Visual Tracking of Non-Rigid Objects with Partial Occlusion through Elastic Structure of Local Patches and Hierarchical Diffusion

Kwang Moo Yi[a], Hawook Jeong[b], Soo Wan Kim[b], Shimin Yin[c], Songhwai Oh[b,*], Jin Young Choi[b]

[a]*Computer Vision Lab, EPFL, Lausanne, Switzerland*
[b]*Department of Electrical and Computer Engineering, ASRI, Seoul National University, Seoul, Korea*
[c]*Samsung Techwin, Gyeonggi-do, Korea*

Abstract

In this paper, a tracking method based on sequential Bayesian inference is proposed. The proposed method focuses on solving both the problem of tracking under partial occlusions and the problem of non-rigid object tracking in real-time on a desktop personal computer (PC). The proposed method is mainly composed of two parts: (1) modeling the target object using elastic structure of local patches for robust performance; and (2) efficient hierarchical diffusion method to perform the tracking procedure in real-time. The elastic structure of local patches allows the proposed method to handle partial occlusions and non-rigid deformations through the relationship among neighboring patches. The proposed hierarchical diffusion method generates

*Corresponding author at: Department of Electrical and Computer Engineering (College of Engineering) Seoul National Univ. #014, 1 Gwanangno, Gwanak-gu, Seoul, Korea, 151-744. Tel: +82 2 880 1511 Fax: +82 2 888 4182

Email addresses: kwang.yi@epfl.ch (Kwang Moo Yi), hwjeong@snu.ac.kr (Hawook Jeong), soowankim@snu.ac.kr (Soo Wan Kim), simcom79@gmail.com (Shimin Yin), songhwai@snu.ac.kr (Songhwai Oh), jychoi@snu.ac.kr (Jin Young Choi)

samples from the region where the posterior is concentrated to reduce computation time. The method is extensively tested on a number of challenging image sequences with occlusion and non-rigid deformation. The experimental results show the real-time capability and the robustness of the proposed method under various situations.

Keywords: Visual Tracking, Local Patches, Markov Random Field, Particle Filtering, Hierarchical Diffusion

---

1. Introduction

Object tracking is an important computer vision problem which can be used for various applications such as robot vision, video analysis, behavior recognition, home automation, and visual surveillance [1]. In order for the whole system to work properly for these applications, accurate tracking results are required. Various tracking methods have been tried [2, 3, 4, 5] during the last decade and they have proven to be successful for these applications. However, the applicability of these algorithms is somewhat limited when applied to objects undergoing various disturbances in real-world scenarios.

The limitations of conventional methods arise from the fact that they have strong assumptions about the input video sequence, such as constant movements of the target object and consistent views. In real-world scenarios, occlusions may occur frequently with the target object showing non-rigid deformations, degrading the performance of conventional methods. For instance, it is easy for people walking nearby to occlude each other. Further-

more, people show non-rigid movements as well, making it extremely difficult for conventional methods to track the target person successfully throughout the image sequence.

Kernel-based tracking [6] is widely used for its simplicity and computational efficiency. The method is generally known to provide tracking results very efficiently, suitable for real-time purposes. However, the method uses local optimization techniques; therefore, it is easy for the tracker to fall into local maxima (or minima). Also, the method gets easily distracted by the background (known as the background clutter problem) and has no clear way of adapting to scale and illumination changes. Particle filtering based methods are another class of methods which became popular after its first introduction in [7]. These methods try to solve the tracking problem with Monte Carlo (MC) simulation. Unlike kernel-based trackers, particle filtering based methods can be easily extended to track objects showing movements other than translational movements, such as affine motions. Many variations have been proposed [8, 9, 10], each with different measurement models. Among these, subspace-based measurements [8], inspired by the work of Black *et al.* [11], have proven to be successful. These methods are able to adapt to various changes such as changes in views and illumination and changes within the model by modeling the target object with a subspace-based representation. However, the methods assume slow changes of the target object and do not consider occlusions during learning, causing the trackers to drift.

Adam *et al.* [12] proposed a fragments-based tracking method to solve occlusion problems. Their method represents objects with multiple image fragments and combines the votes from each fragment to obtain a tracking

result. The method extends traditional kernel-based methods to be robust to partial occlusions, but it is still limited to tracking translational movements. Mei and Ling [13] treated visual tracking as a sparse approximation problem under the particle filtering framework. They address occlusions and corruptions through a set of trivial templates. Each target candidate is then sparsely represented through l_1 minimization and the candidate with the smallest projection error is taken as the tracking target. The sparse representation considers occlusions through trivial templates and it is, therefore, more robust to partial occlusions than other traditional particle filtering methods. However, the results of their method are not reliable when the target object shows non-rigid movements.

To solve the problem of non-rigid object tracking, Kwon and Lee [5] proposed a method which models the target object as a collection of local patches. In [5], the target object is described with a star model of local patches, and Adaptive Basin Hopping Monte Carlo (A-BHMC) sampling is used to minimize the energy of the model. The patches in the model used to describe the target object are consistently updated through a heuristic scheme. This enables the tracker to adapt to drastic changes in the appearance and the shape of the target. A-BHMC reduces the number of particles required for tracking, making the computation time tractable. However, their method tends to have trouble when tracking objects showing large displacements, and still requires large amount of computation even with A-BHMC. Godec *et al*. [14] proposed a tracking method based on the generalized Hough transform. They extended the idea of Hough Forests to the online domain and coupled the voting based detection and back-projection with a rough seg-

mentation based on GrabCut [15]. Their method gets rid of the bounding-box limitation and returns tracking results which contain only the target object. However, their method is limited to handling non-rigid deformations only, and has problems when tracking objects under occlusions.

Recently, methods trying to solve tracking problems with the "tracking by detection" scheme have drawn attention. Grabner *et al.* [4] proposed a method to train a discriminative classifier in an online manner using online boosting to separate the object from the background. Their method treats the tracking result as a positive sample and nearby region as negative samples. Using these samples, the method continuously updates the classifier. However, due to occlusions or slight inaccuracies, the performance of the classifier tends to degrade over time as the classifier is updated. To solve this drift issue, Babenko *et al.* [16] employed multiple instance learning, which updates the classifier with multiple positive examples rather than just one. Also, instead of using multiple instance learning, Stalder *et al.* [17] proposed a multiple classifier system which consists of an off-line classifier, a supervised on-line classifier, and a semi-supervised on-line classifier to prevent the tracker from drifting. However, both methods, [16] and [17], do not consider occlusions, making the methods vulnerable to them. Kalal *et al.* [3] proposed a method using both a tracker and a classifier to create training sets with structural constraints. Their method collects training samples and uses them only when the structural constraint meets, *i.e.*, only when the classifier and the tracker agrees. Through this procedure, their method becomes robust to drifting problems. Unfortunately, their method has weaknesses against occlusions as well.

Our method is targeted to solve both the problem of partial occlusions and the problem of non-rigid deformation in real-time on a modern desktop personal computer (PC). The proposed algorithm models the target object through a structure of local patches with spring-like connections, formulated in a Markov Random Field (MRF) style framework. Each local patch is considered to be connected with its neighbors as the local structures of the target object are embedded into the MRF structure.

The advantage of our method is that when partial occlusions occur, un-occluded patches will enforce the maintenance of the local structures, owing to the spring-like connections among local patches. As a result, occluded patches will be directed to their correct positions through the relationship among neighboring patches, thus making our method robust against partial occlusions. Non-rigid deformations are also well described since they can be explained as a collection of local movements of patches. Unlike other methods which concentrate on occlusions [12, 13], or methods which focus on non-rigid movements only [5, 14], our method addresses both problems simultaneously. In addition, to achieve real-time performance on a modern desktop PC, which is critical in many tracking applications, a hierarchical diffusion approach is proposed to overcome the curse of dimensionality.

The proposed model is similar to the deformable models used for 3D shape recovery tasks [18, 19], but differs in the fact that the structure is not restricted to regular or repetitive grids. Also, for visual tracking tasks, the shape of the tracked object is not known in advance, thus the applicability of methods for 3D shape recovery tasks [18] are limited. Our method is also closely related to pictorial structure frameworks [20, 21] in the fact that we

model the entire object as a collection of parts. Pictorial models have been successfully applied to object recognition and detection tasks [22]. In our work, we focus on the visual tracking task, where we are not provided with a prior dataset to learn about the target object, in contrast to the object recognition and detection task.

To demonstrate the effectiveness of our method, we have experimented with a number of challenging image sequences. The experimental results show that our method is the most robust against both partial occlusions and non-rigid deformations, compared with other methods. Especially, our method runs in real-time (20 to 50 frames per second), whereas other state-of-the-art methods capable of tracking non-rigid objects proposed in [5] and [14] runs only a few frames per second.

A preliminary version of this paper has been presented in [23]. In this version, we extend the preliminary work with a new likelihood function based on linear Support Vector Machine (SVM) [24, 25] and logistic fitting [26]. We also provide thorough analysis of the method's performance through comparison with other methods both qualitatively and quantitatively on a number of challenging image sequences.

2. Tracking with Elastic Structure of Local Patches

2.1. Sequential Bayesian Inference Framework

The proposed tracking method is based on a sequential Bayesian inference framework. We denote the object state at time t as $\mathbf{X}_t$, where $\mathbf{X}_t = \left(\mathbf{X}_t^1, \mathbf{X}_t^2, \cdots, \mathbf{X}_t^N\right)$ and $\mathbf{X}_t^k$ denotes the state of the k^{th} local patch of the object (*e.g.*, the position of the patch) among the N local patches

used to describe the object. Then, if we denote the observations up to time t as $\mathbf{Y}_{1:t}$, the problem of object tracking can be defined as finding $\hat{\mathbf{X}}_t$ such that,

$$\hat{\mathbf{X}}_t = \arg\max_{\mathbf{X}_t} P(\mathbf{X}_t|\mathbf{Y}_{1:t}) \ . \tag{1}$$

For sequential Bayesian inference, the posterior probability $P(\mathbf{X}_t|\mathbf{Y}_{1:t})$ is sequentially updated as the following:

$$P(\mathbf{X}_t|\mathbf{Y}_{1:t}) \propto P(\mathbf{Y}_t|\mathbf{X}_t) \int P(\mathbf{X}_t|\mathbf{X}_{t-1})P(\mathbf{X}_{t-1}|\mathbf{Y}_{1:t-1})d\mathbf{X}_{t-1} \ . \tag{2}$$

Here, $P(\mathbf{Y}_t|\mathbf{X}_t)$ is the conditional probability of the current observation $\mathbf{Y}_t$ given the current state $\mathbf{X}_t$, which is referred to as the likelihood term, and $P(\mathbf{X}_t|\mathbf{X}_{t-1})$ is the transition probability from $\mathbf{X}_{t-1}$ to $\mathbf{X}_t$.

