

PVT++: A Simple End-to-End Latency-Aware Visual Tracking Framework

Bowen Li^{1*}, Ziyuan Huang^{2*}, Junjie Ye³, Yiming Li⁴, Sebastian Scherer¹, Hang Zhao⁵, Changhong Fu^{3†}
¹Carnegie Mellon University, ²National University of Singapore, ³Tongji University
⁴New York University, ⁵Tsinghua University

{bowenli2, basti}@andrew.cmu.edu, ziyuan.huang@u.nus.edu

{ye.jun.jie, changhongfu}@tongji.edu.cn, yimingli@nyu.edu, hangzhao@tsinghua.edu.cn

Abstract

Visual object tracking is essential to intelligent robots. Most existing approaches have ignored the online latency that can cause severe performance degradation during real-world processing. Especially for unmanned aerial vehicles (UAVs), where robust tracking is more challenging and on-board computation is limited, the latency issue can be fatal. In this work, we present a simple framework for end-to-end latency-aware tracking, i.e., end-to-end predictive visual tracking (PVT++). Unlike existing solutions that naively append Kalman Filters after trackers, PVT++ can be jointly optimized, so that it takes not only motion information but can also leverage the rich visual knowledge in most pre-trained tracker models for robust prediction. Besides, to bridge the training-evaluation domain gap, we propose a relative motion factor, empowering PVT++ to generalize to the challenging and complex UAV tracking scenes. These careful designs have made the small-capacity lightweight PVT++ a widely effective solution. Additionally, this work presents an extended latency-aware evaluation benchmark for assessing an any-speed tracker in the online setting. Empirical results on a robotic platform from the aerial perspective show that PVT++ can achieve significant performance gain on various trackers and exhibit higher accuracy than prior solutions, largely mitigating the degradation brought by latency. Our code is public at https://github.com/Jaraxxus-Me/PVT_pp.git.

1. Introduction

Visual object tracking¹ is fundamental for many robotic applications like navigation [49], cinematography [5], and multi-agent cooperation [9]. Most existing trackers are developed and evaluated under an *offline* setting [38, 29, 34,

*These authors contributed equally to this work.

†Corresponding author

¹We focus on single object tracking in this work.

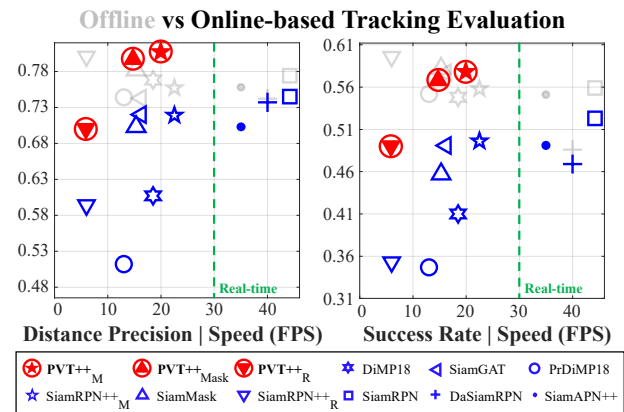


Figure 1. Distance precision and success rate of the trackers on UAVDT dataset [16]. Compared with *offline* evaluation, the trackers suffer a lot from their onboard latency in the *online* setting (30 frames/s (FPS)). Coupled with PVT++, the predictive trackers achieve significant performance gain with very little extra latency, obtaining on par or better results than the *offline* setting.

[33, 6, 8], where the trackers are assumed to have zero processing time. However, in real-world deployment, the online latency caused by the trackers’ processing time cannot be ignored, since the world would have already changed when the trackers finish processing the captured frame. In particular, with limited onboard computation, this issue is more critical in the challenging unmanned aerial vehicle (UAV) tracking scenes [21, 38, 20]. As shown in Fig. 1, compared with *offline* setting (gray markers), the latency can cause severe performance degradation during *online* processing (colored markers). If not handled well, this can easily lead to the failure of robotic applications such as UAV obstacle avoidance [1] and self-localization [61].

To be more specific, the latency hurts online tracking due to: (1) The tracker outputs are always outdated, so there will be mismatch between the tracker result and world state. (2) The trackers can only process the latest frame, so that the non-real-time ones may skip some frames, which makes object motion much larger (see Fig. 2(a) right).

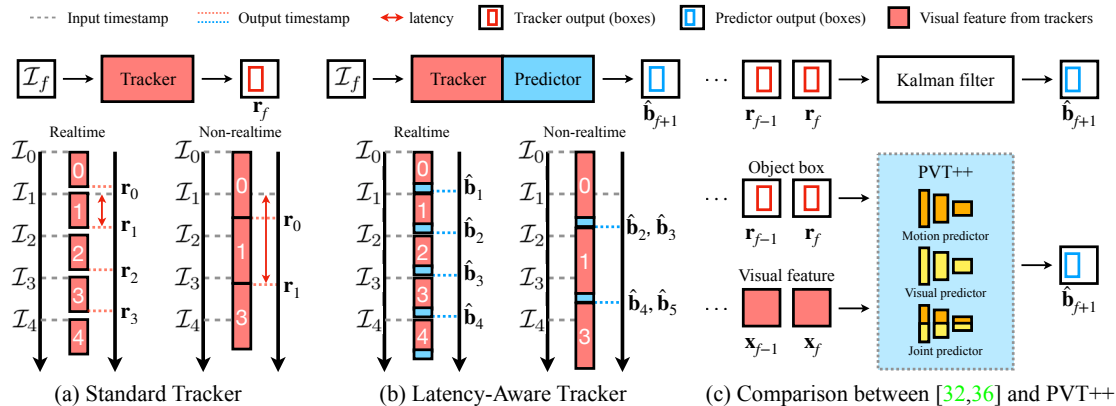


Figure 2. (a) Standard tracker suffers from onboard latency (height of the red boxes). Hence, its result lags behind the world, *i.e.*, r_f is always obtained after \mathcal{I}_f on the timestamp. (b) Latency-aware trackers introduce predictors to compensate for the latency, which predict the world state, \hat{b}_{f+1} , when finishing the processed frame. (c) Compared with prior KF-based solutions [32, 36], our end-to-end framework for latency-aware tracking PVT++ leverages both motion and visual feature for prediction.

The existence of the latency in real-world applications calls for trackers with prediction capabilities, *i.e.*, predictive trackers. While a standard tracker yields the objects’ location in the input frame (*i.e.*, when it *starts* processing the input frame, as in Fig. 2(a)), a predictive tracker predicts where the objects could be when it *finishes* processing the input frame, as illustrated in Fig 2(b).

Existing solutions [32, 36] directly append a Kalman filter (KF) [30] after trackers to estimate the potential object’s location based on its motion model (see Fig 2(c)). However, the rich and readily available visual knowledge from trackers is primarily overlooked, including the object’s appearance and the surrounding environments, which can be naturally exploited to predict the objects’ future paths [51].

To this end, we present a simple framework PVT++ for end-to-end predictive visual tracking. Composed of a tracker and a predictor, PVT++ is able to convert most off-the-shelf trackers into effective predictive trackers. Specifically, to avoid extra latency brought by the predictor, we first design a lightweight network architecture, consisting of a feature encoder, temporal interaction module, and predictive decoder, that leverage both historical motion information and visual cues. By virtue of joint optimization, such a small-capacity network can directly learn from the visual representation provided by most pre-trained trackers for an efficient and accurate motion prediction, as in Fig. 2(c). However, learning this framework is non-trivial due to the training-evaluation domain gap in terms of motion scales. To solve this, we develop a relative motion factor as training objective, so that our framework is independent of the motion scales in training data and can generalize well to the challenging aerial tracking scenes. The integration of lightweight structure and training strategy yields an effective, efficient, and versatile solution.

Beyond methodology, we found that the existing latency-

aware evaluation benchmark (LAE) [32] is unable to provide an effective latency-aware comparison for real-time trackers, since it evaluates the result for each frame as soon as it is given. In this case, the latency for any real-time trackers is one frame. Hence, we present an extended latency-aware evaluation benchmark (e-LAE) for *any-speed* trackers. Evaluated with various latency thresholds, real-time trackers with different speeds can be distinguished.

