Online Graph-Based Tracking

Hyeonseob Nam, Seunghoon Hong, and Bohyung Han

Department of Computer Science and Engineering, POSTECH, Korea
{namhs09,maga33,bhhan}@postech.ac.kr

Abstract. Tracking by sequential Bayesian filtering relies on a graphical model with temporally ordered linear structure based on temporal smoothness assumption. This framework is convenient to propagate the posterior through the first-order Markov chain. However, density propagation from a single immediately preceding frame may be unreliable especially in challenging situations such as abrupt appearance changes, fast motion, occlusion, and so on. We propose a visual tracking algorithm based on more general graphical models, where multiple previous frames contribute to computing the posterior in the current frame and edges between frames are created upon inter-frame trackability. Such data-driven graphical model reflects sequence structures as well as target characteristics, and is more desirable to implement a robust tracking algorithm. The proposed tracking algorithm runs online and achieves outstanding performance with respect to the state-of-the-art trackers. We illustrate quantitative and qualitative performance of our algorithm in all the sequences in tracking benchmark and other challenging videos.

Keywords: Online tracking, Bayesian model averaging, patch matching.

1 Introduction

Most of online probabilistic tracking algorithms employ graphical models with linear structure and estimate the target state sequentially, where the inference of the posterior is based only on the immediately preceding frame due to the first-order Markov assumption. These methods reduce search space for observation by relying on temporal smoothness assumption between two consecutive frames. However, they underestimate other kinds of challenges—for example, radical appearance changes, fast motion, shot changes, and occlusion—and the potential benefit from the collaboration of multiple frames. Therefore, we claim that tracking algorithms should consider the characteristics of target and scene in addition to temporal smoothness and that more general graphical models can reduce tracking errors by propagating densities from multiple tracked frames.

We propose a novel online tracking algorithm beyond the first-order Markov chain, where a more general graph structure is obtained during tracking to adapt sequence structure and propagate the posterior over time. Multiple preceding frames propagate density functions to estimate the optimal target state in the current frame, and the choice of the frames depends on the characteristics of

D. Fleet et al. (Eds.): ECCV 2014, Part V, LNCS 8693, pp. 112–126, 2014.

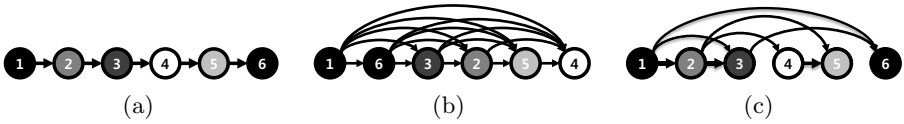


Fig. 1. Framework of our algorithm compared to existing methods. Nodes correspond to frames whose intensities encode the characteristics observed in individual frames. The numbers in the nodes indicate frame indices. (a) Chain model propagates densities sequentially regardless of the characteristics of frames. (b) Bayesian model averaging [1] tracks target in an increasing order of variations and makes a blind average of the posteriors from all previously tracked frames. (c) Our algorithm measures the tracking feasibility between frames, and the density functions are propagated from only relevant frames through a more flexible graphical model compared to (a) and (b).

target and scene. In other words, our framework learns a general directed graphical model actively and tracks a target under the identified graphical model. The proposed framework has some common properties with [1], which also employs a more complex graphical model and estimates the posterior using Bayesian model averaging, but has the following critical differences: 1) our algorithm runs online while [1] is an offline technique, 2) we actively identify appropriate frames for density propagation instead of blind model averaging, and 3) the proposed algorithm is more efficient due to the reduction of the number of posterior propagations. The main concept of our framework is illustrated in Figure 1; instead of using all the tracked frame for density propagation as shown in Figure 1(b), we adaptively determine appropriate frames to improve tracking performance and reduce computational complexity as shown in Figure 1(c). Note that we can avoid further density propagation from the frames with very different characteristics and isolate tracking errors in such frames naturally.

Given a graphical model constructed with tracked frames, we identify a set of nodes in the graph from which the new frame is connected based on tracking plausibility so that we obtain the updated graphical model. Finding the appropriate nodes to track from in the current graphical model is computationally expensive, so we maintain a small subset of representative frames to facilitate the procedure. Once the new graphical model is obtained, we propagate the posterior density to the new frame by an efficient patch matching technique [2]. Our approach has something common with the methods based on multiple target templates [3,4,5] but is clearly different from them because we propagate the posteriors to the current frame from multiple parents using the identified graphical model; it is more advantageous to preserve multi-modality in the posterior. Now, we have an online tracking algorithm that learns a graph structure and solves target tracking jointly. The main contributions of our tracking algorithm are summarized below:

- We propose an adaptive and active algorithm to identify a general graphical model for tracking based on the sequence structure characterized by target and scene.

- Our tracking algorithm estimates the posterior through a selective aggregation of propagated densities, which overcomes the limitation of blind density averaging and isolates tracking errors within local branches in graph.
- The proposed tracking algorithm runs online and improves tracking performance significantly by handling various challenges effectively.

