Part-based Visual Tracking with Online Latent Structural Learning

Rui Yao[1], Qinfeng Shi[2], Chunhua Shen[2], Yanning Zhang[1], Anton van den Hengel[2]
[1] School of Computer Science, Northwestern Polytechnical University, China
[2] School of Computer Science, The University of Adelaide, Australia
yaorui@mail.nwpu.edu.cn, {javen.shi,chunhua.shen,anton.vandenhengel}@adelaide.edu.au

Abstract

*Despite many advances made in the area, deformable targets and partial occlusions continue to represent key problems in visual tracking. Structured learning has shown good results when applied to tracking whole targets, but applying this approach to a part-based target model is complicated by the need to model the relationships between parts, and to avoid lengthy initialisation processes. We thus propose a method which models the unknown parts using latent variables. In doing so we extend the online algorithm pegasos to the structured prediction case (*i.e.*, predicting the location of the bounding boxes) with latent part variables. To better estimate the parts, and to avoid over-fitting caused by the extra model complexity/capacity introduced by the parts, we propose a two-stage training process, based on the primal rather than the dual form. We then show that the method outperforms the state-of-the-art (linear and non-linear kernel) trackers.*

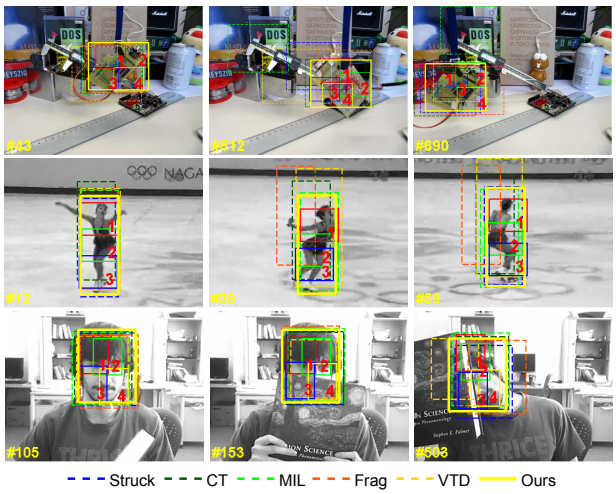

Figure 1: Qualitative tracking results of our tracker and competing trackers over representative frames of three sequences. These sequences contain heavy pose variation (*board*, the first row), shape deformation (*fskater*, the second row), partial occlusion (*faceocc2*, the third row), and *etc*. Note that the number marked on small rectangles represents the index of object parts.

1. Introduction

Visual tracking is a goal in itself, but is also a preprocessing step before further analysis of the activities, behaviours, interactions and relationships between objects of interest. Recent progress in object tracking has yielded a steady increase in performance, but designing a robust algorithm to track generic objects in the presence of partially occluded and deformable targets is still a major challenge (as shown in Fig. 1). This additional difficulty falls on top of those that regularly challenge visual tracking including significant variation in appearance due to factors such as changing pose, viewpoint, and illumination.

To address the appearance variation challenge researchers have developed sophisticated appearance models and methods of adapting these models during tracking. One solution is to build a robust object appearance model and find the best candidate image patch to match the model, see, for example, incremental subspace learning [18], integral histogram tracking [1], visual tracking decomposition [13], and sparse representation tracking [16, 14]. Another approach is to model both the object and the background, and then to distinguish the object from the background using a discriminative classifier [3, 8, 9, 25, 4, 10]. Avidan [3], for example, used a Support Vector Machine as an off-line binary classifier to distinguish target from background. In [4], Babenko *et al*. employ an online Multiple Instance Learning based appearance model to resolve the sample ambiguity problem. Hare *et al*. [10] applied structured learning to (whole) visual object tracking, which builds upon its previous successful application to object detection [5]. This category of approach is termed tracking-by-detection. All of these methods, however, delineate the tracked object by a single regular bounding box (*e.g.*, a rectangular or circular window), which renders them sensitive to partial occlusions and shape deformations.

Deformable part-based models have been successfully applied to object detection [7, 29, 15] and object recognition [2, 19] on numerous occasions. Part-based appearance models [7, 15, 29, 19, 2, 17] have been shown to have

favourable properties such as robustness to partial occlusions [7] and articulation [29]. In those cases, highly variable objects are represented using mixtures of deformable part-based models, which are parametrised by the appearance of each part and a geometric model capturing spatial relationships among parts.

The difficulty in extending part-based appearance models to visual tracking is that adding parts dramatically increases the complexity of the model, and the model needs to be updated online. In addition, explicitly tracking each part would require individual training and initialisation for each part, which would significantly limit the practicality of an online method. We avoid this problem here by instead using latent variables in the representation of each part, thereby avoiding the need for additional interaction. As discussed above, structured SVMs have shown very promising results for whole object tracking. The difficulty with using latent variables in a structured SVM, however, is that the underlying optimisation problem ceases to be convex. This is a problem because structured SVM based tracking methods have thus far optimised the dual form of the underlying problem in order to reduce computational cost. Non-convexity, however, means that the optimal solutions of the primal and dual do not coincide, thus rendering the existing dual-based approach inapplicable.

Developing a practical, online, structured SVM visual tracking method which uses a part-based appearance model thus requires solving the primal form of the underlying optimisation problem. In this paper, an object is made up of a set of parts, each with an associated weight. We employ an online structured output learning with latent variables to learn the weight parameters for an object and its parts, and distinguish the target object from the background using the weight parameters consequently. During tracking, the target object location is estimated by searching for the maximum classification score in the vicinity around the estimate from the previous frame, where the classification score is composed of the score of object and each parts. In doing so, our method possesses the flexibility of object appearance representation by part-based model and the separability of the target object and its background by the power of discriminative learning. Fig. 1 shows some tracking results obtained by our algorithm.

Contributions (1) Unlike existing *offline* latent SVMs in object detection [7, 27, 29], we propose an *online* latent structured SVM for visual tracking. (2) We propose two-stage training: the stage 1 tracks the parts and estimates the part parameter, and the stage 2 tracks the object and estimate the object parameter and correlation parameter. The two-stage training outperforms training all parameters as a whole. (3) We design a part-based linear kernel tracker that has competitive and often better tracking results than non-linear kernel trackers. Our tracker adapts quickly to appearance changes such as partial occlusions and deformations.

1.1. Related Work

Deformable part-based appearance models In [2], Amit *et al*. formulate a deformable template model for an object, where the deformation of an object is defined in terms of the locations of several reference points. Each reference point is associated with a part. Felzenszwalb *et al*. [7], in contrast, develop a multi-scale deformable part model to detect and localise objects of a generic category. The part model is composed of a coarse global template for entire object and higher resolution part templates. However, the part model in [7] requires careful initialisation and complex training. To overcome those problems, Zhu *et al*. [29] present a hierarchical tree model to represent an object, where the nodes of the tree represent object parts.