Empirically, we provide a more general, comprehensive, and practical aerial tracking evaluation for state-of-the-art trackers using our new e-LAE. Converting them into predictive trackers, PVT++ achieves up to **60%** improvement under the *online* setting. As shown in Fig. 1, powered by PVT++, the predictive trackers can achieve comparable or better results than the *offline* setting. Extensive experiments on multiple tracking models [33, 57, 22] and datasets [47, 16, 37] show that PVT++ works generally for latency-aware tracking, which, to the best of our knowledge, is also the first end-to-end framework for *online* visual tracking.

2. Related Work

2.1. Visual Tracking and its Aerial Applications

Visual trackers basically fall into two paradigms, respectively based on discriminative correlation filters [4, 27, 15, 13] and Siamese networks [2, 34, 64, 33, 23, 58]. Compared with general scenarios, aerial tracking is more challenging due to large motions and limited onboard computation resources. Hence, for efficiency, early approaches focus on correlation filters [38, 29, 39, 35]. Later, the development of onboard computation platforms facilitates more robust and applicable Siamese network-based approaches [20, 7, 6, 8].

Most of them are designed under *offline* settings, ignoring the online latency onboard UAVs, which can lead to severe accuracy degradation. We aim to solve the more

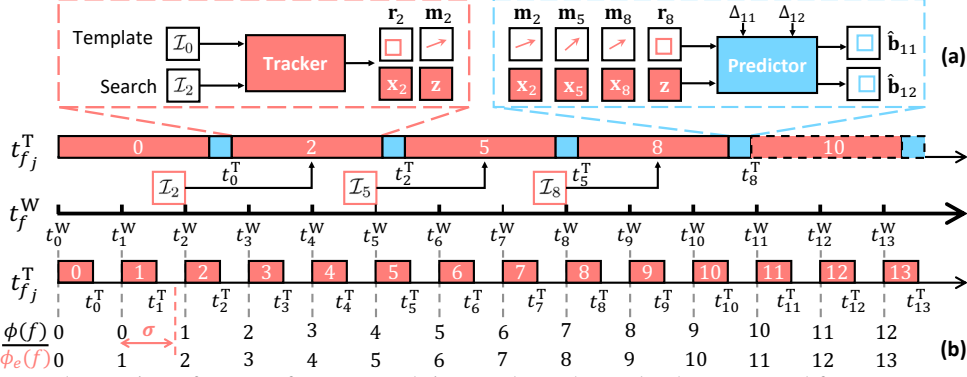


Figure 3. (a) Framework overview of PVT++ for a non-real-time tracker. The tracker has processed frame 0, 2, 5, 8 and obtained corresponding motions \mathbf{m} and visual features \mathbf{x}, \mathbf{z} . The predictor needs to predict future box $\hat{\mathbf{b}}_{11}, \hat{\mathbf{b}}_{12}$ based on tracker result \mathbf{r}_8 . (b) Comparison between LAE ($\phi(f)$) [32] and our e-LAE ($\phi_e(f)$). For real-time trackers, the mismatch between output and input frames will always be one in LAE ($\phi(f) - f \equiv 1$) regardless of the trackers’ various latency. Differently, e-LAE introduces permitted latency thresholds $\sigma \in [0, 1)$, which effectively distinguishes the latency difference of distinct models.

practical and challenging latency-aware perception problem, which goes beyond tracking and is often neglected.

2.2. Latency-Aware Perception

Latency of perception systems is first studied in [36], which introduces a baseline based on the Kalman-filter [30] to compensate for the online latency of object detectors. Inspired by this, [59] converts a real-time detector into a latency-aware one. More recent work [54] also leverage context scenario properties to select best configurations. While most previous work aims at object detection, [32] is more related to us in the field of visual tracking, which introduces dual KFs [30] similar to [36]. Overall, most existing works didn’t address latency-aware tracking through a deeply coupled framework. In this work, we target aerial tracking and present an end-to-end structure.

2.3. Tracking with/by Prediction

Integrating prediction into trackers has been widely adopted for robust perception [41, 51, 46, 19, 42, 44].

Some previous works [42, 44] introduce a predictor to correct the tracker results under object occlusion, which potentially share similar design with PVT++. However, they are still developed for *offline* tracking, *i.e.*, the predictor works only for current frame. Yet predicting the future state for aerial tracking is non-trivial due to the training-evaluation gap. PVT++ aims to address this gap via a relative motion factor, thus working beyond [42, 44].

Others [41, 51, 46, 19] focus on trajectory prediction after tracking. While PVT++ predicts both the trajectory and the object scale. Moreover, as a result of fast camera motion and viewpoint change, UAV tracking scenes are much more complex than the ordered environments in autonomous driving [41, 46]. PVT++ discovers the valuable visual knowledge in pre-trained trackers, yielding its capability for such challenging condition with small capacity.

2.4. Visual Tracking Benchmarks

Various benchmarks are built for large-scale tracking evaluation [18, 48, 28, 17, 43, 47, 31, 62] with different challenges such as first-person perspective [17], aerial scenes [47], illumination conditions [31, 62], and thermal infrared inputs [43]. Since they all adopt *offline* evaluation, the influence of the trackers’ latency is ignored. A recent benchmark targets online evaluation [32], but it falls short in real-time trackers and we aim to improve it in this work.

3. Preliminary

We first introduce the latency-aware tracking task here. The input is an image sequence broadcasting with a certain framerate κ , denoted as $(\mathcal{I}_f, t_f^W), f \in \{0, 1, 2, \dots\}$, where $t_f^W = \frac{f}{\kappa}$ is the world timestamp and f is the frame index. Provided with the ground truth box $\mathbf{b}_0 = [x_0, y_0, w_0, h_0]$ at initial 0-th frame, the tracker estimates the boxes in the following frames. Detailed notation table see Appendix C.

Inference. During inference, the tracker finds the *latest* frame to process when finishing the previous one. Due to the latency, for the j -th frame that the tracker processes, its index j may differ from its frame index f_j in the image sequence. The frame to be processed (frame f_j) is determined by the tracker timestamp $t_{f_{j-1}}^T$ when the model finishes frame f_{j-1} as follows:

$$f_j = \begin{cases} 0 & , j = 0 \\ \arg \max_f t_f^W \leq t_{f_{j-1}}^T & , \text{others} \end{cases} \quad (1)$$

With the frame index f_j , the tracker processes frame \mathcal{I}_{f_j} to obtain the corresponding box $\mathbf{r}_{f_j} = [x_{f_j}, y_{f_j}, w_{f_j}, h_{f_j}]$, forming the raw result of the tracker on the frame $(\mathbf{r}_{f_j}, t_{f_j}^T)$. Since tracker may be non-real-time, input frame ids $f_j, j \in \{0, 1, 2, \dots\}$ may not be consecutive numbers. For example, in Fig. 3 (a), considering a non-real-time tracker, the processed frames are $f_j = 0, 2, 5, 8, \dots$.

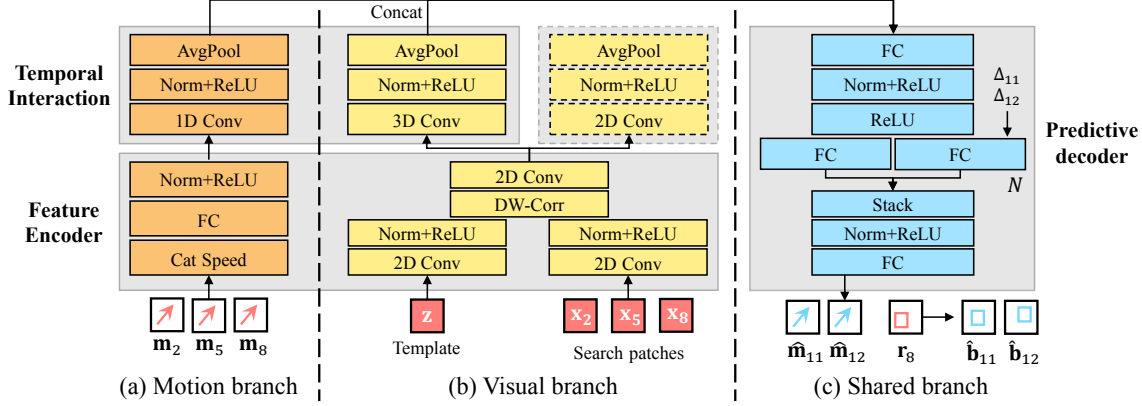


Figure 4. Detailed structure of the predictor modules in PVT++. The models shares similar architecture, *i.e.*, feature encoder, temporal interaction, and predictive decoder. We present the motion branch, visual branch, and share decoding branch in (a), (b), and (c). Dashed blocks denote auxiliary branch, which only exists in training. The input and output are in correspondence to the case in Fig. 3 (a).

