Siamese Adaptive Transformer Network for Real-Time Aerial Tracking

Daitao Xing[1], Athanasios Tsoukalas[2], Nikolaos Evangeliou[2], Nikolaos Giakoumidis[3] and Anthony Tzes[2,4]

***Abstract*— Recent visual object trackers provide strong discriminability towards accurate tracking under challenging scenarios while neglecting the inference efficiency. Those methods handle all inputs with identical computation and fail to reduce intrinsic computational redundancy, which constrains their deployment on Unmanned Aerial Vehicles (UAVs). In this work, we propose a dynamic tracker which selectively activates the individual model components and allocates computation resources on demand during the inference, which allows deep network inference on onboard-CPU at real-time speed. The tracking pipeline is divided into several stages, where each stage consists of a transformer-based encoder that generates a robust target representation by learning pixels interdependence. An adaptive network selection module controls the propagation routing path determining the optimal computational graph according to confidence-based criteria. We further propose a spatial adaptive attention network to avoid computational overhead in the transformer encoder, where the self-attention only aggregates the dependencies information among selected points. Our model achieves a harmonious proportion between accuracy and efficiency for dealing with varying scenarios, leading to notable advantages over static models with a fixed computational cost. Comprehensive experiments on aerial and prevalent tracking benchmarks achieve competitive results while operating at high speed, demonstrating its suitability on UAV-platforms which do not carry a dedicated GPU.**

I. INTRODUCTION

Recent developments on visual object tracking techniques have facilitated their applications in a variety of fields such as path planning [1], visual surveillance [2], and border security [3]. Among those, Unmanned Aerial Vehicle (UAV) tracking has drawn increasing attention for its flexibility and adaptability under intricate circumstances. Recent methods have focused on building more profound and accurate models to deal with challenging scenarios like fast motion, low-resolution, frequent occlusion, etc. However, most of those methods neglect to fulfill the real-time requirement on embedded platforms with limited computational resources.

In the visual tracking community, Deep Siamese networks play an important role and achieve remarkable progress on accurate tracking. Recent years have witnessed many successful Siamese models [4]–[9]. The recent research [10]–[12] on attention and transformer mechanisms further speeds up the process of building more powerful trackers. However, most prevalent deep Siamese trackers perform inference with a cumbrous computational graph or fixed network parameters, which constrains their deployment on mobile devices with limited computational power.

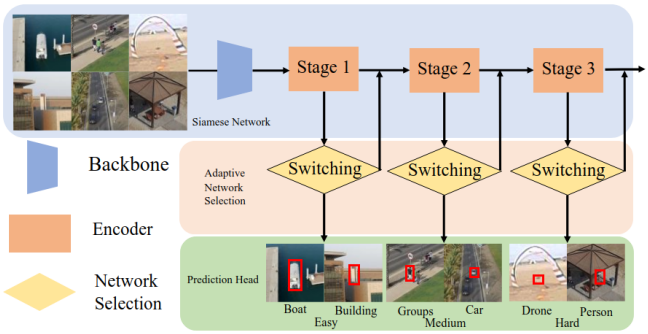

Fig. 1: Overview of the proposed tracking strategy for adaptive inference based on the input scenarios complexity. In most cases, a shallow network with only one encoder block is sufficient to track objects. An array of encoder blocks with increasing representative ability are employed to deal with more challenging cases. The optimal computational graph is determined by the network selection module.

In this work, we aims to achieve a desired trade-off between tracking efficiency and accuracy for dealing with varying computational demand. We introduce a dynamic tracking architecture consisting of two adaptive components, i.e., adaptive attention and adaptive network selection, which adapts the network structures and parameters to the input during inference. As shown in Figure 1, tracking scenarios could be categorized into various levels according to their complexity and can be tracked with the corresponding inference stage. For canonical ("easy") samples where no background noise appears, a shallow network can correctly track the objects. With the occurrence of distractors or other challenging scenarios like fast motion and occlusion caused by the UAV and object relative motion, more computational resources will be allocated for more sophisticated models to recognize the objects. The selection is determined dynamically based on the prediction states from the previous stage without any prior knowledge. To selectively activate the model components, we estimate the Kullback-Leibler divergence between the predicted localization distribution and the expected one. This provides a criterion to determine whether to employ the next inference stage or directly output the results.

