Online Deformable Object Tracking Based on Structure-Aware Hyper-Graph

Dawei Du, Honggang Qi, Wenbo Li, Longyin Wen, Qingming Huang, and Siwei Lyu

***Abstract*— Recent advances in online visual tracking focus on designing part-based model to handle the deformation and occlusion challenges. However, previous methods usually consider only the pairwise structural dependences of target parts in two consecutive frames rather than the higher order constraints in multiple frames, making them less effective in handling large deformation and occlusion challenges. This paper describes a new and efficient method for online deformable object tracking. Different from most existing methods, this paper exploits higher order structural dependences of different parts of the tracking target in multiple consecutive frames. We construct a structure-aware hyper-graph to capture such higher order dependences, and solve the tracking problem by searching dense subgraphs on it. Furthermore, we also describe a new evaluating data set for online deformable object tracking (the Deform-SOT data set), which includes 50 challenging sequences with full annotations that represent realistic tracking challenges, such as large deformations and severe occlusions. The experimental result of the proposed method shows considerable improvement in performance over the state-of-the-art tracking methods.**

***Index Terms*— Online tracking, deformable object tracking, part-based model, structure-aware hyper-graph, dense subgraph searching.**

I. Introduction

ONLINE visual tracking is an important step toward fully automatic understanding of videos, which finds wide applications in video surveillance, behavior analysis, human computer interaction, to name a few. In spite of significant progress in the recent years, online tracking a deformable object accurately remains a difficult problem. Challenges in online visual tracking originate from large variations of the tracking target in appearance, shape, and motion, as well as the occlusions caused by other objects or background.

Currently, the predominant approaches in online visual tracking aim to obtain a bounding box of the tracking target. Many methods are built on models which focus on capturing appearance variation of the tracking target in the bounding box, *e.g.*, correlation filters [11], [32], subspace learning [26], [42], online boosting [4], [22] and sparse representation [29], [50]. Furthermore, the contextual information has been considered in detection and tracking tasks [3], [6], [28]. Recent methods [16], [41], [46] of online visual tracking also focus on exploiting contextual information beyond individual target appearance to improve tracking performance. However, due to the use of bounding boxes that encompass pixels of both the target and the background, it is difficult to construct accurate target appearance model. As such, these methods may be sensitive to background noise, and are easily "drift" away from the tracking target when large deformation occurs.

In addition to tracking methods based on bounding boxes, there have been some developments in part-based trackers [2], [8], [23], [39], [47], which incorporate structural relations among target parts to improve the robustness in handling object deformation and occlusion. Some recent methods take one step further to apply foreground segmentation to delineate the boundary of the target [8], [15], a problem also known as video object segmentation methods [10], [43]. To date, existing part-based tracking methods only consider pairwise correlations in appearance from consecutive frames. However, to effectively handle occlusion for tracking, one needs to consider higher-order dependencies in object motion, *i.e.*, modeling the relations among multiple parts of the target in more than two frames.

In this paper, we propose a new online deformable object tracking method based on a *structure-aware hyper-graph*, which, as illustrated in Fig. 1(b), can effectively incorporate higher-order dependencies among more than two consecutive frames. As such, we refer to our method as *structure aware tracker* (SAT) subsequently. In our method, the tracking target is represented by multiple parts, which are ensembles of super-pixels that are similar in appearance and motion. To find such parts, we first apply the SLIC over-segmentation algorithm [1] to generate super-pixels in each video frame, and then apply the graph cut algorithm [5] to optimize an energy

Manuscript received August 9, 2015; revised January 15, 2016 and May 3, 2016; accepted May 8, 2016. Date of publication May 18, 2016; date of current version June 14, 2016. This work was supported in part by the National Basic Research Program of China (973 Program) under Grant 2012CB316400 and Grant 2015CB351802, in part by the National Natural Science Foundation of China under Grant 61332016, Grant 61472388 and Grant 61025011, in part by U.S. NSF through the CAREER Award IIS under Grant 0953373, in part by the U.S. National Science Foundation (NSF) Research Grant through the Division of Computing and Communication Foundations under Grant 1319800. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Ling Shao. *(Corresponding author: Honggang Qi.)*

D. Du and H. Qi are with the School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100190, China, and also with the Key Laboratory of Big Data Mining and Knowledge Management, University of Chinese Academy of Sciences, Beijing 101408, China (e-mail: dawei.du@vipl.ict.ac.cn; hgqi@jdl.ac.cn).

W. Li is with the Department of Computer Science and Engineering, Lehigh University, Bethlehem, PA 18015 USA (e-mail: wbli@lehigh.edu).

L. Wen and S. Lyu are with the Computer Science Department, University at Albany, State University of New York, Albany, NY 12222 USA (e-mail: lwen@albany.edu; slyu@albany.edu).

Q. Huang is with the School of Computer and Control Engineering, University of Chinese Academy of Sciences, Beijing 100190, China, also with the Key Laboratory of Big Data Mining and Knowledge Management, University of Chinese Academy of Sciences, Beijing 101408, China, and also with the Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China (e-mail: qmhuang@jdl.ac.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2016.2570556

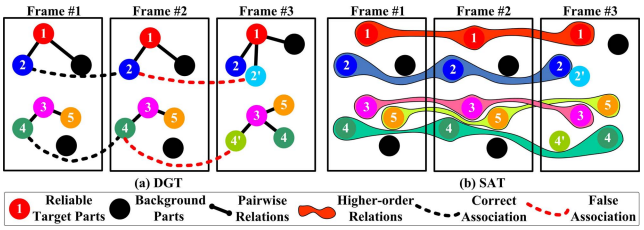


Fig. 1. Importance of higher-order dependencies in tracking. We compare DGT [8] that uses pairwise dependencies and our SAT that uses higher-order dependencies among target parts. To improve the clarity, we illustrate only partial set of parts in both methods. (a) DGT fails when multiple parts with similar appearance or geometric structure appear in close proximity, *e.g.*, Node 2 and Node 4. (b) Our SAT focuses on capturing higher-order dependencies of multiple parts of the frames in a *frame buffer* (described in Section III-A), which is effective to handle more difficult deformation and occlusion challenges compared to DGT. Best viewed in color.

objective function to produce candidate parts. Then, we construct a hyper-graph to capture the higher-order dependencies among candidate parts across multiple frames. Specifically, the nodes of the hyper-graph correspond to the candidate parts, and the hyper-edges encode the higher-order dependencies (as consistencies in both appearance and motion) of the candidate parts across multiple frames. Motivated by [44], we adopt a pairwise updating algorithm [27] to extract parts belonging to the target. Finally, the target state (*i.e.*, the center and scale of the target) is determined by comprehensively analyzing the searched part states.

Our method is related with two previous methods [8], [44]. While these methods share some similarities, we would like to highlight some important differences.

- Although both of our method and DGT [8] are super-pixel based tracking methods, DGT [8] considers matching of super-pixels in two consecutive frames, while our method focuses on exploiting temporal higher-order dependencies among them in multiple consecutive frames. This difference is made clear graphically in Fig. 1.
- A previous multi-target tracking method [44] also constructs a hyper-graph model hierarchically in an offline manner, where *every node* that corresponds to a detected bounding box is required to be classified to its belonging trajectory correctly. In contrast, in our method, the extracted dense subgraphs can be used to determine the optimal target state online, rather than classify every super-pixel (each super-pixel corresponds to a node in the constructed hyper-graph) into the extracted dense subgraphs precisely.

In comparison with the preliminary version [25], this work is substantially enhanced on the aspects of description of the model and experimental evaluation. First and foremost, different from the preliminary version that extracts candidate parts by a simple online Support Vector Machine (SVM) model, we employ a Markov Random Field (MRF) based segmentation method that incorporates both generative HSV histogram and discriminative SVM model, similar to [8]. We also significantly strengthen the experiment part. Specifically, we use two metrics, *precision plot* and *success plot*, to compare our method with 19 state-of-the-art tracking methods on a new collected dataset for *online deformable tracking*, making the comparison more thorough and comprehensive. Furthermore, we provide a detailed experimental analysis on the influence of some important parameters in our method.

A. Contributions

To the best of our knowledge, our work is the first online tracking method to fully exploit the higher-order temporal correlations among target parts. We show that the structure-aware hyper-graph can effectively capture such dependencies and lead to the improved performance and robustness of the tracker. To summarize, the main novelties of our method include

- a new formulation of the online deformable object tracking problem as the dense subgraph searching on the structure-aware hyper-graph, which is efficiently solved by the pairwise updating algorithm [27];
- a new deformable object tracking dataset, *Deform-SOT* dataset, which consists of 50 challenging sequences with full annotations to facilitate the evaluation of deformable object tracking algorithms;
- extensive experiments to demonstrate the effectiveness and robustness of our method over the state-of-the-art online deformable object trackers.

The rest of the paper is organized as follows. In Section II, we review relevant previous works. Section III describes our tracking method in detail, and Section IV presents experimental results. Section V concludes the paper with discussions on future works.

II. Related Works

In this section, we review the most relevant works regarding online tracking methods and available datasets for evaluating tracking methods.

