Tracking-by-Segmentation with Online Gradient Boosting Decision Tree

Jeany Son[1,2] Ilchae Jung[1,2] Kayoung Park[2] Bohyung Han[1,2]
[1]Dept. of Computer Science and Engineering and [2]Computer Vision Lab., POSTECH, Korea
{jeany,chey0313,kayoungpark,bhhan}@postech.ac.kr

Abstract

We propose an online tracking algorithm that adaptively models target appearances based on an online gradient boosting decision tree. Our algorithm is particularly useful for non-rigid and/or articulated objects since it handles various deformations of the target effectively by integrating a classifier operating on individual patches and provides segmentation masks of the target as final results. The posterior of the target state is propagated over time by particle filtering, where the likelihood is computed based mainly on a patch-level confidence map associated with a latent target state corresponding to each sample. Once tracking is completed in each frame, our gradient boosting decision tree is updated to adapt new data in a recursive manner. For effective evaluation of segmentation-based tracking algorithms, we construct a new ground-truth that contains pixel-level annotation of segmentation mask. We evaluate the performance of our tracking algorithm based on the measures for segmentation masks, where our algorithm illustrates outstanding accuracy compared to the state-of-the-art segmentation-based tracking methods.

1. Introduction

Visual tracking problem has been studied extensively and significant progress has been made recently in terms of tracker performance and evaluation methodology. Tracking algorithms typically deal with various challenges by adapting target appearances over time [13, 23, 26], reasoning occlusion [20], handling abrupt motion [21, 24], and/or managing background clutter [6]. However, most of existing algorithms are limited to producing bounding boxes as their outputs, and they often suffer from drifting problems when substantial non-rigid and articulated motions are involved in a target. To overcome such limitations, part- or patch-based tracking algorithms [2, 14, 22] have been proposed, but they still employ bounding boxes for target localization.

We believe that segmentation-based tracking is a natural solution to handle non-rigid and deformable objects effectively. Hence, we propose a novel tracking-by-segmentation framework, which maintains target appearances based on small local patches and performs tracking by classifying the patches into the target or the background. A new online gradient boosting decision tree is integrated for classification, where the loss function is minimized incrementally without offline samples. We employ the distance to the object center as a feature in addition to ordinary visual features for classification, which turns out to improve performance substantially. Particle filter is adopted to propagate the posterior, where a segmentation mask is constructed for each latent target state corresponding to a sample. The final tracking result is given by the mask of the best sample. To evaluate our tracking algorithm, we construct new ground-truths of existing datasets, which contain pixel-level label annotations and facilitate accurate tracking performance evaluation of segmentation-based tracking algorithms.

There are several prior studies about segmentation-based tracking. Aeschliman *et al.* [1] propose pixel-level probabilistic models for joint target tracking and segmentation, but this algorithm assumes a static camera environment similar to a background subtraction. Another segmentation-based tracking algorithm is proposed in a particle filter framework by applying *GrabCut* [27] to each sample for robust observation [4], but the likelihood computation is based on simple features such as color and gradient distributions. A mid-level appearance structure, superpixel, is employed to distinguish the target from background, where target state is obtained by computing Maximum a Posteriori (MAP) based on a confidence map of superpixels [28]. HoughTrack [10] relies on an online Hough forest, where patch-based classification and voting is performed to identify the target area. PixelTrack [7] provides soft segmentations of the target based on pixel-wise classification. Recently, Hong *et al.* [17] propose a multiple quantization technique, which tracks the target by identifying the best configuration across pixel, superpixel and bounding box levels. These methods may have trouble to handle scale variations effectively since the size of voting vector does not adapt a target size [10, 7] and the algorithm is designed for fixed-size bounding boxes [17]. In addition, although all the above tracking algorithms can generate segmentation masks, the evaluations still rely on bounding boxes.

Online boosting algorithms have often been applied to visual tracking due to its simplicity and robustness. A simple online boosting-based tracking algorithm is proposed in [11], and tracking by a semi-supervised online boosting is presented in [12]. Gradient boosting, which is a more sophisticated algorithm to update boosting classifiers online, often improves classification accuracy [9]. To handle labeling ambiguity for semi-supervised learning, [3] describes an online gradient boosting technique based on multiple instance learning. Weak classifiers in boosting are typically simple linear functions but decision trees are sometimes used for better generalization [9]; the resulting classifier is referred to as gradient boosting decision tree [5]. According to our survey, the online version of the gradient boosting decision tree is not explored yet at least for visual tracking.

Our algorithm has several interesting features compared to the existing methods as summarized below:

- We propose a novel learning algorithm for an online gradient decision boosting tree. This machinery classifies each patch into target or background.
- Our classifier employs a distance to the object center as a feature. By exploiting this feature for patch classification and updating the classifier online, we can handle deformations and scale changes of target naturally.
- We construct ground-truth segmentation masks of every frame in testing videos, and evaluate segmentation-based tracking algorithms more rigorously.

The rest of this paper is organized as follows. We first review gradient boosting and gradient boosting decision tree in Section 2. The details about online gradient boosting decision tree are presented in Section 3, and the procedure of our tracking algorithm is discussed in Section 4. Section 5 describes our performance evaluation method and Section 6 illustrates experimental results.

2. Background

This section reviews a gradient boosting based on regression trees, which is referred to as gradient boosting decision tree [5, 9]. Specifically, we first describe the main idea of gradient boosting briefly and discuss how the weak learners are designed in gradient boosting decision tree.

