Robust Discriminative Tracking via Query-by-Bagging

Kourosh Meshgi, Shigeyuki Oba, Shin Ishii
Kyoto University
Yoshida-Honmachi, Kyoto 606-8501, Japan
meshgi-k@sys.i.kyoto-u.ac.jp

Abstract

Adaptive tracking-by-detection is a popular approach to track arbitrary objects in various situations. Such approaches treat tracking as a classification task and constantly update the object model. The update procedure requires a set of labeled examples, where samples are collected from the last observation, and then labeled. However, these intermediate steps typically follow a set of heuristic rules for labeling and uninformed search in the sample space, which decrease the effectiveness of model update. In this study, we present a framework for adaptive tracking that utilizes active learning for effective sample selection and labeling them. The active sampler employs a committee of randomized-classifiers to select the most informative samples and query their label from an auxiliary detector with a long-term memory. The committee is then updated with the obtained labels. Experiments show that our algorithm outperforms state-of-the-art trackers on various benchmark videos.

1. Introduction

In visual object tracking, robustness against ever-changing object in complex environments is an essential requirement in video surveillance, human-machine interfaces, driving-assistant systems and robotics applications. Although some settings allow for strong assumptions about the target, in real-world applications it is desired to track arbitrary objects with little a-priori knowledge. Such model free tracker consist of learning and adjusting the representation of the target on-the-fly. Using tracking-by-detection approaches is a popular trend in recent years, due to significant breakthroughs in object detection domain, yielding strong discriminative power wih offline training. Adopted for visual tracking, many of such trackers are adjusted for online training and accumulate knowledge about a target with each successful detection.

However, there are a multitude of drawbacks in the tracking-by-detection setting: *(i)* The sampling in search

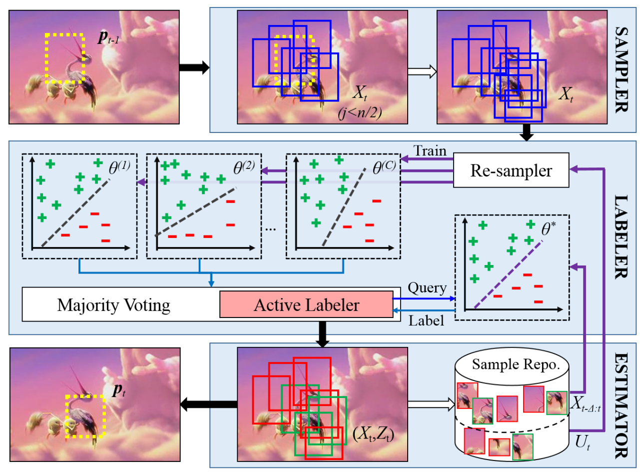


Figure 1. Schematic of the system. The proposed tracker, QBT, utilizes a novel adaptive sampling strategy and collect samples for the active labeler. The labeler performs bagging on an ensemble of randomized classifiers, and query the disputed samples from a complete classifier. The labels are then propagated to the next stage, where the location and scale of the target is estimated. Finally, the ensembe classifiers of the system are updated in a query-by-bagging [1] fashion. To robustify the tracker against motion and appearance jitters, the complete classifier is updated with longer intervals.

space is sparse [13], uninformed and ignores the information contained in each sample about its local neighborhood. In this way, the tracker miss the promising regions of the search space, and blindly examine many possible transformations before finding the optimal one. *(ii)* Classifier is trained using all the examples with equal weights, meaning that negative examples which overlap very little with the target bounding box are treated equally as those negative examples with significant overlaps. Even the slightest tracking error, may lead to poorly labeled examples that cause the tracker to drift. *(iii)* The labeler is typically build upon heuristics and intuitions, rather than using the accumulated knowledge about the target for example. Mistakes in labeling confuses the classifier and is known as *label noise* problem [11]. *(iv)* Adaptive trackers inherently suffer from

978-1-5090-3811-4/16/$31.00 ©2016 IEEE

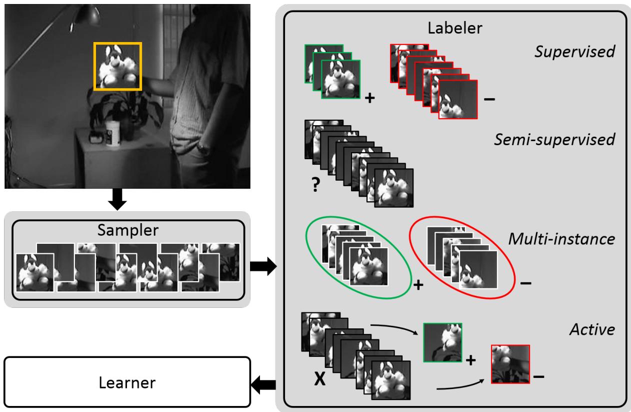


Figure 2. Different adaptive tracking-by-detection paradigms: given the current estimated object location, these approaches generate a set of samples and, depending on the type of learner, produce training labels. Active learning, utilizes the most informative samples to efficiently and effectively update the learner. Unlike others, Struck and operates directly on the tracking output [11].

drifting problem, if the update rate is small, the changes of the target are not reflected into target's template, whereas rapid update of the tracker render it vulnerable to data noise and small target localization errors. This is another instance of *stability plasticity dilemma* [9].