Typically, for object tracking, since we consider many types of movements (*e.g.*, translation, rotation, scale, and affine motions), obtaining an exact analytic solution is not an easy task. Also, when the model is applied, the probabilistic distribution in the solution space is non-convex leading to difficulties when using local optimization based techniques. Therefore as in [7], we use particle filtering (also known as sequential Monte Carlo sampling) to solve the problem. If we denote the l^{th} sample in particle filtering as $\mathbf{X}_{t,[l]}$, then Eq. (1) can be re-written as

$$\hat{\mathbf{X}}_t = \mathbf{X}_{t,[\hat{l}]} \ , \tag{3}$$

where

$$\hat{l} = \arg\max_{l} P(\mathbf{Y}_t|\mathbf{X}_{t,[l]}) \ . \tag{4}$$

Note that since we are performing particle filtering, the likelihood of each particle $P(\mathbf{Y}_t|\mathbf{X}_{t,[l]})$, after diffusion according to the transition probability

and re-sampling, corresponds to the posterior probability. Thus, the problem of object tracking is now the problem of simulating the posterior distribution $P(\mathbf{X}_t|\mathbf{Y}_{1:t})$ with particle filtering, and then taking the particle with the best probability as a solution.

Our method differs from the traditional particle filtering methods due to the fact that the likelihood $P(\mathbf{Y}_t|\mathbf{X}_t)$ is obtained through an MRF-style manner. Through this MRF-style method, both the individual likelihood of each patch and the relationship among them are maximized while tracking. The MRF-style elastic structure of local patches, which will be explained in Section 2.2, has the advantage that the resultant posterior distribution considers both the underlying local structures and the non-rigid deformations simultaneously. To allow the proposed method to perform the tracking procedure within the real-time constraint and to avoid from using excessive number of particles when covering the high dimension solution space, we adopt a hierarchical diffusion scheme which benefits from the assumption that local deformation is not large between consecutive frames. Details of the proposed hierarchical diffusion are explained in Section 2.6.

2.2. Elastic Structure of Local Patches

In our work, we treat the target object as a collection of local parts, rather than treating the target object as a whole. Local parts are described with $n \times n$ size local patches, and local patches are assumed to be connected with nearby neighbors forming an elastic structure as in Fig. 1. This model of the target object is realized through an MRF-style framework. The likelihood of each local patch is considered to be the unary likelihood of the MRF, and the structure among them is considered to be the neighborhood relationship of

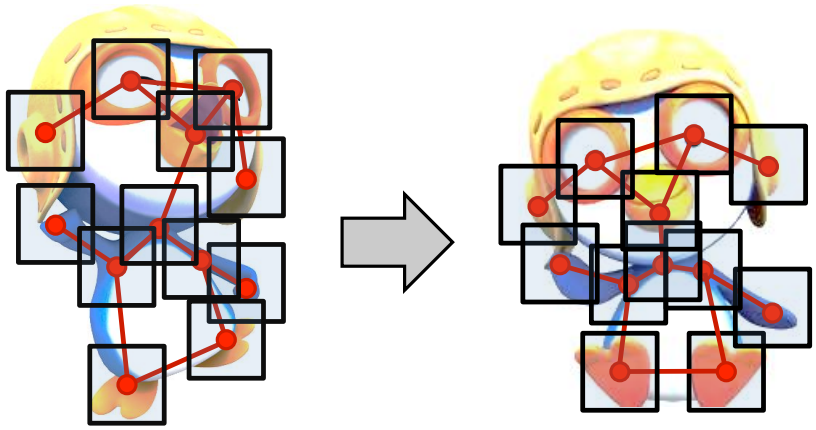


Fig. 1. Example of elastic structure of local patches used to describe the target object. Black boxes denote each local patch and red lines denote each connection.

the MRF. Since each local patch is connected with its neighbors forming an MRF, our model prefers solutions with the local structure of the target object preserved. Therefore, even if some of the patches are occluded, other un-occluded patches will drive occluded patches to the correct positions, causing the proposed model to be robust against partial occlusions. Also, since we describe the target object using local patches, we are able to represent non-rigid deformations as a collection of movements of local patches.

We consider the initial patch positions and connections are given in the first frame. This initial setting can be given manually or automatically by some other detection system or by some certain strategy (*e.g.*, dividing the target bounding box into equal grids and considering each grids to be connected to its direct neighbors). The given patches should cover most of the target object with connections describing the structure of the target object.

The likelihood $P(\mathbf{Y}_t|\mathbf{X}_t)$ in Eq. (2) is modeled with the probability of

the MRF configuration describing the structure of local patches. Therefore, $P(\mathbf{Y}_t|\mathbf{X}_t)$ is designed as

$$P(\mathbf{Y}_t|\mathbf{X}_t) \propto \prod_{k=1}^{N} \left[P(\mathbf{Y}_t|\mathbf{X}_t^k) \prod_{j \in N_k} P(\mathbf{X}_t^k|\mathbf{X}_t^j) \right] , \tag{5}$$

where $P(\mathbf{Y}_t|\mathbf{X}_t^k)$ is the likelihood of a single patch, $P(\mathbf{X}_t^k|\mathbf{X}_t^j)$ is the prior probability describing the relationship among neighboring patches, and N_k denotes the neighbors of the k^{th} patch. Note that $P(\mathbf{Y}_t|\mathbf{X}_t)$ is referred to as the posterior probability in MRF literature, but in our case, it is the likelihood of a single configuration. Since we are based on sequential Bayesian inference, the posterior probability for our model is shown in Eq. (2). We are also following the standard MRF configuration and are assuming conditional independence among patches which are not neighbors, as well as the independence among unary likelihoods of each patch $P(\mathbf{Y}_t|\mathbf{X}_t^k)$. (See [27] for more details on MRFs.)

In the energy form, if we denote the total energy of the configuration as $E(\mathbf{Y}_t; \mathbf{X}_t)$, the energy of a single patch as $E(\mathbf{Y}_t; \mathbf{X}_t^k)$, and the energy from the relationship between patches as $E(\mathbf{X}_t^k, \mathbf{X}_t^j)$, we can write the total energy of the MRF model (which is simply the sum of the observation and neighborhood energy of all patches) as

$$E(\mathbf{Y}_t; \mathbf{X}_t) = Z + \sum_{k=1}^{N} \left[E(\mathbf{Y}_t; \mathbf{X}_t^k) + \sum_{j \in N_k} E(\mathbf{X}_t^k, \mathbf{X}_t^j) \right] , \tag{6}$$

where Z is a normalizing constant which is not needed to be computed during optimization as only the relative difference of energy between the states is used. Here, the relationship between Eq. (5) and Eq. (6) is that $Probability \propto exp(-\lambda Energy)$, assuming the Gibbs distribution. Here, λ is

a design parameter controlling the smoothness of the posterior distribution. Now, with Eq. (6), Eq. (4) can be re-written as

$$\hat{l} = \arg\max_{l} P(\mathbf{Y}_t|\mathbf{X}_{t,[l]}) = \arg\min_{l} E(\mathbf{Y}_t; \mathbf{X}_{t,[l]}) \ . \tag{7}$$

Also, the sample weights for particle filtering is now

$$w_{(l)} \propto P(\mathbf{Y}_t|\mathbf{X}_{t,[l]}) \propto \exp(-\lambda \ E(\mathbf{Y}_t; \mathbf{X}_{t,[l]})) \ . \tag{8}$$

Generally speaking, if λ is large, the posterior distribution becomes spiky and particles become concentrated near the MAP solution, whereas if λ is small, the posterior distribution becomes smooth and more particles far away from the MAP survive the re-sampling process. We empirically found that $\lambda = 10$ work well in most cases.

2.3. Modeling a Single Patch

In order to obtain the energy of a single patch $E(\mathbf{Y}_t; \mathbf{X}_t^k)$, we model each individual patch using a linear classifier in 21 dimensional space. The first nine dimensions are HOG (Histogram of Oriented Gradients) features. We build our HOG by dividing the orientation into eight equal bins, and one bin to denote gradients with response 0. To obtain image gradients, filter kernels $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$ and $\begin{bmatrix} -1 & 0 & 1 \end{bmatrix}^T$ are used. When applying these filters, if the responses were below 10 on a 0 to 255 scale, we considered the response to be 0 to increase robustness to noise. We then assigned the image gradients to one of the eight bins according to their orientations, or the ninth bin if both filter responses were 0. For simplicity, we assigned each gradient to one of the nine bins without weighing them.

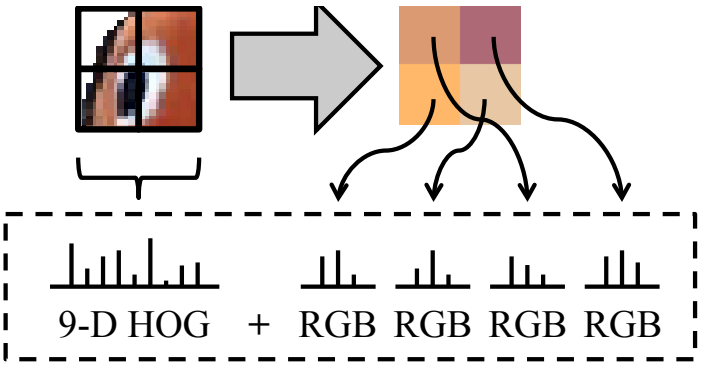
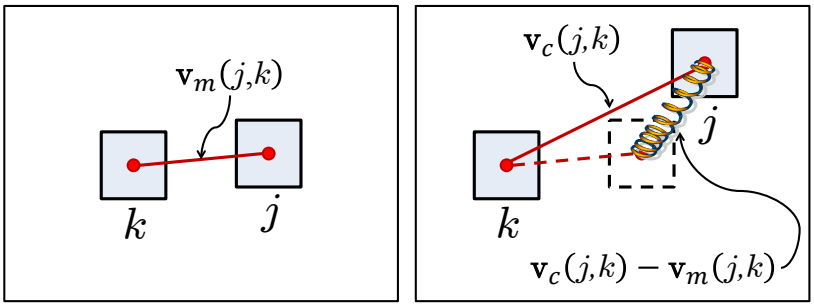


Fig. 2. Example of a 21 dimensional feature descriptor for a single local patch.

For the remaining 12 dimensions, we divide a single local patch into four equal parts (upper left, upper right, bottom left, and bottom right) and obtain the mean RGB values for each sub region as in Figure 2 (3 dimensions for each sub region). More sub regions may be used depending on the level of accuracy required for a single local patch tracking. This feature is similar to the one used in [28], but one feature is assigned to a single patch not a single pixel as in [28]. The advantage of using this 21 dimensional feature is that this feature can be obtained efficiently using integral images.

For each patch, we use the classifier score of the observed 21 dimensional vector to obtain the energy of a single patch, $E(\mathbf{Y}_t; \mathbf{X}_t^k)$ in (6). The classifier is trained so that it gives high scores when the observation is likely to be the modeled patch, and gives low scores (possibly negative) when it is not likely. For the classifier, we use linear SVM [24, 25] with logistic fitting performed on the classification score [26] to perform scaling. Training strategies for both linear SVM and logistic fitting are described in detail in Section 2.5.

