Visual Tracking by Sampling Tree-Structured Graphical Models

Seunghoon Hong and Bohyung Han

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

Abstract. Probabilistic tracking algorithms typically rely on graphical models based on the first-order Markov assumption. Although such linear structure models are simple and reasonable, it is not appropriate for persistent tracking since temporal failures by short-term occlusion, shot changes, and appearance changes may impair the remaining frames significantly. More general graphical models may be useful to exploit the intrinsic structure of input video and improve tracking performance. Hence, we propose a novel offline tracking algorithm by identifying a tree-structured graphical model, where we formulate a unified framework to optimize tree structure and track a target in a principled way, based on MCMC sampling. To reduce computational cost, we also introduce a technique to find the optimal tree for a small number of key frames first and employ a semi-supervised manifold alignment technique of tree construction for all frames. We evaluated our algorithm in many challenging videos and obtained outstanding results compared to the state-of-the-art techniques quantitatively and qualitatively.

Keywords: Visual tracking, tree-structured graphical model, Markov Chain Monte Carlo (MCMC), manifold alignment

1 Introduction

Although visual tracking problem has been studied extensively for decades, the underlying graphical model of most existing probabilistic tracking algorithms is limited to linear structure, *i.e.*, first-order Markov chain. Such chain model is a reasonable choice since it is simple and typically well-suited for online tracking algorithms. However, in offline tracking algorithms (or online tracking algorithms allowing some time delay), more general graphical models are potentially helpful to overcome typical limitations of tracking algorithms relying on graphical models with linear structure. For example, when we track an object with view point changes or alternating appearances, tracking by a graphical model with multiple branches may be a better option to handle multi-modality of target and scene. Also, in chain model, inference of the target posterior at the current frame depends only on the previous frame, which may be problematic when the previous frame is not sufficiently correlated with the current one due to large motion, shot changes, occlusion, etc.

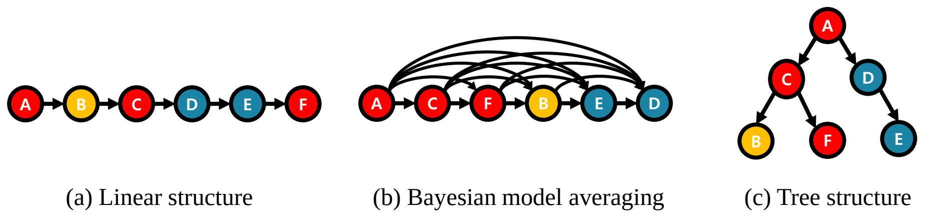


Fig. 1. Potential graphical models for tracking. Each frame is represented by a node, where the color in each node indicates the type of target variation and the temporal order of frames is given by the alphabetical order. (a) In chain model, target is tracked by the temporal order of frames regardless of target variations. (b) Bayesian model averaging [1] infers target states in an increasing order of variations and makes a blind average of the posteriors obtained from all tracked frames. (c) In tree structure, we reorder frames based on target variations similar to (b) and estimate the posterior of a node by propagating density from a single parent, one of the frames already tracked.

Motivated by these facts, we propose a novel offline tracking algorithm based on a tree-structured representation of a video, where a node corresponds to a frame and the posterior is propagated from root to leaves sequentially along multiple branches in the tree. We claim that tracking in a tree-structured graphical model be more advantageous than chain model or Bayesian model averaging [1] due to the following reasons. First, it is natural to handle multi-modality of target and scene by identifying a reasonable tree structure and maintaining only relevant frames in each branch. Second, it is possible to achieve orderless tracking as in [1] by organizing frames based on their tracking complexity, where challenging frames are located near leaf nodes. Third, each node always has a single parent and existing sequential inference techniques can be employed directly without sophisticated posterior aggregation methods. Figure 1 illustrates the main concept of our algorithm compared to chain model and Bayesian model averaging method.

In this work, our goal is to find the optimal tree structure appropriate for tracking and improve performance by exploiting the identified tree. To achieve better tracking performance, the obtained tree structure should reflect correct dependency between frames with respect to target and scene. However, the identification of a good tree structure may require the estimation of target state in each frame. Learning the optimal tree structure and performing accurate tracking is a chicken-and-egg problem since it is practically infeasible to know which tree is optimal before tracking[1].

Therefore, we propose an iterative framework that learns tree structure and solves target tracking jointly, where a new tree structure is proposed by sampling and validated by tracking in each iteration. The proposed iterative algorithm is formulated by a Markov Chain Monte Carlo (MCMC) technique, which improves

[1] It is still difficult to determine the goodness of a tree even after tracking without evaluating results using ground-truth.

the quality of solution and provides the convergence of algorithm. For computational efficiency, the procedure is performed on a small subset of frames, and the learned tree is extended to the rest of frames in the input sequence by semi-supervised manifold alignment. Tracking in a tree-structured graphical model is similar to the standard sequential Bayesian filtering except that there exist multiple branches for density propagation. Main contributions of our study are summarized below:

- We propose an offline tracking algorithm based on a tree-structured graphical model, which is conceptually more general and reasonable compared to chain model; tree structures are well-suited to handle appearance variations, fast motion, shot changes, occlusion, temporal failures, etc.
- We formulate a unified framework of optimizing tree structure and tracking target in a principled way based on MCMC sampling.
- For efficiency, we estimate the optimal tree using a small number of key frames first, and employ a semi-supervised manifold alignment technique to construct a tree for all frames.

This paper is organized as follows. We first review related work in Section 2. The main framework of our algorithm is discussed in Section 3, and the hierarchical tree construction by semi-supervised manifold alignment is presented in Section 4. Section 5 illustrates experimental results.