Current state-of-the-art tracking-by-detection algorithms are trying to overcome some of these shortcomings. Label noise problem is tackled by robust loss functions [17, 19], semi-supervised learning [9, 24], or multi-instance learning [4, 32]. Some other trackers utilize the efficient sparse sampling [13], context information of the examples [7, 10] or sample informativeness for the classifier [34] to bolster the sampling procedure. Furthermore, Struck [11] combines the labeling and learning step by reformulating the tracking-by-detection framework to resolve the label noise problem. Drifting problem have been tackled by utilizing leaky memories [21], online mixture model [14], dictionary updating [35] and incremental subspace update [23]. Some researchers believe in the necessity of having a "teacher" to train the classifier [9]. Adaptive ensemble of classifiers [3] and co-learning [29] in which multiple trackers with different features or inference engines train each other aimed to address this need using other detectors or trackers. Furthermore, some approaches selected the most discriminative feature selection [8, 6] or combined generative and discriminative models [30] to overcome this problem. Generally, discriminative models strongly depend on the sample collection part to robustify the trained classifier. To address all of the aforementioned issues, we propose a framework, in which the contents of the samples are utilized to guide the tracker in sampling and labeling processes, the need for a teacher is minimized for the classifier, and weigh the examples based on their information value.

Having a strong classifier for a general tracking-by-detection requires embedding a lot of prior knowledge to the tracker (e.g., a pre-trained detector) or sophisticated online learning mechanism. Lets assume we have a trained classifier (hereafter *Oracle*). We wish to minimize the need of querying this oracle in the course of our tracking that is known as *query selection problem*.

One of the most theoretically-motivated query selection frameworks is Query-by-Committee (QBC) algorithm [26, 25] that maintain a committee of models which are all trained on the current labeled set, but represent competing hypotheses. Each committee member is then allowed to vote on the labeling of query candidates. The most informative query is considered to be the instance about which they most disagree. The premise behind the QBC framework is minimizing the *version space*, which is the set of hypotheses that are consistent with data.

The original QBC was built upon randomized component learning algorithm. For other model classes, such as discriminative or non-probabilistic models, Abe and Mamitsuka [1] have proposed Query-by-Bagging (QBag), which employ the popular ensemble learning model bagging [5] to construct committees. Bagging is a technique to enhance the performance of the existing learning algorithm by running it many times on a set of re-sampled governed by a uniform distribution and the final hypothesis is obtained by taking majority vote over the output of predictions of the output hypotheses. QBag introduces the randomness in the form of re-sampling from the input data, and is based on the idea that prediction error consist of the *bias*, which is the estimation error due to the smaller input size, and the *variance* which is explained by the statistical variation existing in specific data. Bagging can isolate bias from variance and minimize the latter [1].

We propose Query-by-Bagging Tracker (QBT), which adjust QBag algorithm for online training to solve the label noise problem. Additionally, the drift problem is handled with the use of a dual-memory strategy, where the committee adapts to the changes of target rapidly, whereas the oracle possesses a longer memory to promote the stability of the target template. Furthermore, a k-means-like approach is proposed to improve the sampling process and the tracker considers a weighted vote of the samples to estimate the target location and scale. The proposed approach shows excellent performance in comparison with 15 state-of-the-art trackers on 100 challenging video sequences.

2. Proposed Framework

In the following section, an overview of adaptive tracking-by-detection approaches is provided, their issues are discussed, and the proposed framework to overcome that shortcomings is elaborated.

2.1. Tracking by Detection

A tracking-by-detection algorithm attempts to learn a classifier to distinguish a target object from its local background. The position of the target is denoted by bounding box $\mathbf{p}_t$, that contains the image patch $\mathbf{x}_t^{\mathbf{P}}$ of frame I_t where $t = 1, \ldots, T$ is the time. The features $\phi_t^{\mathbf{P}}$ extracted from the image patch $\mathbf{x}_t^{\mathbf{P}} \in \mathcal{X}_t$ are given to the classifier. Formally, the classifier is trained with example pairs $(\mathbf{x}, z)$, where $z = \pm 1$ is the binary label, and makes its predictions according to $\hat{z} = \mathrm{sign}(h(\mathbf{x}))$ with $h : \mathcal{X} \to \mathbb{R}$ denotes the *classification confidence function*.

