - \frac{1}{m}\sum_{i=1}^{m} \ell(\mathbf{w}, \mathbf{z}_i).$$

The regret reflects the amount of additional loss (compared with batch algorithm) suffered by the online algorithm not being able to see the entire data set in advance. It has been shown in [20, 19] that minimising the risk $J(i, \mathbf{w})$ at each iteration reduces the regret $Q(m, \mathbf{w})$, and that the regret goes to zero when m goes to infinity for a bounded $\Omega(\mathbf{w})$.

2.1 Weighted Risk

In visual tracking, scene variation over time and target appearance changes mean that different frames provide varying levels of information about the current appearance of the target and background. In vehicle tracking, for example, recent frames are often more important than those earlier ones in the sequence due to location (and thus background) and target orientation changes. A weighted formulation of risk is capable of reflecting this variation between the levels of significance of particular frames. Given *nonnegative* weights $(\lambda_1, \lambda_2, \cdots, \lambda_m)$, we define the weighted (empirical) risk as

$$C\sum_{i=1}^{m} \lambda_i \ell(\mathbf{w}, \mathbf{z}_i) + \Omega(\mathbf{w}). \tag{4}$$

In a batch learning setting the miminisation of the weighted risk is straightforward. In online learning, however, the method processes only the latest observation, making it impossible to include weights on previous observations. Extending online learning naively to handle weights will causes several problems. For example, if we drop the regularisation term $\Omega(\mathbf{w})$, the weighting with respect to other data points is of no effect. We thus see that $\operatorname{argmin}_{\mathbf{w}} C\lambda_i \ell(\mathbf{w}, \mathbf{z}_i) = \operatorname{argmin}_{\mathbf{w}} \ell(\mathbf{w}, \mathbf{z}_i)$, and thus that λ_i has no effect. Even with regularisation, the weights imposed still do not faithfully reflect the desired weighting of the data due to the shrinkage of $\Omega(\mathbf{w})$ over time. These are the problems with online learning that we aim to resolve.

2.2 Reservoir and Optimisation

Though (4) is what we would like to minimise, it requires that we are able to recall all of the observed data at every stage, which is not feasible in an online setting. Usually one may need to operate over large data sets and long time periods. Reservoir sampling is an online method for random sampling from a population without replacement. Using weighted reservoir sampling allows us to develop an online method for selecting a (constant-sized) set of observations from a data stream according to their weights.

Proposition 1 (Expected Reservoir Risk). *Minimising (4) is equivalent to minimising*

$$\frac{CZ_m}{|R_m|}\,\mathbb{E}_{R_m}[\sum_{\mathbf{z}\in R_m}\ell(\mathbf{w},\mathbf{z})]+\Omega(\mathbf{w}), \tag{5}$$

where R_m is a weighted reservoir with weights $(\lambda_1,\lambda_2,\cdots,\lambda_m)$ as in [21] and $Z_m=\sum_{i=1}^{m}\lambda_i$.

This is due to a property of weighted reservoir sampling (see the proof in supplementary material). The reservoir is updated by replacing an element in the reservoir by the current datum with certain probability. See [21] for details. The method is applicable to data streams of unknown, or even infinite length, which means that rather than store all observations we might later require, we can instead maintain a reservoir and minimise the expected risk over the samples therein.

Reservoir Risk Due to Proposition 1, at each iteration i, we minimise

$$C\sum_{\mathbf{z}\in R_i}\ell(\mathbf{w},\mathbf{z})+\Omega(\mathbf{w}). \tag{6}$$

One of the properties of weighted reservoir sampling is that data (*i.e.* $\{\mathbf{z}_1,\cdots,\mathbf{z}_i\}$) with high weights have higher probability of being maintained in the reservoir. The advantage of using (6) is that a weighted risk in (4) now is replaced with a unweighted (*i.e.* uniformly weighted) risk, to which traditional online algorithms are applicable. The (sub)gradient of the primal form may be used to update the parameter $\mathbf{w}_i$. Learning in the dual form can be achieved by performing OLaRank [19] on elements in the reservoir. Details are provided in Section 3.1.

2.3 Regret Bound

Shai *et al.* [20] introduced a regret for online learning for binary classification. Bordes *et al.* [19] extended the regret bound to structured online learning. Based on the work in [20, 19], we derive the following bound on the weighted regret as a theoretical guarantee of performance. Without losing generality, from now on, we assume convexity on $\Omega(\mathbf{w})$ and assume a fixed size for the reservoir at any iteration i. That is, denote $N=|R_i|<m$ for all i.