If we let $\mathbf{f}_c^k$ denote the 21 dimension feature vector of the current observa-



(a) Model (b) Observation

Fig. 3. Example of neighboring local patches connected together.

tion for the k^{th} patch, let $s_k\left(\mathbf{f}_c^k\right)$ denote the linear SVM classification score for $\mathbf{f}_c^k$, and let A_k and B_k denote the learned logistic parameters, then

$$E(\mathbf{Y}_t; \mathbf{X}_t^k) = 1 - \frac{1}{1 + e^{A_k s_k\left(\mathbf{f}_c^k\right) + B_k}} \quad . \tag{9}$$

Note that the logistic fitting [26] scales the classifier scores to be in range $[0, 1]$, considering the distribution of scores from the training data. This prevents the problem of certain patches having higher priority than others due to different score range when using raw classifier scores. We also use the classification result of each patch when updating the model to prevent drifting (detailed in Section 2.5).

2.4. Modeling the Relationship between Patches

The relationship between neighboring patches is modeled so that the local structures among neighboring patches are preserved while tracking. To deal with non-rigid deformations, patches behave as if they are connected by springs. Also, to be robust to partial occlusions, the springs of each patch

behave as if they are connected to the patch's expected positions from its neighbors. As in Fig. 3, if we consider an observed patch (patch j in Fig. 3) and its neighbor (patch k in Fig. 3), then in our model, the observed patch j tends to return to its expected position from its neighbor k (expected position is denoted by the dotted box) by the restoring force of a virtual spring connection between the expected position and patch j, which is length zero at rest. In other words, the energy of the connection between the two neighboring patches k and j is defined as the elastic energy of this spring. If we denote the vector difference between j^{th} and k^{th} patches of the current observation and the model as $\mathbf{v}_c(j,k)$ and $\mathbf{v}_m(j,k)$, respectively, then the displacement change x of this virtual spring is

$$x = \|\mathbf{v}_c(j,k) - \mathbf{v}_m(j,k)\|_2 \quad . \tag{10}$$

Also, to make close patches have more effect on one another, the strength of this spring is designed to be inversely proportional to the squared distance between the neighbors. Therefore, the spring constant κ is designed as

$$\kappa = \frac{2}{\|\mathbf{v}_m(j,k)\|_2^2}\beta \quad , \tag{11}$$

where the neighbor strength β is a design parameter controlling the trade-off between the flexibility to adapt to non-rigid motion and the ability to keep the structure against occlusion. Details on the effect of β will be addressed in Section 3.1. Then, the elastic energy between connected local patches $E(\mathbf{X}_t^k, \mathbf{X}_t^j)$ in Eq. (6) is defined as

$$E(\mathbf{X}_t^k, \mathbf{X}_t^j) = \frac{1}{2}\kappa x^2 = \beta\frac{\|\mathbf{v}_c(j,k) - \mathbf{v}_m(j,k)\|_2^2}{\|\mathbf{v}_m(j,k)\|_2^2} \quad . \tag{12}$$

2.5. Model Update

The model of the target object needs to be updated consistently in order for the tracker to be able to adapt to various changes in the target object. Illumination changes and deformations of the target object must be learned by the model to correctly evaluate Eq. (5). In our case, the model update is performed in two steps: (1) updating the linear classifiers and the logistic parameters for each patch, and (2) updating the neighborhood connection parameters. To prevent the model from drifting, the update is performed only when the observed patch is classified as the model, *i.e.* for patch k, only when $s_k\left(\mathbf{f}_c^k\right) > 0$. Also in case of the neighborhood relationship, we only update when the observation for both patches forming the relationship are classified as the model.

The way we update the linear classifiers is through updating training samples. For each patch, we keep a pool containing positive and negative training samples of size $2M$ (M positive samples and M negative samples). The positive samples represent the target object and the negative samples are simply the 21 dimension feature vectors drawn randomly from nearby. At the initial frame, we initialize the positive pools with M identical copies of the initial patches of the target object. For each frame, after obtaining the local patch tracking results, we add the tracking results to the positive pools and take out the oldest samples from the positive pools. When taking samples out from the positive pools, to prevent drifting, we make sure that at least one sample is from the first frame (*i.e.* one sample representing the patch from the initial frame is never taken out for each pool). Then we completely discard the previous negative pools and refresh negative samples from random

nearby patches. Again, for each patch, the classifier from the previous frame is discarded and a new classifier is trained using the new training pools. Note that for each pool, since we update one positive sample at a time, the pool size M acts as a design parameter controlling the update speed of the target object model. When M is large the model is updated slowly and when M is small the model is updated quickly. In general, we empirically found that $M = 100$ gives good performance as well as low computational cost for the update process.

For the relationship update, we simply update $\mathbf{v}_m(j,k)$ by weighted averaging, but only when both patches are classified as the model. In other words, for all j and k, if

$$\left[s_j\left(\mathbf{f}_c^j\right) > 0\right] \wedge \left[s_k\left(\mathbf{f}_c^k\right) > 0\right] \quad , \tag{13}$$

then

$$\mathbf{v}_m\left(j,k\right) \leftarrow \frac{1}{M}\mathbf{v}_c\left(j,k\right) + \left(1 - \frac{1}{M}\right)\mathbf{v}_m\left(j,k\right) \quad . \tag{14}$$

Note that the learning rate is $\frac{1}{M}$ so that the update rate would be the same as for the training pools for the classifiers.

2.6. Hierarchical Diffusion

In our model, the dimension of the solution space is too large to apply simple motion models such as the random-walk model. The required number of particles grow exponential according to the number of patches used. For example, if we were to need 100 particles to track an object with only one patch using the random-walk motion model, then with N patches we would require 100^N particles to track the target object with our model. This is already an astronomical number even with only a few number of patches,

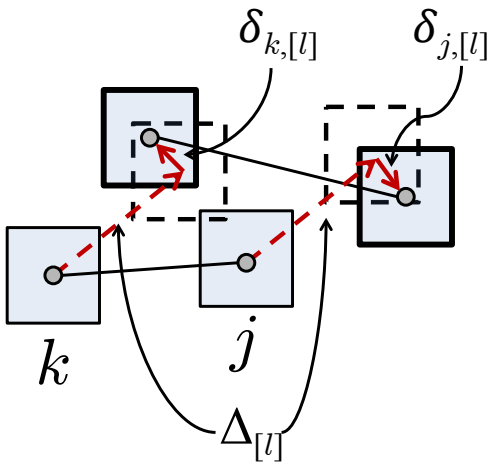


Fig. 4. Illustrative example of hierarchical diffusion performed for a single sample l. Global movement of the total configuration of patches $\Delta_{[l]}$ is sampled first, then local movements of individual patches $\delta_{k,[l]}$ and $\delta_{j,[l]}$ are sampled.

making our method impossible to run in real-time. Therefore, we propose an efficient hierarchical diffusion method.

To use a small number of samples, we focus on sampling from the region where the actual solution would exist with high probability. In case of tracking situations the deformation of the target object is not large between subsequent frames. Considering this as an assumption, we diffuse particles hierarchically in two steps: globally for the motion of the whole object and locally for the deformations of the target object. We first diffuse the position of all local patches equally according to the Gaussian distribution with a relatively large variance, and then diffuse the position of each patch separately according to the Gaussian distribution with a relatively small variance (illustrated in Fig. 4). In the global step, the samples are diffused so that the relative positions between local patches in the sample are preserved. Then,

in the local step, each local patch is diffused independently. Mathematically the proposed hierarchical diffusion method can be described as

$$\mathbf{X}^k_{t,[l]} = \mathbf{X}^k_{t-1,[l]} + \Delta_{[l]} + \delta_{k,[l]} \quad , \tag{15}$$

where, $\mathbf{X}^k_{t,[l]}$ denotes the state of the k^{th} local patch of the l^{th} sample at time t, $\Delta_{[l]}$ denotes the 2-dimensional global diffusion (translation in x, y direction for the whole object) for the l^{th} sample, and $\delta_{k,[l]}$ denotes the 2-dimensional local diffusion (translation in x, y direction for a local patch) for the k^{th} local patch of the l^{th} sample. Here,

$$\Delta_{[l]} \sim \mathcal{N}(0, \sigma_G^2) \quad , \tag{16}$$

and

$$\delta_{k,[l]} \sim \mathcal{N}(0, \sigma_L^2), \tag{17}$$

where $\mathcal{N}(0, \sigma^2)$ denotes a Gaussian distribution with zero mean and standard deviation σ. σ_G and σ_L are parameters for the diffusion. The optimal choice of σ_G and σ_L may vary depending on the image sequence but we empirically found that $\sigma_G = 8$ and $\sigma_L = 4$ works well for most cases. The proposed diffusion produces an accurate solution with a relatively small number of particles compared to the simple random walk approach, which allows the proposed method to achieve real-time performance. Details and discussion on experimental results regarding the effectiveness of hierarchical diffusion are given in Section 3.8.

2.7. Summary of the Proposed Method

The proposed method uses particle filtering to get an MAP solution for the object tracking problem. Given the initial patches and connections

$\left\{\mathbf{f}_m^k, \mathbf{v}_m(j,k); \forall j,k, \right\}$, the proposed method can be summarized as **Algorithm 1**.

Algorithm 1 Tracking with Local Patches (for each frame)

1: For each sample, compute

$E(\mathbf{Y}_t; \mathbf{X}_t) = Z + \sum_{k=1}^{m} \left[E(\mathbf{Y}_t; \mathbf{X}_t^k) + \sum_{j \in N_k} E(\mathbf{X}_t^k, \mathbf{X}_t^j) \right]$ (Eq. (6))[1]

where,

$E(\mathbf{Y}_t; \mathbf{X}_t^k) = 1 - \left(1 + e^{A_k s_k(\mathbf{f}_c^k) + B_k}\right)^{-1}$ (Eq. (9))

$E(\mathbf{X}_t^k, \mathbf{X}_t^j) = \beta \frac{\|\mathbf{v}_c(j,k) - \mathbf{v}_m(j,k)\|_2^2}{\|\mathbf{v}_m(j,k)\|_2^2}$ (Eq. (12))

2: Find MAP solution

$\hat{\mathbf{X}}_t = \mathbf{X}_{t,[\hat{l}]}$ (Eq. (3))

where,

$\hat{l} = \arg\max_l P(\mathbf{Y}_t | \mathbf{X}_{t,[l]}) = \arg\min_l E(\mathbf{Y}_t; \mathbf{X}_{t,[l]})$ (Eq. (7))

3: Update object model (Section 2.5)

4: Assign sample weights $w_{[l]}$ according to the likelihood

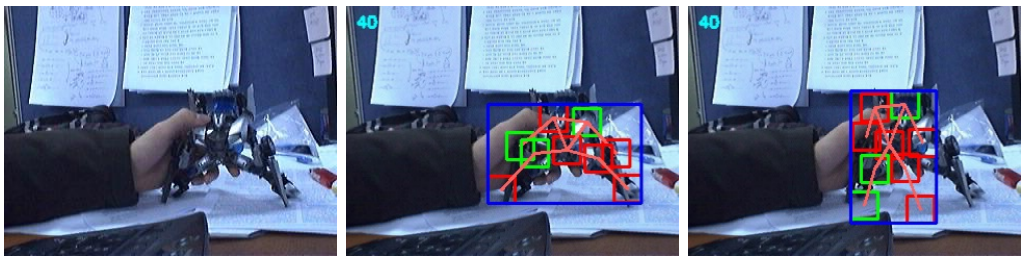
$w_{[l]} \propto P(\mathbf{Y}_t | \mathbf{X}_t) \propto \exp(-\lambda\, E(\mathbf{Y}_t; \mathbf{X}_{t,[l]}))$ (Eq. (8))

5: Re-sample according to weights

6: Diffuse samples

$\mathbf{X}_{t+1,[l]}^k = \mathbf{X}_{t,[l]}^k + \Delta_{[l]} + \delta_{k,[l]}$ (Eq. (15))

[1] Note that Z is a constant and can be safely omitted from the actual computation when optimizing.



