Exploring Motion Information for Distractor Suppression in Visual Tracking

Kaiwen Liu[1,2] Jin Gao[1,2,†] Haowei Liu[1,2] Liang Li[4] Bing Li[1,2]
Weiming Hu[1,2,3]
[1] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences
[2] School of AI, University of Chinese Academy of Sciences
[3] CAS Center for Excellence in Brain Science and Intelligence Technology
[4] Beijing Institute of Basic Medical Sciences
{liukaiwen2019, liuhaowei2019}@ia.ac.cn
{jin.gao, bli, wmhu}@nlpr.ia.ac.cn, liang.li.brain@aliyun.com

Abstract

In the past few years, Siamese networks have achieved outstanding improvements in visual object tracking. However, visual distractors with similar semantics can be easily misclassified as the target by Siamese networks and may consequently result in the drift problem. Besides, the Hanning window penalty, which is generally used to suppress distractors, could fail in many challengeable scenes. Notably, most failures violate the assumption of motion continuity. Thus, in this work, we explore motion information to mitigate the drift problem in visual tracking. First, we introduce a simple linear Kalman filter to predict the bounding box of the target in the current frame, which acts as a reference for decisions. Second, an IoU-Guided penalty is assembled in the post-processing to suppress distractors effectively. It's worth mentioning that our method is almost cost-free. We conduct numerous experimental validations and analyses of our approach on several challenging sequences and datasets. Our tracker runs at approximately 40 fps and performs well on those sequences which include the Background Clutter attribute. Finally, by simultaneously integrating the IoU-Guided penalty and the Hanning window penalty with a strong baseline tracker TransT [7], our method achieves favorable gains by 69.1→71.5, 65.7→66.7, 64.9→65.9 success on OTB-100 [32], LaSOT [12], NFS [18].

1. Introduction

Generic visual object tracking is a significant and fundamental task in computer vision. Given the initial state, this task involves estimating or predicting the state of the target object. Most prior researches concentrate on designing a robust appearance model to differentiate the target and the background. The two currently dominating paradigms are Siamese networks [19,28,41] and discriminative appearance models [4, 10]. The former aims at learning to construct a feature embedding space that is optimal for template matching. The latter focuses on learning robust discriminative features usually equipped with online learning. Although these researches achieve remarkable improvements on numerous benchmarks, the robustness of the tracking algorithm is still far from perfect, especially on the challengeable attributes and the occasions of the open world. For example, the drift problem caused by visual distractors with

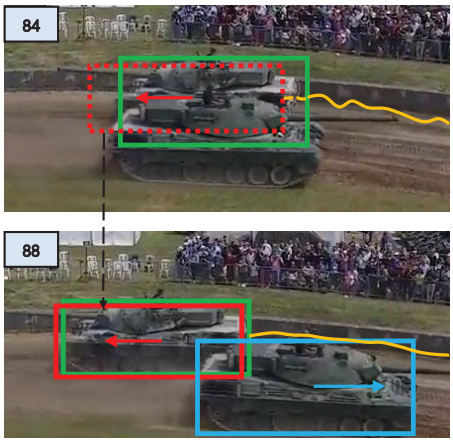

Figure 1. Drift problem due to ignoring the motion information. The green bounding boxes imply the target. The orange curves imply the trajectories of the target. The red bounding boxes imply the prediction generated based on the motion information. The blue bounding box implies the wrong decision that violates the assumption of motion continuity. The arrows represent the velocities of different objects.

[†] Corresponding author.

similar semantics is a tough nut to crack.

Drifting is a common phenomenon in visual tracking. It seriously weakens the robustness of the tracking algorithms and causes continual failures of subsequent frames. Generally, it may be related to many difficult attributes, such as illumination variation, background clutter and similar objects, etc. However, it is worth noting that most drifts essentially violate the assumption of motion continuity. As demonstrated in Fig 1, the predicted bounding box marked as blue drifts to the distractor at the 88th frame due to ignoring the motion information. If we exploit the motion information to predict a reference box like the red bounding box in Fig 1, we can select the right target marked as green and solve the drift problem. Nevertheless, the current Siamese tracking algorithms have ignored this significant prior knowledge but adopted a Hanning window penalty as an implicit spatial constraint.

The Hanning window penalty was first proposed in SiamFC [2]. It has been generally integrated by most of the Siamese networks. However, this penalty has the following disadvantages.

1. **The Hanning window penalty passively trusts the previous prediction but ignores the motion information**. The maximum of the Hanning window is fixed at the center of the previously predicted bounding box. When the previous decision actually drifts to distractors, this penalty is likely to suppress the real target in the current frame. It would prevent the system from correcting the output.We name this phenomenon *winner-take-all*.

