AutoTrack: Towards High-Performance Visual Tracking for UAV with Automatic Spatio-Temporal Regularization

Yiming Li†, Changhong Fu†,* , Fangqiang Ding†, Ziyuan Huang†, and Geng Lu§
†Tongji University  ‡National University of Singapore  §Tsinghua University
yimingli9702@gmail.com, changhongfu@tongji.edu.cn, lug@tsinghua.edu.cn

Abstract

Most existing trackers based on discriminative correlation filters (DCF) try to introduce predefined regularization term to improve the learning of target objects, e.g., by suppressing background learning or by restricting change rate of correlation filters. However, predefined parameters introduce much effort in tuning them and they still fail to adapt to new situations that the designer did not think of. In this work, a novel approach is proposed to online automatically and adaptively learn spatio-temporal regularization term. Spatially local response map variation is introduced as spatial regularization to make DCF focus on the learning of trust-worthy parts of the object, and global response map variation determines the updating rate of the filter. Extensive experiments on four UAV benchmarks have proven the superiority of our method compared to the state-of-the-art CPU- and GPU-based trackers, with a speed of ∼60 frames per second running on a single CPU.

Our tracker is additionally proposed to be applied in UAV localization. Considerable tests in the indoor practical scenarios have proven the effectiveness and versatility of our localization method. The code is available at https://github.com/vision4robotics/AutoTrack.

1. Introduction

Visual object tracking is one of the fundamental tasks in the computer vision community, aiming to localize the object sequentially only with the information given in the first frame. Endowing unmanned aerial vehicle (UAV) with visual tracking capability brings many applications, e.g., aerial cinematography [1], person following [2], aircraft tracking [3], and traffic patrolling [4].

There are currently two main research interests in this area: discriminative correlation filter (DCF)-based methods [5, 6, 7, 8, 9, 10, 11, 12, 13, 14] as well as deep learning-based approaches [15, 16, 17, 18, 19, 20]. In consideration of the limitation of power capacity and computational resources onboard UAVs, DCF framework is selected because of its high efficiency originating from calculation in the Fourier domain.

To improve DCF-based trackers, there are currently three directions: a) building more robust appearance model [18, 17, 20, 21], b) mitigating boundary effect or imposing restrictions in learning [8, 22, 17, 14, 23], and c) mitigating filter degradation [24, 12, 14, 25]. Robust appearance can indeed boost performance, yet it leads to burdensome calculations. Filter degradation, on the other hand, is not improving it fundamentally. Most trackers try to improve performance using option b) by introducing regularization terms.

Recently, some attentions have been brought to using response maps generated in the detection phase to form the restrictions in learning [26]. The intuition behind it is
that the response map contains crucial information regarding the resemblance of current object and the appearance model. However, [26] only exploits what we call the spatially global response map variations, while ignoring local response variation indicating credibility at different locations in the image: drastic local variation means low credibility and vice versa.

We fully exploit the local-global response variation to train our tracker with automatic spatio-temporal regularization, \textit{i.e.}, AutoTrack. While most parameters in regularization terms proposed by others are hyper-parameters that require large effort to tune, and would have a difficult time adjusting to new situations that the designers did not think of, we propose to learn some of the hyper-parameters automatically and adaptively. AutoTrack performs favorably against the state-of-the-art trackers, while running at \(\sim 60\) frames per second (fps) on a single CPU.

Our main contributions are summarized as follows:

- We propose a novel spatio-temporal regularization term to simultaneously exploit local and global information hidden in response maps.
- We develop a novel DCF-based tracker which can automatically tune the hyper-parameters of spatio-temporal regularization term on the fly.
- We evaluate our tracker on 278 difficult UAV image sequences, and the evaluations have validated the state-of-the-art performance of our tracker compared to current CPU- and GPU-based trackers.
- We introduce a novel application of visual object tracking in UAV localization and prove its effectiveness as well as generality in the practical scenarios.