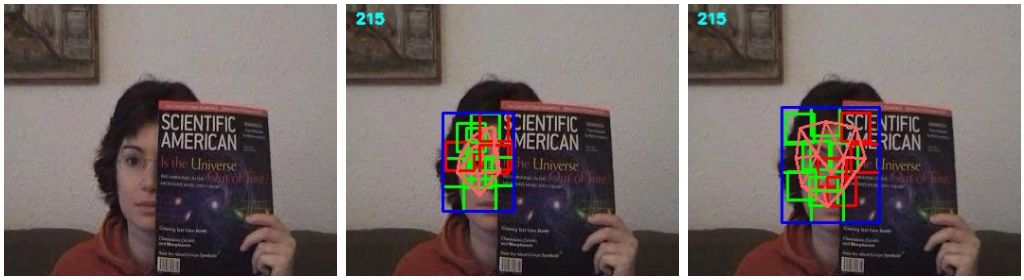
(a) Input image (b) $\beta = 0.2$ (c) $\beta = 2.0$

Fig. 5. Example showing the effect of β parameter on non-rigid object tracking. Tracking results for the **Robot** sequence, frame #40, with $\beta = 0.2$ (b) and $\beta = 2.0$ (c). Blue rectangle is the final tracking result, green rectangles are local patches with high confidence (SVM score above 0), red rectangles are local patches with low confidence, and the orange lines denote the neighborhood relationships.

3. Experiments

3.1. Parameter Effects

The parameter β in subsection 2.2 controls the strength of the neighborhood connections. In other words, large β means that tracking is performed so that the local structure does not change much. Thus, β is a parameter controlling the tradeoff between the tracker's ability to track non-rigid objects and ability to cope with partial occlusions. Fig. 5 is an example showing the effect of the β parameter when tracking object with deformation. As in Fig. 5 (b), if parameter β is small, object deformation is well tracked. On the other hand, if β is large, deformation is less taken into accounted when tracking. For our method, β is the only parameter which requires tuning. In case of other parameters, we found empirically that the parameters noted in the beginning of Section 3.2 works well in most situa-



(a) Input image (b) $\beta = 0.2$ (c) $\beta = 2.0$

Fig. 6. Example showing the effect of β parameter on tracking objects with partial occlusions. Tracking results for the **Face** sequence, frame #215, with $\beta = 0.2$ (b) and $\beta = 2.0$ (c).

tions and without tuning. Unlike the case for traditional particle filtering methods [8, 9, 10, 13] which require the tuning of the diffusion parameters in each dimension when changing the trackers' behaviors, our method only requires tuning of β. Also, tuning β is intuitive and simple. Fig. 6 shows the effect of parameter β when tracking objects with partial occlusions. The tracker better handles occlusions when β is large, as in Fig. 6 (c). Generally, for scenes with high deformation $\beta = 0.2$ shows good results, for scenes with heavy occlusion $\beta = 2.0$, and normally $\beta = 1.0$ is good enough.

3.2. Performance Evaluation

Evaluation of the proposed method was performed through twelve image sequences. Each image sequence consists of different types of situations (occlusion, outer-plane motion, non-rigid deformation, etc.) Throughout the experiments, all parameters except β were fixed. The pool size M was set to 100, and the number of particles was set to 1000. For the linear SVM,

we used default parameters provided in [25]. For the sampling parameters, $\sigma_G = 8$ and $\sigma_L = 4$. Parameter λ controlling the concentration of samples, was set to 10. The implementation was done in C++ without any parallel processing. All experiments were held on a desktop PC (Intel core i5-2500 3.3GHz, Windows 7) and ran comfortably about 20-50 frames per second depending on the number of patches, except for the **Dudek** sequence which was of size 720×480, and ran 12 frames per second.

The test sequences are composed of some well-known sequences and some of our own. The **Dudek** sequence and the **Sylvester** sequence are from the work of Lim *et al*. [8], and the **Face** sequence and the **Woman** sequence are from the [12]. The **Caviar** sequence is from the CAVIAR[2] dataset. These five sequences are some of the well-known sequences for evaluating tracking performances. The **High Jump** sequence is from Kwon *et al*.'s work [5], and the **Motocross 1** and the **Mtn. Bike** sequences are from recent work by Godec *et al*. [14]. The **Robot** sequence and the **Pedestrian** sequence are from [23], and the **Dove** sequence is from [29]. Finally, the **Nemo** sequence is of our own. The resolution for the **Dudek** is 720×480, the **Face** and the **Woman** are 352×288, the **Caviar** is 384×288, the **High Jump** is 416×234, the **Motocross 1** and the **Mtn. Bike** is 640×360, and others are 320×240.

For quantitative analysis, we compared the mean error of the four corner points of the tracking bounding box. The ground truth data was annotated by human hand, so that the target object fitted in the bounding box. In [14],

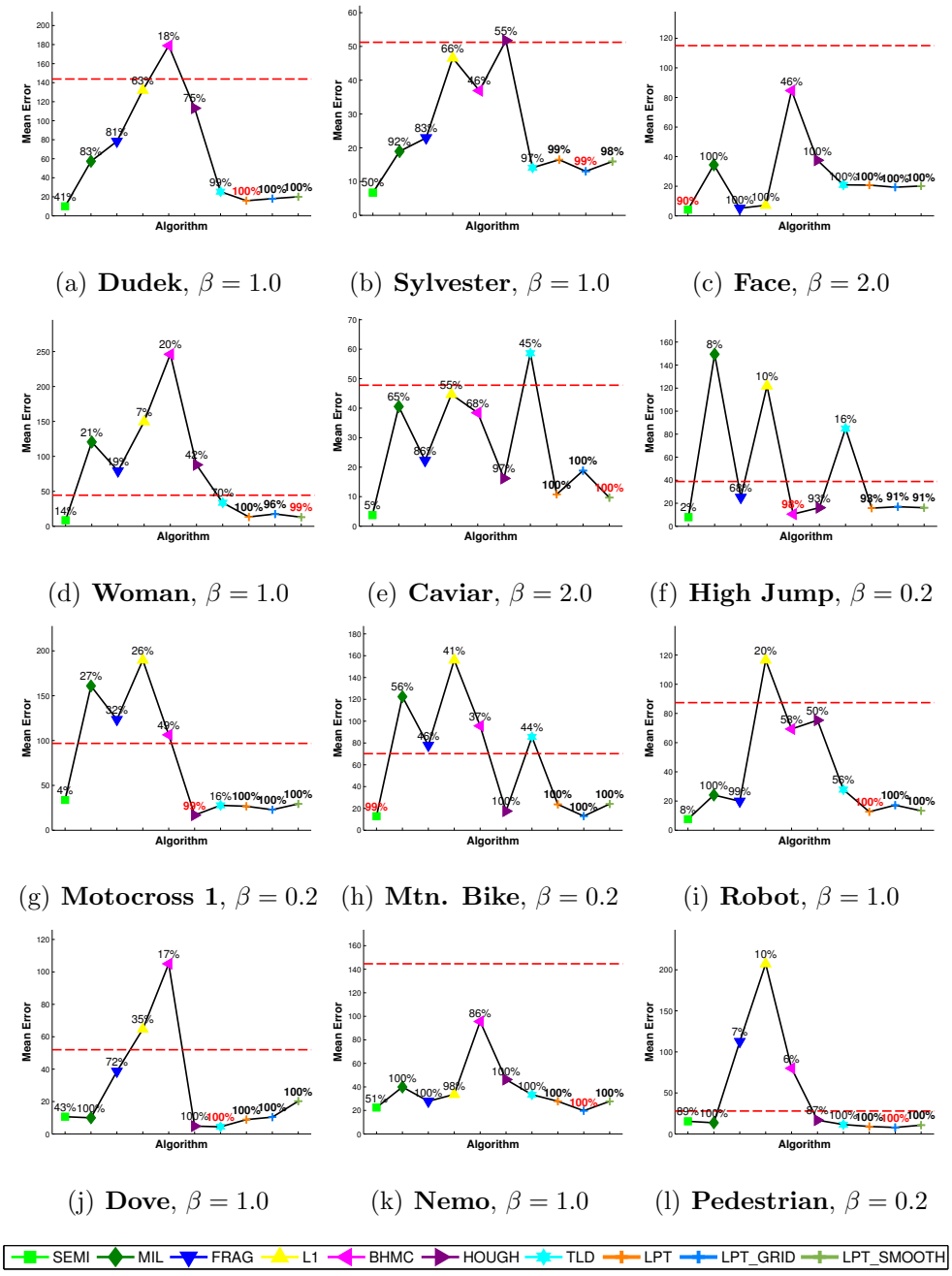
[2]EC Funded CAVIAR project/IST 2001 37540, found at URL: http://homepages.inf.ed.ac.uk/rbf/CAVIAR/

Godec *et al.* used the Agarwal-criterion [30], which was defined as $score = \frac{R_T \cap R_{GT}}{R_T}$, where R_T is the tracking rectangle and R_{GT} the ground truth, but this measure is not suitable for describing how accurate tracking is. A single dot in the ground truth region would return maximum measure. However, by using the mean of the errors of the four corner points of the tracking bounding box we can measure the accuracy of a tracker without such problem. Also, this measure is easily applicable to many trackers since most trackers are based on giving a bounding box of the target object as a result.

The proposed method has been compared against seven other trackers. Beyond Semi-supervised Tracking (SEMI) in [17] and Multiple Instance Learning (MIL) in [16] are methods based on the concept "tracking by detection", Frag-Track (FRAG) in [12] and l_1 minimization (L1) in [13] are some representative methods for solving occlusion problems, Basin Hopping Monte Carlo Tracking (BHMC) in [5] and Hough-based Tracking (HOUGH) in [14] are state-of-the-art methods for solving non-rigid object tracking, and TLD Tracking (TLD) [3] is a method which uses both detectors and trackers. For the experiments, we used the implementation provided by the authors of each paper. Also, we implemented our method in three different ways. The first is with manual initialization (LPT), the second is with initialization by dividing the target bounding box into equal grids (LPT_GRID), and the last one is with simple temporal low-pass filtering to LPT (LPT_SMOOTH) to remove noise from estimation. For LPT_GRID, we assumed the patches were connected to its direct neighbors (patches which share boundaries) and the number of grids was set to 3×3. For LPT_SMOOTH, the temporal smoothing was performed by weighted averaging the new estimated result and the

old estimate (the tracking result from previous frame). When averaging, we applied different weights for the smoothing of the center position and the smoothing of the width and height. For the position, a weight of 0.3 was applied to the new estimate and 0.7 for the previous estimate. For the width and height, we applied stronger smoothing than we did for the position since width and height do not change much between consecutive frames; weight of 0.1 to the new estimate and 0.9 to the previous estimate. Effects of this temporal smoothing will be discussed in Section 3.9. All other parameters were identical for all three implementations.