2.1. Gradient Boosting (GB)

Gradient boosting [5, 9] is a gradient-based approach to learn a boosting classifier incrementally, and approximates a function $f : \mathbb{R}^n \to \mathbb{R}$ based on a linear combination of weak learners $h : \mathbb{R}^n \to \mathbb{R}$ as

$$f(\mathbf{x}) = \sum_{j=1}^{M} \alpha_j h_j(\mathbf{x}; \theta_j), \quad (1)$$

Algorithm 1 Gradient boosting [9]

```
1: Initialize $f_0(\mathbf{x}) = 0$
2: for $j = 1 \ldots M$ do
3:     $g(\mathbf{x}_i) = \left[\frac{\partial \ell(y_i, f(\mathbf{x}_i))}{\partial f(\mathbf{x}_i)}\right]_{f(\mathbf{x})=f_{j-1}(\mathbf{x})}, i = 1 \ldots N$
4:     $\theta_j = \arg\min_{\theta,\beta} \sum_{i=1}^{N} [-g(\mathbf{x}_i) - \beta \cdot h(\mathbf{x}_i; \theta)]^2$
5:     $\alpha_j = \arg\min_{\alpha} \sum_{i=1}^{N} \ell(y_i, f_{j-1}(\mathbf{x}_i) + \alpha \cdot h(\mathbf{x}_i; \theta_j))$
6:     $f_j(\mathbf{x}) = f_{j-1}(\mathbf{x}) + \alpha_j \cdot h(\mathbf{x}; \theta_j)$
7: end for
```

where $\mathbf{x} \in \mathbb{R}^n$ is an input vector, $\alpha_j \in \mathbb{R}$ is a real-valued weight, and M is the number of weak learners.

Given training examples, $\{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathbb{R}^n \text{ and } y_i \in \mathbb{R}\}_{i=1:N}$, $f(\cdot)$ is constructed in a greedy manner by selecting parameter θ_j and weight α_j of a weak learner iteratively to minimize an augmented loss function given by

$$\mathcal{L} = \sum_{i=1}^{N} \ell(y_i, f(\mathbf{x}_i)) \equiv \sum_{i=1}^{N} \exp(-y_i f(\mathbf{x}_i)), \quad (2)$$

where an exponential loss function is adopted[1]. The greedy optimization procedure is summarized in Algorithm 1.

2.2. Gradient Boosting Decision Tree (GBDT)

GBDT is a gradient boosting algorithm that utilizes decision stumps or regression tress as weak classifiers. In GBDT, the weak learners measure the error observed in each node, split the node using a test function $\kappa : \mathbb{R}^n \to \mathbb{R}$ with a threshold τ, and return values η^l and η^r. The optimal split is obtained by identifying triplet (τ, η^l, η^r) to minimize the error after split, which is given by

$$\epsilon(\tau) = \sum_{i:\kappa(\mathbf{x}_i)<\tau} w_i^j (r_i^j - \eta^l)^2 + \sum_{i:\kappa(\mathbf{x}_i)\geq\tau} w_i^j (r_i^j - \eta^r)^2, \quad (3)$$

where w_i^j and r_i^j denote the weight and response of $\mathbf{x}_i$ in the j-th iteration, respectively. Formally, they are given by

$$w_i^j = \exp(-y_i f_{j-1}(\mathbf{x}_i)) \text{ and} \quad (4)$$

$$r_i^j = g(\mathbf{x}_i)/w_i^j = -y_i \exp(-y_i f_{j-1}(\mathbf{x}_i))/w_i^j = -y_i. \quad (5)$$

We identify the optimal triplet $(\tau^*, \eta^{l*}, \eta^{r*})$ by minimizing the error in Eq. (3) over all possible τ's at each node, where, given τ, (η^{l*}, η^{r*}) can be found simply by computing the weighted average of r_i^j's over training examples that fall on the corresponding side of the split. η^{l*} and η^{r*} are given to the left and right children of the current node, respectively, and η's stored in the leaf node is used as a score of the weak learner corresponding to the tree depending on its input $\mathbf{x}$.

The training procedure of gradient boosting decision tree is presented in Algorithm 2, where ν is a shrinkage factor to

[1]A log loss function, $\ell = \log(1 + \exp(-y_i f(\mathbf{x}_i)))$, is also available.

Algorithm 2 Gradient boosting decision tree

```
1: Initialize $f_0(\mathbf{x}) = 0,\ \eta_0,\ d_0 = 0$
2: for $j = 1 \dots M$ do
3:     $w_i = \exp(-y_i f_{j-1}(\mathbf{x}_i)), i = 1 \dots N$
4:     $r_i = -y_i, i = 1 \dots N$
5:     $\mathcal{S} = \{(\mathbf{x}_i, w_i, r_i)\}_{i=1:N}$ and $\mathcal{U} = \{\mathbf{x} | \mathbf{x} \in \mathbb{R}^n\}$
6:     $\mathbf{R} = $ GrowTree$(\mathcal{S}, \mathcal{U}, \eta_0, d_0)$
7:     $f_j(\mathbf{x}) = f_{j-1}(\mathbf{x}) - \nu \cdot \sum_{k=1}^{|\mathbf{R}|} \eta_k \delta(\mathbf{x} \in \mathcal{R}_k)$,
       where $(\mathcal{R}_k, \eta_k) \in \mathbf{R}$
8: end for
9: procedure GrowTree$(\mathcal{S}, \mathcal{R}, \eta, d)$
10:    if $d < d_{\max} \wedge |\mathcal{S}| > N_{\min}$ then
11:        $(\tau, \eta^l, \eta^r) = $ SplitLearning$(\mathcal{S})$
12:        $\mathcal{R}^l = \{\mathbf{x} | \kappa_p(\mathbf{x}) < \tau\}$ and $\mathcal{R}^r = \{\mathbf{x} | \kappa_p(\mathbf{x}) \geq \tau\}$
13:        $\mathbf{R}^l = $ GrowTree$(\mathcal{S}_{i:\kappa_p(\mathbf{x}_i)<\tau}, \mathcal{R}^l, \eta^l, d+1)$
14:        $\mathbf{R}^r = $ GrowTree$(\mathcal{S}_{i:\kappa_p(\mathbf{x}_i)\geq\tau}, \mathcal{R}^r, \eta^r, d+1)$
15:        return $\mathbf{R}^l \cup \mathbf{R}^r$
16:    else
17:        return $\{(\mathcal{R}, \eta)\}$
18:    end if
19: procedure SplitLearning$(\mathcal{S})$
20:    $(\tau^*, \eta^{l*}, \eta^{r*}) = \arg\min_{(\tau, \eta^l, \eta^r)} \epsilon(\tau)$ in Eq. (3)
21:    return $(\tau^*, \eta^{l*}, \eta^{r*})$
```

avoid overfitting, d is a depth of the tree, p is a node index, and $\delta(\cdot)$ denotes an indicator function. Refer to [5] for more details.

