Orderless Tracking through Model-Averaged Posterior Estimation*

Seunghoon Hong Suha Kwak Bohyung Han
Department of Computer Science and Engineering, POSTECH, Korea
{maga33,mercury3,bhhan}@postech.ac.kr

Abstract

We propose a novel offline tracking algorithm based on model-averaged posterior estimation through patch matching across frames. Contrary to existing online and offline tracking methods, our algorithm is not based on temporally-ordered estimates of target state but attempts to select easy-to-track frames first out of the remaining ones without exploiting temporal coherency of target. The posterior of the selected frame is estimated by propagating densities from the already tracked frames in a recursive manner. The density propagation across frames is implemented by an efficient patch matching technique, which is useful for our algorithm since it does not require motion smoothness assumption. Also, we present a hierarchical approach, where a small set of key frames are tracked first and non-key frames are handled by local key frames. Our tracking algorithm is conceptually well-suited for the sequences with abrupt motion, shot changes, and occlusion. We compare our tracking algorithm with existing techniques in real videos with such challenges and illustrate its superior performance qualitatively and quantitatively.

1. Introduction

There is a tremendous amount of archived videos in local hard drive, file repositories, and video sharing websites. Offline tracker is a reasonable option for tracking objects in such videos since more robust tracking results can be obtained by utilizing observations from the multiple frames regardless of their temporal order. However, most of existing online (and even most of offline) tracking algorithms are limited to processing frames in a temporal order. Note that tracking algorithms often fail eventually because a few intermediate frames are extremely challenging due to fast motion, shot changes, occlusion, shadow, and temporary appearance changes. Therefore, we claim that tracking in a non-temporal order improve performance by setting aside troublesome frames and taking easy-to-track ones first.

Recent studies about visual tracking problem are mainly focused on robust appearance modeling [2, 3, 7, 10, 17, 18, 24, 25]; they handle the variations of target appearance online by sparse reconstruction [3, 10, 17, 24, 25], incremental subspace learning [18], multiple instance learning [2], P-N learning [11], and so on. While these methods are successful in handling various appearance changes, they all assume temporal smoothness of target motion; they often fail to track objects in the presence of sudden changes in target and scene. To overcome the challenge related to abrupt motion of target, [14] adopts the Wang-Landau Monte Carlo sampling method. It successfully tracks objects even with sudden changes of target location including shot changes in an online manner. However, it may not be able to recover from temporal failures as other online tracking algorithms, and may not be sufficiently robust to other kinds of challenges such as occlusion, background clutter, and appearance changes.

Offline tracking [4, 6, 20, 22, 23] is an alternative option to handle abrupt motion, occlusion, and shot changes robustly since it can utilize entire frames within video at once. Uchida *et al*. [22] formulates offline tracking as a global optimization problem, and solves it by dynamic programming efficiently. This approach is further extended by [6], which adopts generalized distance transform for additional computational efficiency. Note that dynamic programming estimates target state at each frame recursively, and still follows a predefined order, typically temporal order, of a sequence; the benefit of offline tracking is limited in practice. On the other hand, [20] proposes a bi-directional tracking algorithm, where a full trajectory of target is obtained by connecting a set of short trajectories with occlusion handling through the optimization with a discrete Hidden Markov Model (HMM). Some offline algorithms integrate user interactions [4, 23], where dynamic programming or k-d tree is employed in the optimization process.

We propose an offline tracking algorithm based on model-averaged posterior estimation. Although our tracking algorithm is not based on a temporal order, it actively finds the next frame to track out of the remaining ones and

*This work was supported by MEST Basic Science Research Program through the NRF of Korea (NRF-2012R1A1A1043658), the IT R&D Program of MKE/KEIT (10040246), and Samsung Electronics Co., Ltd.

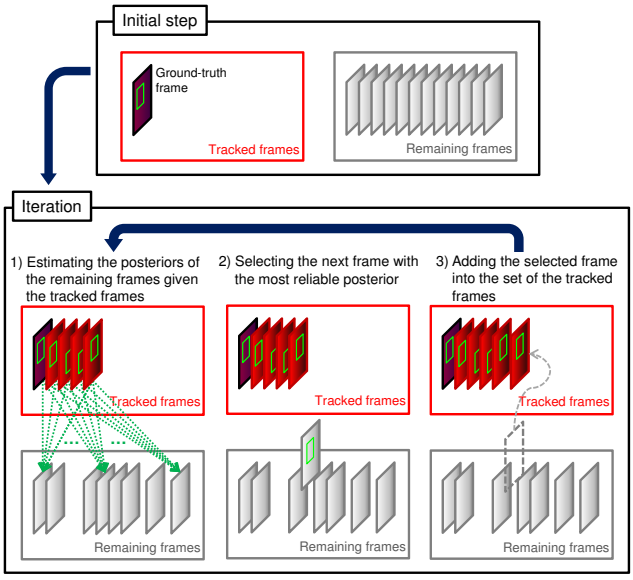


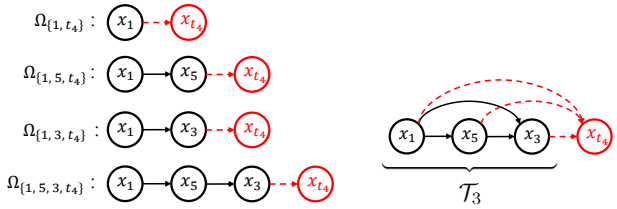
Figure 1. Main framework of our algorithm