2. Related Works

\textbf{Tracking by detection:} tracking-by-detection framework, which regards the tracking as a classification problem, is widely adopted in UAV [5, 6, 7, 27, 28]. Among them, DCF has exhibited good performance with exceptional efficiency. The speed of traditional DCF-based trackers [7, 5, 29] is around hundreds of fps on a single CPU, far exceeding the real-time requirement of UAV (30 fps). Yet they are primarily subjected to the following issues.

a) Boundary effect: the circulant samples suffer from periodical splicing at the boundary, reducing filters’ discriminative power. Several works can mitigate boundary effect [22, 8, 14, 30], but they used a constant spatial penalization which cannot adapt to various changes in different objects. K. Dai \textit{et al}. optimized the spatial regularization in the temporal domain [31]. Different to [31], we exploit the inherent information in DCF framework, so our method is more generic. Also, we have achieved better performance in the aerial scenarios in terms of speed and precision.

b) Filter degradation: the appearance model updated via a linear interpolation method cannot adapt to ubiquitous appearance change, leading to filter degradation. Some attempts are made to tackle the issue, \textit{e.g.}, training set management [20, 24, 32], temporal restriction [14, 25], tracking confidence verification [23, 12] and over-fitting alleviation [33]. Amongst them the temporal regularization is an effective and efficient way. Yet the non-adaptive regularization is prone to tracking drift once the filter is corrupted.

\textbf{Tracking by deep learning:} recently, deep learning-based tracking has caught wide attention due to its robustness, \textit{e.g.}, deep feature representation [18, 34, 17, 20], reinforcement learning [16], residual learning [35] and adversarial learning [36]. However, for mobile robots, the above trackers cannot meet the requirement of real-time perception even with a high-end GPU. Currently, the state-of-the-art deep trackers [14, 37, 38, 39, 40, 41] are mostly built on siamese neural network [15]. The pre-trained siamese trackers just need to traverse in a feed-forward way to get a similarity score for object localization, facilitating real-time implementation on GPU. However, on a mobile device solely with CPU, the speed of siamese-based trackers cannot satisfy the real-time needs. C. Huang \textit{et al}. proposed a CPU-friendly deep tracker [42] by training an agent working in a cascaded manner. It can run at near real-time speed by reducing calculation on easy frames. In summary, deep trackers can hardly meet real-time demands on CPU.

\textbf{Vision-based localization:} vision-based localization is crucial for UAV especially in GPS-denied environments. A. Breitenmoser \textit{et al}. developed a monocular 6D pose estimation system based on passive markers in the visible spectrum [43]. However, it performs worse in low-light environments. M. Faessler \textit{et al}. presented a monocular localization system based on infrared LEDs to raise robustness in cluttered environments [44]. Its generality, however, is limited since the system can only work in the infrared spectrum. Built on [44], we develop a localization system based on visual tracking. In light of robustness and generality of our tracker in various scenarios like illumination variation, occlusion and deformation, our localization system is more versatile compared to the infrared LED-based one [44].

3. Revisit STRCF

In this section, our baseline STRCF [14] is revisited. The optimal filter \(H_t\) in frame \(t\) is learned by minimizing the following objective function:

\[
\mathcal{E}(H_t) = \frac{1}{2} \|y - \sum_{k=1}^{K} x_t^k \odot h_t^k\|_2^2 + \frac{1}{2} \sum_{k=1}^{K} \|u \odot h_t^k\|_2^2
\]

\[+ \frac{\beta}{2} \sum_{k=1}^{K} \|h_t^k - h_{t-1}^k\|_2^2,\]

where \(x_t^k \in \mathbb{R}^{T \times 1} (k = 1, 2, 3, ..., K)\) is the extracted feature with length \(T\) in frame \(t\), and \(K\) denotes number of
channel, $y \in \mathbb{R}^{T \times 1}$ is the desired Gaussian-shaped response. $h^{t}_{1}, h^{t}_{2}, ..., h^{t}_{t-1} \in \mathbb{R}^{T \times 1}$ respectively denote the filter of the $t$-th channel trained in the $t$-th and $(t-1)$-th frame, ⊙ indicates the convolution operator. Noted that $H^{t}_{t} = [h^{t}_{1}, h^{t}_{2}, h^{t}_{3}, ..., h^{t}_{t}]$. As for regularization, the spatial regularization parameter $u \in \mathbb{R}^{T \times 1}$ is bowl-shaped and borrowed from SRDCF [8] for decreasing boundary effect, and temporal regularization, i.e., the third term in Eq. 1, is firstly proposed to restrict filter’s variation by penalizing the difference between the current and previous filters.