A. Online Object Tracking Review

1) Bounding Box Based Tracker: The majority of existing online tracking methods are based on locating the bounding box of the target (*e.g.*, [18], [22], [26], [42]), and rely on models of appearance variations of the target in the bounding box (*e.g.*, [11], [29], [32], [50]). More recently, several algorithms [12], [16], [41], [46], [48] also exploit contextual information in order to increase tracking robustness. For instance, Yang *et al.* [46] integrate multiple automatically discovered auxiliary objects, which are the items that are frequently co-occurrent with the target, into tracking process by frequent item mining to improve the tracking performance. The method presented in [16] uses the general Hough Transform to learn the supporting model, *i.e.*, several informative features used to help predict the position of the target. Dinh *et al.* [12] extract local regions and key points consistently co-occur with the target to verify the genuine one. Wen *et al.* [41] present a spatio-temporal context model to exploit local relations between the target and nearest background objects to improve the tracking performance. Zhang *et al.* [48] formulate the spatio-temporal relations between a target and its local context based on Bayesian framework, to capture the statistical correlation among pixel intensities and positions from the

target and its surrounding background. Although bounding boxes provide accurate locations of the target in some cases, they are insufficient for accurate tracking non-rigid objects that undergo drastic deformation and severe occlusion.

2) Part Based Tracker: To alleviate the difficulties posed by object deformation and occlusion, in recent years there have also been many works on part-based trackers,[1] *e.g.*, [2], [8], [9], [39], [47]. Adam *et al.* [2] segment the target into horizontal and vertical patches to describe the appearance of the target and show this strategy can improve the performance in handling partial occlusion challenge. Yao *et al.* [47] train an online latent structured SVM to predict the location of parts, which employ both a global object box and a small number of part boxes to approximate the deformable target for robustness. Similarly, a discriminative learning method is presented in [39] to infer the position, shape and size of each part, using the Metropolis-Hastings algorithm integrated with an online SVM. A dynamic structure graph based tracker is used in [8] to formulate the tracking problem as graph matching between the geometric structure graph of the target and that of the candidate target proposals graph. Cehovin *et al.* [9] propose a novel coupled-layer visual model that combines the target's global and local appearance to handle tracking objects which undergo rapid and significant appearance change. However, existing part based models give less consideration to motion coherence of target parts in multiple consecutive frames, which is demonstrated as an important cue to help tracking.

3) Segmentation Based Tracker: Many recent tracking algorithms directly segment out the foreground tracking target, which are more precise than only the location and size information provided by the bounding boxes. Given the initial foreground segmentation, Godec *et al.* [15] present a patch-based Hough forest algorithm to complete the tracking task. In [13], a pixel-based tracking method is presented for non-rigid object tracking problem, which consists of two components: a generalized Hough Transform using a detector with pixel based descriptors and a probabilistic segmentation method based on a global model for foreground and background. Moreover in [10], a level set algorithm is employed to precisely segment the target from the background to complete the non-rigid object tracking task. Hong *et al.* [19] propose a hierarchical appearance representation model for tracking, which exploits shared information across multi-level quantization of an image space, *i.e.*, pixels, super-pixels and bounding boxes, by a probabilistic graphical model. Wen *et al.* [43] propose a joint online tracking and segmentation algorithm, which integrates the multi-part tracking and segmentation into a unified energy optimization framework.

B. Object Tracking Dataset Review

To date, there exists only a handful of datasets used for the evaluation of online deformable object tracking methods. Wu *et al.* [45] present a tracking benchmark for online object tracking task, and part of sequences include human or face targets with moderate deformation. Kristan *et al.* [37] host a competition and present a novel tracking evaluation strategy to rank the performance of the tracker in 25 sequences. In [36], a RGB-D tracking dataset is presented, where reliable depth information of objects is obtained by depth sensors. Felsberg *et al.* [14] propose a thermal object tracking benchmark, where the sequences are collected from seven different sources using eight different types of sensors. Li *et al.* [24] describe a large dataset, but most of their sequences focus on person and face tracking. The existing datasets contain a large portion of rigid objects or moderately deformable objects such as person and face. In contrast, we collect a dataset including 50 challenging sequences with full manual annotations, which focuses on *online deformable object tracking* in unconstrained environments.

III. Methodology

In this section we describe in detail our tracking method, the overall procedure of which is presented in Fig. 2. As described in Section I, our method processes multiple consecutive frames (defined as *frame buffer*, see Fig. 2(a)) at a time to incorporate the higher-order structural dependencies of the target. This is the major difference between our method and most existing methods, *e.g.*, [8], [16], [18], [30], [41], [46]–[48], that consider only two consecutive frames.

We first segment each frame into super-pixels, and collect candidate parts in each frame of a frame buffer by the MRF based segmentation method, shown in Fig. 2(b). After that, we construct a structure-aware hyper-graph, whose nodes correspond to the candidate parts in a frame buffer and hyper-edges correspond to the higher-order dependencies among the parts (see Fig. 2(c)). We then group super-pixels into sub-graphs with appearance and motion-consistent target parts corresponding to the object across multiple frames (see Fig. 2(d)). Assembling all parts belonging to the target, we find the precise location and boundary of the target (see Fig. 2(e)) and output the location of the target (see Fig. 2(f)). These steps are also summarized in Algorithm 1.

A. Extracting Candidate Parts

To accommodate the online nature of the tracking algorithm, we use a frame buffer to predict the location, appearance and boundary of the target.[2] Similar to [8], we first apply the SLIC algorithm [1] to segment each frame in the frame buffer into super-pixels.[3] We then form the energy function to generate candidate parts as

$$E(\boldsymbol{\ell}) = \sum_{p \in \mathcal{P}} D_p(\ell_p) + \sum_{(p,q) \in N} V_{p,q}(\ell_p, \ell_q), \tag{1}$$

with regards to the labeling $\boldsymbol{\ell}$ of the super-pixel set $\mathcal{P}$, in the current frame. $D_p(\ell_p)$ is the unary energy corresponding to likelihood of super-pixel p belonging to foreground ($\ell_p = 1$)

[1]The "part" means the local region of the target with similar attributes such as appearance, motion, etc. Usually we extract parts by grid rectangles, super-pixels, and even pixels.

[2]Let the frame buffer include τ frames to be processed at a time. When the latest frame index t such that $t \leq \tau$, we make $\tau - t + 1$ copies of the earliest frame of the frame buffer. For example, when we track the target in Frame #3, and $\tau = 5$, we collect the frame buffer by copying 3 times of Frame #1 firstly and concatenating Frame #2 and Frame #3 subsequently.

[3]The SLIC super-pixel representation is calculated by the vlfeat toolbox in http://www.vlfeat.org/.

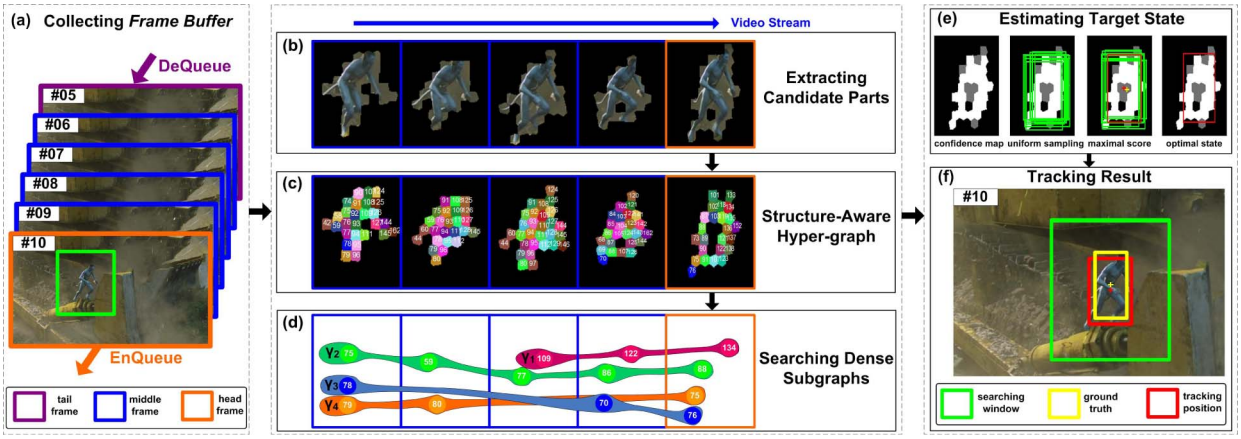


Fig. 2. The overall tracking procedure of the proposed method. (a) Frame buffer consists of several consecutive frames in the video. The green rectangle represents the searching window. (b) Candidate parts are extracted by the MRF based segmentation method to construct the structure-aware hyper-graph. (c) Based on the pairwise updating algorithm [27], the dense subgraphs involving appearance and motion-consistent parts are exploited. The nodes in the subgraph are assigned with the same color. (d) The nodes in the hyper-graph are connected to form dense subgraphs (*e.g.*, $\gamma_1, \cdots, \gamma_4$), described by curved surfaces marked with similar color. To improve the clarity, we illustrate only partial dense subgraphs. (e) From the confidence map determined by exploited dense subgraphs, the optimal target state is estimated by the coarse-to-fine strategy strategy (explained in Section III-D). The green and red rectangle represent sampled and optimal target state, respectively. (f) The output from the algorithm is the target location. Best viewed in color.