To increase the capability of discriminating between the

[1]Department of Computer Science and Engineering, New York University, 11201, New York, USA.
[2]Electrical & Computer Engineering, New York University Abu Dhabi, 129188, Abu Dhabi, UAE.
[3]Kinesis Core Technology Platform (CTP), New York University Abu Dhabi, 129188, Abu Dhabi, UAE.
[4]Center for Artificial Intelligence and Robotics, New York University Abu Dhabi, 129188, Abu Dhabi, UAE.
Corresponding author's email: anthony.tzes@nyu.edu.

target and the background distractors, we employ the transformer as the encode network to model the interdependencies between pixels. However, the computational and memory costs are relatively high even for a feature map with reduced size. To alleviate this problem, we propose a spatial sample attention network to concentrate on the dependencies within selected areas. Based on the predicted object localization distribution, we can preclude most of the irrelevant feature points and construct a robust and holistic target representation more efficiently. Compared with static trackers, which have a fixed network construction and parameters at the inference stage, our tracker (named SiamATN) can adjust the model architecture and allocate appropriate computational resources based on the complexity of the scenario, and therefore reduce the redundant computations on those "easy" cases, leading to notably superior accuracy, computational efficiency, and adaptiveness on the UAV tracking.

Overall, Our main contributions are summarized as follows:

- We introduce an adaptive network selection module to allocate computational resources on-demand at test time, yielding a dynamic tracking network. It fuses the discrimination ability of a deep network and the fast inference of a shallow network and leads to remarkable advantages in terms of efficiency and accuracy.
- We propose a spatial adaptive attention network for learning a robust target representation under an acceptable computational complexity.
- Superior performance on multiple benchmarks demonstrates the effectiveness of the proposed method against other state-of-the-art (SOTA) trackers. Particularly, our approach achieves SOTA results while running at an average of 34 FPS on a single CPU. The performed field tests further validate the efficiency of SiamATN in real world applications.

II. RELATED WORK

Object Tracking. The tracking methods can be divided into: a) Discriminative Correlation Filter (DCF) based trackers and b) deep learning-based trackers. DCF based trackers [13]–[16] could run with real-time speed on CPU, but their performance is constrained by the feature representation ability of handcrafted features. Other DCF tracker [17] introduce deep features to improve on accuracy but suffer in terms of computational speed. In contrast, deep learning based trackers, like the Siamese-based trackers, including anchor based trackers [7], [11] and anchor-free trackers [4], [6], [18], achieve remarkable enhancements in both accuracy and speed by utilizing a high-end GPU device. SiamAPN [8], SiamBAN [9] improves the performance by learning dynamic anchors or adaptive filters. Instead, our tracker adjusts the network architecture to reduce redundant computations.

Transformer. Transformer was first proposed for machine translation in [19] and shows great potential in many sequential tasks. Recent works [10] employ the attention mechanism for more robust tracking. Aerial trackers, SiamAPN++ [11] and HiFT [12], make use of transformer to fuse feature maps for better discrimination and achieve SOTA performance. While those trackers concentrate on building a more robust representation with transformer, our tracker focuses on reducing the intrinsic computational redundancy.

III. SIAMATN FRAMEWORK

In this section, we describe the proposed SiamATN framework. As shown in Figure 2, SiamATN consists of 4 parts: (1) a Siamese network backbone for feature extraction, (2) multiple spatial adaptive attention sub-networks for further feature encoding, (3) the corresponding prediction head for classification and regression and (4) the adaptive network selection module for selectively activating model components.

A. Siamese Networks for Visual Tracking

The Siamese network consists of two parameter sharing branches, i.e., the template branch, which takes the initial cropped frame (denoted as $z \in \mathbb{R}^{H_z \times W_z \times 3}$) as reference image and the search branch, which processes the current frame (denoted as $x \in \mathbb{R}^{H_x \times W_x \times 3}$) for tracking. The Siamese backbone (denoted as $\phi(.)$) performs the same projection on the input z and x and outputs a common embedding feature space $\phi(z) \in \mathbb{R}^{C \times \frac{W_z}{r} \times \frac{H_z}{r}}$ and $\phi(x) \in \mathbb{R}^{C \times \frac{W_x}{r} \times \frac{H_x}{r}}$ for subsequent tasks, where r is the downsample ratio. Specifically, we employ ShuffleNet [20] as the backbone, considering its computational efficiency on mobile devices and extract the feature maps from conv 4 with a spatial downsample ratio $r = \frac{1}{16}$.The lightweight backbones are insufficient for extracting robust discriminative features, which is vital for the tracking performance, especially under uncertainty scenarios. To alleviate this problem, we apply additional encoding blocks to reinforce the feature representation. For an input sample x (or z), the forward propagation of an L-stages encoding network (Sec. III-B) can be formulated as:

