Adaptive Aggregation of Arbitrary Online Trackers with a Regret Bound

Heon Song[1,2] Daiki Suehiro[1,2] Seiichi Uchida[1]
[1]Kyushu University [2]RIKEN AIP
{heon.song@human., suehiro@, uchida@}ait.kyushu-u.ac.jp

Abstract

We propose an online visual-object tracking method that is robust even in an adversarial environment, where various disturbances may occur on the target appearance, etc. The proposed method is based on a delayed-Hedge algorithm for aggregating multiple arbitrary online trackers with adaptive weights. The robustness in the tracking performance is guaranteed theoretically in term of "regret" by the property of the delayed-Hedge algorithm. Roughly speaking, the proposed method can achieve a similar tracking performance as the best one among all the trackers to be aggregated in an adversarial environment. The experimental study on various tracking tasks shows that the proposed method could achieve state-of-the-art performance by aggregating various online trackers.

1. Introduction

Visual-object tracking is still an active research topic [22] and is classified into two types: offline tracking and online tracking. Offline tracking provides the location of the target object at individual frames after obtaining all the frames. Offline tracking usually relies on a global optimization over all frames and therefore the location at frame t is determined not only by the past frames but also by the future frames. Accordingly, it is robust to large variations in the target appearance due to variations such as illumination changes, geometric deformation, and partial or complete occlusion; however, offline tracking cannot realize real-time tracking in principle.

Online tracking has the opposite properties. It can realize real-time tracking; however, it is difficult to realize an accurate and stable online tracker. Fig. 1 shows a ranking histogram based on AUC of seven state-of-the-art online trackers for 50 different image sequences. This histogram proves experimentally that there is no almighty online tracker, which never becomes lower-ranked.

One remedy to improve the robustness of online tracking is to use multiple online trackers. By aggregating them appropriately, it should be possible to realize a more robust online tracker. As an example, assume that there are two online trackers; one is robust to illumination changes and the other is robust to geometric deformations of the target. If we can select one of them by predicting appearance variation, we can expect a more robust tracking result. We can also develop hard-switching or soft-switching algorithms; the former selects one tracker at each frame and the latter merges trackers with weights that are updated at every frame or after a certain number of frames.

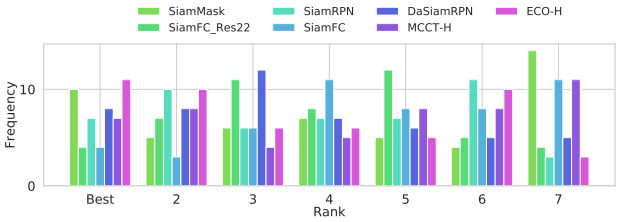

Figure 1. No "almighty" tracker — the raking histogram based on AUC of state-of-the-art online trackers for all 50 sequences of OTB2013 [26]. A smaller rank means better performance.

However, in general, it is still difficult to realize a robust online tracker using the above simple aggregation algorithms. In the practical situation, we cannot know which online tracker will show good performance due to the essential difficulty of predicting the future environment. The weakness will be more enhanced when we consider an *adversarial environment* [21]. Roughly speaking, the adversarial environment is defined as the worst situation for most online trackers as well as the simply aggregated tracker. This adversarial environment seems very rare or even unrealistic; however, it is still important to theoretically understand the fact that a simple aggregation algorithm may not be able to avoid serious errors. Moreover, if we want to develop a mightier tracker that works in arbitrary situations (i.e., arbitrary videos), it is worthy to have some theoretical guarantee by understanding the worst-case performance of the tracker.

In this paper, we propose an aggregation-based online tracking method, called *delayed-Hedge online tracking*. The performance of the proposed method is theoretically guaranteed even in adversarial environments and based on a weighted aggregation of N arbitrary online trackers. Its

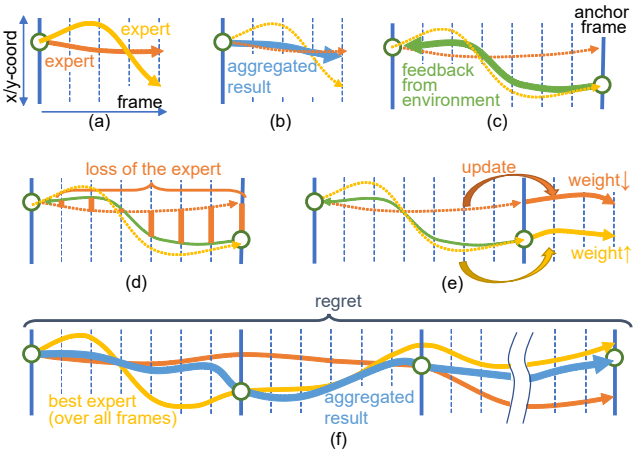

Figure 2. **The overall process of the proposed delayed-Hedge online tracking method.** (a) N arbitrary online trackers (i.e., experts) estimate the location of the target $f_1^t, \ldots, f_N^t$ at frame t. (b) Our online tracking result using adaptive expert aggregation with weights $w_1^t, \ldots, w_N^t$. (c) If the current frame t is an anchor frame, the true or a near-true target trajectory $y^{u+1}, \ldots, y^t$ is given as the feedback from the "environment" (see Sec. 2 for its definition), where u is the last anchor frame. (d) The loss ℓ that evaluates the performance of each expert. (e) The weight of each expert is updated according to the loss. (f) The best expert over all frames and the regret of our online tracking result are determined *in hindsight*.

aggregation scheme relies on the delayed-Hedge algorithm, which was originally developed for the problem called *adaptive expert aggregation*[1](AEA) [23]. In our case, each expert is an online tracker that estimates the location of the target at frame t based on its strategy, as shown in Fig. 2(a). At each frame, a weighted aggregation of the N online trackers is performed to obtain the final tracking result, as shown in Fig. 2(b).

