Tracking by Sampling Trackers

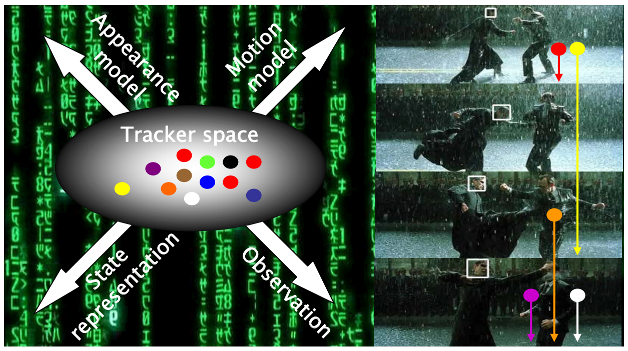
Junseok Kwon and Kyoung Mu Lee
Department of EECS, ASRI, Seoul National University, 151-742, Seoul, Korea
{paradis0, kyoungmu}@snu.ac.kr, http://cv.snu.ac.kr

Abstract

We propose a novel tracking framework called visual tracker sampler that tracks a target robustly by searching for the appropriate trackers in each frame. Since the real-world tracking environment varies severely over time, the trackers should be adapted or newly constructed depending on the current situation. To do this, our method obtains several samples of not only the states of the target but also the trackers themselves during the sampling process. The trackers are efficiently sampled using the Markov Chain Monte Carlo method from the predefined tracker space by proposing new appearance models, motion models, state representation types, and observation types, which are the basic important components of visual trackers. Then, the sampled trackers run in parallel and interact with each other while covering various target variations efficiently. The experiment demonstrates that our method tracks targets accurately and robustly in the real-world tracking environments and outperforms the state-of-the-art tracking methods.

1. Introduction

It is a challenging problem to track a target in the real-world tracking environment where different types of variations such as illumination, shape, occlusion, or motion changes occur at the same time [23]. Recently, several tracking methods solved the problem and successfully tracked targets in the real-world environment [2, 6, 8, 10, 11, 12, 14, 17, 18, 21]. Among them, one of promising methods is the visual tracking decomposition (VTD), which utilizes a set of multiple trackers and runs them simultaneously and interactively [11]. The method assumes that, given a fixed number of trackers, at least one tracker can deal with target variations at each time. However, this assumption is insufficient to cope with the complicated real-world tracking environment. Since generally the tracking environment severely varies from frame to frame, trackers should not be fixed but should be generated dynamically depending on the current tracking environment. This paper focuses on how to construct appropriate trackers automatically and how to integrate the constructed trackers to track the target successfully under challenging real scenarios.



(a) Tracker space (b) Tracking results

Figure 1. **Visual tracker sampler** (a) The figure describes our four-dimensional tracker space, in which the axes are the appearance model, motion model, state representation type, and observation type. A tracker is determined by sampling a point in the tracker space, where each circle represents a different tracker. (b) Our visual tracker sampler tracks the target robustly in the challenging *matrix* sequence by choosing appropriate trackers adaptively during the tracking process.

The philosophy of our method is that trackers can be constructed probabilistically. With a sampling method, the trackers themselves are sampled, as well as the states of the targets. In our framework, a sample includes information about not only a proposed state but also a proposed tracker. During the sampling process, our method obtains several trackers and states as the samples, and decides whether they are accepted or not. By choosing an accepted sample that gives the highest value on the Conditioned Maximum a Posteriori (CMAP) estimate, the method simultaneously finds a highly possible tracker and a highly possible state. The highly possible tracker indicates the best tracker that tracks the target robustly with high probability, while the highly possible state denotes the best state where the target might be. We exploit the complementary relationship between the states and the trackers, in which good states help construct more robust trackers. On the other hand, the robust trackers produce more accurate states. Fig.1 shows our visual tracker sampler (VTS).

Our method is superior to the conventional tracking methods in the following three aspects. The first is related to the novel tracking framework. Our method can change the number of trackers adaptively over time. If there are severe appearance or motion changes, the method increases the number of trackers and spends more resources to track the target. If not, the method decreases them and saves resources. This can be done since a tracker itself may be added or deleted by our VTS. By doing this, our method enhances the sampling efficiency compared with the conventional methods, which always utilize a fixed number of trackers or samples. The second is that our tracker is designed in a more sophisticated manner to describe the real-world environment completely. To design trackers, we fully consider four important ingredients of the Bayesian tracking approach, which are the appearance model, motion model, state representation type, and observation type. This makes the trackers robust against a wider range of variations including severe noise and motion blur. Additionally, our trackers evolve toward reflecting the characteristic of the target over time. The ingredients and parameters that consist of the trackers adaptively change during the tracking process by learning multiple cues in the video. So, our method greatly improves tracking accuracy in the real-world tracking environment. The last is a rigorous derivation of why utilizing multiple trackers provides a better tracking performance. We prove that, compared with any single tracker, it provides better average predictive ability, as measured by a logarithmic scoring rule, to construct multiple trackers and run them interactively.

2. Related Works

Among many approaches for the real-world tracking problem, Ross et al. [17] showed robustness to large changes in pose, scale, and illumination by proposing the incremental principal component analysis. Babenko et al. [2] solved appearance ambiguity occurred by illumination and occlusion using multiple instance learning and showed very good tracking results. Kwon et al. [11] tracked the target successfully via visual tracking decomposition (VTD) when several types of appearance and motion changes occur. Note that our foremost contribution is the novel concept of "SAMPLING the best TRACKERS adaptively from a TRACKER SPACE". To the best of our knowledge, this work is the first trial to define the tracker space and sample trackers directly in this space. Although VTD employs multiple trackers, the number and types of trackers are predefined by a user. On the other hand, our method can replace current trackers by new sampled trackers during the tracking process and change the total number of trackers, which run in parallel, by adding good trackers and removing bad or redundant ones. This cannot be accomplished by VTD.