2 Related Work

Visual tracking algorithm involves two main components, appearance modeling and tracker control. Recently, research on online appearance modeling is active, and various techniques have been proposed so far, *e.g.*, sparse reconstruction [2–4], online density estimation [5], incremental subspace learning [6], multi-task learning [7], multiple instance learning [8], P-N learning [9], and so on. However, the progress on tracker control issue is slow and many recent tracking algorithms still rely on sequential Bayesian filtering, tracking-by-detection or their variations, which are based on chain models with temporal smoothness assumption. Tracking algorithms based on temporally ordered chain models are inherently weak in abrupt motion, shot changes, significant appearance changes, and occlusion; [10] proposes an online tracking algorithm to tackle abrupt motion and shot changes by adopting Wang-Landau Monte Carlo sampling, but it still has the weakness of chain models and suffers from weak observation model.

Offline trackers utilize all frames in input video, and are more suitable to tackle abrupt target variations. They typically formulate tracking as a global optimization problem and solve it with standard techniques such as dynamic programming [11–14], Hidden Markov Model [15], and so on. However, even most of offline tracking algorithms employ chain models, which are not flexible enough to handle various challenges. The problem is sometimes alleviated by employing multiple key frames given beforehand [12, 15] or by user interaction [11, 14]. Recently, [1] proposes an algorithm to actively search an appropriate 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. This method seems to be better than chain models, but the blind averaging of density functions may contaminate the target posterior due to propagation of unreliable density functions.

MCMC is a useful and flexible tool to explore large solution space efficiently in many applications. Learning a graph structure by MCMC sampling is not new in computer vision community; [16] investigates image segmentation problem by constructing a tree and [17] samples graph structures of video for event summarization and rare event detection. However, to the best of our knowledge, there is no prior study about learning a graphical model for visual tracking problems through MCMC or any other methods.

3 Main Framework

Given an input video, we aim to simultaneously obtain tree structure for tracking and improve tracking based on the tree. Let $\mathbf{G} = (\mathcal{V}, \mathcal{E})$ be a tree, where $\mathcal{V} = \{v_1, \ldots, v_N\}$ denotes a set of nodes corresponding to N frames and $\mathcal{E} = \{e_1, \ldots, e_{N-1}\}$ denotes a set of directed edges defining conditional dependency between frames. Our goal is to obtain a set of target states for all frames $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and corresponding target templates $\mathcal{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$ through tracking on $\mathbf{G}$. Since tracking is impossible without a tree and the optimal tree is difficult to obtain without tracking, we alternate the following steps in each iteration of our algorithm.

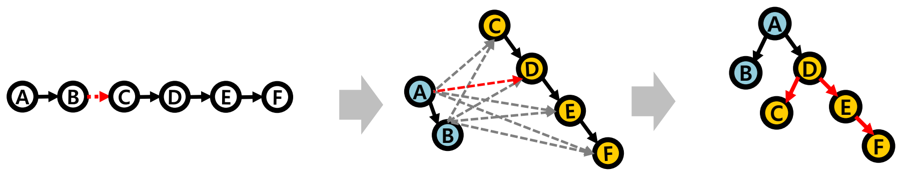
- Given a tree structure $\mathbf{G}$, perform tracking on the current tree-based probabilistic graphical model, and obtain the target states $\mathcal{X}$ and templates $\mathcal{Y}$ in all frames.
- Given tracking results $\mathcal{X}$ and $\mathcal{Y}$, propose a new tree structure $\mathbf{G}^*$ based on individual edge reliabilities in $\mathbf{G}$.

These procedures are repeated until it converges to a local optimum or reaches the predefined number of iterations. The final outputs of the algorithm are an optimal tree structure $\widehat{\mathbf{G}}$ and corresponding tracking results, $(\widehat{\mathcal{X}}, \widehat{\mathcal{Y}})$. We discuss each step of the MCMC framework in detail.

3.1 Learning Tree Structure by MCMC

There are a tremendous number of tree-structured graphical models to represent a video, and it is not straightforward to find an optimal tree for tracking efficiently. Hence, we design a good strategy to find the optimal solution in a limited number of iterations by employing MCMC sampling.

Each iteration of MCMC is composed of two steps—proposal step and acceptance step; the former draws a sample from a proposal distribution and the latter performs a probabilistic test to accept or reject the sample.



(a) edge deletion (b) edge addition (c) edge reversing

Fig. 2. Proposal of a new tree by $Q(\mathbf{G}^*|\mathbf{G})$. Given a current tree, deleting and adding edges are determined based on P_{delete} and P_{add}, respectively. Then, the directions of some edges are reversed to make all edges be directed from root to leaves.

Proposal step Given a current tree structure denoted by $\mathbf{G}$, a new tree sample $\mathbf{G}^*$ is proposed from proposal distribution $Q(\mathbf{G}^*|\mathbf{G})$. A new tree is sampled by replacing a subset of existing edges in $\mathbf{G}$ with new ones. For simplicity, we limit the space of $Q(\mathbf{G}^*|\mathbf{G})$ to a single edge *delete-and-add* operation as

$$Q(\mathbf{G}^*|\mathbf{G}) = P_{delete} \cdot P_{add} \tag{1}$$

where P_{delete} and P_{add} denote the probabilities to delete and add an edge, respectively.

It is practically impossible to compute the two probabilities without tracking on $\mathbf{G}$. We assume that a set of target templates $\mathcal{Y}$ is obtained by tracking on $\mathbf{G}$ in the previous MCMC iteration. Note that detailed tracking algorithm is presented in Section 3.2. The reliability of an edge is measured by the distance based on the amount of deformation between target templates in two connected frames, which is given by

$$d(i,j) \equiv \underset{m}{\text{median}} \left(\|\mathbf{v}_i^m - f_{PM}(\mathbf{v}_i^m; \mathbf{y}_j)\| \right), \tag{2}$$

where $f_{PM}(\mathbf{v}_i^m; \mathbf{y}_j)$ is a patch matching function [18] from the m^{th} patch located at $\mathbf{v}_i^m$ in $\mathbf{y}_i$ to $\mathbf{y}_j$. P_{delete} and P_{add} between two frames i and j are defined as

$$P_{delete}(i,j) = \frac{\exp(d(i,j))}{\sum_{(a,b)\in\mathcal{E}} \exp(d(a,b))} \qquad (i,j) \in \mathcal{E} \tag{3}$$

$$P_{add}(i,j) = \frac{\exp(-d(i,j))}{\sum_{(a,b)\in\overline{\mathcal{E}}} \exp(-d(a,b))} \qquad (i,j) \in \overline{\mathcal{E}} \tag{4}$$

where $\overline{\mathcal{E}}$ is the complement of $\mathcal{E}$ and does not include the edges creating loops.