estimates its posterior sequentially by a variant of sequential Bayesian filtering. The posterior is represented with a mixture model—mixture of potential tracking orders—and estimated by a weighted sum of multiple posteriors corresponding to the models. The observation in each frame is performed by a patch matching through hashing, which is appropriate for computing likelihoods without temporal smoothness assumption. Additionally, we present a hierarchical key frame based tracking algorithm, which exploits the temporal unorderedness of our algorithm and reduces computational cost significantly. The main framework is illustrated in Figure 1.

The characteristics and benefits of our tracking algorithm are summarized below:

- Our tracking algorithm does not have any temporal smoothness assumption, and is conceptually more robust to abrupt motion, occlusion, and shot changes of target than existing techniques.
- We design a mixture of sequential—not necessarily temporally ordered—Bayesian filters in a principled way to support the correctness and efficient implementation of our tracking algorithm.
- A hierarchical tracking approach is proposed for further efficiency, where a small number of key frames are tracked first and non-key frames are handled by nearby key frames.

The rest of the paper is organized as follows. We first describe overall framework of our algorithm in Section 2. The probabilistic framework of our algorithm is discussed in Section 3, and hierarchical extension of our algorithm is described in Section 4.



(a) (b)

Figure 2. An example of possible chain models when $k = 4$ and $\mathcal{T}_3 = \{1, 5, 3\}$. Suppose that frame t_4 is selected for tracking in the 4th time step. (a) There are four possible ways to reach the frame t_4 from the initial frame. (b) Tracking result of the frame t_4 is determined by average estimate of the four chain models.

2. Algorithm Overview

Let $\mathbf{x}_i$ be the target state in the ith frame. Our objective is to estimate the posterior density functions $P(\mathbf{x}_i)$ for all frames in a sequential but non-temporal order one-by-one in a greedy manner, where the next frame is selected for tracking based on the uncertainty score of each $P(\mathbf{x}_i)$. Let $\mathcal{F} = \{1, \ldots, N\}$ be a set of frame indices. In the kth time step of our algorithm, $\mathcal{F}$ is given by the union of two disjoint sets as $F = \mathcal{T}_k \cup \mathcal{R}_k$, where $\mathcal{T}_k = \{t_1, \ldots, t_k\}$ is a set of *tracked frames* sorted in the tracked order, and $\mathcal{R}_k = F \setminus \mathcal{T}_k = \{r_1, \ldots, r_{N-k}\}$ is a set of *remaining frames*.

At the time step $k + 1$, where k frames are tracked, our tracking algorithm performs the following procedure:

1. Given target posterior density functions of all tracked frames $P(\mathbf{x}_t)$, $\forall t \in \mathcal{T}_k$, it estimates the intermediate posterior density functions of all remaining frames $P(\mathbf{x}_i)$, $\forall i \in \mathcal{R}_k$, recursively.
2. Evaluate the uncertainty of each $P(\mathbf{x}_i)$, $\forall i \in \mathcal{R}_k$
3. Select a frame t_{k+1} with minimum uncertainty, and set $\mathcal{T}_{k+1} = \mathcal{T}_k \cup \{t_{k+1}\}$, and $\mathcal{R}_{k+1} = \mathcal{R}_k \setminus \{t_{k+1}\}$.

Note that the new element t_{k+1} in $\mathcal{T}_{k+1}$ also updates the posterior densities of remaining frames from the next iteration. Once a frame is inserted into the tracked list $\mathcal{T}_k$, corresponding target posterior density would not change any more. Our algorithm is terminated when $\mathcal{T}_N = \mathcal{F}$ and $\mathcal{R}_N = \emptyset$, which indicates that we tracked all frames.

3. Probabilistic Framework of Our Algorithm

We present the main probabilistic framework of our algorithm; we describe our posterior estimation technique called model-averaged posterior estimation and discuss how it is related to patch-based observation technique.

3.1. Model-Averaged Posterior Estimation

We first describe how target posterior density functions are estimated for the remaining frames given $k - 1$ tracked

frames and their corresponding posterior density functions. Since our algorithm is not limited to temporally ordered estimation of posterior density functions, we employ a novel Bayesian formulation to handle a tracking scenario with an arbitrary sequence of frames.