The delayed-Hedge algorithm updates the weight of each expert adaptively, if the current frame t is an *anchor frame*. We assume that, at an anchor frame, the true or a near-true target trajectory from the last anchor frame is given as the feedback (from the environment), as shown in Fig. 2(c). Using the feedback, the loss of each expert is calculated to evaluate the performance of the expert between the current and the last anchor frames, as shown in Fig. 2(d). Then, the loss is fed to the experts to update their weight, as shown in Fig. 2(e). Note that the term "delayed" in the delayed-Hedge algorithm refers to the fact that the weight updating is kept waiting (i.e., delayed) until we have an anchor frame.

As noted above, the proposed method has a very important property that is its performance is *theoretically guaranteed even in an adversarial environment with arbitrary online trackers* by the adaptive expert aggregation theory. Roughly speaking, the worst-case performance of the proposed method is bounded in terms of *regret*. The regret is the difference between the result given by the proposed aggregation method and the best online tracker (i.e., the best expert) of the N trackers, as shown in Fig. 2(f). In other words, even the worst-case performance of the proposed method is still not far away from the best expert and therefore we can expect better performance by employing state-of-the-art online trackers as experts.

It may be surprising that the performance of the proposed method is still guaranteed in terms of regret, even though the best tracker and the regret are determined *only after* we get all T image frames. This fact proves that the proposed method makes an optimal decision at each frame even when the environment is totally unpredictable. Different from typical online tracking methods that often make various assumptions about their target and environment to keep their performance reasonable, the proposed method optimizes its result without any assumption and still guarantees the accuracy of the result, thanks to the theory behind AEA.

Our main contributions are summarized as follows.

- We propose a new online tracking method based on the delayed-Hedge algorithm. To the authors' best knowledge, this is the first trial to introduce this algorithm, which was developed in the field of theoretical machine learning, to a computer-vision problem.
- The proposed algorithm can aggregate any online trackers and therefore its performance will be improved continuously by incorporating future state-of-the-art trackers.
- Although it is not guaranteed that the proposed method always outperforms online trackers to be aggregated, it is theoretically guaranteed that its performance is always bounded by the regret term *arbitrary* image sequences. This theoretical bound is very meaningful for the practical purpose. In fact, our experiments on various image sequences show that the proposed method achieves near-best performance more frequently and stably than other trackers.

In this paper, we focus on the scenario of tracking a single target to evaluate the performance of the proposed method with a simple task; however, it is not difficult to extend our framework to deal with multiple-object tracking scenarios.

2. Related work

2.1. Adaptive expert aggregation (AEA)

Theoretical background In the theoretical machine learning research, the AEA problem is described as a re-

[1]In the field of theoretical machine learning, adaptive expert aggregation is called *online prediction with expert advice* (e.g., [23]). In this paper, we do not use this term because the term "online prediction" reminds us of real-time forecasting techniques. This technique has also been called *online learning* (e.g., [21]), but we also do not use this term because it can be confused with various learning techniques with iterative updating. Note that AEA is an active topic in the field of theoretical machine learning.

peated game between an algorithm and an adversarial environment as follows. At each round $t = 1, \ldots, T$, (i) N experts make predictions $f_1^t, \ldots, f_N^t$, (ii) the algorithm makes its prediction p^t (based on the experts' predictions), (iii) the environment returns a feedback y^t, and (iv) the algorithm incurs a loss $\ell(p^t, y^t) \in [0, 1]$. In our tracking application, f_i^t and p^t represent the location of the target object at frame t, and y^t represents the true (or a near-true) location.

The goal of the algorithm is to minimize the *"regret,"* which is defined as the performance difference between the algorithm and the best expert, that is:

$$R_T = \mathbb{E}\left[\sum_{t=1}^{T} \ell(p^t, y^t)\right] - \min_{i=1,\ldots,N} \sum_{t=1}^{T} \ell(f_i^t, y^t). \quad (1)$$

Here, the *best expert* is i^* that gives the minimum for the second term on the right-side in Eq. 1. If the performance of the algorithm is closer to the best expert, the algorithm achieves a smaller regret after T rounds.

Hedge algorithm and its applications *Hedge* [6] algorithm is designed for the AEA problem and achieves a small regret bounded by $O(\sqrt{M^* \log N})$, where M^* is the cumulative loss of the best expert after T rounds (i.e. $M^* = \min_{i=1,\ldots,N} \sum_{t=1}^{T} \ell(f_i^t, y^t)$. This strong theoretical result holds in adversarial environments; that is, it is not necessary to make any assumption about the experts. One well-known real-world application of the Hedge is an adaptive disk spin-down problem [10]. Its goal is to minimize the energy cost of disk spin-down by aggregating experts that suggest different spin-down timings.

Delayed-Hedge algorithm There is another AEA problem setting where the environment provides feedback with delay [13, 20] (we define the details of the delay in Sec. 4). There are several algorithms for various delayed settings (see, e.g., [20]), and we refer to the Hedge algorithm for our delayed setting as *delayed-Hedge*, which is the key technique used in this paper. The delayed-hedge algorithm and other AEA algorithms (including the hedge algorithm) have been examined only in theoretical research and applied no real-world task, except for a cloud computing task [18].