By sampling from Eq. (1), deleting and adding edges are determined and a new tree $\mathbf{G}^*$ is proposed. If necessary, we change the directions of some edges and make all edges have the same direction—from root to leaves. The procedure to propose a new tree sample $\mathbf{G}^*$ from $Q(\mathbf{G}^*|\mathbf{G})$ is illustrated in Figure 2. By replacing an unreliable edge in $\mathcal{E}$ with a probable one in $\overline{\mathcal{E}}$, our algorithm learns a better tree $\mathbf{G}^*$, which does not rely on temporal order, and improves tracking performance by potentially utilizing a different path to track frames failed in the previous iterations.

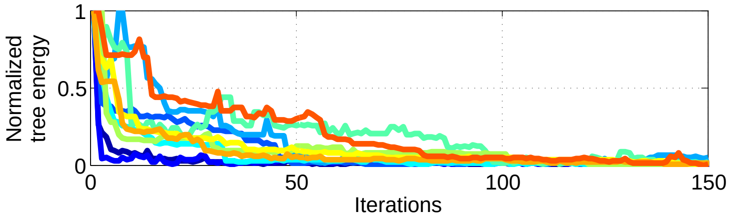


Fig. 3. Tree energy in each MCMC iteration for 10 videos used in our experiment.

Acceptance step Once a new tree $\mathbf{G}^*$ is proposed, acceptance step decides whether the tree is accepted or not. To make the decision, we define recursively a tree energy to measure the quality of tree for tracking as

$$-\log p(\mathcal{Y}|\mathbf{G}) = \sum_{i}^{N} c_i = \sum_{i}^{N} \max(d(i, p_i), c_{p_i}) \tag{5}$$

where c_i denotes the cost to track from root to frame i, and p_i is the frame index of i's parent. Given the proposed tree structure $\mathbf{G}^*$ from Eq. (1), MCMC framework accepts or rejects the sample based on the acceptance ratio α, which is given by

$$\alpha = \min\left[1, \frac{(-\log p(\mathcal{Y}^*|\mathbf{G}^*))^{-1} Q(\mathbf{G}|\mathbf{G}^*)}{(-\log p(\mathcal{Y}|\mathbf{G}))^{-1} Q(\mathbf{G}^*|\mathbf{G})}\right]. \tag{6}$$

An important property of the tree energy function defined in Eq. (5) is that tracking error in any node is propagated to its descendants; internal nodes receive much larger penalty for tracking failures than leaf nodes. Therefore, large tracking errors in internal nodes tend to be corrected at an early iteration of MCMC and the algorithm gradually places the challenging frames to near leaf nodes, which isolates tracking failures to local areas (subtrees) and reduces error propagation. Most energy functions converge fast as illustrated in Figure 3, which may result from the reason described above.

Once MCMC iterations are completed, we take the best tree by

$$\widehat{\mathbf{G}} = \arg\min_{\mathbf{G}^l} -\log p(\mathcal{Y}^l|\mathbf{G}^l), \quad l = 1, \ldots, M \tag{7}$$

where l is the sample index and M is the number of iterations. Overall algorithm is summarized in Algorithm 1. In our algorithm, tree sampling and target tracking are tightly coupled to learn a graphical model; tracking is used to measure tree energy in MCMC iterations. We set the initial tree $\mathbf{G}^0$ to the conventional chain model in temporal order.

Due to huge search space, it is difficult to achieve the globally optimal solution after convergence. However, the tree structure obtained from MCMC sampling typically reduces tracking complexity and consequently is much better than temporally ordered chain model. In practice, tracking on the converged tree structure improves performance significantly, which will be presented in Section 5.

Algorithm 1: Tree construction by MCMC sampling

```
   Input: Initial tree $\mathbf{G}^0$
   Output: Best tree $\widehat{\mathbf{G}}$, $\widehat{\mathcal{X}} = \{\mathbf{x}_1^*, \ldots, \mathbf{x}_N^*\}, \widehat{\mathcal{Y}} = \{\mathbf{y}_1^*, \ldots, \mathbf{y}_N^*\}$
1  foreach $l = 0, ..., M-1$ do
2      Propose $\mathbf{G}^*$ from $Q(\mathbf{G}^*|\mathbf{G}^l)$ by Eq. (1)
3      Evaluate edge cost of $\mathbf{G}^*$ by
4         1) Running tracking on $\mathbf{G}^*$ by Section 3.2
5         2) Obtaining $\mathcal{Y}^*$
6         3) Calculating $d(i,j)$ by Eq. (2)
7      Calculate $\alpha$ with $-\log p(\mathcal{Y}^*|\mathbf{G}^*)$ in Eq. (5)
8      $\theta \sim U[0,1]$
9      if $\alpha \geq \theta$ then
10         $\mathbf{G}^{l+1} = \mathbf{G}^*$
11     else
           $\mathbf{G}^{l+1} = \mathbf{G}^l$
13     end
   end
```

3.2 Tracking on Tree Structure

When a tree is proposed, we estimate the posterior of each frame based on the tree and decide its acceptance. This subsection describes a tracking algorithm to evaluate the proposed tree and generate the final tracking result.