Evaluation. Latency-aware evaluation (LAE) [32] compares the ground-truth \mathbf{b}_f in frame \mathcal{I}_f with the *latest* result $\hat{\mathbf{b}}_f$ from the tracker at t_f^W for evaluation. For standard trackers, the latest result $\hat{\mathbf{b}}_f$ to be compared with the ground-truth is obtained as $\hat{\mathbf{b}}_f = \mathbf{r}_{\phi(f)}$, where $\phi(f)$ is defined as:

$$\phi(f) = \begin{cases} 0 & , t_f^W < t_{f_0}^T \\ \arg \max_{f_j} t_{f_j}^T \leq t_f^W & , \text{others} \end{cases} . \quad (2)$$

For instance, in Fig. 3 (b), LAE compares the ground truth \mathbf{b}_3 with the raw tracker result \mathbf{r}_2 .

4. Extended Latency-Aware Benchmark

Existing latency-aware evaluation [36, 32] adopt Eq. (2) to match the raw output $(\mathbf{r}_{f_j}, t_{f_j}^T)$ to every input frame f . However, such a policy fails to reflect the latency difference among real-time trackers. As shown in Fig. 3, since the real-time methods is faster than frame rate, every frame will be processed, *i.e.*, $[f_0, f_1, f_2, \dots] = [0, 1, 2, \dots]$. In this case, the *latest* results will always be from the previous one frame, *i.e.*, using Eq. (2), $\phi(f) \equiv f - 1$. Differently, we extend Eq. (2) to:

$$\phi(f)_e = \begin{cases} 0 & , t_f^W < t_{f_0}^T \\ \arg \max_{f_j} t_{f_j}^T \leq t_f^W + \sigma & , \text{others} \end{cases} , \quad (3)$$

where $\sigma \in [0, 1)$ is the variable permitted latency. Under e-LAE, $\phi(f)_e$ can be $f - 1$ or f for real-time trackers depending on σ . For instance, $\phi(f)_e$ would turn from $f - 1$ to f at larger σ for slower real-time trackers. This extension distinguishes different real-time trackers (see Section 6.2).

5. Predictive Visual Tracking

Because of the unavoidable latency introduced by the processing time, there is always a mismatch between $\phi(f)$ (or $\phi(f)_e$) and f (when σ is small), where $\phi(f)$ is always smaller than f , *i.e.*, $\phi(f) < f$, $f > 0$. To compensate for

the mismatch, we resort to predictive trackers that predicts possible location of the object in frame f . For the evaluation of f -th frame, prior attempts [36, 32] adopt traditional KF [30] to predict the result based on the raw tracking result $\mathbf{r}_{\phi(f)}$ in $\mathcal{I}_{\phi(f)}$ [36], *i.e.*, $\hat{\mathbf{b}}_f = \text{KF}(\mathbf{r}_{\phi(f)})$. Since previous work [36, 32] are not learnable, neither existing large-scale datasets nor the visual feature are leveraged. Differently, our predictive visual tracking framework PVT++ aims for an end-to-end predictive tracker, which takes both the historical motion and visual features for a more robust and accurate prediction. Note that we use $\hat{\cdot}$ to represent the prediction (results for evaluation) and others are from the tracker output or ground-truth in the following subsections.

5.1. General Framework

As in Fig. 3 (a), PVT++ consists of a tracker \mathcal{T} and a predictor \mathcal{P} . For the f -th frame at world time t_f^W , the latest result from the tracker is $\mathbf{r}_{\phi(f)}$ obtained from frame $\mathcal{I}_{\phi(f)}$, *i.e.*, $\mathbf{r}_{\phi(f)} = \mathcal{T}(\mathbf{x}_{\phi(f)}, \mathbf{z})$, where $\mathbf{x}_{\phi(f)}$ is the search feature from $\mathcal{I}_{\phi(f)}$ and \mathbf{z} is the template feature.

After this, the predictor \mathcal{P} takes input from the information generated during tracking of the k past frames (including $\mathcal{I}_{\phi(f)}$), denoted as $\text{Input}_{\phi(f)}$, and predict the position offset normalized by object’s scale, *i.e.*, motion $\hat{\mathbf{m}}_f = [\frac{\Delta_{\hat{x}}(f)}{w_{\phi(f)}}, \frac{\Delta_{\hat{y}}(f)}{h_{\phi(f)}}, \text{Log}(\frac{\hat{w}_f}{w_{\phi(f)}}), \text{Log}(\frac{\hat{h}_f}{h_{\phi(f)}})]$, where $\Delta_{\hat{x}}(f)$ and $\Delta_{\hat{y}}(f)$ denote the predicted box center distance between the f -th and $\phi(f)$ -th frame. $w_{\phi(f)}$ and $h_{\phi(f)}$ are the tracker’s output box scale in frame $\phi(f)$ and \hat{w}_f, \hat{h}_f are the predicted scale in f -th frame. With the raw output $\mathbf{r}_{\phi(f)}$ at $\phi(f)$ and the motion $\hat{\mathbf{m}}_f$ from $\mathcal{I}_{\phi(f)}$ to the f -th frame, the predicted box $\hat{\mathbf{b}}_f$ can be easily calculated.

Relative Motion Factor: Due to the large domain gap between the training [52] and evaluation [37] in terms of the absolute motion scale, we find directly using the absolute motion value $\hat{\mathbf{m}}_f$ as the objective can result in poor performance (see Table 4). Therefore, we define the output of

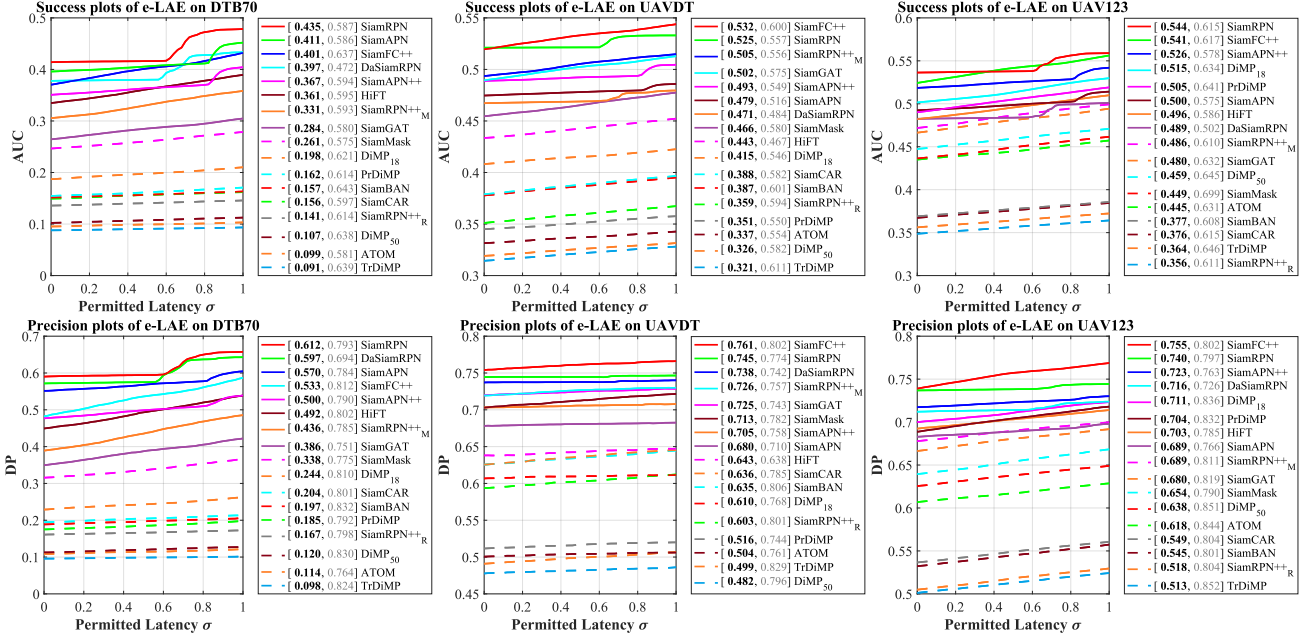


Figure 5. The performance of the SOTA trackers in authoritative UAV tracking benchmarks under our e-LAE benchmark. We report [online mAUC and mDP, offline AUC and DP] in the legend. All trackers struggle to overcome onboard latency in online tracking.