2. **The Hanning window penalty only has effects on constraining the center positions of the regressed bounding boxes rather than the overlaps (generally evaluated by IoUs) between the regressed bounding boxes and the ground truth box**. The value of the Hanning function is determined by the 2D coordinate (x, y). Thus, it cannot guarantee the motion continuity.

Therefore, we infer that this penalty acts as an implicit penalty which may not be optimal and may aggravate the drift problem. Hence, it is natural to raise *two corresponding questions*:

1. How can we actively exploit motion information to generate a new reference box rather than use the previously predicted bounding box?

2. How can we use the reference box to construct an explicit IoU constraint rather than the implicit center position constraint?

In this work, we explicitly exploit the motion information to construct a direct penalty and mitigate the drift problem. *Firstly*, in order to use the motion information to predict a reference bounding box in the current frame, we explore two common methods (the *Kalman filter* and the *optical flow*). On the one hand, we follow the classical algorithms SORT [3] and DeepSORT [31] in multi-object tracking to design our Kalman filter. Notably, We just adopt a simple linear Kalman filter to predict a reference bounding box of the target in the current frame. On the other hand, we follow the previous research MedianFlow [16] based on the optical flow for prediction. We compare these two methods and find that the simple linear Kalman filter performs more effectively than the complicated MedianFlow [16]. *Secondly*, in order to construct an explicit spatial constraint, we use the prediction of the motion model (the Kalman filter) as a reference box. We calculate the intersection over union (IoU) between the reference box and all of the regressed bounding boxes exported by the Siamese networks. We name this novel penalty *IoU-Guided Penalty*. Then, we combine the classification scores with the calculated ious and select the bounding box ranked first as the final prediction. *Finally*, numerous qualitative and quantitative experiments are organized to verify the effectiveness and universality of our method. Our method effectively mitigates the drift problem and performs well on several benchmarks.

The main contributions of this work are threefold.

- We analyze the causes of the drift problem in visual tracking and the disadvantages of the current Hanning window penalty.

- We explore and compare two common methods for motion prediction and select a cost-free scheme to predict a reference box to approximate the ground truth box in the current frame.

- We propose a novel IoU-Guided penalty and explore a suitable way to combine the classification scores with the calculated ious for the final decision. By integrating the IoU-Guided penalty into the baseline tracker TransT [7], it achieves remarkable performance gains with neglectable time cost running at 40 fps and successfully mitigates the drift problem.

2. Related Work

In this section, we review the related work about the drift problem and distractors in visual object tracking, as well as introduce the ways to exploit motion information of traditional methods before deep learning era.

2.1. The Drift Problem and Distractors

In visual tracking, appearance models are the most significant components of the whole pipeline. Most prior researches have focused on the design of robust appearance models and appearance features to locate the target object in each frame. However, it may be impossible to only use

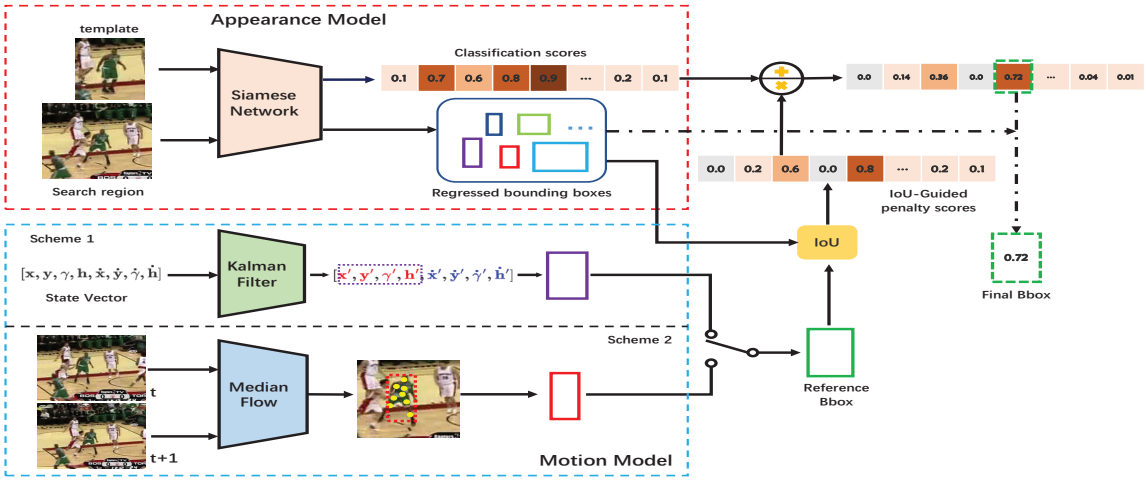

Figure 2. Overview of the proposed tracking framework, consisting of an appearance model (top) and a motion model (bottom). The appearance model may be anchor-based models (SiamRPN++ [19]) or Transformer- based models (TransT [7]). The motion model can be a linear Kalman filter or MedianFlow [16].

