ICT Express 12 (2026) 523–529

Contents lists available at ScienceDirect

ICT Express

journal homepage: www.elsevier.com/locate/icte

Gated side adapters with memory-efficient fine tuning for RGB-T tracking

Dae-Hyeon Park, Mina Baek, Seung-Hwan Bae *

Department of Electrical and Computer Engineering, Inha University, Incheon, Republic of Korea

ARTICLE INFO

Keywords:
Object tracking
RGB-T tracking
Memory-efficient tuning
side-tuning
PEFT

ABSTRACT

Multi-modal tracking fuses different domain features to compensate for each other. Due to the large training complexity of the foundation RGB model, several parameter-efficient fine-tuning (PEFT) methods have been presented for RGB-T tracking. Although these PEFT methods effectively reduce the number of parameters, they still require significant resources. They demand more training memory and longer training times, similar to full fine-tuning. To solve this problem, we propose gated side adapters that remove the backpropagation through the foundation model. Furthermore, we propose a modality fusion module to adaptively integrate the foundation model with side model to overcome the domain gap. We reduce training time by 29.2% and memory usage by 51.7%, compared to full fine-tuning.

1. Introduction

Recognizing and locating objects [1–8] is one of the most challenging areas in the field of computer vision. Among them, RGB-X tracking aims to localize the target in search frames and exploit the knowledge of various domains by fusing RGB and X domains [4,9]. RGB-T tracking can effectively capture discriminative features and lead to more robust tracking performance, especially in low-illumination conditions where thermal features are robust to lighting variations. To this end, features from the RGB and thermal infrared (TIR) modalities should be fused into a unified representation. To consider both RGB and TIR domains, learning the overall parameters of the model requires significant computational overhead. To solve this problem, PEFT methods have been adopted, which add and tune only small modules in the foundation model. It is proved that PEFT is effective not only for large language models but also for vision backbones [2,10,11]. Due to this advantage, RGB-T trackers train only lightweight modules to transfer domains from RGB to RGB-T, instead of tuning the overall foundation model. Although PEFT significantly reduces the number of learnable parameters, it still demands high training time and memory usage, comparable to full fine-tuning. A major problem occurs when training a tiny module that is directly added into the foundation model, as the overall foundation model must perform backpropagation to learn a limited number of parameters. To address this problem, we are inspired by side-tuning methods [12,13]. They propose a side model with a small number of parameters, which not only freezes the parameters of the foundation model but also eliminates the need for backpropagation through the foundation model. They focus solely on learning this side module. These side tuning methods can significantly reduce training time and memory requirements by excluding backpropagation of the foundation model. To remove backward computations of the pretrained foundation model, we adopt this paradigm instead of PEFT. In this paper, we propose gated side adapters (GSA) that take TIR embedding features and fuse them with the foundation model through a gating factor, denoted as α in Section 3. In contrast to [12], this gating factor is adaptive, based on the outputs from each layer of the gated side adapter and the foundation model. It trains the fusion ratio to consider both model capacity and domain distribution. This tracking architecture enables training-efficient fine tuning, even without performing backpropagation. Then, we combine the GSA with a tracker based on vision transformer (ViT) [14]. In contrast to PEFT, which is trained inside the foundation model, training only side adapters externally prevents the foundation model from directly transferring to the RGB-T domain. This limitation arises because the adaptation process does not update the internal parameters of the foundation network. As a result, the model struggles to adapt its representations under domain shifts, degrading tracking accuracy beyond the original RGB domain. So, we suggest a modality fusion module (MFM) that flexibly adjusts the importance of the foundation model and the GSA using cross-attention mechanisms. Through this module, we integrate the RGB-tuned foundation model with the TIR-tuned GSA, taking into account heterogeneous model architectures and their varying capacities. In addition, robust object tracking requires the ability to adapt to temporal appearance variations of the target. Accordingly, we incorporate an online template update mechanism (OTU) that selectively updates the target representation during inference. By maintaining both the initial template and an online template updated based on classification confidence, our method

* Corresponding author.
E-mail addresses: saintpalite2221@inha.edu (D.-H. Park), dkdend123@inha.edu (M. Baek), shbae@inha.ac.kr (S.-H. Bae).

https://doi.org/10.1016/j.icte.2026.02.004
Received 23 June 2025; Received in revised form 29 December 2025; Accepted 4 February 2026
Available online 9 February 2026

enables continuous refinement of target appearance while preserving the stability of the frozen foundation model. This design allows the tracker to capture temporal variations effectively without introducing additional backpropagation. To compare the effectiveness of PEFT with our method on RGB-T tracking, we apply various PEFT methods to the vision transformer-based tracker and evaluate them against our methods in terms of tracking accuracy and training efficiency. In experiments, we train and evaluate our tracking model on the LasHeR and the RGBT234 datasets. Our contributions are as below:

- To enhance the efficiency of training time and memory usage, we combine side tuning paradigm to RGB-T tracking architecture.
- To handle the model capacity and domain distribution differences between the foundation model and the side model, we propose both gated side adapters (GSA) and a modality fusion module (MFM) in terms of global and layer-specific feature fusion.
- To capture temporal variation of the appearance of tracked object, we adopt the online template update mechanism.
- To verify our method, we analyze and compare the performance of our methods with various PEFT methods on LasHeR and RGBT234 dataset.