2.2. Online tracking and offline tracking

As noted in Sec. 1, visual-object-tracking methods can be classified into online tracking and offline tracking.

Online tracking Recent online tracking methods mostly employ some machine learning framework, such as Siamese networks. Bertinetto et al. [3] used a fully convolutional Siamese network for balancing accuracy and speed. Li et al. [15] introduced the regional proposal network (RPN) for Siamese network. Zhu et al. [30] proposed a Siamese network-based tracker that focuses on the semantic distractors for long-term tracking. Wang et al. [25] proposed an object tracking method with semi-supervised video object segmentation (VOS) based on Siamese network. Zhipeng et al. [29] introduced residual modules to improve the robustness and accuracy of Siamese network based tracker. In addition to them, Danelljan et al. [4] used the discriminative correlation filter (DCF) for tracking.

Ensemble online tracking The idea of ensembling (i.e., aggregating) online trackers has already been proposed with various strategies. Grabner et al. [7] and Avidan et al. [1] used AdaBoost to ensemble trackers. Babenko et al. [2] presented a novel online boosting algorithm based on multiple-instance learning framework. Such most ensemble methods take advantage of the correlations of experts to cancel out independent errors and increase the performance of the method. Thus, it is not straightforward for ensemble methods to incorporate arbitrary online trackers who do not correlate with each other. Wang et al. [24] recently provided an ensembling method which achieves the state-of-the-art performance. In contrast to the aforementioned ensembling trackers, their method treats the experts as black boxes, and use only bounding boxes of the output of the experts for the aggregation (i.e., it can be used for aggregating arbitrary trackers). They showed that their aggregating method succeeded empirically when employing some specific experts and use their key strategy for updating the experts. However, the empirical or theoretical performance when aggregating arbitrary experts has not been shown yet.

Offline tracking Most offline tracking problems are formulated as a path optimization problem and solved by a global optimization technique. Especially, recent methods are formulated as a network flow problem [5, 11, 28] and can track multiple objects at the same time. The proposed method is an online tracking method, and therefore this paper focuses on it; however, in the proposed method, offline tracking plays an important role to provide feedback, which is an accurate target trajectory between two anchor frames.

Online tracking with regret minimization One exceptional trial of using the term "regret" for tracking is Li et al. [16], where a regret minimizing algorithm is used for the target appearance adaption problem in a formulation of multiple-instance learning. Qi et al. [19] proposed Hedged Deep Tracking (HDT). They considered the ensembling method based on the Hedge algorithm which aggregates the experts defined by CNN features from different layers. In common with the aforementioned ensembling methods, their approaches can only employ some specific experts, but not arbitrary experts.

3. Delayed-Hedge tracking with arbitrary online trackers

3.1. Overview

As already shown in Fig. 2, the proposed delayed-Hedge tracking method assumes N *arbitrary* online trackers as multiple experts and determines the target location at frame

Algorithm 1 The delayed-hedge online tracking algorithm

Inputs:
N arbitrary online trackers.
Outputs:
Target trajectory $p^1, \ldots, p^t, \ldots$
Initialize:
$w_1^1 = \cdots = w_N^1 = 1/N$. $u = 0$.
$\eta \in \Re^+$. // Initialize η with an arbitrary value.
for $t = 1, \ldots$ **do**
  Estimate the current target locations $f_1^t, \ldots, f_N^t$ at the frame image by N online trackers.
  if frame t is determined as an anchor frame by having a reliable target location y^t **then**
    Update η using the doubling trick.
    Obtain the feedback $y^{u+1:t} = y^{u+1}, \ldots, y^t$ by the offline tracking result between y^u and y^t.
    Determine the current target location: $p^t = y^t$.
    Calculate the loss between anchor frames:

$$L_i^{u+1:t} = \sum_{k=u+1}^{t} \ell(f_i^k, y^k). \tag{2}$$

    Update each weight:

$$w_i^{t+1} = \frac{w_i^t \exp\left(-\eta L_i^{u+1:t}\right)}{\sum_{j=1}^{N} w_j^t \exp\left(-\eta L_j^{u+1:t}\right)}. \tag{3}$$

    $u = t$.
  else
    Determine the current target location:

$$p^t = \sum_{i=1}^{N} w_i^t f_i^t. \tag{4}$$

  end if
end for

t by AEA, that is, a weighted aggregation of the estimated target locations by the experts at t. The weight of each expert is updated at an anchor frame by evaluating its performance in the interval between the present and the last anchor frames. For the evaluation, the feedback (from the environment) and the loss play an important role. The feedback (Fig. 2(c)) is the true or a near-true target trajectory between two anchor frames and the loss (Fig. 2(d)) is the difference between the expert's estimation and the feedback. The best expert (Fig. 2(f)) is defined as the expert with the minimum accumulated loss over all the T frames. The proposed method aims to aggregate experts for minimizing the regret Eq. (1), that is, its difference from the best expert.

The feedback affects the loss evaluation and thus should be as close as possible to the true target trajectory. Since it is practically impossible to get the true target trajectory, we employ an offline tracker to get an accurate trajectory between two anchor frames as detailed in Sec. 3.3. Since the interval between the anchor frames is often not too large, the computational cost for the offline tracker is small and the proposed method still realizes real-time tracking (> 30 fps).