predictor \mathcal{P} to be the relative motion factor based on the average moving speed \mathbf{p}_{f_j} from the past k frames, which we find is easier to generalize after training:

$$\hat{\mathbf{m}}_f = \mathcal{P}(\text{Input}_{\phi(f)}, \Delta_f) \odot \mathbf{p}_{f_j}, \quad \mathbf{p}_{f_j} = \frac{1}{k} \sum_{i=j-k+1}^j \frac{\mathbf{m}_{f_i}}{\Delta_{f_i}}, \quad (4)$$

where $\Delta_f = f - \phi(f)$ denotes the frame interval between current and target frame, and f_j is the latest processed frame, *e.g.*, $\phi(f)$. $\Delta_{f_i} = f_i - f_{i-1}$ denotes the frame interval between $(i-1)$ and i -th processed frame. \odot indicates element-wise multiplication. \mathbf{m}_{f_i} is the normalized input motion defined as $\mathbf{m}_{f_i} = [\frac{\Delta_x(f_i)}{w_{f_{i-1}}}, \frac{\Delta_y(f_i)}{h_{f_{i-1}}}, \text{Log}(\frac{w_{f_i}}{w_{f_{i-1}}}), \text{Log}(\frac{h_{f_i}}{h_{f_{i-1}}})]$, where $\Delta_x(f_i) = x_{f_i} - x_{f_{i-1}}$ and $\Delta_y(f_i) = y_{f_i} - y_{f_{i-1}}$ are the distance from tracker results \mathbf{r}_{f_i} and $\mathbf{r}_{f_{i-1}}$. Such design has made PVT++ agnostic to the specific motion of a dataset, which is crucial for its generalization capability.

We next present the predictor of PVT++ step by step as motion-based \mathcal{P}_M , visual-appearance-based \mathcal{P}_V and multi-modal-based \mathcal{P}_{MV} . All the predictors share the same training objective (Eq. (4)) and a similar structure, consisting of feature encoding, temporal interaction, and predictive decoding as in Fig. 4. In practice, a predictor may need to predict N results, depending on the tracker’s latency.

5.2. Motion-based Predictor

The motion-based predictor \mathcal{P}_M only relies on the past motion, *i.e.*, $\text{Input}_{\phi(f)} = \mathbf{m}_{f_{j-k+1}:f_j}$,

$$\hat{\mathbf{m}}_{f,M} = \mathcal{P}_M(\mathbf{m}_{f_{j-k+1}:f_j}, \Delta_f) \odot \mathbf{p}_{f_j}, \quad (5)$$

where $\mathbf{m}_{f_{j-k+1}:f_j} = [\mathbf{m}_{f_{j-k+1}}, \dots, \mathbf{m}_{f_j}] \in \mathbb{R}^{k \times 4}$.

The detailed model structure of the motion predictor \mathcal{P}_M is presented in Fig. 4(a). For pre-processing, the motion data $\mathbf{m}_{f_{j-k+1}}, \dots, \mathbf{m}_{f_j}$ are first concatenated. Then we apply a fully connected (FC) layer with non-linearity for feature encoding and a 1D convolution followed by activation and global average pooling to obtain the temporally interacted motion feature. In the predictive decoding head, a share FC layer with non-linearity is used for feature mapping. N independent FCs map the feature to N future latent spaces. Finally, the latency features are stacked and transformed to 4 dimension output using a shared FC.

For training, we adopt \mathcal{L}_1 loss between prediction and ground-truth $\mathcal{L}_M = \mathcal{L}_1(\hat{\mathbf{m}}_{f,M}, \mathbf{m}_f)$.

5.3. Visual Appearance-based Predictor

For efficiency, our visual predictor \mathcal{P}_V takes search and template features directly from the tracker backbone as input. Besides, we also find the strong representation in the pre-trained tracker models can boost the small-capacity predictor network. Specifically, template feature $\mathbf{z} \in \mathbb{R}^{1 \times C_V \times a \times a}$ is extracted from the given object template patch in the initial frame and search feature $\mathbf{x}_{f_j} \in \mathbb{R}^{1 \times C_V \times s \times s}$ is obtained from the f_j -th frame patch cropped around $(x_{f_{j-1}}, y_{f_{j-1}})$. Given k past search features $\mathbf{x}_{f_{j-k+1}:f_j} \in \mathbb{R}^{k \times C_V \times s \times s}$ and \mathbf{z} , we have:

$$\hat{\mathbf{m}}_{f,V} = \mathcal{P}_V(\mathbf{x}_{f_{j-k+1}:f_j}, \mathbf{z}, \Delta_f) \odot \mathbf{p}_{f_j}. \quad (6)$$

The detailed model structure of \mathcal{P}_V is shown in Fig. 4(b). Inspired by Siamese trackers [33], the feature encoding

Table 1. The effect of PVT++ on the four SOTA trackers with different inference speeds and backbones. Our models work generally for different tracker structures and can achieve up to **60%** performance gain. The best scores are marked out in **gray** for clear reference. We present some qualitative visualization in Appendix D and the supplementary video.

Tracker	Dataset PVT++	DTB70		UAVDT		UAV20L		UAV123	
		AUC@La0 $_{\Delta\%}$	DP@La0 $_{\Delta\%}$	AUC@La0 $_{\Delta\%}$	DP@La0 $_{\Delta\%}$	AUC@La0 $_{\Delta\%}$	DP@La0 $_{\Delta\%}$	AUC@La0 $_{\Delta\%}$	DP@La0 $_{\Delta\%}$
SiamRPN++ _M (~22FPS) (MobileNet [53])	N/A	0.305+0.00	0.387+0.00	0.494+0.00	0.719+0.00	0.448+0.00	0.619+0.00	0.472+0.00	0.678+0.00
	\mathcal{P}_M	0.385+26.2	0.523+35.1	0.529+7.10	0.745+3.60	0.481+7.40	0.647+4.50	0.537+13.8	0.737+8.70
	\mathcal{P}_V	0.352+15.4	0.472+22.0	0.564+14.2	0.799+11.1	0.488+8.90	0.675+9.00	0.504+6.80	0.703+3.70
	\mathcal{P}_{MV}	0.399+30.8	0.536+38.5	0.576+16.6	0.807+12.2	0.508+13.4	0.697+12.6	0.537+13.8	0.741+9.30
SiamRPN++ _R (~6FPS) (ResNet50 [26])	N/A	0.136+0.00	0.159+0.00	0.351+0.00	0.594+0.00	0.310+0.00	0.434+0.00	0.349+0.00	0.505+0.00
	\mathcal{P}_M	0.199+46.3	0.258+62.3	0.449+27.9	0.684+15.2	0.404+30.3	0.560+29.0	0.442+26.6	0.627+24.2
	\mathcal{P}_V	0.179+31.6	0.225+41.5	0.403+14.8	0.665+12.0	0.398+28.4	0.548+26.3	0.398+14.0	0.559+10.7
	\mathcal{P}_{MV}	0.205+50.7	0.256+61.0	0.488+39.0	0.726+22.2	0.416+34.2	0.568+30.9	0.442+26.6	0.619+22.6
SiamMask (~14FPS) (ResNet50 [26])	N/A	0.247+0.00	0.313+0.00	0.455+0.00	0.703+0.00	0.405+0.00	0.571+0.00	0.436+0.00	0.639+0.00
	\mathcal{P}_M	0.370+49.8	0.508+62.3	0.531+16.7	0.760+8.10	0.449+10.9	0.607+6.30	0.532+22.0	0.743+16.9
	\mathcal{P}_V	0.292+18.2	0.405+29.4	0.532+16.9	0.777+10.5	0.430+6.20	0.601+5.30	0.503+15.4	0.705+10.7
	\mathcal{P}_{MV}	0.342+29.5	0.463+47.9	0.566+24.4	0.797+13.4	0.469+15.8	0.644+12.8	0.536+22.9	0.749+17.2
SiamGAT (~16FPS) (GoogleNet [55])	N/A	0.264+0.00	0.347+0.00	0.489+0.00	0.720+0.00	0.475+0.00	0.663+0.00	0.466+0.00	0.666+0.00
	\mathcal{P}_M	0.396+50.0	0.520+49.9	0.549+12.3	0.777+7.90	0.519+9.26	0.705+6.33	0.530+13.7	0.739+11.0
	\mathcal{P}_V	0.392+48.5	0.536+54.4	0.575+17.6	0.801+11.3	0.503+5.90	0.686+3.50	0.514+10.3	0.713+7.10
	\mathcal{P}_{MV}	0.415+57.2	0.561+61.7	0.583+19.2	0.796+10.6	0.519+9.3	0.708+6.80	0.531+13.9	0.733+10.1