Algorithm 1 Structure-Aware Tracker

Input: A tracking video sequence and the manually labeled bounding box in the first frame

Output: The target state in each frame $\{\varrho^*, s^*\}$

1: **while** Load the current frame buffer. **do**
2: Over-segment the images in the current frame buffer into a super-pixel set $\mathcal{P}$.
3: Extract candidate parts $\{p|\ell_p = 1\}$ by the appearance model with (1).
4: Construct a structure-aware hyper-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ with (6).
5: Calculate the corresponding appearance and motion weights $\mathcal{W}(e)$ with (7) (8) (9).
6: Search dense subgraphs Γ from the structure-aware hyper-graph $\mathcal{G}$ with (11).
7: Determine the set of appearance and motion-consistent target parts from dense subgraphs Γ^* without conflicts in Section III-C.
8: Determine the optimal target state $\{\varrho^*, s^*\}$ via coarse-to-fine strategy with (13) (14).
9: Update the structure-aware hyper-graph and appearance model (*i.e.*, discriminative SVM classifier and generative HSV histogram) in Section III-E.
10: **end while**

or background ($\ell_p = 0$). $V_{p,q}(\ell_p, \ell_q)$ is the binary energy encoding the consistency between pairs of spatially neighboring super-pixels, which encourages the target to be a collection of connected parts with similar appearance. N is the spatial neighborhoods of super-pixels: two super-pixels p and q are in N if the Euclidean distance between their centers in the image plane satisfies, *i.e.*, $\|d(p) - d(q)\|_2 \leq 2\sqrt{\kappa}$. $\kappa = W \cdot H/\rho$ is the number of pixels in each super-pixel, where ρ is the number of super-pixels in the searching window with width W and height H. The energy function (1) is minimized with the graph cut algorithm [5], leading to a coarse labeling of each super-pixel as belonging to the target and the background.

1) Unary Energy: We define the unary energy $D_p(\ell_p)$ as

$$D_p(\ell_p) = \lambda_1 \cdot \Theta_p(\ell_p) + \Psi_p(\ell_p), \tag{2}$$

where λ_1 controls the influence of generative term $\Theta_p(\ell_p)$ and discriminative term $\Psi_p(\ell_p)$. The two terms are employed to support each other and model appearance for more robustness, which are further defined as follows.

- Generative term:

$$\Theta_p(\ell_p) = \begin{cases} -\dfrac{1}{n_p}\sum_{i=1}^{n_p} \log(\mathcal{H}_f(c_i) + \varsigma) & \ell_p = 1, \\ -\dfrac{1}{n_p}\sum_{i=1}^{n_p} \log(\mathcal{H}_b(c_i) + \varsigma) & \ell_p = 0, \end{cases} \tag{3}$$

  where c_i corresponds to the HSV bin value of pixel i, and n_p the number of pixels in super-pixel p. $\mathcal{H}_f(c_i)$ and $\mathcal{H}_b(c_i)$ are the probability of c_i from the normalized foreground and background histograms, respectively. $\varsigma = 0.0001$ is added to avoid taking logarithm of zeros. We use 16 bins for HSV histograms for each channel.

- Discriminative term:

$$\Psi(\ell_p) = \begin{cases} -\lambda_2 \cdot \mathcal{S}(p) & \mathcal{S}(p) \geq 0, \ell_p = 1, \\ \lambda_2 \cdot \mathcal{S}(p) & \mathcal{S}(p) \geq 0, \ell_p = 0, \\ -\mathcal{S}(p) & \mathcal{S}(p) < 0, \ell_p = 1, \\ \mathcal{S}(p) & \mathcal{S}(p) < 0, \ell_p = 0, \end{cases} \tag{4}$$

  $\mathcal{S}(p) = \vec{a} \cdot \varphi(p) + \eta$ is the SVM classification score of super-pixel p belonging to the target, where $\vec{a}$ is the normal vector to the hyperplane and η controls the offset of the hyperplane from the origin along the normal vector. $\varphi(p)$ is the HSV color feature extracted from super-pixel p with 6 bins for each channel. λ_2 controls the sensitivity of the classification score.

The generative HSV histogram and discriminative SVM classifier [35] are initialized based on the manually annotated ground-truth provided in the first frame. Specifically, the positive samples are selected in the target bounding box, while the samples outside the box are considered as negative ones. The only difference of two terms in initialization is that HSV histogram accumulates pixels while SVM classifier

collects super-pixels. They are incrementally updated every fixed number of frames, which is explained in Section III-E.

2) Binary Energy: We define the binary energy $V_{p,q}(\ell_p, \ell_q)$ by calculating the similarities of super-pixels in appearance and motion, as

$$V_{p,q}(\ell_p, \ell_q) = \mathbb{I}(\ell_p \neq \ell_q) \cdot \big(\lambda_3 \cdot \Delta_c(p,q) + (1-\lambda_3) \cdot \Delta_u(p,q)\big) \quad (5)$$

where $\mathbb{I}(\cdot) = 1$ if its argument is true, and 0 otherwise, and λ_3 balances the influence of appearance and motion cues. $\Delta_c(p,q) = \|c_p - c_q\|_2$ and $\Delta_u(p,q) = \|u_p - u_q\|_2$ correspond to the Euclidean distance between the average values of pixels of two spatial neighboring parts p and q in the RGB color space c and optical flow field u,[4] respectively.

B. Structure-Aware Hyper-Graph

Given the collected candidate parts, we construct the structure-aware hyper-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ encoding the dependencies among candidate parts. Specifically, each node in the node set $\mathcal{V}$ corresponds to a candidate part, and each hyper-edge in $\mathcal{E}$ represents the relations among the nodes, *i.e.*,

$$\begin{cases} \mathcal{V} = \{v_i\} = \{p | \forall p \in \mathcal{P}, \ell_p = 1\} \\ \mathcal{E} = \{e | \forall v_i, v_j \in e, t_i \neq t_j, \|d(v_i) - d(v_j)\|_2 \leq \varepsilon\} \end{cases} \quad (6)$$

where v_i and v_j are the i-th and j-th nodes, t_i and t_j are the frame index of the i-th and j-th nodes. $e = (v_1, \cdots, v_m)$ is a hyper-edge enclosing a subset of nodes, and m is the order of the hyper-edge and fixed for all hyper-edges, *e.g.*, $m = 3$ in this work. $\|d(v_i) - d(v_j)\|_2$ is the Euclidean distance between the center of v_i and v_j in the image plane, and the spatial distance threshold is set as $\varepsilon = 25$ pixels in the subsequent experiment.

Each hyper-edge e in $\mathcal{G}$ is associated with the weight $\mathcal{W}(e)$, *i.e.*,

$$\mathcal{W}(e) = \frac{1}{m} \cdot \sum_{v_i, v_j \in e, t_i < t_j} \lambda_4 \cdot \psi_{aw}(v_i, v_j) + \lambda_5 \cdot \psi_{mw}(e), \quad (7)$$

where $\psi_{aw}(v_i, v_j)$ is the appearance weight between v_i and v_j, and $\psi_{mw}(e)$ is the motion weight of parts coupled in e, and λ_4 and λ_5 are the balance factors reflecting the contributions of the two weight terms.

1) Appearance Weight: The appearance weight is computed from HSV color features of v_i and v_j, as

$$\psi_{aw}(v_i, v_j) = \exp(-\mathcal{X}^2(v_i, v_j)/\sigma_{aw}^2), \quad (8)$$

where $\mathcal{X}^2(\cdot,\cdot)$ is the chi-square distance of features of two nodes, and σ_{aw} controls the sensitivity of the appearance term.

2) Motion Weight: The motion weight among nodes coupled in the hyper-edge provides an important cue for grouping nodes into subgraphs. Based on the assumption that target parts move smoothly in a short time interval, we compute motion weight based on fitting their motions using a simple linear model. Specifically, for each hyper-edge e, the parameters of the linear model $\{\hat{A}, \hat{B}\}$ are determined based on every node in e by least squares fitting, *i.e.*, $\mathcal{C} = \hat{A} \cdot t + \hat{B}$, where t is the frame index of the node. The motion affinity is then computed as the Euclidean distance of the center of each node in e to their predictions based on the fitted linear model, as

$$\psi_{mw}(e) = \exp(-\sum_{v_i \in e} \|\mathcal{C}(t_i) - d(v_i)\|_2^2 / \sigma_{mw}^2), \quad (9)$$

where $d(v_i)$ is center of node v_i, $\mathcal{C}(t_i)$ is the position of the projection of node v_i in frame t_i on the linear model, and σ_{mw} controls the sensitivity of the deviation to the fitted model.

C. Searching Dense Subgraphs

After constructing the hyper-graph, we search the dense subgraphs on it to determine the state of each target part. We call a subgraph of $\mathcal{G}$ "dense" if its nodes are interconnected by a large set of hyper-edges with high weights. First of all, we calculate the summation of the weights of hyper-edges in a subgraph as

$$\omega(\vec{y}) = \sum_{v_1, \cdots, v_m \in \mathcal{V}} \mathcal{W}(v_1, \cdots, v_m) \prod_{i=1}^{m} y_{v_i}, \quad (10)$$

where $\vec{y} = \{y_{v_1}, \cdots, y_{v_{|\mathcal{V}|}}\}$ is an $|\mathcal{V}| \times 1$ indicator vector of subgraph γ, *i.e.*, if $v_i \in \gamma$, $y_{v_i} = 1$; otherwise, $y_{v_i} = 0$. In other words, the weight of hyper-edge e contributes to $\omega(\vec{y})$ only if nodes $v_1, \cdots, v_m$ composing it belong to γ, *i.e.*, $\prod_{i=1}^{m} y_{v_i} = 1$. $|\mathcal{V}|$ is the cardinality. k-subgraph includes k nodes, *i.e.*, $\sum_{i=1}^{|\mathcal{V}|} y_{v_i} = k$, which can be inferred automatically. Since $\omega(\vec{y})$ reflects the strength of overall internal relations in subgraph γ, we use it as a criterion to detect a dense subgraph.