3. Online Gradient Boosting Decision Tree

This section describes a generic framework to learn an online gradient boosting decision tree. If additional data are given sequentially, we need to update the weak classifiers in an online manner, where the data used for training until the previous stages are no more available.

Suppose that we have a classifier at the current time step t and its error at a node p in a weak classifier is defined as

$$\epsilon_t(\tau_{t,p}) = \sum_{i=1}^{N_p} w_{t,i}(r_i - \eta_{t,p})^2, \qquad (6)$$

where N_p is the number of examples falling on node p, and

$$\eta_{t,p} = \begin{cases} \eta^l_{t,p}, & \text{if } \kappa_p(\mathbf{x}_i) < \tau_{t,p} \\ \eta^r_{t,p}, & \text{otherwise} \end{cases} .$$

Note that $\eta_{t,p}$ represents either $\eta^l_{t,p}$ or $\eta^r_{t,p}$ depending on $\kappa(\mathbf{x}_i)$ in the following derivations for simplicity and that $\epsilon_t(\tau_{t,p})$ in Eq. (6) has two terms as in Eq. (3).

The goal of our online gradient boosting decision tree is to learn new parameters $\tau_{t+1,p}$ and $\eta_{t+1,p}$, given a new training example to the classifier, through optimization of the following error function:

$$\begin{aligned} \epsilon_{t+1}(\tau_{t+1,p}) &= \sum_{i=1}^{N_p+1} w_{t+1,i}(r_i - \eta_{t+1,p})^2 \\ &= \sum_{i=1}^{N_p+1} w_{t+1,i}(r_i - (\eta_{t,p} + \Delta\eta))^2. \end{aligned} \qquad (7)$$

Assuming that the optimal $\tau_{t+1,p}$ is known at the moment, we optimize $\eta_{t+1,p}$ first. A critical issue in this procedure is that $\{(w_{t+1,i}, r_i)\}_{i=1\dots N_p}$ is unavailable in online learning. Therefore, it is impossible to minimize the error exactly by adjusting $\tau_{t+1,p}$ and computing the weighted average of r_i's as we can do in offline learning. Instead, we should update the weak classifier based on the new example as well as the limited information of the current classifier.

If we represent $\epsilon_{t+1}(\tau_{t+1,p})$ in Eq. (7) with a function of $\Delta\eta$ as

$$\begin{aligned} \epsilon_{t+1}(\tau_{t+1,p}) = \sum_{i=1}^{N_p+1} \{ & w_{t+1,i}(r_i - \eta_{t,p})^2 \\ & + w_{t+1,i}\left(-2(r_i - \eta_{t,p})\Delta\eta + (\Delta\eta)^2\right)\}, \end{aligned} \qquad (8)$$

we can apparently minimize this quadratic function with respect to $\Delta\eta$ and obtain the following solution:

$$\Delta\eta(\tau_{t+1,p}) = \begin{cases} \Delta\eta^l(\tau_{t+1,p}), & \text{if } \kappa_p(\mathbf{x}_i) < \tau_{t+1,p} \\ \Delta\eta^r(\tau_{t+1,p}), & \text{otherwise} \end{cases}, \qquad (9)$$

where

$$\Delta\eta^l(\tau_{t+1,p}) = \frac{\sum_{i:\kappa_p(\mathbf{x}_i)<\tau_{t+1,p}}^{N_p+1} w_{t+1,i} r_i}{\sum_{i:\kappa_p(\mathbf{x}_i)<\tau_{t+1,p}}^{N_p+1} w_{t+1,i}} - \eta^l_{t,p} \qquad (10)$$

and

$$\Delta\eta^r(\tau_{t+1,p}) = \frac{\sum_{i:\kappa_p(\mathbf{x}_i)\geq\tau_{t+1,p}}^{N_p+1} w_{t+1,i} r_i}{\sum_{i:\kappa_p(\mathbf{x}_i)\geq\tau_{t+1,p}}^{N_p+1} w_{t+1,i}} - \eta^r_{t,p}. \qquad (11)$$

However, since $\{(w_{t+1,i}, r_i)\}_{i=1\dots N_p}$ are not available as mentioned earlier, we employ a recursive method to find η. Note that $\eta^l_{t+1,p}$ and $\eta^r_{t+1,p}$ are given respectively by

$$\begin{aligned} \eta^l_{t+1,p}(\tau_{t+1,p}) &= \frac{\sum_{i|\kappa_p(\mathbf{x}_i)<\tau_{t+1,p}}^{N_p+1} w_{t+1,i} r_i}{\sum_{i|\kappa_p(\mathbf{x}_i)<\tau_{t+1,p}}^{N_p+1} w_{t+1,i}} \\ &\approx (1-\alpha)\eta^l_{t,p} + \alpha w_{t+1,N_p+1} r_{N_p+1} \end{aligned} \qquad (12)$$

and

$$\begin{aligned} \eta^r_{t+1,p}(\tau_{t+1,p}) &= \frac{\sum_{i|\kappa_p(\mathbf{x}_i)\geq\tau_{t+1,p}}^{N_p+1} w_{t+1,i} r_i}{\sum_{i|\kappa_p(\mathbf{x}_i)\geq\tau_{t+1,p}}^{N_p+1} w_{t+1,i}} \\ &\approx (1-\alpha)\eta^r_{t,p} + \alpha w_{t+1,N_p+1} r_{N_p+1} \end{aligned} \qquad (13)$$