If the temporal order of frames is ignored, there are a number of possible ways to reach the kth tracking frame, t_k, from the tracked $k-1$ frames as illustrated in Figure 2(a). We can take any sequence generated by any subset of frames in $\mathcal{T}_{k-1}$[1] as intermediate hops. Note that each potential path from t_1 to t_k is modeled by the first order Markov chain. We call each path *chain model*, which is denoted by $\Omega_{\mathcal{S}}$, $\mathcal{S} \subseteq \mathcal{T}_k$.

Choosing a chain model out of all possible ones is not straightforward since it is often uncertain which model is good for tracking. So, tracking by a single model is risky especially when there are critical challenges in the model such as abrupt motion and shot changes. For this reason, we adopt a model averaging strategy [9]. That is, tracking result of a frame is determined by average estimate of all possible chain models instead of choosing one, as illustrated in Figure 2(b).

Let t_k be the frame index selected for tracking in the kth time step. In principle, to estimate the posterior of $\mathbf{x}_{t_k}$, we need to enumerate all chain models whose last nodes are t_k and calculate the average of their posteriors. We can estimate the posterior of $\mathbf{x}_{t_k}$ in a simple and recursive fashion, which is formally given by

$$\widetilde{P}(\mathbf{x}_{t_k}) \propto \frac{1}{k-1} \sum_{t \in \mathcal{T}_{k-1}} P(Z_{t_k}|\mathbf{x}_{t_k}) \int P(\mathbf{x}_{t_k}|\mathbf{x}_t)\widetilde{P}(\mathbf{x}_t)d\mathbf{x}_t, \tag{1}$$

where t denotes a frame tracked before the kth time step, and Z_{t_k} is an observation variable of frame t_k. The posterior probabilities are denoted by $\widetilde{P}$ without observation variables for notational simplicity. The above equation is similar to an ordinary sequential Bayesian filtering equation except that we marginalize out all possible models rather than choose one. Note that all chain models arriving at frame t are already averaged when calculating $\widetilde{P}(\mathbf{x}_t)$. Once frame t_k is selected and tracked, $\widetilde{P}(\mathbf{x}_{t_k})$ will not change any more, so that we can reuse it for tracking in the future without updating it. Therefore, we can marginalize all possible chain models recursively, which makes our algorithm efficient. In the followings, we show how to derive Eq. (1) in detail.

Let $\mathcal{S}_t$ be a subset of $\mathcal{T}_{k-1}$ whose last element is $t \in \mathcal{T}_{k-1}$. By Bayesian model averaging strategy, the posterior of $\mathbf{x}_t$ is given by

$$\widetilde{P}(\mathbf{x}_t) = \sum_{\mathcal{S}_t \subseteq \mathcal{T}_{k-1}} P\big(\mathbf{x}_t|\Omega_{\mathcal{S}_t}, Z_{\mathcal{S}_t}\big)P(\Omega_{\mathcal{S}_t}), \tag{2}$$

[1]Since $\mathcal{T}_k$ is an ordered set, only one sequence is generated from each subset of $\mathcal{T}_k$.

where $Z_{\mathcal{S}_t}$ is an observation variable corresponding to the frames in $\mathcal{S}_t$, and $\Omega_{\mathcal{S}_t}$ denotes a candidate chain model given by $\mathcal{S}_t$. $P(\Omega_{\mathcal{S}_t})$ is the model prior corresponding to $\mathcal{S}_t$. The chain model is defined recursively by

$$\Omega_{\mathcal{S}_t \cup \{t_k\}} = \langle \Omega_{\mathcal{S}_t}, \mathrm{p}_{t \to t_k} \rangle, \tag{3}$$

where $\mathrm{p}_{t \to t_k}$ denotes the path from the last node in $\Omega_{\mathcal{S}_t}$ to t_k and $\langle \cdot, \cdot \rangle$ the concatenation operator of two paths. Note that $P(\langle \Omega_{\mathcal{S}_t}, \mathrm{p}_{t \to t_k} \rangle) = P(\Omega_{\mathcal{S}_t})P(\mathrm{p}_{t \to t_k})$ holds probabilistically. Then, $\widetilde{P}(\mathbf{x}_{t_k})$ is given recursively from Eq. (2) by

$$\begin{aligned}
&\widetilde{P}(\mathbf{x}_{t_k}) \\
&= \sum_{t \in \mathcal{T}_{k-1}} \sum_{\mathcal{S}_t \subseteq \mathcal{T}_{k-1}} P\big(\mathbf{x}_{t_k}|\Omega_{\mathcal{S}_t \cup \{t_k\}}, Z_{\mathcal{S}_t \cup \{t_k\}}\big)P(\Omega_{\mathcal{S}_t \cup \{t_k\}}) \\
&= \sum_{t \in \mathcal{T}_{k-1}} P(\mathrm{p}_{t \to t_k}) \sum_{\mathcal{S}_t \subseteq \mathcal{T}_{k-1}} P\big(\mathbf{x}_{t_k}|\Omega_{\mathcal{S}_t \cup \{t_k\}}, Z_{\mathcal{S}_t \cup \{t_k\}}\big)P(\Omega_{\mathcal{S}_t}).
\end{aligned} \tag{4}$$