Our tracking algorithm on a tree structure $\mathbf{G}$ has the following properties: 1) we set the root of tree as the initial frame for tracking, and 2) there exists a unique path from the root to any other node in the tree, which is modeled by a first-order Markov chain. Given these properties, the target posterior at frame t is estimated by a sequential Bayesian filtering as

$$P(\mathbf{x}_t|\{\mathbf{z}_i\}_{i=1,\ldots,t}, \mathbf{G}) \propto P(\mathbf{z}_t|\mathbf{x}_t) \int_{\mathbf{x}_{p_t}} P(\mathbf{x}_t|\mathbf{x}_{p_t}) P(\mathbf{x}_{p_t}|\{\mathbf{z}_i\}_{i=1,\ldots,p_t}, \mathbf{G}) d\mathbf{x}_{p_t} \quad (8)$$

where $\{\mathbf{z}_i\}_{i=1,\ldots,t}$ denotes a set of observations on the path up to node t and p_t denotes the index of t's parent. We start the density propagation from the root, which is performed independently in each branch by Eq. (8). It enables separate branches to model different types of the target variations. This property has another benefit in the next iteration of the MCMC; it makes it possible to reuse posteriors of common branches in different trees. In other words, although we track on a new tree $\mathbf{G}^*$ in each iteration, we do not need to track all frames again because the update in tree structure is minimal and the posteriors in many frames often remain unchanged. In addition, difficult frames for tracking tend to be relocated near leaf nodes, and computational cost gets smaller in later iterations since modified parts in a new tree $\mathbf{G}^*$ are mostly small.

To propagate the target posterior from parent to child, we need to define transition and likelihood model, $P(\mathbf{z}_t|\mathbf{x}_t)$ and $P(\mathbf{x}_t|\mathbf{x}_{p_t})$, respectively, in Eq. (8). Since the temporal order of frames may not be preserved in tree hierarchy, we cannot assume temporal smoothness or spatial coherency between connected frames. For this reason, we adopt patch matching and voting process as in [1] because of its robustness in this situation. For the purpose, we represent the posterior in Eq. (8) with a set of discrete samples as

$$P(\mathbf{x}_t|\{\mathbf{z}_i\}_{i=1,\ldots,t}, \mathbf{G}) \approx \sum_{\mathbf{x}_{p_t}^j \in \mathbb{S}_{p_t}} P(\mathbf{z}_t|\mathbf{x}_t)P(\mathbf{x}_t|\mathbf{x}_{p_t}^j), \tag{9}$$

where $\mathbb{S}_{p_t}$ denotes a set of samples drawn from $P(\mathbf{x}_{p_t}|\{\mathbf{z}_i\}_{i=1,\ldots,p_t}, \mathbf{G})$ denoting the posterior of parent. The prediction and measurement steps are modeled jointly by patch matching and voting process; for every rectangular patches in frame p_t, we apply the patch matching function f_{PM} to finding patch correspondences between frame p_t and t. Similar to the method in implicit shape model [19], every matched patch votes to the target center as

$$P(\mathbf{x}_t|\{\mathbf{z}_i\}_{i=1,\ldots,t}, \mathbf{G}) \approx \sum_{\mathbf{x}_{p_t}^j \in \mathbb{S}_t} \sum_{m=1}^{K_j} \mathcal{N}(\mathbf{x}_t; f_{PM}(\mathbf{v}_j^m) - \mathbf{c}_j^m, \mathbf{\Sigma}), \tag{10}$$

where K_j is the number of patches in a sample bounding box, and $\mathbf{c}_j^m$ is the offset from $\mathbf{v}_j^m$ to $\mathbf{x}_{p_t}^j$. Each voting is smoothed by a Gaussian kernel with bandwidth $\mathbf{\Sigma}$. An efficient matching function f_{PM} is obtained from coherency sensitive hashing [18] as suggested in [1]. To further reduce voting errors, the voting map from the initial frame is used as an additional source of observation and the posterior is updated by the following equation:

$$P(\mathbf{x}_t|\{\mathbf{z}_i\}_{i=1,\ldots,t}, \mathbf{G}) \approx P(\mathbf{x}_t|\mathbf{z}_1, \mathbf{z}_t) \left[\sum_{\mathbf{x}_{p_t}^j \in \mathbb{S}_t} \sum_{m=1}^{K_j} \mathcal{N}(\mathbf{x}_t; f_{PM}(\mathbf{c}_j^m) - \mathbf{a}_j^m, \mathbf{\Sigma}) \right], \tag{11}$$

where $P(\mathbf{x}_t|\mathbf{z}_1, \mathbf{z}_t)$ denotes density propagation from the initial frame. Note that this term is not related to graph structure. We can estimate target posterior in every frame by propagating density based on Eq. (11). The target state in each frame is obtained by

$$\mathbf{x}_t^* = \arg\max_{\mathbf{x}_t} P(\mathbf{x}_t|\{\mathbf{z}_i\}_{i=1,\ldots,t}, \mathbf{G}). \tag{12}$$

4 Hierarchical Construction of Tree Structure

The computational complexity of our tree construction algorithm is $O(MN)$, where M and N denote the numbers of iterations and frames, respectively, and tracking on a tentative tree in an MCMC iteration is time consuming. Tracking results in some frames, *e.g.*, frames with smooth or repetitive motion patterns in temporal neighborhood, are strongly correlated, and we propose a hierarchical approach to construct the tree by performing MCMC iterations only for a small subset of frames and extending the tree to the remaining frames. We perform a hierarchical tree construction based on semi-supervised manifold alignment.

4.1 Tree Construction on Key Frames

To obtain a meaningful tree structure of a video based on a small number of key frames, they should be representative and preserve crucial information about the variations in the input video. We adopt k-means clustering for key frame selection, for which we define the distance between frames. A patch-based bidirectional dissimilarity is employed as distance measure, which is given by

$$\mathbf{\Delta}_f(i,j) = \frac{1}{n_i} \sum_{\mathbf{a}\in I_i} \min_{\mathbf{b}\in I_j} \delta(\mathbf{a},\mathbf{b}) + \frac{1}{n_j} \sum_{\mathbf{b}\in I_j} \min_{\mathbf{a}\in I_i} \delta(\mathbf{b},\mathbf{a}), \tag{13}$$

where I_i and I_j denote a pair of images that contain n_i and n_j patches, respectively, and $\delta(\mathbf{a},\mathbf{b})$ denotes dissimilarity between patches $\mathbf{a} \in I_i$ and $\mathbf{b} \in I_j$. Since triangle inequality is not preserved with the measure in Eq. (13), all frames are embedded onto a metric space by Isomap [20]. Simple k-means clustering is applied on the embedded space, and the closest frame to each cluster center is selected as a key frame. This idea is similar to [21] except that k-means clustering is used instead of k-center method.