Fig. 7 shows the mean error value of each tracker for each image sequence. β was set as in the sub-captions. The mean error value was calculated only for the frames the tracker returned results. This is because SEMI and TLD returned results only when they are confident, and if they are not, returned results indicating tracking failures. The number on top of each marker denotes the percentage of frames that gave meaningful results, which mean that, if we denote the mean error of the four corner points as e_{mean}, width of the ground truth as w_{GT}, and height of the ground truth as h_{GT}, then $e_{mean} < \min(w_{GT}, h_{GT})$. The tracker with lowest mean error and with over 90% of the tracking results meaningful is marked with red bold text. The red dotted horizontal line denotes the mean value of $\min(w_{GT}, h_{GT})$ throughout each sequence. Tracker with mean error above the dotted red line means that most of the tracking results from that tracker were meaningless for that sequence. In general, all three implementations of our method consistently outperform or show comparable results against other compared methods. Note that although other methods sometimes give better results than ours



(a) **Dudek**, $\beta = 1.0$ (b) **Sylvester**, $\beta = 1.0$ (c) **Face**, $\beta = 2.0$

(d) **Woman**, $\beta = 1.0$ (e) **Caviar**, $\beta = 2.0$ (f) **High Jump**, $\beta = 0.2$

(g) **Motocross 1**, $\beta = 0.2$ (h) **Mtn. Bike**, $\beta = 0.2$ (i) **Robot**, $\beta = 1.0$

(j) **Dove**, $\beta = 1.0$ (k) **Nemo**, $\beta = 1.0$ (l) **Pedestrian**, $\beta = 0.2$

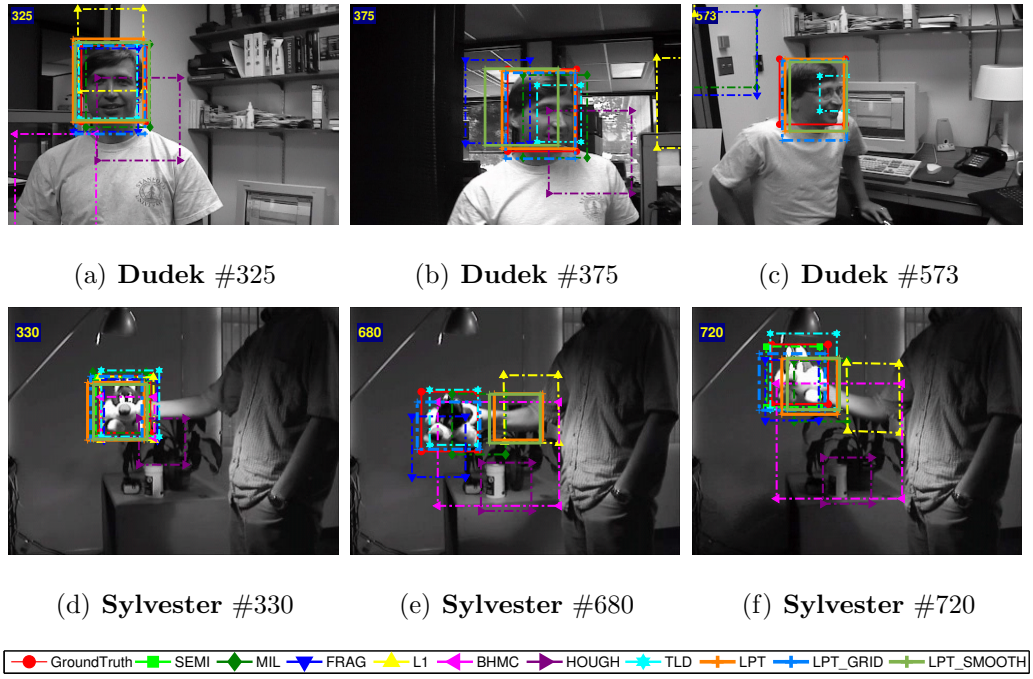
Fig. 7. Mean errors for each sequence. Numbers on top of each marker denote the percentage of meaningful tracking results. Best tracker denoted by red bold text.

depending on the image sequence, our method consistently gives good results, regardless of the image sequence used. Also, even though our method is based on sampling, we did not find significant difference in the results between multiple runs with the same parameters.

3.3. Discussion on Translation, Rotation, and Illumination Changes

The **Dudek** sequence and the **Sylvester** sequence are well known datasets for testing robustness on translation, rotation, and illumination changes. In both image sequences, SEMI gave the most accurate result (Fig. 7 (a) and (b)). However, in both sequences, SEMI was able to track less than half of the whole sequence, whereas our method (LPT) was able to track most of the sequence (100% for **Dudek** and 99% for **Sylvester**) with promising results. For the **Sylvester** sequence, LPT_GRID and LPT_SMOOTH show similar results.

Critical frames for the two sequences are shown in Fig. 8. In Fig. 8 (a), as the target person stands up, L1, BHMC, and HOUGH loses track. Also, as the person turns around in Fig. 8 (b), FRAG fails whereas our method successfully tracks the whole sequence. In Fig. 8 (b), LPT_SMOOTH also drifts a bit due to the abrupt change in motion, but still keeps track of the target object and recovers after a few frames. In Fig. 8 (d), we can see that HOUGH starts to drift off. The authors of [14] reported that they were able to track 99% of the image sequence, but even with same initial conditions the authors provided, we were not able to reproduce their result (we obtained slightly degraded results). We suspect that this is because HOUGH uses fully randomized trees, meaning that the accuracy of their algorithm may



(a) **Dudek** #325 (b) **Dudek** #375 (c) **Dudek** #573

(d) **Sylvester** #330 (e) **Sylvester** #680 (f) **Sylvester** #720

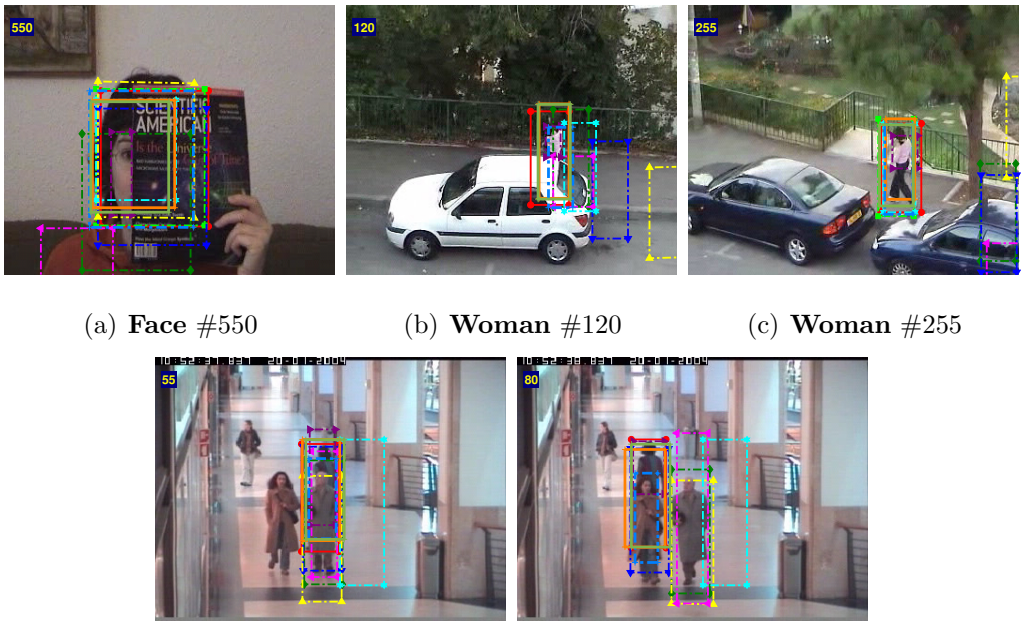
Fig. 8. Tracking results for the **Dudek** sequence and the **Sylvester** sequence. *Best viewed in color.*

vary according to the random seed. For all image sequences, even with the same initial settings, HOUGH returned various results, making the tracking accuracy unstable. In Fig. 8 (e), we can see that our method also fails due to large outer-plane motion, but our method quickly recovers as shown in Fig. 8 (f), giving good overall performance considering all frames.

3.4. Discussion on Partial Occlusions

The sequences **Face**, **Woman**, and **Caviar** are sequences which contain partial occlusions. For the **Face** sequence, all three implementations of our method gave comparable results against other methods (Fig. 7 (c), (d), and (e)). Moreover, our method was also capable of tracking objects showing non-rigid deformations, whereas methods showing good performances on this sequence (SEMI, FRAG, and L1) did not show good performances in other situations (**High Jump**, **Motocross 1**, and **Mtn. Bike**). Also, note that the methods designed for non-rigid object tracking (BHMC and HOUGH) do not show good results for **Face** and **Woman**. For the **Face** and **Woman** sequences, the target object is occluded gradually and severely, where some parts of the sequence have over half of the target occluded.