In sampling-based tracking approaches, the particle filter developed by Isard et al. [7] showed good performance in tracking targets by solving the non-Gaussianity and multimodality of the tracking problem. Markov Chain Monte Carlo (MCMC) based methods were proposed by Khan et al. [9] and Zhao et al. [25] to reduce the computational cost especially in a high-dimensional state space. However, conventional sampling methods only considered the uncertainty of the target state given a fixed tracker. Our method is an intuitively attractive solution to the problem of accounting for the uncertainty of the tracker.

Multiple cues were used for tracking. Collins et al. [3] used multiple features and selected robust ones through an on-line feature-ranking mechanism to deal with changing appearances. Stenger et al. [19] used multiple observation models and fused them to track targets accurately. However, the methods only considered information related to the target appearance to improve the tracking performance. Compared with these methods, our method exploits useful information on the target motion and the target representation as well.

3. Our Bayesian Tracking Approach

3.1. Basic Ingredients of Bayesian Tracker

The visual tracking problem is efficiently formulated as Bayesian filtering. Given the state at time t and the observation up to time t, the Bayesian filter updates the posteriori probability, $p(\mathbf{X}_t|\mathbf{Y}_{1:t})$ with the following formula:

$$p(\mathbf{X}_t|\mathbf{Y}_{1:t}) \propto p(\mathbf{Y}_t|\mathbf{X}_t) \int p(\mathbf{X}_t|\mathbf{X}_{t-1}) p(\mathbf{X}_{t-1}|\mathbf{Y}_{1:t-1}) d\mathbf{X}_{t-1}, \tag{1}$$

where it consists of four important basic ingredients.

- **Appearance model (A):** $p(\mathbf{Y}_t|\mathbf{X}_t)$ describes the appearance of a target while measuring how much the target and observation at the proposed state coincide.
- **Motion model (M):** $p(\mathbf{X}_t|\mathbf{X}_{t-1})$ models the characteristic of the target motion by predicting the next state, $\mathbf{X}_t$ based on the previous state, $\mathbf{X}_{t-1}$.
- **State representation type (S):** $\mathbf{X}_t$ designs the configuration of the target called the state.
- **Observation type (O):** $\mathbf{Y}_t$ denotes visual cues in the video.

In our framework, each basic ingredient forms the set: $\mathbf{A}_t = \{\mathrm{A}_t^i | i = 1, \ldots, |\mathbf{A}_t|\}$, $\mathbf{M}_t = \{\mathrm{M}_t^i | i = 1, \ldots, |\mathbf{M}_t|\}$, $\mathbf{S}_t = \{\mathrm{S}_t^i | i = 1, \ldots, |\mathbf{S}_t|\}$, and $\mathbf{O}_t = \{\mathrm{O}_t^i | i = 1, \ldots, |\mathbf{O}_t|\}$ where $\mathbf{A}_t$, $\mathbf{M}_t$, $\mathbf{S}_t$, and $\mathbf{O}_t$ indicate the set of the appearance models, motion models, state representation types, and observation types at time t, respectively, and $|\cdot|$ indicates cardinality of the set. Then, the i-th tracker at time t, T_t^i is constructed by choosing a specific appearance model A_t^i, motion model M_t^i, state representation type S_t^i, and observation type O_t^i from the sets, $\mathbf{A}_t$, $\mathbf{M}_t$, $\mathbf{S}_t$, and $\mathbf{O}_t$, respectively:

$T_t^i = (A_t^i, M_t^i, S_t^i, O_t^i)$. In a similar manner, our method finally makes the $|\mathbf{T}_t|$ number of trackers at time t, $\mathbf{T}_t = \{T_t^i | i = 1, \ldots, |\mathbf{T}_t|\}$ by fully associating the four basic ingredients in the sets, where $|\mathbf{T}_t| = |\mathbf{A}_t| \times |\mathbf{M}_t| \times |\mathbf{S}_t| \times |\mathbf{O}_t|$.

3.2. Decomposed Posterior Probability

The posterior probability in (1) can be efficiently estimated by the weighted linear combination of the decomposed posterior probabilities as in [11]:

$$p(\mathbf{X}_t|\mathbf{Y}_{1:t}) \approx \sum_{i=1}^{|\mathbf{T}_t|} p(T_t^i|\mathbf{Y}_{1:t}) p(\mathbf{X}_t|T_t^i, \mathbf{Y}_{1:t}), \qquad (2)$$

where $p(\mathbf{X}_t|T_t^i, \mathbf{Y}_{1:t})$ represents the i-th decomposed posteriori probability and $p(T_t^i|\mathbf{Y}_{1:t})$ indicates its weight.

Compared with direct estimation of the posterior probability, the decomposition strategy in (2) produces better performance under the logarithmic scoring criterion as follows.

Theorem 1. Averaging the decomposed posterior probabilities is optimal under the logarithmic scoring criterion in [5]:

$$E\left[log\left\{\sum_{i=1}^{|\mathbf{T}_t|} p(T_t^i|\mathbf{Y}_{1:t}) p(\mathbf{X}_t|T_t^i, \mathbf{Y}_{1:t})\right\}\right] \geq E\left[log\, p(\mathbf{X}_t|\mathbf{Y}_{1:t})\right], \qquad (3)$$

for any distribution $p(\mathbf{X}_t|\mathbf{Y}_{1:t})$ where the expectation is with respect to $\sum_{i=1}^{|\mathbf{T}_t|} p(T_t^i|\mathbf{Y}_{1:t}) p(\mathbf{X}_t|T_t^i, \mathbf{Y}_{1:t})$.

Proof. Inequality follows from the non-negative property of the Kullback-Leibler information divergence [1].

To decompose the posterior probability efficiently while reflecting the various changes in visual tracking, each decomposed posterior probability, $p(\mathbf{X}_t|T_t^i, \mathbf{Y}_{1:t})$ should be conditioned on the tracker, T_t^i, which runs robustly in the current tracking environment. The next section explains how to obtain the set of trackers and find the best state of the target using them.