Once we select a set of key frames, MCMC iterations are performed on the key frames as described in Section 3. After convergence, we obtain the optimal tree structure $\widehat{\mathbf{G}}$ and tracking results $(\widehat{\mathcal{X}}, \widehat{\mathcal{Y}})$ for key frames.

4.2 Tree Extension by Manifold Alignment

It is a reasonable idea to construct a tree by simply running a minimum spanning tree algorithm in the metric space identified in Section 4.1. However, it is based only on scene dissimilarities and does not count target-specific information. Therefore, it would be better to find a joint embedding space that considers both scene and target distances. Suppose that we have another embedding for key frames based on target distance $\mathbf{\Delta}_o(i,j)$ similar to Eq. (13), where patches are extracted from target templates instead of whole images. Then, we construct a tree to cover all frames by extending $\widehat{\mathbf{G}}$ using the partial correspondences between two embeddings, which are given by key frames.

We adopt a semi-supervised manifold alignment algorithm [22] to establish the joint metric space. Given two dissimilarity matrices, $\mathbf{\Delta}_f$ and $\mathbf{\Delta}_o$, and the correspondences between key frames across two embeddings, our goal is to find a new metric space given by $\mathbf{h} = [\mathbf{s}^{\mathrm{T}}\ \mathbf{t}^{\mathrm{T}}]^{\mathrm{T}}$, where $\mathbf{s}$ and $\mathbf{t}$ represent the embedding coordinates extracted separately for scene and target, respectively. Key frames in the two different spaces are mapped to similar locations in the new space by optimizing the following objective function:

$$\min_{\mathbf{s},\mathbf{t}} \varPhi(\mathbf{s},\mathbf{t}) \equiv \mu \sum_{i\in\mathcal{K}} |s_i - t_i|^2 + \mathbf{s}^{\mathrm{T}}\mathbf{L}^s\mathbf{s} + \mathbf{t}^{\mathrm{T}}\mathbf{L}^t\mathbf{t}, \tag{14}$$

where μ is a weighting factor, $\mathcal{K}$ is a set of key frames, and $\mathbf{L}^s$ and $\mathbf{L}^t$ are graph Laplacian matrices based on scene and target distances, respectively. The optimization of Eq. (14) is an ill-posed problem, but can be solved by minimizing Rayleigh quotient as

$$\min_{\mathbf{h}} \hat{\Phi}(\mathbf{h}) = \min_{\mathbf{h}} \frac{\mathbf{h}^{\mathrm{T}}\mathbf{L}^{c}\mathbf{h}}{\mathbf{h}^{\mathrm{T}}\mathbf{h}} \tag{15}$$

where $\mathbf{L}^c$ is the combined graph Laplacian matrix, which is given by

$$\mathbf{L}^c = \begin{bmatrix} \mathbf{L}^s + \mathbf{U}^{ss} & -\mathbf{U}^{st} \\ -\mathbf{U}^{ts} & \mathbf{L}^t + \mathbf{U}^{tt} \end{bmatrix}, \tag{16}$$

where $[\mathbf{U}]_{ij} = \mu$ if $i = j \in \mathcal{K}$ and 0 otherwise.

After we obtain the solution of Eq. (15), all frames are embedded to new metric space defined by $\mathbf{h}$. We first add edges between key frames by $\widehat{\mathbf{G}}$ optimized in Section 3. Then, the tree is extended to remaining frames based on a similar way to minimum spanning tree algorithm in the new metric space, and tracking is performed on the extended tree with all frames. The computational complexity of this approach is $O(kM + N - k)$, where k ($\ll N$) is the number of key frames. The complexity of a comparable algorithm OMA [1] is $O(kN)$, which shows that our algorithm is more efficient than OMA since M is typically smaller than N.

5 Experiments

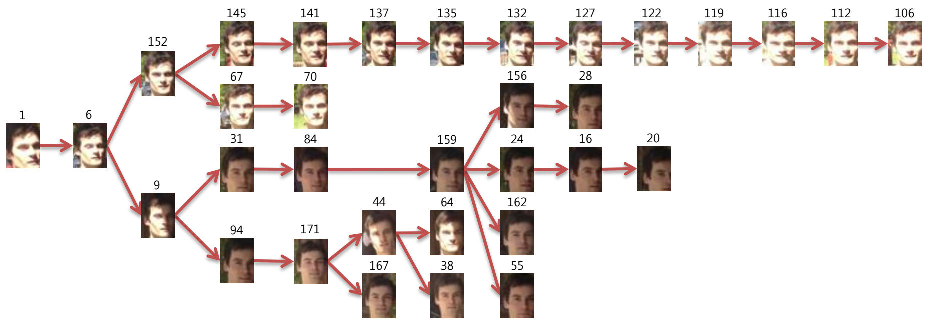
We describe the details about our experiment setting, and illustrate the performance of our algorithm compared to the state-of-the-art techniques in challenging video sequences.