Critical frames for these sequences are shown in Fig. 9. In Fig. 9 (b), as occlusion occurs, we can see other methods failing, whereas our methods, LPT, LPT_GRID, and LPT_SMOOTH all three, successfully track. As in Fig. 9 (c), TLD re-detects the target object when the occlusion is gone, but as the target object gets occluded again, the tracker fails. This sequence was used in [12], and FRAG was able to track the target object throughout the whole sequence in their paper. However, in [12], only a portion of the whole



(a) **Face** #550 (b) **Woman** #120 (c) **Woman** #255

(d) **Caviar** #55 (e) **Caviar** #80

Fig. 9. Tracking results for the **Face** sequence, the **Woman** sequence, and the **Caviar** sequence. *Best viewed in color.*

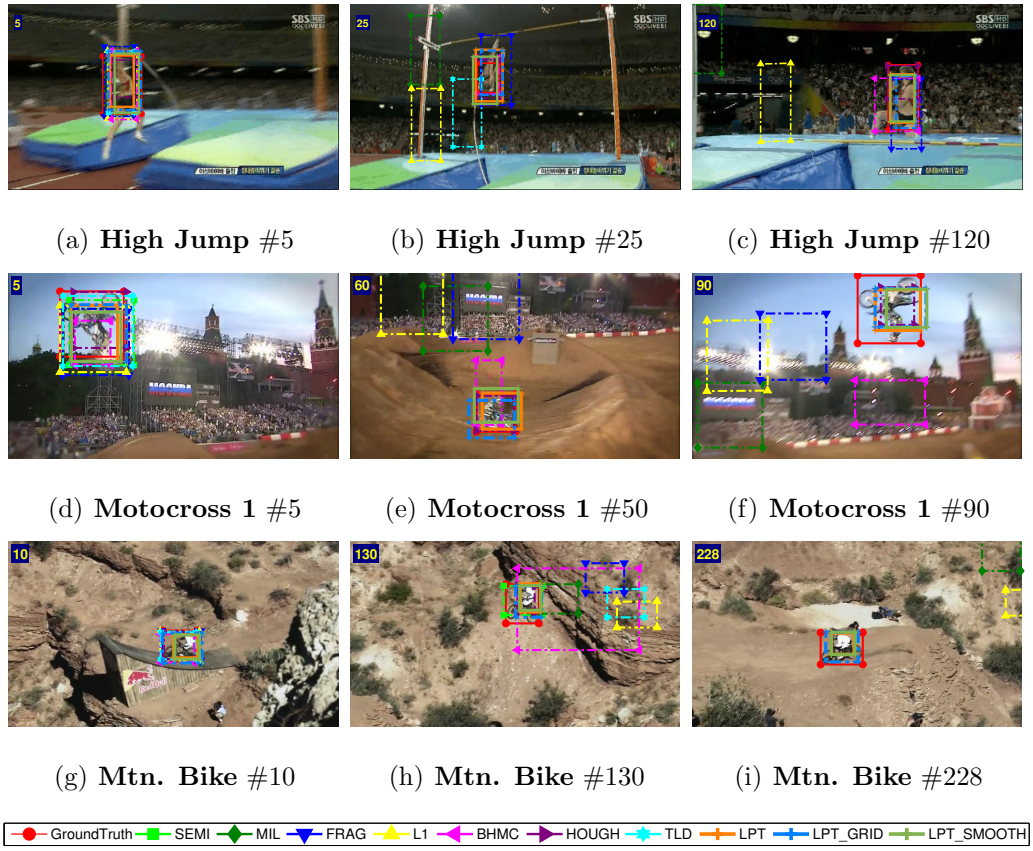
sequence was used. When started from the first frame, FRAG loses track as well.

3.5. Discussion on Non-Rigid Deformations

The sequences **High Jump**, **Motocross 1**, and **Mtn. Bike** are sequences with non-rigid deformations. For all sequences, all three implementation of our method gave promising results (Fig. 7 (f), (g), and (h)). BHMC and HOUGH show good results for objects with non-rigid deformations, but do not show good results in general (especially for sequences with occlusions) and run only a few frames per second, whereas our method runs 20 to 50 frames per second. Note that the trackers showing good performance against partial occlusions (FRAG and L1) tend to show unsatisfactory results in these cases, whereas our method consistently gives good results. Critical frames for these sequences are shown in Fig. 10.

3.6. Discussion on Additional Cases

The **Robot** sequence contains both non-rigid deformations and partial occlusions. The **Dove** sequence show fast object motion, and the **Nemo** sequence has scale changes and in-plane rotations. Finally, the **Pedestrian** sequence is with fast and abrupt camera movements.Some critical frames for these sequences are shown in Fig. 11. For the **Robot** sequence, as in Fig. 7 (i), the proposed method outperformed all other trackers (SEMI showed lower mean error, but was able to track only 7.7% of the sequence). Also, for the sequences **Dove**, **Nemo**, and **Pedestrian**, both LPT and LPT_GRID outperformed or showed comparable results against other trackers. In case of LPT_SMOOTH, the accuracy degraded for the **Dove** sequence due to the



(a) **High Jump** #5 (b) **High Jump** #25 (c) **High Jump** #120

(d) **Motocross 1** #5 (e) **Motocross 1** #50 (f) **Motocross 1** #90

(g) **Mtn. Bike** #10 (h) **Mtn. Bike** #130 (i) **Mtn. Bike** #228

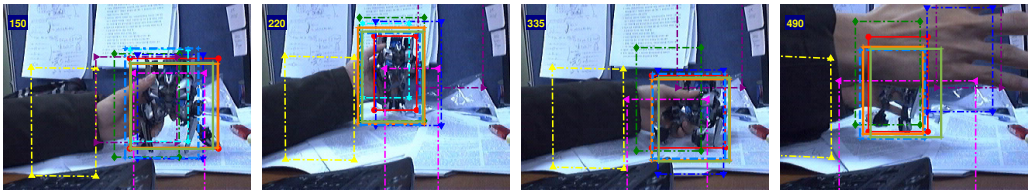
Fig. 10. Tracking results for the **High Jump** sequence, the **Motocross 1** sequence, and the **Mtn. Bike** sequence. *Best viewed in color.*

fast movement of the target object (example shown in Fig. 11 (g)), but is still comparable to other trackers. Note that in Fig. 11 (a), (c), and (f), the toy shows complex movements (spreading legs apart, bending, and sitting down), which other trackers fail to describe. Since the neighborhood connections in the grid initialization are not accurate, LPT_GRID also fails to describe the movement of the target object in this case. In Fig. 11 (d) and (e), LPT shows robust performance against severe partial occlusions.

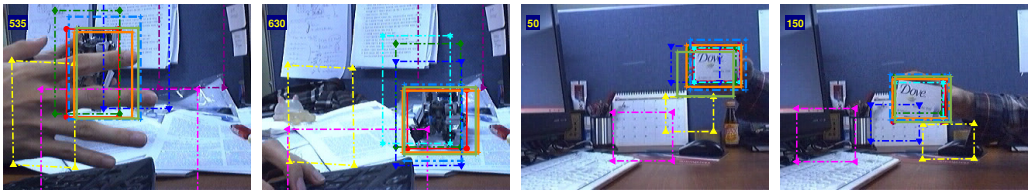
3.7. Summary of Tracking Results

The evaluation results are summarized in Table 1. The results in this table are obtained by concatenating frame-by-frame results of all sequences together, so that the number of frames of each sequence is taken into account. Note that since the percentage of meaningful tracking results considers the target object size, the scale differences of each sequence is also considered in this measure. The mean error values can be understood as expected error in pixels, regardless of the target object size. Also, we have roughly evaluated the real-time capability (over 15 FPS on 320×240 image sequence) of each method. We did not compare exact run-time of each method since each implementations had different level of optimization (from MATLAB to parallel processing using multi-processing techniques). Roughly speaking, L1 and BHMC require a few seconds per frame, SEMI and HOUGH is between few to ten frames per second, and others are over 15 frames per second. Note that our method provides $20 \sim 50$ FPS in C++, with no parallel processing.

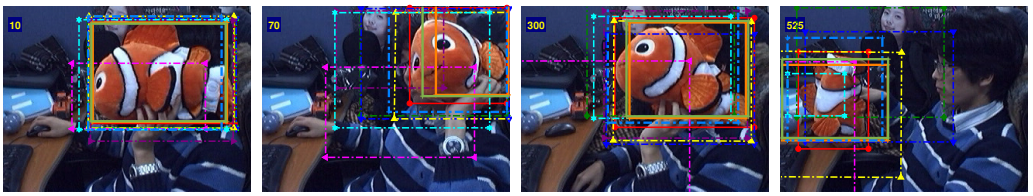
As illustrated in Table 1, our method (LPT) outperforms all other trackers we compared against, with respect to the percentage of meaningful tracking



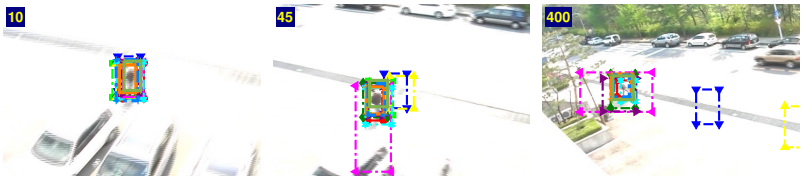
(a) **Robot** #150 (b) **Robot** #220 (c) **Robot** #335 (d) **Robot** #490



(e) **Robot** #535 (f) **Robot** #630 (g) **Dove** #50 (h) **Dove** #150



(i) **Nemo** #10 (j) **Nemo** #70 (k) **Nemo** #300 (l) **Nemo** #525



(m) **Pedestrian** #10 (n) **Pedestrian** #45 (o) **Pedestrian** #400

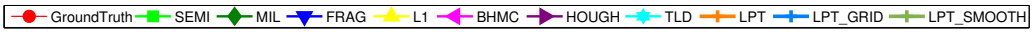


Fig. 11. Tracking results for the **Robot** sequence, the **Dove** sequence, the **Pedestrian** sequence, and the **Nemo** sequence. *Best viewed in color.*

Table 1

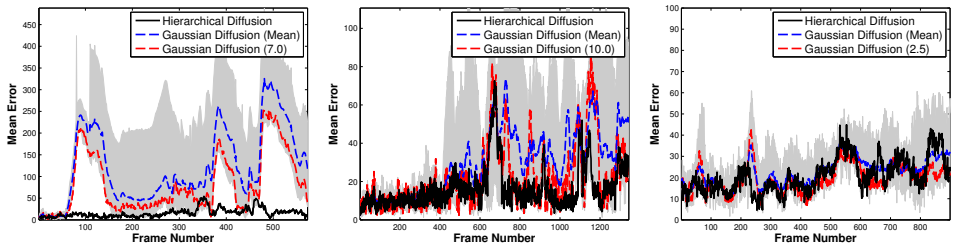
The mean error values and the percentage of meaningful tracking results with all frames in all image sequences for each algorithm. Last column denotes whether the average frames per second (FPS) for a 320×240 image sequence is over 15 (roughly real-time on webcams). Red underlined text indicates best result and the plain blue text indicates second best.

	Mean Error	% Meaningful	Over 15 FPS
SEMI	10.10	48.43	NO
MIL	48.57	82.11	YES
FRAG	43.51	73.04	YES
L1	88.00	54.11	NO
BHMC	97.80	42.93	NO
HOUGH	53.53	74.57	NO
TLD	26.52	83.86	YES
LPT	17.03	99.48	YES
LPT_GRID	16.03	99.14	YES
LPT_SMOOTH	17.85	99.33	YES

results. Except for SEMI, our method (LPT_GRID) also outperforms all other compared methods in terms of mean error as well. SEMI showed the lowest mean error, but showed the worst result when considering the percentage of meaningful tracking results. Note that as shown in Fig. 7, in few cases, our method perform slightly worse than some methods depending on the image sequence. However when all frames in all sequences are considered, our method outperforms all methods we compared against. This means that our method gives consistently good results among all sequences. The grid initialization version of our method, LPT_GRID, shows similar results as LPT, as in many tracking situations, the local structure of the target object is preserved, and the grid configuration is accurate enough for the tracker to work. Also, LPT_SMOOTH shows slightly degraded performance, but gives more stable results than LPT (see Section. 3.9 for details on the stability of the estimated tracking result).