Theorem 1 (Weighted Regret Bound). *At each iteration i we perform* OLaRank*[19] on each training example $\mathbf{b}_{ij}\in R_i$, where R_i is the reservoir at iteration i, $1\le j\le N$. Assume that for each update, the dual objective value increase after seeing the example $\mathbf{b}_{ij}$ is at least $C\mu(\ell(\mathbf{w}_{ij},\mathbf{b}_{ij}))$, with*

$$\mu(x)=\frac{1}{C}\min(x,C)(x-\frac{1}{2}\min(x,C)),$$

then, we have for any $\mathbf{w}$,

$$\frac{1}{m}\sum_{i=1}^{m}\frac{1}{N}\,\mathbb{E}_{R_i}\Big[\sum_{j=1}^{N}\ell(\mathbf{w}_{ij},\mathbf{b}_{ij})\Big]\le\frac{1}{m}\sum_{i=1}^{m}\Big(\frac{1}{Z_i}\sum_{j=1}^{i}\lambda_j\ell(\mathbf{w},\mathbf{z}_j)\Big)+\frac{\Omega(\mathbf{w})}{CmN}+\frac{C}{2},$$

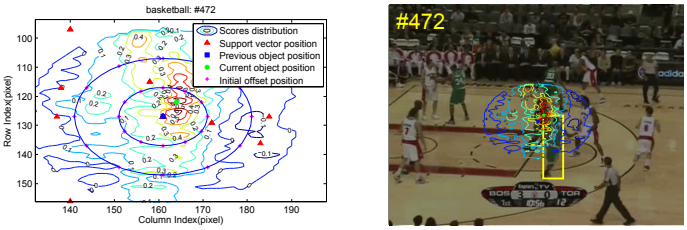

Fig. 1. An illustration of greedy inference. **LEFT**: a contour plot of f in (8) for varying $\mathbf{y}'_t$. The centre blue solid square indicates the position of object in previous frame. The centre point and some random points (purple) are used as the initial positions. The $\mathbf{y}^*_t$ with the highest f is obtained (green solid point) via Algorithm 2. **RIGHT**: The contour plot and true bounding box shown over the relevant frame of the video.

where $Z_i = \sum_{j=1}^{i} \lambda_j$.

The proof is provided in supplementary material.

3 Tracking with Weights

In tracking, $\mathbf{z} = (\mathbf{x}, \mathbf{y}, \mathbf{y}') \in R_t$ as mentioned earlier. Thus we have the loss

$$\ell(\mathbf{w}, (\mathbf{x}, \mathbf{y}, \mathbf{y}')) = [\Delta(\mathbf{y}, \mathbf{y}') - \langle \Psi(\mathbf{x}, \mathbf{y}), \mathbf{w} \rangle + \langle \Psi(\mathbf{x}, \mathbf{y}'), \mathbf{w} \rangle]_+,$$

where $[x]_+ = \max(0, x)$. The risk at the t-th frame becomes

$$C \sum_{(\mathbf{x}, \mathbf{y}, \mathbf{y}') \in R_t} \ell(\mathbf{w}, (\mathbf{x}, \mathbf{y}, \mathbf{y}')) + \Omega(\mathbf{w}).$$

3.1 Kernelisation

Updating the dual variable $\boldsymbol{\alpha}$ enables us to use kernels, which often produce superior results in many computer vision tasks. Using a similar derivation to that introduced by [18], we rewrite the optimisation problem of (2) by introducing coefficients $\beta_i^{\mathbf{y}}$ for each $(\mathbf{x}_i, \mathbf{y}_i, \mathbf{y}'_i) \in R_t$, where $\beta_i^{\mathbf{y}} = -\alpha_i^{\mathbf{y}}$ if $\mathbf{y} \neq \mathbf{y}_i$ and $\sum_{\bar{\mathbf{y}} \neq \mathbf{y}_i} \alpha_i^{\bar{\mathbf{y}}}$ otherwise and denote $\boldsymbol{\beta}$ as the vector of the dual variables β,

$$\max_{\boldsymbol{\beta}} -\sum_{i,\mathbf{y}} \Delta(\mathbf{y}, \mathbf{y}_i) \beta_i^{\mathbf{y}} - \frac{1}{2} \sum_{i,\mathbf{y},j,\bar{\mathbf{y}}} \beta_i^{\mathbf{y}} \beta_j^{\bar{\mathbf{y}}} k((\mathbf{x}_i, \mathbf{y}), (\mathbf{x}_j, \bar{\mathbf{y}})) \tag{7}$$

$$s.t.\ \forall i, \forall \mathbf{y} : \beta_i^{\mathbf{y}} \leq \delta(\mathbf{y}, \mathbf{y}_i) C;\ \forall i : \sum_{\mathbf{y}} \beta_i^{\mathbf{y}} = 0,$$

where $\delta(\mathbf{y}, \mathbf{y}_i)$ is 1 when $\mathbf{y} = \mathbf{y}_i$ and 0 otherwise, and $k(\cdot, \cdot)$ is the kernel function. The discriminant scoring function becomes

$$f(\mathbf{x}, \mathbf{y}) = \sum_{i,\bar{\mathbf{y}}} \beta_i^{\bar{\mathbf{y}}} k((\mathbf{x}_i, \bar{\mathbf{y}}), (\mathbf{x}, \mathbf{y})). \tag{8}$$