$$M_x^L = \mathscr{F}^L \circ \mathscr{F}^{L-1} \circ \cdots \circ \mathscr{F}^1(\phi(x)) \tag{1}$$

where $\mathscr{F}^\ell$ is the encoding network at stage $\ell, 1 \leq \ell \leq L$. Considering the diverse computational demands for different tracking scenarios, we may terminate the inference procedure at an intermediate stage. Specifically, the correlation features on stage ℓ can be represented as:

$$M_x^\ell = \mathscr{F}^\ell \circ \mathscr{F}^{\ell-1} \circ \cdots \circ \mathscr{F}^1(\phi(x)), 1 \leq \ell \leq L \tag{2}$$

where the ℓ is determined base on the adaptive router network (Sec. III-C) and $M_x^0 = \phi(x)$. Afterwards, a depth-wise cross-correlation is performed between M_x^ℓ and M_z^ℓ as $R^\ell = M_x^\ell \star M_z^\ell$, where $\star$ denotes the depth-wise correlation operation and the R^ℓ is a multi-channel response map which is adopted as the input of the prediction head. Inspired from [18], the correlation response map R^ℓ is fed into two parallel branches: one for object classification and centerness prediction and another for bounding box regression. Each branch consists of 3 stacked convolution layers to generate the final results A_{cls}^ℓ, A_{cen}^ℓ and A_{reg}^ℓ, where A_{cls}^ℓ represents the foreground and background probability score map, A_{cen}^ℓ denotes the predicted centerness scores and A_{reg}^ℓ predicts the distances from each feature point to the four sides of the bounding box.

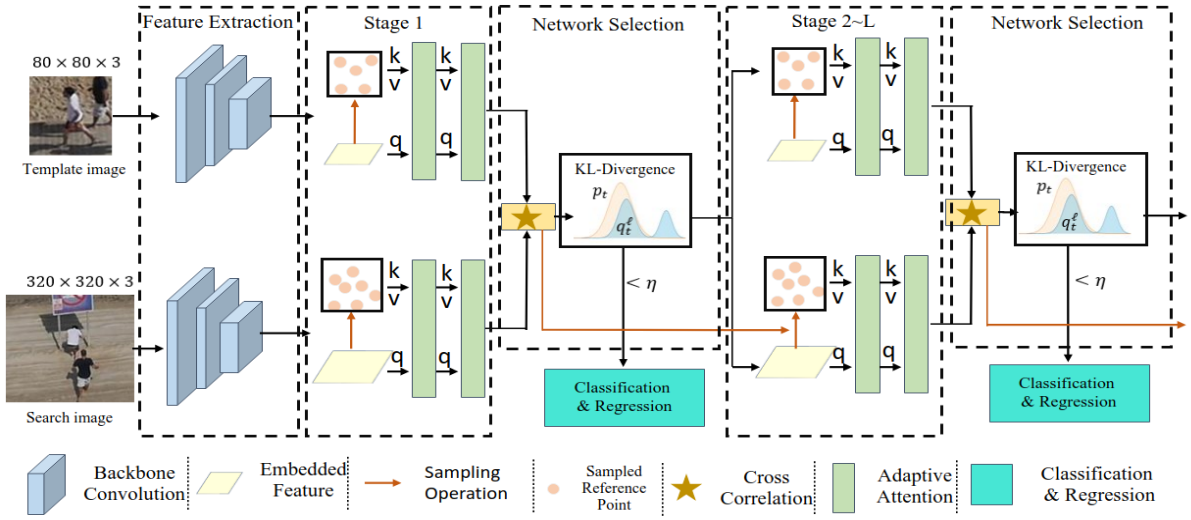

Fig. 2: The overview of the SiamATN tracking framework. It consists of a shared backbone followed by encoding stages and corresponding prediction head. The optimal exit depth is determined by Network Selection module. Auxiliary stages will be activated if the KL-divergence is over η and perform self-attention over adaptive selected point set. The procedure will continue until the exit criterion is satisfied or the network reaches depth maximum.