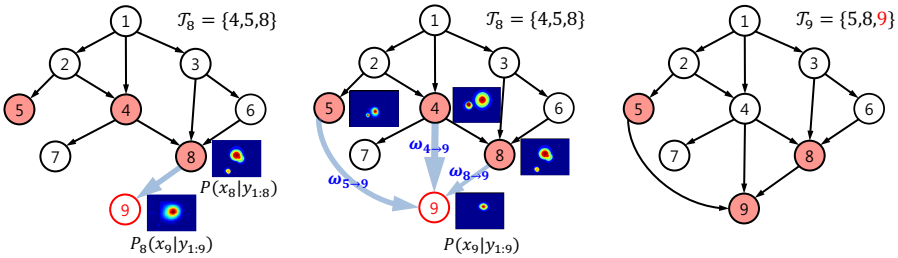
This paper is organized as follows. We first review related work in Section 2, and describe the overall framework of our algorithm in Section 3. The detailed formulation and methodology are described in Section 4 and Section 5 illustrates experimental results.

2 Related Work

Visual tracking algorithms are composed of tracking control and observation, and each algorithm is typically constructed by a combination of two components. There are several options in tracking control, which include local optimization methods, sampling-based approaches and tracking-by-detection framework. Local optimization methods [6,7,8] are simple and easy to implement, but may be stuck at local optima. To avoid this limitation, many tracking algorithms employ sampling based density propagation techniques, which are based on either sequential Bayesian filtering [9,3,4,10,5] or MCMC [11]. Recently, thanks to the advance of object detection technique, tracking-by-detection approaches are used widely [12,13,14], which can also be regarded as dense sampling method.

All the tracking algorithms listed above depend on linear graphical model or sequential processing based on the first-order Markov assumption. They focus on the density propagation or the optimal search problem between two temporally adjacent frames. This framework is useful to exploit temporal smoothness between frames but has critical limitations in handling the challenges violating the property, *i.e.*, significant appearance changes, shot changes, and occlusion. To ameliorate these problems, [11] proposes an online tracking algorithm to propagate the posterior by MCMC sampling, [15] utilizes high-order Markov chain, and [16] models occlusion explicitly using a SVM-based classifier. However, note that these efforts are still limited to the optimization of target state given the information of target and tracker state in the a single or at most a few preceding frame(s). Recently, [1] proposes an offline algorithm to actively search a suitable order of frames for tracking, where the posterior of a new frame is estimated by propagating posteriors from all tracked frames and aggregating them through Bayesian model averaging. Although these methods do not rely on chain models any more, the graphical model for tracking is fixed and are not adaptive to the characteristics of the input video. Also, note that most of offline tracking algorithms still depend on linear graphical model [17,18,19,20,21].

Another main challenges in visual tracking is how to maintain the appearance of target in a robust manner. Many tracking algorithms have been investigating this problem and some promising solutions have been proposed such as template update [22,23,6], sparse representation [3,4,5], incremental subspace learning [9], multi-task learning [10], multiple instance learning [13], P-N learning [12], and



(a) Density propagation (b) Density aggregation (c) Model update

Fig. 2. Procedure of the proposed algorithm performed in a frame. (a) When a new frame (red hollow node) is given, each representative frame (shaded in pink) propagates the posterior to the new frame, creating a corresponding edge in the existing graphical model. (b) Propagated posteriors from all representative frames are weighted and aggregated to compute the final target posterior in the new frame. (c) The target state is estimated and the list of representative frames is updated if necessary.

so on. Although robust appearance models enable tracker to handle various appearance changes effectively, fundamental limitation of sequential approaches by linear graphical models—its weakness to temporal failures and multi-modal variations—still remain. This is partly because tracking control and observation are investigated separately even though joint optimization of the two problems is potentially helpful to improve overall performance.

Contrary to prior studies, our approach couples the two components more tightly. We employ preliminary observation to determine the structure of graphical model, and the adaptively identified graphical model facilitates robust observation. This procedure performs online, and we implement an online tracking algorithm based on the adaptively constructed graph structure.

3 Algorithm Overview

The main goal of this work is to progressively construct a graphical model that is appropriate for tracking but is not necessarily limited to chain models, and sequentially estimate the posterior of target state $\mathbf{x}_t$ at the t^{th} frame given observation $\mathbf{y}_{1:t}$. When a new frame t arrives, our algorithm selectively propagates density functions to estimate the posterior $P(\mathbf{x}_t|\mathbf{y}_{1:t})$. To reduce the computational cost, we maintain the m most representative frames within the graph, $\mathcal{T}_{t-1} = \{t^{(1)}, ..., t^{(m)}\}$ $(m \ll t-1)$, in an online manner and allow only the frames in $\mathcal{T}_{t-1}$ to propagate densities to frame t with relevant weights. Our algorithm performs the following procedures to track the target and update the graphical model at each frame:

1. **Density propagation step** propagates density functions from $P(\mathbf{x}_u|\mathbf{y}_{1:u})$ $\forall u \in \mathcal{T}_{t-1}$ to the frame t through a patch matching technique [2], and creates an edge from each frame in $\mathcal{T}_{t-1}$ to the frame t.

2. **Density aggregation step** estimates the target posterior $P(\mathbf{x}_t|\mathbf{y}_{1:t})$ by a weighted Bayesian model averaging, where the weight of each edge is computed based on its tracking plausibility.
3. **Model update step** first evaluates the reliability of the tracking result and updates the model if necessary. Specifically, if the tracking result at frame t is also more reliable and distinctive than the frames in $\mathcal{T}_{t-1}$, the new representative frame set $\mathcal{T}_t$ is obtained by adding frame t to $\mathcal{T}_{t-1}$ and removing the least appropriate one from $\mathcal{T}_{t-1}$. Otherwise, we set $\mathcal{T}_t = \mathcal{T}_{t-1}$.