As emphasized above, *arbitrary* online trackers can be used as experts. Since it is guaranteed that the performance of the proposed method is similar to the best expert, the use of accurate experts is always appreciated. In other words, if a promising new online tracker is proposed, we can employ it as an expert to improve the performance of the aggregated online tracker in adversarial environments. Fortunately, as proved in Sec. 4, the proposed method allows us to use a large number of arbitrary experts without careful selection.

3.2. The detailed algorithm

Algorithm 1 shows the delayed-Hedge online tracking method. At each frame t, N online trackers give their estimation of the current target locations as $f_1^t, \ldots, f_N^t$. Then the current target location p^t is determined by their weighted aggregation of Eq. (4), where $\sum_{i=1}^{N} w_i^t = 1$, if the current frame is not an anchor frame.

When the current frame is an anchor frame, we receive the feedback from the environment. As shown in Fig. 2(b), the feedback is determined by an offline tracker as a near-true target trajectory. The offline tracker gives the trajectory from y^u to y^t, i.e., from the target location at the last anchor frame u to that at the current frame t.

Using the feedback $y^{u+1:t} = y^{u+1}, \ldots, y^t$, the loss $L_i^{u+1,t}$ over the interval $[u+1, t]$ is simply calculated using Eq. (2). The function $\ell(,) \in [0, 1]$ is an arbitrary function that evaluates the difference between two locations f_i^k and y^k, such as the Euclidean distance $\|f_i^k - y^k\|$ or IoU of object bounding boxes centered at f_i^k and y^k.

Finally, the weights $w_1^{t+1}, \ldots, w_N^{t+1}$ are updated by Eq. (3). This updating formula is simple but the key idea of the delayed-Hedge algorithm. As shown in Sec. 4, it achieves a strong theoretical regret bound with arbitrary online trackers even in any adversarial environments. Note that η in the formula is automatically updated via the *doubling trick*, which is described in Sec. 4, and its initial value is not very important. Therefore, Algorithm 1 is practically free from any hyper-parameter.

3.3. Anchor frames

Anchor frames are determined in an online manner; at each frame, we need to determine whether the frame can be an anchor frame or not without watching future frames. Considering that the feedback relies on the target location at each anchor frame, determination of the anchor frame is crucial in practice. In other words, we need to determine the frame t as an anchor frame only when the target object is detected with sufficiently high confidence. In Sec. 5.2, we show a practical approach for obtaining accurate feedback.

Note that it is possible to detect the target without any doubt, in special situations. For example, if an ID-tag is at-

tached to the target object and then detected by a tag sensor, the target will be detected when it lies in front of the sensor. Another example is a team sport player who wears a jersey with an ID number. If the ID number of the target player is detected at frame t (by scene text OCR), it can provide an anchor frame with the true location of the target.

4. Why does the delayed-Hedge online tracking achieve a small regret? — Theoretical guarantee of the proposed method

In this section, we show that the proposed method has strong theoretical support from regret analysis. Assume we have Q anchor frames as $u_1, \ldots, u_q, \ldots, u_Q$. As we saw in Algorithm 1, the feedback for the interval $[u_{q-1}+1, u_q]$ is given at the anchor frame u_q. It should be emphasized that this is a *delayed* feedback because we need to wait for the feedback to the frame $t \in [u_{q-1}+1, u_q - 1]$ until $t = u_q$.

We denote the total delay as $D = \sum_{q=1}^{Q} \sum_{t=1}^{u_q - u_{q-1}} t$, where $u_0 = 0$. For example, if the total number of frames $T = 100$ and the environment provides the feedback at anchor frames $\{u_1, \ldots, u_{10}\} = \{10, 20, 30, \ldots, 100\}$, then $D = (1 + \cdots + 10) \times 10 = 550$. If all frames are anchor frames ($Q = T$), there is no delay ($D = T$). In this case, the delayed-hedge algorithm is reduced to the standard AEA.

Even for AEA with delayed-feedback, it is known that we can achieve a small regret using the delayed-Hedge algorithm [20][2], which achieves the following regret bound:

Theorem 1 (Theorem A.5 of [20])**.** *Assume that a loss function ℓ takes values in $[0, 1]$. The regret of the delayed-Hedge algorithm with a learning rate $\eta \propto \sqrt{\ln N/(T+D)}$ after T frames is bounded as follows:*

$$R_T = O\left(\sqrt{(M^* + DM^*/T)\ln N}\right),$$

where M^* is the minimum cumulative loss of the N experts.

Note that M^* can be replaced by T because M^* is upper-bounded by T [20]. Theorem 1 states that the regret may become larger when D becomes larger. Especially, if there is no feedback (i.e., no anchor frame) until the final frame T, D achieves its maximum as $(T/2)(T+1) = O(T^2)$. In this worst case, the regret increases linearly with T and the performance of the proposed method may be different from the best expert.

In a realistic case, however, we can have the delayed-feedback at multiple anchor frames and thus have a much better bound than the worst-case. Specifically, if we have feedback with a constant probability r at each frame, we can derive the following regret bound that says the regret can be upper bounded by $O(\sqrt{T})$:

Theorem 2. *Assume that feedback from the environment comes with probability $r \in (0, 1]$ on each frame. The expectation of the regret of the delayed-Hedge is then bounded as follows:*

$$\mathrm{E}_r[R_T] = O\left(\sqrt{(M^* + M^*/r)\ln N}\right).$$

The proof of Theorem 2 is included in supplementary materials.

Moreover, Theorems 1 and 2 say that we may use a large number of experts without the risk of performance degradation because the regret bound only depends on $\ln N$. In other words, we do not need to select the set of experts carefully in advance.