2. Related works

2.1. Parameter efficient fine-tuning

PEFT is a method that trains only a subset of parameters or modules, rather than fine-tuning the overall parameters of a foundation model. First, we explain methods that introduce additional modules into the foundation model and train only these modules. Prompt tuning [2] is a method to learn only prompts rather than tuning all the parameters of a foundation model. A prompt is a small learnable parameter that is fed into each of encoder layers. The adapter module [15] inserts a multi-layer perceptron (MLP) with a minimal number of parameters into the encoder and fine-tunes these parameters. By updating only the MLP module without changing the weights of the existing parameters, we can adapt the model to specific tasks and datasets through only a limited number of tuned parameters. LoRA (Low-Rank Adaptation) [16] freezes the weights of the pretrained model and introduces learnable rank decomposition matrices to each layer of the transformer architecture, so that we can reduce the number of tuned parameters. On the other hand, some PEFT methods tune only specific layers of the foundation model without introducing extra networks. Bias updates only the bias parameters within the weights and biases of the foundation model. Unlike weight parameters, bias parameters are relatively small size, which enables efficient training. Partial fine-tuning is a method that focuses on learning specific layers in a foundation model. Layernorm [17] tunes only the layer normalization layers and the tracking header network. Conv-Adapter [18] is composed of depth-wise and point-wise convolutions with a bottleneck structure and a learnable scaling factor, enabling adaptation of features by training only a small portion of additional parameters. Attention Prompt Tuning (APT) [19] injects learnable prompts directly into keys and values of the multi-head attention mechanism in the transformer encoder. However, these PEFT methods still require a large amount of training time and memory usage due to backpropagation through the overall foundation model. To address this problem, we employ the side-tuning paradigm to our method.

2.2. Side-tuning

Unlike PEFT, which adds existing modules within the foundation model, side-tuning [20] is a fine-tuning method that introduces side modules externally. Although side-tuning is a straightforward method that eliminates backpropagation for the foundation model, its training efficiency surpasses that of existing models that require backpropagation on the foundation model. PEFT methods significantly reduce learnable parameters, but they require relatively large amounts of computational cost and memory due to the backpropagation of the foundation model. To address this, LST [12] only trains several small networks outside the foundation model to prevent backpropagation in the backbone network. LAST [13] introduces a low-rank self-attention module as a side network. The outputs from each LSA block fuse with the foundation model using residual summation. To increase the training efficiency, we get inspiration from these memory-efficient side-tuning methods.

3. Methods

3.1. Revisiting ViT-based tracker

We employ a ViT-based tracker, OSTrack [1] as shown in Fig. 1 (a). We set vision the transformer optimized for RGB images, as the foundation model. $I^s_{RGB/TIR}$ are search images, and $I^t_{RGB/TIR}$ indicate cropped images of the tracked object from $I^s_{RGB/TIR}$ in both RGB and TIR domains. By linear projection layers with 2D convolution and flatten, $I^s_{RGB/TIR}$ and $I^t_{RGB/TIR}$ become tokenized features $T^s_{RGB/TIR} \in \mathbb{R}^{B\times N_s\times C}$ and $T^t_{RGB/TIR} \in \mathbb{R}^{B\times N_t\times C}$, respectively. Here, B and C denote the batch size and the number of channels, respectively. N_s and N_t denote the number of search and template tokens, respectively. After that, we concatenate T^s and T^t by RGB domain and TIR domain to generate token features M_0 and N_0, respectively. M_0 and N_0 are RGB and TIR token features, respectively. The foundation model that consists of l-layers of feature extractor $E(\cdot)$ takes M_0 as input and produces an RGB embedding feature M_l that highlights the tracked object. We exploit ViT [14] encoder as $E_i(\cdot)$. The outputs of the $E(\cdot)$ and the GSA are fused via the MFM to generate the final embedding feature $S \in \mathbb{R}^{B\times(N_s+N_t)\times C}$. This embedding feature is then sliced into $V \in \mathbb{R}^{B\times N_s\times C}$ with the same size as the search token, which is the input of the tracking header. We adopt the tracking header from [1], which consists of classification, size, and bounding box regression headers.