These three steps are illustrated in Figure 2, and we discuss the detailed procedure of each step in the next section.

4 Main Algorithm

This section describes our main algorithm, which includes progressive graph construction technique and density propagation through weighted Bayesian model averaging [24]. We first present how the target posterior is estimated by the weighted Bayesian model averaging based on patch matching [2]. Then we discuss how to construct a general graphical model in a progressive fashion and how to maintain target models for persistent tracking.

4.1 Density Propagation by Patch Matching

In the sequential Bayesian filtering framework, the density function is propagated recursively and the posterior is estimated through prediction and update steps. In our scenario, density propagation does not necessarily happen between two temporally adjacent frames but can be performed via any frames tracked previously. The propagated density at frame t from frame u denoted by $P_u(\mathbf{x}_t|\mathbf{y}_{1:t})$ is defined as

$$P_u(\mathbf{x}_t|\mathbf{y}_{1:t}) = \alpha_{u\to t} P_u(\mathbf{y}_t|\mathbf{x}_t) \int P(\mathbf{x}_t|\mathbf{x}_u) P(\mathbf{x}_u|\mathbf{y}_{1:u}) d\mathbf{x}_u, \tag{1}$$

where $P(\mathbf{x}_t|\mathbf{x}_u)$ is the transition model from frame u to frame t, $P_u(\mathbf{y}_t|\mathbf{x}_t)$ is likelihood at frame t with respect to frame u, and $\alpha_{u\to t}$ is a normalization constant.

The recursive posterior estimation is implemented through patch matching with sampling [1], where the prediction and update steps in each Bayesian filter are handled jointly. Each patch inside a candidate bounding box defined by a sample in frame u are matched with a certain patch in frame t, and the patch votes for the target center using a Gaussian kernel. The voting map for each sample is obtained by aggregating the votes from all patches in the bounding box, and the further aggregation of the voting maps constructs the posterior of frame t. Mathematically, the posterior is approximated as

$$P_u(\mathbf{x}_t|\mathbf{y}_{1:t}) \approx \sum_{\mathbf{x}_u^i \in \mathbb{S}_u} P_u(\mathbf{y}_t|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{x}_u^i)$$

$$= \sum_{\mathbf{x}_u^i \in \mathbb{S}_u} \sum_{j=1}^{r_u^i} \mathcal{N}(\mathbf{x}_t; f_{u \to t}(\mathbf{c}_i^j) - \mathbf{a}_i^j, \mathbf{\Sigma}), \tag{2}$$

where $\mathbb{S}_u$ denotes a set of samples drawn from $P(\mathbf{x}_u|\mathbf{y}_{1:u})$, r_u^i is the number of patches within the bounding box defined by each sample $\mathbf{x}_u^i$. The patch match function $f_{u \to t}(\mathbf{c}_i^j)$ finds correspondence of the patch centered at $\mathbf{c}_i^j$, and $\mathbf{a}_i^j$ is the offset from $\mathbf{x}_u^i$ to $\mathbf{c}_i^j$. Each voting is smoothed using Gaussian kernel $\mathcal{N}(\cdot)$ with a variance $\mathbf{\Sigma}$. We maintain multiple posteriors in several different scales to handle the variation in size of the target. Note that each propagation creates a directed edge between two corresponding frames in the graphical model.

4.2 Density Aggregation by Weighted Bayesian Model Averaging

Our tracking algorithm employs a weighted Bayesian model averaging to propagate the posterior density functions similar to [1]. Since we do not rely on the first-order Markov chain any more, there are a number of options to propagate the posterior to the current frame from all the previous frames.

Let $\mathcal{T}_{t-1} = \{t^{(1)}, \ldots, t^{(m)}\}$, where t is the index of the current frame and $t^{(1)}, \ldots, t^{(m)} \leq t-1$, be a set of representative frame indices that have outgoing edges to the t^{th} frame in the graphical model. In other words, only the frames corresponding to the elements in $\mathcal{T}_{t-1}$ among all the tracked frames propagate densities to the t^{th} frame. Then, the posterior at the current frame $P(\mathbf{x}_t|\mathbf{y}_{1:t})$ is estimated by a weighted sum of the propagated densities from $u \in \mathcal{T}_{t-1}$ denoted by $P_u(\mathbf{x}_t|\mathbf{y}_{1:t})$ as illustrated in Figure 2, which is formally given by

$$P(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto \sum_{u \in \mathcal{T}_{t-1}} \omega_{u \to t} P_u(\mathbf{x}_t|\mathbf{y}_{1:t})$$

$$= \sum_{u \in \mathcal{T}_{t-1}} \omega_{u \to t} P_u(\mathbf{y}_t|\mathbf{x}_t) \int P(\mathbf{x}_t|\mathbf{x}_u) P(\mathbf{x}_u|\mathbf{y}_{1:u}) d\mathbf{x}_u, \tag{3}$$

where $\omega_{u \to t}$ is the weight for each posterior. This formulation is similar to the one proposed in [1] and the detailed derivation is omitted due to space limitation. By integrating patch matching process in Eq. (2), the posterior at frame t is estimated approximately by the following equation:

$$P(\mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{u \in \mathcal{T}_{t-1}} \omega_{u \to t} \sum_{\mathbf{x}_u^i \in \mathbb{S}_u} \sum_{j=1}^{r_u^i} \mathcal{N}(\mathbf{x}_t; f_{u \to t}(\mathbf{c}_u^j) - \mathbf{a}_u^j, \mathbf{\Sigma}). \tag{4}$$

Propagating density from one frame to another means that there exists a directed edge between the two frames in the graphical model, where the weight for the edge is $\omega_{u \to t}$, where u is a parent frame of t. All nodes are supposed to have

$|\mathcal{T}_{t-1}|$ incoming edges, where $|\cdot|$ denotes the number of elements in a set. The remaining issue is how to determine the weight factor, $\omega_{u\to t}$.