Because each chain is modeled by the first order Markov chain, we can use Bayes theorem as follows:

$$\begin{aligned}
&P\big(\mathbf{x}_{t_k}|\Omega_{\mathcal{S}_t \cup \{t_k\}}, Z_{\mathcal{S}_t \cup \{t_k\}}\big) \\
&\propto \int P(Z_{t_k}|\mathbf{x}_{t_k})P(\mathbf{x}_{t_k}|\mathbf{x}_t)P\big(\mathbf{x}_t|\Omega_{\mathcal{S}_t}, Z_{\mathcal{S}_t}\big)d\mathbf{x}_t.
\end{aligned} \tag{5}$$

By Eq. (5), we rewrite Eq. (4) as

$$\begin{aligned}
\widetilde{P}(\mathbf{x}_{t_k}) &\propto \sum_{t \in \mathcal{T}_{k-1}} P(\mathrm{p}_{t \to t_k}) \sum_{\mathcal{S}_t \subseteq \mathcal{T}_{k-1}} P(\Omega_{\mathcal{S}_t}) \times \\
&\quad \left\{ \int P(Z_{t_k}|\mathbf{x}_{t_k})P(\mathbf{x}_{t_k}|\mathbf{x}_t)P\big(\mathbf{x}_t|\Omega_{\mathcal{S}_t}, Z_{\mathcal{S}_t}\big)d\mathbf{x}_t \right\}, \\
&= \sum_{t \in \mathcal{T}_{k-1}} P(\mathrm{p}_{t \to t_k}) \int P(Z_{t_k}|\mathbf{x}_{t_k})P(\mathbf{x}_{t_k}|\mathbf{x}_t) \times \\
&\quad \underline{\left\{ \sum_{\mathcal{S}_t \subseteq \mathcal{T}_{k-1}} P(\Omega_{\mathcal{S}_t})P\big(\mathbf{x}_t|\Omega_{\mathcal{S}_t}, Z_{\mathcal{S}_t}\big) \right\}} d\mathbf{x}_t.
\end{aligned} \tag{6}$$

Since the underlined term in Eq. (6) is identical to Eq. (2),

$$\begin{aligned}
&\widetilde{P}(\mathbf{x}_{t_k}) \propto \\
&\sum_{t \in \mathcal{T}_{k-1}} P(\mathrm{p}_{t \to t_k})P(Z_{t_k}|\mathbf{x}_{t_k}) \int P(\mathbf{x}_{t_k}|\mathbf{x}_t)\widetilde{P}(\mathbf{x}_t)d\mathbf{x}_t,
\end{aligned} \tag{7}$$

where the posterior density function for the kth tracking frame is now defined recursively. The prior of the path, $P(\mathrm{p}_{t \to t_k})$, represents which path would be preferred to tracking frame t_k, and is simply given by

$$P(\mathrm{p}_{t \to t_k}) \propto \frac{1}{k-1}, \quad t \in \mathcal{T}_{k-1}. \tag{8}$$

Finally, we can obtain Eq. (1) by substituting Eq. (8) to the path prior of Eq. (7).

3.2. Density propagation

The integration in Eq. (1) corresponds to density propagation process; given the target density at frame $t \in \mathcal{T}_{k-1}$, we want to estimate $\widetilde{P}(\mathbf{x}_{t_k})$ through prediction and update steps, $P(\mathbf{x}_{t_k}|\mathbf{x}_t)$ and $P(Z_{t_k}|\mathbf{x}_{t_k})$, respectively. One important property of our algorithm is that we do not assume any spatio-temporal coherency of target between t and t_k. Because the integral in continuous domain in Eq. (1) is infeasible, we approximate it by sampling as

$$\widetilde{P}(\mathbf{x}_{t_k}) \approx \frac{1}{k-1} \sum_{t \in \mathcal{T}_{k-1}} \sum_{\mathbf{x}_t^i \in \mathbb{S}_t} P(Z_{t_k}|\mathbf{x}_{t_k}) P(\mathbf{x}_{t_k}|\mathbf{x}_t^i), \quad (9)$$

where $\mathbb{S}_t$ denotes a set of samples drawn from $\widetilde{P}(\mathbf{x}_t)$.