B. Spatial Adaptive Attention Network

Given the embedding feature $\phi(x)$, the encoder network $\mathscr{F}$ is designed to learn a robust appearance model. However, the Convolution layers are insufficient to learn global dependencies which may degrade the model capacity for localizing the target objects under complex scenarios. Accordingly, we explore the multi-head self-attention mechanism to capture the spatial relationship between feature points and generate discriminative features for target localization. Specifically, let $\mathbf{x}_p^\ell \in \mathbb{R}^d$ denotes an element point of M_x^ℓ, where p is the spatial position. The attention function is performed on query vector $\mathbf{q}_p^\ell$, key vector $\mathbf{k}_p^\ell$ and value vector $\mathbf{v}_p^\ell$ which are learned from $\mathbf{x}_p^\ell$ as:

$$\mathbf{q}_p^\ell = \mathbf{W}_q^\ell \mathbf{x}_p^\ell, \mathbf{k}_p^\ell = \mathbf{W}_k^\ell \mathbf{x}_p^\ell, \mathbf{v}_p^\ell = \mathbf{W}_v^\ell \mathbf{z}_p^\ell \tag{3}$$

where $\mathbf{W}_q^\ell, \mathbf{W}_k^\ell, \mathbf{W}_v^\ell \in \mathbb{R}^{d\times d}$ are learnable weights. By dividing the embedded vector into N parts, each part represents an attention head. And the final attention feature $\hat{\mathbf{x}}_p^\ell$ is calculated by:

$$\hat{\mathbf{x}}_p^\ell = Concat([\sum_{p'=1}^{\chi^\ell} \sigma\left(\frac{\left(\mathbf{q}_p^\ell(i)\right)^T \mathbf{k}_{p'}^\ell(i)}{\sqrt{d_{head}}}\right) \cdot \mathbf{v}_{p'}^\ell(i)]_i^N)\mathbf{W}_o^\ell \tag{4}$$

where $\mathbf{W}_o^\ell \in \mathbb{R}^{d\times d}$ is the learnable projection matrix, d_{head} is the dimension of each head, equal to $\frac{d}{N}$ by default. χ^ℓ is the sample space on ℓ stage which provides reference points as keys and values and is set to M_x^ℓ in full attention scheme. σ is the $softmax$ function and Concat refers to the concatenation operation. The query $\mathbf{q}_p^\ell$ is compared with every key point $\mathbf{k}_{p'}^\ell$ on χ^ℓ and extract useful information from the corresponding value $\mathbf{v}_{p'}^\ell$. The final output of the multi-head attention layer is given after a fully connected feed forward network as:.

$$\mathscr{F}^\ell(\mathbf{x}_p^\ell) = \mathbf{x}_p^\ell + \hat{\mathbf{x}}_p^\ell + \rho(\mathbf{W}_2^\ell \rho(\mathbf{W}_1^\ell(\mathbf{x}_p^\ell + \hat{\mathbf{x}}_p^\ell))) \tag{5}$$

where $\mathbf{W}_1^\ell$ and $\mathbf{W}_2^\ell$ are learnable linear transform weights and ρ is activation function. The same network and parameters are applied on M_z^ℓ to generate the correlation filter. And each $\mathscr{F}^\ell$ consists of $D = 2$ multi-head attention layers.

Despite the strong discriminability provided by the self-attention module, its computational and memory costs are relatively high when traversing the whole feature map. Additionally, most reference points are irrelevant to the query point but only deliver noisy background information. To address this problem, we introduce the spatial attention module, which adaptively selects the reference points based on the response map from previous layer. By ranking A_{cen}^ℓ according to the response score, only the pixels which share a similarity with the target object are taken as the reference features:

$$\chi^\ell = \left[M_x^\ell(i,j,:)\right]_{A_{cen}^{\ell-1}(i,j)>\tau} \tag{6}$$

Only the points with score higher than the threshold τ during tracking are selected as the reference points in χ^ℓ. For batch processing efficiency during training, we replace the above criterion with top-k similarities, i.e., $(i,j) \in TOP(A_{cen}^{\ell-1}; K)$. The selected points serve as key and value features. Then the attention is operated between the query vectors and the reference points and its computational complexity is linear to K. Thus, we model the pixels interdependencies in a computational efficient way.