The weight factor $\omega_{u\to t}$ is determined by the suitability of tracking along the edge from frame u to t, considering the path from the initial frame to frame u in the graph structure. For the purpose, we define a measure to estimate the potential risk resulting from tracking between frame u and t, which is given by

$$\delta_{u\to t} = \max\left(\delta_u, d_c(u,t)\right), \tag{5}$$

where $d_c(u,t)$ represents the estimated tracking error between two directly connected frames u and t, and δ_u represents the accumulated tracking error up to frame u. $d_c(u,t)$ measures the dissimilarity between target appearances in frame u and t, where the target at frame t is tentatively obtained based on the propagated posterior from frame u as

$$\mathbf{x}^*_{u\to t} = \arg\max_{\mathbf{x}_t} P_u(\mathbf{x}_t|\mathbf{y}_{1:t}). \tag{6}$$

Given the target templates τ_u and τ_t, which are obtained from the bounding boxes corresponding to $\mathbf{x}^*_u$ and $\mathbf{x}^*_{u\to t}$, respectively, we compute the deformation cost between τ_u and τ_t as follows:

$$d_c(u,t) \equiv \underset{j}{\text{median}}\left(||\mathbf{c}^j_u - f_{u\to t}(\mathbf{c}^j_u;\tau_t)||\right), \tag{7}$$

where $f_{u\to t}(\mathbf{c}^j_i;\tau_t)$ is a patch matching function from the j^{th} patch centered at $\mathbf{c}^j_u$ inside template τ_u at frame u to template τ_t at frame t. The accumulated tracking error δ_u is obtained by the maximum tracking error in the minimax path from the initial frame to frame u, which is formally given by

$$\delta_u = \min_{v\in\mathcal{T}_{u-1}} \delta_{v\to u} = \min_{v\in\mathcal{T}_{u-1}} \left(\max\left(\delta_v, d_c(v,u)\right)\right), \tag{8}$$

where v denote the parent frames of u in the graph. Note that δ_u is computed when tracking at frame u is completed and hence given at frame t.

Based on tracking error $\delta_{u\to t}$, defined in Eq. (5), the normalized weight for each outgoing edge $\omega_{u\to t}, \forall u \in \mathcal{T}_{t-1}$ is given by,

$$\omega_{u\to t} = \frac{\exp(-\eta\cdot\delta_{u\to t})}{\sum_{s\in\mathcal{T}_{t-1}}\exp(-\eta\cdot\delta_{s\to t})}, \tag{9}$$

where η is a scale factor and set to $\eta = (\min_u \delta_{u\to t})^{-1}$. In this way, each propagated density $P_u(\mathbf{x}_t|\mathbf{y}_{1:t})$ along each directed edge from $\mathcal{T}_{t-1}$ is aggregated to obtain the posterior $P(\mathbf{x}_t|\mathbf{y}_{1:t})$ at frame t.

4.3 Model Update

When the density propagation and aggregation steps are completed, the optimal target state is given by the MAP solution as

$$\mathbf{x}^*_t = \arg\max_{\mathbf{x}_t} P(\mathbf{x}_t|\mathbf{y}_{1:t}). \tag{10}$$

After obtaining the tracking result $\mathbf{x}_t^*$ in a new frame and augmenting the graphical model including the new frame t, we update the list of representative frames, $\mathcal{T}_{t-1}$. Note that maintaining an appropriate set of frames $\mathcal{T}_t$ is important to track subsequent frames and avoid drift problem.

To achieve this goal, we apply an online classifier based on k-nearest neighbors and decide whether the tracking result corresponds to target object or background. If it turns out to be target, we update the representative frame set based on the predefined measure. We discuss these two issues next.

Template Classification. Suppose that we have collected a set of positive and negative templates, corresponding to target and background respectively, during tracking until frame $t-1$. The set is denoted by $\mathcal{D}_{t-1} = \{\boldsymbol{\tau}_1^+, ..., \boldsymbol{\tau}_{N_p}^+, \boldsymbol{\tau}_1^-, ..., \boldsymbol{\tau}_{N_n}^-\}$, where $\boldsymbol{\tau}^+$ and $\boldsymbol{\tau}^-$ represent positive and negative templates, respectively, and N_p and N_n denote the numbers of positive and negative examples, respectively. To classify the obtained target template $\boldsymbol{\tau}_t^*$ corresponding to $\mathbf{x}_t^*$, we use the following measure

$$S = \frac{S_p}{S_p + S_n} \tag{11}$$

where S_p is an average distance between $\boldsymbol{\tau}_t^*$ and k-nearest positive templates in $\mathcal{D}_{t-1}$ and S_n is the distance between $\boldsymbol{\tau}_t^*$ and the nearest negative template. The distance measure used to compute S_p and S_n is the Sum of Squared Distance (SSD)[1]. The estimated template $\boldsymbol{\tau}_t^*$ is determined as the target object if $S < \rho$, where ρ is a classifier threshold and typically set to 0.5. Note that S_n considers only a single nearest negative template while S_p considers k-nearest positive templates, which is useful to make the classifier robust to false positives. One may argue that this strategy may not be appropriate to handle radical target appearance changes, but there is a trade-off between avoiding drift problem and adapting new appearances; we found that the conservative method works better in practice.