Although STRCF [14] has achieved competent performance, it does have two limitations: a) the fixed spatial regularization failing to address appearance variation in the unforeseeable aerial tracking scenarios, b) the unchanged temporal penalty strength $\theta$ (set as 15 in [14]) which is not general in all kinds of situations.

4. Automatic Spatio-Temporal Regularization

In this work, both local and global response variations are fully utilized to achieve simultaneous spatial and temporal regularizations, as well as automatic and adaptive hyper-parameter optimization.

4.1. Response Variation

First of all, we define local response variation vector $\Pi = [\Pi^{1}, \Pi^{2}, ..., \Pi^{T}]$, as can be seen in Fig. 1 for its 2D visualization in the object bounding box, in preparation for spatial regularization. Its $i$-th element $|\Pi^{i}|$ is defined as:

$$\Pi^{i} = \frac{R^{i}_{t} [\psi^{(i)}_{\Delta}] - R^{i}_{t-1}}{R^{i}_{t-1}}, \quad (2)$$

where $[\psi^{(i)}_{\Delta}]$ is the shift operator to make two peaks in two response maps $R^{i}_{t}$ and $R^{i}_{t-1}$ coincide with each other, in order for removing the motion influence [26]. $R^{i}$ denotes the $i$-th element in response map $\mathcal{R}$.

**Automatic spatial regularization:** local response variation reveals the credibility of every pixel in the search area of the current frame. Therefore, filters located where the pixel credibility is low should be restricted in learning. We achieve this by introducing local variation $\Pi$ to the spatial regularization parameter $u$:

$$\hat{u} = \mathbf{P}^{T} \delta \log(\Pi + 1) + u, \quad (3)$$

where $\mathbf{P}^{T} \in \mathbb{R}^{T \times T}$ is used to crop the central part of the filter where the object is located. $\delta$ is a constant to adjust the weight of local response variations, and $u$ is inherited from STRCF [14] to mitigate boundary effects. Through Eq. 3, filters located at pixels with dramatic response variation will be partially reframed from learning the new appearance because of the spatial punishment.

**Automatic temporal regularization:** in STRCF [14], the change rate of filters between two frames is punished in the loss by a fixed parameter $\theta$. AutoTrack tries to adaptively and automatically determine the value of this hyper-parameter by jointly optimization of its value and the filter. So we define a reference $\hat{\theta}$ in preparation for the objective function with regard to the global response:

$$\hat{\theta} = \frac{\zeta}{1 + \log(u^{(i)}\Pi^{i} \| 2 + 1), \quad (4)$$

where $\zeta$ and $\nu$ denote hyper parameters. When the global variation is higher than the threshold $\phi$, it means that there are aberrances in response maps [26], so correlation filter ceases to learn. If it is lower than the threshold, the more dramatic the response map varies, the smaller the reference value will be, so that the restriction on temporal change of the correlation filters can be loosened and it can learn more rapidly in situations like large appearance variations.

**Remark 1:** Note that what we defined here is the reference value rather than the hyper-parameter itself. For the hyper-parameter of the temporal regularization, we use joint optimization to online estimate the value of it, so that the restriction can be online adaptively adjusted according to the response map variations. When appearance changes drastically, correlation filter learns more rapidly and vice versa.