3.2. Gated side adapters (GSA)

In order to transfer the tracker to the RGB-T domain, it is common to train the overall parameter of foundation model. However, this requires a significant amount of training time and memory. To address this problem, we adopt the side tuning paradigm, which trains only a sub-network instead of performing backpropagation on the foundation model. Therefore, we also handle the RGB and TIR domains without backpropagation in the foundation model. We adopt the side adapter as a learnable sub-network and fuse the outputs from the foundation model $E(\cdot)$ and sub-network $D(\cdot)$ using a gating network. The side adapter $D(\cdot)$ takes tokenized TIR feature N_0 as input and produces a TIR embedding feature N_l. The $D(\cdot)$ consists of a feedforward down-projection layer $\mathrm{FC}_1(\cdot)$, Gaussian Error Linear Unit (GELU) as an activation layer $\mathrm{GELU}(\cdot)$, and a feedforward up-projection layer $\mathrm{FC}_2(\cdot)$, as described in [15]. The output dimension of the down-projection layer is $1/r$, while the up-projection layer has an output dimension that is r times the hidden layer dimension. We ensure that the output dimension of the $D_i(\cdot)$ matches the input dimension. r represents the reduction factor, which we set to 8 for our experiments. We implement $D_i(\cdot)$ as follow:

$$N_{i+1} = N_i + \mathrm{FC}_2(\mathrm{GELU}(\mathrm{FC}_1(N_i))) \quad i = 0, \ldots, l-1 \tag{1}$$

To reduce the imbalance between the foundation and the side adapter model, we employ a gating network to fuse the outputs from the $E(\cdot)$ and the $D(\cdot)$, as shown in Fig. 1 (a). We exploit the gating network to generate the gating factor $\alpha \in \mathbb{R}^{B\times(N_s+N_t)\times 1}$, which serves as the

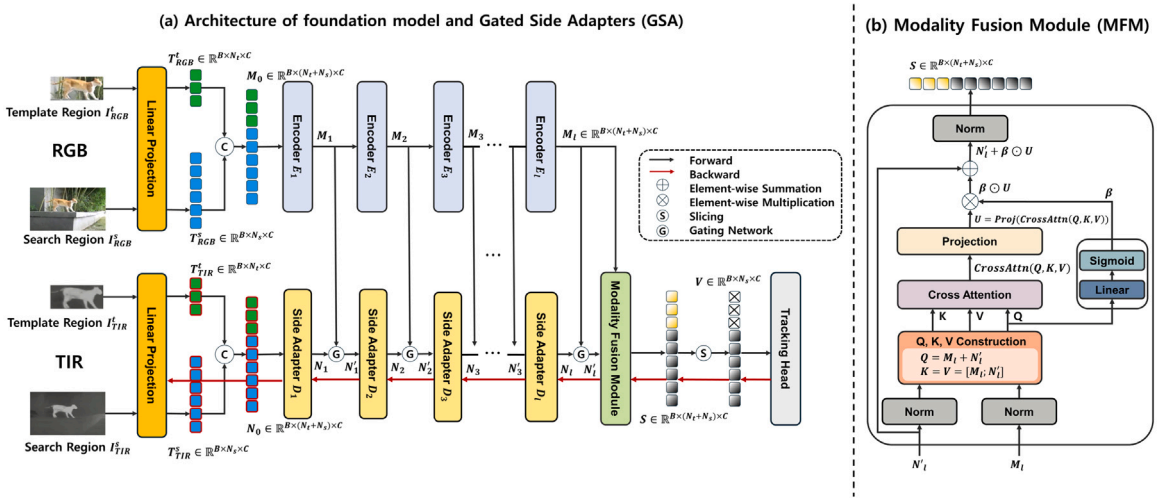

Fig. 1. (a) The architecture of a vision transformer-based RGB-T tracker with gated side adapters integrates both RGB and TIR images. Search and template tokens are concatenated along a token axis and subsequently fed into the encoder layers and the GSA, respectively. The gated side adapters learn the relationship between the search and template regions even without backpropagation through the encoder layers. (b) The modality fusion module, located on the right side of the figure, fuses the final output from the encoder and the adapter by performing cross-attention.

fusion ratio between $E(\cdot)$ and $D(\cdot)$. We implement the gating network as follows:

$$\begin{aligned} \alpha_i &= \mathrm{Sigmoid}\big(\mathrm{FC}_s(\mathrm{Norm}(M_i + N_i))\big) \quad && i = 1, \ldots, l \\ N_i' &= \alpha_i \odot M_i + (1 - \alpha_i) \odot N_i && i = 1, \ldots, l \end{aligned} \tag{2}$$

$\mathrm{Sigmoid}(\cdot)$ and $\odot$ denote a sigmoid function and the element-wise multiplication, respectively. $\mathrm{Norm}(\cdot)$ is a layer normalization operation here. $\mathrm{FC}_s(\cdot)$ denotes a fully connected layer that maps the output dimension to 1. Eq. (2) shows the proposed gating network at layer i. α_i adjusts the proportion of the M_i and N_i to fuse the encoder output and the GSA output. N_i' is fed into the next side adapter D_{i+1}.