stage adopts 1×1 convolution before depth-wise correlation (DW-Corr) to produce the similarity map $\mathbf{x}_{f_j-k+1:f_j}^e \in \mathbb{R}^{k \times C_V \times s' \times s'}$. For temporal interaction, we apply 3D convolution and global average pooling.

We find directly training \mathcal{P}_V meets convergence difficulty (See Section 6.3). We hypothesize this is because the intermediate similarity map $\mathbf{x}_{f_j-k+1:f_j}^e$ fails to provide explicit motion information. To solve this, we introduce an auxiliary branch \mathcal{A} , which takes $\mathbf{x}_{f_j-k+1:f_j}^e$ as input to obtain the corresponding motion $\mathbf{m}_{f_j-k+1:f_j}^e$,

$$\mathbf{m}_{f_j-k+1:f_j}^e = \mathcal{A}(\mathbf{x}_{f_j-k+1:f_j}^e). \quad (7)$$

During training, we supervise both the auxiliary branch and the predictive decoder, *i.e.*, $\mathcal{L}_V = \mathcal{L}_1(\hat{\mathbf{m}}_{f,V}, \mathbf{m}_f) + \mathcal{L}_1(\mathbf{m}_{f_j-k+1:f_j}^e, \mathbf{m}_{f_j-k+1:f_j})$.

5.4. Multi-Modality-based Predictor

The final predictor \mathcal{P}_{MV} is constructed as a combination of motion \mathcal{P}_M and visual predictors \mathcal{P}_V as,

$$\hat{\mathbf{m}}_{f,MV} = \mathcal{P}_{MV}(\mathbf{m}_{f_j-k+1:f_j}, \mathbf{x}_{f_j-k+1:f_j}, \mathbf{z}, \Delta_f) \odot \mathbf{p}_{f_j}. \quad (8)$$

As shown in Fig. 4, the encoding and temporal interaction parts of \mathcal{P}_M and \mathcal{P}_V run in parallel to form the first two stages of \mathcal{P}_{MV} . We concatenate the encoded feature vectors to obtain the multi-modal feature. The predictive decoder follows the same structure to obtain future motions $\hat{\mathbf{m}}_{f,MV}$. We also tried different fusion strategy in Appendix H.

For training, we add the two additional predictive decoders respectively after motion and visual predictors to help them predict $\hat{\mathbf{m}}_{f,M}$ and $\hat{\mathbf{m}}_{f,V}$, which yields the loss $\mathcal{L}_{MV} = \alpha_M \mathcal{L}_M + \alpha_V \mathcal{L}_V + \mathcal{L}_1(\hat{\mathbf{M}}_{f,MV}, \mathbf{M}_f)$. During inference, we only use the joint predictive decoder.

Remark 1: The predictors \mathcal{P}_M , \mathcal{P}_V and \mathcal{P}_{MV} can be jointly optimized with tracker \mathcal{T} .

6. Experiments

6.1. Implementation Details

Platform and Datasets. PVT++ is trained on VID [52], LaSOT [18], and GOT10k [28] using one Nvidia A10 GPU. The evaluation takes authoritative UAV tracking datasets, UAV123, UAV20L [47], DTB70 [37], and UAVDT [16] on typical UAV computing platform, Nvidia Jetson AGX Xavier, for realistic robotic performance. Since the online latency can fluctuate, we run three times and report the average performance. For simplicity, we only consider the tracker’s processing latency during evaluation.

Metrics. Following [21], we use two basic metrics, the distance precision (DP) based on center location error (CLE) and area under curve (AUC) based on intersection over union. Under e-LAE, different permitted latency σ corresponds to different DP and AUC, *i.e.*, DP@La σ and AUC@La σ . We use mDP and mAUC to indicate the area under curve for DP@La σ and AUC@La σ , $\sigma \in [0 : 0.02 : 1]$.

Parameters. For e-LAE, all the evaluated trackers use their official parameters for fairness. To represent the most common case, the image frame rate is fixed to $\kappa = 30$ frames/s (FPS) in all the online evaluation. For the PVT++ models, we use $k = 3$ past frames. To determine N for different models, we pre-run the trackers 3 times and record the maximum number of skipped frames, so that when the latency of one specific frame fluctuates, PVT++ can always cover the skipped frame and make sufficient predictions. Detailed training configurations can be found in Appendix B.

6.2. Extended Latency-Aware Evaluation

We evaluate a total of 17 SOTA trackers² under e-LAE: SiamRPN [34], SiamRPN++_M [33], SiamRPN++_R [33], SiamMask [57], SiameseFC++ [58], DaSiamRPN [64],

²Subscripts denote the backbone used, *i.e.*, MobileNet [53], and ResNet 18 or 50 [26].

Table 2. Attribute-based analysis of PVT++ in UAVDT [16]. We found different modality has their specific advantage. Together, the joint model can utilize both and is the most robust under complex UAV tracking challenges. Gray denotes best results.

Tracker	Metric Att.	AUC@La0							
		BC	CR	OR	SO	IV	OB	SV	LO
SiamRPN++ _M	N/A	0.448	0.450	0.438	0.494	0.539	0.525	0.490	0.422
	\mathcal{P}_M	0.461	0.495	0.481	0.549	0.578	0.542	0.505	0.521
	\mathcal{P}_V	0.504	0.520	0.538	0.525	0.588	0.568	0.584	0.436
	\mathcal{P}_{MV}	0.505	0.535	0.549	0.545	0.599	0.589	0.586	0.511
SiamMask	N/A	0.404	0.425	0.404	0.468	0.475	0.471	0.438	0.389
	\mathcal{P}_M	0.465	0.503	0.491	0.536	0.558	0.542	0.526	0.421
	\mathcal{P}_V	0.488	0.498	0.504	0.495	0.563	0.527	0.541	0.494
	\mathcal{P}_{MV}	0.520	0.522	0.541	0.540	0.596	0.560	0.566	0.520

Tracker	Metric Att.	DP@La0							
		BC	CR	OR	SO	IV	OB	SV	LO
SiamRPN++ _M	N/A	0.659	0.643	0.638	0.779	0.777	0.772	0.680	0.569
	\mathcal{P}_M	0.666	0.684	0.681	0.815	0.811	0.778	0.691	0.717
	\mathcal{P}_V	0.733	0.720	0.753	0.793	0.835	0.822	0.796	0.585
	\mathcal{P}_{MV}	0.727	0.732	0.764	0.814	0.848	0.846	0.794	0.694
SiamMask	N/A	0.628	0.620	0.612	0.803	0.743	0.756	0.650	0.571
	\mathcal{P}_M	0.672	0.702	0.709	0.818	0.797	0.802	0.729	0.590
	\mathcal{P}_V	0.718	0.696	0.723	0.787	0.817	0.801	0.763	0.696
	\mathcal{P}_{MV}	0.731	0.712	0.752	0.819	0.829	0.813	0.783	0.711

Table 3. Dimension analysis of different modules in PVT++ on DTB70 [37] and UAVDT [16]. Enc._M and Enc._V represent the motion and visual encoders, respectively. Dec._{MV} denotes the joint decoder. * indicates our default setting. We find the channel dimension of PVT++ can be small, so that it introduces very few extra latency on robotics platforms.