Algorithm 3 Online gradient boosting decision tree

```
1:  Input : Current classifier $f_0 = f_t, \{\mathbf{x}_i, y_i\}_{i=1:N^o}$
2:  for $j = 1 \dots M$ do
3:      $w_i = \exp(-y_i f_{j-1}(\mathbf{x}_i)), i = 1 \dots N^o$
4:      $r_i = -y_i, i = 1 \dots N^o$
5:      Identify examples falling on each node in the tree.
6:      for each node $p$ from root do
7:          Update $\eta^l_{t+1,p}(\tau_{t+1,p})$ for each $\tau_{t+1,p}$ : Eq. (12)
8:          Update $\eta^r_{t+1,p}(\tau_{t+1,p})$ for each $\tau_{t+1,p}$ : Eq. (13)
9:          Compute $\epsilon_{t+1}(\tau_{t+1,p})$ for each $\tau_{t+1,p}$ : Eq. (8)
10:         $\tau^*_{t+1,p}$ = arg $\min_{\tau_{t+1,p}} \epsilon_{n+1}(\tau_{t+1,p})$
11:         Find $(\mathcal{R}_{t+1,p}, \eta^{l*}_{t+1,p}, \eta^{r*}_{t+1,p})$ using $\tau^*_{t+1,p}$
12:     end for
13: end for
```

where α denotes learning rate. Therefore, we obtain the following two equations to update weak classifier

$$\Delta\eta^l(\tau_{t+1,p}) = \alpha(w_{t+1,N_p+1} r_{N_p+1} - \eta^l_{t,p}) \tag{14}$$

$$\Delta\eta^r(\tau_{t+1,p}) = \alpha(w_{t+1,N_p+1} r_{N_p+1} - \eta^r_{t,p}). \tag{15}$$

Intuitively, the return value η is updated with proportion to the amount of the difference between the weighted response of the new example and the previous return value.

We have described how to update weak learner given a new example, but it is trivial to extend the algorithm when multiple new examples are available. Specifically, $\Delta\eta$'s are

$$\Delta\eta^l(\tau_{t+1,p}) = \alpha\left(\sum_{i=1}^{N^o_p} w_{t+1,i} r_i - \eta^l_{t,p}\right) \tag{16}$$

$$\Delta\eta^r(\tau_{t+1,p}) = \alpha\left(\sum_{i=1}^{N^o_p} w_{t+1,i} r_i - \eta^r_{t,p}\right), \tag{17}$$

where N^o_p denotes the number of online examples falling on node p.

We have discussed how to find the optimal $\eta_{t+1,p}$ given $\tau_{t+1,p}$. Note that we need to search over $\tau_{t+1,p}$ to find the optimal triplet $(\tau^*_{t+1,p}, \eta^{l*}_{t+1,p}, \eta^{r*}_{t+1,p})$ by minimizing $\epsilon_{t+1}(\tau_{t+1,p})$ in Eq. (8). Since it is implausible to search over all possible τ's, we sample τ's from a uniform distribution and interpolate missing values. Once the optimal parameters are obtained, we estimate the error in the current stage $\epsilon_{t+1}(\tau_{t+1,p})$ including the information of new example $(w_{t+1,N_p+1}, r_{N_p+1})$, and propagate it to the next stage.

Algorithm 3 presents the online update procedure of gradient boosting decision tree when N^o new examples with either positive or negative labels are given for each update.

4. Online Tracking for Non-Rigid Objects

Our tracking algorithm propagates the posterior through particle filtering to estimate the optimal target state, and obtain target segmentation by classifying patches in the region of interest. We employ an online gradient boosting decision tree for the classification. Since the proposed algorithm provides the segmentation mask as well as the bounding box of the target, it is useful to track non-rigid and articulated objects. The details of our algorithm is described next.

4.1. Latent Target State Estimation

We adopt particle filtering framework [18] to estimate the posterior of target state, which is propagated through prediction and measurement steps as

$$\begin{aligned} p(\mathbf{s}_t|\mathbf{z}_{1:t}) &\propto p(\mathbf{z}_t|\mathbf{s}_t)p(\mathbf{s}_t|\mathbf{z}_{1:t-1}) \\ &= p(\mathbf{z}_t|\mathbf{s}_t)\int p(\mathbf{s}_t|\mathbf{s}_{t-1})p(\mathbf{s}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{s}_{t-1}, \end{aligned} \tag{18}$$

where $\mathbf{s}_t$ and $\mathbf{z}_t$ are state and observation variable at frame t, respectively. The posterior is approximated by a set of particles and their weights $\{(\mathbf{s}^{(k)}_t, \omega^{(k)}_t)\}_{k=1\dots K}$. Each particle represents the center location of target in our implementation. Although the state space is 2-dimensional, our patch classification algorithm and its online update can handle moderate scale changes of target over time. The optimal target state is given by the particle with maximum weight.

4.2. Generating Patch Confidence Map

Given a classifier by online gradient boosting decision tree learned based on the algorithm discussed in Section 3, our tracking algorithm classifies each patch in a frame into either the target or background.

We employ two different kinds of features to represent individual patches; one is an ordinary visual feature $\phi(\mathbf{p}) \in \mathbb{R}^m$ and the other is the distance to target center $\upsilon(\mathbf{p};\mathbf{s}) \in \mathbb{R}$, where $\mathbf{p}$ and $\mathbf{s}$ denote the center of patch and target, respectively. The score of each patch is given by applying the learned function to the joint feature vector $\Phi(\mathbf{p};\mathbf{s}) \equiv \left[\phi(\mathbf{p})^\top, \upsilon(\mathbf{p};\mathbf{s})\right]^\top$.

Note that, we obtain a confidence map for each latent target center $\mathbf{s}$, which corresponds to each sample from particle filter. Since the feature vector $\Phi(\mathbf{p};\mathbf{s})$ contains the field corresponding to the distance to target center denoted by $\upsilon(\mathbf{p};\mathbf{s})$, the confidence map typically has higher values around the optimal target location $\mathbf{s}^*$.

4.3. Likelihood

The weight of a particle, $\omega^{(k)}_t \propto p(\mathbf{z}_t|\mathbf{s}^{(k)}_t)$, is measured by two factors—classification scores of patches within the region of interest and similarity of holistic appearances between target model and candidate, which is given by

$$p(\mathbf{z}_t|\mathbf{s}^{(k)}_t) = p_{\text{cls}}(\mathbf{z}_t|\mathbf{s}^{(k)}_t)p_{\text{app}}(\mathbf{z}_t|\mathbf{s}^{(k)}_t), \tag{19}$$

where $p_{\text{cls}}(\mathbf{z}_t|\mathbf{s}^{(k)}_t)$ and $p_{\text{app}}(\mathbf{z}_t|\mathbf{s}^{(k)}_t)$ are likelihoods for classification scores and holistic appearance similarity, respectively.