3.3. Conditioned Maximum a Posteriori Estimate

Our method finds the best state of the target, $\hat{\mathbf{X}}_t$, at time t using the Conditioned Maximum a Posteriori (CMAP) estimate:

$$\hat{\mathbf{X}}_t \equiv \arg\max_{\mathbf{X}_t} p(\mathbf{X}_t|\mathbf{T}_t, \mathbf{Y}_{1:t}). \qquad (4)$$

Since the posterior probability in (4) is conditioned on the set of trackers, $\mathbf{T}_t$, we should search all possible trackers and states of the targets to obtain the CMAP estimate. However, it is unfeasible because the search space is drastically large and high dimensional.

[1]The decomposition strategy of the posterior probability is directly related to the Bayesian Model Averaging approach in [16].

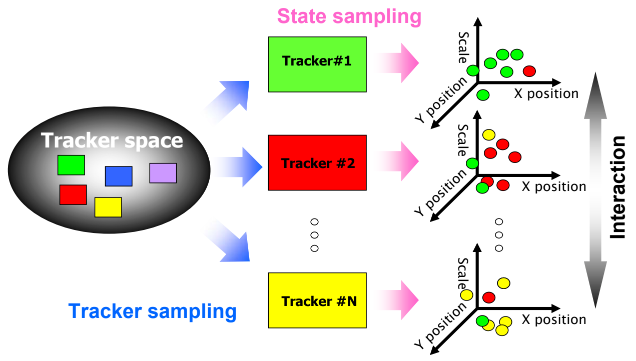


Figure 2. **Entire procedure of VTS** The sampler constructs the trackers by sampling them, runs the sampled trackers in parallel and interactively, and obtains samples of the target state utilizing the trackers.

Our method solves the aforementioned problem by approximately estimating the posterior probability in (4) with the samples of trackers and states. To do this, the method first obtains the samples of trackers and then, given the sampled trackers, it gets the samples for states. And, among the sampled states, the method chooses the best one, $\hat{\mathbf{X}}_t$, which gives the highest value on (4). Now the remaining task is how to obtain theses samples of trackers and states simultaneously in our visual tracker sampler framework.

4. Visual Tracker Sampler

VTS utilizes multiple Markov Chains. After each Markov Chain is modeled by each sampled tracker, T_t^i, the Markov Chains run in parallel and produce samples of the states, $\mathbf{X}_t$, to estimate each decomposed posteriori probability, $p(\mathbf{X}_t|T_t^i, \mathbf{Y}_{1:t})$ in (2) via the Metropolis Hastings algorithm. When the Chains are in the interacting mode, they communicate with the others and leap to better states of the target. In this mode, our method adjusts the contribution of each tracker by assessing the weights of the trackers implicitly utilizing (19). During the sampling process, the number of Markov Chains changes by either increasing or decreasing the number of trackers.

VTS consists of two different sampling processes: tracker sampling and state sampling. In the former, the sampler proposes new trackers and determines whether they can be accepted or not (section 4.1). Given the trackers, new states of the target are obtained by the state sampling process (section 4.2). Fig.2 describes the entire procedure of our VTS.

4.1. Tracker Sampling

Since the aforementioned four ingredients characterize trackers, sampling a tracker can be viewed as sampling its basic ingredients. While sampling them, the basic ingredients should be considered together because they are interrelated to each other. Note, however, that this is intractable

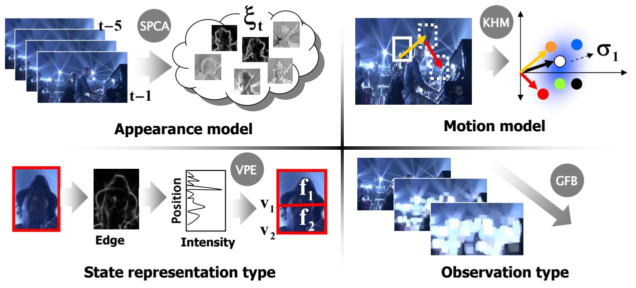


Figure 3. **Ingredients of the visual tracker** We design an appearance model, motion model, state representation type, and observation type utilizing SPCA, KHM, VPE, and GFB, respectively, to construct a specific tracker.

problem. So, in our work, we use the Gibbs sampling strategy instead. With this strategy, we decide one of them at a time, while the others are fixed to the best ones. To decide each basic ingredient properly, following aspects should be considered. The sampler should choose only the required models or types by accepting the ones that help track the target under the current environment, while avoiding unnecessarily complex ones by the acceptance ratio. By doing this, the sampler maintains the number of models or types as small as possible and shows good performance in terms of scalability. Additionally, to find good models or types efficiently in the drastically vast tracker space and reduce the convergence time, the sampler should utilize the proposal that sufficiently exploits underlying cues in the video. The next section describes how to design the acceptance ratio and proposal for each ingredient to achieve these goals.

4.1.1 State Representation Type

A state representation type should be designed to preserve the spatial information of the target while also covering the geometric variations of the target to some degree. To do this, our sampler represents the target as a combination of multiple fragments by adopting the philosophy of [1]. Then, the i-th state representation type is defined by:

$$\mathrm{S}_t^i \equiv \mathbf{X}_t = \{x_t, y_t, s_t, \mathbf{V}_t^i\}, \tag{5}$$

where x_t, y_t, and s_t indicates the x, y center position, and scale of the bounding box of the target, respectively, and $\mathbf{V}_t^i = \{\mathrm{v}_j | j = 1, \dots, |\mathbf{V}_t^i|\}$ denotes the set of vertical sub-indexes of the bounding box. Our sampler produces the $|\mathbf{F}_t^i|$ number of image fragments, $\mathbf{F}_t^i = \{\mathrm{f}_j | j = 1, \dots, |\mathbf{V}_t^i| + 1\}$, by dividing the bounding box horizontally at each vertical sub-index, v_j. The vertical sub-index, v_j, is efficiently achieved by the vertical projection of edge (VPE) [22] as shown in Fig.3. Then, the type S_t^i is added into $\mathbf{S}_t$ by the proposal function, $Q_S(\mathbf{S}_t^*; \mathbf{S}_t)$, which proposes the new set of state representation types, $\mathbf{S}_t^*$:

$$\mathbf{S}_t^* \sim Q_S(\mathbf{S}_t^*; \mathbf{S}_t) = \mathbf{S}_t \bigcup \mathrm{S}_t^i. \tag{6}$$