appearance features for decisions, especially when there are other similar objects in the scene. The two currently dominating paradigms are Siamese networks and discriminative appearance models. While Siamese networks [2, 19, 20] exploit a Hanning window penalty to suppress distractors, discriminative appearance models [4, 10, 14] aim to mitigate drifting by integrating background information when learning the target classifier online. Although these methods enhance the robustness of the output bounding box, the capacity for distinguishing different objects is limited. Thus, a few approaches have been proposed to deal with distractors. Xiao *et al.* [33] describe the positions of distractors and design a few hand-crafted rules to differentiate similar objects. DaSiamRPN [41] tackles distractors by subtracting corresponding image features from the target template during online tracking. SiamRCNN [28] refers to the classical data association ideology of multi-object tracking using dynamic programming and a hand-crafted association score to form short tracklets. It also resorts to hard example mining to improve its robustness. KYS [5] uses a RNN to transfer information about the scene across frames. It maintains learnable state vectors to record the context. KeepTrack [22] introduces a learnable graph embedding network that explicitly associates target candidates from frame to frame. In contrast, we describe motion information to predict a reference bounding box for decisions.

2.2. Motion Information in Tracking

In visual tracking before deep learning era, motion models [36] act as important prior constraints and enhance the robustness of the output in many constrained scenes. Besides, estimating the trajectory and the state of the target object is another vital process. In *Point Tracking*, the deterministic methods [27] use qualitative motion heuristics to constrain the correspondence problem while the statistical correspondence methods [6] use the state space approach to model the object properties such as position, velocity, and acceleration. In *Kernel Tracking* [9, 24, 26], the object motion is generally in the form of parametric motion (translation, conformal, affine, etc.) or the dense flow field computed in subsequent frames. In *Silhouette Tracking* [1, 15], the object state is defined in terms of the shape and the motion parameters of the contour. Moreover, the Kalman filter has been extensively used in the vision community for tracking. SORT [3] and DeepSORT [31] describe a linear Kalman filter to predict the bounding box of the target object and use it as a reference to finish object association. In addition, the optical flow is another regular method to build the correspondence of pixels across frames. In this work, we use these two regular methods to achieve our goal, predicting the motion of the target object.

By the way, there are two similar works [30, 40] that attempt to integrate the Kalman filter with Siamese networks. However, these works concentrate on croping a more appropriate searching window rather than selecting the best sample when using motion information. Meanwhile, our method performs better with strong universality.

3. Methodology

Here, we describe our tracking approach, which actively exploits the motion information to predict a reference box and uses it to construct a novel IoU-Guided penalty. *Firstly*, we introduce the pipeline of our tracking method. *Secondly*, in order to predict a reference box, we explore two common motion methods (the *Kalman filter* and *MedianFlow* [16]

based on the optical flow). We compare these two methods and summarize the differences between them. Finally, we analyze the disadvantages of the current Hanning window penalty and propose a novel IoU-Guided penalty to better exploit the predicted reference box and construct an explicit spatial constraint.

3.1. Overview

Firstly, we give an overview of our tracking pipeline, which is shown in Fig 2. The submodule in the red dashed box is the same as typical Siamese Tracking algorithms. We employ a Siamese network to extract appearance features and export the classification scores map and corresponding regression boxes. The Siamese network may not be restricted to a specific architecture. We name this characteristic *model-free*. Besides, The submodule in the blue dashed box is the core motion module which can be set as one of the above modules, the Kalman filter or MedianFlow [16]. The output of the motion module is a reference bounding box. Then, we use it to calculate IoUs with the regression bounding boxes exported by the appearance model. We combine the IoUs with the classification scores and select the maximum value together with its corresponding regression bounding box as the final prediction. In the end, we use the final output bounding box to update the state of the target which is a vital step for the motion model. **Notably, our method works with no need for training.** Both the motion model and the IoU-Guided penalty act as an effective post-processing to suppress distractors.

3.2. Motion Model

Secondly, in order to answer the first question mentioned in the Introduction, we introduce two common motion methods, the *Kalman filter* and *MedianFlow* [16].