5.1 Datasets

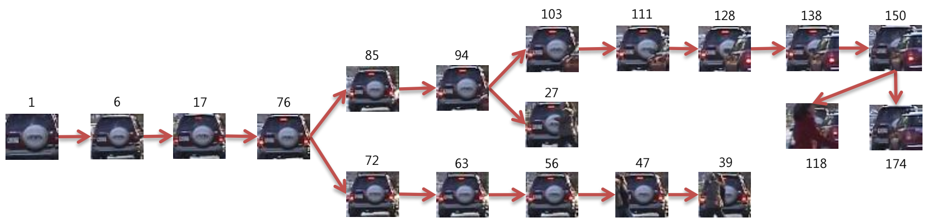
For the evaluation of our tracking algorithm, we collected 10 video sequences, which are collected from [1, 23, 24] and also included new sequences. These sequences are often too difficult to be handled by traditional online tracking algorithms and appropriate to test the benefit of our offline tracking algorithm since they involve interesting challenges such as low frame rate (*tennis*), shot changes (*boxing, young, skating, dance*), occlusion (*campus, TUD*), appearance changes (*sunshade*), fast motion (*bike*), motion blur (*jumping*), etc.

5.2 Identified Tree Structure

The tree structures learned from our MCMC sampling and semi-supervised manifold alignment are reasonable and appropriate for tracking in many cases. We illustrate two examples of key frame trees. As shown in Figure 4(a), the identified tree structure of key frames for *sunshade* sequence divides two cases effectively—when human face is located in sunny side and under shadow—and enables tracker to handle them independently. On the other hand, in Figure 4(b), frames that are difficult to track due to occlusion are placed close to leaf nodes. These two



(a) Tree obtained from the key frames in *sunshade* sequence. Two main subtrees maintain bright and dark sides of face appearances.



(b) Tree obtained from the key frames in *campus* sequence. Frames with occlusion are near leaf nodes, which minimizes error propagation.

Fig. 4. Examples of tree structures identified with key frames.

examples present the effectiveness of our tree construction algorithm and the potential to improve tracking performance. Also, Figure 5 visualizes a tree with all frames for *skating* sequence, where we can observe a complex tree structure completely different from chain model.

5.3 Quantitative and Qualitative Performance

We compared the proposed algorithm with 18 state-of-the-art tracking methods. Most of them are online trackers, which include MIL [8], LSK [25], CSK [26], DFT [27], IVT [6], MTT [7], VTD [28], VTS [29], FRAG [30], L1APG [2], CXT [31], ASLA [32], SCM [4], Struck [33], TLD [9] and WLMC [10], while OTLE [12] and OMA [1] are offline trackers. We received source codes or binaries from authors of individual tracking algorithms. Our tracking algorithm is denoted by TST (Tracking by Sampling Tree). Among the tracking algorithms, OMA is most related to ours since it uses the same observation model and shares the temporal orderlessness property.

To evaluate performance, two most common measures—center location error and bounding box overlap ratio—are employed. For patch matching, the same parameter setting with [1] is employed; patch size is 8×8 and 900 samples are drawn in 9 scales from 0.6 to 1.4 to populate hypotheses to other frames. The

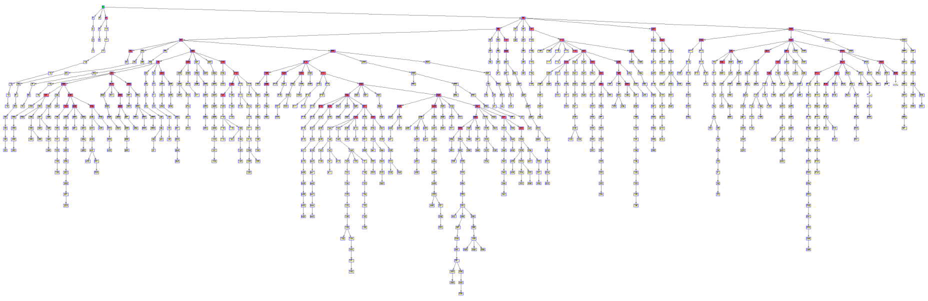


Fig. 5. An example of the entire tree structure for *skating* sequence. The solid (magenta) nodes indicate key frames. There exist multiple branches that are expected to model various aspects of target and scene.

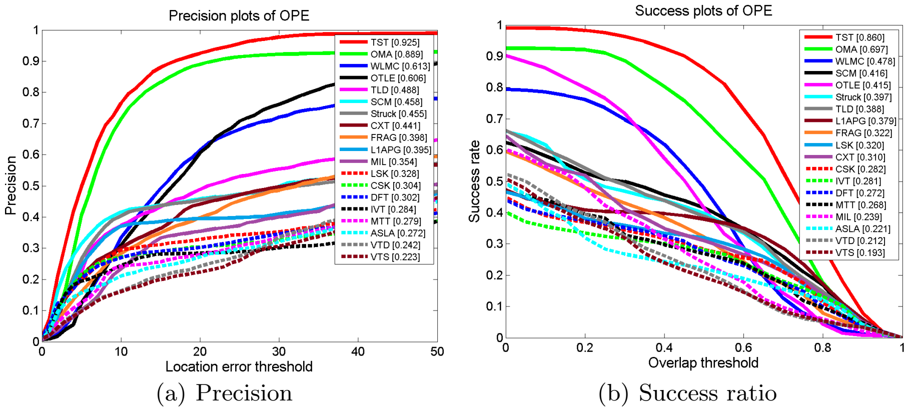


Fig. 6. Precision and success ratio of all compared algorithms. Precision and success ratio are measured by center location error and bounding box overlap ratio, respectively. The ranks are determined at center location error 25 and overlap ratio 0.5.

number of key frames in each sequence is set to 10% of all frames. Note that all parameters are fixed across sequences.