Given the proposed set of state representation types, $\mathbf{S}_t^*$, our sampler decides whether it is accepted or not with the acceptance ratio. The acceptance ratio is designed so that the state representation types in $\mathbf{S}_t^*$ reduces variations of the target appearance for the most recent five frames:

$$\begin{aligned} a_S &= min\left[1, \frac{p(\mathbf{S}_t^*|\mathbf{X}_t, \mathbf{Y}_{1:t})Q(\mathbf{S}_t; \mathbf{S}_t^*)}{p(\mathbf{S}_t|\mathbf{X}_t, \mathbf{Y}_{1:t})Q(\mathbf{S}_t^*; \mathbf{S}_t)}\right] \\ \text{where } & -\log p(\mathbf{S}_t^*|\mathbf{X}_t, \mathbf{Y}_{1:t}) \propto \\ & \sum_{i=1}^{|\mathbf{S}_t^*|} \sum_{j=1}^{|\mathbf{F}_t^i|} VAR(\mathrm{f}_j) + \lambda_S \log |\mathbf{S}_t^*|. \end{aligned} \tag{7}$$

In (7), $VAR(\mathrm{f}_j)$ returns variance of the j-th image fragment, f_j for the most recent five frames, $\log |\mathbf{S}_t^*|$ prevents the set $\mathbf{S}_t^*$ from having large numbers of state representation types, and λ_S is the weighting parameter.

4.1.2 Observation Type

Biological evidence shows that the human visual system uses the response of multiple filters called the filter bank to observe visual information. Similarly, more robust observation types can be achieved by using the Gaussian filter bank (GFB) [20]. The i-th observation type is constructed by the convolution between the image $\mathbf{I}_t$ and the Gaussian distribution with mean $\{x_t, y_t\}$ and variance Σ_i^2 for all $\{x_t, y_t\}$ of $\mathbf{X}_t$.

$$\mathrm{O}_t^i \equiv \mathrm{Y}_t = \mathbf{I}_t * G(\{x_t, y_t\}, \Sigma_i^2), \ \forall \{x_t, y_t\}, \tag{8}$$

where Σ_i is selected randomly from the uniform distribution, $U[0, 10]$, in component-wise manner for x_t and y_t. The new model O_t^i is inserted into $\mathbf{O}_t$ by the proposal function, $Q_O(\mathbf{O}_t^*; \mathbf{O}_t)$, which proposes the new set of observation types, $\mathbf{O}_t^*$:

$$\mathbf{O}_t{}^* \sim Q_O(\mathbf{O}_t^*; \mathbf{O}_t) = \mathbf{O}_t \bigcup \mathrm{O}_t^i. \tag{9}$$

The acceptance ratio is designed so that the response of the observation types in $\mathbf{O}_t{}^*$ become more similar among foreground images, but more different between foreground and background images for the most recent five frames. The foreground and background images are obtained by cropping the images within and around the bounding box of the target, respectively. Then, the acceptance ratio is defined by:

$$\begin{aligned} a_O &= min\left[1, \frac{p(\mathbf{O}_t^*|\mathbf{X}_t, \mathbf{Y}_{1:t})Q(\mathbf{O}_t; \mathbf{O}_t^*)}{p(\mathbf{O}_t|\mathbf{X}_t, \mathbf{Y}_{1:t})Q(\mathbf{O}_t^*; \mathbf{O}_t)}\right] \\ \text{where } & -\log p(\mathbf{O}_t^*|\mathbf{X}_t, \mathbf{Y}_{1:t}) \propto \\ & \frac{\sum_{i=1}^{|\mathbf{O}_t^*|} \sum_{j,k=t-5}^{t-1} DD(\phi_j^i, \phi_k^i)}{\sum_{i=1}^{|\mathbf{O}_t^*|} \sum_{j,k=t-5}^{t-1} DD(\phi_j^i, \psi_k^i)} + \lambda_O \log |\mathbf{O}_t^*|, \end{aligned} \tag{10}$$

where λ_O is the weighting parameter, and ϕ_j^i and ψ_k^i represent the foreground and background image of the *i*-th observation type at time j and k, respectively. In (10), the $DD(\phi_j^i, \psi_k^i)$ function [13] returns the diffusion distance between ϕ_j^i and ψ_k^i.

4.1.3 Appearance Model

An appearance model should cover most appearance changes of the target. Such model can be efficiently obtained by sparse principal component analysis (SPCA) described in [11]. SPCA finds several sparse principal components, in which each component is composed of a mixture of templates that describe the target appearance as shown in Fig.3. By choosing a principal component, ξ_t^i, with higher eigenvalue, the *i*-th appearance model is constructed as:

$$\mathrm{A}_t^i \equiv p(\mathbf{Y}_t|\mathbf{X}_t) = exp^{-\gamma DD(\mathbf{Y}_t(\mathbf{X}_t), \xi_t^i)}, \tag{11}$$

where γ denotes the weighting parameter, and $\mathbf{Y}_t(\mathbf{X}_t)$ indicates the observation at the state $\mathbf{X}_t$. Note that the state representation type $\mathbf{X}_t$ and observation type $\mathbf{Y}_t$ are fixed to the best ones during sampling the appearance models such like the Gibbs sampling strategy, as explained in section 4.1. Then, the proposal function, $Q_A(\mathbf{A}_t^*; \mathbf{A}_t)$ adds the new model, A_t^i into $\mathbf{A}_t$ and proposes the new set of appearance models, $\mathbf{A}_t^*$:

$$\mathbf{A}_t^* \sim Q_A(\mathbf{A}_t^*; \mathbf{A}_t) = \mathbf{A}_t \bigcup \mathrm{A}_t^i. \tag{12}$$