The Kalman filter is an efficient recursive filter, which can estimate the state of a dynamic system with noise. We follow SORT [3] and DeepSORT [31], assume that the state noise has a Gaussian distribution and describe a linear Kalman filter to predict the next state. The state of the target is modelled as eight dimensional vector $[x, y, \gamma, h, \dot{x}, \dot{y}, \dot{\gamma}, \dot{h}]$ that contains the bounding box center position $[x, y]$, aspect ratio γ, height h, and their respective velocities in image coordinates. The typical Kalman filter contain two steps, *Predict* and *Update*.

Predict:

1. Extrapolate the state.
   $\mathbf{\hat{x}_{n+1,n}} = \mathbf{F}\mathbf{\hat{x}_{n,n}}$

2. Extrapolate the uncertainty.
   $\mathbf{P_{n+1,n}} = \mathbf{F}\mathbf{P_{n,n}}\mathbf{F^T} + \mathbf{Q}$

$\mathbf{\hat{x}_{n+1,n}}$ represents the predicted state vector about the reference bounding box. $\mathbf{F}$ represents the state transition matrix. $\mathbf{Q}$ represents the process noise uncertainty. $\mathbf{P}$ implies the uncertainty of the state.

Update:

1. Compute the Kalman Gain.
   $\mathbf{K_n} = \mathbf{P_{n,n-1}}\mathbf{H^T}(\mathbf{H}\mathbf{P_{n,n-1}}\mathbf{H^T} + \mathbf{R_n})^{-1}$

2. Update the measurement.
   $\mathbf{\hat{x}_{n,n}} = \mathbf{\hat{x}_{n,n-1}} + \mathbf{K_n}(\mathbf{z_n} - \mathbf{H}\mathbf{\hat{x}_{n,n-1}})$

3. Update the uncertainty.
   $\mathbf{P_{n,n}} = (\mathbf{I} - \mathbf{K_n}\mathbf{H})\mathbf{P_{n,n-1}}(\mathbf{I} - \mathbf{K_n}\mathbf{H})^{\mathbf{T}}$
   $+ \mathbf{K_n}\mathbf{R_n}\mathbf{K_n^T}$

$\mathbf{z_n}$ represents the final bounding box which is determined by our method. $\mathbf{K_n}$ represents the Kalman Gain. $\mathbf{H}$ implies the observation matrix. $\mathbf{R}$ implies the measurement uncertainty. These above-mentioned two steps are running iteratively. Although assuming simple linear property, the filter has acceptable prediction accuracy and can stabilize the prediction. Notably, the Kalman filter predicts the reference box as a whole. By contrast, MedianFlow [16] predicts the reference box based on the pixels rather than a whole box.

MedianFlow [16] is a classical tracker based on optical flow which has been incorporated in the OpenCV tracking library. The well-known TLD [17] tracking algorithm employs MedianFlow as its base tracker. Given a pair of images I_t, I_{t+1}, and a bounding box b_t, this tracker outputs the predicted bounding box in frame $t + 1$. Concretely, a set of points is initialized on a rectangular grid within the bounding box b_t. These points are marked as reference points and tracked using the Lucas-Kanade optical flow method which generates the new predictions. Then, we calculate FB errors and NCC errors used in MedianFlow [16] as a quality indicator and filter out 50% of the worst predictions. The remaining predictions are used to estimate the displacement of the bounding box. Estimation of the bounding box displacement from these predictions is performed naively using median over each spatial dimension.

In summary, the Kalman filter can model the motion pattern based on the motion history and estimate the state of the target as a whole even if with noise. By contrast, MedianFlow [16] can predict the displacement based on the motion of the pixels. It only requires two adjacent frames, an initial box and reference points as inputs but may fail due to some noise. Both of these two methods can exploit the motion information to generate a reference box.

3.3. IoU-Guided Penalty

After generating a reference bounding box, there remains the second question mentioned in the section 1. It is worth noting that the intersection over union (IoU) acts as a significant indicator which has been widely ignored during inference. When training Siamese networks, the regres-

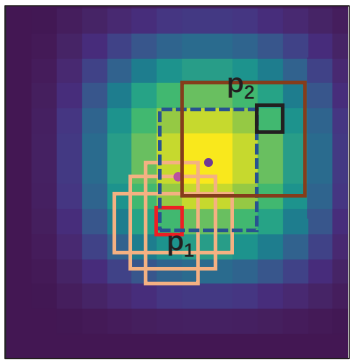

Figure 3. Hanning window penalty located at the purple center of the reference box. The magenta point implies the center of the search region. The dark blue box is the reference box. The three orange boxes centered at the red square are corresponding to three aspect ratios in anchor-based tracking algorithms. The red and black squares represent two different positions that possess identical Hanning window penalty scores but different IoUs with the reference box.