To extract such dense subgraphs fully, we set each node v in the hyper-graph as the starting node to search its neighborhood $\mathcal{N}(v)$. That is to say, we aim to find a subset of $\mathcal{N}(v)$ with $k-1$ nodes such that they jointly lead to a k-subgraph that has large weight of all hyper-edges. To describe different cardinality of subgraphs, it is more appropriate to use the average weights to reflect the confidence of the subgraph than the sum weights. Since there are k^m summands in (10), $\omega(\vec{x}) = \frac{1}{k^m}\omega(\vec{y})$ is the average of these entries. Combined with (10), the dense subgraph searching problem for the starting node v is formulated as

$$\begin{aligned} \vec{x}^* = \underset{\vec{x}}{\text{argmax}}\ \omega(\vec{x}) = \underset{x_{v_i}: v_i \in v \cup \mathcal{N}(v)}{\text{argmax}} \sum_{e \in \mathcal{U}(v)} \mathcal{W}(e) \prod_{i=1}^{m} x_{v_i}, \\ \text{s.t.} \sum_{i=1}^{|\mathcal{V}|} x_{v_i} = 1, \quad \forall v_i \in v \cup \mathcal{N}(v) : x_{v_i} \in \{0, \frac{1}{k}\}, \\ k \geq m, \end{aligned} \quad (11)$$

where $\mathcal{U}(v)$ is the hyper-edge set corresponding to the node set $v \cup \mathcal{N}(v)$. The indicator vector $\vec{x} = \frac{\vec{y}}{k}$ is constrained by $\sum_i x_{v_i} = 1$ because of $\sum_i y_{v_i} = k$. Moreover, each coordinate of $\vec{x}$ is either 0 or $\frac{1}{k}$. To avoid the degeneracy problem, the number of nodes in the subgraph is larger than or equal to the order of the hyper-graph, *i.e.*, $k \geq m$.

The discrete optimization problem in (11) is a known NP-hard problem. To obtain a solution, we relax the discrete constraint $x_{v_i} \in \{0, \frac{1}{k}\}$ to its continuous counterpart $x_{v_i} \in [0, \frac{1}{k}]$, and thus convert the problem into a continuous one. A practical and efficient solution based on pairwise

[4]The optical flow is calculated by the method in [7].

updating is given in [27]. After solving the optimization problem, we can determine subgraph γ and corresponding confidence ω w.r.t. $\vec{x}^*$ for the starting node v as

$$\begin{cases} \gamma(\vec{x}^*) = \{v_i | v_i \in \mathcal{V}, x^*_{v_i} > 0\}, \\ \omega(\vec{x}^*) = \sum_{v_i \in \gamma(\vec{x}^*), e \in \mathcal{U}(v)} \mathcal{W}(e) \prod_{i=1}^{m} x^*_{v_i}, \end{cases} \tag{12}$$

Since we conduct subgraph searching based on each node in the hyper-graph, one node may appear in multiple subgraphs. Similar to [44], we filter out the conflicts involved in subgraphs. We first produce an ordered subgraphs set $\Gamma = \{\gamma_1, \cdots, \gamma_{|\mathcal{V}|}\}$ according to the confidence values in descending order. Let the set of subgraphs without conflicts be Γ^*. We set $\Gamma^* = \emptyset$ at first and add the sorted dense subgraphs sequentially. Then, for each non-empty subgraph $\gamma_i \in \Gamma, \gamma_i \neq \emptyset$, we check whether it intersects with all members in Γ^*. If $\gamma_i \cap \gamma^*_j = \emptyset, \forall j, \gamma^*_j \in \Gamma^*$, we add it directly to Γ^*, *i.e.*, $\Gamma^* \leftarrow \Gamma^* \cup \{\gamma_i\}$; otherwise, we remove the overlapping part from γ_i and then add it to Γ^*, *i.e.*, $\hat{\gamma}_i \leftarrow \gamma_i / \gamma^*_j, \forall j, \gamma^*_j \in \Gamma^*$ and $\Gamma^* \leftarrow \Gamma^* \cup \{\hat{\gamma}_i\}$. Thus target parts can be extracted from such dense subgraphs in Γ^*.

D. Estimating Target State

Given target parts from Γ^*, we can infer the optimal target state, including center ϱ^* and scale s^* of the target in two phases similar to the coarse-to-fine strategy used in [8].

In the first phase, we estimate an initial target state $\{\hat{\varrho}_t, \hat{s}_t\}$ in each frame t of the frame buffer. The initial scale in the current frame is given by the optimal scale in the previous frame, *i.e.*, $\hat{s}_t = s^*_{t-1}$. For more clarity, we omit the frame index t in the following equations. The initial center is obtained by calculating the weighted mean of the part center $d(v_i)$ in $\gamma^* \in \Gamma^*$, *i.e.*,

$$\hat{\varrho} = \sum_{v_i \in \gamma^*} d(v_i) \cdot \frac{\omega_{v_i}}{\sum_{v_i \in \gamma^*} \omega_{v_i}}, \tag{13}$$

where ω_{v_i} is the confidence value of subgraph γ^* including target part v_i, *i.e.*, if $v_i \in \gamma^*$, $\omega_{v_i} = \omega(\gamma^*)$.

In the second phase, we adjust the target center and scale with the perturbation term δ for a better location such that the tracking bounding box covers more foreground regions. Specifically, we form the optimization problem as

$$\begin{aligned} \{\varrho^*, s^*\} = \underset{\varrho, s}{\text{argmax}} \ & \left\{\lambda_6 \cdot \beta_{tp}(\varrho, s) + \beta_{cp}(\varrho, s) - \beta_{bp}(\varrho, s)\right\}, \\ \text{s.t. } & \varrho = \hat{\varrho} + \delta, \quad s = \hat{s} + \delta, \ \delta \in [-\zeta, \zeta], \end{aligned} \tag{14}$$

where $\beta_{tp}(\varrho, s)$ and $\beta_{cp}(\varrho, s)$ are the number of pixels in the target parts and other candidate parts within the bounding box centered at ϱ with scale s, respectively. $\beta_{bp}(\varrho, s)$ is the number of pixels located outside the box that a target part covers. λ_6 is a balance factor to boost the influence of the pixels in the target parts. ζ is set as the mean diameter of super-pixels to determine the range of the perturbation term.

As shown in Fig. 2(e), the optimal target state $\{\varrho^*, s^*\}$ is obtained by optimizing (14) using a sampling strategy. In practice, we generate 1000 candidate states around the initial target state $\{\hat{\varrho}, \hat{s}\}$. The center is sampled uniformly in the range $[\hat{\varrho} - \zeta, \hat{\varrho} + \zeta]$ and the scale (including width and height) is sampled uniformly in the range $[\hat{s} - \zeta, \hat{s} + \zeta]$. Then we select the state with the maximal score in (14) as the optimal one.

E. Online Updating

Online updating is an important step to prevent drifting problem for online tracking. For better efficiency, the frame buffer is implemented as a *queue*. As illustrated in Fig. 2(a), we add the head frame (*EnQueue*) and drop the tail frame (*DeQueue*) to collect a new frame buffer. Then the structure-aware hyper-graph is updated by extracting candidate parts in the new frame buffer.

Moreover, because of possible significant change of target appearance, we update the appearance model (*i.e.*, SVM classifier and HSV histogram) in (2) after a fixed number of incoming frames. The target parts (*i.e.*, nodes from dense subgraphs) in the target bounding box are used as positive training samples, and other parts outside the target bounding box are considered as negative ones. To reduce tracking drift caused by outliers, data is collected from both the current frame buffer and the first frame. Thus the foreground/background histogram $\hat{\mathcal{H}}^t_\star$ ($\star \in \{f, b\}$) at frame t is updated incrementally from the corresponding training samples, *i.e.*,

$$\begin{cases} \hat{\mathcal{H}}^{t+1}_\star = \hat{\mathcal{H}}^t_\star + \Xi^t_\star - \Xi^{t-\tau}_\star, & \text{if } t > \tau \\ \hat{\mathcal{H}}^{t+1}_\star = \sum_{i=1}^{t} \Xi^i_\star + \Xi^1_\star, & \text{if } 1 < t \leq \tau \\ \hat{\mathcal{H}}^{t+1}_\star = \Xi^1_\star, & \text{if } t = 1 \end{cases} \tag{15}$$

where $\Xi^t_\star$ is the count of HSV bin values of training samples at frame t. Then the normalized version $\mathcal{H}^t_\star$ can be inferred from $\hat{\mathcal{H}}^t_\star$ easily.

IV. Experiments

A. Dataset

We collect 50 challenging video sequences that focus on tracking totally deformable targets in unconstrained environments, termed as the *Deform-SOT* dataset, to evaluate online deformable object tracking methods. Of the 50 videos, 20 sequences have been used in previous works, *e.g.*, *avatar* [8], *carscale* [45], and *waterski* [17], while the remaining 30 sequences are collected by us from the Internet, such as *bike*, *lola* and *uneven-bars*. The range of video frames is approximately from 100 frames to 1300 frames and all frames are annotated with bounding boxes. In Fig. 3, we show annotations in the first frame of the dataset. The collected sequences are diverse with respect to object categories, camera viewpoints, sequence lengths and challenging levels. Different from 11 attributes for general object tracking in [45], our dataset includes videos reflecting typical challenges in tracking, as described below:

- **large deformation.** The non-rigid target occurs with local structural or significant deformation in shape.
- **severe occlusion.** The target is partially or fully occluded by other objects or background.
- **abnormal movement.** The target moves abnormally, including fast motion, in-plane and out-of-plane rotation and other complex motions, etc.
- **illumination variation.** The illumination in the target region is moderately to significantly changed.