3.3. Modality fusion module (MFM)

This module aims to more comprehensively integrate RGB and TIR domains through the late fusion of their features. It fuses the global representation from the output of the final layer of the foundation model and the GSA. Fig. 1-(b) illustrates the detailed computational flow of the proposed MFM. Let $M_l \in \mathbb{R}^{B\times(N_s+N_t)\times C}$ denote the final layer RGB embedding from the foundation encoder, and let $N_l' \in \mathbb{R}^{B\times(N_s+N_t)\times C}$ denote the corresponding TIR embedding after the GSA. To enable cross-modal interaction while preserving modality-specific representations, we construct the query, key, and value embeddings as follows. The query is defined as $Q = M_l + N_l'$, which aggregates cues from both modalities. The keys and values are defined as $K = V = [M_l; N_l']$, allowing the attention mechanism to operate over modality-separated features. To control the contribution of the cross-attention output, we introduce a learnable fusion factor β that adaptively weights the cross-modal interaction. With these values, we propose the modality fusion module as follows:

$$\begin{aligned} \beta &= \mathrm{Sigmoid}(\mathrm{FC}(M_l + N_l')) \\ U &= \mathrm{Proj}(\mathrm{CrossAttn}(Q, K, V)) \\ S &= \mathrm{Norm}(N_l' + \beta \odot U) \end{aligned} \tag{3}$$

$\mathrm{CrossAttn}(\cdot)$ denotes cross-attention with matrix multiplication and scaling. We set the scaling factor as $\frac{1}{\sqrt{d_h}}$, while d_h denotes the dimension divided by the number of heads. $\mathrm{Proj}(\cdot)$ denotes a linear projection layer here. Given Q, K, and V, cross-attention is applied to extract cross-modal contextual features. The output U preserves the token length of Q, ensuring compatibility with subsequent fusion. $\mathrm{FC}(\cdot)$ is a fully connected layer to combine RGB and TIR features, M_l and N_l'. Finally, S is normalized to serve as the input for the tracking head. This formulation ensures that cross-modal information is incorporated in a controlled and adaptive manner, while preserving the stability of the frozen foundation model.

3.4. Online template update mechanism (OTU)

This strategy adopts the online template update mechanism used in [21,22]. Specifically, N_0 is replaced by $[T^o_{TIR}; T^t_{TIR}; T^s_{TIR}]$, and M_0 is replaced by $[T^o_{RGB}; T^t_{RGB}; T^s_{RGB}]$. Here, $T^o_{RGB/TIR}$ denotes the template extracted from the previous frame, which is called as the online template. The template provided in the first frame $T^t_{RGB/TIR}$ is referred to as the initial template. This mechanism is able to capture the temporal variation of tracked object. However, existing methods such as [21,22] adopt a conservative online template update mechanism to mitigate error accumulation and model drift. To maximize the capture of variations in the tracked target, we determine whether to update the online template on a per-frame basis. Instead, the online template is updated at each frame only when the maximum response of the tracking classification head exceeds an update threshold τ. To sum up, we investigate whether the simultaneous use of the initial template and the online template is feasible, even under a frozen backbone setting.

4. Experiments

4.1. Implementation details

Our baseline is described in Section 3. We discard the token pruning method from the baseline and adopt the weights trained by OSTrack [1]. We set the image size of the search region to 256 and the template region to 128, with a learning rate of 0.0008 for training. The batch size is set to 64, and the number of training epochs is 20. The reduction factor of the GSA is set to 8 and 16 on LasHeR and RGBT234, respectively. We set d_h, N_s, and N_t to 768, 256, and 64, respectively. Also, we set the update threshold τ to 0.7 for the OTU. Each epoch contains 60,000 pairs of image samples, including both RGB and TIR images. We employ AdamW as the optimizer for training the model. During training, validation is performed every 5 epochs. We set the weight decay to 1×10^{-4} for all trainable parameters. For learning rate scheduling, we adopt a step-based scheduler with a decay rate of 0.1. We exploit the loss function from [1]. In all experiments of PEFT methods, we tune the linear projection that generates the tokenized TIR feature N_0 from given TIR images $I^{s/t}_{TIR}$ while freezing all the other

Table 1
Ablation results of methods on LasHeR testingset and RGBT234 dataset. GSA, MFM, and OTU present the gated side adapters, the modality fusion module, and the online template update mechanism of our method, respectively. For detailed ablation, we split the GSA into side adapters $D(\cdot)$ and gating factor α. Likewise, we also split the MFM into cross-attention (C·Attn) and fusion factor β. ✓: applied, ✗: not applied.

	GSA		MFM		OTU	Tuned	VRAM	Training	RGBT234		LasHeR	
	$D(\cdot)$	α	C·Attn	β		Param.		Time	MPR	MSR	PR	SR
(1)	✗	✗	✗	✗	✗	–	–	–	71.12	53.91	48.35	39.61
(2)	✗	✗	✓	✗	✗	2.95M	6.3 GB	6,579 s	78.43	57.30	56.45	45.52
(3)	✗	✗	✓	✓	✗	3.54M	6.6 GB	6,623 s	79.02	57.53	57.13	45.85
(4)	✓	✗	✗	✗	✗	1.78M	5.6 GB	7,126 s	80.32	58.19	59.26	47.39
(5)	✓	✓	✗	✗	✗	1.96M	6.9 GB	7,542 s	80.54	58.74	61.09	48.82
(6)	✓	✓	✓	✓	✗	5.51M	9.6 GB	8,213 s	81.52	59.33	61.54	49.10
(7)	✓	✓	✓	✓	✓	5.51M	11.5 GB	9,580 s	84.78	61.53	64.12	51.03

Table 2
Comparison between PEFT on LasHeR testing set and RGBT234 dataset. †/‡ are efficient parameter fine tuning method with/without added tiny module, respectively.