Algorithm 1 Online Dual Update for Tracking with Weights

Require: current frame $\mathbf{x}_t$, offset $\mathbf{y}_t$, previous dual variable $\boldsymbol{\beta_{t-1}}$, reservoir R_{t-1}, the preset reservoir size N, number of offsets n_t, and the preset time factor $q > 1$ (we use $q = 1.8$ in our experiments).

Generates n_t offsets $\{\mathbf{y}_{tj}\}_{j=1}^{n_t}$ in the vicinity of the $\mathbf{y}_t$ and $R_t = R_{t-1}$.
for each $\mathbf{y}_{tj}$ **do**
  Calculate a key $v_{tj} = u_{tj}^{1/\lambda_t}$, where $\lambda_t = q^t$ and $u_{tj} \sim \text{Uniform}(0, 1)$.
  if $|R_t| < N$ **then**
    $R_t = R_t \cup (\mathbf{x}_t, \mathbf{y}_t, \mathbf{y}_{tj})$. [In implementation, only pointers are stored]
  else
    Find the element $(\mathbf{x}, \mathbf{y}, \mathbf{y}')$ in R_t with the smallest key denoted as v^*.
    If $v_{tj} > v^*$, replace $(\mathbf{x}, \mathbf{y}, \mathbf{y}')$ in R_t with $(\mathbf{x}_t, \mathbf{y}_t, \mathbf{y}_{tj})$
  end if
end for
for each $(\mathbf{x}, \mathbf{y}, \mathbf{y}') \in R_t$ **do**
  Perform OLaRank to update $\boldsymbol{\beta_{t-1}}$.
end for
$\boldsymbol{\beta_t} = \boldsymbol{\beta_{t-1}}$
return R_t and $\boldsymbol{\beta_t}$

As in [14], at the t-th frame, n_t many $\mathbf{y}_{ti}$ are generated. We update the reservoir and perform OLaRank on the elements of the reservoir. Pseudo code for online update of the dual variables and the reservoir is provided in Algorithm 1. By Theorem 1 we know the below holds (for the proof see supplementary material).

Corollary 1 (Tracking Bound). *Assume that for each update in Algorithm 1, the dual increase after seeing the example $\mathbf{b}_{ij} \in R_i$ is at least $C\mu(\ell(\mathbf{w}_{ij}, \mathbf{b}_{ij}))$, then we have for any $\mathbf{w}$,*

$$\frac{1}{tN}\sum_{i=1}^{t}\mathbb{E}_{R_i}\Big[\sum_{j=1}^{N}\ell(\mathbf{w}_{ij},\mathbf{b}_{ij})\Big] \leq \frac{1}{t}\sum_{i=1}^{t}\Big(\frac{1}{Z_i}\sum_{j=1}^{i}\lambda_j\sum_{l=1}^{n_j}\ell(\mathbf{w},\mathbf{x}_j,\mathbf{y}_j,\mathbf{y}_{jl})\Big) + \sqrt{\frac{2\Omega(\mathbf{w})}{tN}},$$

where $Z_i = \sum_{j=1}^{i}\lambda_j n_j$.

3.2 Greedy Inference

Given f, we can predict $\mathbf{y}_t$ using (1) and (8). The exhaustive search used in [14] is computationally expensive, because kernel evaluations on all support vectors and all possible $\mathbf{y} \in \mathcal{Y}$ are required. Note, however, that the contour of (8) is very informative and can guide the moving of the offset candidate $\mathbf{y}'_t$. We have therefore developed a significantly faster greedy search approach using the contour information in Algorithm 2 to predict offset $\mathbf{y}^*_t$. The current bounding box is predicted via $B_t = B_{t-1} + \mathbf{y}^*_t$. An example is given in Fig. 1 for the inference. The experiments in Section 4.2 show that our greedy inference accelerates the speed of prediction without compromising predictive accuracy.

Algorithm 2 Greedy inference for prediction

Require: Frame $\mathbf{x}_t$, B_{t-1}, search radius s, and the number of initial points n.