sion losses are always formulated using the IoU between the predicted bounding box and the ground-truth bounding box. However, there is only a fixed Hanning window penalty to suppress distractors during inference. The formula of the Hanning function is demonstrated below:

$$h(u) = \left[0.5 - 0.5cos\left(\frac{2\pi u}{M-1}\right)\right]$$
$$w(x,y) = h(x) \cdot h(y)$$
$$\forall 0 \leq x, y \leq M-1$$

M represents the output size of the final feature map. (x, y) represents the coordinate in this feature map. Apparently, this function only has effects on constraining the center positions of the regressed bounding boxes rather than the IoU with the reference box. In order to demonstrate the drawbacks of the Hanning window penalty, we visualize it in Fig 3. The dark blue dashed box refers to the reference bounding box. In anchor- based Siamese methods, the three orange dashed boxes are corresponding to three different aspect ratios centered at $\mathbf{p_1}$. Apparently, these three bounding boxes possess different IoUs with the reference box but identical Hanning penalty scores. Besides, the red square $\mathbf{p_1}$ and the black square $\mathbf{p_2}$ represent two positions which have an identical penalty score. In TransT [7], the regressed bounding boxes centered at $\mathbf{p_1}$ and $\mathbf{p_2}$ probably possess different IoUs with the reference box.

Based on the above observation, we conclude that the Hanning window penalty is an indirect way to build the spatial constraint. It cannot evaluate and distinguish the regressed bounding boxes with different IoUs but identical penalty scores. Therefore, it is natural to use IoUs to construct the spatial constraint. Concretely, we calculate IoUs between the reference box and all of the regressed boxes exported by Siamese networks. Besides, we use the results as a new penalty score and name our novel spatial constraint *IoU-Guided Penalty*. In *Experiments*, we verify the effectiveness of our new penalty for solving the drift problem and suppressing distractors.

4. Experiments

4.1. Implementation Details

We adopt the anchor-based tracker SiamRPN++ [19] or the Transformer-based tracker TransT [7] as the appearance model due to their outstanding performances. The appearance model is fixed and only the Hanning window penalty is changed. We copy the parameters of the Kalman filter from DeepSORT [31]. For MedianFlow [16], we set the window size as 3, the pyramid levels as 5. Our method works with no need for training.

4.2. Schemes for Generation and Combination

Firstly, in order to generate the reference bounding box mentioned above, we adopt and compare four different methods named *Prev*, *KF*, *FlowKF*, *FlowPrev*. *Prev* means using the last previous final output box as the current reference box. *KF* implies using a linear Kalman filter to predict the reference box. *FlowKF* means using MedianFlow [16] for prediction and switching to *KF* when failed. *FlowPrev* implies using MedianFlow [16] for prediction and switching to *Prev* when failed. Meanwhile, in order to combine the calculated ious with the classification scores, we explore three different ways named *Add*, *Multiply*, *NonLinear* respectively.

- Add: $(1-w) \times s + w \times ious$
- Multiply: $s \times ious$
- NonLinear: $ious + s \times (2-s)$

s represents the classification score and $ious$ represents the calculated ious. w represents the weight of the calculated ious and is set to 0.5. The *NonLinear* function changes more smoothly than the *Multiply* style which can reserve more proposals with high ious.

Then, we use TransT as the base model and compare different combinations between the ways to generate the reference box and the ways to combine the calculated ious with the classification scores. The results are shown in Table 1. On the one hand, *KF* surprisingly achieves the best performance even if we only adopt a simple linear Kalman filter. By contrast, *FlowKF* and *FlowPrev* perform unsatisfactorily even if with complicated designs. We infer that the inferior performance of MedianFlow [16] results from the drastic motion of the camera or the target and it would

Table 1. Exploring the best scheme. The different columns represent different ways to generate the reference box. The different rows represent different ways to combine the classification scores with the IoU-Guided scores. The values in this table represents the AUC scores on OTB-100 [32].

Method	Prev	KF	FlowKF	FlowPrev
Add	0.694	**0.709**	0.690	0.693
Multiply	0.694	0.691	0.680	0.685
Nonlinear	0.678	0.681	0.688	0.689

cause a lot of failures. Moreover, MedianFlow [16] essentially cannot model the long-term motion pattern. On the other hand, the simplest scheme *Add* surpasses others apparently. We infer that weighted summation is suitable to build the spatial constraint but multiplication and nonlinear operation weaken the importance of the classification scores.

In summary, we select the *KF* to predict the reference box and the weighted summation *Add* as the final scheme.