The target position in next frame, is assumed to be found in the local neighborhood of its previous location $\mathbf{p}_{t-1}$, where the value of h is maximized. The tracker aims to find the transformation $\mathbf{y}_t \in \mathcal{Y}$, where $\mathcal{Y}$ is called the *search space* and its form depends on the expected transformation of the target (e.g., 2D affine transformation [16]). Typically, in tracking-by-detection algorithms the case of 2D translation is used, in which $\mathcal{Y} = \{(u, v)|u^2 + v^2 < r^2\}$ where r is search radius [11]. With transformation $\mathbf{y}_t$, the new position of the object is approximated by the composition $\mathbf{p}_t = \mathbf{p}_{t-1} \circ \mathbf{y}_t$, thus the classifier estimates the relative transformation between two frames with

$$\mathbf{y}_t = \underset{\mathbf{y} \in \mathcal{Y}}{\mathrm{argmax}} \; h(\mathbf{x}^{\mathbf{p}_{t-1} \circ \mathbf{y}_t}). \tag{1}$$

Having located the target object, a set of training examples are selected from the current frame to update the classifier to reflect the recent changes of target appearance. This process is separated into *sampling* and *labeling* phases. The sampling process generates a set of n different transformations $\{\mathbf{y}_t^1, \ldots, \mathbf{y}_t^n\}$ that yields the set of training examples $\{\mathbf{x}_t^{\mathbf{p}_t \circ \mathbf{y}_t^1}, \ldots, \mathbf{x}_t^{\mathbf{p}_t \circ \mathbf{y}_t^n}\}$. Depending on the type of classifier, the labels $\{z_t^1, \ldots, z_t^n\}$ are selected for these samples and the classifier is updated using these samples and their labels.

The labeling process typically employs a *transformation similarity function* to determine the label of sampled position $\mathbf{p}_t \circ \mathbf{y}_t^i$. Such function take the form of $s_{\mathbf{p}}(\mathbf{y}_t^i, \mathbf{y}_t^j) \in \mathbb{R}$, that measures the similarity of two patches that are obtained from transformations $\mathbf{y}_t^i$ and $\mathbf{y}_t^j$ with respect to $\mathbf{p}$. In order to determine the label z_t^i of a sample generated by transformation $\mathbf{y}_t^i$, a labeling function $\ell(.)$ is utilized such that $z_t^i = \ell(s_{\mathbf{p}}(\mathbf{y}_t^i, \mathbf{y}_t^j))$. This function is expressed as

$$\ell(s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^i))) = \begin{cases} +1 & s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^i)) > \tau_u \\ -1 & s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^i)) < \tau_l \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where τ_u and τ_l are upper and lower thresholds, and $\mathbf{y}^0$ is considered as null transformation such that $\mathbf{p} = \mathbf{p} \circ \mathbf{y}^0$. Most, if not all, variants of tracking-by-detection algorithms use a labeler which can be expressed in a similar fashion [11].

Algorithm 1: Query-by-Bagging Tracker

input : Target position in last frame $\mathbf{p}_{t-1}$
output: Target position in current frame $\mathbf{p}_t$

Initiate search center $\mu_t \leftarrow \mathbf{p}_{t-1}$
for $j \leftarrow 1$ **to** n **do**
  Sample transformation $\mathbf{y}_t^j \sim \mathcal{N}(\mu_t, \Sigma_{search})$
  Update search center μ_t (eq(4))
  if $s_t^{overlap}(\mathbf{y}^0, \mathbf{y}_t^j) < \tau_{bkg}$ **then** Sample is too far
    Label the sample $z_t^j \leftarrow -1$
  else Sample worth investigating
    Label the sample $z_t^j \leftarrow \ell(s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^j)))$ (eq(6))
    if $\tau_l < s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^j)) < \tau_u$ **then**
      $\mathcal{U}_t \leftarrow \mathcal{U}_t \cup \{\langle \mathbf{x}^{\mathbf{p}_{t-1} \circ \mathbf{y}_t^j}, z_t^j \rangle\}$
for $c \leftarrow 1$ **to** C **do**
  Draw m *samples from* $\mathcal{U}_t$
  Update committee model $\theta^{(c)}$ *with them*
Update oracle model θ^* *with all samples of* $\mathcal{X}_t$
Approximate transformation $\hat{\mathbf{y}}_t$ (eq(7))
Calculate target position $\mathbf{p}_t = \mathbf{p}_{t-1} \circ \hat{\mathbf{y}}_t$