1: Generate a set $\hat{\mathcal{Y}} = \{\mathbf{y}_{tj} = (\Delta c_{tj}, \Delta l_{tj}, 0, 0)\}_{j=1}^{n}$, whose elements are randomly generated within the radius s, that is $(\Delta c_{tj})^2 + (\Delta l_{tj})^2 < s^2$.
2: **for** each offset $\mathbf{y}_{tj}$ **do**
3: $\mathbf{y}_{tj} = \mathbf{y}_{tj} + \Delta\mathbf{y}^*$ where $\Delta\mathbf{y}^* = \text{argmax}_{\Delta\mathbf{y}} f(\mathbf{x}_t, \mathbf{y}_{tj} + \Delta\mathbf{y})$ as in (8). Here $\Delta\mathbf{y} \in \{(\Delta\mathbf{y}_c, \Delta\mathbf{y}_l, 0, 0) | \Delta\mathbf{y}_c, \Delta\mathbf{y}_l = 0, 1, -1\}$ and same location won't be re-checked.
4: **end for**
5: **return** $\mathbf{y}_t^* = \text{argmax}_{\mathbf{y} \in \hat{\mathcal{Y}}} f(\mathbf{x}_t, \mathbf{y})$. [search over $\hat{\mathcal{Y}}$ can be avoided by updating the maximum value and index]

4 Experiments

In this section we first demonstrate the performance impact of different reservoir sizes, and different numbers of initial offsets on greedy inference. We then present qualitative and quantitative tracking results for the proposed method and its competitors on 12 challenging sequences.

Experiment setup In order to enable a fair comparison, our tracker uses the same features as Struck [14], MIL [11] and OAB [9]. The search radius is set to 30 pixels for all sequences. As will be discussed in Section 4.1, the maximum reservoir size is set to 200, time factor q is set to 1.8, and the weights to q^t. The number of initial offsets for inference is set to 48 for all sequences. Furthermore, the proposed tracking algorithm is implemented in C++ on a workstation with an Intel Core 2 Duo 2.66GHz processor and 4.0G RAM. Without specific code optimisation, the average running time of our tracker is about 0.08 seconds per frame in the above setting.

Test sequences and compared algorithms Tracking evaluation uses 12 challenging sequences with $6,435$ frames in total. The *coke*, *tiger1*, *tiger2*, and *sylv* sequences have been used in [14]. The *basketball*, *shaking*, *singer1* sequence were obtained from [4]. Other sequences were obtained from [1]. These video sequences contain a variety of scenes and object motion events. They present challenging lighting, large variation in pose and scale, frequent half or full occlusions, fast motion, shape deformation and distortion, and similar interference. Our tracker is compared with 8 of the latest state-of-the-art tracking methods named Struck(structured output tracker [14]), MIL(multiple instance boosting-based tracker [11]), OAB(online AdaBoost [9]), Frag(Fragment-based tracker [2]), IVT(incremental subspace visual tracker [3]), VTD(visual tracking decomposition [4]), L1T(L1 minimisation tracker [6]). These trackers were evaluated on the basis of their publicly available source codes from the original authors. For OAB, there are two different versions with $r = 1$ (OAB1) and $r = 5$ (OAB5) [11].

Evaluation criteria For quantitative performance comparison we use three popular evaluation criteria, *i.e.* centre location error (CLE), Pascal VOC overlap ratio(VOR), VOC tracking success rates. VOC overlap ratio is defined as

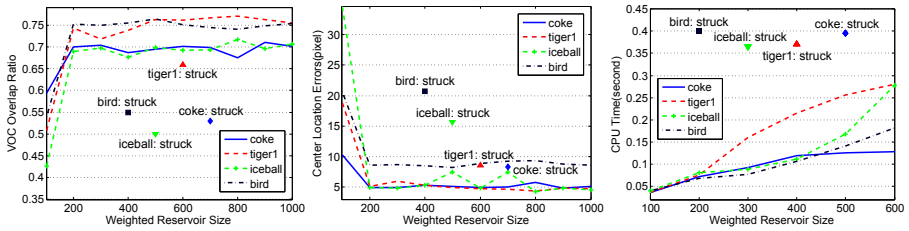

Fig. 2. Performance of our tracker with various reservoir sizes on four sequences(*coke, tiger1, iceball, bird*). **LEFT and MIDDLE**: the average VOR and CLE. **RIGHT**: the run time. The square, diamond, upward-pointing and down-pointing triangle in three plots correspond to the performance of Struck on four sequences, respectively.