The transition model and likelihood are jointly defined by patch matching and voting process. Let $\mathbf{c}_t$ denote the center of a patch in I_t, image at frame t. Suppose that we match all patches in I_t to I_{t_k} by a function f_{PM}. For a set of patches within target window corresponding to each sample $\mathbf{x}_t^i$ in I_t, we obtain a voting result with respect to I_{t_k} as

$$V(\mathbf{x}_{t_k}; \mathbf{x}_t^i) = \sum_{j=1}^{K_t^i} \mathcal{N}(\mathbf{x}_{t_k}; f_{PM}(\mathbf{c}_t^j) - \mathbf{a}_t^j, \mathbf{\Sigma}), \quad (10)$$

where $\mathbf{c}_t^j$ is the center position of the jth patch within the target bounding box centered at $\mathbf{x}_t^i$, K_t^i is the number of patches within the bounding box, and $\mathbf{a}_t^j$ is the offset from $\mathbf{c}_t^j$ to $\mathbf{x}_t^i$. Then, we use Eq. (10) to rewrite Eq. (9) as

$$\widetilde{P}(\mathbf{x}_{t_k}) \approx \quad (11)$$
$$\frac{1}{k-1} \sum_{t \in \mathcal{T}_{k-1}} \sum_{\mathbf{x}_t^i \in \mathbb{S}_t} \sum_{j=1}^{K_t^i} \mathcal{N}(\mathbf{x}_{t_k}; f_{PM}(\mathbf{c}_t^j) - \mathbf{a}_t^j, \mathbf{\Sigma}).$$

Note that our algorithm has no predefined appearance model. Since it relies only on patch matching and voting between frames, a density propagated from a single frame may cause drift problem. In our algorithm, however, the problem can be alleviated by aggregating all the propagated densities through Bayesian model averaging.

As the matching function f_{PM}, we adopt Coherency Sensitive Hashing (CSH) [12], which is fast and efficient. To handle multiple scales, we generate multiple voting maps with several different offsets $\mathbf{a}_t^j$. Matching and voting by CSH has several good properties as follows:

1. CSH searches entire image area with very low computational cost, and it is natural to effectively handle abrupt motion, shot changes, and occlusion of target without temporal smoothness assumption.

2. Our patch-based voting algorithm is robust to local changes of target appearance such as partial occlusion and moderate non-rigid transformation.

3. Although we need to consider multiple hypotheses and scales for the computation of the posterior in Eq. (11), matching between a pair of frames are to be computed at most once throughout tracking.

Note that the choice of density propagation technique in this work is orthogonal to our model-averaged posterior estimation framework for offline tracking. For example, it is possible to use [14] for this step.

3.3. Identifying Subsequent Frame

Until now, we have assumed that the newly selected frame in each time step is given. We now describe how to determine the next frame t_k out of $\mathcal{R}_{k-1}$ based on the uncertainty analysis for the rest of frames, which corresponds to the second and third steps of Section 2.

Tracking result in a frame is likely to be reliable if its posterior density function has a clear mode, and we measure the uncertainty using entropy. To calculate the entropy of the target density, we first divide the state space into M regular grid blocks, B^m $(m = 1, \ldots, M)$. For each frame $r \in \mathcal{R}_{k-1}$, we compute the marginalized posterior probability of each block, which is given by

$$P(B_r^m) = \sum_{\mathbf{x}_r \in B_r^m} \widetilde{P}(\mathbf{x}_r), \quad (12)$$

and obtain an M-dimensional vector. Note that $\widetilde{P}(\mathbf{x}_r)$ is estimated by applying Eq. (11) to frame r. We perform the same procedure for different scales, concatenate the probability vectors, and normalize the concatenated vector together. Then, entropy $\mathcal{H}$ for $\mathbf{x}_r$ is given by

$$\mathcal{H}(\mathbf{x}_r) = \sum_s \sum_{m=1}^{M} -P(B_r^{m,s}) \log P(B_r^{m,s}), \quad (13)$$

where s denotes scale index. We compute the entropy for every frame in $\mathcal{R}_{k-1}$ and choose the frame with minimum entropy as

$$t_k = \arg\min_r \mathcal{H}(\mathbf{x}_r), \quad r \in \mathcal{R}_{k-1}. \quad (14)$$

By the above criterion, our algorithm tends to select easy-to-track frames first regardless of their temporal order, and it helps to prevent the entire track from being corrupted by few tracking failures. After frame selection, we update the sets for tracked and remaining frames by $\mathcal{T}_k = \mathcal{T}_{k-1} \cup \{t_k\}$ and $\mathcal{R}_k = \mathcal{R}_{k-1} \setminus \{t_k\}$. Once a frame is inserted into $\mathcal{T}_k$, its posterior does not change during remaining iterations any more.

For the state estimation of target, the location of target for each scale is obtained first by

$$X_{t_k}^* = \arg\max_{\mathbf{x}_{t_k}} \widetilde{P}(\mathbf{x}_{t_k}), \quad (15)$$

and we choose the scale with maximum probability.

4. Efficient Hierarchical Approach

A drawback of the proposed algorithm is that the computational cost of the proposed algorithm increases quadratically with respect to the length of video because it needs to propagate the density function $\binom{N}{2}$ times by Eq. (11). Contrary to most of tracking algorithm, our techniques does not rely on any temporal or spatial coherency of target and it is reasonable to track a subset of frames first and estimate the posteriors of the rest of frames based only on the tracked frames. Although this idea does not improve theoretical time complexity, it reduces empirical processing time significantly. This section describes the details about this hierarchical approach[2].