Discriminative learning with latent variables Motived by object detection with parts, Felzenszwalb *et al*. [7] introduce latent variables into discriminative training models for binary classification. They propose a latent binary SVM to handle object parts, which are not labelled during training. Inspired by this work, Yu *et al*. [27] extend this latent approach to use structured prediction, and solve the ensuing non-convex optimisation problem using Concave-Convex Procedure (CCCP). In [29] the viewpoint and the positions of object parts are treated as structural latent variables, and the latent structural SVM optimised in its dual form using an incremental CCCP algorithm. Vedaldi *et al*. [23] also use a latent SVM framework, but develop a structured output model for detection with partial truncation. Although all of these part-based object detection frameworks use latent variables as part of a structured SVM, they rely on the availability of a model at training time, and thus cannot easily be used for online object tracking.

Object tracking with parts In [11], each part is tracked independently, and the results treated as multiple measurements. Tracking is then achieved by identifying inconsistent measurements. Nejhum *et al*. [20] model the foreground object shape in terms of a small number of rectangular blocks. The algorithm tracks objects by matching foreground intensity histograms, and updating the part-based appearance model on-the-fly. However, these methods require manual initialisation of part locations carefully. Kwon *et al*. [12] represent an object by a fixed number of local patches and update the model during tracking. Their approach requires prior knowledge of local patches.

Structured output tracking In [5], Balschko *et al*. propose an object detection algorithm using a structured output SVM [22]. Motived by this success Hare *et al*. [10] apply structured learning to *online* visual tracking. In [26], a weighted online structural learning approach is used to deal with the inevitable changes in target appearance over time. In comparison, the method we propose adds a part-based

model to a discriminative training framework, and solves the ensuing linear structured SVM optimisation problem in primal form.

2. Modelling Tracking with Unknown Parts

In this section, we show how to track an object with unknown parts. In Section 2.1, we will introduce the part-based model, and in Section 2.2, we describe a simple way to train the model. After showing that this simple training mechanism is not as effective as may be hoped we develop in Section 3 a more effective two-stage training process.

2.1. Representation

The method proceeds as follows: a bounding box on the target object is given in the 1st frame of the video, and the tracker is then required to track the object (by predicting a bounding box containing the object) from the 2nd frame to the end of the video. At the t-th frame, the bounding box of the target object is represented by $B_t = (c_t, r_t, w_t, h_t)$, where c_t, r_t are the column and row coordinates of the upper-left corner, and w_t, h_t are the width and height. The offset is $\mathbf{y}_t = (\Delta c_t, \Delta r_t, \Delta w_t, \Delta h_t) \in \mathcal{Y}$. As in [10, 26], we only consider a fixed size bounding box (*i.e.* $\Delta w_t = \Delta h_t = 0$), though in principle bounding boxes with varying sizes can be incorporated with slight modification.

We consider a part, which can contain a piece of object, or a piece of background or both, to be represented by a smaller bounding box $\mathbf{b}_t^j = (c_t^j, r_t^j, w_t^j, h_t^j)$, which we call *part box* from now on. Denote by M the number of parts, we use $\mathbf{z}_t = (\mathbf{z}_t^1, \cdots, \mathbf{z}_t^M) \in \mathcal{Z}$ to represent the offsets of M part boxes at the t-th frame. Here each $\mathbf{z}_t^j = (\Delta c_t^j, \Delta r_t^j, 0, 0), j \in \{1, \cdots, M\}$ is the offset of j-th part box $\mathbf{b}_t^j$, *i.e.* $\mathbf{b}_t^j = \mathbf{b}_{t-1}^j + \mathbf{z}_t^j$.

The task can be cast as that of learning a function $f : \mathcal{X} \times \mathcal{Y} \times \mathcal{Z} \to \mathbb{R}$, such that, given the t-th frame $\mathbf{x}_t \in \mathcal{X}$, and the bounding box B_{t-1} of the object in the $(t-1)$-th frame, the bounding box B_t in the t-th frame can be obtained via predicting offset $\mathbf{y}_t^*$,

$$B_t = B_{t-1} + \mathbf{y}_t^*, \tag{1}$$

$$\mathbf{y}_t^* = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} (\max_{\mathbf{z} \in \mathcal{Z}} f(\mathbf{x}_t, \mathbf{y}, \mathbf{z})). \tag{2}$$

As in many learning methods, we consider functions that are linear in some feature representation Φ,

$$f(\mathbf{x}_t, \mathbf{y}, \mathbf{z}; \mathbf{w}) = \langle \mathbf{w}, \Phi(\mathbf{x}_t, \mathbf{y}, \mathbf{z}) \rangle. \tag{3}$$

The feature Φ consists of three types of local features,

$$\Phi(\mathbf{x}_t, \mathbf{y}, \mathbf{z}) = \big[\phi_1(\mathbf{x}_t, \mathbf{z}^1), \phi_1(\mathbf{x}_t, \mathbf{z}^2), \cdots, \phi_1(\mathbf{x}_t, \mathbf{z}^M), \phi_2(\mathbf{x}_t, \mathbf{y}), \phi_3(\mathbf{y}, \mathbf{z}^1), \phi_3(\mathbf{y}, \mathbf{z}^2), \cdots, \phi_3(\mathbf{y}, \mathbf{z}^M)\big]. \tag{4}$$

Here $\phi_1(\mathbf{x}_t, \mathbf{z}^j)$ represents the appearance of the j-th part box, $\phi_2(\mathbf{x}_t, \mathbf{y})$ represents the appearance of the bounding box, and $\phi_3(\mathbf{y}, \mathbf{z}^j)$ reflects the compatibility between bounding box and the j-th part box. Letting

$$\mathbf{w} = [\mathbf{u}^1, \cdots, \mathbf{u}^M, \mathbf{v}^0, \mathbf{v}^1, \cdots, \mathbf{v}^M], \tag{5}$$

we have

$$f(\mathbf{x}_t, \mathbf{y}, \mathbf{z}; \mathbf{w}) = \sum_{j=1}^{M} \langle \mathbf{u}^j, \phi_1(\mathbf{x}_t, \mathbf{z}^j) \rangle + \langle \mathbf{v}^0, \phi_2(\mathbf{x}_t, \mathbf{y}) \rangle + \sum_{j=1}^{M} \langle \mathbf{v}^j, \phi_3(\mathbf{y}, \mathbf{z}^j) \rangle. \tag{6}$$

Features are detailed in Section 2.4.

2.2. Latent Pegasos for Training Online

As in existing tracking applications [10, 26], we would like the tracker to track the object from the 2nd frame on the basis of a bounding box of the target object provided in the 1st frame. That means that the true value of $\mathbf{z}$ is not available, as obtaining it would require more user input. The fact that users may well not be able to accurately divide an object into its component parts is also an issue. By defining $\mathbf{z}$ to be a latent variable avoids these problems.

Latent variables have been used with SVMs previously [7, 27, 29] but only in a batch learning scenario (which implies training offline). In order to allow the use of latent variables in an *online* SVM for visual tracking, we propose what can be seen an extension of online pegasos [21] (originally for binary or multi-class problems, but not latent variables) to structured output.