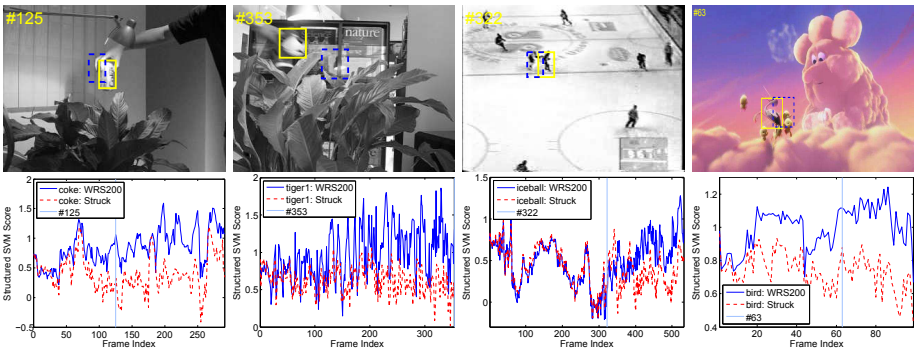

Fig. 3. Comparison of structured SVM scores and tracking results under various scenes. **TOP ROW**: the tracking result of our tracker with reservoir size $N = 200$ and Struck, where the solid rectangle shows result of our tracker and the dashed rectangle marks the result of Struck. **BOTTOM ROW**: structured SVM score over time for our tracker and Struck. Ours always has a higher score.

$R_{overlap} = Area(B_T \cap B_{GT})/Area(B_T \cup B_{GT})$, where B_T is the tracking bounding box and B_{GT} the ground truth bounding box. If the VOC overlap ratio is larger than 0.5, it is considered to be successful in visual tracking for each frame.

4.1 Effect of Differing Reservoir Sizes

To show the effect of the reservoir size we use ten different reservoir sizes N, from 100 to 1000. We compute the average CLE, VOR and CPU time for each sequence with each different reservoir size, and plot in Fig. 2. We see that for $100 \leq N \leq 200$, increasing the reservoir size N increases the CLE and VOR significantly. However, for $N > 200$, CLE and VOR do not change much. Meanwhile, the CPU run time also increases as the reservoir size increases as expected. The result of Struck is also shown in Fig. 2.

Evaluation of structured SVM score Fig. 3 shows the quantitative structured SVM scores and qualitative tracking results of our tracker and Struck. Due

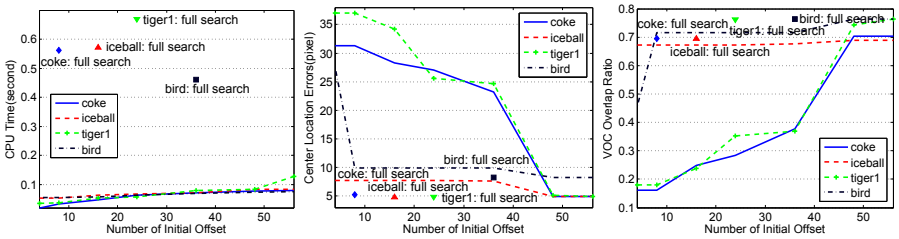

Fig. 4. Quantitative evaluation of tracking performance with different number of initial offsets for greedy inference. **LEFT**: the CPU time of the proposed tracker with greedy inference under various initial offsets on four video sequences. **MIDDLE and RIGHT**: the average CLE and VOR. The square, diamond, upward-pointing and down-pointing triangle in three plots correspond to the performance of the proposed tracker with full search on four video sequences, respectively.

to the observation in Fig. 2, we set the reservoir size to 200. As we can see, our tracker always has higher SVM score than Struck under the same setting (*i.e.* same feature, kernel and regularisation). Our tracker manages to keep tracking of the object even under challenging illumination (*coke*), fast movement (*tiger1*), similar interference (*iceball*) and occlusion (*bird*).

4.2 Effect of the Number of Initial Offsets on Inference

The number of initial offsets for greedy inference affects tracking speed and accuracy. To investigate this, we fix the reservoir size to 200, and use the same four video sequences as Fig. 2. A quantitative evaluation is shown in Fig. 4 under different number of initial offsets that equal to 4, 8, 16, 24, 36, 48 and 56. As can be seen in the left plot of Fig. 4, the CPU time for full search is significantly higher than that for the greedy algorithm on four video sequences. In some cases, the performance will not be boosted as the number of initial offset increases. For example, performance varies little with varying numbers of initial offsets on the *iceball* and *bird* sequences. This is due to simple scene background and slow the object movement. In this case, the local maximum of structured SVM score is often the global optimal. Furthermore, when the number of initial offsets is already large (*e.g.* ≥ 48), no significant improvement is obtained via increasing the number of initial offsets (see Fig. 4).