4.2. Objective Optimization

Our objective function for joint optimization of filter as well as temporal regularization term can be written as:

$$\mathcal{E}(H^{t}_{t}, \theta^{t}_{t}) = \frac{1}{2} \parallel y - \sum_{k=1}^{K} x^{k}_{t} \odot h^{k}_{t} \parallel^{2} + \frac{1}{2} \sum_{k=1}^{K} \parallel \hat{u} \odot h^{k}_{t} \parallel^{2} \quad (5)$$

$$+ \frac{\theta_{t}}{2} \sum_{k=1}^{K} \parallel h^{k}_{t} - h^{k}_{t-1} \parallel^{2} + \frac{1}{2} \parallel \theta_{t} - \hat{\theta}_{t} \parallel^{2}$$

where $\hat{\theta}$ and $\theta_{t}$ respectively denote the reference and optimized temporal regularization parameter, and $\hat{u}$ represents the automatic spatial regularization calculated via Eq. 3.

For optimization, we introduce an auxiliary variable $\mathbf{g}_{t}$ by ordering $\mathbf{g}_{t} = \sqrt{T} \mathbf{F}h_{t}(\mathbf{G} = [h^{1}_{t}, h^{2}_{t}, h^{3}_{t}, ..., h^{K}_{t}])$ where $\mathbf{F} \in \mathbb{C}^{T \times T}$ denotes the orthonormal matrix and the symbol $\ast$ denotes the discrete Fourier transform (DFT) of a signal. Then Eq. 5 is converted into the frequency domain:

$$\mathcal{E}(H^{t}_{t}, \theta^{t}_{t}, \mathbf{G}_{t}) = \frac{1}{2} \parallel y - \sum_{k=1}^{K} x^{k}_{t} \odot \mathbf{g}^{k}_{t} \parallel^{2} + \frac{1}{2} \sum_{k=1}^{K} \parallel \hat{u} \odot h^{k}_{t} \parallel^{2} \quad (6)$$

$$+ \frac{\theta_{t}}{2} \sum_{k=1}^{K} \parallel \mathbf{g}^{k}_{t} - \mathbf{g}^{k}_{t-1} \parallel^{2} + \frac{1}{2} \parallel \theta_{t} - \hat{\theta}_{t} \parallel^{2}$$

By minimizing Eq. 6, an optimal solution can be obtained through alternating direction method of multipliers (ADMM) [45]. The Augmented Lagrangian form of equa-
tion Eq. 6 can be formulated as:

\[
\mathcal{L}_t(H_t, \theta_t, \tilde{G}_t, \tilde{M}_t) = \mathcal{E}(H_t, \theta_t, \tilde{G}_t) + \frac{\gamma}{2} \sum_{k=1}^{K} \|g_{k}^t - \sqrt{T}Fh_{k}^t\|^2
\]

\[
+ \sum_{k=1}^{K} (g_{k}^t - \sqrt{T}Fh_{k}^t)^\top m_{k}^t,
\]

where \(\tilde{M}_t = [\tilde{m}_1, \tilde{m}_2, ..., \tilde{m}_K] \in \mathbb{R}^{T \times K}\) is the Fourier transform of the Lagrange multiplier and \(\gamma\) denotes the step size regularization parameter. By assigning \(\gamma_k = m_k^t (v_k^t = [v_1^t, v_2^t, ..., v_K^t])\), Eq. 7 can be reformulated as:

\[
\mathcal{L}_t(H_t, \theta_t, \tilde{G}_t, \tilde{V}_t) = \mathcal{E}(H_t, \theta_t, \tilde{G}_t)
\]

\[
+ \frac{\gamma}{2} \sum_{k=1}^{K} \|g_{k}^t - \sqrt{T}Fh_{k}^t + \tilde{v}_{k}^t\|^2.
\]

Then we solve the following subproblems by ADMM.