Once $\boldsymbol{\tau}_t^*$ is classified as a target object, we construct $\mathcal{D}_t$ from $\mathcal{D}_{t-1}$ by replacing old templates in $\mathcal{D}_{t-1}$ with new positive and negative templates. The positive template is obtained from our MAP solution $\mathbf{x}_t^*$, and the negative templates are generated considering background context and distracting regions. The background context is captured by sampling background templates around the identified target $\mathbf{x}_t^*$. The distractors are regions that have similar appearance as target, therefore sampled from modes in $P(\mathbf{x}_t|\mathbf{y}_{1:t})$ except $\mathbf{x}_t^*$. Note that the number of elements in $\mathcal{D}$ remains same in each frame.

Maintaining Representative Frames. Maintaining a small subset of frames propagating densities, $\mathcal{T}_t$, is crucial to achieve good tracking performance with efficiency. We consider the following two properties to maintain a reasonable set of representative frames $\mathcal{T}_t$:

[1] We used SSD instead of patch matching here because SSD is more efficient and patch matching is not particularly better than SSD for this purpose.

- **Distinctiveness**: Frames in $\mathcal{T}_t$ should be unique in the set to effectively cover various aspects of target in the entire graph.
- **Usefulness**: Frames in $\mathcal{T}_t$ should potentially contribute to tracking subsequent frames.

We now discuss how we update $\mathcal{T}_t$ online by taking the both properties into account. Let $\kappa_{u,t}$ denote the weight of frame $u \in \mathcal{T}_t$ in terms of representativeness for further tracking. The weight depends on two factors, $\kappa_u^{(1)}$ and $\kappa_{u,t}^{(2)}$, which correspond to distinctiveness and usefulness, respectively.

To make the elements in $\mathcal{T}_t$ distinctive, frames with redundant target appearances need to be removed from the set. The weight for frame distinctiveness is computed by

$$\kappa_u^{(1)} = \min_{v \in \mathcal{T}_{t-1} \setminus \{u\}} \Delta(u, v), \tag{12}$$

which corresponds to the distance between the template in frame u and the most similar target template within other frames in $\mathcal{T}_{t-1}$. Specifically, $\Delta(u, v)$ is determined by the two factors as

$$\Delta(u, v) = d_c(u, v) \cdot d_p(u, v), \tag{13}$$

where d_c measures the degree of the target deformation, as defined in Eq. (7), and d_p measures the dissimilarity of target appearances based on average ℓ_2 distances between all matched patches within the target templates.

On the other hand, we claim that the frames having recently propagated densities with large weights are more useful to track subsequent frames, and such potential of frames are measured by

$$\kappa_{u,t}^{(2)} = (1 + \sigma y_t \omega_{u \to t}) \kappa_{u,t-1}^{(2)}, \tag{14}$$

where $\omega_{u \to t}$ is weight for density propagation from frame $u \in \mathcal{T}_{t-1}$ to frame t as defined in Eq. (9), $y_t \in \{+1, -1\}$ indicates whether tracking result at frame t is classified as foreground $(+1)$ or background (-1) in the template classification step, and σ controls update rate set to 0.1.

By combining the weights for the two different aspects, the weight for each frame $u \in \mathcal{T}_t$ is given by

$$\kappa_{u,t} = \kappa_u^{(1)} \cdot \kappa_{u,t}^{(2)}. \tag{15}$$

The weight for the new frame t is computed by the same manner, except that $\kappa_{t,t-1}^{(2)}$ is computed by the median of $\kappa_{u,t-1}^{(2)}, \forall u \in \mathcal{T}_{t-1}$. Given these weights for frames in $\mathcal{T}_{t-1}$ and t, we update the $\mathcal{T}_t$ as follows:

$$\mathcal{T}_t = \begin{cases} (\mathcal{T}_{t-1} \setminus \{m\}) \cup \{t\}, & \text{if } \kappa_{t,t} > \kappa_{m,t} \\ \mathcal{T}_{t-1}, & \text{otherwise} \end{cases} \tag{16}$$

where

$$m = \operatorname*{argmin}_{u \in \mathcal{T}_{t-1}} \kappa_{u,t}.$$

After $\mathcal{T}_t$ is obtained by Eq. (16), the weights $\kappa_{i,t}, \forall i \in \mathcal{T}_t$ are re-normalized such that all weights are summed up to one.

5 Experiment

Our tracking algorithm is tested in a variety of challenging sequences and compared with many state-of-the-art online tracking algorithms included in the tracking benchmark [25]. We present implementation details of the proposed algorithm and extensive experimental results.