4.3 Comparison of Competing Trackers

We compare our tracker to eight state-of-the-art trackers. For visualisation purpose, we only show the results of best four methods in Fig. 5. The full qualitative tracking results of all the nine trackers over twelve challenging video sequences are provided in supplementary material.

Pose and scale changes As can be seen in first row of Fig. 5, our algorithm exhibits the best tracking performance amongst the group. In the *box* sequence,

Table 1. Compared average VOC overlap ratio on 12 sequences

Sequence	Ours	Struck	MIL	OAB1	OAB5	Frag	IVT	VTD	L1T
Coke	**0.69±0.15**	0.54±0.17	0.35±0.22	0.18±0.18	0.17±0.16	0.13±0.16	0.08±0.22	0.09±0.22	0.05±0.20
Tiger1	**0.75±0.13**	0.66±0.22	0.61±0.18	0.44±0.23	0.24±0.26	0.21±0.29	0.02±0.12	0.10±0.24	0.45±0.37
Tiger2	**0.68±0.15**	0.57±0.18	0.63±0.14	0.37±0.25	0.18±0.20	0.15±0.23	0.02±0.12	0.20±0.23	0.27±0.32
Sylv	0.67±0.11	**0.68±0.14**	0.67±0.18	0.47±0.38	0.12±0.12	0.60±0.23	0.06±0.16	0.58±0.31	0.45±0.35
Iceball	**0.69±0.18**	0.50±0.33	0.37±0.28	0.08±0.22	0.40±0.30	0.52±0.31	0.03±.12	0.56±0.28	0.03±0.10
Basketball	**0.72±0.14**	0.45±0.34	0.15±0.27	0.12±0.23	0.12±0.24	0.46±0.35	0.02±0.07	0.64±0.10	0.03±0.10
Shaking	**0.72±0.14**	0.10±0.18	0.60±0.26	0.56±0.28	0.50±0.21	0.33±0.27	0.03±0.11	0.69±0.14	0.05±0.16
Singer1	**0.71±0.08**	0.33±0.36	0.18±0.32	0.17±0.32	0.07±0.19	0.13±0.26	0.20±0.28	0.49±0.20	0.18±0.32
Bird	**0.75±0.10**	0.55±0.25	0.58±0.32	0.56±0.28	0.59±0.30	0.34±0.32	0.08±0.19	0.10±0.26	0.44±0.37
Box	**0.74±0.17**	0.29±0.38	0.10±0.18	0.16±0.27	0.08±0.18	0.06±0.17	0.46±0.29	0.41±0.34	0.13±0.26
Board	**0.69±0.25**	0.39±0.29	0.16±0.27	0.12±0.24	0.17±0.26	0.49±0.25	0.10±0.21	0.31±0.27	0.11±0.27
Walk	**0.63±0.18**	0.46±0.32	0.50±0.34	0.52±0.36	0.49±0.34	0.08±0.25	0.05±0.14	0.08±0.23	0.09±0.25
Average	**0.71**	0.46	0.41	0.31	0.26	0.29	0.10	0.35	0.19

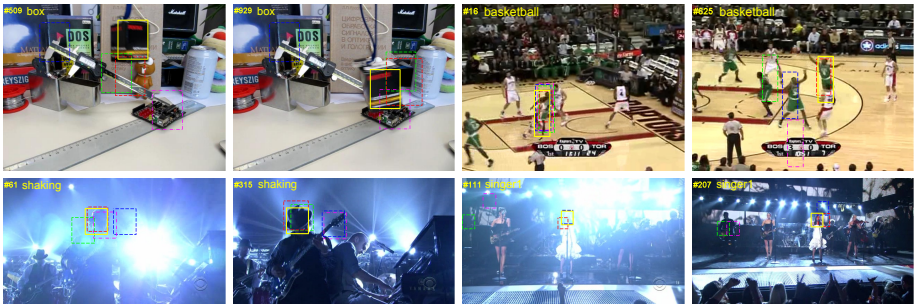

Fig. 5. Visual tracking result. Yellow, blue, green, magenta and red rectangles show results by our tracker, Struck, MIL, Frag and VTD methods, respectively.

the other four trackers cannot keep track of the object after rotation, as they are not effective in accounting for appearance change due to large pose change. In the *basketball* sequences, Struck, MIL and Frag methods lose the object completely after frame 476 due to the rapid appearance change of object. The VTD tracker gets an average error similar to ours on the *basketball* sequence, but its VOC overlap ratio is worse (see Tab. 2 and Tab. 1).