**Subproblem G:** given \(H_t, \theta_t, \tilde{V}_t\), the optimal \(G^t\) is:

\[
\hat{G}^t = \arg\min_{G} \frac{1}{2} \|\tilde{y} - \sum_{k=1}^{K} \tilde{X}_k^t \odot \tilde{g}_k^t\|^2
\]

\[
+ \frac{\theta_t}{2} \sum_{k=1}^{K} \|g_{k}^t - \tilde{g}_{k}^t - \tilde{v}_{k}^t\|^2 + \frac{\gamma}{2} \sum_{k=1}^{K} \|g_{k}^t - \sqrt{T}Fh_{k}^t + \tilde{v}_{k}^t\|^2.
\]

Solving Eq. 9 directly is very difficult because of its complexity. So we decide to sample \(\tilde{X}_k\) across all \(K\) channels in each pixel to simplify our formulation written by:

\[
I_j^g(\tilde{G}_t) = \arg\min_{I_j^g(\tilde{G}_t)} \|\hat{y}_j - I_j(\tilde{X}_t)^\top I_j(\tilde{G}_t)\|^2
\]

\[
+ \gamma \|I_j(\tilde{G}_t) + I_j(\tilde{V}_t) - I_j(\sqrt{T}Fh_{k}^t)\|^2
\]

\[
+ \theta_t \|I_j(\tilde{G}_t) - I_j(\tilde{G}_{t-1})\|^2,
\]

where \(I_j(\tilde{X}_t) \in \mathbb{C}^{K \times 1}\) represents the vector containing values of all \(K\) channels of \(\tilde{X}\) on pixel \(j (j = 1, 2, ..., T)\). After derivation using Sherman Morrison formula, we can obtain its solution:

\[
I_j^g(\tilde{G}_t) = \frac{1}{\gamma + \theta_t} (I - \frac{I_j(\tilde{X}_t)I_j(\tilde{X}_t)^\top}{\theta_t + \gamma + I_j(\tilde{X}_t)^\top \nu_j(\tilde{X}_t)}) \rho.
\]

where the vector \(\rho\) takes the form \(\rho = I_j(\tilde{X}_t)\tilde{y}_j + \theta_t I_j(\tilde{G}_t)I_j(\tilde{V}_t) + \gamma I_j(\sqrt{T}Fh_{k}^t)\) for presentation.

**Subproblem H:** given \(\theta_t, \tilde{G}_t, \tilde{V}_t\), we can optimize \(h_{k}^t\) by:

\[
\arg\min_{h_{k}^t} \frac{1}{2} \|\tilde{u} \odot h_{k}^t\|^2 + \gamma \|g_{k}^t - \sqrt{T}Fh_{k}^t + \tilde{v}_{k}^t\|^2.
\]

The closed-form solution of \(h_{k}^t\) can be written by:

\[
h_{k}^t = (\tilde{U}^\top \tilde{U} + \gamma T I)^{-1} \gamma T (v_{k}^t + g_{k}^t).
\]

where \(\tilde{U} = \text{diag}(\tilde{u}) \in \mathbb{R}^{T \times T}\) represents diagonal matrix.

**Subproblem \(\theta_t\):** given other variables in Eq. 8, the optimal solution of \(\theta_t\) can be determined as:

\[
\theta_t^* = \arg\min_{\theta_t} \frac{1}{2} \sum_{k=1}^{K} \|g_{k}^t - \tilde{g}_{k-1}\|^2 + \frac{1}{2} \|\theta_t - \tilde{\theta}\|^2,
\]

\[
\tilde{\theta} = \frac{1}{2} \sum_{k=1}^{K} \|g_{k}^t - \tilde{g}_{k-1}\|^2.
\]

**Lagrange multiplier update:** after solving three subproblems above, we can update Lagrangian multipliers as:

\[
\tilde{V}^{t+1} = \tilde{V} + \gamma (\tilde{G}^{t+1} - \tilde{H}^{t+1}),
\]

where \(i\) and \(i + 1\) denotes the iteration index and the step size regularization constant \(\gamma\) (initially equals to 1) takes the form of \(\gamma = \min (\gamma_{max}, \beta \gamma^{i})\). (\(\beta = 10, \gamma_{max} = 10000\))

By iteratively solving the four subproblems above, we can optimize our objective function effectively and obtain the optimal filter \(\tilde{G}_t\) and temporal regularization parameter \(\theta_t\) in frame \(t\). Then \(\tilde{G}_t\) is used for detection in frame \(t + 1\).