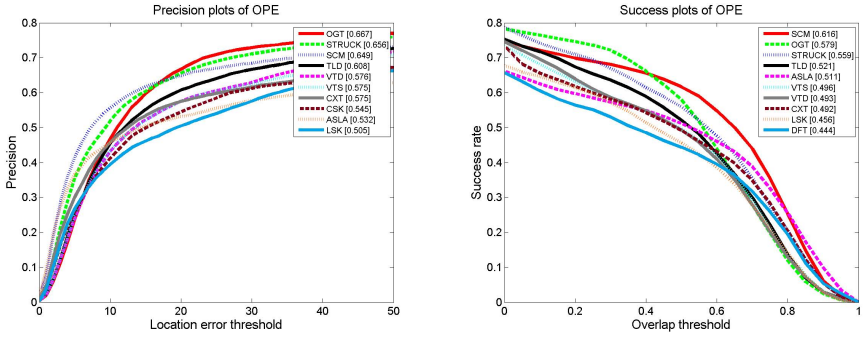
5.1 Datasets and Compared Algorithms

To evaluate the performance of our tracking algorithm in various scenarios, we conducted experiments on all the 50 sequences in the tracking benchmark dataset [25] and added 10 more sequences publicly available, which are more challenging and difficult to be handled by online trackers. The sequences from the benchmark dataset contain various challenges such as illumination variation, background clutter, occlusion, etc. More challenges are included in the 10 additional sequences: heavy occlusion (*TUD, campus, accident*), abrupt target motion (*bike, tennis*) and shot changes (*boxing, youngki, skating, dance, psy*).

We compared our algorithm with top 10 trackers by one-pass evaluation (OPE) in the tracking benchmark [25], which include SCM [5], Struck [14], TLD [12], ASLA [26], CXT [27], VTD [28], VTS [29], CSK [30], LSK [7] and DFT [31]. In addition, the state-of-the-art offline tracking algorithm OMA [1] is also included in our evaluation. We used default parameters for all compared methods. Our method is denoted by OGT (Online Graph-based Tracking).

5.2 Implementation and Performance

Our algorithm employs patch matching technique across multiple scales for density propagation. Specifically, 4×4 patches are used for patch matching and 13



(a) Precision (b) Success rate

Fig. 3. Tracking performance in all the 50 sequences in the tracking benchmark dataset. Precision and success ratio are measured by center location errors and bounding box overlap ratios, respectively. The ranks are set with center location error 25 and overlap ratio 0.5.

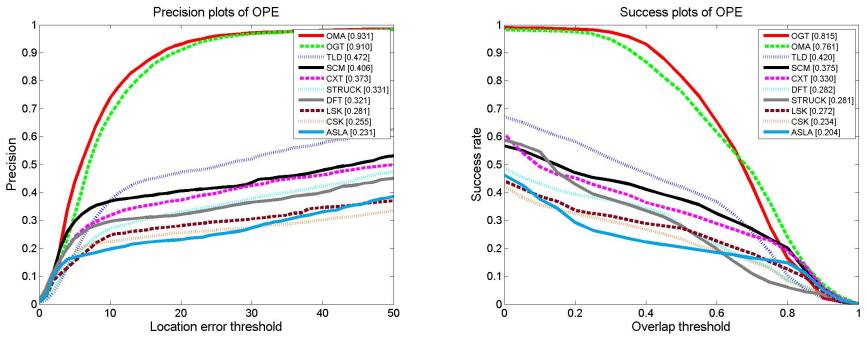


Fig. 4. Tracking performance in the additional 10 sequences with more challenging events. Precision and success ratio are measured by center location errors and bounding box overlap ratios, respectively. The ranks are set with center location error 25 and overlap ratio 0.5.

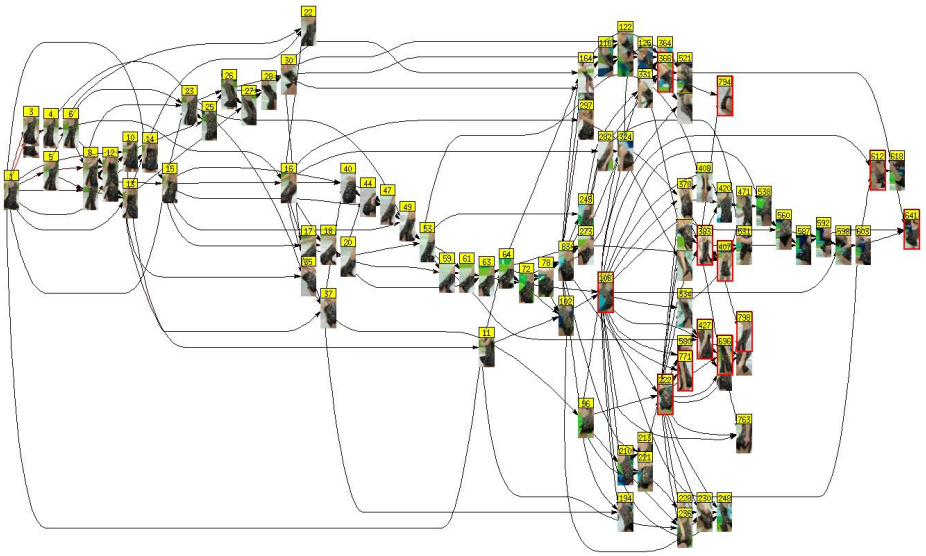


Fig. 5. The example of identified graph structure in *skating* sequence. The whole graph is very complex, and we illustrate only the frames that have been included in $\mathcal{T}$ and the associated edges with high weights. Frames included in the final $\mathcal{T}$ are highlighted with red bounding boxes.

different scales $(1.1^{-6}, 1.1^{-5}, \ldots, 1.1^{0}, \ldots, 1.1^{5}, 1.1^{6})$ are considered for tracking. The number of representative frames in $\mathcal{T}$ set to 12 except the initial part of each sequence[2]. The number of templates in $\mathcal{D}$ for k-nearest neighbor classification is 600 with 300 positive and negative examples, and k is set to 10. All parameters

[2] At the beginning of sequences, the new frame is added to $\mathcal{T}$ without removing elements as long as the estimated target is classified as foreground and $|\mathcal{T}| < 12$.