We assume $B_0 = B_1$ *i.e.* $\mathbf{y}_1 = (0, 0, 0, 0)$ and sample some $\mathbf{y} \neq \mathbf{y}_1$. The sampled $\mathbf{y}$s, and $\mathbf{y}_1$, are provided to our online learner to update $\mathbf{w}$. We then use this $\mathbf{w}$ to predict $\mathbf{y}_2$ to obtain tracking result for the 2nd frame *i.e.* the bounding box B_2. We then take the predicted $\mathbf{y}_2$ as the true label, sample $\mathbf{y} \neq \mathbf{y}_2$, and feed them into the online learner to update $\mathbf{w}$, and so on.

Denote $\mathbf{w}_t$ the parameter to predict $\mathbf{y}_t$ for the t-th frame. After predicting $\mathbf{y}_t$ via $f(\mathbf{x}_t, \mathbf{y}, \mathbf{z}; \mathbf{w}_t)$, we sample offsets $\{\mathbf{y}_{t,i} \neq \mathbf{y}_t\}_{i=1}^N$, and estimate $\mathbf{w}_{t+1}$ via

$$\mathbf{w}_{t+1} = \operatorname*{argmin}_{\mathbf{w}} g(\mathbf{w}, t), \tag{7}$$

where we consider the following objective

$$g(\mathbf{w}; t) = \frac{\lambda}{2} ||\mathbf{w}||^2 + \frac{1}{N} \sum_{i=1}^{N} \Big[\Delta(\mathbf{y}_t, \mathbf{y}_{t,i}) + \max_{\mathbf{z}'} \langle \mathbf{w}, \Phi(\mathbf{x}_t, \mathbf{y}_{t,i}, \mathbf{z}') \rangle - \max_{\mathbf{z}} \langle \mathbf{w}, \Phi(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}) \rangle \Big]_+. \tag{8}$$

Here $[a]_+ = \max(0, a)$. We refer the readers to supplementary material for how we get Eq. (7) and Eq. (8). The label

cost $\Delta(\mathbf{y}_t, \mathbf{y})$ measures dis-similarity between the true output $\mathbf{y}_t$ and a candidate output $\mathbf{y}$. Here we use

$$\Delta(\mathbf{y}_t, \mathbf{y}) = 1 - \frac{(B_{t-1} + \mathbf{y}_t) \cap (B_{t-1} + \mathbf{y})}{(B_{t-1} + \mathbf{y}_t) \cup (B_{t-1} + \mathbf{y})}, \quad (9)$$

which was introduced in [6] to measure the VOC (Visual Object Classes) bounding box overlap ratio. We consider the sub-gradient of the above objective,

$$\nabla_t = \lambda \mathbf{w}_t - \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\big[\max_{\mathbf{z}'} f(\mathbf{x}_t, \mathbf{y}_{t,i}, \mathbf{z}'; \mathbf{w}_t) - \max_{\mathbf{z}} f(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}; \mathbf{w}_t) + \Delta(\mathbf{y}_t, \mathbf{y}_{t,i}) > 0\big] \delta\Phi_t(\mathbf{y}_t), \quad (10)$$

where $\delta\Phi_t(\mathbf{y}_t) = \Phi(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{z}}) - \Phi(\mathbf{x}_t, \mathbf{y}_{t,i}, \hat{\mathbf{z}'})$, $\hat{\mathbf{z}} = \operatorname{argmax}_{\mathbf{z}} f(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}; \mathbf{w}_t)$, $\hat{\mathbf{z}'} = \operatorname{argmax}_{\mathbf{z}'} f(\mathbf{x}_t, \mathbf{y}_{t,i}, \mathbf{z}'; \mathbf{w}_t)$ and indicator function $\mathbb{1}(a) = 1$ when the statement a is true, 0 otherwise. We then update $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \nabla_t$ with step size $\eta_t = 1/(\lambda t)$, it can be written as:

$$\mathbf{w}_{t+1} \leftarrow (1 - \eta_t \lambda)\mathbf{w}_t + \frac{\eta_t}{N} \sum_{i=1}^{N} \mathbb{1}\big[\max_{\mathbf{z}'} f(\mathbf{x}_t, \mathbf{y}_{t,i}, \mathbf{z}'; \mathbf{w}_t) - \max_{\mathbf{z}} f(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}; \mathbf{w}_t) + \Delta(\mathbf{y}_t, \mathbf{y}_{t,i}) > 0\big] \delta\Phi_t(\mathbf{y}_t). \quad (11)$$

2.3. Inference

Given the parameters $\mathbf{w}_t$ and an image $\mathbf{x}_t$, the inference is to find the offset of bounding box of the target object with the best score,

$$(\mathbf{y}_t^*, \mathbf{z}_t^*) = \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}, \mathbf{z} \in \mathcal{Z}} f(\mathbf{x}_t, \mathbf{y}, \mathbf{z}; \mathbf{w}_t) \quad (12)$$

$$= \operatorname*{argmax}_{\mathbf{y} \in \mathcal{Y}} \Big[\sum_{j=1}^{M} \operatorname*{argmax}_{\mathbf{z}^j \in \mathcal{Z}^j} \Big(\langle \mathbf{u}_t^j, \phi_1(\mathbf{x}_t, \mathbf{z}^j) \rangle + \langle \mathbf{v}_t^j, \phi_3(\mathbf{y}, \mathbf{z}^j) \rangle \Big) + \langle \mathbf{v}_t^0, \phi_2(\mathbf{x}_t, \mathbf{y}) \rangle \Big]. \quad (13)$$

We restrict all possible offset $\mathbf{y}$ within a radius s, *i.e.* $\mathcal{Y} = \{(\Delta c_t, \Delta r_t, \Delta w_t, \Delta h_t) | (\Delta c_t)^2 + (\Delta r_t)^2 < s^2\}$. Similarly all possible $\mathbf{z}^j \in \mathcal{Z}^j$ is restricted within a ball of radius s^j. Since the goal is to track the object, $\mathbf{z}_t^*$ does not need to be stored. We keep and visualise $\mathbf{z}_t^*$ to show the part boxes are indeed meaningful.

2.4. Features

We use Haar-like features [24] to represent the appearances of the bounding box of object and object parts. Each feature consists of 2 scales, each scale has 6 types rectangles on a 4×4 grid. We use integral image proposed in [24] to compute the feature efficiently. The integral image requires computing the sum of rectangular region, the tracking performance will be reduced when the shape of the target cannot be well fitted by rectangle. However, our part-based appearance model can alleviate this problem by decomposing the whole rectangular region into small parts.