C. Adaptive Network Selection

To adapt the network structure to the input during the inference, we set up the adaptive routing network to select the appropriate depth of encoding layers and the corresponding classifier. We utilize the background appearance information learned from the prediction heads to estimate the tracking complexity. For stage ℓ at frame t, we estimate the divergence

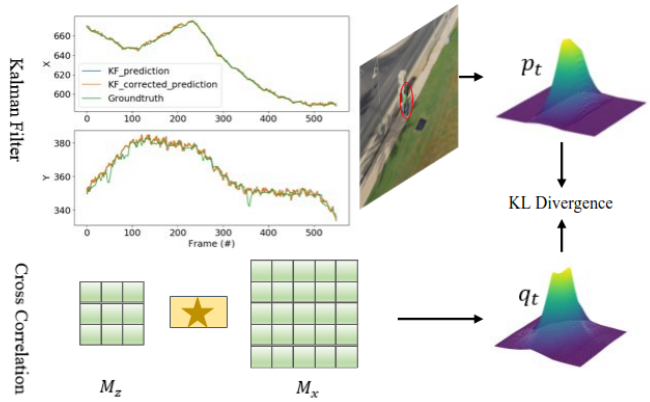

Fig. 3: Visualization of network selection workflow. The trajectory is recorded and predicted by a Kalman filter (KF). The expected distribution p_t (red circle in the frame) generated from KF prediction is compared with estimated distribution q_t generated from cross correlation, yielding KL score for network selection. The line plots are X&Y coordinates from KF predictions $\hat{b}_t^-$,$\hat{b}_t$ and ground truth b_t.

between the predicted target localization distribution q_t^ℓ and the expected distribution p_t with:

$$KL\left(p_t \| q_t^\ell\right) = \int_{\mathscr{X}} p_t(x) \log \frac{p_t(x)}{q_t^\ell(x)} dx \tag{7}$$

where KL is the Kullback-Leibler divergence and $\mathscr{X}$ is the sample space which equals to the size of the feature maps, as shown in Figure 3. In practice, we choose the predicted and ground truth centerness score map as q_t^ℓ and p_t respectively. Note that A_{cls} and A_{reg} are not calculated at this stage. While p_t can be approximated from $\hat{b}_{t-1}$, it may bring target drifts caused by fast motion, camera movement, etc. Therefore, we apply the Kalman filter to correct the observation. Given the estimated state $\hat{b}_t^-$ from the Kalman filter, the optimal stage ℓ_t^{opt} of the encoding network $\mathscr{F}$ is selected by :

$$\ell_t^{opt} = \min\left(\left\{\ell : \max\left\{0, KL(p_t \| q_t^\ell) - \eta\right\} = 0\right\}\right) \tag{8}$$

Where η is the threshold to control the routing path.

D. Training Objectives

Our model is trained in an end-to-end way, where the training objective is a weighted average loss for the prediction branches on each stage:

$$\mathscr{L} = \sum_{\ell=1}^{L} \mathscr{L}^\ell \tag{9}$$

And the loss on stage $\mathscr{L}^\ell$ is:

$$\mathscr{L}^\ell = \lambda_{\text{cls}} \mathscr{L}_{cls}^\ell + \lambda_{cen} \mathscr{L}_{\text{cen}}^\ell + \lambda_{\text{iou}} \mathscr{L}_{\text{iou}}^\ell + \lambda_{L_{reg}} \mathscr{L}_{reg}^\ell, \tag{10}$$

where $\mathscr{L}_{cls}$ is the cross-entropy loss for classification, $\mathscr{L}_{cen}$ is the binary cross entropy loss for the centerness score, $\mathscr{L}_{iou}$ is the GIOU [21] loss between prediction boxes and the ground truth box and $\mathscr{L}_{reg}$ is the $L1$ loss for regression. Constants λ_{cls}, λ_{cen}, λ_{reg} and λ_{iou} weight the losses.