Patch confidence map The likelihood by patch-level classifier scores is based on the summation of the confidence map scores of positive patches, which is given by

$$p_{\text{cls}}(\mathbf{z}_t|\mathbf{s}_t^{(k)}) = \exp\left(\lambda \sum_{i=1}^{N_t} \max\{f(\Phi(\mathbf{p}_t^i; \mathbf{s}_t^{(k)}), 0\}\right), \quad (20)$$

where N_t is the number of patches at frame t, k is the sample index, and λ is a constant.

Holistic appearances The holistic appearance models are based on Histogram of Oriented Gradients (HOG) and color histogram. The HOG feature is extracted from the bounding box of each sample, and color histogram is constructed only with positive patches. The likelihood of holistic appearances, $p_{\text{appr}}(\mathbf{z}_t|\mathbf{s}_t^{(k)})$, is computed by using the Bhattacharyya distances between target and candidate histograms.

4.4. Generating Segmentation Mask

Once the likelihood of all samples are computed, we identify the target state by maximum a posterior solution. We have the confidence map for the best sample, and the preliminary target segmentation result is obtained by simple thresholding in the map. After that, we employ simple morphological operations and connected component analysis to improve segmentation results. We apply closing operation first to connect regions with small gaps, and then opening operation to remove trivial noises. After that, we find a largest segment by connected component analysis. It is possible to use more sophisticated binary segmentation algorithms such as *GrabCut* [27], but we could not find any noticeable improvement when *GrabCut* is used.

Our tracking algorithm propagates the posterior of target only in 2-dimensional state space through particle filtering. However, we can handle moderate scale changes of target since our patch-level classification is robust to small variations in the distance to target center and the partition function for distance feature is updated in each frame.

4.5. Online Sample Selection

Our online gradient boosting decision tree is updated based on the segmentation output obtained from Section 4.4. Since online learning techniques often suffer from drifting issue due to labeling error in new data, we update the classifier using reliable patches. In our algorithm, we select top 20% of patches with high absolute scores in both positive and negative sides to retrain the classifier.

The overall procedure of our tracking algorithm is summarized in Algorithm 4.

Algorithm 4 Tracking by online GBDT

```
1: Initialize gradient boosting classifier $f_1$
2: for each frame $t = 2 \ldots T$ do
3:     for each sample $k = 1 \ldots K$ do
4:         Compute $p(\mathbf{z}_t|\mathbf{s}_t^{(k)})$ using Eq. (19)
5:     end for
6:     $\mathbf{s}_t^* = \arg\max_{\mathbf{s}_t} p(\mathbf{s}_t|\mathbf{z}_t)$
7:     Predict patch labels: $y_i = \text{sgn}(f_{t-1}(\Phi(\mathbf{p}_t^i; \mathbf{s}_t^*))), \forall i$
8:     Update classifier by Algorithm 3:
           $f_t \leftarrow \text{OnlineGBDT}(f_{t-1}, \{\Phi(\mathbf{p}_t^i; \mathbf{s}_t^*), y_i\}_{i=1}^{N_t})$
9: end for
```

5. Evaluation Methodology

This section presents the motivation of new ground-truth construction and describes the details of our performance evaluation methods based on the ground-truth.

5.1. Motivation for New Ground-Truths

Performance evaluation of tracking algorithms based on bounding boxes is straightforward and there are already a few well-known standard evaluation suites such as online tracking benchmark [29] and VOT challenge benchmark [19]. However, the evaluation of segmentation-based tracking algorithms is more complicated because ground-truth annotation is extremely time consuming and requires huge human efforts. Hence, most of existing works still rely on the evaluation protocol of bounding box trackers; results may not be sufficiently reliable since one needs to compare tracking results in segmentation masks and ground-truths in bounding boxes to compute quantitative scores.

We combined two datasets released in [10, 22] for evaluation. They contain 11 videos altogether, each of which involves significant non-rigid and articulated motions of target, and are often used to evaluate segmentation-based tracking algorithms. However, the ground-truth annotations of these datasets are based only on bounding box, and more importantly the quality of annotations are very poor. Figure 1 illustrates several examples of ground-truth errors; red bounding boxes are the ground-truths provided in the dataset. The ground-truths are not consistent at all and sometimes misaligned with target. It is obvious that evaluation based on such poor ground-truths is not reliable.

5.2. Ground-truth Annotations

To handle the problems discussed above, we constructed new ground-truth of pixel-level segmentation in each frame of all videos in the datasets. Our annotation results are given in Figure 1 as well, where the segments for target objects are highlighted and target bounding boxes are illustrated with green bounding boxes. The pixel-level annotations were performed manually, and we generated bounding box an-

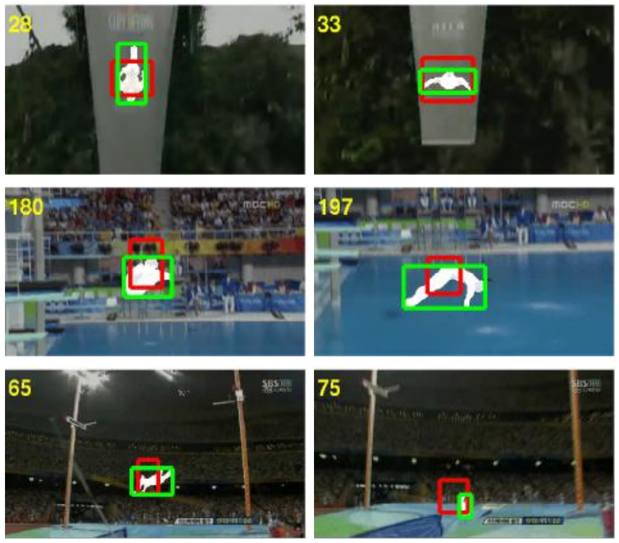


Figure 1. Examples of inaccurate ground-truth annotations and our new annotations for *Cliff-dive2*, *Diving*, and *High-jump* sequences. (Red) existing ground-truth (Green) our new ground-truth (Highlight) segmentation mask.

notations by computing the tightest rectangular bounding boxes containing all target pixels; although this may not be the best way to obtain the bounding box annotation, we believe that this is at least consistent.