Algorithm 1: Two-stage Training

Input: λ, the number of part M, number of training samples N, B_t, $\mathbf{b}_t^1, \cdots, \mathbf{b}_t^M$, $\mathbf{w}_t$, $\mathbf{x}_t$
Output: $\mathbf{w}_{t+1} = [\mathbf{u}_{t+1}, \mathbf{v}_{t+1}]$

Stage 1: To get $\mathbf{u}_{t+1} = [\mathbf{u}_{t+1}^1, \cdots, \mathbf{u}_{t+1}^M]$
for $j = 1, 2, \cdots, M$ **do**
  Generate a set of offset $\{\mathbf{z}_{t,k}^j\}_{k=1}^N$, where all offset $\mathbf{z}$ are restricted within a radius s; then we have training set $S_t^j = \{(\mathbf{x}_t, \mathbf{z}_{t,k}^j)\}_{k=1}^N$;
  Crop out the corresponding image regions of $\{\mathbf{b}_t^j + \mathbf{z}_{t,k}^j\}_{k=1}^N$ in frame $\mathbf{x}_t$;
  Extract feature $\{\phi_1(\mathbf{x}_t, \mathbf{z}_{t,k}^j)\}_{k=1}^N$;
  Set $(S_t^j)^+ = \{(\mathbf{x}_t, \mathbf{z}_{t,k}^j) \in S_t^j : \Delta(\mathbf{z}_t^j, \mathbf{z}_{t,k}^j) + \langle \mathbf{u}_t^j, \phi_1(\mathbf{x}_t, \mathbf{z}_{t,k}^j) \rangle - \langle \mathbf{u}_t^j, \phi_1(\mathbf{x}_t, \mathbf{z}_t^j) \rangle > 0\}$;
  Set $\eta_t = \frac{1}{\lambda t}$;
  Update $\mathbf{u}_{t+1}^j \leftarrow (1 - \eta_t)\lambda \mathbf{u}_t^j + \frac{\eta_t}{|S_t^j|} \sum_{(\mathbf{x}_t, \mathbf{z}_{t,k}^j) \in (S_t^j)^+} (\phi_1(\mathbf{x}_t, \mathbf{z}_t^j) - \phi_1(\mathbf{x}_t, \mathbf{z}_{t,k}^j))$.
end
Stage 2: To get $\mathbf{v}_{t+1} = [\mathbf{v}_{t+1}^0, \mathbf{v}_{t+1}^1, \cdots, \mathbf{v}_{t+1}^M]$
Generate a set of offset $\{\mathbf{y}_{t,i}\}_{i=1}^N$ just as $\{\mathbf{z}_{t,k}^j\}_{k=1}^N$, $j \in \{1, \cdots, M\}$;
Crop out the corresponding image regions of $\{B_t + \mathbf{y}_{t,i}\}_{i=1}^N$ in frame $\mathbf{x}_t$;
Extract feature $\{\phi_2(\mathbf{x}_t, \mathbf{y}_{t,i})\}_{i=1}^N$ and $\{\phi_3(\mathbf{y}_{t,i}, \mathbf{z}_{t,k}^j) : i = 1, \cdots, N, k = 1, \cdots, N, j = 1, \cdots, M\}$;
Update the weight vector $\mathbf{v}_{t+1}$ using Eq. (15).

The Haar-like feature of object and object parts are concatenated as a vector, *i.e.* $\phi_1(\mathbf{x}_t, \mathbf{z}^j)$ and $\phi_2(\mathbf{x}_t, \mathbf{y})$. The spatial relationship between bounding box and part box is defined as $\phi_3(\mathbf{y}, \mathbf{z}^j) = (\mathbf{a}, \mathbf{a}^2) \in \mathbb{R}^4$ where

$$\mathbf{a} = (B_{t-1} + \mathbf{y})[1:2] - (b_{t-1} + \mathbf{z}^j)[1:2] \in \mathbb{R}^2.$$

Here $[1:2]$ means taking the first two dimensions of the vector. $\langle \mathbf{v}^j, \phi_3(\mathbf{y}, \mathbf{z}^j) \rangle$ measures the compatibility between the bounding box and the j-th part box. For example, the part boxes far away from the bounding box will be discouraged for an appropriate $\mathbf{v}^j$.

3. Two-stage Training

The latent pegasos proposed in Section 2.2 has two problems: (1) the label cost Δ in (8) does not take into account the part boxes. This can potentially cause part boxes are predicted badly (though there is no ground truth of $\mathbf{z}$, one can still see if the predicted $\mathbf{z}$ makes sense); (2) incorporating part boxes increases the dimensionality of the parameter significantly, hence may cause over-fitting during

Algorithm 2: Part-based Tracking

Input: t-th frame $\mathbf{x}_t$, $\mathbf{w}_t$, B_{t-1}, $\mathbf{b}_{t-1}^1, \cdots, \mathbf{b}_{t-1}^M$, number of training samples for N, number of tracking samples K

Output: $\mathbf{w}_{t+1}$, B_t, $\mathbf{b}_t^1, \cdots, \mathbf{b}_t^M$

1 Sample a set of candidate offset $\{\mathbf{y}_{t,i}\}_{i=1}^K \in \mathcal{Y}$ uniformly that are within search radius s;

2 Crop out the corresponding image region of $\{B_{t-1} + \mathbf{y}_{t,i})\}_{i=1}^K$ in frame $\mathbf{x}_t$, and extract features;

3 Estimate $\mathbf{y}_t^*$ and $\mathbf{z}_t^*$ via Eq. (12) with letting $\mathcal{Z}^j = \mathcal{Y}$;

4 Update $B_t = B_{t-1} + \mathbf{y}_t^*$, and $\mathbf{b}_t^j = \mathbf{b}_{t-1}^j + (\mathbf{z}^j)^*, j = 1, \cdots, M$;

5 $\mathbf{w}_{t+1} \leftarrow$ two-stage training as Algorithm 1.

training. If the model is trained well, the problem (1) may not be so severe, because a well learnt $\mathbf{w}$ should ensure $\max_{\mathbf{z}} f(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}; \mathbf{w}) \geq \max_{\mathbf{z}'} f(\mathbf{x}_t, \mathbf{y}, \mathbf{z}'; \mathbf{w}) + \Delta(\mathbf{y}_t, \mathbf{y})$. However, because of problem (2), the model is often not well learnt. For example in Fig. 3 (Left), a good $\mathbf{y}$ with red highlighted frame boundary does not receive the highest score in latent pegasos, and the predicted $\mathbf{y}$ via the parameter learnt by latent pegasos is obviously not a good one. To tackle above problems, we propose a two-stage training method below.

At every frame, say the t-th frame, we learn the weight vector $\mathbf{w_{t+1}} = [\mathbf{u_{t+1}}, \mathbf{v_{t+1}}]$ in two stages.

In the first stage, given t-th frame $\mathbf{x}_t$ and predicted $\mathbf{z}_t^j$, we sample offsets of part box $\{\mathbf{z}_{t,k}^j \neq \mathbf{z}_t^j\}_{k=1}^N$. We would like to update $\mathbf{u}_{t+1}^j, j = 1, \cdots, M$ via,

$$\mathbf{u}_{t+1}^j = \underset{\mathbf{u}^j}{\operatorname{argmin}} \frac{\lambda}{2}||\mathbf{u}^j||^2 + \frac{1}{N}\sum_{k=1}^{N}\Big[\Delta(\mathbf{z}_t^j, \mathbf{z}_{t,k}^j) + \langle \mathbf{u}^j, \phi_1(\mathbf{x}_t, \mathbf{z}_{t,k}^j)\rangle - \langle \mathbf{u}^j, \phi_1(\mathbf{x}_t, \mathbf{z}_t^j)\rangle\Big]_+. \quad (14)$$

The $\Delta(\mathbf{z}_t^j, \mathbf{z}_{t,k}^j)$ is still the VOC label cost defined in Eq. (9). Learning $\mathbf{u}_{t+1}^j$ is effectively tracking part boxes, thus the learnt $\mathbf{u}_{t+1}^j$ gives pretty good estimate of where the parts should be. Estimating $\mathbf{u}_{t+1}^j$ via sub-gradient method [21] can be found in the stage 1 of Algorithm 1.