IV. Experimental and Simulation Studies

A. Implementation Details

The backbone, i.e., ShuffleNet [20] is pre-trained on Imagenet [22] and the *conv* 4 of with depth of $C = 232$ is extracted for further encoding. We add a convolution layer to reduce the feature dimension into $d = 192$. The network is trained offline for 100 epochs with 128 image pairs per batch. The patch pairs z and x are cropped from two images of the same video with a maximum gap of 100 frames and are resized into 80×80 pixels and 320×320 respectively. The training data consists of the training splits from LaSOT [23], TrackingNet [24], Got10K [25] and COCO [26]. We set up spatial adaptive attention stages number $L = 3$ as it provides enough representation power for complex cases.For search branch, We take $K = 100$ all stages. We use the ADAMW [27] optimizer with an initial learning rate of 10^{-5} for the backbone parameters and 10^{-4} for rest of components. During training, the first layer and all BatchNorm layers from the backbone are frozen. The learning rate drops by a factor 0.1 on 90 epochs. For all stages, the prediction losses are weighted with $\lambda_{L_{cls}} = 5$, $\lambda_{iou} = 5$ and $\lambda_{reg} = \lambda_{cen} = 2$ respectively. During tracking, η is set to 0.1. All experiments are conducted on an laptop with an Intel i7-9750H CPU and an NVidia 2060 for GPU speed test.

B. Evaluation on Visual Tracking Benchmarks

In this section, we compare our approach with 15 SOTA trackers. There are 3 anchor-based Siamese methods (SiamRPN++ [5], DaSiamRPN [7], SiamAPN [8]), 4 anchor-free Siamese methods (SiamFC [4], SiamFC++ [6], SiamBAN [9] and SiamCar [18]), 5 DCF based methods (ECO [15], fDSST [14], KCF [13], CSRDCF [28], CCOT [17]) and 3 attention based methods (SiamGAT [10], SiamAPN++ [11], HiFT [12]).

UAV123 [29]. is one of the largest UAV tracking benchmarks, including 123 low altitude aerial videos with more than 112K frames, and adopts success and precision metrics for evaluation. We report the performance diversity when using different stages for tracking. As shown in Figure 4, the combination of a lightweight backbone with one attention layer, i.e., SiamATN-stage1 brings remarkable performance enhancement over trackers based on deep networks (Reset50 [30]) or handcraft features (ECO [15]) especially on precision score. With two attention blocks, our tracker achieves comparable results with SOTA free-anchor Siamese tracker SiamBAN [9]. Introducing adaptive network selection won't effect the tracking performance, but help in achieving more stable results with a precision score of 85.2% and success score of 65.0%. The overall results demonstrate that SiamATN achieves superior performance against other SOTA trackers.

UAV20L [29] contains 20 long-term sequences with an average of 3k frames per sequence. As shown in Table I, the classic DCF trackers based on the handcrafted features run at real-time speed on the CPU but have limited accuracy.

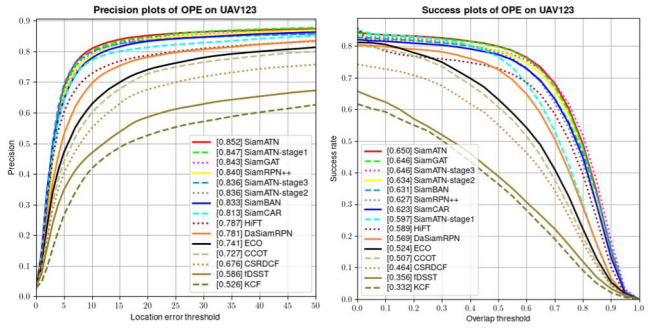

Fig. 4: Evaluation results of trackers on UAV123 [29] dataset.

In contrast, deep trackers relying can achieve high performance but are only applicable on GPU devices. Instead, our SiamATN runs at real-time speed (34Hz) on the CPU while obtaining SOTA results. Specifically, SiamATN gains a precision score of 86.5% and an AUC score of 68.2%, outperforming the recent SOTA Siamese aerial tracker HiFT. Similar to UAV123, we also report the performance and speed under different settings. The tracker without adaptive module gives the best AUC score but costs an extra 2× inference time.

C. Ablation Analysis

Speed, FLOPs and Params. Table II illustrates the complexity analysis of the proposed tracker. For reference, SiamRPN++ [11] has almost 60G multiply–accumulate operations which is too heavy to run fast on the CPU. The adaptive network selection enables the advantage of the lightweight backbone and spatial adaptive attention, and fulfills the balance between the model complexity and the inference speed without adding extra operations.