Our sampler accepts the proposed set of appearance models, $\mathbf{A}_t^*$ with high probability if the appearance models in $\mathbf{A}_t^*$ produce higher likelihood scores than those in $\mathbf{A}_t$ at the MAP state, $\hat{\mathbf{X}}_t$ for the most recent five frames, in which the MAP state at time t found by (4) indicates the best state of the target at time t:

$$\begin{aligned} a_A &= min\left[1, \frac{p(\mathbf{A}_t^*|\hat{\mathbf{X}}_t, \mathbf{Y}_{1:t})Q(\mathbf{A}_t; \mathbf{A}_t^*)}{p(\mathbf{A}_t|\hat{\mathbf{X}}_t, \mathbf{Y}_{1:t})Q(\mathbf{A}_t^*; \mathbf{A}_t)}\right] \\ \text{where } & -\log p(\mathbf{A}_t^*|\hat{\mathbf{X}}_t, \mathbf{Y}_{1:t}) \propto \\ & \sum_{i=1}^{|\mathbf{A}_t^*|} \sum_{j=t-5}^{t-1} DD(Y_j(\hat{\mathbf{X}}_j), \xi_t^i) + \lambda_A \log |\mathbf{A}_t^*|. \end{aligned} \tag{13}$$

In (13), $Y_j(\hat{\mathbf{X}}_j)$ indicates the observation at the MAP state, $\hat{\mathbf{X}}_j$ at time j, and λ_A is the weighting parameter.

4.1.4 Motion Model

A motion model has to describe representative characteristics of the target motion over time. It is efficiently found by the K-Harmonic Means (KHM) method, which clusters data and finds centers of the clusters as shown in Fig.3, where KHM is known to be insensitive to the initialization of the centers [24]. The data, $\mathbf{D}_t$, for KHM is acquired by gathering velocity vectors between accepted neighbor states for the most recent five frames. By choosing a cluster center, $\sigma_i = [\sigma_i^x, \sigma_i^y, \sigma_i^s, \sigma_i^s]^T$ of $\mathbf{D}_t$ with a higher confidence value, the *i*-th motion model is constructed as:

$$\mathrm{M}_t^i \equiv p(\mathbf{X}_t^*|\mathbf{X}_t) = G(\mathbf{X}_t, \sigma_i^2), \tag{14}$$

where G denotes the Gaussian function with mean $\mathbf{X}_t$ and variance σ_i^2. Note that the state representation type $\mathbf{X}_t$ is fixed to the best one during sampling the motion models following the Gibbs sampling strategy, as explained in section 4.1. Then, the proposal function, $Q_M(\mathbf{M}_t^*; \mathbf{M}_t)$ adds the new model, M_t^i into $\mathbf{M}_t$ and proposes the new set of motion models, $\mathbf{M}_t^*$:

$$\mathbf{M}_t^* \sim Q_M(\mathbf{M}_t^*; \mathbf{M}_t) = \mathbf{M}_t \bigcup \mathrm{M}_t^i. \tag{15}$$

Our sampler accepts $\mathbf{M}_t^*$ with high probability if the motion models in $\mathbf{M}_t^*$ have more accurate cluster centers, σ_i than those in $\mathbf{M}_t$:

$$\begin{aligned} a_M &= min\left[1, \frac{p(\mathbf{M}_t^*|\mathbf{X}_t, \mathbf{Y}_{1:t})Q(\mathbf{M}_t; \mathbf{M}_t^*)}{p(\mathbf{M}_t|\mathbf{X}_t, \mathbf{Y}_{1:t})Q(\mathbf{M}_t^*; \mathbf{M}_t)}\right] \\ \text{where } & -\log p(\mathbf{M}_t^*|\mathbf{X}_t, \mathbf{Y}_{1:t}) \propto \\ & \sum_{i=1}^{|\mathbf{M}_t^*|} VAR(\mathbf{D}_t, \sigma_i) + \lambda_M \log |\mathbf{M}_t^*|. \end{aligned} \tag{16}$$

In (16), $VAR(\mathbf{D}_t, \sigma_i)$ returns the variance of data, $\mathbf{D}_t$, that belongs to the cluster centered on σ_i, and λ_M is the weighting parameter.

For removing models or types from the current set, the sampler randomly selects a model or type, proposes a new set that does not contain it, and accepts the proposal with the corresponding acceptance ratio in (7)(10)(13)(16).

4.2. State Sampling

Each sampled tracker in section 4.1 produces the predefined number of states. For example, using the tracker constructed by the *i*-th appearance model, the *j*-th motion model, the *k*-th state representation type, and the *l*-th observation type, our sampler proposes a new state, $\mathbf{X}_t^{j^*}$:

$$\mathbf{X}_t^{j^*} \sim Q_j(\mathbf{X}_t^{j^*}|\mathbf{X}_t^j) = G(\mathbf{X}_t^j, \sigma_j^2), \tag{17}$$

and determines whether the proposed state is accepted or not with the following acceptance ratio:

$$\begin{aligned} a_P &= min\left[1, \frac{p(\mathbf{Y}_t|\mathrm{A}_t^i, \mathrm{S}_t^k, \mathrm{O}_t^l, \mathbf{X}_t^{j^*})Q_j(\mathbf{X}_t^j; \mathbf{X}_t^{j^*})}{p(\mathbf{Y}_t|\mathrm{A}_t^i, \mathrm{S}_t^k, \mathrm{O}_t^l, \mathbf{X}_t^j)Q_j(\mathbf{X}_t^{j^*}; \mathbf{X}_t^j)}\right] \\ \text{where } & p(\mathbf{Y}_t|\mathrm{A}_t^i, \mathrm{S}_t^k, \mathrm{O}_t^l, \mathbf{X}_t^{j^*}) = \prod_{m=1}^{|\mathbf{F}_t^k|} exp^{-\gamma\frac{DD(\mathrm{Y}_t^l(\mathrm{f}_m), \xi_t^i)}{|\mathbf{F}_t^k|}}. \end{aligned} \tag{18}$$