Figure 6 summarizes quantitative evaluation results for all tested sequences and algorithms in terms of center location error and bounding box overlap ratio. Table 1 and 2 present more comprehensive results for the selected algorithms that perform well either in tracking benchmark [24] or Figure 6. Our algorithm generally outperforms other methods and this fact is clear in Figure 6. Performance of OMA is comparable to ours, especially in the sequences with shot changes, but its overall accuracy is lower than ours according to our experiment. OMA fails to handle strong bi-modal target appearance as in *sunshade* sequence due to its blind model-averaging property. When we construct a tree structure for all frames from the initial chain model with key frames and run our tracking algorithm, we obtain 17.8 and 0.64 for the center location error

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

	FRAG	L1APG	CXT	ASLA	SCM	Struck	TLD	WLMC	OTLE	OMA	TST
bike	104.2	39.3	22.2	88.6	13.8	8.4	16.9	34.4	20.1	17.7	15.6
campus	3.3	16.1	33.4	12.2	12.2	83.1	46.7	13.5	5.8	3.2	1.4
TUD	17.3	7.4	36.4	72.6	12.2	54.4	18.9	68.2	27.3	4.4	4.1
sunshade	35.8	42.8	30.6	37.2	44.9	3.9	19.9	61.1	9.1	88.1	5.3
jumping	21.8	3.2	12.6	49.0	3.1	3.3	11.7	127.6	20.2	3.4	2.8
tennis	67.4	84.9	129.8	67.2	65.9	109.5	64.5	30.9	36.2	6.9	5.6
boxing	80.0	117.4	137.3	137.3	96.0	122.7	73.3	11.7	41.6	10.5	10.6
youngki	97.5	144.1	68.1	144.1	115.0	115.1	60.2	16.0	15.7	11.4	13.5
skating	35.4	143.9	41.5	45.2	49.4	23.8	35.3	14.7	18.3	8.0	6.1
dance	132.4	167.2	176.8	117.5	208.0	107.1	105.0	39.7	118.8	15.1	18.6
Average	59.5	76.6	68.9	77.1	62.1	63.1	45.2	45.2	31.3	16.9	8.4

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

	FRAG	L1APG	CXT	ASLA	SCM	Struck	TLD	WLMC	OTLE	OMA	TST
bike	0.08	0.18	0.39	0.16	0.46	0.54	0.45	0.39	0.27	0.40	0.56
campus	0.77	0.52	0.56	0.63	0.62	0.24	0.50	0.52	0.72	0.78	0.86
TUD	0.59	0.85	0.51	0.30	0.67	0.30	0.67	0.38	0.49	0.82	0.80
sunshade	0.33	0.32	0.49	0.43	0.45	0.78	0.57	0.24	0.60	0.29	0.70
jumping	0.31	0.77	0.40	0.20	0.76	0.75	0.56	0.07	0.26	0.74	0.79
tennis	0.11	0.29	0.08	0.12	0.11	0.28	0.10	0.43	0.33	0.63	0.74
boxing	0.22	0.13	0.01	0.03	0.13	0.04	0.21	0.65	0.38	0.70	0.71
youngki	0.19	0.02	0.38	0.12	0.13	0.09	0.24	0.62	0.54	0.62	0.66
skating	0.25	0.02	0.25	0.13	0.20	0.40	0.33	0.46	0.41	0.42	0.55
dance	0.14	0.02	0.08	0.10	0.07	0.08	0.07	0.45	0.30	0.52	0.52
Average	0.30	0.31	0.32	0.22	0.36	0.35	0.36	0.42	0.43	0.60	0.70

and the bounding box overlap ratio on average, respectively, which are better than OMA but non-trivially worse than our algorithm. These results indicate the benefit of tracking on tree structure learned from MCMC procedure. The results for qualitative evaluation are illustrated in Figure 7.

6 Conclusion

We presented a novel offline tracking algorithm based on automatic tree-structured graphical model construction. Our algorithm optimizes the structure of input video through MCMC sampling and performs tracking using the identified tree structure. Since an MCMC iteration is computationally expensive, we proposed a hierarchical tree construction algorithm by a semi-supervised manifold alignment technique. The learned tree structure tends to locate challenging frames near leaf nodes and isolate potential tracking failures in a small region. The proposed algorithm improves tracking performance substantially.

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; 14-824-09-014, Basic software research in human-level lifelong machine learning (Machine Learning Center)], and Samsung Electronics Co., Ltd.