Fig. 6. Qualitative performance evaluation for 10 additional sequences. From top to bottom, tracking results for *tennis, skating, accident, bike, boxing, campus, TUD, youngki, psy* and *dance* sequences are presented.

Table 1. Average center location error (in pixels). Red: best, blue: second best.

	LSK	ASLA	CXT	DFT	CSK	Struck	SCM	VTD	VTS	TLD	OMA	OGT
tennis	79.3	67.2	129.8	87.1	112.3	109.5	66.0	81.0	58.9	64.5	6.4	4.9
skating	85.9	45.2	41.5	83.3	61.1	23.8	49.0	37.1	42.9	35.3	7.9	6.4
accident	68.4	59.4	9.0	13.6	76.0	56.4	2.8	70.8	70.4	5.4	3.1	6.0
bike	70.4	88.6	22.2	131.1	103.3	8.4	13.6	85.7	82.1	16.9	17.4	7.2
boxing	97.1	137.3	137.3	73.2	108.8	122.7	95.9	82.9	80.4	73.3	10.3	11.1
campus	44.6	12.2	33.4	1.3	2.1	83.1	12.6	45.1	44.8	46.7	2.6	2.5
TUD	21.1	72.6	36.4	8.2	55.4	54.4	12.2	46.5	48.2	18.9	4.2	11.8
youngki	108.9	144.1	67.9	72.6	163.9	115.1	114.5	112.5	116.0	60.2	11.0	11.5
psy	153.9	188.8	143.7	205.3	1022.9	76.3	211.6	129.5	123.6	55.6	14.7	26.2
dance	163.9	117.5	176.8	157.1	147.0	107.1	208.5	188.4	201.1	105.0	14.9	23.7
Average	89.4	93.3	79.8	83.3	185.3	75.7	78.7	88.0	86.8	48.2	9.3	11.1

Table 2. Average bounding box overlap ratio. Red: best, blue: second best.

	LSK	ASLA	CXT	DFT	CSK	Struck	SCM	VTD	VTS	TLD	OMA	OGT
tennis	0.20	0.12	0.08	0.06	0.04	0.28	0.11	0.07	0.09	0.10	0.65	0.77
skating	0.04	0.13	0.25	0.11	0.09	0.40	0.20	0.25	0.21	0.33	0.45	0.54
accident	0.35	0.43	0.80	0.47	0.32	0.32	0.86	0.41	0.41	0.75	0.75	0.66
bike	0.20	0.16	0.39	0.02	0.15	0.54	0.49	0.16	0.17	0.45	0.43	0.57
boxing	0.07	0.03	0.01	0.17	0.05	0.04	0.13	0.14	0.16	0.21	0.70	0.66
campus	0.58	0.63	0.56	0.81	0.81	0.24	0.62	0.35	0.36	0.50	0.81	0.83
TUD	0.62	0.30	0.51	0.60	0.38	0.30	0.68	0.41	0.38	0.67	0.83	0.65
youngki	0.12	0.12	0.38	0.14	0.10	0.09	0.13	0.16	0.14	0.24	0.63	0.64
psy	0.11	0.11	0.10	0.07	0.09	0.34	0.07	0.20	0.25	0.38	0.64	0.56
dance	0.12	0.10	0.08	0.11	0.12	0.08	0.07	0.09	0.09	0.07	0.53	0.57
Average	0.24	0.21	0.32	0.26	0.22	0.26	0.34	0.22	0.23	0.37	0.64	0.65

are fixed throughout the experiment. Our algorithm runs at 1 frame/sec in average based on an unoptimized Matlab implementation except patch matching function [2], which is written in C/C++.

To evaluate performance of our algorithm, we followed the same protocols in [25], where precision and success rate are measured by using densely sampled thresholds on center location error and bounding box overlap ratio, respectively. Figure 3 illustrates quantitative evaluation for all the sequences in the tracking benchmark. Performance of our tracker in benchmark dataset is competitive with the state-of-the-art online trackers, which indicates that our tracker is suitable to handle general challenges for tracking.

In the additional sequences involving more challenges, on the other hand, our tracker outperforms all other trackers with large margin and even comparable the state-of-the-art offline tracker, OMA [1]. It is because existing techniques typically rely on the first-order Markov assumption for tracking a new frame while our algorithm relaxes the restriction and isolates temporal tracking failures within local branches in the graph structure. The results for this experiment

are illustrated in Figure 4. Table 1 and 2 summarize the average scores of center location error and overlap ratio for the additional 10 sequences, respectively. The identified graph reflects the structure of an input sequence pretty well, which is illustrated in Figure 5. As shown in the figure, our algorithm maintains a reasonable representative frames $\mathcal{T}$ in each frame and propagates density function successfully to the new frame. The results for qualitative evaluation are presented in Figure 6.