Many of the tracking-by-detection schemes use spatial distance function as their transformation similarity functions, where closer patches are assumed as positive samples and further ones are considered as background and serve as negative samples, i.e., $s_{\mathbf{p}}^{dist}(\mathbf{y}_t^i, \mathbf{y}_t^j)) = -dist(\mathbf{y}_t^i, \mathbf{y}_t^j)$. Another example of such function is based on overlap between two bounding boxes,

$$s_{\mathbf{p}}^{overlap}(\mathbf{y}_t^i, \mathbf{y}_t^j)) = \frac{(\mathbf{p}_t \circ \mathbf{y}_t^i) \cap (\mathbf{p}_t \circ \mathbf{y}_t^j)}{(\mathbf{p}_t \circ \mathbf{y}_t^i) \cup (\mathbf{p}_t \circ \mathbf{y}_t^j)} \tag{3}$$

where based on labeler function in eq (2), the boxes with large overlap with current estimated target are selected as positive samples. The unlabeled examples are generally ignored in binary classifiers [8], whereas in trackers based on semi-supervised learning they are used to update the classifier [9, 24]. Labelers in trackers based on multi-instance learning [4, 32, 34] collect examples in bags and assigns the label to the bag based on the majority vote of the examples inside them (Figure 2).

2.2. Query-by-Bagging Tracker (QBT)

We propose an online query selection mechanism called QBT. QBT is made of an ensemble of classifiers $\mathcal{C} = \{\theta^{(1)}, \ldots, \theta^{(C)}\}$, called the *committee*, and a fine special classifier θ^*, called the *oracle*. We throe=w a query to the oracle when the ensemble classifiers show the highest degree of uncertainty (Figure 1).

To localize the target at time t, n samples are drawn from the Gaussian distribution $\mathcal{N}(\mu_t, \Sigma_{search})$ where μ_t is the mean of distribution (the mean of target bounding box location and its scale), and Σ_{search} is the search space variance. To guide the search space toward obtaining better samples, the search center should start from last known position of the target $\mathbf{p}_{t-1}$, then move towards the positive samples and away from negative samples. Every sampling phase involves sampling n new samples from the frame I_{t+1}. In our approach, half of these samples are drawn with the center of last know target position $\mathbf{p}_t$, and after that, with every new sample, based on its label from the committee, the search space is slightly shifted. For sample $j = 1, \ldots, n$, the search center is updated as

$$\mu_t = \begin{cases} p_t & , j < \frac{n}{2} \\ \mu_t + \alpha \mathrm{sign}(s_{\mathbf{p}}^{qbag}(\mathbf{y}^0, \mathbf{y}_t^j)) \frac{\mathbf{p}_t - (\mathbf{p}_t \circ \mathbf{y}_t^j)}{\|\mathbf{p}_t - (\mathbf{p}_t \circ \mathbf{y}_t^j)\|_2} & , j \geq \frac{n}{2} \end{cases} \tag{4}$$

where α is a constant, the sign of the $s_{\mathbf{p}}^{qbag}(\mathbf{y}^0, \mathbf{y}_t^j)$ that is defined later in eq(5), determines the orientation of the step, and the step size is proportional to the sample distance from the search center ($\| \ . \ \|_2$ denotes Euclidean distance).

Having a committee, the transformation similarity matrix is reduced to the vote score of the different classifiers of committee $\mathcal{C}$,

$$s_{\mathbf{p}}^{qbag}(\mathbf{y}_t^i, \mathbf{y}^0) = \sum_{c=1}^{C} \mathrm{sign}\left(h(\mathbf{x}^{\mathbf{p}_t \circ \mathbf{y}_t^i} | \theta^{(c)}) \right). \tag{5}$$

If the committee is somewhat unanimous about the label of a sample j, it is used as the label of the sample z_t^j. On the other hand, if the committee disagrees about the label of this sample, based on the principle of query-by-bagging, the sample is labeled by the oracle and is added to the list of uncertain samples $\mathcal{U}_t$.

$$\ell(s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^i))) = \begin{cases} +1 & s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^i)) > \tau_u \\ -1 & s_{\mathbf{p}}(\mathbf{y}^0, \mathbf{y}_t^i)) < \tau_l \\ \mathrm{sign}(h(\mathbf{x}^{\mathbf{p}_t \circ \mathbf{y}_t^i}) | \theta^*) & \text{otherwise} \end{cases} \tag{6}$$

where τ_u and τ_l are thresholds with which the tracker controls its reliance on the oracle. It should be mentioned that we automatically assign negative labels to the patches that has a small overlap with the target patch $\mathbf{p}_t$ based on eq(3).