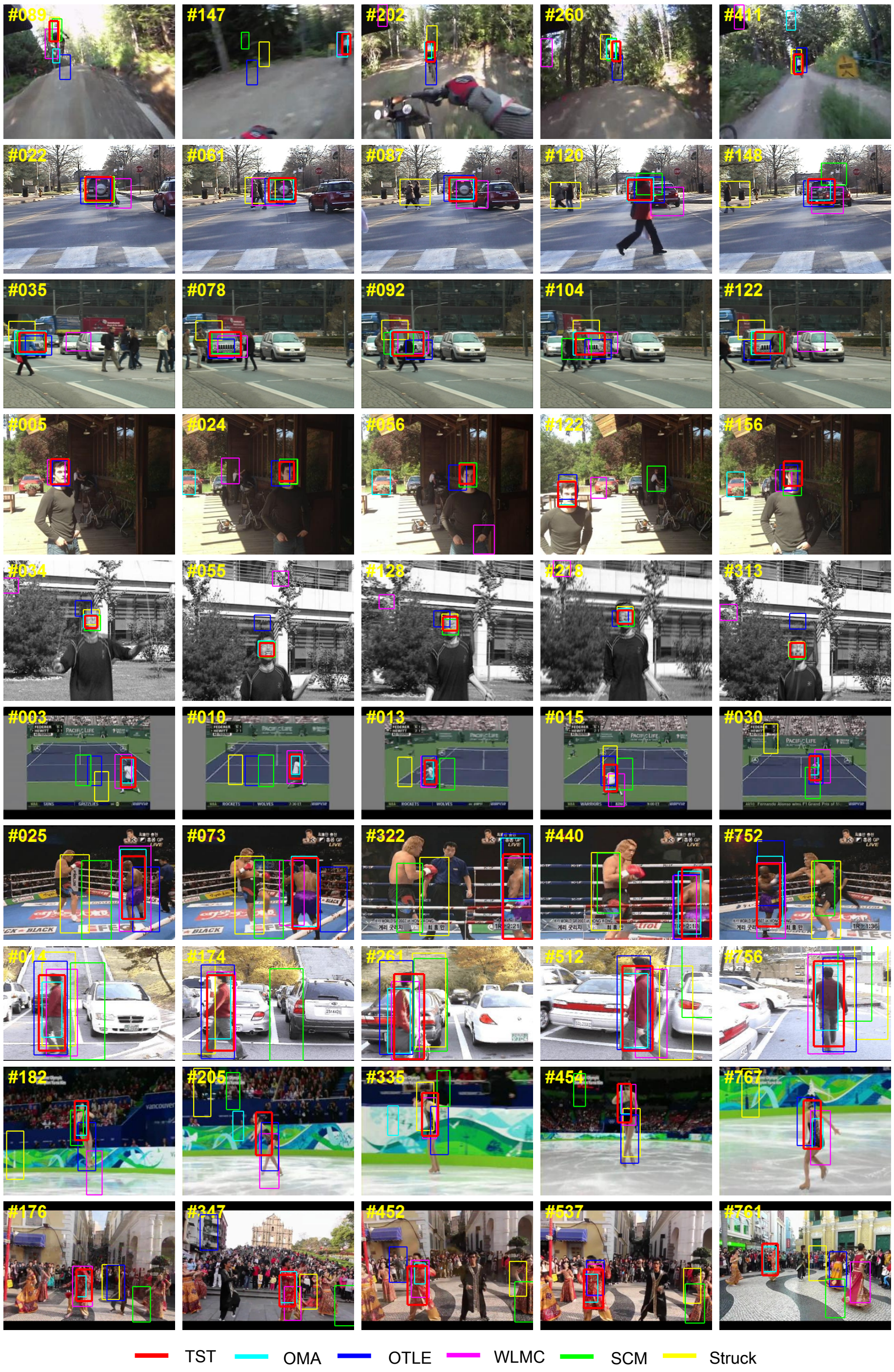


Fig. 7. Tracking results for all sequences: from top to bottom, *bike*, *campus*, *TUD*, *sunshade*, *jumping*, *tennis*, *boxing*, *youngki*, *skating* and *dance* sequence.

References

1. Hong, S., Kwak, S., Han, B.: Orderless tracking through model-averaged posterior estimation. In: ICCV. (2013)
2. Bao, C., Wu, Y., Ling, H., Ji, H.: Real time robust l1 tracker using accelerated proximal gradient approach. In: CVPR. (2012)
3. Mei, X., Ling, H.: Robust visual tracking using $l1$ minimization. In: ICCV. (2009)
4. Zhong, W., Lu, H., Yang, M.H.: Robust object tracking via sparsity-based collaborative model. In: CVPR. (2012)
5. Han, B., Comaniciu, D., Zhu, Y., Davis, L.: Sequential kernel density approximation and its application to real-time visual tracking. TPAMI **30** (2008)
6. Ross, D.A., Lim, J., Lin, R.S., Yang, M.H.: Incremental learning for robust visual tracking. IJCV **77** (2008)
7. Zhang, T., Ghanem, B., Liu, S., Ahuja, N.: Robust visual tracking via multi-task sparse learning. In: CVPR. (2012)
8. Babenko, B., Yang, M.H., Belongie, S.: Robust object tracking with online multiple instance learning. TPAMI **33** (2011)
9. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-Learning-Detection. TPAMI (2012)
10. Kwon, J., Lee, K.M.: Tracking of abrupt motion using wang-landau monte carlo estimation. In: ECCV. (2008)
11. Buchanan, A.M., Fitzgibbon, A.W.: Interactive feature tracking using K-D trees and dynamic programming. In: CVPR. (2006)
12. Gu, S., Zheng, Y., Tomasi, C.: Linear time offline tracking and lower envelope algorithms. In: ICCV. (2011)
13. Uchida, S., Fujimura, I., Kawano, H., Feng, Y.: Analytical dynamic programming tracker. In: ACCV. (2011)
14. Wei, Y., Sun, J., Tang, X., Shum, H.Y.: Interactive offline tracking for color objects. In: ICCV. (2007)
15. Sun, J., Zhang, W., Tang, X., yeung Shum, H.: Bi-directional tracking using trajectory segment analysis. In: ICCV. (2005)
16. Tu, Z., Zhu, S.C.: Image segmentation by data-driven markov chain monte carlo. TPAMI **24** (2002) 657–673
17. Kwon, J., Lee, K.M.: A unified framework for event summarization and rare event detection. In: CVPR. (2012)
18. Korman, S., Avidan, S.: Coherency sensitive hashing. In: ICCV. (2011)
19. Leibe, B., Leonardis, A., Schiele, B.: Combined object categorization and segmentation with an implicit shape model. In: ECCV workshop on statistical learning in computer vision. (2004)
20. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290** (2000)
21. Han, B., Hamm, J., Sim, J.: Personalized video summarization with human in the loop. In: WACV. (2011)
22. Ham, J., Lee, D., Saul, L.: Semisupervised alignment of manifolds. In: 10th International Workshop on Artificial Intelligence and Statistics. (2005)
23. Kwak, S., Nam, W., Han, B., Han, J.H.: Learning occlusion with likelihoods for visual tracking. In: ICCV. (2011)
24. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: CVPR. (2013)
25. Liu, B., Huang, J., Yang, L., Kulikowski, C.A.: Robust tracking using local sparse appearance model and k-selection. In: CVPR. (2011) 1313–1320

26. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: ECCV. (2012)
27. Sevilla-Lara, L., Learned-Miller, E.: Distribution fields for tracking. In: CVPR. (2012)
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. Adam, A., Rivlin, E., Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In: CVPR. (2006)
31. Dinh, T.B., Vo, N., Medioni, G.: Context tracker: Exploring supporters and distracters in unconstrained environments. In: CVPR. (2011)
32. Jia, X., Lu, H., Yang, M.H.: Visual tracking via adaptive structural local sparse appearance model. In: CVPR. (2012)
33. Hare, S., Saffari, A., Torr, P.H.S.: Struck: Structured output tracking with kernels. In: ICCV. (2011)