Tuning Type	Tuned Param.	VRAM	Training Time	Training FLOPs	RGBT234 MPR	MSR	LasHeR PR	SR
Baseline	–	–	–	–	71.12	53.91	48.35	39.61
Full Fine-Tuning	93.10M	23.8 GB	13,521 s	87.72G	83.52	61.27	68.33	54.65
Bias†	0.69M	18.8 GB	11,097 s	87.72G	82.28	60.80	60.77	48.41
Partial Fine-Tuning†	7.67M	19.0 GB	10,359 s	87.72G	80.75	58.97	59.64	48.05
LayerNorm† [17]	7.69M	18.8 GB	10,931 s	87.72G	82.29	60.21	61.44	48.96
Prompt‡ [2]	0.68M	19.7 GB	11,005 s	90.27G	83.15	61.46	65.35	52.52
Adapter‡ [15]	4.14M	19.6 GB	10,950 s	91.11G	83.44	62.23	66.41	53.42
LoRA‡ [16]	0.89M	19.7 GB	12,425 s	87.72G	77.92	58.12	58.49	46.18
Conv-Adapter‡ [18]	4.24M	20.3 GB	11,779 s	91.20G	81.93	60.79	62.44	50.10
APT‡ [19]	0.60M	18.9 GB	10,469 s	87.72G	82.70	61.11	63.87	51.32
Ours	**5.51M**	**11.5 GB**	**9,580 s**	**48.59G**	**84.78**	**61.53**	**64.12**	**51.03**

modules and layers and learning only a tiny module. All experiments are conducted with a fixed random seed for reproducibility. Finally, we train and evaluate the model using 4 NVIDIA RTX 3090 GPUs.

4.2. Datasets and metrics

These experiments are learned on the LasHeR training dataset and evaluated on the LasHeR [23] testing set and the RGBT234 [24] dataset. The LasHeR and the RGBT234 dataset consist of image pairs including both RGB and TIR domain images. The TIR domain image is generated through an infrared sensor, and visualizes temperature information with infrared information emitted by an object. The LasHeR dataset consists of 1224 sequences including 730K images, and it can correspond to various tracking environments such as day/night and various weather conditions. LasHeR exploits the precision rate (PR) and success rate (SR) as metrics. RGBT234 provides 234 sequences including 233K images, and exploits the maximum precision rate (MPR) and maximum success rate (MSR) as metrics. As shown in Table 2, we define training FLOPs to measure the computational cost during training for comparing training efficiency. Training FLOPs are the sum of the cost of forward and backward pass. Following prior work [25], we approximate the cost of backpropagation as twice that of the forward pass when reporting training FLOPs.

4.3. Ablation study

As shown in Table 1, we present an ablation study to investigate the impacts of the proposed GSA, the MFM and OTU. We evaluate them within computational cost and tracking accuracy on the LasHeR and RGBT234 datasets. Table 1-(1) serves as the baseline without any module from our method.

Gated Side Adapters (GSA). Table 1-(4)/(5) employs only the GSA without/with the gating factor α, respectively. Table 1-(5) demonstrates a significant performance improvement over the baseline, achieving +12.74%/+12.21% and +9.42%/+4.83% for PR/SR and MPR/MSR on LasHeR and RGBT234 datasets, respectively. Compared with Table 1-(4) and Table 1-(5), we implement experiments to compare the static gating factor with dynamic gating factor. Static gating factor is to replace α with the learnable parameters. As shown in Eq. (2), our dynamic gating factor α is represented by $\alpha \in \mathbb{R}^{B\times(N_s+N_t)\times 1}$. Therefore, we define the static gating factor as a learnable parameter $a \in \mathbb{R}^{B\times(N_s+N_t)\times 1}$ to ensure a fair comparison. Therefore, for any given input image, the ratio between the foundation and the side network remains fixed. In contrast, the dynamic gating factor determines the ratio between the two networks depending on the input. Compared Table 1-(4) with Table 1-(5), Table 1-(5) has more PR/SR gain +0.83%/+1.43% and +0.22%/+0.55% on LasHeR and RGBT234. We provide the static and dynamic gating factor results on Fig. 2. There are significant gap in the High Illumination (HI) and Camera Moving (CM) scenarios. The dynamic gating factor achieves PR 65.4/59.1% while the static gating factor lags behind at PR 60.6/54.7% on HI/CM scenarios. The static gating factor fails because it relies on meaningless TIR embeddings

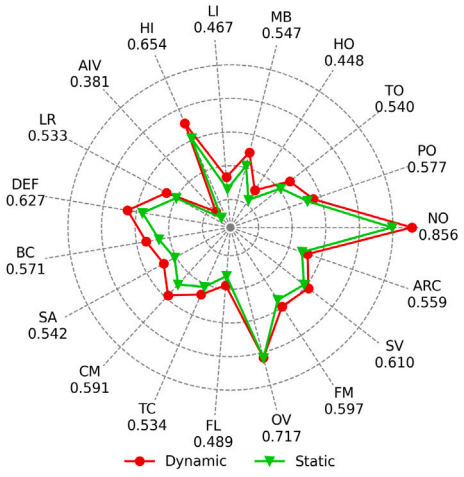

Fig. 2. Comparison of static and dynamic gating factors in terms of precision rate for 19 attributes on LasHeR.

when the corresponding thermal data is compromised. In contrast, our dynamic gating factor adaptively adjusts the fusion ratio to utilize only the useful TIR embeddings.