3.8. Effectiveness of Hierarchical Diffusion

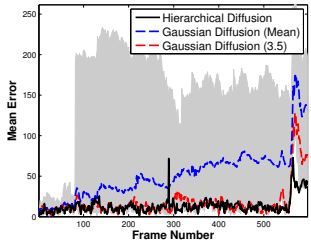
To demonstrate the effectiveness of hierarchical diffusion, we have compared the mean error of the tracking results obtained with hierarchical diffusion and with simple Gaussian diffusion (the random-walk motion model). During the experiment, to compare only the effect of different diffusion strategies, all parameters including the number of particles were fixed except for the diffusion parameters. For hierarchical diffusion, $\sigma_G = 8$ and $\sigma_L = 4$, whereas for simple Gaussian diffusion, we varied σ from $\sigma = 0.5$ to $\sigma = 10$ having 0.5 as step size. Comparison results are shown in Fig. 12. In Fig. 12, results of hierarchical diffusion are denoted by the solid black lines, the mean results of



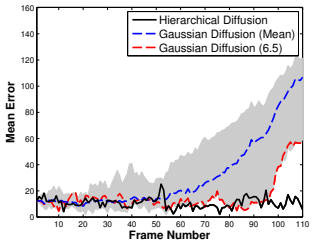
(a) **Dudek**

(b) **Sylvester**

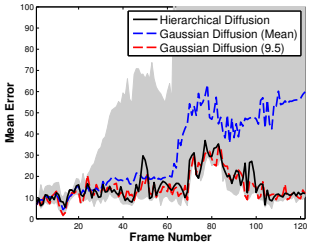
(c) **Face**



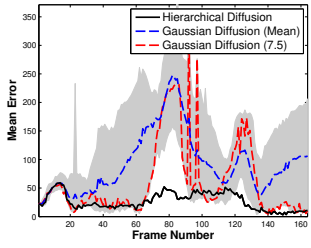
(d) **Woman**



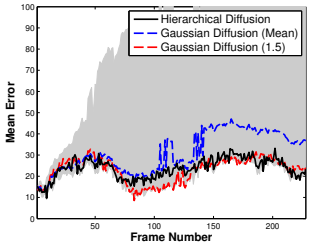
(e) **Caviar**



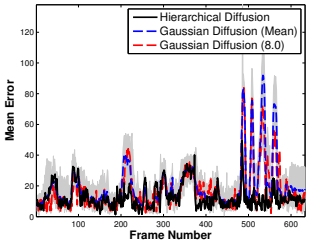
(f) **High Jump**



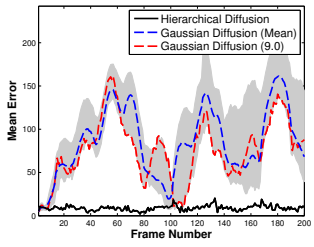
(g) **Motocross 1**



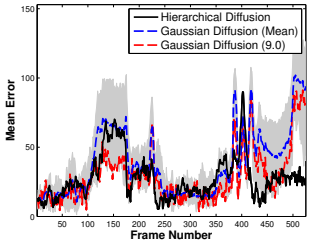
(h) **Mtn. Bike**



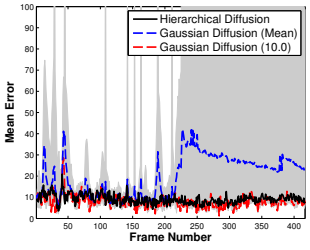
(i) **Robot**



(j) **Dove**



(k) **Nemo**



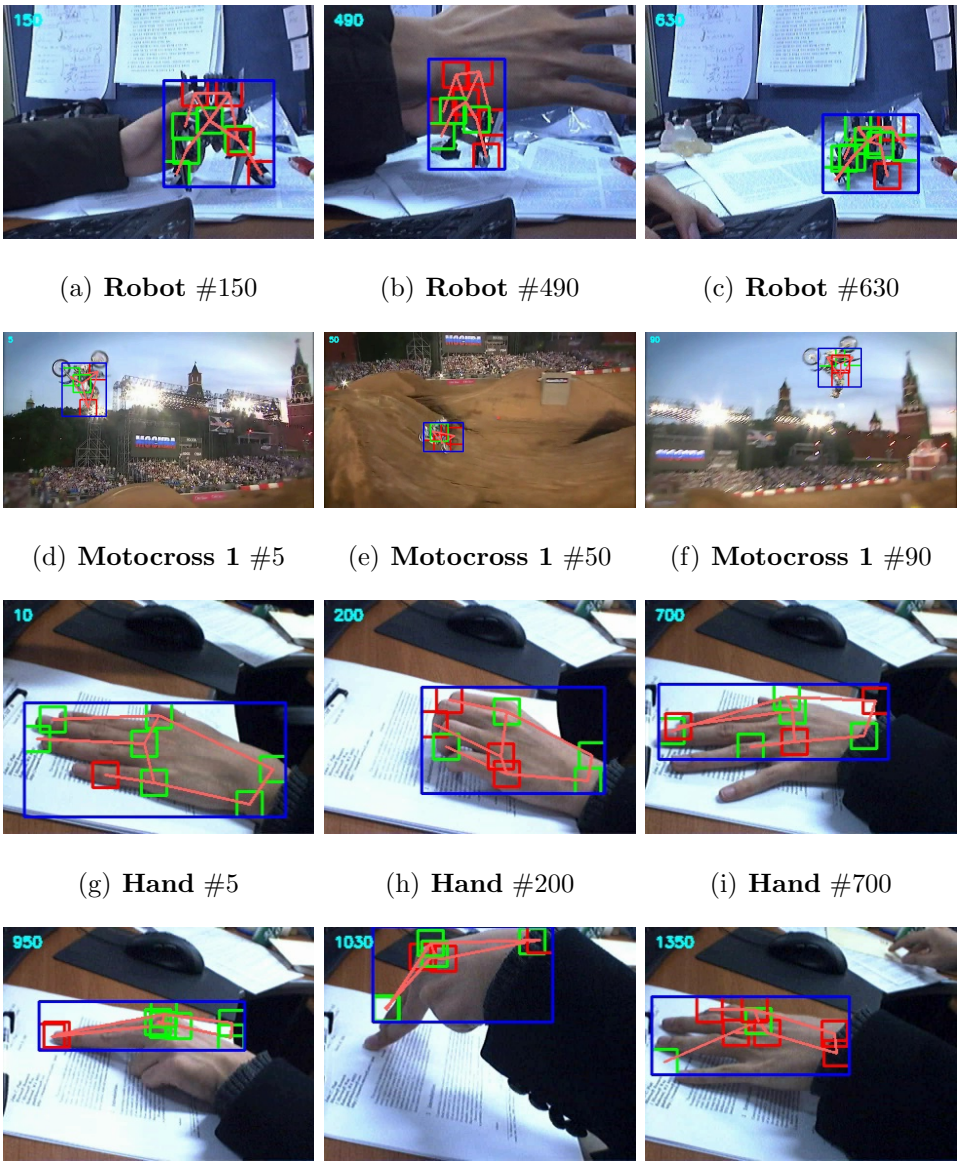
(l) **Pedestrian**

Fig. 12. Mean error obtained using hierarchical diffusion and simple Gaussian diffusion. See text for details.

simple Gaussian diffusion are denoted by blue dashed lines, best results using simple Gaussian diffusion are denoted by red dashed lines (with the best parameter in brackets), and the range in which results of simple Gaussian diffusion lie in are denoted with the grey area. As shown in Fig. 12, the proposed hierarchical diffusion generally gives better tracking results with the same number of particles. Note that with fine-tuned parameter for simple Gaussian diffusion, there are cases which simple Gaussian diffusion provides results as good as or even better than the proposed method. However, note that their parameters are all different, and these are cases when global motion of the target object is not large. When global motion is significant, as in Fig. 12 (a), (g), and (j), hierarchical diffusion achieves lower mean error regardless of the choice of parameter for simple Gaussian diffusion. Also, note that for the proposed hierarchical diffusion, the same diffusion parameters were used for all sequences.

3.9. Limitations

Though our method outperforms other methods in terms of mean error and the percent of meaningful frames, there are some limitations to the proposed method. The first limitation is the accuracy of individual local patches. The accuracy of individual patches may not be as good when large deformations exist or when the local patch is not very distinctive from its surroundings. However, when all local patches are considered together, the overall tracking result for the whole object is quite accurate owing to the proposed elastic structure. Example tracking results and their local patch structures are shown in Fig. 13. As shown in Fig. 13 (a) - (c), the position



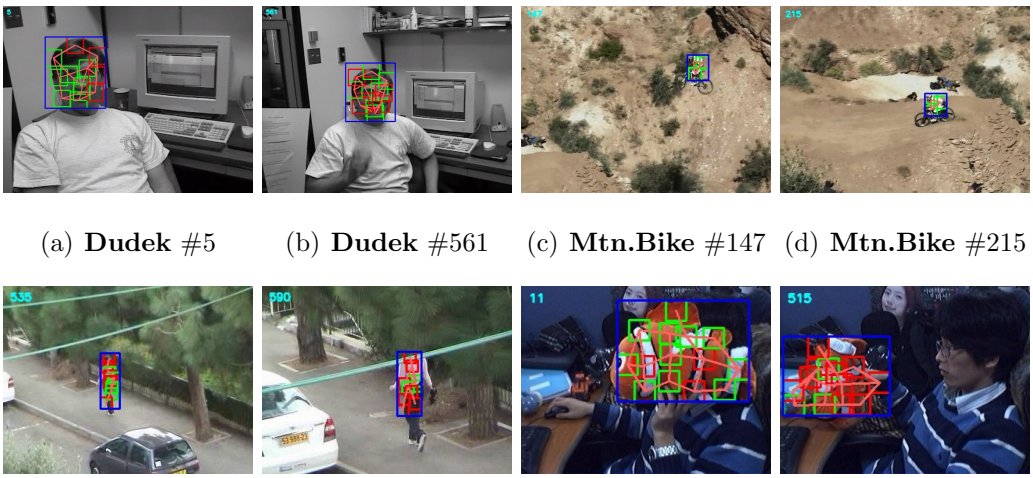
(a) **Robot** #150 (b) **Robot** #490 (c) **Robot** #630

(d) **Motocross 1** #5 (e) **Motocross 1** #50 (f) **Motocross 1** #90

(g) **Hand** #5 (h) **Hand** #200 (i) **Hand** #700

(j) **Hand** #950 (k) **Hand** #1030 (l) **Hand** #1350

Fig. 13. Example tracking results and their local patch structures. *Best viewed in color.*



(a) **Dudek** #5 (b) **Dudek** #561 (c) **Mtn.Bike** #147 (d) **Mtn.Bike** #215

(e) **Woman** #535 (f) **Woman** #590 (g) **Nemo** #11 (h) **Nemo** #515

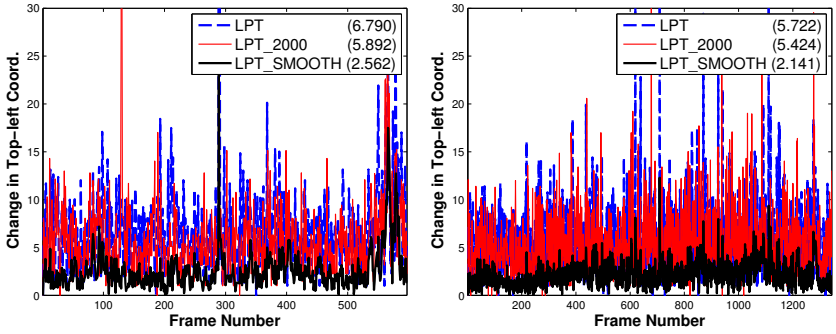
Fig. 14. Example tracking results with scale and orientation changes. *Best viewed in color.*

of each individual patch is quite accurate when the patches are discriminant from its surroundings. In Fig. 13 (d) - (f), the patch near the head of the person drifts and the position of the patches is not as accurate as in Fig. 13 (a) - (c) due to large deformation and fast motion. An extreme case for the individual patch accuracy limitation is shown in Fig. 13 (g) - (l). In this hand tracking sequence, the target object is highly deformative and the local patches look similar to their surroundings. As a result, patches in the lower part of the hand drift. However, note that the majority of the hand is still tracked when considering the entire configuration (the blue rectangle), which is one of the advantages of the elastic structure.