5.3. Evaluation Protocol

We compared our tracking algorithms with three recent segmentation based methods, which are focused on non-rigid and deformable objects and can generate segmentation masks, and five state-of-the-art regular tracking methods. The segmentation based algorithms include HoughTrack (HT) [10], Superpixel Tracker (SPT) [28], PixelTrack (PT) [7]. We selected Struck [15], SCM [31], MEEM [30], MQT [17] and MUSTer [16] as the regular tracking algorithms. To investigate the benefit of online GBDT classifier and integrated distance feature, we tested the performance of the tracking algorithm based on online boosting [25] with and without distance feature, which are denoted by OABd and OAB, respectively, and online GBDT without distance feature (Ours-d). The number of weak classifiers in OABd and OAB is determined empirically and set to 400.

To measure the accuracy of the trackers, the PASCAL VOC [8] overlap ratio is employed for the evaluation of success ratios based on both bounding boxes and segmentation masks. The center location error between ground-truth and tracking results is analyzed as well. Table 1 summarizes the methods to construct the final segmentation masks and bounding boxes. We adopt the options used in the original algorithms by default, but generate segmentation masks for SPT and PT through simple thresholding followed by morphological operations like our algorithm. Note that prior

Table 1. Methods for segmentation masks and bounding boxes

	Segmentation mask	Bounding box
HT [10]	*GrabCut*	Fit to segmentation
SPT [28]	*Thresholding from CM	MAP from particle filter
PT [7]	*Thresholding from CM	Center of mass in CM+voting (fixed scale)
Ours	Thresholding from CM	Fit to segmentation

CM: confidence map, *: integrated in this paper for evaluation purpose

works [10, 7, 17] employ the success ratio at 10% of bounding box overlap for performance evaluation. However, 10% overlap ratio is too low and it is difficult to interpret the evaluation result; we believe that our evaluation method is more intuitive and comprehensive.

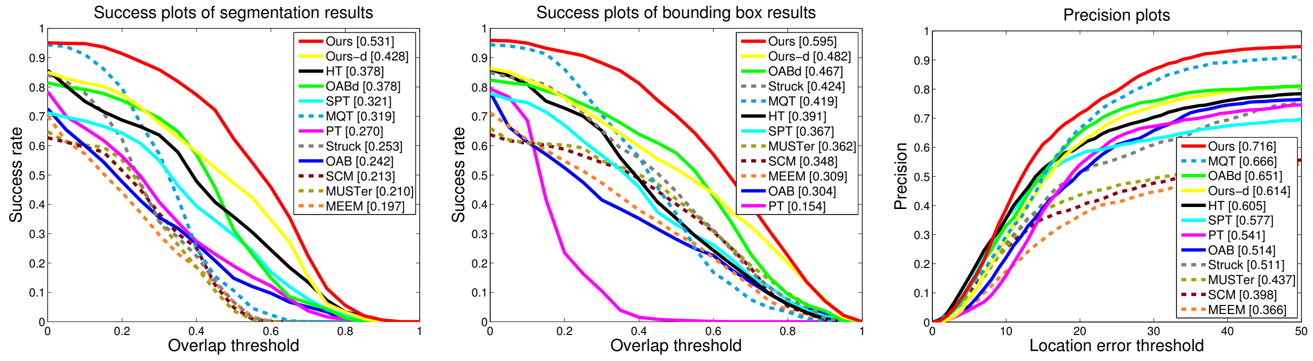
6. Experiments

6.1. Implementation Details

We implemented an online extension of gradient boosting decision tree based on the code by Becker *et al.* [5]. We used all patches centered at every pixels within the region of interest. The patch size for classification is 8×8, from which features are extracted. As visual features, we employed the means and the standard deviations of 6 color channels in RGB and HSV (12D) and Gabor filter bank responses (2 scales $\times$ 2 frequencies $\times$ 8 orientations = 32D). Euclidean distance (1D) to the object center is used as distance feature, and the number of samples for particle filter is 100. The shrinkage factor ν of boosting in Algorithm 2 is set to 0.1. We used 20 weak-classifiers, 1000 bins for sampling τ, and maximum depth of tree $d_{max} = 5$. Distance measure is used only in the last level; classification by distance measure is based on whether the patch falls between certain lower and upper bounds of distance. All algorithms are given bounding box initializations. Our algorithm constructs the initial segmentation masks at the first frame from the bounding box annotation using *GrabCut* [27] automatically. Our tracking algorithm runs at 2-3 frame/sec in the standard PC.

6.2. Results and Discussion

Figure 2 illustrates the performance of all compared algorithms in terms of success ratios of segmentation and bounding box overlaps and precision of center location errors, which are based on the standard criteria used in online tracking benchmark [29]. The trackers providing target segmentation masks are denoted by solid lines, and dashed lines correspond to the methods returning bounding boxes only. For Figure 2(a), we evaluate the tracking algorithms with no segmentation mask based on overlap ratios between ground-truth segmentation mask and bounding box outputs. Table 2 presents the average overlap ratios for bounding box and segmentation mask, where the best results are marked in red and the second best in blue.