4.1. Key Frame Selection

Key frames should capture important characteristics of entire video, especially in case that the video contains a lot of variations inside such as shot changes, fast motion, and occlusion. We employ a similar idea in [8] to identify key frames from an input video.

Our key frame selection technique first embeds all frames in a metric space, and selects a subset of frames by solving a metric facility location problem. For the purpose, we compute dissimilarities between all pairs of frames based on the bidirectional similarity [19], and construct an $N \times N$ dissimilarity matrix, which is given by

$$\mathbf{D}(I_1, I_2) = \frac{1}{n_1} \sum_{P \in I_1} \min_{Q \in I_2} d(P, Q) + \frac{1}{n_2} \sum_{Q \in I_2} \min_{P \in I_1} d(Q, P), \quad (16)$$

where n_1 and n_2 indicates the number of patches in I_1 and I_2, respectively, and P and Q denote patches in I_1 and I_2, respectively. To identify matching patches by minimizing distance between patches, $d(P, Q)$ or $d(Q, P)$, we also adopt the same patch matching algorithm [12].

Given the dissimilarity matrix $\mathbf{D}$, all frames can be embedded in a metric space by a non-linear manifold learning technique, and we employ Isomap [21] algorithm. Note that original dissimilarities (distances) between frames are preserved maximally through the manifold embedding; if the distance between two frames is small, they are likely to be located in a neighborhood.

To select κ most representative frames, we solve the κ-center problem [16] in the metric space. The κ-center problem is NP-hard, but a simple and efficient 2-approximation algorithm—Gonzalez's algorithm [16]—is well known. We find a subset of frames $\mathcal{K} \subseteq \mathcal{F}$, where $|\mathcal{K}| = \kappa$, based on the following objective function:

$$\mathcal{K}^* = \arg \min_{\mathcal{K} \subseteq \mathcal{F}} \max_{v \in \mathcal{F}} \min_{u \in \mathcal{K}} d_E(u, v) \quad (17)$$

[2]Sequence partitioning is another idea to make our algorithm much faster, but we focus on this hierarchical approach in this paper.

where $\mathcal{F}$ denotes the entire set of frames and d_E is the distance in the embedded space. The selected frames by solving κ-center problem serve as anchor frames to the rest of frames in the local area in the embedded space. Therefore, it typically captures important variations in the input video. We set κ to 10% of an input video length.

4.2. Density Propagation to Non-Key Frames

After selecting key frames by the method described in Section 4.1, we perform the inference for the posterior of the key frames based on the procedure presented in Section 3. To propagate the density functions estimated in the key frames to the non-key frames, we exploit the embedding result as described below.

1. Compute the distance in the embedded space between the frames, $d_E(u, v)$, where $v \in \mathcal{K}$ and $u \in \mathcal{F} \setminus \mathcal{K}$.
2. Convert the distance into similarity by $w_{uv} = \exp\left(-\alpha \cdot d_E(u, v)\right)$, and normalize it by $\widetilde{w}_{uv} = \frac{w_{uv}}{\sum_v w_{uv}}$.
3. For each frame $u \in \mathcal{F} \setminus \mathcal{K}$, compute the posterior of each frame by a single hop density propagation, which is given by

$$\widetilde{P}(\mathbf{x}_u) = \sum_{v \subseteq \mathcal{K}} P\left(\mathbf{x}_u | \mathrm{p}_{v \to u}, Z_{\mathcal{K}}\right) \cdot \widetilde{w}_{uv}. \quad (18)$$

In step 2, we discard the key frames with negligible weights by setting their weights to 0, and re-normalize weights. As a result, only a few frames—typically less than 5—involved in the computation of Eq. (18). Note that all models are based on single hops and we replace $P(\mathrm{p}_{v \to u})$ by $\widetilde{w}_{uv}$. By Eq. (18), we give more weights on the closer key frames to estimate the posterior for each frame in $\mathcal{F} \setminus \mathcal{K}$.

5. Experiment

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 sequences.

5.1. Datasets

For the evaluation of our tracking algorithm, we collected 10 video sequences; 7 out of 10 sequences—*animal* from [15], *tennis, boxing, youngki* from [14], and *TUD, campus, accident* from [13]—are well known to tracking community, and the rest—*skating, psy*, and *dance*—are downloaded from *YouTube*.

All the sequences involve at least one critical challenges; *animal* has fast motion and motion blur, *tennis* is with abrupt location changes and pose variations, *TUD, campus*, *accident* contain severe occlusions, and the others involve shot changes and pose variations. Additionally, *psy* has a lot of lighting condition changes.