Modality Fusion Module (MFM). Table 1-(2)/(3) employs only the MFM without/with the fusion factor β, respectively. Table 1-(3) also improves performance compared to Table 1-(1) achieving +8.78%/+6.24% and +7.90%/+3.62% for PR/SR and MPR/MSR on LasHeR and RGBT234 datasets, respectively. When the MFM is added to Table 1-(5), we acquire the performance gain of +0.45%/+0.28% and +0.98%/+0.59%, respectively. This demonstrates that the MFM compensates for GSA's lack of modality fusion abilities, reducing the disparity due to the heterogeneous architecture and domains in RGB-T tracking. Comparing Table 1-(2) with Table 1-(3), Table 1-(3) has more PR/SR gain +0.68%/+0.33% and +0.59%/+0.23%. It shows that the adaptive fusing between N_l' and M_l using β is effective. By combining MFM and GSA as shown in Table 1-(6), we have the best tracking accuracy among Table 1-(1-6).

Online Template Update Mechanism. Table 1-(5) evaluates the full configuration, in which GSA, MFM, and online template update mechanism (OTU) are all enabled. This configuration achieves the best tracking accuracy among all compared settings. On the LasHeR/RGBT234, it records PR/MPR 64.12/84.78% and SR/MSR 51.03/61.53%, showing consistent improvements over all partial configurations, respectively. These results demonstrate that continuously refining target representations across frames through online template updating enables more stable and accurate tracking in challenging RGB-T tracking scenarios.

4.4. Comparison with PEFT

In this section, we apply various parameter-efficient fine-tuning methods to a ViT-based tracker and compare them with ours. We evaluate adapter tuning, prompt tuning, layer normalization, bias tuning, partial fine-tuning, and LoRA, together with full fine-tuning. Full fine-tuning is to tune all the model parameters. In Table 2, the tuned parameter size (Tuned Param.) denotes the number of learnable parameters in training. We measure training time on a single GPU for a total of 20 epochs, excluding validation time performed every 5 epochs. VRAM usage shows GPU memory consumption and FLOPs is floating point operations per second. We use the symbol † to denote a method without the tiny module and ‡ to denote a method with the tiny module. † and ‡ correspond to group bias, partial fine-tuning, layer-norm tunings and prompt, adapter, LoRA tunings, APT, and Conv-Adapter, respectively. As shown in Table 2, on the LasHeR dataset, ours outperforms bias, partial fine-tuning, layernorm, LoRA, APT, and Conv-Adapter. We achieve the highest MPR on the RGBT234 dataset among all PEFT baselines, even including full fine-tuning. We can achieve high tracking accuracy while saving resources due to our MFM that combines RGB with TIR features and OTU that adaptively updates the template image. In respect of model complexity, the difference in FLOPs between PEFT methods are minor, because the number of parameters has not changed significantly. Since our framework completely removes backpropagation through the foundation model, the overall training FLOPs are significantly reduced, resulting in the lowest training FLOPs among all compared methods. In particular, there is not much difference between full-fine tuning and PEFT methods for training time and VRAM usage because it is still necessary to store tensors and calculate a backpropagation. In contrast, the training time and VRAM usage of ours are much lower than other methods, demonstrating the efficiency of our method. If we compare ours with full fine-tuning, the difference in training time, VRAM usage and training FLOPs is 3941 s, 12.3 GB, and 39.13G, respectively. The reason for these results is the backpropagation of the foundation model. We can skip the backward propagation of the pretrained foundation model by training only our GSA, whereas other PEFT methods must learn the overall foundation model to adjust tiny submodules.

Table 3
Statistical reliability analysis on RGBT234 dataset. We investigate MPR@20 performance differences (Δmean = Ours - PEFT) along with 95% confidence intervals (CI). The values in brackets $[a, b]$ denote the lower (a) and upper (b) bounds of the CI.

Tuning Type	Δmean [95% CI]
Full Fine-Tuning	+0.0126 [−0.0135, +0.0364]
Bias	+0.0250 [−0.0015, +0.0501]
Partial Fine-Tuning	+0.0403 [+0.0140, +0.0653]
LayerNorm	+0.0249 [+0.0009, +0.0475]
Prompt	+0.0163 [+0.0108, +0.0420]
Adapter	+0.0134 [−0.0127, +0.0366]
APT	+0.0208 [−0.0060, +0.0459]
Conv-Adapter	+0.0285 [+0.0026, +0.0550]
LoRA	+0.0734 [+0.0428, +0.1053]

Table 4
Comparison of cross-dataset validation between PEFT on LasHeR and RGBT234.