Illumination variation and occlusion The second row of Fig. 5 illustrates performance under heavy lighting and occlusion scenes. The MIL and Frag methods drift to a background area in early frames in the *singer1* sequence as they are not designed to handle full occlusions. Although Struck maintains track under strong illumination (frame 111 in the *singer1* sequence), it loses the object completely after illumination changes. As can be seen in the right image of Fig. 3, frequent occlusion also affects the performance of Struck. On the contrary, our tracker adapts to the appearance changes, and achieves robust tracking results under illumination variation and partial or full occlusions.

Quantitative comparisons to competing trackers Tab. 1 and Tab. 2 show the average VOR and CLE (best results are highlighted in bold) obtained by our tracker and other eight competitor trackers. Because several trackers in-

Table 2. Compared average centre location errors(pixel) on 12 sequences

Sequence	Ours	Struck	MIL	OAB1	OAB5	Frag	IVT	VTD	L1T
Coke	**5.0±4.5**	8.3±5.6	17.5±9.5	25.2±15.4	56.7±30.0	61.5±32.4	10.5±21.9	47.3±21.9	55.3±22.4
Tiger1	**5.1±3.6**	8.6±12.1	8.3±6.6	17.6±16.9	39.2±32.8	40.1±25.6	56.3±20.6	68.7±36.2	23.2±30.6
Tiger2	**6.1±3.7**	8.5±6.0	7.0±3.7	19.7±15.2	37.3±27.0	38.7±25.1	92.7±34.7	37.0±29.7	33.1±28.1
Sylv	8.8±3.5	**8.4±5.3**	8.7±6.4	33.0±36.6	75.3±35.3	12.6±11.5	73.9±37.6	21.6±35.9	21.7±32.6
Iceball	**4.9±3.6**	15.6±22.1	61.6±85.6	97.7±53.4	58.1±84.0	40.3±72.9	90.2±67.5	13.5±26.0	77.1±50.0
Basketball	**9.6±6.2**	89.6±163.7	163.9±104.0	167.8±100.6	199.5±99.9	53.9±59.8	31.5±36.7	9.8±5.0	120.1±66.2
Shaking	**9.2±5.7**	123.1±54.3	37.8±75.6	26.9±49.2	29.1±48.7	47.1±40.5	94.3±36.8	12.5±6.7	132.8±56.9
Singer1	**5.1±1.8**	29.2±23.6	187.9±120.1	189.4±114.3	158.0±48.6	172.5±94.5	17.3±13.0	10.5±7.5	108.5±78.5
Bird	**8.6±4.1**	20.7±14.7	49.0±85.3	47.9±87.7	48.6±86.2	50.0±43.3	107.7±57.6	143.9±79.3	46.3±49.3
Box	**13.4±15.7**	162.2±132.8	140.9±77.0	153.5±95.8	165.4±84.1	147.0±67.8	34.7±43.8	63.5±65.7	150.0±82.7
Board	**32.9±32.5**	85.3±56.5	211.7±124.8	216.5±111.6	213.7±96.1	65.9±48.6	223.8±97.5	112.3±66.9	240.5±96.2
Walk	**11.3±7.7**	36.8±46.4	33.8±47.3	35.5±49.0	36.9±48.5	102.6±46.1	90.2±56.5	100.9±46.9	79.6±41.3
Average	**9.9**	49.6	77.3	85.8	93.1	69.3	76.9	53.4	90.6

Table 3. Compared success rates on 12 sequences

Sequence	Ours	Struck	MIL	OAB1	OAB5	Frag	IVT	VTD	L1T
Coke	**0.95**	0.71	0.27	0.05	0.03	0.06	0.08	0.08	0.05
Tiger1	**0.96**	0.82	0.83	0.41	0.18	0.19	0.01	0.13	0.47
Tiger2	**0.85**	0.66	0.81	0.34	0.12	0.09	0.01	0.15	0.30
Sylv	**0.93**	0.90	0.76	0.52	0.01	0.69	0.03	0.67	0.53
Iceball	**0.86**	0.61	0.30	0.09	0.42	0.67	0.02	0.76	0.02
Basketball	**0.90**	0.61	0.16	0.13	0.13	0.56	0.01	**0.90**	0.02
Shaking	**0.96**	0.07	0.78	0.55	0.46	0.29	0.01	0.86	0.02
Singer1	**0.99**	0.39	0.21	0.21	0.07	0.17	0.18	0.63	0.26
Bird	**0.97**	0.53	0.73	0.74	0.75	0.34	0.06	0.14	0.48
Box	**0.92**	0.36	0.06	0.19	0.07	0.04	0.55	0.48	0.15
Board	**0.70**	0.29	0.13	0.08	0.09	0.45	0.09	0.23	0.12
Walk	**0.76**	0.58	0.67	0.68	0.62	0.10	0.03	0.08	0.10
Average	**0.90**	0.54	0.48	0.33	0.25	0.30	0.09	0.43	0.21

volve some randomness, we also show the standard deviation of each evaluation result. Tab. 3 reports the success rates of the nine trackers over the twelve video sequences. From Tab. 1, Tab. 2 and Tab. 3, we found that the proposed tracking algorithm achieves the best tracking performance on most video sequences. Furthermore, we report the CLE of tracking result in supplementary material[3].