4.3. Object Localization

The tracked object is localized by searching for the maximum value of response map \(R_t\) calculated by:

\[
R_t = \mathcal{F}^{-1} \sum_{k=1}^{K} (\tilde{X}_k^t \odot \tilde{g}_{k-1}^t).
\]

where \(R_t\) is the response map in frame \(t\), \(\mathcal{F}^{-1}\) denotes the inverse Fourier transform (IFT) operator and \(\tilde{X}_k^t\) represents the Fourier form of extracted feature map in frame \(t\).

5. Localization by Tracking

Self-localization for UAV is essential for autonomous navigation. To develop a robust and universal localization system in dynamic and uncertain environments, we introduce visual object tracking into UAV localization for the first time. Specifically, we utilize the open-source software in [44], but employ AutoTrack to track four objects simultaneously instead of segmenting LEDs in the infrared spectrum. The main workflow is briefly described below.

**Prerequisites:** the system requires the knowledge of four object configuration (non-symmetric), i.e., their positions in the world coordinate (observed in motion capture system), and intrinsic UAV-mounted camera parameters.

**Initialization and tracking:** after manually assigning four objects, AutoTrack starts to track them independently and output their location in the RGB image. Different to the system [44] only applicable in infrared spectrum, our system can be used in versatile environments.

**Correspondence search and pose optimization:** correspondence between the tracked object configuration in the world coordinate and tracked results in image frames is firstly clarified, then the final 6D pose is optimized by fine-tuning the reprojection error [44].
6. Experiments

In this section, we firstly evaluate the tracking performance of AutoTrack with current state-of-the-art trackers on four difficult UAV benchmarks [46, 47, 48, 49]. Then, the proposed localization system is evaluated on Quanser® platform in the indoor practical scenarios. The experiments of tracking performance evaluation are conducted using MATLAB R2018a on a PC with an i7-8700K processor (3.7GHz), 32GB RAM and NVIDIA GTX 2080 GPU. The tests of localization system are run on ROS [50] using C++. For the hyper parameters of AutoTrack, we set $\delta = 0.2$, $\nu = 2 \times 10^{-5}$, $\zeta = 13$. The threshold of $\phi$ is 3000, ADMM iteration is set to 4. The sensitivity analysis of all the parameters can be found in the supplementary material.

6.1. Evaluation on UAV Datasets

For rigorous and comprehensive evaluation, the comparison between AutoTrack with the state-of-the-art methods is reported on four challenging and authoritative UAV benchmarks: DTB70 [46], UAVDT [47], UAV123@10fps [48] and VisDrone2018-test-dev [49], with a total number of 119,830 frames. Noted that we use the same evaluation criteria with the four benchmarks [46, 47, 48, 49].

6.1.1 Comparison with deep-based trackers

**DTB70:** DTB70 [46], composed of 70 difficult UAV image sequences, primarily addresses the problem of severe UAV motion. In addition, various cluttered scenes and objects with different sizes as well as aspect ratios are included. We compare AutoTrack with nine state-of-the-art deep trackers, i.e., ASRCF [31], TADT [51], HCF [18], ADNet [16], CFNet [52], UDT+ [53], IBCCF [54], MDNet [55], MCPF [19], on DTB70, and the final results are reported in Fig. 2. Only with hand-crafted features, AutoTrack outperforms deep feature-based trackers (ASRCF [31], HCF [18], MCPF [19] and IBCCF [54]) and pre-trained deep architecture-based trackers, i.e., MDNet [55], ADNet [16], UDT+ [53] and CFNet [52]. In summary, AutoTrack exhibits strong robustness against drastic UAV motion without losing efficiency, and also demonstrates a generality in tracking different objects in various scenes.

**UAVDT:** UAVDT [47] mainly emphasizes vehicle tracking in various scenarios. Weather condition, flying altitude and camera view are three categories addressed by UAVDT. Compared to deep trackers including ASRCF [31], TADT [51], SiameseFC [15], DSiam [56], MCCT [13], ADNet [16], CFNet [52], DeepSTRCF [14], UDT+ [53], HCF [18], C-COT [17], ECO [20], IBCCF [54], MCPF [19] and CREST [35], AutoTrack with a single CPU exhibits the best performance in terms of precision and speed, as shown in Table 1. In a word, AutoTrack has extraordinary performance in vehicle tracking despite omnipresent challenges.