Dim. of Modules	DTB70		UAVDT	
	Enc. _M	Enc. _V	mAUC	mDP
Base Tracker			0.305	0.387
32			0.359	0.483
64*	64*	32*	0.399	0.536
128			0.373	0.504
32			0.363	0.493
64*	64*	32*	0.399	0.536
128			0.364	0.486
32			0.359	0.483
64*	64*	32*	0.399	0.536
128			0.373	0.503
32			0.363	0.493
64*	64*	32*	0.399	0.536
128			0.362	0.485

SiamAPN [20], SiamAPN++ [7], HiFT [6], SiamGAT [22], SiamBAN [10], SiamCAR [23], ATOM [12], DiMP₅₀ [3], DiMP₁₈ [3], PrDiMP [14], and TrDiMP [56].

As in Fig. 5, we draw curve plots to reflect their performance in AUC and DP metrics under different permitted latency σ . We report the [online mAUC and mDP, offline AUC and DP] in the legend. Some *offline* highly accurate trackers like SiamRPN++_R [33], SiamCAR [23], SiamBAN [10], and ATOM [12] can degrade by up to **70%** in our online evaluation setting.

Remark 2: e-LAE can better assess the real-time trackers. In DTB70, SiamAPN++ and HiFT are two real-time trackers with HiFT more accurate in success. While since SiamAPN++ is faster, its e-LAE performance will be better.

6.3. Empirical Analysis of PVT++

Overall Effect. To evaluate PVT++, we construct predictive trackers with four well-known methods, *i.e.*, SiamRPN++_M [33], SiamRPN++_R [33], SiamMask [57],

Table 4. Ablation studies on DTB70 [37]. Official version of PVT++ is marked out in Blackbody. The subscripts * means predicting raw value instead of motion factor, † denotes training without auxiliary supervision, and ‡ indicates training with tracker fixed. Red denotes improvement and blue represents dropping.

Ablate Module	Motion Factor*	Auxiliary Supervision† and Joint Training‡							
Method	Base	\mathcal{P}_M	\mathcal{P}_M^*	\mathcal{P}_V	\mathcal{P}_V^\dagger	\mathcal{P}_V^\ddagger	\mathcal{P}_{MV}	\mathcal{P}_{MV}^\dagger	$\mathcal{P}_{MV}^\ddagger$
AUC@La0	0.305	0.385	0.300	0.352	0.311	0.278	0.399	0.323	0.294
Delta%	0.00	26.2	-1.60	15.4	2.00	-8.90	30.8	5.90	-3.60
DP@La0	0.387	0.523	0.383	0.472	0.412	0.349	0.536	0.429	0.387
Delta%	0.00	35.1	-1.00	22.0	6.50	-9.80	38.5	-10.9	0.00

and SiamGAT [22]. As in Table 1, with PVT++, their online performance can be significantly boosted by up to **60%**, which sometimes is better than their *offline* performance. PVT++ also works for recent transformer-based trackers [60, 11], the results can be found in Appendix F.

Remark 3: Real-time trackers [20, 7, 6, 34] perform generally better than non-real-time ones in online evaluation. While we observe that non-real-time trackers empowered by PVT++ can notably outperform real-time ones. *E.g.*, SiamRPN++_M [33] with \mathcal{P}_{MV} achieves an amazing **0.807** mDP in UAVDT, better than SiamFC++ [58] (0.761).

Attribute-based Analysis. For a comprehensive evaluation, we follow [16] and evaluate PVT++ on various challenge attributes³. From Table 2, We found that motion and vision have advantages in different attributes. \mathcal{P}_V improves CR and OR, while \mathcal{P}_M is good at SO and LO. The joint model \mathcal{P}_{MV} makes use of both and is the most robust under various complex aerial tracking challenges. For the full attribute analysis, please see Appendix I.

Dimension Analysis. In addition to its promising performance, PVT++ can also work with very small capacity, which contributes to its lightweight architecture and high efficiency on low-powered UAVs. We analyse the modules of PVT++ with different feature channels in Table 3, where 64 channels for encoders (Enc._M, Enc._V) and 32 channels for the joint decoder (Dec._J) work best. We present more efficiency and complexity comparisons with other motion predictors [63, 24, 40] in Appendix G.

Ablation Studies. We ablate the effect of motion factor prediction, auxiliary branch, and the joint training of PVT++ on DTB70 [37] with SiamRPN++_M in Table 4. Compared with directly predicting the motion value (\mathcal{P}_M^\dagger), using *motion factor* as the prediction target (\mathcal{P}_M) can yield much better performance. Removing *auxiliary branch* \mathcal{A} in \mathcal{P}_V and \mathcal{P}_{MV} to be \mathcal{P}_V^\dagger and \mathcal{P}_{MV}^\dagger , we observe a significant performance drop due to the difficulty in convergence. *Joint training* the tracker and the predictor (\mathcal{P}_V & \mathcal{P}_{MV}) perform much better than fixing the tracker (\mathcal{P}_V^\dagger and \mathcal{P}_{MV}^\dagger). Training loss of the ablation studies are visualized in Appendix J.

³Background clutter (BC), camera rotation (CR), object rotation (OR), small object (SO), illumination variation (IV), object blur (OB), scale variation (SV), and large occlusion (LO).

Table 5. Averaged results comparison on four datasets [47, 16, 37]. \mathcal{P}_M can achieve better results than prior KF-based solutions [36, 32]. Further introducing visual cues, PVT++ can acquire higher robustness. KF^\dagger and \ddagger denotes learnable baselines [50, 25].

Tracker	Type	Tradition Model			Learning-based				
	Pred.	N/A	KF[36]	PVT[32]	KF^\dagger [50]	KF^\ddagger [25]	\mathcal{P}_M	\mathcal{P}_V	\mathcal{P}_{MV}
SiamRPN++ _M	AUC@La0	0.43	0.462	0.473	0.466	0.481	0.483	0.477	0.505
	DP@La0	0.601	0.639	0.651	0.642	0.658	0.663	0.662	0.695
SiamMask	AUC@La0	0.386	0.441	0.465	0.458	0.468	0.471	0.439	0.478
	DP@La0	0.557	0.607	0.639	0.631	0.638	0.655	0.622	0.663
SiamRPN++ _R	AUC@La0	0.287	0.361	0.374	0.376	0.386	0.374	0.345	0.388
	DP@La0	0.423	0.502	0.523	0.527	0.532	0.532	0.499	0.542

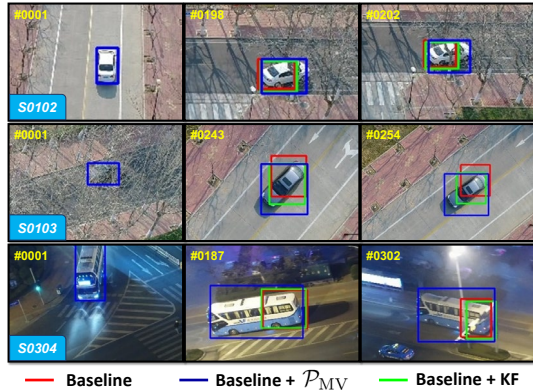


Figure 6. Prediction comparison on UAVDT [16]. Red denotes the original trackers, green indicates KF [36] predictions, and blue represents PVT++ prediction. Compared to KF, PVT++ is better at handling in-plane rotation, scale variation, and view point change.