5 Conclusion

We have developed a novel approach to online tracking which allows continuous development of the target appearance model, without neglecting information gained from previous observations. The method is extremely flexible, in that the weights between observations are freely adjustable, and it is applicable to binary, multiclass, and even structured output labels in both linear and non-linear kernels. In formulating the method we also developed a novel principled method of optimising weighted risk using weighted reservoir sampling. We also derived a weighted regret bound in order to provide a performance guarantee, and showed that the tracker outperformed the state-of-the-art methods.

[3] The demonstration videos are available at `http://www.youtube.com/woltracker12`.

Acknowledgments. This work was partially supported by Australian Research Council grants DP1094764, LE100100235, DE120101161, and DP11010352. It was also partially supported by NSF (60872145, 60903126), National High-Tech. (2009AA01Z315), Postdoctoral Science Foundation (20090451397, 201003685) and Cultivation Fund from Ministry of Education (708085) of China. R. Yao's contribution was made when he was a visiting student at the University of Adelaide.

References

1. Wang, S., Lu, H., Yang, F., Yang, M.H.: Superpixel tracking. In: Proc. ICCV. (2011) 1323 –1330
2. Adam, A., Rivlin, E., Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In: Proc. CVPR. Volume 1. (2006) 798 – 805
3. Ross, D.A., Lim, J., Lin, R.S., Yang, M.H.: Incremental learning for robust visual tracking. Int. J. Comput. Vision **77** (2008) 125–141
4. Kwon, J., Lee, K.M.: Visual tracking decomposition. In: Proc. CVPR. (2010)
5. García Cifuentes, C., Sturzel, M., Jurie, F., Brostow, G.J.: Motion models that only work sometimes. In: Proc. BMVC. (2012)
6. Mei, X., Ling, H.: Robust visual tracking and vehicle classification via sparse representation. IEEE Trans. on PAMI. **33** (2011) 2259 –2272
7. Li, H., Shen, C., Shi, Q.: Real-time visual tracking using compressive sensing. In: Proc. CVPR. (2011) 1305 –1312
8. Grabner, H., Leistner, C., Bischof, H.: Semi-supervised on-line boosting for robust tracking. In: Proc. ECCV. (2008) 234–247
9. Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via on-line boosting. In: Proc. BMVC. Volume 1. (2006) 47–56
10. Avidan, S.: Ensemble tracking. IEEE Trans. on PAMI. **29** (2007) 261 –271
11. Babenko, B., Yang, M.H., Belongie, S.: Visual Tracking with Online Multiple Instance Learning. In: Proc. CVPR. (2009) 983 – 999
12. Li, X., Shen, C., Shi, Q., Dick, A.R., van den Hengel, A.: Non-sparse linear representations for visual tracking with online reservoir metric learning. In: Proc. CVPR. (2012)
13. Li, X., Dick, A.R., Wang, H., Shen, C., van den Hengel, A.: Graph mode-based contextual kernels for robust svm tracking. In: Proc. ICCV. (2011) 1156–1163
14. Hare, S., Saffari, A., Torr, P.: Struck: Structured output tracking with kernels. In: Proc. ICCV. (2011) 263–270
15. Avidan, S.: Support vector tracking. IEEE Trans. on PAMI. **26** (2004) 1064 –1072
16. Blaschko, M.B., Lampert, C.H.: Learning to localize objects with structured output regression. In: Proc. ECCV. Volume 5302. (2008) 2–15
17. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. JMLR **6** (2005) 1453–1484
18. Bordes, A., Bottou, L., Gallinari, P., Weston, J.: Solving multiclass support vector machines with larank. In: Proc. ICML. (2007) 89–96
19. Bordes, A., Usunier, N., Bottou, L.: Sequence labelling svms trained in one pass. In: Proc. ECML/PKDD. (2008) 146–161
20. Shalev-Shwartz, S., Singer, Y.: A primal-dual perspective of online learning algorithms. Machine Learning **69** (2007) 115–142
21. Efraimidis, P.S., Spirakis, P.G.: Weighted random sampling with a reservoir. Inf. Process. Lett. **97** (2006) 181–185