4.3. Effectiveness and Universality

Secondly, we compare four penalty methods, *Np*, *Hanning*, *HanningWM*, and our novel *IoU-Guided* penalty respectively. *Np* means that the original Hanning window penalty is removed. It acts as a baseline about penalty methods. *HanningWM* is another penalty that we fix the Hanning window at the center position predicted by the Kalman filter. For our *IoU-Guided penalty*, we select the best scheme determined in the above section. Then, we compare these four methods on LaSOT [12] and OTB-100 [32] using SiamRPN++ [19] and TransT [7] respectively. As shown in Table 2, SiamRPN++ [19] with IoU-Guided penalty increases 8.8% relatively compared with no penalty on OTB-100 [32]. Besides, TransT [7] with IoU-Guided penalty increases about 4% relatively on OTB-100 [32] and LaSOT [12]. Notably, our IoU-Guided penalty achieves the best AUC and precision score when based on TransT [7].

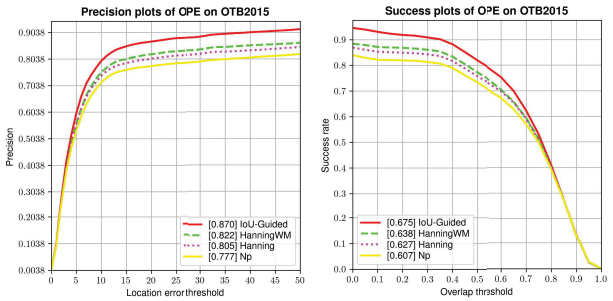

Figure 4. Visualization of results comparison on *Background Clutters* attribute, OTB-100 [32]

Table 2. Comparison of four different penalty methods. **Dc** implies whether to apply distance constraints like Hanning window. **Mo** implies whether to apply the motion model. **Ic** implies whether to apply IoU constraints.

Base	Penalty	Dc	Mo	Ic	OTB-100		LaSOT	
					Succ.	Prec.	Succ.	Prec.
SRPN++	Np				0.643	0.833	0.498	0.492
	Hanning	✓			0.696	0.905	0.495	0.488
	HanningWM	✓	✓		0.660	0.860	**0.504**	**0.497**
	IoU-Guided*		✓	✓	**0.700**	**0.905**	0.499	0.493
TransT	Np				0.679	0.873	0.628	0.668
	Hanning	✓			0.691	0.893	0.649	0.690
	HanningWM	✓	✓		0.700	0.906	0.650	0.690
	IoU-Guided*		✓	✓	**0.709**	**0.917**	**0.654**	**0.691**

The results can verify the effectiveness and universality of our method on the one hand.

On the other hand, in order to verify the effectiveness of our IoU-Guided penalty for solving the drift problem, we compare the calculated IoUs with different penalty meth-

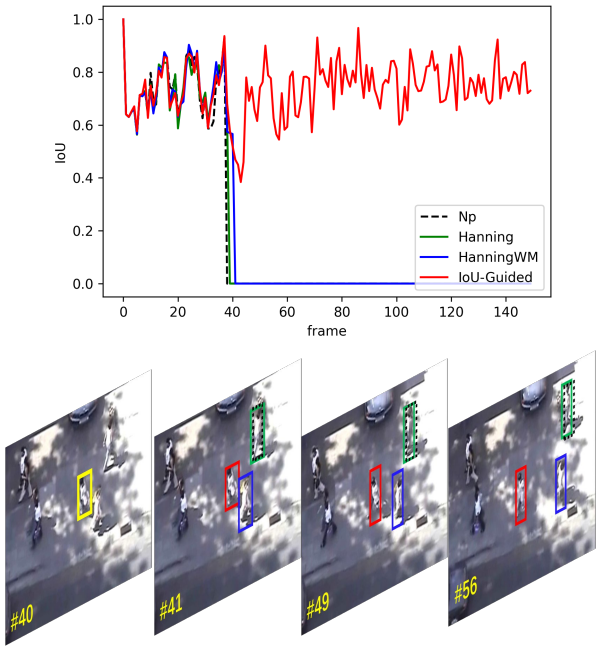

Figure 5. Comparisons of four different penalty methods on the *Crowds* of OTB-100 [32]. Only the IoU-Guided penalty can always track the target. The other methods fail since the 40th frame. The yellow box in the bottom left image is the ground-truth box. The red, blue, green and dashed black boxes correspond to the items in the legend demonstrated in the top figure.

ods on the *Crowds* sequence of OTB-100 [32] and visualize them in Fig 5. The IoU-Guided penalty can always track the target even if there are many distractors. On the contrary, other methods fail since the 40th frame due to distractors and drift to a distant position which violates the assumption of motion continuity. Meanwhile, we compare these four methods on the specific *Background Clutters* attribute on OTB-100 [32] and demonstrate the results in Fig 4. The IoU-Guided penalty surpasses other methods apparently.