(a) Success ratios of segmentation (b) Success ratios of bounding box (c) Precision with center location errors

Figure 2. Success and precision plots in the non-rigid object tracking dataset

Table 2. Quantitative results based on bounding box and segmentation in the non-rigid object tracking dataset

	(a) Bounding box overlap ratio												(b) Segmentation overlap ratio						
	HT	SPT	PT	Struck	SCM	MEEM	MQT	MUSTer	OAB	OABd	Ours-d	Ours	HT	SPT	PT	OAB	OABd	Ours-d	Ours
Cliff-dive1	61.0	66.5	29.9	62.4	61.8	33.3	65.1	60.9	66.7	68.2	74.3	75.9	64.2	54.6	60.1	57.9	58.7	65.8	67.6
Cliff-dive2	52.0	30.3	13.4	34.0	30.0	27.6	35.0	14.8	32.2	24.6	36.6	49.3	49.4	41.8	16.0	16.8	16.3	25.8	36.7
Motocross1	55.1	11.2	3.6	29.2	10.8	9.1	24.0	22.1	5.4	11.2	26.8	62.4	52.2	8.9	1.4	4.4	8.3	24.1	53.1
Motocross2	62.0	46.1	21.3	70.6	41.7	60.4	66.3	68.1	56.6	55.0	69.8	72.1	53.0	37.1	39.7	46.3	37.1	59.4	64.5
Skiing	50.2	27.7	21.4	3.2	7.4	29.5	31.3	3.5	21.3	20.0	21.1	39.0	41.0	37.3	43.0	19.5	17.7	26.7	32.1
Mtn-bike	48.6	53.0	12.9	66.2	71.4	64.8	57.4	69.5	13.5	62.4	38.7	61.2	53.4	43.0	32.1	10.2	46.8	37.9	54.9
Volleyball	27.7	26.8	15.2	36.2	13.2	31.1	54.5	19.0	0.3	33.8	16.3	46.2	31.1	6.5	25.1	0.0	24.7	10.2	41.1
Diving	7.9	35.2	12.3	33.6	15.1	12.6	30.9	20.7	36.3	52.7	50.5	50.6	6.7	21.2	25.5	30.4	44.0	43.3	44.1
Gymnastics	10.4	42.6	26.0	53.7	13.9	17.6	35.7	47.7	56.1	58.3	77.0	70.4	9.2	10.6	52.0	39.3	44.8	67.3	69.8
Transformer	63.4	55.8	13.9	57.7	55.6	51.4	54.2	59.6	72.1	85.4	84.6	86.6	45.0	2.8	5.5	59.1	72.0	74.1	74.0
High_jump	39.1	5.3	0.6	15.4	8.4	21.1	33.0	8.3	4.7	23.7	42.8	51.5	40.4	52.1	0.9	4.9	19.3	18.5	42.8
average	43.4	36.4	15.5	42.0	29.9	32.6	44.3	35.8	33.2	45.0	49.0	60.5	40.5	28.7	27.4	26.3	35.4	41.2	52.8

The results in Figure 2 and Table 2 clearly show that our algorithm outperforms all other methods in terms of all metrics. In particular, the proposed algorithm is generally good for segmentation and bounding box alignment while the improvement of center location errors is relatively small. According to our experiments, both online GBDT classifier and distance feature made substantial contribution to improve performance, which is noticeable in Figure 2 and Table 2. Although MEEM [30] and MUSTer [16] are outstanding in the online tracking benchmark [29], they are not successful to track deformable and articulated objects. In our experiments, PT uses small bounding boxes inside the standard ground-truths for initialization, tracking and evaluation, thus their success ratio on bounding box overlaps is low while it has relatively high precision. All the results reported here are based on the new annotations for bounding box and segmentation mask described in Section 5.2.

We obtained reasonably successful results (success rate = 0.524, precision = 0.748) in the online tracking benchmark [29]. Although our algorithm is not the best compared to the latest ones optimized in the benchmark, it is still competitive. Note that we aim to generate pixel-level segmentation, which is more difficult than bounding box tracking.

Figure 3 illustrates the qualitative results on a representative frame of each sequence, where segmentation mask and bounding box results of each algorithm are illustrated. The first column presents original frames, the second one is ground-truth segmentations, and the remaining ones correspond to the results of HT, SPT, PT and our algorithm. The segmentation overlap ratio is shown at the bottom of each result. Again, it clearly shows that our tracker handles diverse challenges, including deformations, in-plane rotations, and articulations, effectively and presents good results compared to other techniques.

The detailed results including new ground-truth annotations are provided in our project website[2].

7. Conclusions

We proposed a novel online learning algorithm for gradient boosting decision tree to track and segment non-rigid and deformable objects. For more rigorous performance evaluation, we constructed a new ground-truth for both segmentation mask and bounding box. The experimental results demonstrate that our algorithm outperforms the state-of-the-art tracking algorithms substantially in terms of success ratio, precision, and average overlap ratio.

Acknowledgement

This work was supported by the ICT R&D program of MSIP/IITP [B0101-15-0307, B0101-15-0552].