Method	LasHeR → RGBT234		RGBT234 → LasHeR	
	PR	SR	PR	SR
Full Fine-Tuning	83.52	61.27	54.26	40.32
Bias	82.28	60.08	56.73	41.74
Partial Fine-Tuning	80.75	58.97	54.85	41.13
LayerNorm	82.29	60.21	56.53	42.21
Prompt	83.15	61.46	60.16	44.46
Adapter	83.44	62.23	58.84	43.41
LoRA	77.92	58.12	54.38	41.61
Ours	**84.78**	**61.53**	**57.05**	**43.48**

4.5. Statistical reliability

Since our method achieves the highest MPR on the RGBT234 dataset, as shown in Table 2, we analyze whether this performance gain is statistically reliable. To this end, we conduct paired t-tests on per-sequence MPR@20 scores evaluated on RGBT234. Here, MPR@20 denotes the maximum precision rate, defined as the percentage of frames with a predicted center error within 20 pixels of the ground truth. Table 3 shows the mean differences (Δmean = Ours - PEFT) together with their 95% confidence intervals (CI). As shown in Table 3, all mean differences are positive, indicating that our method consistently outperforms the compared PEFT baselines. Several methods, including Partial Fine-Tuning, LayerNorm, Prompt, Conv-Adapter, and LoRA, exhibit confidence intervals that do not include zero, confirming statistically significant improvements. For the remaining methods, although the confidence intervals overlap with zero, the overall tendency still favors our method. These results demonstrate that the superior MPR performance of our method on RGBT234 is statistically reliable rather than incidental.

4.6. Comparison with cross-validation

For comparison with PEFT and ours in terms of model generalization, we evaluate cross-validation on LasHeR and RGBT234. A→B denotes that it evaluates the model trained with A at B. We show the results from LasHeR→RGBT234 and RGBT234→LasHeR as shown in Table 4. For comparison among PEFT and ours in terms of model generalization, we evaluate cross-validation on LasHeR and RGBT234. Since RGBT234 is not split to training and testing sets, the results from LasHeR→RGBT234 are the same as Table 2. We observe that full fine tuning degrades PR 14.07% and SR 14.33%. The other PEFT methods, along with ours, show a consistent tendency similar to Table 1 when excluding the extreme cases of full fine-tuning.

4.7. Sensitivity analysis for reduction factor

To affect our model according the reduction factor in GSA, we evaluate our model with revising various reduction factor. As shown

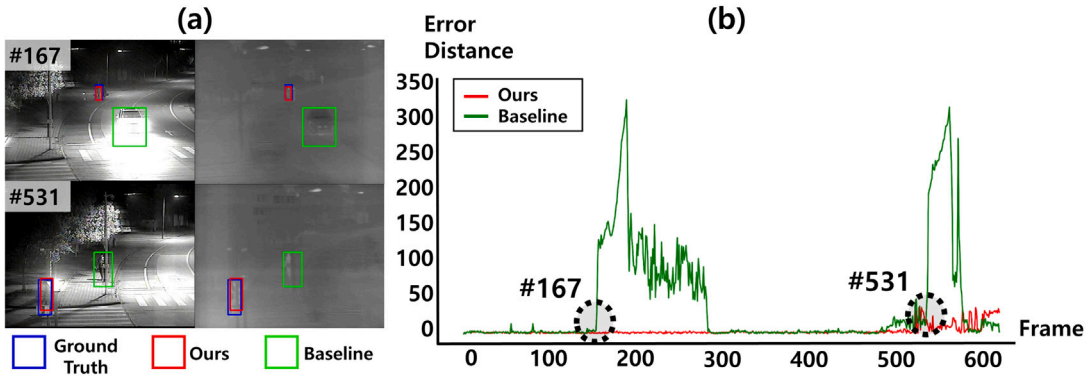

Fig. 3. Comparison with qualitative results from darkgirl sequence in the LasHeR dataset. Error distance is the Euclidean distance based on the center position of the ground truth and the predicted results. The black dot circle appears in certain frames that correspond to the images shown left. The green line represents the baseline model, and the red line represents our model.

Table 5
Sensitivity analysis of the reduction factor r in the side adapter on LasHeR and RGBT234. The best accuracy is highlighted in bold.

Method	Reduction factor r	LasHeR PR	LasHeR SR	RGBT234 MPR	RGBT234 MSR
Ours	1	63.29	50.36	82.34	60.18
	2	62.22	49.36	83.90	60.41
	4	63.58	50.50	83.10	60.90
	8	**64.12**	**51.03**	83.36	60.46
	16	63.18	50.48	**84.78**	**61.53**
	32	63.86	50.79	84.07	61.21

in Table 5, we set reduction factor r to 1, 2, 4, 8, 16, and 32. We figure out that the larger r, the smaller the number of learnable parameters. Because of showing the best PR/SR on LasHeR/RGBT234 in case of $r = 8/16$, we determine that r is set to 8/16 on LasHeR/RGBT234, respectively.