	MC	IVT	MIL	VTD	VTS*	VTD*	#N
soccer	53	116	41	**23**	**17**	21	408
skating1	172	213	85	**8**	**8**	7	304
animal	26	**21**	30	22	**10**	11	696
shaking	98	150	38	**20**	**5**	5	616

Table 1. **Comparison of tracking accuracy.** The numbers denote the center location errors in pixels, where red is the best result and blue is the second-best result. The green numbers indicate the total number of samples utilized to track the target.

In (18), $p(\mathbf{Y}_t|\mathrm{A}_t^i, \mathrm{S}_t^k, \mathrm{O}_t^l, \mathbf{X}_t^{j^*})$ is the modified appearance model of (11), which further considers the k-th state representation type, and the l-th observation type, where $\mathbf{Y}_t^l(\mathrm{f}_m)$ indicates the l-th observation at the m-th image fragment.

When the sampler is in the interacting mode, the trackers communicate with each others and leap to better states of the target. A tracker accepts the state of another tracker constructed by the i-th appearance model, the j-th motion model, the k-th state representation type, and the l-th observation type as its own state with the following probability:

$$a_I = \frac{p(\mathbf{Y}_t|\mathrm{A}_t^i, \mathrm{S}_t^k, \mathrm{O}_t^l, \mathbf{X}_t^{j^*})}{\sum_{i=1}^{|\mathbf{A}_t|}\sum_{j=1}^{|\mathbf{M}_t|}\sum_{k=1}^{|\mathbf{S}_t|}\sum_{l=1}^{|\mathbf{O}_t|} p(\mathbf{Y}_t|\mathrm{A}_t^i, \mathrm{S}_t^k, \mathrm{O}_t^l, \mathbf{X}_t^{j^*})}. \tag{19}$$

5. Experimental Results

Using the 7 datasets which are publicly available and 4 datasets made by us, our method (VTS) [2] was compared with five different tracking methods: standard MCMC (MC) based on [9][15], Mean Shift (MS) [4] based on the implemented function in OpenCV, Incremental Visual Tracking (IVT) in [17], Multiple Instance Learning (MIL) in [2], and Visual Tracking Decomposition (VTD) in [11]. Same initializations were set to all methods for fair comparison and the parameters of the methods were adjusted to show the best performance. To obtain the tracking results of IVT, MIL, and VTD, we used the software provided by authors.

In all experiments, we set λ_S,λ_O,λ_A, and λ_M in (7),(10),(13), and (16) to 0.05 and γ in (11)(18) to 5, which hardly affects on the tracking results. We used hue, saturation, intensity, and edge templates as the features of VTS.

5.1. Quantitative results

5.1.1 Performance of Tracker Sampling Process

To evaluate the performance of tracker sampling process of VTS, we compared conventional methods with VTS*, where VTS* indicates our VTS that constructs trackers by changing the appearance model and motion model only. If we test VTS directly, it is not fair comparison with VTD in [11], because VTS considers the additional tracker elements such as the state representation type and observation type. Table 1 summarizes location errors on publicly available video sequences for different tracking algorithms. Since VTS* automatically utilizes different numbers of trackers according to the tracking environment, we adjusted the total number of samples to that of VTS* to compare the sampling-based methods, which are MC, IVT, and VTD. With a small number of samples, VTS* produced similar and even better tracking results compared with VTD*, which is the visual tracking decomposition method utilizing more samples, 800. Moreover, VTS* showed the best tracking accuracy when the same number of samples were utilized for all tracking methods. The better performance of VTS* comes from the tracker sampling process, in which VTS* changes the number of trackers and maintains only the required trackers by adaptively selecting appropriate ones depending on the current tracking environment.

[2]The description of the entire algorithm, result videos and datasets can be found at http://cv.snu.ac.kr/.

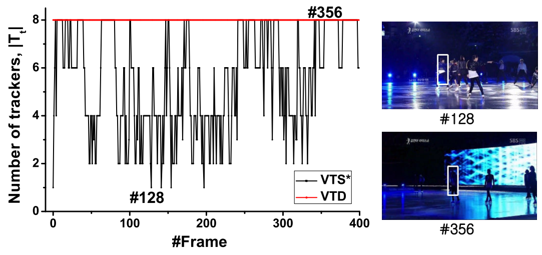


Figure 4. **Number of trackers** at each frame in *skating1* sequence.

As illustrated in Fig.4, the tracker sampling process of VTS* adaptively changed the number of trackers according to the tracking environment over time. For example, it decreased the number of trackers and saved the resource at frame #128, because the frame included almost no movements and appearance changes of the target. At frame #356, VTS* increased the number of trackers to capture the appearance variations due to the severe illumination changes. On the other hand, VTD wasted the resource by always using 8 trackers, so inaccurately track the target with a small number of samples. IVT and MIL failed to adapt the tracker and finally missed the target, although they showed real-time performance.