[2]http://cvlab.postech.ac.kr/research/ogbdt_track

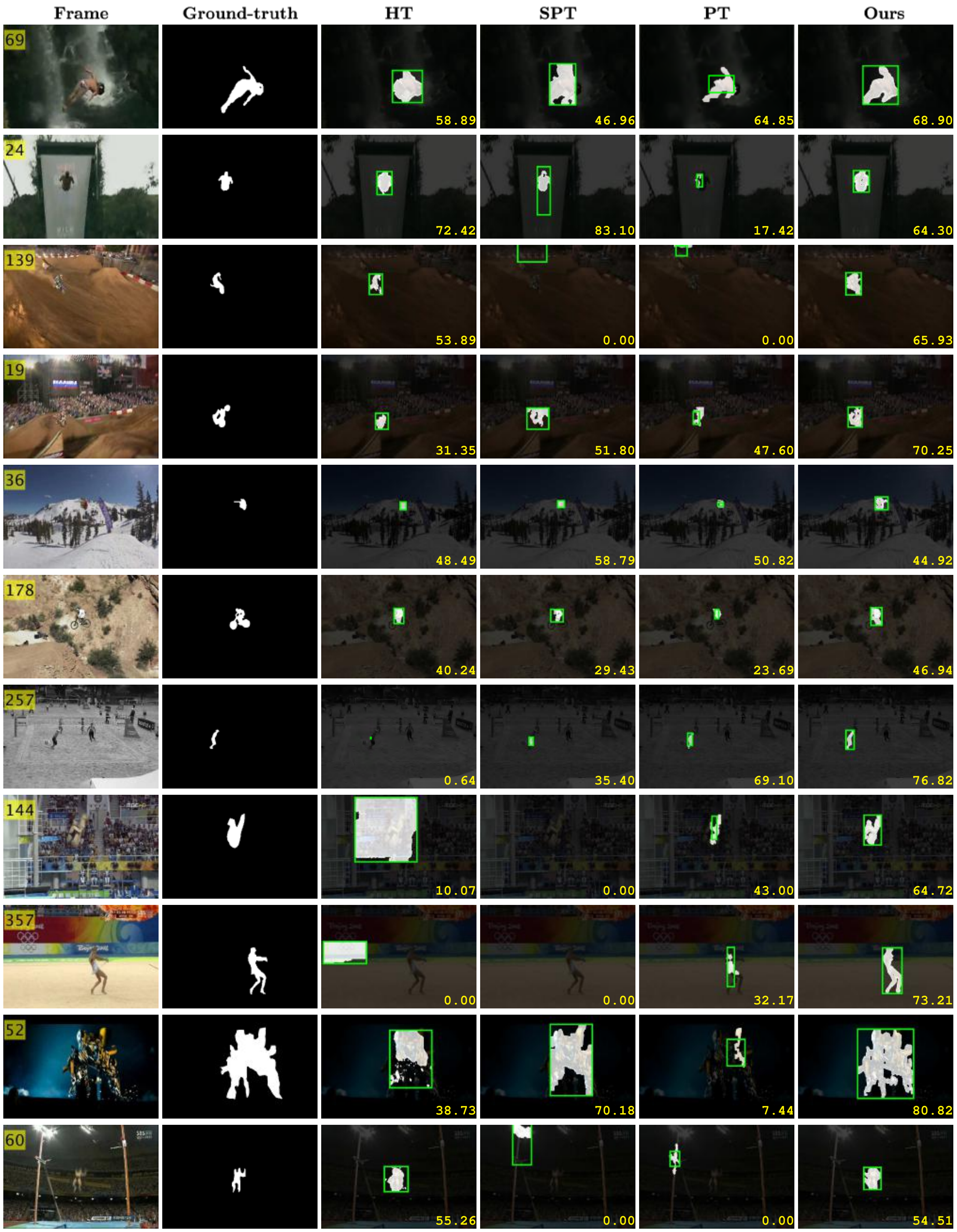


Figure 3. Qualitative performance evaluation. The order of sequences is same as the order in Table 2.

References

[1] C. Aeschliman, J. Park, and A. C. Kak. A probabilistic framework for joint segmentation and tracking. In *CVPR*, pages 1371–1378, 2010. 1

[2] S. Avidan. Ensemble tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):261–271, 2007. 1

[3] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1619–1632, 2011. 2

[4] V. Belagiannis, F. Schubert, N. Navab, and S. Ilic. Segmentation based particle filtering for real-time 2d object tracking. In *ECCV*, pages 842–855, 2012. 1

[5] V. L. C. Becker, R. Rigamonti and P. Fua. Supervised feature learning for curvillinear structure segmentation. In *MICCAI*, 2013. 2, 3, 6

[6] T. B. Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *CVPR*, pages 1177–1184, 2011. 1

[7] S. Duffner and C. Garcia. Pixeltrack: a fast adaptive algorithm for tracking non-rigid objects. In *ICCV*, pages 2480–2487, 2013. 1, 6

[8] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) challenge. *Int. J. Comput. Vision*, 88(2), June 2010. 6

[9] J. H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000. 2

[10] M. Godec, P. M. Roth, and H. Bischof. Hough-based tracking of non-rigid objects. In *ICCV*, pages 81–88, 2011. 1, 5, 6

[11] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *BMVC*, page 6, 2006. 2

[12] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *ECCV*, pages 234–247, 2008. 2

[13] B. Han, D. Comaniciu, Y. Zhu, and L. Davis. Sequential kernel density approximation and its application to real-time visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(7):1186–1197, 2008. 1

[14] B. Han and L. S. Davis. Probabilistic fusion-based parameter estimation for visual tracking. *Computer Vision and Image Understanding*, 113(4):435–445, 2009. 1

[15] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *ICCV*, pages 263–270, 2011. 6

[16] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (MUSTer): a cognitive psychology inspired approach to object tracking. In *CVPR*, 2015. 6, 7

[17] Z. Hong, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Tracking using multilevel quantizations. In *ECCV*, pages 155–171, 2014. 1, 6

[18] M. Isard and A. Blake. Condensation – Conditional density propagation for visual tracking. *Int. J. Comput. Vision*, 29(1), 1998. 4

[19] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, L. Cehovin, G. Nebehay, G. Fernandez, T. Vojir, A. Gatt, et al. The visual object tracking VOT2013 challenge results. In *ICCVW*, pages 98–111, 2013. 5

[20] S. Kwak, W. Nam, B. Han, and J. H. Han. Learning occlusion with likelihoods for visual tracking. In *ICCV*, pages 1551–1558, 2011. 1

[21] J. Kwon and K. M. Lee. Tracking of abrupt motion using wang-landau monte carlo estimation. In *ECCV*, pages 387–400, 2008. 1

[22] 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. In *CVPR*, pages 1208–1215, 2009. 1, 5

[23] X. Mei and H. Ling. Robust visual tracking using ℓ_1 minimization. In *ICCV*, pages 1436–1443, 2009. 1

[24] H. Nam, S. Hong, and B. Han. Online graph-based tracking. In *ECCV*, 2014. 1

[25] N. C. Oza. Online bagging and boosting. In *IEEE International Conf. on Systems, Man and Cybernetics*, pages 2340–2345, 2005. 6

[26] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *Int. J. Comput. Vision*, 77(1-3):125–141, 2008. 1

[27] C. Rother, V. Kolmogorov, and A. Blake. GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004. 1, 5, 6

[28] S. Wang, H. Lu, F. Yang, and M.-H. Yang. Superpixel tracking. In *ICCV*, pages 1323–1330, 2011. 1, 6

[29] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, pages 2411–2418, 2013. 5, 6, 7

[30] J. Zhang, S. Ma, and S. Sclaroff. MEEM: robust tracking via multiple experts using entropy minimization. In *ECCV*, pages 188–203. Springer, 2014. 6, 7

[31] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, pages 1838–1845, 2012. 6