6.4. Comparison with KF-based Solutions

Quantitative Results. Prior attempts to latency-aware perception [36, 32] have introduced model-based approach, *i.e.*, KF [30], as predictors. Based on traditional KF, we also designed stronger learnable baselines, KF^\dagger [50] and KF^\ddagger [25], which adopt the same training as PVT++ models. Basically, KF^\dagger [50] learns the two noise matrix and KF^\ddagger denotes joint training of KF^\dagger and trackers via back-propagation [25]. We compare these KF-based solutions with PVT++ in Table 5, where the same base tracker models are adopted. Compared with the KFs [36, 32], our learning framework holds the obvious advantage in complex UAV tracking scenes. We also observed that PVT++ is very efficient and introduces very little extra latency on the trackers. For specific results per dataset, please refer to Appendix E.

Qualitative Results. To better present the priority of PVT++, some representative scenes are displayed in Fig. 6. Given the same history trajectory, PVT++ holds its advantage against KF-based solution [36]. Especially, when UAV tracking challenges like in-plane rotation (sequence S0103) and aspect ration change (sequence S0304) appear, PVT++ is capable of fully utilizing the appearance change for robust prediction while simple motion-based KF easily fails.

Remark 4: PVT++ outputs N results in a single forward pass instead of autoregressively like KF, resulting in its high

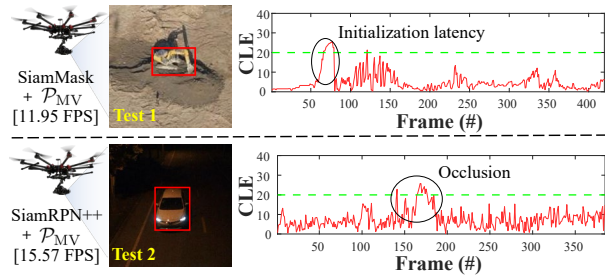


Figure 7. Real-world tests of PVT++. Thanks to PVT++, the non-real-time trackers work effectively under real-world challenges like object deformation in Test 1 and occlusion in Test 2.

efficiency (especially for the motion predictor).

Apart from the robustness priority, PVT++ is also easier to be deployed. Once trained, no further tuning is needed for PVT++ to fit various scenes. Differently, the noise matrix of the KFs is dependent on the environment, which is hard to tune and may not generalize well.

6.5. Real-World Tests

We further deploy SiamMask [57] (~ 11 FPS) and SiamRPN++_M [33] (~ 15 FPS) with PVT++ on a UAV with Nvidia Jetson AGX Xavier as onboard processor. The onboard speed and center location error (CLE) results are shown in Fig. 7. Despite that the original tracker is not real-time, our PVT++ framework can convert it into a predictive tracker and achieve a good result (CLE < 20 pixels) in real-world tracking. More tests see Appendix M and the video.

7. Conclusion

In this work, we present a simple end-to-end framework for latency-aware visual tracking, PVT++, which largely compensates for onboard latency. PVT++ integrates a lightweight predictor module that discovers the visual representation from pre-trained trackers for precise and robust future state estimation. To bridge the training-evaluation domain gap, we propose the relative motion factor, which yields a generalizable framework. In addition to PVT++, we introduce extended latency-aware evaluation benchmark (e-LAE), which assesses an *any-speed* tracker in the *online* setting. Extensive evaluations on robotics platform from the challenging aerial perspective show the effectiveness of our PVT++, which improves the *offline* tracker by up to **60%** in the *online* setting. Real-world tests are further conducted to exhibit the efficacy of PVT++ on physical robots.

Acknowledgments

This work is conducted when Bowen Li and Ziyuan Huang were visitors at Shanghai Qizhi Institute. This research is supported by the Agency for Science, Technology and Research (A*STAR) under its AME Programmatic Funding Scheme (Project #A18A2b0046).