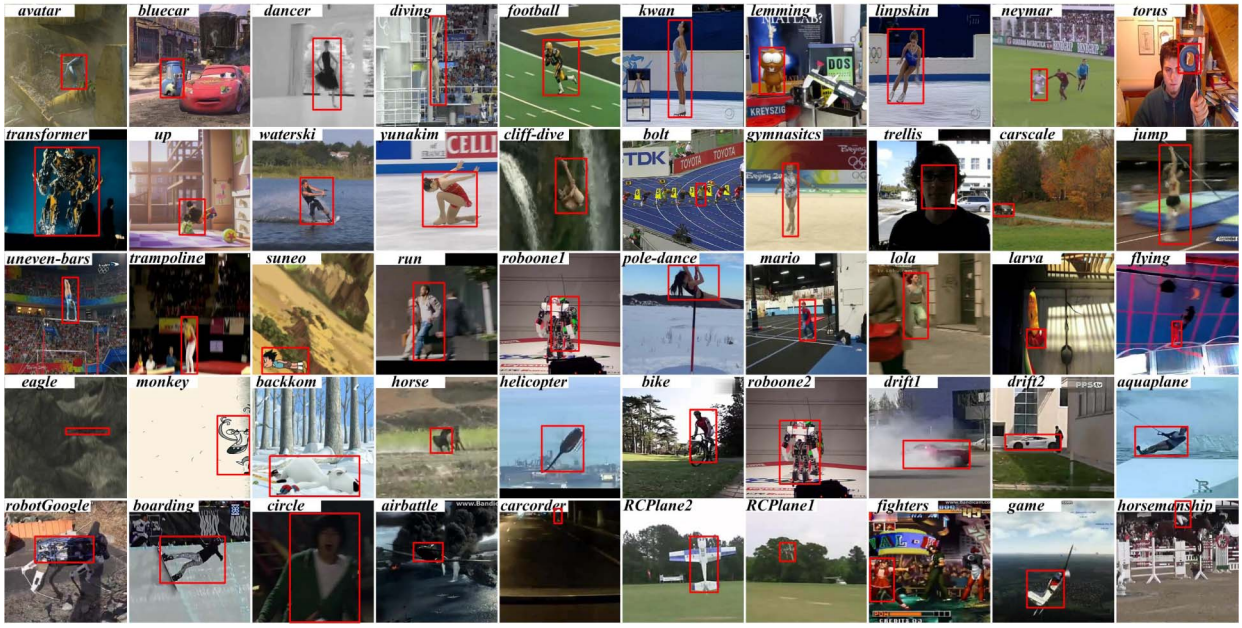


Fig. 3. The annotated target location in the first frame from the *Deform-SOT* dataset. The first two rows correspond to videos that have been used in previous works and the last three rows are new videos we collected from the Internet. Best viewed in color.

TABLE I

THE *Deform-SOT* DATASET. THE NEWLY COLLECTED SEQUENCES ARE NAMED WITH BOLD FONT, AND THE OTHER ONES ARE USED IN PREVIOUS WORKS

Main Challenges	Tracking Sequences
Large Deformation (22 sequences)	*avatar*[8], ***backkom***, ***boarding***, *bolt*[38], *dancer*[30], *diving*[23], ***eagle***, *gymnastics*[23], ***fighters***,*football*[33], ***horse***, *jump*[45], *kwan*[33], *lipinski*[17], *neymar*[8], ***pole-dance***, ***trampoline***, *transformer*[23], ***uneven-bars***, *up*[8], *waterski* [17], *yunakim*[17]
Severe Occlusion (20 sequences)	***aquaplane***, *avatar*[8], *bluecar*[8], ***boarding***, *carscale*[45], ***drift1***, ***drift2***, ***flying***, *football*[33], ***horse***, ***larva***, *lemming*[34], ***lola***, *neymar*[8], ***roboone1***, ***roboone2***, ***robotGoogle***, ***run***, ***trampoline***, *transformer*[23]
Abnormal Movement (26 sequences)	***airbattle***, *avatar*[8], ***backkom***, ***bike***, *bluecar*[8], ***boarding***, ***circle***, *cliff-dive*[13], *diving*[23], ***drift1***, ***drift2***, ***fighters***, ***game***, ***helicopter***, ***horsemanship***, ***larva***, *lipinski*[17], ***mario***, ***RCPlane1***, ***RCPlane2***, ***suneo***, *torus*[9], ***trampoline***, ***uneven-bars***, *waterski*[17], *yunakim*[17]
Illumination Variation (17 sequences)	***airbattle***, ***bike***, ***carcorder***, ***circle***, ***drift1***, ***drift2***, ***eagle***, ***flying***, ***game***, ***helicopter***, *lemming*[34], ***RCPlane1***, ***RCPlane2***, ***robotGoogle***, *transformer*[23], *trellis*[45], *up*[8]
Scale Change (22 sequences)	*avatar*[8], ***aquaplane***, ***carcorder***, *carscale*[45], ***circle***, ***drift1***, ***drift2***, ***eagle***, ***helicopter***, ***horse***, ***horsemanship***, *jump*[45], ***lola***, ***mario***, ***monkey***, *transformer*[23], ***RCPlane1***, ***RCPlane2***, ***robotGoogle***,***suneo***, *waterski*[17], *yunakim*[17]
Background Clutter (15 sequences)	***airbattle***, ***carcorder***, ***drift1***, ***drift2***, ***eagle***, *football*[33], ***fighters***, ***horsemanship***, ***larva***, ***mario***, ***RCPlane1***, ***RCPlane2***, ***run***, *trellis*[45], *up*[8]

- **scale change.** The scale of the target changes drastically.
- **background clutter.** The background near the target has the similar appearance as the target.

A detailed break-down descriptions of the dataset are given in Table I.

B. Implementation Details

The proposed structure-aware tracker is implemented with MATLAB and C and runs at 0.5 frames per second (FPS), on a machine with a 2.9 GHz Intel i7 processor and 16 GB memory.[5]

The parameters in our algorithm are chosen empirically by making grid search of one parameter over a range of values while keep other ones fixed, and all of them are fixed in the experiments as follows. The frame buffer consists of $\tau = 5$ consecutive frames. For the searching window, we search the target location in the current frame by 2.5 times size of the previous one. For the SLIC method used to generate the candidate parts, the number of pixels in each super-pixel is set as $\kappa = 100$, and the range of number of super-pixels is set as $[100, 200]$. For the update process, the generative HSV histogram in (3) is updated every frame, and the discriminative SVM classifier in (4) is updated every 3 frames. In the energy function (1), we take the following default values: $\lambda_1 = 0.9$, $\lambda_2 = 15$, and $\lambda_3 = 0.5$. The balance factors in the weight calculation in (7) are $\lambda_4 = 1.0$ and $\lambda_5 = 2.0$. We set $\sigma_{aw}^2 = 2$ in (8), and $\sigma_{mw}^2 = 16$ in (9). In (14), the term $\lambda_6 = 3.0$. The impact of some important parameters in our method are analyzed and discussed in Section IV-E.

C. Experiment Setup

We evaluate the proposed method against 19 state-of-the-art trackers, as described in Table II, on the *Deform-SOT* dataset. For fair comparison, we use the *same* initial bounding box of the first frame of each sequence for all trackers.

[5]We make the source code of our tracker and the *Deform-SOT* dataset available on our website: https://sites.google.com/site/daviddo0323/.

TABLE II
THE COMPARED STATE-OF-THE-ART TRACKERS IN THE EXPERIMENT

Main Strategies	Tracking Methods
bounding box based methods (8 trackers)	Incremental Visual Tracker (IVT [26]), $\ell 1$ tracker (L1T [29]), Tracking-Learning-Detection tracker (TLD [21]), Multiple Instance Learning tracker (MIL [4]), Structured output tracker (Struck [18]), Compressive Tracker (CT [49]), Multi-Task sparse learning Tracker (MTT [50]), Spatio-Temporal structural context Tracker (STT [40]), Spatio-Temporal Context tracker (STC [48]), Color Name tracker (CN [11])
part based methods (11 trackers)	Fragment tracker (Frag [2]), Super-Pixel Tracker (SPT [38]), Sparsity-based Collaborative Model based tracker (SCM [51]), Locally Orderless Tracker (LOT [31]), Adaptive Structural Local sparse Appearance model based tracker (ASLA [20]), Latent Structural Learning tracker (LSL [47]), Local and Global Tracker (LGT [9]), Dynamic Graph Tracker (DGT [8]), Temporally Coherent Part based tracker (our prior work TCP [25])