6 Conclusion

We presented a novel online tracking algorithm, which progressively construct a graphical model beyond chain model, which is more appropriate for tracking. The target posterior of a new frame is estimated by propagating densities from previously tracked frames and making a weighted average of the densities based on the relevance of the existing frames with respect to the new frame. For computational efficiency, only a small number of frames is maintained for density propagation in an online manner, so that they capture important characteristics of input video and are potentially useful for tracking subsequent frames. Outstanding experimental results on 50 sequences in the tracking benchmark and 10 more challenging sequences show the benefit of our progressive graph construction algorithm for tracking.

Acknowledgments. This work was supported partly by MEST Basic Science Research Program through the NRF of Korea (NRF-2012R1A1A1043658), ICT R&D program of MSIP/IITP [14-824-09-006, Novel computer vision and machine learning technology with the ability to predict and forecast], and Samsung Electronics Co., Ltd.

References

1. Hong, S., Kwak, S., Han, B.: Orderless tracking through model-averaged posterior estimation. In: ICCV (2013)
2. Korman, S., Avidan, S.: Coherency sensitive hashing. In: ICCV (2011)
3. Bao, C., Wu, Y., Ling, H., Ji, H.: Real time robust l1 tracker using accelerated proximal gradient approach. In: CVPR (2012)
4. Mei, X., Ling, H.: Robust visual tracking using $l1$ minimization. In: ICCV (2009)
5. Zhong, W., Lu, H., Yang, M.-H.: Robust object tracking via sparsity-based collaborative model. In: CVPR (2012)
6. Han, B., Comaniciu, D., Zhu, Y., Davis, L.: Sequential kernel density approximation and its application to real-time visual tracking. TPAMI 30 (2008)
7. Liu, B., Huang, J., Yang, L., Kulikowski, C.A.: Robust tracking using local sparse appearance model and k-selection. In: CVPR, pp. 1313–1320 (2011)
8. Sevilla-Lara, L., Learned-Miller, E.G.: Distribution fields for tracking. In: CVPR, pp. 1910–1917 (2012)
9. Ross, D.A., Lim, J., Lin, R.-S., Yang, M.-H.: Incremental learning for robust visual tracking. IJCV 77 (2008)

10. Zhang, T., Ghanem, B., Liu, S., Ahuja, N.: Robust visual tracking via multi-task sparse learning. In: CVPR (2012)
11. Kwon, J., Lee, K.M.: Tracking of abrupt motion using wang-landau monte carlo estimation. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 387–400. Springer, Heidelberg (2008)
12. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-Learning-Detection. TPAMI (2012)
13. Babenko, B., Yang, M.-H., Belongie, S.: Robust object tracking with online multiple instance learning. TPAMI 33 (2011)
14. Hare, S., Saffari, A., Torr, P.H.S.: Struck: Structured output tracking with kernels. In: ICCV (2011)
15. Pan, P., Schonfeld, D.: Visual tracking using high-order particle filtering. Signal Processing Letters 18, 51–54 (2011)
16. Kwak, S., Nam, W., Han, B., Han, J.H.: Learning occlusion with likelihoods for visual tracking. In: ICCV (2011)
17. Buchanan, A.M., Fitzgibbon, A.W.: Interactive feature tracking using K-D trees and dynamic programming. In: CVPR (2006)
18. Gu, S., Zheng, Y., Tomasi, C.: Linear time offline tracking and lower envelope algorithms. In: ICCV (2011)
19. Uchida, S., Fujimura, I., Kawano, H., Feng, Y.: Analytical dynamic programming tracker. In: ACCV (2011)
20. Wei, Y., Sun, J., Tang, X., Shum, H.-Y.: Interactive offline tracking for color objects. In: ICCV (2007)
21. Sun, J., Zhang, W., Tang, X., Shum, H.-Y.: Bi-directional tracking using trajectory segment analysis. In: ICCV (2005)
22. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. IEEE Trans. Pattern Anal. Mach. Intell. 26, 810–815 (2004)
23. Jepson, A.D., Fleet, D.J., El-Maraghi, T.F.: Robust online appearance models for visual tracking. IEEE Trans. Pattern Anal. Mach. Intell. 25, 1296–1311 (2003)
24. Hoeting, J.A., Madigan, D., Raftery, A.E., Volinsky, C.T.: Bayesian model averaging: A tutorial. Statistical Science 14 (1999)
25. Wu, Y., Lim, J., Yang, M.-H.: Online object tracking: A benchmark. In: CVPR (2013)
26. Jia, X., Lu, H., Yang, M.-H.: Visual tracking via adaptive structural local sparse appearance model. In: CVPR (2012)
27. Dinh, G.T.B., Vo, N., Medioni: Context tracker: Exploring supporters and distracters in unconstrained environments. In: CVPR (2011)
28. Kwon, J., Lee, K.-M.: Visual tracking decomposition. In: CVPR (2010)
29. Kwon, J., Lee, K.M.: Tracking by sampling trackers. In: ICCV (2011)
30. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 702–715. Springer, Heidelberg (2012)
31. Sevilla-Lara, L., Learned-Miller, E.: Distribution fields for tracking. In: CVPR (2012)