In the second stage, we update $\mathbf{v}_{t+1}$ via

$$\mathbf{v}_{t+1} \leftarrow (1 - \eta_t\lambda)\mathbf{v}_t + \frac{\eta_t}{N}\sum_{i=1}^{N}\mathbb{1}\big[\max_{\mathbf{z}'} f(\mathbf{x}_t, \mathbf{y}_{t,i}, \mathbf{z}'; \mathbf{v}_t) - \max_{\mathbf{z}} f(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}; \mathbf{v}_t) + \Delta(\mathbf{y}_t, \mathbf{y}_{t,i}) > 0\big]\delta\Psi_t(\mathbf{y}_t), \quad (15)$$

where $\delta\Psi_t(\mathbf{y}_t) = \Psi(\mathbf{x}_t, \mathbf{y}_t, \hat{\mathbf{z}}) - \Psi(\mathbf{x}_t, \mathbf{y}_{t,i}, \hat{\mathbf{z}'})$, and $\Psi(\mathbf{x}_t, \mathbf{y}, \mathbf{z})$ defines as:

$$\Psi(\mathbf{x}_t, \mathbf{y}, \mathbf{z}) = \big[\phi_2(\mathbf{x}_t, \mathbf{y}), \phi_3(\mathbf{y}, \mathbf{z}^1), \cdots, \phi_3(\mathbf{y}, \mathbf{z}^M)\big]. \quad (16)$$

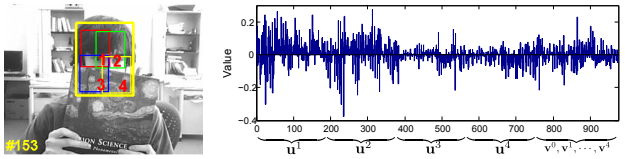

Figure 2: Left: The image and tracking results. The yellow rectangle shows the bounding box of object, and four small rectangles show the bounding box of parts. **Right:** The plot of the values of weight vector $\mathbf{w}$ corresponding to the left image. The values of vector $\mathbf{u}^3$ and $\mathbf{u}^4$ is less than the values of $\mathbf{u}^1$ and $\mathbf{u}^2$ due to the occlusion of the 3rd and 4th part.

Algorithm 1 describes the whole procedure of two-stage training. The stochastic sub-gradient method [21] is guaranteed to converge to the optimal SVM solution. After using the above two-stage training, we have updated the weight vector $\mathbf{w}_{t+1}$ at time step t. The parameter $\mathbf{w}_{t+1}$ can be used to predict the best offset $\mathbf{y}_{t+1}^*$ relative to the bounding box B_{t+1} for tracking task, and used as the starting point in next iteration. Fig. 2 shows the illustration of the values of weight vector $\mathbf{w}$ at frame 153 of sequence *faceocc2* while a partial occlusion appears. The tracker adapts to deal with partial occlusion quickly: $\mathbf{u}^1$ and $\mathbf{u}^2$ (not occluded) adapt to have higher weights than $\mathbf{u}^3$ and $\mathbf{u}^4$ (occluded), thus the tracking result will rely more on the not occluded parts. Investigating the trends of weights will also help to understand which feature contributes more in various scenarios.

So far, we have introduced the part-based appearance model and online updated discriminatively structured output classifier to identify the target object from candidates. The overall procedure of the proposed tracking algorithm we implemented in this work is shown in Algorithm 2.

4. Experiments

We evaluated our tracking system on thirteen challenging sequences, which include eight tracking by detection benchmark sequences (*coke, david, faceocc1, faceocc2, girl, sylvester, tiger1, tiger2*) and additional five sequences (*board, fskater, threemen, trellis, dollar*). These sequences contain varied tracking targets (e.g., human body, car, face, *et al.*) and different challenging situations in object tracking, as shown in Table 2.

The proposed algorithm is implemented in C++ on a workstation with an Intel Core 2 Duo 2.66GHz processor and 4.0G RAM. The average running time of our tracker is 2-3 frames per second, which is similar to Struck under the same experimental environment. To make fair comparisons, we use the same image feature as Struck [10] and MIL [4]. The parameters are presented as follows. λ in Eq. (8) and Eq. (14) are all set to 0.1. As mentioned in Sec. 2.1, we simply consider the offset in 2D translation. we sample offsets $\mathbf{y}$ and $\mathbf{z}$ within a search radius $s = 30$ exhaustively in tracking stage, while on a polar grid within a search radius $s = 60$ in training stage. The search radius of object tracker and part trackers are all the same for all sequences. We

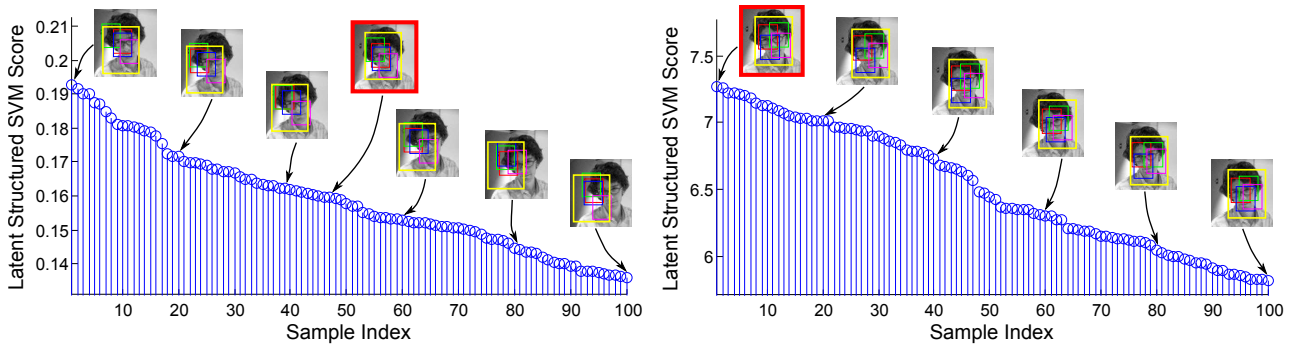

Figure 3: Latent Pegasos v.s. two-stage training in the *david* sequence. **Left**: Latent Pegasos; **Right**: two-stage training. The score $\max_{\mathbf{z}} f(\mathbf{x}, \mathbf{y}, \mathbf{z}; \mathbf{w})$ for 100 different $\mathbf{y}$s are sorted (several representative images are shown). The big yellow rectangle indicates the bounding box of the object. Clearly the highlighted red frame obtains the best $\mathbf{y}$ among all shown results. However, it does not receive highest score in Latent Pegasos. The $\mathbf{y}'$ with the highest score in Latent Pegasos has much worse tracking result than the red highlighted one. In two-stage training, highlighted red frame does receive the highest score as expected.