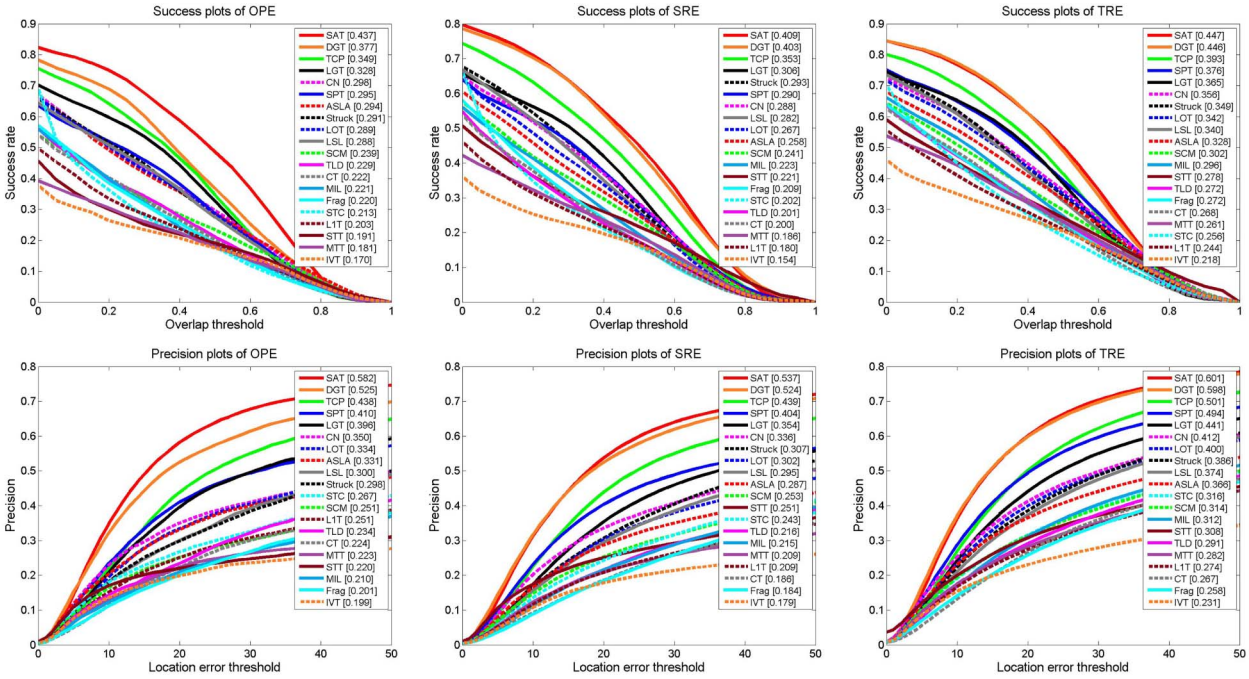


Fig. 4. The success and precision plots over the dataset using OPE, SRE and TRE. The representative scores for each tracker are reported in the legends. Best viewed in color.

The experimental results of other trackers are reproduced from the available source codes with recommended parameters. Then we use two popular measures, precision plot and success plot, to compare performances. We choose these two measures because they contain more comprehensive information than the other metrics such as Center Location Error (CLE) and Success Rate (SR). The precision plot shows the percentage of successfully tracked frames vs. the center location error in pixels, which ranks the trackers as *precision score* at 20 pixels. The success plot draws the percentage of successfully tracked frames vs. the bounding box overlap threshold, where Area Under the Curve (AUC) is used as *success score* for ranking. Specifically, we run the One-Pass Evaluation (OPE), Spatial Robustness Evaluation (SRE) and Temporal Robustness Evaluation (TRE) (see definitions in [45]) on the dataset.

D. Quantitative Evaluation

As shown in Fig. 4, the precision plot and success plot for the overall dataset indicate that our method performs against other 19 state-of-the-art ones. We further present the experimental results in Fig. 5 for each challenging scenario. In addition, visual comparison results of the top five performing methods on several sequences are displayed in Fig. 6. To aid in the discussion of evaluation, we group evaluated sequences based on 6 challenging factors and discuss the capabilities of all evaluated trackers under each factor subsequently.

1) Large Deformation: For the sequences with only local structural deformation (*e.g.*, *neymar* and *up*), bounding box based tracking method (*e.g.*, [11], [18], [48]) can work well. However, for the sequences with rapid large deformation (*e.g.*, *avatar*, and *transformer*), the large changes to the target appearances may render the bounding box based tracker ineffective. On the other hand, on these sequences, the part based trackers (*e.g.*, [8], [9], [38]) usually show better and more robust performance (see Fig. 6 for visual examples). This is mainly due to the fact that part based trackers using local (part-based) appearance model that can better adapt to target appearance changes during deformation. Compared with the previous part based trackers [2], [20], [38], [51], our tracker achieves better performance, due to the incorporation of temporal higher-order dependencies among target parts.

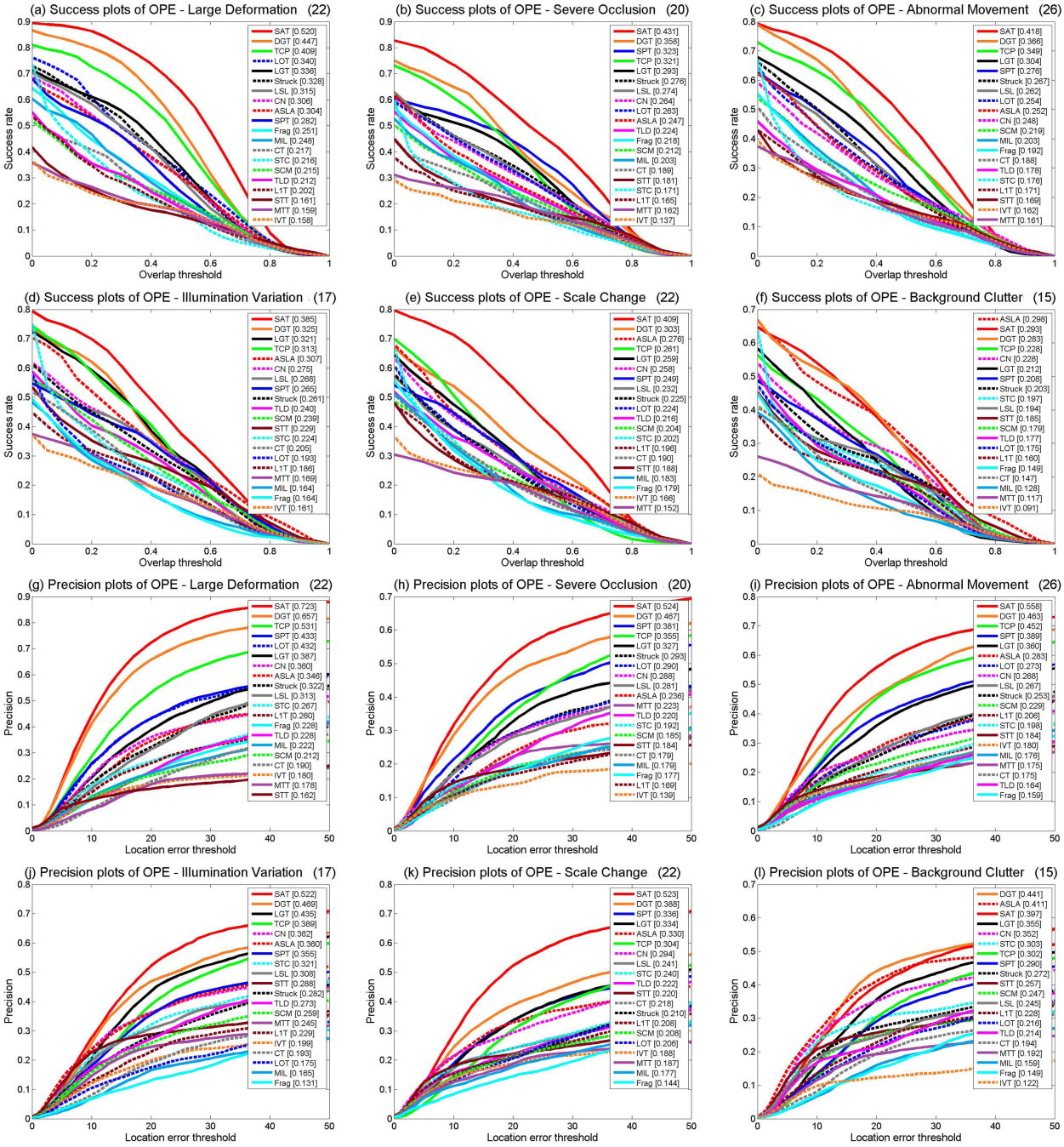


Fig. 5. The plots of OPE with different attributes. Best viewed in color.

2) Severe Occlusion: Mild to severe occlusions occur in sequences such as *bluecar*, *neymar* and *lola*, etc. Tracking methods usually use built-in mechanism to deal with occlusions, but some of them do not perform well in sequences involved with other challenges, such as large deformation and abnormal movement. When partial occlusion occurs, our tracker will reduce to the non-occluded target parts (*e.g.*, *trampoline* #120). However, our tracker is able to continue tracking the target after occlusion, as long as sufficient candidate parts are extracted. The extracted candidate parts facilitates recovering the right size of the target throughout the tracking process gradually. For example, as shown in Fig. 6, there exists a smaller bounding box due to an occlusion in *bluecar* #012 and #277 compared to the fixed-size tracker [11]. But after the occlusion, the bounding box of our tracker re-approaches the right size of the target while other trackers fail to handle the scale change in *bluecar* #372.