5.1.2 Performance of Overall Process

We compared conventional methods with VTS by evaluating tracking accuracy. For this, we constructed highly challenging video sequences as follows. We manually added noise and motion blur into the *soccer* and *skating1* sequences, and made new sequences, *soccer** and *skating1**. Then, the sequences include severe illumination, viewpoint changes, occlusions, noise, and motion blur at the same time. Moreover, we obtained new tracking se-

	MS	MC	IVT	MIL	VTD	VTS	#N
*soccer**	192	72	225	147	**34**	**24**	1224
*skating1**	211	126	291	87	**16**	**8**	976
iron	98	78	104	122	**30**	**15**	1188
matrix	130	123	**50**	57	80	**12**	1036
tiger1	93	32	83	**15**	23	**12**	642
david	88	41	**5**	23	43	**7**	576
occlface	45	19	20	27	**9**	**8**	408

Table 2. **Comparison of tracking accuracy.** For the fair comparison with [2, 11], we run our algorithm five times and averaging the results.

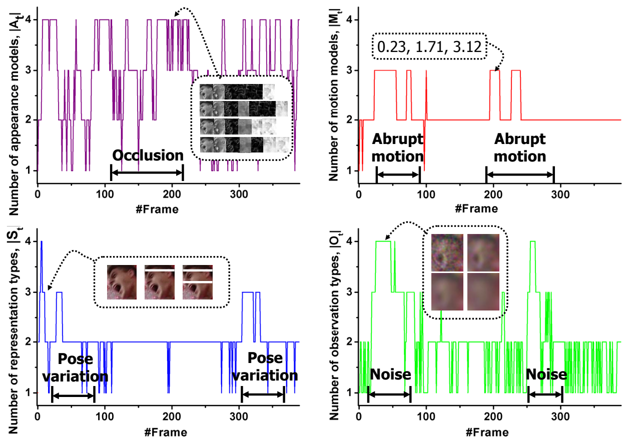


Figure 5. **Number of each basic ingredient** at each frame in *soccer** sequence.

quences captured from real movies, which are *iron* and *matrix* sequences in which challenging appearance and motion changes exist. In these sequences, VTS most accurately tracked the targets as shown in Table 2. VTS robustly dealt with noise and motion blur since it newly constructed the robust trackers that can cope with the current tracking environment and further considered the state representation and observation types to construct trackers as comparison with VTD. During the tracking process, it found the appropriate observation types by determining the variances of the Gaussian filter toward making the observation robust to noise, and the appropriate state representation type by separating the target into several fragments, of which combination is robust to motion blur. Note that VTS also produced most accurate tracking results in the conventional *tiger1*, *david*, and *occlface* sequences.

To demonstrate how VTS produces the accurate tracking results with the understanding of its mechanisms, we provide intermediary results of the four ingredients in VTS. As shown in Fig.5, VTS increased the number of each ingredient appropriately when there were specific changes in appearance or motion. For example, VTS constructed three motion models, of which proposal variances are 0.23, 1.71, and 3.12, and successfully tracked the complex motions. To overcome severe noise in the sequence, VTS automatically employed four observation types, in which the degree of gaussian blur is different. When there were pose variations, VTS made appearance of the target insensitive to the variations as passible by utilizing three state representation types. With four appearance models, VTS described both occluded and non-occluded target and robustly tracked it.

5.2. Qualitative results

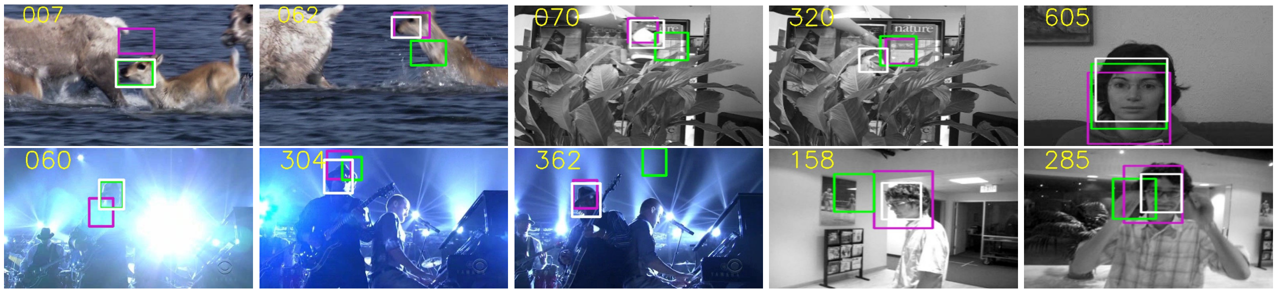
Fig.6(a) shows the qualitative tracking performance with a small number of samples. In the sequences, VTS successfully tracked the targets by searching the state space efficiently. Our algorithm is robust to the drift problems since it utilizes multiple trackers. Although some sampled trackers may fail to track the target due to drift, the others successfully track it with different models. It is also proven mathematically in section 3.2 by showing that multiple trackers produce more accurate posterior probability. On the other hand, the results of VTD and MIL drifted into a background with a small number of samples when there were abrupt motions as in the frame #62 of the *animal* sequence, and severe illumination changes as in the frame #304 of the *shaking* sequence.

Fig.6(b) illustrates the tracking results in the highly challenging sequences, which include severe noise and motion blur as well. VTS tracked the targets accurately and robustly although severe types of appearance changes occurred at the same time. Note that VTD successfully tracked the targets if only noise or motion blur existed, but failed to track it, when these changes occurred with illumination changes as in the frame #377 of the *skating1** sequence, and with occlusions as in the frame #279 of the *soccer** sequence.