Visualization of the adaptive network selection. Figure 5 shows a tracking sequence with its KL-divergence score

TABLE I: Overall evaluation on UAV20L [29]. Prec. and Succ. respectively denote precision score at 20 pixels and AUC of success plot. The speed marked with $*$ are test under GPU.SiamATN(i) denotes i stage.

	KCF [13]	CSRDCF [28]	TADT [31]	CCOT [17]	ECO [15]	Siam FC [4]	DaSiam RPN [7]	Siam FC++ [6]
Prec.	0.507	0.588	0.609	0.542	0.589	0.599	0.665	0.695
Succ.	0.298	0.443	0.459	0.403	0.427	0.402	0.465	0.533
FPS	95	6	32.5	1	30	27*	134*	95*
	Siam RPN++ [5]	Siam APN [8]	Siam APN++ [11]	HiFT [12]	Siam ATN(1)	Siam ATN(2)	Siam ATN(3)	Siam ATN
Prec.	0.696	0.721	0.736	0.763	0.827	0.83	0.842	0.865
Succ.	0.528	0.539	0.560	0.566	0.641	0.674	0.685	0.682
FPS	35*	200*	35*	135*	36	25	16	34

TABLE II: Speed, FLOPs, and Params comparison with SiamRPN++ [11]. All speed are tested on CPU. SiamATN(i) denotes i stage.

Trackers	Siam RPN++ [11]	Siam ATN(1)	Siam ATN(2)	Siam ATN(3)	Siam ATN
FLOPs(G)	59.6	0.92	1.6	2.28	0.92-2.31
Params(M)	53.9	2.52	4.16	5.8	5.8
FPS(CPU)	0.8	36	25	16	34
FPS(GPU)	35	89	73	60	83

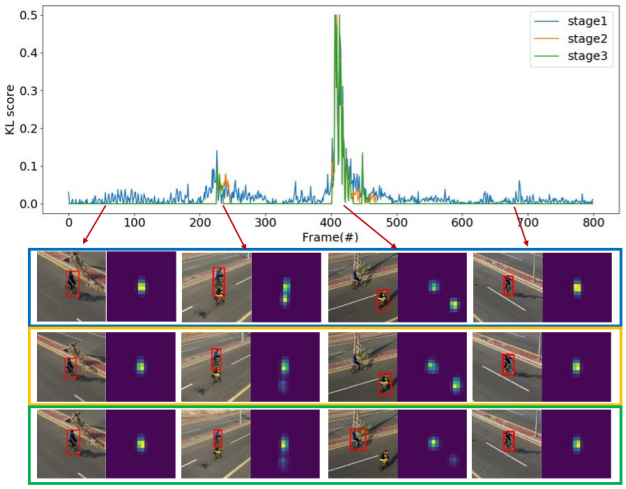

Fig. 5: A long-term tracking example where the the tracking network is adaptively selected according to the KL-divergence score. The bounding box and centerness score predictions for each stage are shown in row 2 to row 4.

for each frame. Only stage 1 is used for tracking when no distractors appears in the background. Otherwise, stage 2 and stage 3 will be triggered to deal with hard negative examples and keep the focus on the tracking object until the background is clear again. Thus, our model allocates model components on demand at test time, leading into a notable advantage in the computational efficiency while maintain a high accuracy.

D. Experimental Field-Test

The field tests are set to: (1) track a fast-moving drone with ground PTZ camera (first video), (2) track a moving person with flying UAV, and (3) track a drone with PTZ camera which is mounted on a flying drone. Those configurations are quite challenging due to a several factors including camera motion, out-of-view, motion blur, scale variance, partial occlusion and object deformation. Figure 6 shows the precise tracking results obtained in complex environments and constrained power resources in real-time on CPU, exhibiting the robustness and practicability of the tracker in real-world applications; the blue lines illustrate the FPS changes during tracking.

V. Conclusions

In this work, a dynamic tracker is implemented achieving real-time speed on CPU with high accuracy. The network selection module adaptively determines the optimal computational graph based on the tracking complexity. The experiments both on aerial benchmark and real-world tests demonstrate its effectiveness and portability on UAV-tracking.

References

[1] K.-H. Lee and J.-N. Hwang, "On-road pedestrian tracking across multiple driving recorders," *IEEE Transactions on Multimedia*, vol. 17, no. 9, pp. 1429–1438, 2015.