The learning rate η cannot be estimated prior to the beginning of the tracking because $T + D$ is unknown in general. However, we can use a *doubling trick* technique [20] to set η adaptively. Specifically, if the current $t+D$ is larger than an initial parameter $T + D$ of η at frame t, we reset the parameter $\eta = \sqrt{\ln N/2(T+D)}$ from $t+1$, and therefore we can retain the aforementioned theoretical performance.

This upper bound of the performance difference between the algorithm and the best expert represents that the proposed method achieves similar accuracy to the best expert. Thus, this guarantee implies that the proposed method appropriately aggregates multiple trackers and yields robust performance.

5. Experimental results

We evaluate the proposed method by comparison with state-of-the-art online tracking methods on OTB2013 [26], OTB2015 [27], VOT2018 [14], and TColor128 [17]. We mainly focus on two standard performance measures, AUC score of success plots, and average distance precision (DP) on 20 pixels[3].

5.1. Online trackers as experts and competitors

We employ seven ($N = 7$) state-of-the-art fast online trackers as experts of the proposed method. They are: SiamMask [25], SiamFC_Res22 [29], SiamRPN [15], SiamFC [3], DaSiamRPN [30], MCCT-H [24], and ECO-H [4]. We also use the above trackers as competitors for performance comparison. Moreover, as an ablation study, we compare with the following baselines that aggregate arbitrary trackers, namely the above trackers. "Baseline

[2]This study reports that the online mirror descent algorithm achieves a good regret bound. The Hedge algorithm is known to be a special version of the online mirror descent algorithm (see, e.g., [8]), and therefore we can apply this regret bound to the Hedge algorithm. The detailed discussion can be found in the supplementary material.

[3]The performance on VOT2018 is often evaluated in supervised experiment where it is allowed for the failed tracker to restart from the correct position. However in our experiment, we evaluate it in unsupervised experiment as same as other datasets.

Tracker	OTB2013			OTB2015			VOT2018			TColor128			FPS
	AUC	DP	AR	AUC	DP	AR	AUC	DP	AR	AUC	DP	AR	
SiamMask	0.608	0.840	5.94	0.623	0.841	5.37	**0.478**	*0.682*	**3.77**	0.529	0.740	5.88	62
SiamFC_Res22	0.627	0.830	5.45	0.612	0.817	5.60	0.334	0.464	6.92	0.538	0.742	5.58	55
SiamRPN	0.641	0.855	5.37	0.630	0.837	5.49	*0.478*	**0.687**	4.10	0.533	0.736	5.82	86
SiamFC	0.585	0.775	6.45	0.575	0.762	6.39	0.348	0.515	6.65	0.509	0.690	6.09	126
DaSiamRPN	*0.660*	**0.885**	4.94	*0.649*	*0.861*	5.07	0.473	0.664	*3.82*	0.546	*0.753*	5.19	82
MCCT-H	0.597	0.781	6.08	0.596	0.797	6.00	0.365	0.509	6.38	0.535	0.712	5.65	15
ECO-H	0.639	0.841	*4.94*	0.631	0.836	5.09	0.350	0.485	6.27	*0.559*	0.750	*5.10*	17
Baseline(Average)	0.576	0.740	5.04	0.584	0.743	**4.76**	0.328	0.420	6.10	0.492	0.641	5.43	15
Baseline(MCCT)	0.576	0.744	6.20	0.563	0.750	6.45	0.299	0.438	6.80	0.487	0.658	6.19	13
Ours	**0.675**	*0.884*	**4.59**	**0.662**	**0.875**	*4.78*	0.469	0.663	4.20	**0.613**	**0.826**	**4.06**	12

Table 1. **Performance comparison with the state-of-the-arts on several benchmark datasets.** The best tracker is indicated with **red bold** and the second is indicated with *blue italic*. AUC: average area-under-curve score, DP: average distance precision, AR: average rank based on AUC, and FPS: average speed. Bigger AUC and DP, and smaller AR mean better performance

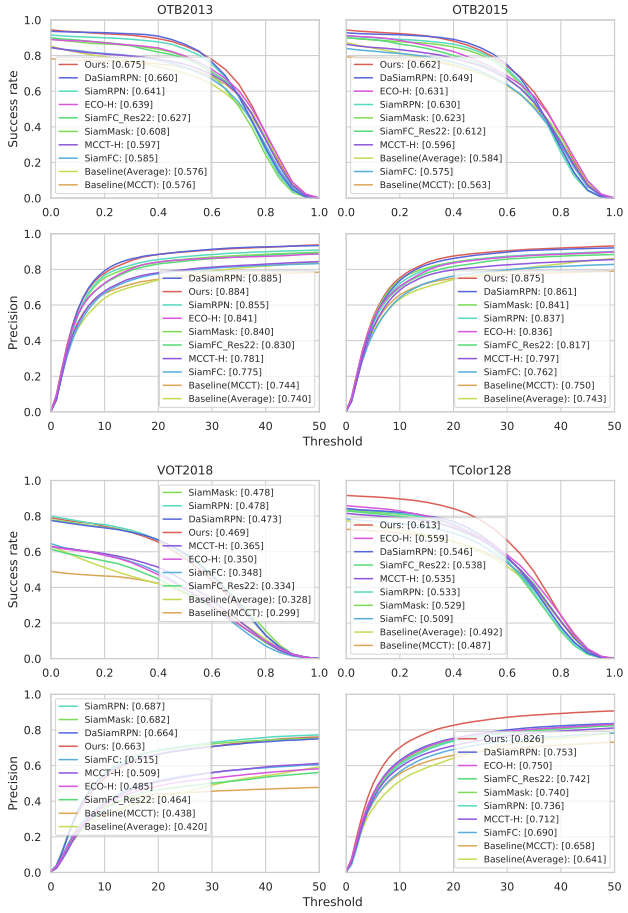

Figure 3. **Success and precision plots on OTB2013, OTB2015, VOT2018, and TColro128.** The legends of the plots mean area-under-curve scores and average distance precisions, respectively.