6.1.2 Comparison with CPU-based trackers

Twelve real-time trackers (with a speed of $>30$fps), i.e., KCF [7], DCF [7], KCC [57] fDSST [11], DSST [58], BACF [22], STAPLE-CA [10], STAPLE [10], MCCT-H [13], STRCF [14], ECO-HC [20], ARCF-H [26], and five non-real-time ones, i.e., SRDCF [8], SAML [6], CSR-DCF [30], SRDCFdecon [24], ARCF-HC [26] are used for comparison. The results of real-time trackers on four datasets are displayed in Fig. 3. Besides, the average performance of top ten CPU-based trackers in terms of speed and precision is demonstrated in the Table 2. It can be seen that AutoTrack is the best real-time tracker on CPU. Some tracking results are demonstrated in Fig. 4 and Fig. 6.

**Overall performance evaluation:** AutoTrack has outperformed all the CPU-based real-time trackers in both precision and success rate on DTB70 [46], UAVDT [47] and UAV123@10fps [48]. On VisDrone2018-test-dev [49], AutoTrack achieves comparable performance with the best tracker MCCT-H and ECO-HC in terms of precision and success rate. As for the average performance of top ten trackers, AutoTrack w...
CPU-based trackers, AutoTrack has the best performance in precision with the second fast speed of 59.2fps, only slower than ECO-HC (69.5fps), however, we have achieved an average improvement of 4.8% in precision compared to ECO-HC. Moreover, AutoTrack has an advantage of 7.9% in precision and 108.5% in speed over the baseline STRCF.

Remark 2: M. Muller et al. created a 10fps dataset from the recorded 30fps one [48], thus the movement of tracked object between successive frames is larger, bringing more challenges. On UAV123@10fps, AutoTrack achieves a remarkable advantage of 5.8% in precision than the second best ECO-HC, proving its robustness against large motion.

Remark 3: Compared to ARCF-HC [26] solely repressing the global response variation using a fixed parameter, we fully utilize the local-global information to fine-tune the spatio-temporal regularization term in an automatic manner. Extensive experiments have shown that AutoTrack achieves better performance while providing a much faster speed which is 3.1 times that of ARCF-HC.

Attribute-based evaluation: Success plots of eight attributes are exhibited in Fig. 5. In the normal appearance change scenarios (deformation, in-plane-rotation, viewpoint change), AutoTrack improves STRCF by 15.9%, 15.5% and 4.6% in success rate because the automatic temporal regularization can smoothly help filter adapt to new appearance. In illumination variation and large occlusion (aberrant appearance variation), AutoTrack has a superiority of 7.0% and 15.7% compared to STRCF in light of adaptive spatial regularization as well as aberrance monitoring mechanism which can stop training before contamination.
6.1.3 Ablation study

To validate the effectiveness of our method, AutoTrack is compared to itself with different modules enabled. The overall evaluation is presented in Table 3. With each module (automatic spatial regularization ASR, automatic temporal regularization ATR) added to the STRCF, the performance is smoothly improved. It is noted that ATR can also bring a gain in speed compared to ASR because we can reduce meaningless and detrimental training on contaminated samples. In addition, response maps of some frames are illustrated in Fig. 4. It can be clearly seen that response of our method is more reliable than that of baseline.

Table 3. Ablation study of AutoTrack. ASR and ATR respectively represent automatic spatial and temporal regularization.

<table>
<thead>
<tr>
<th>Tracker</th>
<th>Precision</th>
<th>AUC</th>
<th>FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRCF</td>
<td>0.671</td>
<td>0.468</td>
<td>28.4</td>
</tr>
<tr>
<td>STRCF + ASR</td>
<td>0.716</td>
<td>0.489</td>
<td>53.7</td>
</tr>
<tr>
<td>STRCF + ATR</td>
<td>0.714</td>
<td>0.492</td>
<td>60.0</td>
</tr>
<tr>
<td>AutoTrack</td>
<td>0.724</td>
<td>0.495</td>
<td>59.2</td>
</tr>
</tbody>
</table>

6.2. Evaluation of Localization System

We evaluate our localization system on six datasets covering 2,666 images, and in each dataset, the camera is moving at a distinct trajectory as UAV flies. The image is captured with a resolution of 1280 × 720 pixels at 10fps, using Intel RealSense (R200) camera looking ahead to perform building inspection, as shown in Fig. 7.