The second limitation is related to the robustness of the proposed method against scale and orientation changes. The proposed method models the

movement of individual patches considering translational movements only. In general, this does not cause major problems since minor scale and orientation changes can be described as the change in elastic structure. Example of the proposed method adapting to minor scale change is shown in Fig. 14 (a) and (b), and example of the proposed method tracking a target object with minor orientation change is shown in Fig. 14 (c) and (d). Also, even if the structure fails to describe the change, a few local patches with good matching scores are enough to track the target object. Fig. 14 (e) - (h) are examples of such failures. In Fig. 14 (e) and (f), the target object undergoes a drastic scale change as the camera zooms in. Our method fails in adapting to the fast scale change, but still does not lose track of the target object. In Fig. 14 (g) and (h), the target object undergoes large in-plane orientation change. Our method also fails to accurately describe the target object motion in this case. However, note that in both cases, though the accuracy of the estimate may decrease, our method does not lose track of the target object, owing to a few patches with strong matches directing the entire structure to the correct position.

The last limitation is the noisy nature of the estimated tracking result. Since our problem space is high-dimensional, even with an efficient diffusion scheme, the number of particles is relatively scarce when trying to obtain a real-time solution. As a result, we end up with noisy estimates, since particles are relatively positioned sparsely. As shown in Fig. 15 with the dashed gray lines, this leads to abrupt changes in the estimated position. However, as shown in Table 1, when all frames are considered, the results are good with small mean error on average. Thus, we can simply apply a temporal low-



(a) **Sylvester** (b) **Woman**

Fig. 15. Change in the coordinates of the top-left corner point of the estimated bounding box for each frame. LPT_2000 is when we use 2000 samples, which is double the number compared to LPT. Numbers in brackets are the average value for all frames. Note that the smoothed version (LPT_SMOOTH, solid black line) shows much less change than both the original (LPT, blue dashed line) and using more particles (LPT_2000, red solid line).

pass filtering to reduce the noise in the estimate, which is LPT_SMOOTH. As shown in Fig. 15 with the black solid lines and overall performance in Table 1, this simple low-pass filtering reduces the abruptness in the tracking result greatly without much harm in the performance. Note that we can also increase the number of samples to reduce this abruptness, but due to the high dimensionality of our problem formulation, this is not very effective as shown in Fig. 15 with LPT_2000 (doubling the number of samples).

4. Conclusions and Future Work

A new tracking method based on sequential Bayesian inference has been proposed. The proposed method tackled both the problem of partial occlusions and non-rigid deformation when tracking objects, by modeling the target object with an elastic structure of local patches, and by performing

hierarchical diffusion in the solution space. By modeling the target object with an elastic structure of local patches, the proposed method was able to track objects with partial occlusions and non-rigid deformations. Also, through hierarchical diffusion, the tracking procedure was performed in real-time on a desktop PC. The method was evaluated against state-of-the-art trackers through twelve image sequences with large occlusions and non-rigid deformations. The experimental results showed that the proposed method outperformed all other methods that were compared against. The robustness of the proposed method was also demonstrated against various situations including partial occlusion, non-rigid motion, abrupt motion, translation, rotation, and illumination change.

As discussed in the experiments, even with a simple grid initialization strategy, we were able to obtain good results with the proposed method. However, with better initialization, the performance of our method would be enhanced. Therefore, providing sophisticated initializations would be one of the most beneficial directions for future research. Recently, detecting and recognizing objects with part-based models and pictorial structures have drawn much interest [21, 31, 22]. As a result, importance of part-based tracking methods is increasing. Incorporating part-based detection and recognizing methods for the initialization of our method would be a promising way to enhance initialization. Also, our method uses the same measurements for tracking the target object and determining the update. We believe having an independent strategy for determining the update as in [3] would further enhance the performance of the proposed method. Finally, the neighborhood connection strength parameter β needs to be predetermined by the user in

our method. Learning strategies for automatic selection of the parameter β would be an interesting direction for future research.

Acknowledgement

The research was sponsored by the SNU Brain Korea 21 Information Technology program.

References

[1] I. Kim, H. Choi, K. Yi, J. Choi, S. Kong, Intelligent visual surveillance - a survey, International Journal of Control, Automation and Systems 8 (5) (2010) 926–939.

[2] A. Yilmaz, O. Javed, M. Shah, Object tracking: A survey, ACM Computing Surveys 38 (4) (2006) 1–45.

[3] Z. Kalal, J. Matas, K. Mikolajczyk, P-N learning: Bootstrapping binary classifiers by structural constraints, in: Proceedings of Computer Vision and Pattern Recognition, IEEE Conference on, 2010, pp. 49 –56.

[4] H. Grabner, M. Grabner, H. Bischof, Real-time tracking via on-line boosting, in: Proceedings of the British Machine Vision Conference, Vol. 1, 2006, pp. 47–56.

[5] J. Kwon, K. M. Lee, Tracking of a non-rigid object via patch-based dynamic appearance modeling and adaptive basin hopping Monte Carlo sampling, in: Proceedings of Computer Vision and Pattern Recognition, IEEE Conference on, 2009, pp. 1208 –1215.

[6] D. Comaniciu, V. Ramesh, P. Meer, Kernel-based object tracking, Pattern Analysis and Machine Intelligence, IEEE Transactions on 25 (5) (2003) 564 – 577.

[7] M. Isard, A. Blake, CONDENSATION–conditional density propagation for visual tracking, International Journal of Computer Vision 29 (1998) 5–28.

[8] D. A. Ross, J. Lim, R.-S. Lin, M.-H. Yang, Incremental learning for robust visual tracking, International Journal of Computer Vision 77 (2008) 125–141.

[9] P. Pérez, C. Hue, J. Vermaak, M. Gangnet, Color-based probabilistic tracking, in: Proceedings of the 7th European Conference on Computer Vision, 2002, pp. 661–675.

[10] J. Kwon, K. M. Lee, F. Park, Visual tracking via geometric particle filtering on the affine group with optimal importance functions, in: Proceedings of Computer Vision and Pattern Recognition, IEEE Conference on, 2009, pp. 991 –998.

[11] M. J. Black, A. D. Jepson, EigenTracking: Robust matching and tracking of articulated objects using a view-based representation, International Journal of Computer Vision 26 (1998) 63–84.

[12] A. Adam, E. Rivlin, I. Shimshoni, Robust fragments-based tracking using the integral histogram, in: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, Vol. 1, 2006, pp. 798 – 805.

[13] X. Mei, H. Ling, Robust visual tracking using l_1 minimization, in: Computer Vision, IEEE International Conference on, 2009, pp. 1436 –1443.

[14] M. Godec, P. Roth, H. Bischof, Hough-based tracking of non-rigid objects, in: Computer Vision, IEEE International Conference on, 2011, pp. 81 –88.

[15] C. Rother, V. Kolmogorov, A. Blake, GrabCut: Interactive foreground extraction using iterated graph cuts, ACM Transaction on Graphics 23 (2004) 309–314.

[16] B. Babenko, M.-H. Yang, S. Belongie, Robust object tracking with online multiple instance learning, Pattern Analysis and Machine Intelligence, IEEE Transactions on 33 (8) (2011) 1619 –1632.

[17] S. Stalder, H. Grabner, L. van Gool, Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition, in: Computer Vision Workshops, IEEE International Conference on, 2009, pp. 1409 –1416.

[18] M. Salzmann, R. Urtasun, P. Fua, Local deformation models for monocular 3D shape recovery, in: Proceedings of Computer Vision and Pattern Recognition, IEEE Conference on, 2008, pp. 1213–1220.

[19] A. Varol, M. Salzmann, P. Fua, R. Urtasun, A constrained latent variable model, in: Proceedings of Computer Vision and Pattern Recognition, IEEE Conference on, Ieee, New York, 2012, pp. 2248–2255.

[20] M. A. Fischler, R. A. Elschlager, The representation and matching of

pictorial structures, IEEE Transactions on Computers 22 (1) (1973) 67–92.

[21] P. F. Felzenszwalb, D. P. Huttenlocher, Pictorial structures for object recognition, International Journal of Computer Vision 61 (1) (2005) 55–79.

[22] P. Felzenszwalb, R. Girshick, D. McAllester, D. Ramanan, Object detection with discriminatively trained part-based models, Pattern Analysis and Machine Intelligence, IEEE Transactions on 32 (9) (2010) 1627 – 1645.

[23] K. M. Yi, S. W. Kim, H. Jeong, S. Oh, J. Y. Choi, Non-rigid object tracking with elastic structure of local patches and hierarchical sampling, in: Proceedings of Image and Vision Computing New Zealand, 25th International Conference of, 2010, pp. 1 –8.

[24] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[25] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, LIBLINEAR: A library for large linear classification, The Journal of Machine Learning Research 9 (2008) 1871–1874.

[26] J. Platt, et al., Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods, Advances in large margin classifiers 10 (3) (1999) 61–74.

[27] S. Z. Li, Markov Random Field Modeling in Computer Vision, Springer-Verlag New York, Inc., 1995.

[28] S. Avidan, Ensemble tracking, Pattern Analysis and Machine Intelligence, IEEE Transactions on 29 (2) (2007) 261 –271.

[29] S. Yin, J. H. Na, J. Y. Choi, S. Oh, Hierarchical Kalman-particle filter with adaptation to motion changes for object tracking, Computer Vision and Image Understanding 115 (6) (2011) 885 – 900.

[30] S. Agarwal, A. Awan, D. Roth, Learning to detect objects in images via a sparse, part-based representation, Pattern Analysis and Machine Intelligence, IEEE Transactions on 26 (2004) 1475–1490.

[31] K. Mikolajczyk, C. Schmid, A. Zisserman, Human detection based on a probabilistic assembly of robust part detectors, in: Proceedings of the 8th European Conference on Computer Vision, 2004, pp. 69–82.