References

- [1] Wilbert G Aguilar, Leandro Álvarez, Santiago Grijalva, and Israel Rojas. Monocular Vision-Based Dynamic Moving Obstacles Detection and Avoidance. In *Proceedings of the International Conference on Intelligent Robotics and Applications (ICIRA)*, pages 386–398, 2019. [1](#)
- [2] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional Siamese Networks for Object Tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 850–865, 2016. [2](#)
- [3] 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 (ICCV)*, pages 6181–6190, 2019. [7](#)
- [4] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual Object Tracking Using Adaptive Correlation Filters. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2544–2550, 2010. [2](#)
- [5] Rogerio Bonatti, Cherie Ho, Wenshan Wang, Sanjiban Choudhury, and Sebastian Scherer. Towards a Robust Aerial Cinematography Platform: Localizing and Tracking Moving Targets in Unstructured Environments. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 229–236, 2019. [1](#)
- [6] Ziang Cao, Changhong Fu, Junjie Ye, Bowen Li, and Yiming Li. HiFT: Hierarchical Feature Transformer for Aerial Tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15457–15466, 2021. [1](#), [2](#), [7](#)
- [7] Ziang Cao, Changhong Fu, Junjie Ye, Bowen Li, and Yiming 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)*, pages 3086–3092, 2021. [2](#), [7](#)
- [8] Ziang Cao, Ziyuan Huang, Liang Pan, Shiwei Zhang, Ziwei Liu, and Changhong Fu. TCTrack: Temporal Contexts for Aerial Tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2022. [1](#), [2](#)
- [9] Yu-Jia Chen, Deng-Kai Chang, and Cheng Zhang. Autonomous Tracking Using a Swarm of UAVs: A Constrained Multi-Agent Reinforcement Learning Approach. *IEEE Transactions on Vehicular Technology*, 69(11):13702–13717, 2020. [1](#)
- [10] 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 (CVPR)*, pages 6668–6677, 2020. [7](#)
- [11] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-End Tracking with Iterative Mixed Attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13608–13618, 2022. [7](#), [12](#), [13](#), [14](#)
- [12] 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 (CVPR)*, pages 4655–4664, 2019. [7](#)
- [13] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ECO: Efficient Convolution Operators for Tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6638–6646, 2017. [2](#)
- [14] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic Regression for Visual Tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7181–7190, 2020. [7](#)
- [15] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Learning Spatially Regularized Correlation Filters for Visual Tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4310–4318, 2015. [2](#)
- [16] Dawei Du, Yuankai Qi, Hongyang Yu, Yifan Yang, Kaiwen Duan, Guorong Li, Weigang Zhang, Qingming Huang, and Qi Tian. The Unmanned Aerial Vehicle Benchmark: Object Detection and Tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 370–386, 2018. [1](#), [2](#), [6](#), [7](#), [8](#), [12](#), [13](#), [14](#), [15](#), [17](#)
- [17] Matteo Dunnhofer, Antonino Furnari, G. Farinella, and C. Micheloni. Is First Person Vision Challenging for Object Tracking? The TREK-100 Benchmark Dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 300–317, 2020. [3](#)
- [18] 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 (CVPR)*, pages 5369–5378, 2019. [3](#), [6](#), [17](#)
- [19] Tharindu Fernando, Simon Denman, Sridha Sridharan, and Clinton Fookes. Tracking by Prediction: A Deep Generative Model for Multi-Person Localisation and Tracking. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1122–1132, 2018. [3](#)
- [20] Changhong Fu, Ziang Cao, Yiming Li, Junjie Ye, and Chen Feng. Siamese Anchor Proposal Network for High-Speed Aerial Tracking. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 510–516, 2021. [1](#), [2](#), [7](#)
- [21] Changhong Fu, Bowen Li, Fangqiang Ding, Fuling Lin, and Geng Lu. Correlation Filters for Unmanned Aerial Vehicle-Based Aerial Tracking: A Review and Experimental Evaluation. *IEEE Geoscience and Remote Sensing Magazine*, 10(1):125–160, 2022. [1](#), [6](#)
- [22] Dongyan Guo, Yanyan Shao, Ying Cui, Zhenhua Wang, Liyan Zhang, and Chunhua Shen. Graph Attention Tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9543–9552, 2021. [2](#), [7](#), [12](#), [13](#)
- [23] 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 (CVPR)*, pages 6269–6277, 2020. [2](#), [7](#)
- [24] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social Gan: Socially Acceptable Trajectories with Generative Adversarial Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2255–2264, 2018. [7](#), [12](#), [13](#)
- [25] Tuomas Haarnoja, Anurag Ajay, Sergey Levine, and Pieter Abbeel. Backprop KF: Learning Discriminative Deterministic State Estimators. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*, volume 29, 2016. [8](#), [13](#)
- [26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2018. [6](#), [12](#)
- [27] 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, 2015. [2](#)
- [28] Lianghua Huang, Xin Zhao, and Kaiqi Huang. GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019. [3](#), [6](#), [17](#)
- [29] Ziyuan Huang, Changhong Fu, Yiming Li, Fuling Lin, and Peng Lu. Learning Aberrance Repressed Correlation Filters for Real-Time UAV Tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2891–2900, 2019. [1](#), [2](#)
- [30] Rudolph Emil Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME-Journal of Basic Engineering*, pages 35–45, 1960. [2](#), [3](#), [4](#), [8](#), [12](#)
- [31] Bowen Li, Changhong Fu, Fangqiang Ding, Junjie Ye, and Fuling Lin. ADTrack: Target-Aware Dual Filter Learning for Real-Time Anti-Dark UAV Tracking. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 496–502, 2021. [3](#)
- [32] Bowen Li, Yiming Li, Junjie Ye, Changhong Fu, and Hang Zhao. Predictive Visual Tracking: A New Benchmark and Baseline Approach. *arXiv preprint arXiv:2103.04508*, pages 1–8, 2021. [2](#), [3](#), [4](#), [8](#), [13](#), [14](#)
- [33] 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 (CVPR)*, pages 4277–4286, 2019. [1](#), [2](#), [5](#), [6](#), [7](#), [8](#), [12](#), [13](#), [17](#), [18](#)
- [34] 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, 2018. [1](#), [2](#), [6](#), [7](#)
- [35] Fan Li, Changhong Fu, Fuling Lin, Yiming Li, and Peng Lu. Training-Set Distillation for Real-Time UAV Object Tracking. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 9715–9721, 2020. [2](#)
- [36] Mengtian Li, Yu-Xiong Wang, and Deva Ramanan. Towards Streaming Perception. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 473–488, 2020. [2](#), [3](#), [4](#), [8](#), [13](#), [14](#)
- [37] Siyi Li and Dit-Yan Yeung. Visual Object Tracking for Unmanned Aerial Vehicles: A Benchmark and New Motion Models. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 4140–4146, 2017. [2](#), [4](#), [6](#), [7](#), [8](#), [12](#), [13](#), [14](#), [15](#)
- [38] Yiming Li, Changhong Fu, Fangqiang Ding, Ziyuan Huang, and Geng Lu. AutoTrack: Towards High-Performance Visual Tracking for UAV with Automatic Spatio-Temporal Regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11920–11929, 2020. [1](#), [2](#)
- [39] Yiming Li, Changhong Fu, Ziyuan Huang, Yinqiang Zhang, and Jia Pan. Keyfilter-Aware Real-Time UAV Object Tracking. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 193–199, 2020. [2](#)
- [40] Junwei Liang, Lu Jiang, Juan Carlos Niebles, Alexander G Hauptmann, and Li Fei-Fei. Peeking into the Future: Predicting Future Person Activities and Locations in Videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5725–5734, 2019. [7](#), [12](#), [13](#)
- [41] Ming Liang, Bin Yang, Wenyuan Zeng, Yun Chen, Rui Hu, Sergio Casas, and Raquel Urtasun. PnPNet: End-to-End Perception and Prediction With Tracking in the Loop. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11553–11562, 2020. [3](#)
- [42] Ningxin Liang, Guile Wu, Wenxiong Kang, Zhiyong Wang, and David Dagan Feng. Real-Time Long-Term Tracking with Prediction-Detection-Correction. *IEEE Transactions on Multimedia*, 20(9):2289–2302, 2018. [3](#)
- [43] Qiao Liu, Zhenyu He, Xin Li, and Yuan Zheng. PTB-TIR: A Thermal Infrared Pedestrian Tracking Benchmark. *IEEE Transactions on Multimedia*, 22:666–675, 2020. [3](#)
- [44] Yuan Liu, Ruoteng Li, Yu Cheng, Robby T Tan, and Xiubao Sui. Object Tracking using Spatio-Temporal Networks for Future Prediction Location. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 1–17, 2020. [3](#)
- [45] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–18, 2018. [13](#)
- [46] Wenjie Luo, Bin Yang, and Raquel Urtasun. Fast and Furious: Real Time End-to-End 3d Detection, Tracking and Motion Forecasting with a Single Convolutional Net. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3569–3577, 2018. [3](#)
- [47] Matthias Mueller, Neil Smith, and Bernard Ghanem. A Benchmark and Simulator for UAV Tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 445–461, 2016. [2](#), [3](#), [6](#), [8](#), [12](#), [14](#), [15](#), [16](#)

- [48] M. Müller, Adel Bibi, Silvio Giancola, Salman Al-Subaihi, and Bernard Ghanem. TrackingNet: A Large-Scale Dataset and Benchmark for Object Tracking in the Wild. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 300–317, 2018. [3](#)
- [49] Yuya Nishida, Takashi Sonoda, Shinsuke Yasukawa, Kazunori Nagano, Mamoru Minami, Kazuo Ishii, and Tamaki Ura. Underwater Platform for Intelligent Robotics and its Application in Two Visual Tracking Systems. *Journal of Robotics and Mechatronics*, 30(2):238–247, 2018. [1](#)
- [50] Nicola A Piga, Ugo Pattacini, and Lorenzo Natale. A Differentiable Extended Kalman Filter for Object Tracking Under Sliding Regime. *Frontiers in Robotics and AI*, 8:686447, 2021. [8](#), [13](#)
- [51] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrilă, and Kai O Arras. Human Motion Trajectory Prediction: A Survey. *The International Journal of Robotics Research*, 39(8):895–935, 2020. [2](#), [3](#)
- [52] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. [4](#), [6](#), [12](#), [17](#)
- [53] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018. [6](#), [12](#)
- [54] Gur-Eyal Sela, Ionel Gog, Justin Wong, Kumar Krishna Agrawal, Xiangxi Mo, Sukrit Kalra, Peter Schafhalter, Eric Leong, Xin Wang, Bharathan Balaji, et al. Context-Aware Streaming Perception in Dynamic Environments. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 621–638, 2022. [3](#)
- [55] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. [6](#), [12](#)
- [56] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer Meets Tracker: Exploiting Temporal Context for Robust Visual Tracking. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1571–1580, 2021. [7](#)
- [57] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip HS Torr. Fast Online Object Tracking and Segmentation: A Unifying Approach. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1328–1338, 2019. [2](#), [6](#), [7](#), [8](#), [12](#), [13](#), [17](#), [18](#)
- [58] 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 (AAAI)*, pages 12549–12556, 2020. [2](#), [6](#), [7](#)
- [59] Jinrong Yang, Songtao Liu, Zeming Li, Xiaoping Li, and Jian Sun. Real-time Object Detection for Streaming Perception. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2022. [3](#)
- [60] Botao Ye, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Joint Feature Learning and Relation Modeling for Tracking: A One-Stream Framework. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 341–357, 2022. [7](#), [12](#), [13](#), [14](#)
- [61] Junjie Ye, Changhong Fu, Fuling Lin, Fangqiang Ding, Shan An, and Geng Lu. Multi-Regularized Correlation Filter for UAV Tracking and Self-Localization. *IEEE Transactions on Industrial Electronics*, 69(6):6004–6014, 2022. [1](#)
- [62] Junjie Ye, Changhong Fu, Guangze Zheng, Danda Pani Paudel, and Guang Chen. Unsupervised Domain Adaptation for Nighttime Aerial Tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8896–8905, 2022. [3](#)
- [63] Pu Zhang, Wanli Ouyang, Pengfei Zhang, Jianru Xue, and Nanning Zheng. SR-LSTM: State Refinement for LSTM towards Pedestrian Trajectory Prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12085–12094, 2019. [7](#), [12](#), [13](#)
- [64] 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)*, 2018. [2](#), [6](#)