Sequence	Ours	Struck [10]	StruckL [10]	CT [28]	MIL [4]	OAB [8]	BHMC [12]	Frag [1]	ℓ1T [16]	VTD [13]	IVT [18]
faceocc1	**0.87±0.05**	0.83±0.06	0.84±0.08	0.63±0.16	0.61±0.20	0.50±0.21	0.28±0.06	0.86±0.09	0.78±0.28	0.82±0.20	0.63±0.14
faceocc2	**0.83±0.06**	0.80±0.09	0.77±0.09	0.82±0.12	0.80±0.09	0.67±0.15	0.23±0.06	0.70±0.13	0.19±0.28	0.64±0.21	0.20±0.27
threemen	**0.76±0.12**	0.19±0.33	0.21±0.32	0.19±0.33	0.21±0.35	0.21±0.34	0.11±0.16	0.76±0.10	0.41±0.31	0.18±0.31	0.21±0.33
fskater	**0.81±0.11**	0.74±0.14	0.73±0.13	0.60±0.12	0.76±0.12	0.71±0.11	0.52±0.11	0.51±0.24	0.62±0.12	0.65±0.11	0.19±0.26
dollar	**0.81±0.09**	0.70±0.11	0.69±0.11	0.66±0.14	0.58±0.16	0.55±0.18	0.38±0.22	0.31±0.38	0.35±0.44	0.71±0.29	0.26±0.36
david	**0.81±0.10**	0.79±0.12	0.36±0.36	0.67±0.13	0.62±0.10	0.38±0.23	0.52±0.11	0.18±0.24	0.22±0.33	0.29±0.27	0.32±0.27
trellis	**0.63±0.14**	0.44±0.35	0.09±0.24	0.11±0.17	0.20±0.31	0.17±0.28	0.23±0.25	0.40±0.36	0.38±0.39	0.33±0.37	0.15±0.23
board	**0.84±0.12**	0.40±0.29	0.59±0.35	0.55±0.17	0.16±0.27	0.12±0.25	0.01±0.02	0.48±0.26	0.11±0.27	0.31±0.27	0.10±0.21
sylvester	**0.76±0.14**	0.69±0.29	0.66±0.25	0.52±0.20	0.67±0.18	0.47±0.38	0.54±0.14	0.60±0.23	0.45±0.36	0.58±0.31	0.06±0.16
girl	0.72±0.12	**0.81±0.10**	0.78±0.11	0.52±0.16	0.62±0.12	0.45±0.23	0.33±0.20	0.65±0.20	0.71±0.09	0.64±0.12	0.38±0.31
coke*	**0.62±0.21**	0.55±0.18	0.46±0.23	0.36±0.22	0.35±0.23	0.09±0.19	0.01±0.01	0.07±0.21	0.05±0.20	0.09±0.22	0.08±0.22
tiger1*	0.61±0.19	**0.66±0.23**	0.59±0.21	0.37±0.29	0.64±0.19	0.44±0.23	0.27±0.19	0.20±0.30	0.40±0.36	0.10±0.24	0.02±0.11
tiger2*	0.51±0.23	0.57±0.20	0.45±0.24	0.51±0.18	**0.63±0.14**	0.35±0.24	0.15±0.23	0.15±0.24	0.27±0.32	0.19±0.23	0.02±0.12
Average	**0.73**	0.62	0.55	0.50	0.52	0.39	0.28	0.45	0.38	0.42	0.20

Table 1: Compared average VOC overlap ratio on thirteen sequences. The target object of three sequences marked with star (∗) wasn't decomposed into parts in experiment since the object is too small. Our algorithm without parts is different from Struck with linear kernel, we solve the structured SVM based on primal form.

Main Challenges	Sequences
Partial or full occlusion	*faceocc1, faceocc2, threemen*
Shape deformation	*fskater, dollar*
Illumination	*david, trellis, coke, tiger1, tiger2*
Pose and scale variation	*board, sylvester, girl, fskater, david, coke, tiger1, tiger2*

Table 2: The main challenges showed in thirteen experimental sequences

employ a simple heuristic to determine the number of part within the tracking object. If the ratio of width and height of object is larger than 0.5, we divide the object into four parts (two rows and two columns). Otherwise, the tracking object is divided into three parts (in vertical direction if width is larger than height, or in horizontal direction).

To examine the performance of the proposed tracking algorithm, we run nine state-of-the-art algorithms with the same initial position of the tracking object. These algorithms are Struck (structured output tracker [10]), CT (compressive tracking [28]), MIL (multiple instance boosting-based tracker [4]), OAB (online AdaBoost [8], $r = 1$), BHMC (Basin Hopping Monte Carlo tracker [12]), Frag (Fragment-based tracker [1]), IVT (incremental subspace visual tracker [18]), VTD (visual tracking decomposition [13], and ℓ1T ($\ell 1$ minimisation tracker [16]). We use the publicly available source code or binary code of those trackers in the experiments. For BHMC, only partial frames can be processed by using the binary code released by authors, we report the tracking results of those frames. Because our tracker use latent structured SVM with linear kernel, we also compared Struck with linear kernel (named as Struck Linear or StruckL). For quantitative performance comparison, three popular evaluation criteria are introduced. Firstly, the mean center position error (CLE) per frame is calculated for each tracker. Secondly, we report the Pascal VOC overlap ratio, which is defined as $R_{overlap} = Area(B_T \cap B_{GT})/Area(B_T \cup B_{GT})$, 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 tracking for each frame.

Comparison of different latent structured learning schema In order to evaluate the performance of different latent learning schema, we conduct an experiment on *david* sequence using different form of latent structural SVM. We sort the score of candidate offsets $\mathbf{y}$s at frame 112 with different latent learning schema, and report the score of maximum 100 samples in Fig. 3. From the experimental results, we can see that the part boxes and object boxes with la-