We adopt the UAV location in motion capture system

![Figure 7. Experiment setup (left) and view from the UAV-mounted camera (right). The tracked objects (reflective markers whose ground truth locations are known in Quanser motion-capture system) for UAV localization are denoted as four green rectangles.](image-url)
Figure 8. Estimation of camera position and the respective errors on six datasets. Lines with red, green and blue color denote x, y and z positions, respectively. The ground truth is not displayed because there is no noticeable differences with our results at such scale.

as the ground truth. The mean position errors in x, y and z directions are reported in Table 4. Figure 8 exhibits the estimated position as well as respective error in every frame. The root-mean-square error (RMSE) of our method on 2,666 frames is 3.44 centimeters.

**Remark 4:** It is noted that our system is applicable in various scenarios because our tracker can track any arbitrary objects once given their information in the first frame. In summary, compared to LED-based localization system [44], our method is more versatile and can run at real-time frame rates in the real-world scenarios.

Table 4. Illustration of estimation errors on six datasets covering 2,666 frames. The dataset is in line with the (a)-(f) in Fig. 8.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>x(cm)</th>
<th>y(cm)</th>
<th>z(cm)</th>
<th>RMSE</th>
<th>Frame number</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>1.90</td>
<td>2.25</td>
<td>2.38</td>
<td>3.79</td>
<td>652</td>
</tr>
<tr>
<td>(b)</td>
<td>1.06</td>
<td>1.88</td>
<td>1.13</td>
<td>2.44</td>
<td>431</td>
</tr>
<tr>
<td>(c)</td>
<td>3.01</td>
<td>3.51</td>
<td>1.30</td>
<td>4.80</td>
<td>400</td>
</tr>
<tr>
<td>(d)</td>
<td>1.01</td>
<td>1.73</td>
<td>1.16</td>
<td>2.32</td>
<td>381</td>
</tr>
<tr>
<td>(e)</td>
<td>3.77</td>
<td>1.77</td>
<td>1.05</td>
<td>4.30</td>
<td>352</td>
</tr>
<tr>
<td>(f)</td>
<td>2.27</td>
<td>2.77</td>
<td>0.91</td>
<td>3.69</td>
<td>450</td>
</tr>
<tr>
<td>Average</td>
<td>2.17</td>
<td>2.32</td>
<td>1.32</td>
<td>3.44</td>
<td>444</td>
</tr>
</tbody>
</table>

7. Conclusion

In this work, a generally applicable automatic spatio-temporal regularization framework is proposed for high-performance UAV tracking. Local response variation indicates local credibility, thus restricting local correlation filter learning. Global variation is able to control how much the correlation filter learns from the whole object. Comprehensive experiments have validated that AutoTrack is the best CPU-based tracker with a speed of ~60fps, and even outperforms some state-of-the-art deep trackers on two UAV datasets [46, 47]. In addition, we try to bridge the gap between the theory and practice by utilizing visual tracking in UAV localization in the real world. Considerable tests proved the effectiveness and generality of our method. We strongly believe that our work can promote the development of visual tracking and its application in robotics.

**Acknowledgment:** This work is supported by the National Natural Science Foundation of China (No.61806148), the Fundamental Research Funds for the Central Universities (No.22120180009), and Tsinghua University Initiative Scientific Research Program.
References


[33] Yuxuan Sun, Chong Sun, Dong Wang, You He, and Huchuan Lu. Roi pooled correlation filters for visual tracking. In CVPR, pages 5783–5791, 2019. 2


[47] 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 ECCV, pages 370–386, 2018. 5, 6, 8