3) Abnormal Movement: The sequences in our dataset exhibit several types of abnormal movements, such as fast motion like jumping and diving (*avatar* and *up*), in-plane and out-of-plane rotation (*drift2* and *waterski*) and complex motion (*uneven-bars* and *trampoline*). Many existing trackers lose the target when such abnormal movements occur as shown in Fig. 6. Similarly, most trackers become unstable for

Fig. 6. Tracking results of top five performing methods in representative frames of a few example videos in our dataset. The name of sequences and the index of frames are shown in the top left of each figure. More results and visual comparisons can be found in http://youtu.be/NgN1u8z48bo.

complex motions involving combination of different types of abnormal movements such as spinning in *diving* and flipping in air in *trampoline*. The results in Fig. 5(c)(i) indicate that DGT [8] and our tracker perform much better than the other trackers on such challenging sequences. That is due to the fact that both methods benefit from coarse candidate parts generation by the MRF based segmentation method. Moreover, the dense subgraphs searching method filters out more noisy parts to resist the performance degradation caused by abnormal movement.

4) Illumination Variation: Target appearance is strongly affected by illumination variation. The tracking results of *up* and *trellis* in Fig. 6 show that frequent illumination variation causes some trackers to fail or not locate well. It is worth mentioning that CN [11] extracts complex color features combined with luminance from the target instead of simple color histogram, which underlies its favorable performance with regards to illumination variation. However, CN [11] with fixed-size bounding box performs poorly undergoing other challenges (*e.g.*, large deformation and scale change) simultaneously. On the other hand, DGT [8] and our tracker use an online updating mechanism for the appearance model, making them resistant to illumination variation. Furthermore, both trackers use the appearance-free information (the geometric structure in DGT [8] and the motion coherence in our tracker) to assist tracking. Considering multiple frames in a frame buffer, the higher-order dependencies in our tracker help in associating the reliable spatio-temporal target parts with severe illumination variation, as shown in Fig. 5(d)(j).

5) Scale Change: Sequences such as *carscale*, and *transformer* contain significant scale change of the target. Similar as handling the large deformation challenge, part based

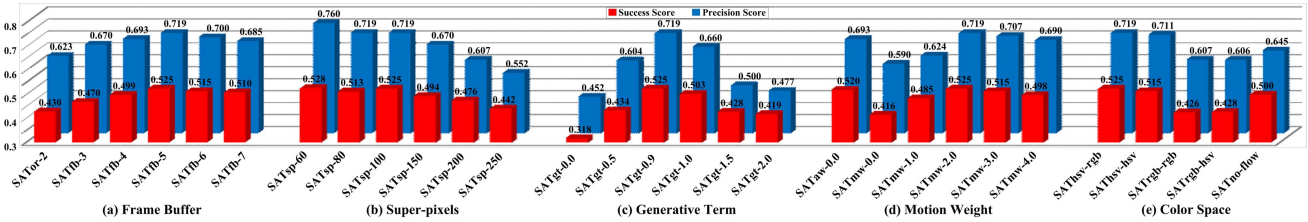


Fig. 7. The tracking performance for different parameters. The blue and red bar denote the precision score and success score (see definitions in Section IV-C), respectively. (a) different length of frame buffer; (b) different number of pixels in each super-pixel; (c) different balance factor of generative term; (d) different balance factor of motion weight; (e) different combinations of color spaces. Best viewed in color.

trackers [8], [9], [20] have more flexibility to handle changes of target scale, compared to other trackers assuming fixed size of the target [18], [47], [49]. However, some part based trackers [8], [20] employ strict geometric structural constraint, resulting in limiting the performance in the sequences with large scale change; while our method focuses on temporal coherence of target parts to extract the change of the whole target. The results in Fig. 5(e)(k) indicate our tracker achieves better performance, and examples showing difference in tracking results over sequences with the sizes of target undergoing large scale change, *e.g.*, *lola* and *avatar*, are given in Fig. 6.

6) Background Clutter: Many sequences in our dataset have backgrounds with similar appearance as the target, *e.g.*, *drift2*, *trellis*, and *up*, which can seriously impact tracking performance. This is alleviated by considering the context around each target, *e.g.*, [40], [48], or part based representation with structural constraints, *e.g.*, [8], [9], [20]. Since background and the target with similar appearance usually have inconsistent motion, the temporal higher-order dependencies of parts can distinguish them based on the determined main motion direction of target parts. As shown in Fig. 5(f)(i), our method shows favorably tracking performance compared to other methods on the sequences with strong background clutter.

E. Discussion and Analysis

We perform analyses on the effect of several important parameters in our method to the final tracking performance. Specifically, we study the influence of parameters including the length of frame buffer τ, the number of pixels in each super-pixel κ, the balance factor λ_1 in (2) and the balance factor λ_5 in (7). Besides, we justify the color spaces employed in our algorithm. For simplicity, the evaluations are performed on 15 sequences selected from the dataset with different kinds of challenges.

1) Frame Buffer: The length of frame buffer is an important parameter describing the number of frames integrated in constructing the structure-aware hyper-graph. We consider different values of the length of frame buffer, denoted as SATfb-τ. As shown in Fig. 7(a), SATfb-5 with a relatively longer length of frame buffer in general leads to better performance, which integrates more spatio-temporal context information for the target. However the computation complexity will increase with increased frame buffer. Moreover, the longer frame buffer may contain noisy frames to bring a slight decrease in performance for sequences with fast change in target appearance or background illumination. Stated thus, we set $\tau = 5$ in our method.

2) Higher-Order Dependencies: In Fig. 7(a), we report the performance of SATor-2 considering just pairwise relations, which brings a big accuracy loss compared to SATfb-τ considering higher-order dependencies. It indicates the importance and effectiveness of our hyper-graph representation in the tracking task.

3) Super-Pixels: The number of super-pixels controls the number of parts (*i.e.*, the number of nodes in the hyper-graph). As shown in Fig. 7(b), the number of pixels in each super-pixel κ (defined in Section III-A) is numerated in our model, denoted as SATsp-κ. If the number of pixels in each super-pixel is too large (*i.e.*, SATsp-250), it is hard to extract precise motion coherence among relative large size of parts by (9). On the other hand, if it is too small (*i.e.*, SATsp-60), the large number of parts considerably increases the computation complexity and slightly improves the performance. We observe that the performance is not so sensitive when the number of pixels in each super-pixel is small (*e.g.*, $\kappa \leq 100$). Considering the tradeoff of performance and running speed, we set $\kappa = 100$ in our algorithm.

4) Generative Term: Different from our previous work [25], this work improves the candidate parts generation step by incorporating a generative HSV histogram using the MRF based segmentation method. As shown in Fig. 7(c), we consider different balance factor of generative term λ_1 in (2), *i.e.*, SATgt-λ_1. If it is underrated or overrated, the generated candidate parts will miss true target parts or introduce more background parts, both negatively affect the performance. On the contrary, appropriate values (*e.g.*, SATgt-0.9 and SATgt-1.0) lead to better performance so that we set $\lambda_1 = 0.9$ in the model.

5) Motion Weight: The balance factor λ_5 represents the importance of motion coherence considered in our model. Here we consider different values of λ_5, *i.e.*, SATmw-λ_5. As shown in Fig. 7(d), SATmw-2.0 gives the best performance. It indicates that too small or too large balance factor will degrade the tracking results. Consequently our reported experimental results are obtained with the empirically optimal value $\lambda_5 = 2.0$. Besides, we report the performance of baseline tracker SATaw-0.0 using only the motion weight (*i.e.*, without appearance weight), which is worse than SATmw-2.0. These results show that motion weight plays a more important role in our model for tracking performance than appearance weight.

6) Color Space: Multiple color spaces (*e.g.*, HSV, RGB) are employed in the proposed method. We compare different combinations of them, including unary and binary terms using HSV and RGB color spaces, denoted as SAThsv-rgb, SAThsv-hsv, SATrgb-rgb, and SATrgb-hsv.[6] SAThsv-rgb, *i.e.*, unary term using HSV and binary term using RGB, gives the best performance among all the combinations in Fig. 7(e). Therefore, we employ this combination in our method.

7) Optical Flow: The result in Fig. 7(e) indicates that the tracker using only the color binary term (*i.e.*, without optical flow), SATno-flow, performs worse than the one considering optical flow information, SAThsv-rgb. It demonstrates experimentally that optical flow facilitates to exploit motion coherence of local parts, leading to better tracking performance.

V. Conclusion

In this paper, we describe a structure-aware hyper-graph based tracker. Our method formulates the tracking task as the dense subgraph searching problem on the dynamically constructed hyper-graph integrating the higher-order structural dependencies in temporal domain. The optimal target state is determined by extracting dense subgraphs using a coarse-to-fine strategy. We demonstrate the effectiveness of our method and compare its performance with that of state-of-the-art online tracking methods on the *Deform-SOT* dataset.

There are a few directions we would like to further extend the current work. First, in the current method, we only consider temporal higher-order dependencies among the parts. As a next step, we will also investigate incorporating *spatial* higher-order dependencies among the parts. Considering both spatial and temporal dependencies among parts is expected to further improve tracking performance and robustness under deformation and occlusion. Another important issue is to develop a more efficient algorithm to search dense subgraphs over a large number of candidate parts with the higher-order hyper-graph ($m > 3$). Good sampling scheme to select more reliable and important hyper-edges from candidate parts can simplify the construction and solution of the hyper-graph.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, Nov. 2012.