1-13-42-50-64-72-81-102-92-123-35-29

(a) *TUD* sequence

1-17-6-4-3-5-16-13-14-26-27-21-19-28-20-24-8-29-22-30

(b) *tennis* sequence

1-246-647-638-57-40-22-2-753-770-795-603-526-521-268-340-288-297-501-503

(c) *boxing* sequence

Figure 3. Found tracking order. We present the subsets of target windows by their tracked orders. Numbers below each sequence correspond to frame indices. Note that tracked order is not consistent with temporal order; we can observe that the proposed algorithm tends to track easy frames first.

5.2. Found Tracking Order

We first present ordered tracked list $\mathcal{T}$ that our tracker automatically finds during tracking. In each time step k, our tracker analyzes all the frames in $\mathcal{R}_{k-1}$ to decide next move; it measures uncertainty of every frame, and add the most confident one into $\mathcal{T}_k$. It tends to choose frames in an increasing order of difficulty, as illustrated in Figure 3, which may not be same with temporal order. Note that, in Figure 3(a) and 3(c), visually similar frames to the initial frame have been selected at the beginning even with very different temporal locations. Also, Figure 3(b) shows that targets in different poses and relatively low resolutions compared to the initial target are typically selected later.

5.3. Quantitative and Qualitative Performance

We compared our algorithm with the state-of-the-art tracking methods, which include $L1$ [17], $L1$-APG [3], SCM [25], ASLSA [10], MTT [24], MIL [2], IVT [18], FRAG [1], WLMC [14], and OTLE [6]. Most of them

Figure 5. Examples of tracking failure

are online trackers except OTLE, and WLMC is a specialized technique to handle abrupt motion of target; these two methods are more related to the proposed algorithm. We also compare our algorithm with its sequential version[3] to show the advantages of orderless approach. Our algorithm and its sequential version are denoted by OMA (orderless model-averaged) and SMA (sequential model-averaged), respectively. To evaluate performance, we use two common measures—center location error and bounding box overlap ratio [5]. For fair comparison, we used downloaded source codes with default setting but made target state space of all methods identical to ours by eliminating affine transform and non-uniform scaling. In our algorithm, the patch size is 8×8, 9 scales are used from 0.6 to 1.4, and the number of samples to populate hypotheses to other frames is 900.

As observed in Table 1 and 2, our algorithm performs very well compared to other methods, especially in the challenging sequences with shot changes. As expected, the performance of WLMC is comparable to ours in the sequences, but it does not work well for the sequences with other challenges. It is probably because the algorithm is specialized for the sudden location changes of target but is not good enough to handle other variations such as occlusion in *TUD* and *campus* and background clutter in *animal*. The offline tracking algorithm, OTLE, is generally worse than ours. Other trackers have significant troubles to handle shot changes and abrupt motion of target. The qualitative results of a subset of algorithms are illustrated in Figure 4.

Some examples of tracking failure in *psy* are presented in Figure 5. Although our algorithm fails in some frames due to severe deformation or lighting changes, error propagation to other frames is minor since our algorithm tends to postpone processing the failed frames and their influence is curbed by the model-averaged posterior estimation.

6. Conclusion

We presented a novel offline tracking algorithm based on model-averaged density estimation, where the posterior of a newly selected frame for tracking is estimated by a weighted mixture model. In other words, this strategy hypothesizes all the linear first-order Markov chains available from tracked frames, and estimates the target density of the new frame by marginalizing the densities of all chain mod-

[3]The sequential version is identical to the proposed algorithm except that it tracks both key and non-key frames in their temporal order.

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

	IVT	MIL	SCM	L1APG	MTT	ASLSA	L1	FRAG	WLMC	OTLE	OMA	SMA
animal	10.6	32.0	16.6	48.8	12.6	179.6	164.9	94.1	64.8	19.4	7.7	7.4
TUD	12.6	57.1	12.2	7.4	37.2	67.2	64.7	17.3	68.2	27.4	4.4	5.9
campus	38.7	37.1	12.2	16.1	6.0	12.2	68.4	3.3	13.5	5.8	3.2	7.0
accident	27.6	24.8	3.0	20.3	21.9	2.9	32.4	7.4	12.2	9.1	2.6	6.5
tennis	68.7	74.4	65.9	85.0	65.6	68.8	111.4	67.4	31.0	37.0	6.9	11.9
boxing	128.1	88.9	96.0	117.6	87.0	106.8	103.5	80.0	11.7	41.7	10.5	22.6
youngki	95.2	115.2	115	137.9	176.5	151.8	121.8	97.5	16.0	15.7	11.4	14.0
skating	77.8	15.0	49.4	143.9	100.4	22.8	72	35.4	14.7	18.3	8.0	10.8
psy	156.5	220.6	213.3	71.8	117.8	146.8	124.6	95.2	66.0	61.2	15.0	21.9
dance	283.9	169.4	208.0	113.9	133.4	118.1	143.1	132.4	39.7	118.8	15.1	19.7