Sequence	Ours	Struck [10]	StruckL [10]	CT [28]	MIL [4]	OAB [8]	BHMC [12]	Frag [1]	ℓ1T [16]	VTD [13]	IVT [18]
faceocc1	**7.4±2.8**	10.1±3.8	9.0±4.6	28.6±17.7	29.2±18.6	42.2±23.1	47.9±8.7	8.6±6.6	20.7±41.6	10.3±16.9	24.8±11.6
faceocc2	**7.7±2.9**	9.2±4.5	11.0±4.8	8.9±7.2	9.4±4.4	18.4±9.7	41.3±5.4	15.3±7.7	78.5±38.1	14.4±17.9	117±84.5
threemen	**9.9±6.3**	61.5±34.7	51.0±29.5	68.4±35.7	69.8±41.4	68.8±39.9	63.2±38.6	10.3±3.4	16.8±16.6	61.8±35.5	44.5±25.7
fskater	**7.9±4.3**	10.9±5.8	11.7±6.1	23.0±5.4	14.2±8.0	14.6±5.6	17.9±7.4	28.8±18.3	24.4±10.1	18.8±10.3	86.7±49.1
dollar	**7.0±3.4**	14.2±5.4	14.7±5.6	15.6±6.4	23.1±9.1	23.4±11.7	36.4±23.9	70.2±59.2	66.4±58.9	21.3±41.2	79.6±61.6
david	**7.6±4.1**	8.9±5.7	60.1±51.6	14.7±7.0	18.9±5.9	38.8±27.6	20.7±11.1	73.6±36.8	81.2±57.7	65.7±56.1	11.9±6.4
trellis	**10.6±5.3**	41.0±49.4	107±54.6	88.4±57.0	81.3±61.4	100±72.1	58.6±48.3	34.4±31.9	52.6±54.4	54.7±51.9	135±95.1
board	**14.1±13.3**	85.4±56.5	72.1±98.9	52.9±27.1	211±124	216±111	307±95	65.9±48.6	241±110	112±67.0	223±97.5
sylvester	**5.7±5.2**	14.4±26.1	10.7±10.9	15.3±10.3	9.1±6.4	33.1±36.7	12.4±9.5	12.4±11.5	29.6±32.6	21.6±36.0	74.0±37.6
girl	15.2±7.5	**10.0±5.4**	11.6±6.9	32.1±14.4	22.8±8.7	42.8±26.3	47.4±26.6	23.0±22.7	11.0±5.6	18.0±11.4	43.1±52.6
coke*	**7.1±6.5**	8.1±5.6	11.9±8.4	17.5±9.8	17.5±9.6	34.8±15.6	55.9±15.6	70.0±32.3	55.3±22.3	47.4±21.9	39.5±22.1
tiger1*	8.7±5.7	**8.2±12.2**	9.6±6.7	28.3±29.9	8.5±6.6	18.0±16.8	21.2±13.9	39.6±32.5	29.1±30.2	68.7±36.2	56.7±20.5
tiger2*	11.8±10.4	8.3±6.1	14.9±13.4	10.6±5.3	**7.2±3.7**	20.1±15.1	51.2±37.3	38.5±25.1	33.9±28.1	37.5±29.7	93.3±34.7
Average	**9.2**	22.3	30.4	31.1	40.1	51.6	82.2	60.1	56.9	42.4	79.1

Table 3: Compared average center location error (pixels) on thirteen sequences. The explanation of star ($*$) marked on the last three sequences is the same as Table 1.

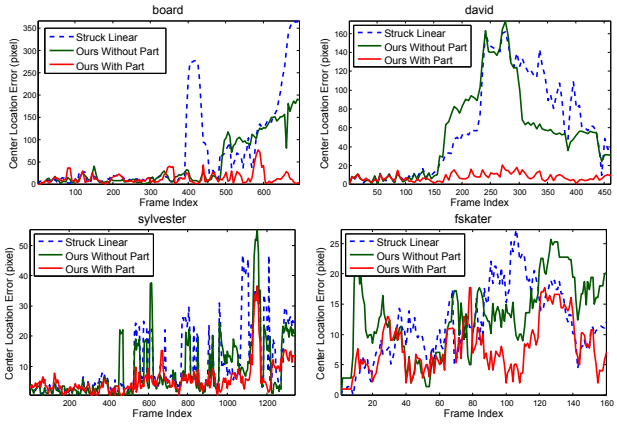

Figure 4: The centre location error plots of our tracker with/without parts and struck with linear kernel on four sequences.

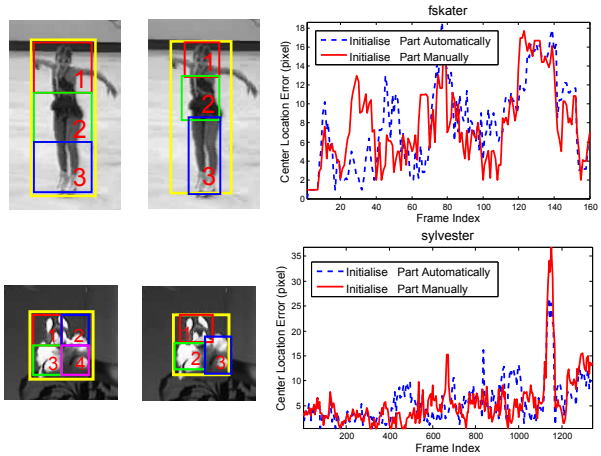

Figure 5: Quantitative evaluation of our tracker with different part initialisation on 2 sequences. **Left:** two kinds of initialisation (automatically and manually). **Right:** the center location error of different part initialisations.

tent pegasos shown in Fig. 3 (Left) are drifting heavily. On the contrary, the tracker with two-stage training maintains tracking the object and the parts robustly shown in Fig. 3 (Right). Note that the output score of two-stage training is higher than latent pegasos due to different parameter scales. What really matters is the ranking of the scores for each tracker. A good tracker should rank a good result higher. For example, the best candidate sample (the sample with red rectangle in Fig. 3) doesn't not get the highest score using the latent pegasos. The success rate is 0.57 and 1.00 for using latent pegasos and two-stage training, respectively.

Evaluation of our tracking algorithm with and without part To justify the effect of object parts, we run Struck with linear kernel, ours tracking algorithm with and without part. Note that our algorithm without part is different with Struck with linear kernel, the structured SVM of Struck is solved in dual, but for us, it's optimised based on primal form. For generality, we choose four video sequences with different kind of objects (human body, face, rigid object and toy). Fig. 4 shows the performance of these algorithms in centre location error. Note that the quantitative VOR and CLE results, and VOR curve for these four sequences can be found in the supplementary material. We can see that ours without part gets the similar performance to Struck with linear kernel, and ours with part achieves higher success rate and lower CLE than another two algorithms.

Evaluation of different part initialisations In [29], the authors point out that the traditional latent SVM using a gradient decent algorithm requires careful part initialisation. As mentioned above, in this paper, we use a simple heuristic method to initialise part number and position. To investigate the effect of different part initialisations, we conduct two experiments with different initialised method. In the first experiment as shown in the upper row of Fig. 5, we initialise part with the same number in sequence *fskater* by using our automatic heuristic method and dividing meaningful object parts manually, respectively. The lower row of Fig. 5 shows the result of the second experiment on sequence *sylvester* with different number of parts. From Fig. 5, we found that two kinds of part initialisation achieve the similar CLE performance in most frames. The results indicate that our tracking algorithm is robust to part initialisation. Note that the quantitative VOR, CLE, and success rate results, and VOR curve for these four sequences can be found in the supplementary material.