After sampling and labeling, the classifiers should be updated to reflect the recent changes of the target. Inspired by query-by-bagging framework, we randomly re-sample m samples from $\mathcal{U}_t$ to train every model $\theta^{(i)}$ of committee $\mathcal{C}$ ($m < n$). Additionally, all samples are stacked and every 10 frames, the oracle model θ^* is updated with these samples. This strategy, update the committee in a frame-by-frame basis to reflect the latest target changes, but grant a longer-memory to oracle to navigate the tracker through occlusions and other temporal inconsistencies of target appearance.

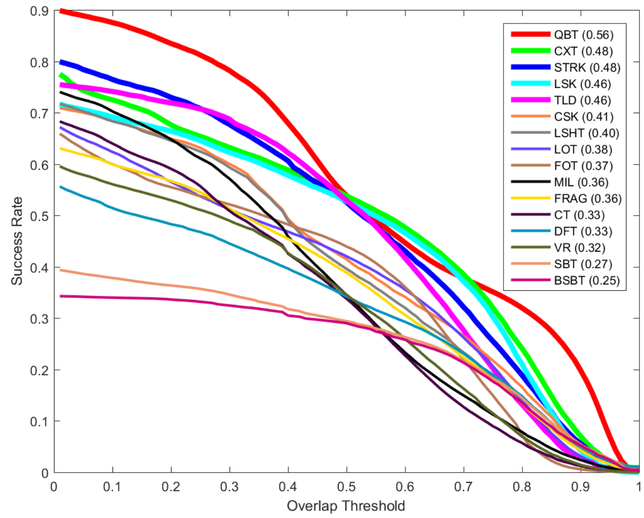


Figure 3. Quantitative comparison of the proposed tracker, QBT, with the state-of-the-art trackers using success plot and its AUC.

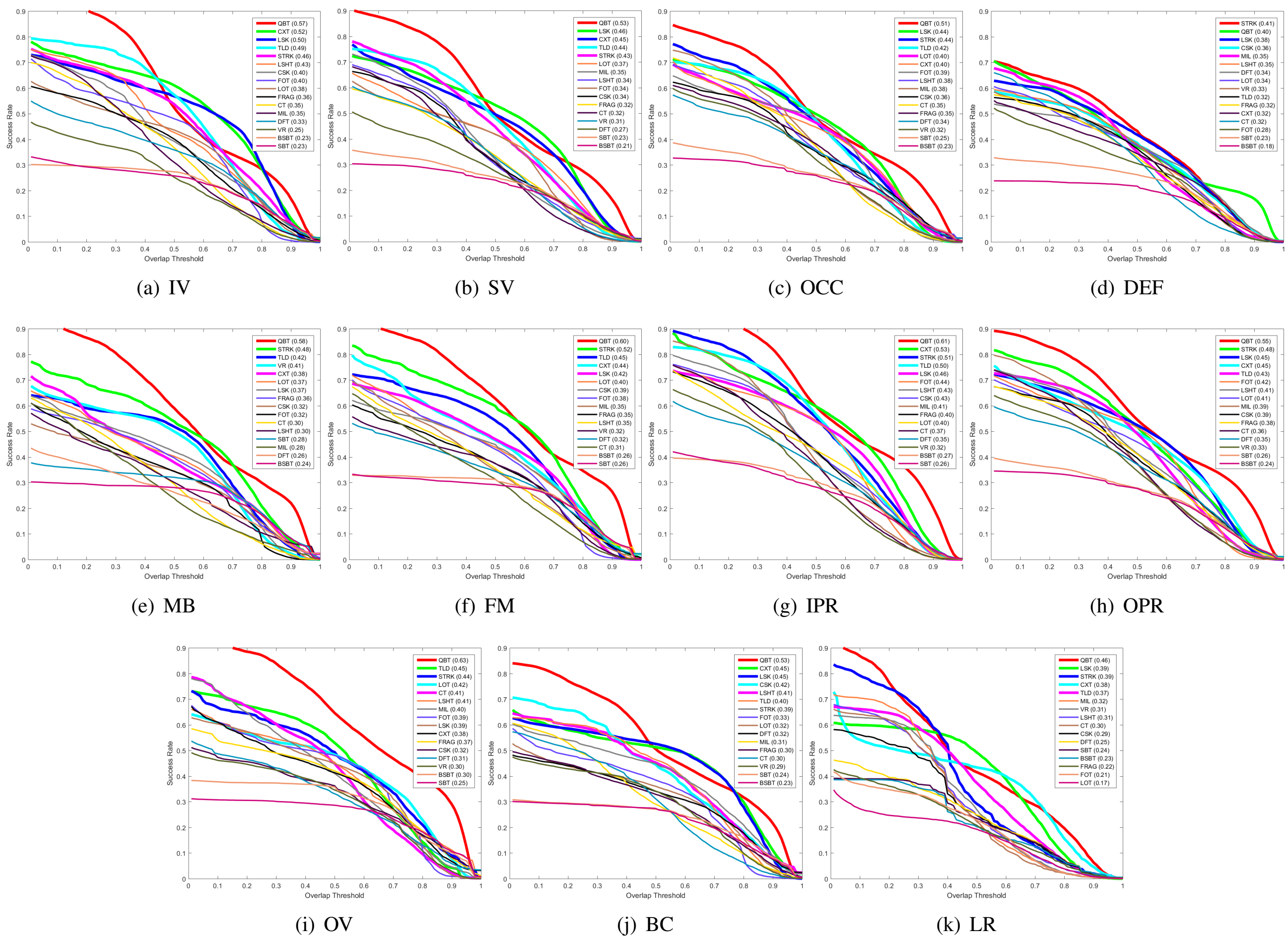
Finally, the target transformation is approximated by weighted averaging of all positive samples (the weight is their committee score):