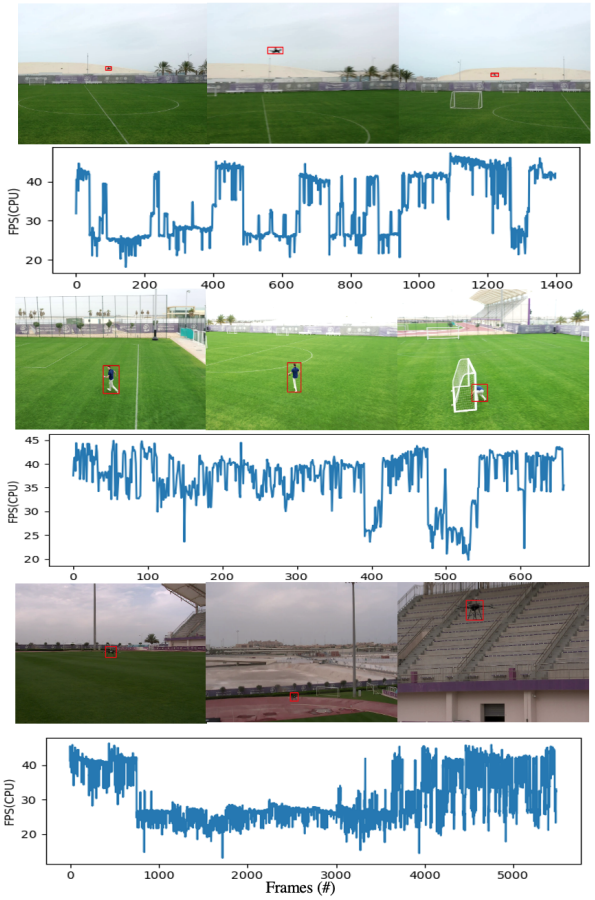

Fig. 6: Visualization of real-world UAV tracking.

[2] S. Tang, M. Andriluka, B. Andres, and B. Schiele, "Multiple people tracking by lifted multicut and person re-identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3539–3548.

[3] A. Tsoukalas, D. Xing, N. Evangeliou, N. Giakoumidis, and A. Tzes, "Deep learning assisted visual tracking of evader-UAV," in *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2021, pp. 252–257.

[4] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 850–865.

[5] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4282–4291.

[6] Y. Xu, Z. Wang, Z. Li, Y. Yuan, and G. Yu, "Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines." in *AAAI*, 2020, pp. 12 549–12 556.

[7] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 101–117.

[8] C. Fu, Z. Cao, Y. Li, J. Ye, and C. Feng, "Siamese anchor proposal network for high-speed aerial tracking," *arXiv preprint arXiv:2012.10706*, 2020.

[9] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6668–6677.

[10] D. Guo, Y. Shao, Y. Cui, Z. Wang, L. Zhang, and C. Shen, "Graph attention tracking," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021.

[11] Z. Cao, C. Fu, J. Ye, B. Li, and Y. Li, "SiamAPN++: Siamese Attentional Aggregation Network for Real-Time UAV Tracking," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 1–7.

[12] ——, "Hift: Hierarchical feature transformer for aerial tracking," *arXiv preprint arXiv:2108.00202*, 2021.

[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2014.

[14] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 8, pp. 1561–1575, 2016.

[15] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6638–6646.

[16] H. Kiani Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1135–1143.

[17] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 472–488.

[18] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "Siamcar: Siamese fully convolutional classification and regression for visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6269–6277.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[20] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 116–131.

[21] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 658–666.

[22] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009.

[23] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, "Lasot: A high-quality benchmark for large-scale single object tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 5374–5383.

[24] M. Muller, A. Bibi, S. Giancola, S. Alsubaihi, and B. Ghanem, "Trackingnet: A large-scale dataset and benchmark for object tracking in the wild," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 300–317.

[25] L. Huang, X. Zhao, and K. Huang, "Got-10k: A large high-diversity benchmark for generic object tracking in the wild," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[26] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[27] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[28] A. Lukezic, T. Vojir, L. ˇCehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6309–6318.

[29] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 445–461.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[31] X. Li, C. Ma, B. Wu, Z. He, and M.-H. Yang, "Target-aware deep tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1369–1378.