[2] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2006, pp. 798–805.

[3] O. Arandjelović, "Contextually learnt detection of unusual motion-based behaviour in crowded public spaces," in *Proc. Int. Symp. Comput. Inf. Sci.*, 2011, pp. 403–410.

[4] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1619–1632, Aug. 2011.

[5] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, Sep. 2004.

[6] G. J. Brostow and R. Cipolla, "Unsupervised Bayesian detection of independent motion in crowds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2006, pp. 594–601.

[6]The appearance feature employs the same color representation as the unary term.

[7] T. Brox and J. Malik, "Large displacement optical flow: Descriptor matching in variational motion estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 500–513, Mar. 2011.

[8] Z. Cai, L. Wen, Z. Lei, N. Vasconcelos, and S. Z. Li, "Robust deformable and occluded object tracking with dynamic graph," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5497–5509, Dec. 2014.

[9] L. Cehovin, M. Kristan, and A. Leonardis, "Robust visual tracking using an adaptive coupled-layer visual model," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 4, pp. 941–953, Apr. 2013.

[10] J. Chang and J. W. Fisher, III, "Topology-constrained layered tracking with latent flow," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 161–168.

[11] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jan. 2014, pp. 1090–1097.

[12] T. B. Dinh, N. Vo, and G. Medioni, "Context tracker: Exploring supporters and distracters in unconstrained environments," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 1177–1184.

[13] S. Duffner and C. Garcia, "PixelTrack: A fast adaptive algorithm for tracking non-rigid objects," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2480–2487.

[14] M. Felsberg *et al.*, "The thermal infrared visual object tracking VOT-TIR2015 challenge results," in *Proc. Workshops Conjunct Int. Conf. Comput. Vis.*, Dec. 2015, pp. 639–651.

[15] M. Godec, P. M. Roth, and H. Bischof, "Hough-based tracking of non-rigid objects," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 81–88.

[16] H. Grabner, J. Matas, L. Van Gool, and P. Cattin, "Tracking the invisible: Learning where the object might be," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1285–1292.

[17] M. Grundmann, V. Kwatra, M. Han, and I. Essa, "Efficient hierarchical graph-based video segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2141–2148.

[18] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 263–270.

[19] Z. Hong, C. Wang, X. Mei, D. Prokhorov, and D. Tao, "Tracking using multilevel quantizations," in *Proc. Eur. Conf. Comput. Vis.*, vol. 8694. 2014, pp. 155–171.

[20] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1822–1829.

[21] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-N learning: Bootstrapping binary classifiers by structural constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 49–56.

[22] D. A. Klein and A. B. Cremers, "Boosting scalable gradient features for adaptive real-time tracking," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2011, pp. 4411–4416.

[23] 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. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1208–1215.

[24] A. Li, M. Li, Y. Wu, M.-H. Yang, and S. Yan, "NUS-PRO: A new visual tracking challenge," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 335–349, Feb. 2016.

[25] W. Li, L. Wen, M. C. Chuah, Y. Zhang, Z. Lei, and S. Z. Li, "Online visual tracking using temporally coherent part cluster," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Jan. 2015, pp. 9–16.

[26] J. Lim, D. A. Ross, R.-S. Lin, and M.-H. Yang, "Incremental learning for visual tracking," in *Proc. Adv. Neural Inf. Process. Syst.*, 2004, pp. 793–800.

[27] H. Liu, L. J. Latecki, and S. Yan, "Robust clustering as ensembles of affinity relations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1414–1422.

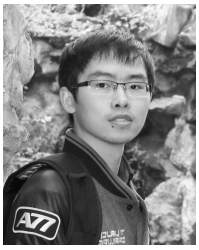
[28] R. Martin and O. Arandjelović, "Multiple-object tracking in cluttered and crowded public spaces," in *Proc. Int. Symp. Vis. Comput.*, 2010, pp. 89–98.

[29] X. Mei and H. Ling, "Robust visual tracking using ℓ_1 minimization," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 1436–1443.

[30] S. M. S. Nejhum, J. Ho, and M.-H. Yang, "Online visual tracking with histograms and articulating blocks," *Comput. Vis. Image Understand.*, vol. 114, no. 8, pp. 901–914, Aug. 2010.

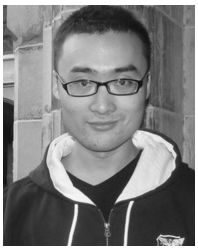
[31] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1940–1947.

[32] Y. Qi *et al.*, "Hedged deep tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, to be published.
[33] X. Ren and J. Malik, "Tracking as repeated figure/ground segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[34] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, "PROST: Parallel robust online simple tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 723–730.
[35] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for SVM," *Math. Program.*, vol. 127, no. 1, pp. 3–30, Mar. 2011.
[36] S. Song and J. Xiao, "Tracking revisited using RGBD camera: Unified benchmark and baselines," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 233–240.
[37] M. Kristan *et al.*, "The visual object tracking VOT2014 challenge results," in *Proc. Workshops Conjunct Eur. Conf. Comput. Vis.*, 2014, pp. 191–217.
[38] S. Wang, H. Lu, F. Yang, and M.-H. Yang, "Superpixel tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1323–1330.
[39] L. Wen, Z. Cai, D. Du, Z. Lei, and S. Z. Li, "Learning discriminative hidden structural parts for visual tracking," in *Proc. Workshops Conjunct Asian Conf. Comput. Vis.*, 2014, pp. 262–276.
[40] L. Wen, Z. Cai, Z. Lei, D. Yi, and S. Z. Li, "Online spatio-temporal structural context learning for visual tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 716–729.
[41] L. Wen, Z. Cai, Z. Lei, D. Yi, and S. Z. Li, "Robust online learned spatio-temporal context model for visual tracking," *IEEE Trans. Image Process.*, vol. 23, no. 2, pp. 785–796, Feb. 2014.
[42] L. Wen, Z. Cai, M. Yang, Z. Lei, D. Yi, and S. Z. Li, "Online multiple instance joint model for visual tracking," in *Proc. IEEE Int. Conf. Adv. Video Signal-Based Surveill.*, Sep. 2012, pp. 319–324.
[43] L. Wen, D. Du, Z. Lei, S. Z. Li, and M.-H. Yang, "JOTS: Joint online tracking and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 2226–2234.
[44] L. Wen, W. Li, J. Yan, Z. Lei, D. Yi, and S. Z. Li, "Multiple target tracking based on undirected hierarchical relation hypergraph," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1282–1289.
[45] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.
[46] M. Yang, Y. Wu, and G. Hua, "Context-aware visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1195–1209, Jul. 2009.
[47] R. Yao, Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel, "Part-based visual tracking with online latent structural learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2363–2370.
[48] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang, "Fast visual tracking via dense spatio-temporal context learning," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 127–141.
[49] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 864–877.
[50] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 2042–2049.
[51] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1838–1845.



Dawei Du received the B.Eng. degree in automation and the M.S. degree in detection technology and automatic engineering from the University of Electronic Science and Technology of China, in 2010 and 2013, respectively. He is currently pursuing the Ph.D. degree with the School of Computer and Control Engineering, University of Chinese Academy of Sciences. His research interests include graph theory, visual tracking, and video segmentation.

Honggang Qi received the M.S. degree in computer science from Northeast University, Shenyang, China, in 2002, and the Ph.D. degree in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, in 2008. He is currently an Associate Professor with the School of Computer and Control Engineering, University of Chinese Academy of Sciences. His research interests include video coding and VLSI design.



Wenbo Li received the B.Eng. degree from Tianjin University, China, in 2014. He is currently pursuing the Ph.D. degree with the Department of Computer Science and Engineering, Lehigh University. His research interests are computer vision, pattern recognition, multimedia analysis, and machine learning.

Longyin Wen received the B.Eng. degree in automation from the University of Electronic Science and Technology of China, in 2010, and the Ph.D. degree from the Institute of Automation, Chinese Academy of Sciences, in 2015. He is currently with University at Albany, State University of New York, for a post-doctoral research. His research interests are computer vision, pattern recognition, and object tracking in particular.

Qingming Huang is currently a Professor and the Deputy Dean with the School of Computer and Control Engineering, University of Chinese Academy of Sciences. He has authored over 300 academic papers in international journals, such as the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE Transactions on CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, and top level international conferences, including the ACM Multimedia, ICCV, CVPR, ECCV, VLDB, and IJCAI. His research interests include multimedia computing, image/video processing, pattern recognition, and computer vision.

Siwei Lyu received the B.S. degree in information science and the M.S. degree in computer science from Beijing University, in 1997 and 2000, respectively, and the Ph.D. degree in computer science from Dartmouth College, in 2005. He was a Post-Doctoral Research Associate with the Center for Neural Science, Howard Hughes Medical Institute, New York University, and an Assistant Researcher with Microsoft Research Asia. He is currently an Associate Professor with the Computer Science Department, University at Albany, State University of New York, and a Visiting Professor with the School of Computer and Information, Tianjin Normal University, Tianjin, China. He has authored one book, one book chapter, and over 70 refereed technical papers. His main research interests include digital image forensics, computer vision, and machine learning. He is a recipient of the IEEE Signal Processing Society Best Paper Award in 2010, and the NSF CAREER Award in 2010. He is a recipient of the IEEE Signal Processing Society Best Paper Award in 2011, and the U.S. NSF CAREER Award in 2010.