(Average)" is a simple aggregation method which averages the prediction of the experts at every frame . "Baseline (MCCT)" aggregates the above trackers by evaluating the experts at every frame using the key technique of [24] (see details in Sec. 3.3 of the paper). Note that what we want to

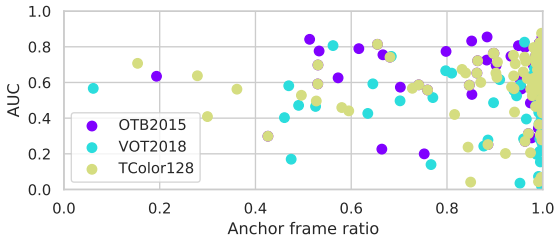

Figure 4. Correlation between AUC and anchor frame ratio of the proposed method for image sequences in OTB2015, VOT2018 and TColor128.

demonstrate in the experiments is the tracking performances "in the wild". Therefore, for all methods (and experts) we did not use any tuned hyper-parameter specialized in particular datasets. For all datasets, we employed the default parameters recommended by the authors, and thus the results we provide may be slightly different from the results in the original papers.

5.2. Anchor frames and feedback

As mentioned in Section 3.3, we should properly determine anchor frames online (i.e., at each frame) for getting the delayed-feedback. Since the object location at the anchor frame should be accurate as possible, we use the following careful steps at each frame t: [Step 1] We crop N bounding boxes given as the tracking results of the N experts. [Step 2] We convert each bounding box into a 512-dimensional feature vector by using the bottleneck of ResNet-18 [9] trained with ImageNet. [Step 3] We calculate N cosine similarities between the feature vectors and the feature vector of the template image (i.e., the object image in the first frame). [Step 4] If the maximum among N cosine similarities is larger than a threshold θ, the current frame t is determined as an anchor frame. We fixed $\theta = 0.74$ in the experiments. The bounding box that gives the maximum becomes the reliable target location, y_t (in Algorithm 1).

The delayed-feedback, $y^{u+1}, \ldots, y^t$, is given by

the state-of-the-art offline-tracker [28], which gives the globally-optimal tracking result between the anchor frames based on network flow. For Eq. (2), we use the following IoU-based loss function between the target bounding box B_f by an expert and B_y by the feedback $y \in \{y^{u+1}, \ldots, y^t\} : \ell(f, y) = 1 - (B_f \cap B_y)/(B_f \cup B_y)$.

Fig. 4 shows the relationship between the AUC by the proposed method and the anchor frame ratio for (Q/T) all image sequences. This figure reveals that the anchor frame ratio is more than 0.8 for many image sequences. This means that at least one expert is very confident of its tracking result at each frame and our tracker could utilize it. Note that even if two consecutive frames $t-1$ and t become anchor frames, our algorithm still can update the weights of the experts by utilizing y^{t-1} and y^t as feedbacks.

Another, more important fact shown in Fig. 4 is that there is neither positive nor negative correlation between the AUC and the anchor frame ratio. This proves that our delayed-hedge algorithm works very reasonably. If there is any positive correlation, we can improve the performance just by increasing the anchor frame ratio. However, our algorithm did not do it because we cannot get reliable anchor frames frequently in some image sequences and the algorithm automatically decides to wait for a reliable anchor frame. In other words, the algorithm optimizes the anchor frame ratio according to each image sequence and could get the best compromise on its performance.

5.3. Comparison with state-of-the-arts

Fig. 3 represents the success and precision plots of the proposed method and state-of-the-art trackers, and Table 1 shows the results of our tracking performance of them. In addition to AUC and DP, we show the average ranking (AR) of AUC for each tracker. The ranking result is a very important measure for the proposed method because our theoretical motivation is to achieve similar accuracy to the best expert on each image sequence. We show the performance and the frequency of rank of the trackers in Fig. 5.

Table 1 also shows the speed (FPS) of the individual methods. Since the speed of the proposed method depends on the speed of the slowest expert (i.e., MCCT-H in the experiments), our current FPS is nearly real-time (12 FPS). Of course, if we employ only real-time tracking experts, the proposed method runs in real-time. More importantly, we have to emphasize that the overhead for determining anchor frames, calculating delayed-feedback by the offline tracker, and aggregating experts, is very small.