$$\hat{\mathbf{y}}_t = \frac{\sum_{\mathbf{y}_t \in \mathcal{Y}} \mathrm{R}(s_{\mathbf{p}}^{qbag}(\mathbf{y}_t^i, \mathbf{y}^0))\mathbf{y}_t}{\sum_{\mathbf{y}_t \in \mathcal{Y}} \mathrm{R}(s_{\mathbf{p}}^{qbag}(\mathbf{y}_t^i, \mathbf{y}^0))} \tag{7}$$

where $\mathrm{R}(x)$ is the ramp function that equals x for $x > 0$ and 0 otherwise. Algorithm 1 summarizes the proposed tracker.

3. Experiments

This section reports on the experiments comparing the QBT with relevant algorithms on benchmark sequences that are commonly used in the literature. The experiment is conducted on 100 challenging video sequences from [31], which involves many visual tracking challenges such as illumination variation (IV), scale variation (SV), occlusions (OCC), deformations (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-play rotation (OPR), out-of-view problem (OV), background clutter (BC) and low resolution (LR). The performance of the occlusion is measures by the area under the surface of its success plot (AUC). The performance of the tracker is measured by the area under the surface of its success plot. A tracker in time t succeed to track the object if its response $\mathbf{p}_t$ overlaps with the ground truth $\mathbf{p}_t^*$ more than a threshold τ_{ov}. Success plot, graphs the success of the tracker against different values of



(a) IV (b) SV (c) OCC (d) DEF

(e) MB (f) FM (g) IPR (h) OPR

(i) OV (j) BC (k) LR

Figure 4. Quantitative evaluation of trackers under different visual tracking challenges. The best performance is plotted with **red**, while the second and third best performance is depicted with **green** and **blue** lines respectively. QBT outperformed other trackers when dealing with different tracking challenges at all the panels except 4(d) in which QBT achieved the second place.

the threshold τ_{ov} and its AUC is obtained from

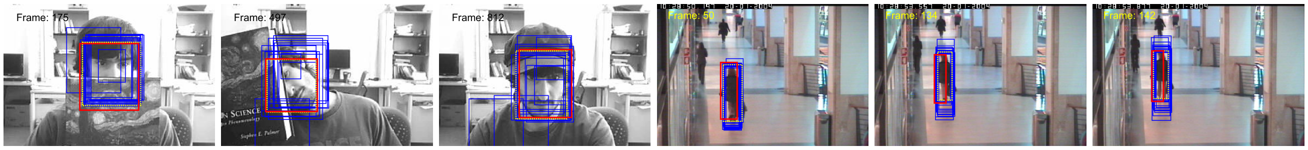
$$AUC = \frac{1}{T}\int_0^1 \sum_{t=1}^{T} u\left(\frac{|\mathbf{p}_t \cap \mathbf{p}_t^*|}{|\mathbf{p}_t^* \cup \mathbf{p}_t^*|} > \tau_{ov}\right) d_\tau \tag{8}$$

where T is the length of sequence, $|.|$ denotes the area of the region, $\cap$ and $\cup$ stands for intersection and union of the regions respectively, and $u(.)$ denotes the step function that returns 1 iff its argument is positive and 0 otherwise.