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

	IVT	MIL	SCM	L1APG	MTT	ASLSA	L1	FRAG	WLMC	OTLE	OMA	SMA
animal	0.60	0.42	0.55	0.4	0.57	0.04	0.04	0.08	0.31	0.48	0.71	0.71
TUD	0.65	0.34	0.67	0.85	0.52	0.32	0.62	0.59	0.38	0.48	0.82	0.75
campus	0.56	0.45	0.62	0.52	0.76	0.63	0.01	0.77	0.52	0.72	0.78	0.67
accident	0.58	0.53	0.87	0.69	0.69	0.84	0.45	0.60	0.57	0.59	0.85	0.76
tennis	0.06	0.20	0.11	0.29	0.11	0.12	0.03	0.11	0.43	0.31	0.63	0.56
boxing	0.05	0.06	0.13	0.13	0.06	0.11	0.16	0.22	0.65	0.38	0.70	0.51
youngki	0.09	0.13	0.13	0.02	0.10	0.06	0.02	0.19	0.62	0.54	0.62	0.54
skating	0.01	0.41	0.20	0.02	0.03	0.29	0.06	0.25	0.46	0.41	0.42	0.37
psy	0.07	0.08	0.07	0.02	0.23	0.17	0.25	0.23	0.39	0.40	0.63	0.57
dance	0.03	0.07	0.07	0.10	0.10	0.11	0.11	0.14	0.45	0.30	0.52	0.50

els. Our tracking algorithm is free from temporal smoothness assumption, and tends to choose easy-to-track frames first and challenging frames last. So, it is conceptually robust to various challenges such as abrupt motion, occlusion, and shot changes. To handle observations efficiently without temporal coherency across frames, a patch matching technique by hashing is employed. We evaluated the performance of our algorithm qualitatively and quantitatively, and compared with the state-of-the-art tracking algorithms.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, 2006. 6

[2] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *TPAMI*, 33(8), 2011. 1, 6

[3] C. Bao, Y. Wu, H. Ling, and H. Ji. Real time robust l1 tracker using accelerated proximal gradient approach. In *CVPR*, 2012. 1, 6

[4] A. M. Buchanan and A. W. Fitzgibbon. Interactive feature tracking using K-D trees and dynamic programming. In *CVPR*, 2006. 1

[5] M. Everingham, L. van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL visual object classes (VOC) challenge. *IJCV*, 88(2), 2010. 6

[6] S. Gu, Y. Zheng, and C. Tomasi. Linear time offline tracking and lower envelope algorithms. In *ICCV*, 2011. 1, 6

[7] B. Han, D. Comaniciu, Y. Zhu, and L. Davis. Sequential kernel density approximation and its application to real-time visual tracking. *TPAMI*, 30(7), 2008. 1

[8] B. Han, J. Hamm, and J. Sim. Personalized video summarization with human in the loop. In *WACV*, 2011. 5

[9] J. A. Hoeting, D. Madigan, A. E. Raftery, and C. T. Volinsky. Bayesian model averaging: A tutorial. *Statistical Science*, 14(4), 1999. 3

[10] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012. 1, 6

[11] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-Learning-Detection. *TPAMI*, 2012. 1

[12] S. Korman and S. Avidan. Coherency sensitive hashing. In *ICCV*, 2011. 4, 5

[13] S. Kwak, W. Nam, B. Han, and J. H. Han. Learning occlusion with likelihoods for visual tracking. In *ICCV*, 2011. 5

[14] J. Kwon and K.-M. Lee. Tracking of abrupt motion using wang-landau monte carlo estimation. In *ECCV*, 2008. 1, 4, 5, 6

[15] J. Kwon and K.-M. Lee. Visual tracking decomposition. In *CVPR*, 2010. 5

[16] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38, 1985. 5

[17] X. Mei and H. Ling. Robust visual tracking using $l1$ minimization. In *ICCV*, 2009. 1, 6

[18] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3), 2008. 1, 6

[19] D. Simakov, Y. Caspi, E. Shechtman, and M. Irani. Summarizing visual data using bidirectional similarity. In *CVPR*, 2008. 5

[20] J. Sun, W. Zhang, X. Tang, and H. yeung Shum. Bi-directional tracking using trajectory segment analysis. In *ICCV*, 2005. 1

[21] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290, 2000. 5

[22] S. Uchida, I. Fujimura, H. Kawano, and Y. Feng. Analytical dynamic programming tracker. In *ACCV*, 2011. 1

[23] Y. Wei, J. Sun, X. Tang, and H.-Y. Shum. Interactive offline tracking for color objects. In *ICCV*, 2007. 1

[24] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Robust visual tracking via multi-task sparse learning. In *CVPR*, 2012. 1, 6

[25] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012. 1, 6

Figure 4. Tracking results for all sequences: (From top to bottom) *animal, TUD, campus, accident, tennis, boxing, youngki, skating, psy,* and *dance*