4.4. Comparison with Baseline and State-of-the-art

Finally, we combine the IoU-Guided penalty with the Hanning window penalty as the final constraint and compare the performance on several benchmark datasets with the baseline which only exploits the Hanning window penalty. As shown in Table 3, the combination of the IoU-Guided penalty and the Hanning window penalty achieves the best performance and runs at approximately 40 fps, which explicitly satisfies the requirement of real-time tracking. We infer that the combination can simultaneously constrain the spatial distance and the overlap between the reference bounding box and all the regression boxes. Meanwhile, our tracker with IoU-Guided penalty based on TransT [7] performs well on three benchmarks, achieving an AUC score of 71.5% on OTB-100 [32], 66.7% on NFS [18] and 65.9% on LaSOT [12]. The results are demonstrated in Table 3.

Table 3. Result comparisons on three tracking benchmarks. The red and blue indicate performances ranked at the first and second places.

Method	OTB-100 [32]		LaSOT [12]		NFS30 [18]
	Succ.	Prec.	Succ.	Prec.	Succ.
SiamFC [2]	58.7	77.2	33.6	33.9	-
MDNet [23]	67.8	90.9	39.7	37.3	-
ECO [11]	69.1	91.0	32.4	30.1	-
VITAL [25]	69.1	91.7	39.0	36.0	-
GradNet [21]	63.9	86.1	36.5	35.1	-
SiamDW [38]	67.4	90.5	38.4	35.6	-
SiamRPN++ [19]	69.6	92.3	49.6	49.1	-
ATOM [10]	66.7	87.9	51.5	50.5	59.0
DiMP [4]	68.6	89.9	56.9	56.7	62.0
SiamFC++ [34]	68.3	91.2	54.3	54.7	-
MAMLTrack [29]	71.2	92.6	52.3	53.1	-
SiamAttn [35]	71.2	92.6	56.0	-	-
SiamCAR [13]	-	-	50.7	51.0	-
SiamBAN [8]	69.6	91.0	51.4	52.1	59.4
KYS [5]	69.5	91.0	55.4	55.8	63.5
Ocean [39]	67.2	90.2	52.6	52.6	-
SiamRCNN [28]	70.1	89.1	64.8	72.2	63.9
AutoMatch [37]	71.4	**92.6**	58.3	59.9	-
TransT [7] (base)	69.1	89.3	64.9	69.0	65.7
Ours (40fps)	**71.5**	**92.3**	**65.9**	**70.2**	**66.7**
Δ	+2.4%	+3%	+1.0%	+1.2%	+1.4%

5. Conclusion

In this work, we explore the motion information to mitigate the drift problem in visual tracking. Qualitative and quantitative analyses imply that the typical Hanning window penalty may not be the optimal penalty style to act as the spatial constraint. We demonstrate the disadvantages of the Hanning window penalty and propose a novel IoU-Guided penalty based on the prediction of the motion. We find that the prediction of a linear Kalman filter can surprisingly improve the performance. Notably, our method is *cost-free*, *model-free* and with no need for training. Meanwhile, we believe that the post-processing is too simple to throughly solve the drift problem. In future work, we will explore more effective post-processing methods (e.g. RNN), stabilize the output bounding box and improve the accuracy.

Acknowledgment This work was supported by the National Key R&D Program of China (Grant No. 2018AAA0102802, 2018AAA0102800), the Natural Science Foundation of China (Grant No. U2033210, 62172413, 61972394, 62036011, 62192782, 61721004), the Key Research Program of Frontier Sciences, CAS (Grant No. QYZDJ-SSW-JSC040), the China Postdoctoral Science Foundation (Grant No. 2021M693402). Jin Gao was also supported in part by the Youth Innovation Promotion Association, CAS.