Comparison of competing tracking algorithms We compare the proposed tracking algorithm with nine state-of-

Sequence	Ours	Struck	StruckL	CT	MIL	OAB	BHMC	Frag	ℓ1T	VTD	IVT
faceocc1	**1.00**	**1.00**	**1.00**	0.73	0.66	0.48	0.01	**1.00**	0.88	0.91	0.84
faceocc2	**1.00**	**1.00**	**1.00**	0.98	**1.00**	0.82	0.01	0.94	0.17	0.78	0.20
threemen	**1.00**	0.24	0.24	0.22	0.25	0.26	0.03	**1.00**	0.39	0.23	0.24
fskater	**1.00**	**1.00**	0.99	0.76	**1.00**	0.99	0.61	0.61	0.93	0.97	0.11
dollar	**1.00**	**1.00**	**1.00**	0.86	0.64	0.52	0.20	0.39	0.39	0.85	0.18
david	**1.00**	0.97	0.38	0.88	0.90	0.31	0.65	0.08	0.24	0.26	0.24
trellis	**0.84**	0.46	0.10	0.20	0.24	0.14	0.23	0.38	0.44	0.37	0.19
board	**0.98**	0.29	0.65	0.59	0.13	0.08	0.01	0.46	0.12	0.23	0.09
sylvester	**0.95**	0.83	0.72	0.59	0.77	0.52	0.62	0.70	0.54	0.67	0.04
girl	0.98	**1.00**	0.99	0.47	0.85	0.38	0.25	0.79	0.99	0.86	0.45
coke*	**0.78**	0.71	0.47	0.29	0.27	0.05	0.03	0.07	0.05	0.10	0.08
tiger1*	0.74	**0.83**	0.65	0.36	0.81	0.40	0.13	0.19	0.49	0.14	0.01
tiger2*	0.58	0.68	0.41	0.49	**0.86**	0.32	0.12	0.11	0.29	0.15	0.01
Average	**0.91**	0.77	0.66	0.57	0.64	0.41	0.22	0.52	0.46	0.50	0.21

Table 4: Compared success rates on thirteen sequences. The explanation of star ($*$) marked on the last three sequences is the same as Table 1.

the-art tracking algorithms. Fig. 1 shows some qualitative tracking results of our tracker and another several competing trackers on three sequences (for clarity, we just show the results of five trackers with better performance.) More tracking results can be found in the supplemental material. Table 1 reports the average VOC overlap ratio of ten trackers over all sequences. Table 3 summarises the average centre location error performance of the compared tracking algorithms over the thirteen sequences. Table 4 shows the success rate. From the experimental results, we can see that our tracking algorithm obtains the best result on ten sequences. As for sequence *girl*, the tracking performance of our tracker is slightly lower than Stuck, it's partly due to the part will not help when each part appears similarly (*e.g*. the whole object region is filled with hair when the girl swivelled around). As mentioned in Table 1, there are not many practical implications to divide the target object of sequence *coke*, *tiger1* and *tiger2* into parts due to their small object region. The results are not as well as the competing trackers. But the results of our tracker on the above mentioned four sequences are better than Struck with linear kernel. In conclusion, the proposed tracker performs well on rigid objects such as board and faces as well as deformable objects such as persons.

Conclusion We have introduced a part-based tracking algorithm with online latent structured learning. We use a global object box and a small number of part boxes to approximate the irregular object, and the model can be initialised easily. The proposed appearance model reduced the amount of visual drift. We developed an online two-stage training mechanism to learn the parameter of the part-based model. The new online learning schema overcame the over-fitting caused by the complexity of part model, thus improved the tracking accuracy of object parts. We believe that the proposed part-based tracker provides powerful framework for visual tracking. We performed object tracking using the proposed algorithm on thirteen challenging sequences comparable with a few of state-of-the-art methods, the results demonstrated the effectiveness and robustness of the proposed tracker.

Acknowledgement This work was in part supported by National Nature Science Foundation of China (61231016, 61272288); and Australian Research Council grants DP1094764, DE120101161, FT120100969, and DP11010352.

R. Yao's contribution was made when he was a visiting student at the University of Adelaide.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, volume 1, pages 798 – 805, 2006.

[2] Y. Amit and A. Trouvé. POP: Patchwork of Parts Models for Object Recognition. *Int. J. Comp. Vis.*, 75(2):267–282, 2007.

[3] S. Avidan. Support vector tracking. *IEEE Trans. Pattern Anal. & Mach. Intelli.*, 26(8):1064 –1072, 2004.

[4] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *IEEE Trans. Pattern Anal. & Mach. Intelli.*, 33(8):1619 –1632, 2011.

[5] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *Proc. Eur. Conf. Comp. Vis.*, volume 5302, pages 2–15, 2008.

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comp. Vis.*, 88(2):303–338, June 2010.

[7] P. F. Felzenszwalb, R. B. Girshick, D. A. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, 2010.

[8] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *Proc. British Mach. Vis. Conf.*, volume 1, pages 47–56, 2006.

[9] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *Proc. Eur. Conf. Comp. Vis.*, pages 234–247, 2008.

[10] S. Hare, A. Saffari, and P. Torr. Struck: Structured output tracking with kernels. In *Proc. Int. Conf. Comp. Vis.*, pages 263–270, 2011.

[11] G. Hua and Y. Wu. Measurement integration under inconsistency for robust tracking. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, 2006.

[12] J. Kwon and K. M. Lee. Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, 2009.

[13] J. Kwon and K. M. Lee. Visual tracking decomposition. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, pages 1269–1276, 2010.

[14] H. Li, C. Shen, and Q. Shi. Real-time visual tracking using compressive sensing. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, pages 1305 –1312, 2011.

[15] Z. Lin, G. Hua, and L. S. Davis. Multiple instance feature for robust part-based object detection. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, 2009.

[16] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE Trans. Pattern Anal. & Mach. Intelli.*, 33:2259 –2272, 2011.

[17] M. Pedersoli, A. Vedaldi, and J. Gonzalez. A coarse-to-fine approach for fast deformable object detection. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, 2011.

[18] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *Int. J. Comp. Vis.*, 77:125–141, 2008.

[19] P. Schnitzspan, S. Roth, and B. Schiele. Automatic discovery of meaningful object parts with latent crfs. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, pages 121–128, 2010.

[20] S. M. N. Shahed, J. Ho, and M.-H. Yang. Visual tracking with histograms and articulating blocks. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, 2008.

[21] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated subgradient solver for SVM. In *Proc. Int. Conf. Mach. Learn.*, 2007.

[22] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res.*, 6:1453–1484, 2005.

[23] A. Vedaldi and A. Zisserman. Structured output regression for detection with partial truncation. In *Proc. Adv. Neural Info. Process. Syst.*, 2009.

[24] P. A. Viola and M. J. Jones. Rapid object detection using a boosted cascade of simple features. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, pages 511–518, 2001.

[25] X. Wang, G. Hua, and T. X. Han. Discriminative tracking by metric learning. In *Proc. Eur. Conf. Comp. Vis.*, pages 200–214, 2010.

[26] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel. Robust tracking with weighted online structured learning. In *Proc. Eur. Conf. Comp. Vis.*, 2012.

[27] C. N. J. Yu and T. Joachims. Learning structural svms with latent variables. In *Proc. Int. Conf. Mach. Learn.*, pages 1169–1176, 2009.

[28] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In *Proc. Eur. Conf. Comp. Vis.*, 2012.

[29] L. Zhu, Y. Chen, A. L. Yuille, and W. T. Freeman. Latent hierarchical structural learning for object detection. In *Proc. IEEE Conf. Comp. Vis. & Pattern Recogn.*, pages 1062–1069, 2010.