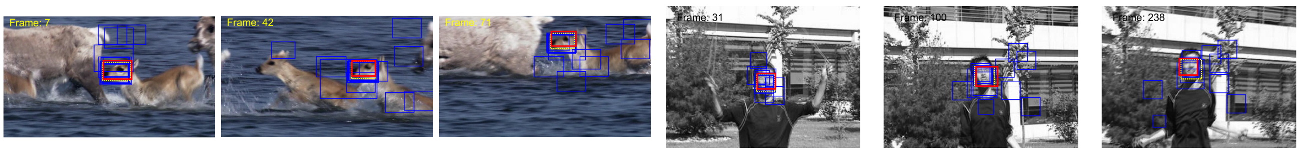
To establish a fair comparison with the state-of-the-art, we select some of the most popular discriminative and generative trackers (according to a recent large benchmark [31]): BSBT [28], CSK [13], CT [33], CXT [7], DFT [27], FOT [20], FRAG [2], LOT [22], LSHT [12], LSK [18], MIL [4], SBT [10], STRUCK [11], TLD [15], and VR [6]. We perform a benchmark on the whole videos of the dataset, along with partial subsets of the dataset with a distinguishing attribute to evaluate the tracker performance under different situations.

Figure 3 depicts the overall performance of the proposed tracker against other benchmarked algorithms on the all sequences of the dataset. The plots shows that QBT has a superior performance on this dataset. It also reveals that the tracker has many accurate estimations of the target (sharp slope between $0.9 \geq \tau_{ov} > 1$). Furthermore, the other steep slope around $\tau_{ov} \approx 0.4$ and high value when $\tau_{ov} \to 0$ suggest that tracker was able to keep track of the target in most cases, and the devised scheme effectively reduced the drift problem.

Figure 4 present the performance of the trackers, in the case of prominent tracking challenges. The proposed tracker, significantly perform better than its competitors in the cases of illumination variation (Fig.4(a)), occlusion (Fig.4(c)), in-plane and out-of-plane rotations (Fig.4(g) and 4(h)), out-of-view target (Fig.4(i)), background clutter (Fig.4(j)), and low resolution (Fig.4(k)) because of its effective use of a committee of classifiers and long-memory oracle that handle a variety of appearance changes. It is consid-

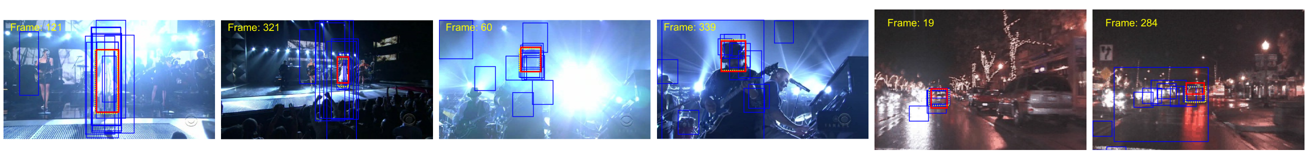


(a) Tracking results of sequence *FaceOcc2* and *Walking2* with severe occlusions

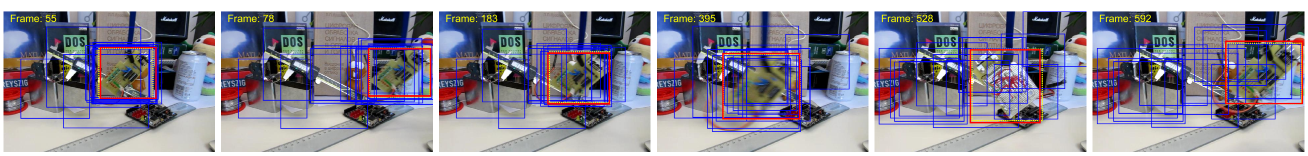


(b) Tracking results of sequence *Deer* and *Jumping* with motion blur

(c) Tracking results of sequence *Girl* and *Ironman* with in-plane and out-of-plane rotations



(d) Tracking results of sequence *Singer1*, *Shaking* and *CarDark* with drastic illumination changes



(e) Tracking results of sequence *Board* with background clutter

Figure 5. Sample tracking results of evaluated algorithms on several challenging video sequences. In these sequences the red box depicts the QBT against other trackers (blue). The ground truth is illustrated with yellow dashed box. The results are available in the `webpage`.

erably successful in cases of scale variation (Fig.4(b)), motion blur (Fig.4(e)) and fast motion (Fig.4(f)) thanks to the informed search scheme and sample averaging that robustify the tracker to changes in target size, velocity changes, and its motion trail blur in the sequences. The performance of QBT is comparable to that of STRUCK in deformation case (Fig.4(d)), however, there is room for improvement by allowing QBT to sample from more general forms of transformations (e.g. 3D translations or affine transformations). Nevertheless, even in this plot, QBT shows high accuracy tracking more than other trackers (look at the sharp slope between $0.9 \geq \tau_{ov} > 1$). A qualitative comparison of the QBT and other trackers is presented in Figure 5.