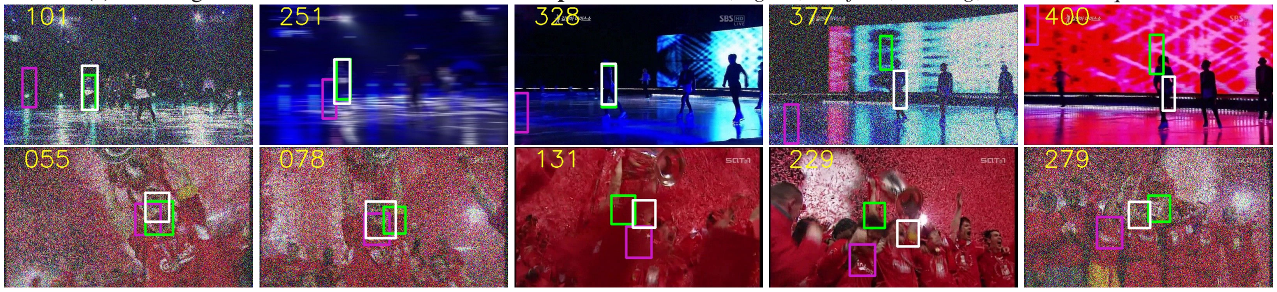
Fig.6(c) presents the tracking results under the real-world tracking environment utilizing the *iron* and *matrix* sequences. As shown in the figure, VTS covered most variations occurring in the sequences and robustly tracked the target. However, MIL and VTD failed to track the target accurately due to the drastically severe appearance changes at the frame #102 in the *iron* sequence and at the frame #054 in the *matrix* sequence. Moreover, MIL and VTD tracker were frequently hijacked by other objects looking similar to the target at the frame #045 in the *matrix* sequence.

6. Conclusion

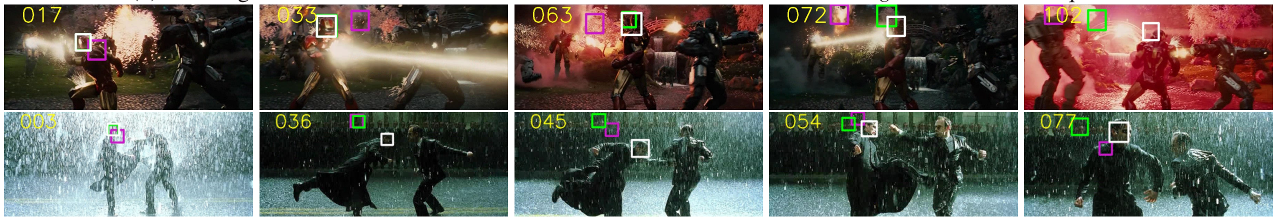
In this paper, we proposed a novel tracking framework called visual tracker sampler. Visual tracker sampler efficiently samples multiple good trackers from the tracker space dynamically, and tracks the target robustly and successfully utilizing them in challenging tracking environments. The experimental results demonstrated that the proposed method outperformed conventional tracking algorithms in terms of tracking accuracy and efficiency.



(a) Tracking results with a **small number of samples** in the *animal*, *tiger1*, *occlface*, *shaking*, and *david* sequences.



(b) Tracking results when there are severe **noise** and **motion blur** in the *skating1** and *soccer** sequences.



(c) Tracking results in the **challenging** *iron* and *matrix* sequences.

Figure 6. **Qualitative tracking results.** White, green, and purple rectangles represent tracking results of VTS, VTD, and MIL, respectively.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. *CVPR*, 2006. 4

[2] B. Babenko, M. Yang, and S. Belongie. Visual tracking with online multiple instance learning. *CVPR*, 2009. 1, 2, 6, 7

[3] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *PAMI*, 27(10):1631–1643, 2005. 2

[4] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. *CVPR*, 2000. 6

[5] I. J. Good. Rational decisions. *J. Roy. Statistical Society*, Ser. B.(14):107–114, 1952. 3

[6] B. Han and L. Davis. On-line density-based appearance modeling for object tracking. *ICCV*, 2005. 1

[7] M. Isard and A. Blake. Icondensation: Unifying low-level and high-level tracking in a stochastic framework. *ECCV*, 1998. 2

[8] A. D. Jepson, D. J. Fleet, and T. F. E. Maraghi. Robust online appearance models for visual tracking. *PAMI*, 25(10):1296–1311, 2003. 1

[9] Z. Khan, T. Balch, and F. Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *PAMI*, 27(11):1805–1918, 2005. 2, 6

[10] J. Kwon and K. M. Lee. Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling. *CVPR*, 2009. 1

[11] J. Kwon and K. M. Lee. Visual tracking decomposition. *CVPR*, 2010. 1, 2, 3, 5, 6, 7

[12] B. Leibe, K. Schindler, N. Cornelis, and L. Van Gool. Coupled object detection and tracking from static cameras and moving vehicles. *PAMI*, 30(10):1683–1698, 2008. 1

[13] H. Ling and K. Okada. Diffusion distance for histogram comparison. *CVPR*, 2006. 5

[14] X. Mei and H. Ling. Robust visual tracking using l1 minimization. *ICCV*, 2009. 1

[15] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. *ECCV*, 2002. 6

[16] A. E. Raftery and Y. Zheng. Discussion: Performance of bayesian model averaging. *J. Amer. Statistical Assoc.*, 98, 2003. 3

[17] D. A. Ross, J. Lim, R. Lin, and M. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1):125–141, 2008. 1, 2, 6

[18] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof. Prost: Parallel robust online simple tracking. *CVPR*, 2010. 1

[19] B. Stenger, T. Woodley, and R. Cipolla. Learning to track with multiple observers. *CVPR*, 2009. 2

[20] J. Sullivan, A. Blake, M. Isard, and J. MacCormick. Bayesian object localisation in images. *IJCV*, 44(2):111–135, 2001. 4

[21] K. Toyama and E. Horvitz. Bayesian modality fusion: Probabilistic integration of multiple vision algorithms for head tracking. *ACCV*, 2000. 1

[22] F. Wang, S. Yua, and J. Yanga. Robust and efficient fragments-based tracking using mean shift. *Int. J. Electron. Commun.*, 64(7):614–623, 2010. 4

[23] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4), 2006. 1

[24] B. Zhang, M. Hsu, and U. Dayal. K-harmonic means - a data clustering algorithm. *HP Technical Report*, 1999. 5

[25] T. Zhao and R. Nevatia. Tracking multiple humans in crowded environment. *CVPR*, 2004. 2