Evaluation on OTB2013 and OTB2015 In addition to Table 1, Fig. 3 also shows that the proposed method outperforms the state-of-the-art trackers. In Fig. 5, we can see that various experts equally contributes to the proposed method. That is, this result proves that the best expert drastically changes over the image sequences in OTB and the proposed

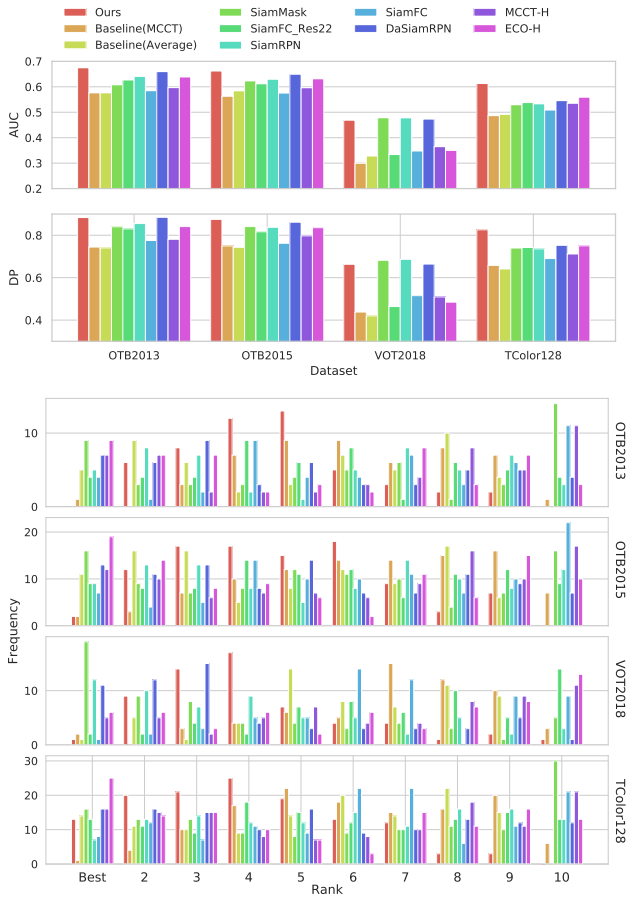

Figure 5. **Performance histogram and ranking histogram based on AUC in OTB2015, VOT2018, and TColor128.** A smaller rank means better performance.

method effectively performs even in such an adversarial situation for the experts.

Evaluation on VOT2018 In VOT2018, the AUC of the proposed method cannot outperform the top-three trackers, SiamMask, SiamRPN, and DaSiamRPN. We need to emphasize that this is not disappointing because the theory behind the proposed method guarantees that the difference from the best expert (i.e., regret) is *bounded*; that is, the proposed method can achieve similar performance to the best tracker. Since these three trackers competitively perform well over VOT2018, the proposed method can achieve similar accuracy to them. More importantly, the proposed method is not degraded by the others. In fact, as shown in Fig. 5, the experts are divided to accurate and poor trackers over VOT2018. Consequently, we can say that the proposed method is really effective for the adversarial environment where the best expert drastically changes for each image sequence and less effective for the less adversarial environment where we have constantly-overwhelming experts.

Evaluation on TColor128 Table 1 and Fig. 3 shows that the proposed method significantly outperforms state-of-the-

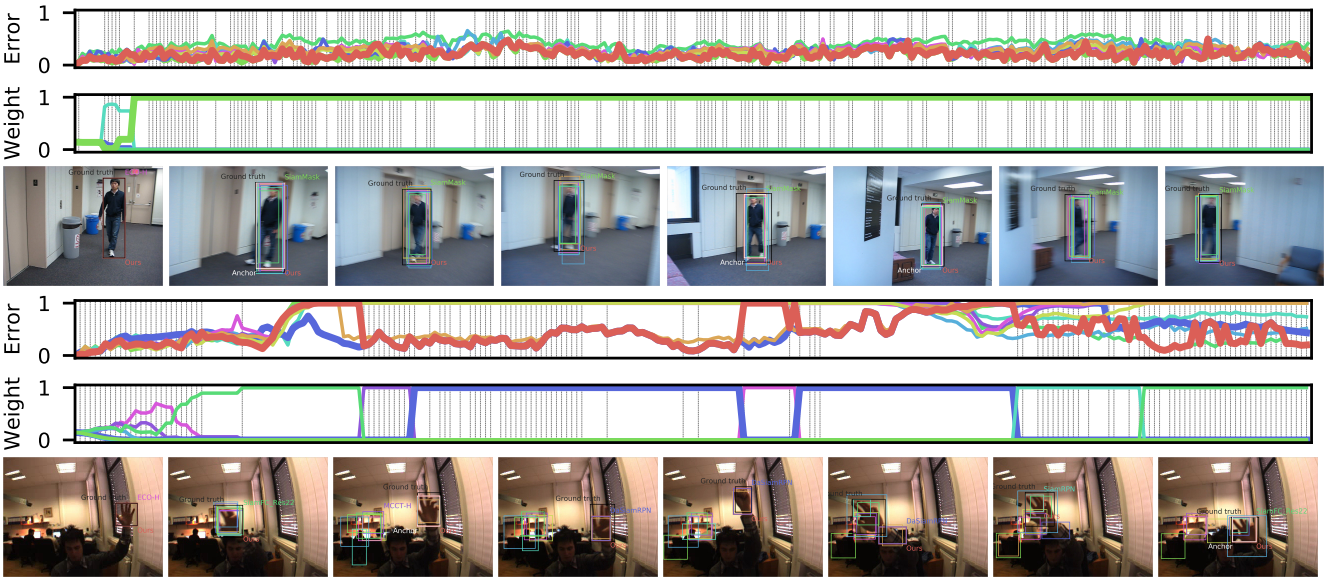

Figure 6. **Tracking examples.** The top is "BlurBody" in OTB2015 and the bottom is "Hand" in Tcolor2018. For the picked-up frame, we annotate ground truth (black), the proposed method (red), and only for anchor frames, we annotate the detected bounding box (white). The expert which has the highest weight is also annotated. In the error and weight graph, gray vertical lines show anchor frames.

art trackers. As also shown in Fig. 5, the proposed method averagely achieves high performance. The reason is the same on OTB2013 and OTB2015; that is, the best experts drastically changes depending on the tracking tasks. TColor128 may contain various kinds of tracking tasks and that makes it difficult to have an expert overwhelming over all the image sequences in the dataset; in contrast, the proposed method adaptively follows the best expert for individual image sequences or even individual frames.