References

[1] Marcelo Bertalmio, Guillermo Sapiro, and Gregory Randall. Morphing active contours. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):733–737, 2000. 3

[2] Luca Bertinetto, Jack Valmadre, João F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. Fully-convolutional siamese networks for object tracking, June 2016. 2, 3, 7

[3] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2, 3, 4

[4] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6182–6191, 2019. 1, 3, 7

[5] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Know your surroundings: Exploiting scene infor-

mation for object tracking, Mar. 2020. 3, 7

[6] Ted J Broida and Rama Chellappa. Estimation of object motion parameters from noisy images. *IEEE transactions on pattern analysis and machine intelligence*, (1):90–99, 1986. 3

[7] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking, 2021. 1, 2, 3, 5, 6, 7

[8] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, and Rongrong Ji. Siamese box adaptive network for visual tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6668–6677, 2020. 7

[9] Dorin Comaniciu, Visvanathan Ramesh, and Peter Meer. Kernel-based object tracking. *IEEE Transactions on pattern analysis and machine intelligence*, 25(5):564–577, 2003. 3

[10] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4660–4669, 2019. 1, 3, 7

[11] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6638–6646, 2017. 7

[12] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5374–5383, 2019. 1, 6, 7

[13] Dongyan Guo, Jun Wang, Ying Cui, Zhenhua Wang, and Shengyong Chen. Siamcar: Siamese fully convolutional classification and regression for visual tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6269–6277, 2020. 7

[14] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE transactions on pattern analysis and machine intelligence*, 37(3):583–596, 2014. 3

[15] Michael Isard and Andrew Blake. Condensation—conditional density propagation for visual tracking. *International journal of computer vision*, 29(1):5–28, 1998. 3

[16] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In *2010 20th international conference on pattern recognition*, pages 2756–2759. IEEE, 2010. 2, 3, 4, 5, 6

[17] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409–1422, 2011. 4

[18] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1125–1134, 2017. 1, 7

[19] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4282–4291, 2019. 1, 3, 5, 6, 7

[20] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8971–8980, Salt Lake City, UT, USA, 2018. IEEE. 3

[21] Peixia Li, Boyu Chen, Wanli Ouyang, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Gradnet: Gradient-guided network for visual object tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6162–6171, 2019. 7

[22] Christoph Mayer, Martin Danelljan, Danda Pani Paudel, and Luc Van Gool. Learning target candidate association to keep track of what not to track, Mar. 2021. 3

[23] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4293–4302, 2016. 7

[24] Jianbo Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994. 3

[25] Yibing Song, Chao Ma, Xiaohe Wu, Lijun Gong, Linchao Bao, Wangmeng Zuo, Chunhua Shen, Rynson WH Lau, and Ming-Hsuan Yang. Vital: Visual tracking via adversarial learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8990–8999, 2018. 7

[26] Hai Tao, Harpreet S Sawhney, and Rakesh Kumar. Object tracking with bayesian estimation of dynamic layer representations. *IEEE transactions on pattern analysis and machine intelligence*, 24(1):75–89, 2002. 3

[27] Cor J Veenman, Marcel JT Reinders, and Eric Backer. Resolving motion correspondence for densely moving points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(1):54–72, 2001. 3

[28] Paul Voigtlaender, Jonathon Luiten, Philip H.S. Torr, and Bastian Leibe. Siam r-cnn: Visual tracking by re-detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 3, 7

[29] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. Tracking by instance detection: A meta-learning approach. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6288–6297, 2020. 7

[30] Youming Wang and Xiaoyang Mu. Dynamic siamese network with adaptive kalman filter for object tracking in complex scenes. *IEEE Access*, 8:222918–222930, 2020. 3

[31] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017. 2, 3, 4, 5

[32] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE con-*

ference on computer vision and pattern recognition, pages 2411–2418, 2013. 1, 6, 7

[33] Jingjing Xiao, Linbo Qiao, Rustam Stolkin, and Aleš Leonardis. Distractor-supported single target tracking in extremely cluttered scenes. In *European Conference on Computer Vision*, pages 121–136. Springer, 2016. 3

[34] Yinda Xu, Zeyu Wang, Zuoxin Li, Ye Yuan, and Gang Yu. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12549–12556, 2020. 7

[35] Kai Yang, Zhenyu He, Zikun Zhou, and Nana Fan. Siamatt: Siamese attention network for visual tracking. *Knowledge-based systems*, 203:106079, 2020. 7

[36] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13–es, 2006. 3

[37] Zhipeng Zhang, Yihao Liu, Xiao Wang, Bing Li, and Weiming Hu. Learn to match: Automatic matching network design for visual tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13339–13348, 2021. 7

[38] Zhipeng Zhang and Houwen Peng. Deeper and wider siamese networks for real-time visual tracking, Jan. 2019. 7

[39] Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. *arXiv preprint arXiv:2006.10721*, 2020. 7

[40] Lijun Zhou and Jianlin Zhang. Combined kalman filter and multifeature fusion siamese network for real-time visual tracking. *Sensors*, 19(9):2201, 2019. 3

[41] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 101–117, 2018. 1, 3