4. Conclusions

This study proposed to employ a committee of classifiers, each trained incrementally on a randomized portion of the latest obtained training samples, to enhance the discriminative power of the tracker. This idea is inspired from query-by-bagging framework that follow the version-space shrinking strategy to distinguish the most informative samples. Such samples are then queried from an auxiliary classifier with longer memory that is robust against fluctuations in target appearance and occlusions. Another novelty of this study is to use a guided search in sample space, to find more suitable and relevant samples for better training the tracker classifiers. Furthermore, a solution is proposed to compensate of over-reliance of the tracker on the classifiers confidence function. The proposed tracker, QBT, incorporates all of these solutions in a discriminative tracking framework and outperform state-of-the-art discriminative and generative trackers on a large video dataset with various types of tracking challenges such as appearance changes and occlusions.

References

[1] N. Abe and H. Mamitsuka. Query learning strategies using boosting and bagging. In *ICML'98*, 1998. 1, 2

[2] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR'06*, 2006. 5

[3] S. Avidan. Ensemble tracking. *PAMI*, 2007. 2

[4] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR'09*, 2009. 2, 3, 5

[5] L. Breiman. Bagging predictors. *Machine learning*, 1996. 2

[6] R. T. Collins, Y. Liu, and M. Leordeanu. Online selection of discriminative tracking features. *PAMI*, 2005. 2, 5

[7] T. B. Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *CVPR'11*, 2011. 2, 5

[8] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *BMVC'06*, 2006. 2, 3

[9] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *ECCV'08*. 2008. 2, 3

[10] H. Grabner, J. Matas, L. Van Gool, and P. Cattin. Tracking the invisible: Learning where the object might be. In *CVPR'10*, 2010. 2, 5

[11] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *ICCV'11*, 2011. 1, 2, 3, 5

[12] S. He, Q. Yang, R. Lau, J. Wang, and M.-H. Yang. Visual tracking via locality sensitive histograms. In *CVPR'13*, 2013. 5

[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *PAMI*, 2015. 1, 2, 5

[14] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *PAMI*, 2003. 2

[15] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *PAMI*, 2012. 5

[16] J. Kwon, K. M. Lee, and F. C. Park. Visual tracking via geometric particle filtering on the affine group with optimal importance functions. In *CVPR'09*, pages 991–998, 2009. 3

[17] C. Leistner, A. Saffari, P. M. Roth, and H. Bischof. On robustness of on-line boosting-a competitive study. In *ICCVw'09*, 2009. 2

[18] B. Liu, J. Huang, L. Yang, and C. Kulikowsk. Robust tracking using local sparse appearance model and k-selection. In *CVPR'11*, 2011. 5

[19] H. Masnadi-Shirazi, V. Mahadevan, and N. Vasconcelos. On the design of robust classifiers for computer vision. In *CVPR'10*, 2010. 2

[20] J. Matas and T. Vojir. Robustifying the flock of trackers. In *CVWW'11*, 2011. 5

[21] I. Matthews, T. Ishikawa, and S. Baker. The template update problem. *PAMI*, 2004. 2

[22] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan. Locally orderless tracking. *IJCV*, 2015. 5

[23] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 2008. 2

[24] A. Saffari, C. Leistner, M. Godec, and H. Bischof. Robust multi-view boosting with priors. In *ECCV'10*. 2010. 2, 3

[25] B. Settles. *Active learning*. Morgan & Claypool Publishers, 2012. 2

[26] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294. ACM, 1992. 2

[27] L. Sevilla-Lara and E. Learned-Miller. Distribution fields for tracking. In *CVPR'12*, 2012. 5

[28] S. Stalder, H. Grabner, and L. Van Gool. Beyond semi-supervised tracking: Tracking should be as simple as detection, but not simpler than recognition. In *ICCVw'09*, 2009. 5

[29] F. Tang, S. Brennan, Q. Zhao, and H. Tao. Co-tracking using semi-supervised support vector machines. In *ICCV'07*, 2007. 2

[30] T. Woodley, B. Stenger, and R. Cipolla. Tracking using online feature selection and a local generative model. In *BMVC*, 2007. 2

[31] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *PAMI*, 2015. 4, 5

[32] B. Zeisl, C. Leistner, A. Saffari, and H. Bischof. On-line semi-supervised multiple-instance boosting. In *CVPR'10*, 2010. 2, 3

[33] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In *ECCV'12*. Springer, 2012. 5

[34] K. Zhang, L. Zhang, M.-H. Yang, and Q. Hu. Robust object tracking via active feature selection. *IEEE CSVT*, 2013. 2, 3

[35] T. Zhang, B. Ghanem, C. Xu, and N. Ahuja. Object tracking by occlusion detection via structured sparse learning. In *CVPRw'13*, 2013. 2