In the ranking histogram of Fig. 5, our method shows significant robust performance on all four datasets. Remarkably, the proposed method achieved higher ranks for many datasets. Moreover, while the other methods shared lower-ranked performances as we introduced in Sec. 1, only our proposed method did not. We can say that this robustness is the result of our regret-minimization approach with theoretical support.

5.4. Tracking examples

In this section, several tracking examples are shown to observe how the proposed method aggregates the experts to achieve similar performance to the best expert. Fig. 6 shows two examples of the tracking results with temporal transitions of the overlap error of the proposed method and experts, and the weights for the experts in Algorithm 1. In "BlurBody", SiamMask (the blue one) is the best expert, and we can see that the proposed method gives the highest weight for it from early frames of the image sequence. Moreover, the proposed method can keep the high weight even after the weight updating at several anchor frames.

On the other hand, in "Hand", we can see that the best-performing expert drastically changes in this image sequence by observing the overlap error transition. More precisely, SiamFC_Res22 achieves the best in the beginning part but it degrades the performance in the middle part. DaSiamRPN achieves the best in the second part, but in the final part SiamFC_Res22 achieves the best again. From the perspective of our regret theory, it seems that the proposed method cannot follow such changes but only follow the "overall" best expert. However, in fact, it is theoretically known that the Hedge algorithm follows such a "partial best expert" adaptively for minimizing not only the standard regret R_T (of Eq. (1) but also *shifting regret*, which is another regret measure considering the switching (see, e.g., [12]). Thus, the proposed method can utilize the partial best experts adaptively even against such fluctuations within a image sequence.

6. Conclusion

In this paper, we propose an online tracking method based on the delayed-Hedge algorithm, which allows us to aggregate arbitrary multiple online trackers. Its robustness in the tracking performance is guaranteed theoretically in term of "regret." The experimental study on various tracking tasks shows that the proposed method could achieve the state-of-the-art performance by aggregating various online trackers especially for adversarial environments.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number JP17H06100 and JP18K18001.

References

[1] S. Avidan. Ensemble tracking. In *Proc. CVPR*, 2015.

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

[3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr. Fully-convolutional siamese networks for object tracking. In *Proc. ECCV Workshops*, 2016.

[4] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg. ECO: Efficient convolution operators for tracking. In *Proc. CVPR*, 2017.

[5] A. Dehghan, S. M. Assari, and M. Shah. GMMCP tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In *Proc. CVPR*, 2015.

[6] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *JCSS*, 55(1):119–139, 1997.

[7] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *Proc. BMVC*, 2006.

[8] E. Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016.

[10] D. P. Helmbold, D. D. Long, T. L. Sconyers, and B. Sherrod. Adaptive disk spin-down for mobile computers. *MONET*, 5(4):285–297, 2000.

[11] J. F. Henriques, R. Caseiro, and J. Batista. Globally optimal solution to multi-object tracking with merged measurements. In *Proc. ICCV*, 2011.

[12] M. Herbster and M. K. Warmuth. Tracking the best linear predictor. *JMLR*, 1:281–309, 2001.

[13] P. Joulani, A. Gyorgy, and C. Szepesvári. Online learning under delayed feedback. In *Proc. ICML*, 2013.

[14] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Cehovin. A novel performance evaluation methodology for single-target trackers. *TPAMI*, 38(11):2137–2155, 2016.

[15] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu. High performance visual tracking with siamese region proposal network. In *Proc. CVPR*, 2018.

[16] M. Li, J. T. Kwok, and B.-L. Lu. Online multiple instance learning with no regret. In *Proc. CVPR*, 2010.

[17] P. Liang, E. Blasch, and H. Ling. Encoding color information for visual tracking: Algorithms and benchmark. *TIP*, 24(12):5630–5644, 2015.

[18] I. Menache, O. Shamir, and N. Jain. On-demand, spot, or both: Dynamic resource allocation for executing batch jobs in the cloud. In *Proc. ICAC*, 2014.

[19] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang. Hedged deep tracking. In *Proc. CVPR*, 2016.

[20] K. Quanrud and D. Khashabi. Online learning with adversarial delays. In *Proc. NIPS*, 2015.

[21] S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2011.

[22] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *TPAMI*, 36(7):1442–1468, 2014.

[23] V. G. Vovk. A game of prediction with expert advice. In *Proc. COLT*, 1995.

[24] N. Wang, W. Zhou, Q. Tian, R. Hong, M. Wang, and H. Li. Multi-cue correlation filters for robust visual tracking. In *Proc. CVPR*, 2018.

[25] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. Torr. Fast online object tracking and segmentation: A unifying approach. In *Proc. CVPR*, 2019.

[26] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *Proc. CVPR*, 2013.

[27] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *TPAMI*, 37(9):1834–1848, 2015.

[28] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *Proc. CVPR*, 2008.

[29] Z. Zhipeng, P. Houwen, and W. Qiang. Deeper and wider siamese networks for real-time visual tracking. In *Proc. CVPR*, 2019.

[30] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu. Distractor-aware siamese networks for visual object tracking. In *Proc. ECCV*, 2018.