4.8. Qualitative comparison

We visualize the tracking results of our model and plot the error distance, which represents the Euclidean distance between the predictions and the ground truth. As shown in Fig. 3-(a), we evaluate our model on a specific sequence from the LasHeR testing set that includes high illumination (HI) and camera moving (CM) situations near frames 100–200 and 500–600, respectively. At frame 167, the tracked object is in an HI situation. We observe that our model successfully tracks the target, while the baseline fails and tracks another object. At frame 531, our model precisely tracks the target, while the baseline fails in a CM situation. The plot in Fig. 3-(b) also illustrates these results using error distances. In this experiment, we demonstrate that our tracker performs robustly under extreme conditions compared to the baseline model.

5. Conclusion

We propose a training-efficient RGB-T tracker with two modules, the gated side adapters (GSA) and the modality fusion module (MFM) with the online template update mechanism (OTU). To verify the effectiveness of our method, we compare the performance of different fine-tuning methods based on the ViT-based tracker. Through experiments on the LasHeR and RGBT234 datasets, we prove that our method is competitive in terms of training-time computational cost while maintaining high tracking accuracy. We demonstrate that our method achieves a good balance between efficiency and accuracy, making it suitable for real-world RGB-T tracking.

CRediT authorship contribution statement

Dae-Hyeon Park: Writing – original draft, Validation, Software, Methodology. **Mina Baek:** Writing – original draft, Software, Methodology, Formal analysis, Data curation. **Seung-Hwan Bae:** Writing – original draft, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea (NRF) grants funded by the Korea government (MSIT) (No. RS-2022-NR071978) and funded by the Ministry of Education (No. RS-2022-NR070869); supported in part by Institute of Information & communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No. RS-2022-II220448: Deep Total Recall, 10%); supported in part by INHA UNIVERSITY Research Grant.

References

[1] B. Ye, H. Chang, B. Ma, S. Shan, X. Chen, Joint feature learning and relation modeling for tracking: A one-stream framework, in: ECCV, Springer, 2022, pp. 341–357.
[2] M. Jia, L. Tang, B.-C. Chen, C. Cardie, S. Belongie, B. Hariharan, S.-N. Lim, Visual prompt tuning, in: ECCV, Springer, 2022, pp. 709–727.
[3] J.-Y. Baek, Y.-S. Yoo, S.-H. Bae, A new multi-source light detection benchmark and semi-supervised focal light detection, Adv. Neural Inf. Process. Syst. 37 (2024) 69637–69651.
[4] B. Cao, J. Guo, P. Zhu, Q. Hu, Bi-directional adapter for multimodal tracking, in: AAAI, vol. 38, (2) 2024, pp. 927–935.
[5] E.-S. Lee, Y.-I. Lee, Y.-C. Kim, E.-K. Jo, B.-S. Shin, An efficient intrusion detecting method using multiple sensors and edge computing, Human-Centric Comput. Inf. Sci. 13 (2023) 15.
[6] N. Noor, I.K. Park, A lightweight skeleton-based 3D-CNN for real-time fall detection and action recognition, in: ICCVW, 2023, pp. 2179–2188.
[7] J.D. Choi, M.Y. Kim, A sensor fusion system with thermal infrared camera and LiDAR for autonomous vehicles and deep learning based object detection, ICT Express 9 (2) (2023) 222–227.
[8] D. Lim, J. Kim, H. Kim, Efficient robot tracking system using single-image-based object detection and position estimation, ICT Express 10 (1) (2024) 125–131.
[9] J. Zhu, S. Lai, X. Chen, D. Wang, H. Lu, Visual prompt multi-modal tracking, in: CVPR, 2023, pp. 9516–9526.

[10] L. Lin, H. Fan, Z. Zhang, Y. Wang, Y. Xu, H. Ling, Tracking meets lora: Faster training, larger model, stronger performance, in: ECCV, Springer, 2024, pp. 300–318.
[11] L. Hong, S. Yan, R. Zhang, W. Li, X. Zhou, P. Guo, K. Jiang, Y. Chen, J. Li, Z. Chen, et al., Onetracker: Unifying visual object tracking with foundation models and efficient tuning, in: CVPR, 2024, pp. 19079–19091.
[12] Y.-L. Sung, J. Cho, M. Bansal, Lst: Ladder side-tuning for parameter and memory efficient transfer learning, NIPS 35 (2022) 12991–13005.
[13] N. Tang, M. Fu, K. Zhu, J. Wu, Low-rank attention side-tuning for parameter-efficient fine-tuning, 2024, arXiv preprint arXiv:2402.04009.
[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, ICLR (2021).
[15] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, S. Gelly, Parameter-efficient transfer learning for NLP, in: ICML, PMLR, 2019, pp. 2790–2799.
[16] E.J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen, et al., Lora: Low-rank adaptation of large language models., ICLR 1 (2) (2022) 3.
[17] B. Zhao, H. Tu, C. Wei, J. Mei, C. Xie, Tuning LayerNorm in attention: Towards efficient multi-modal llm finetuning, ICLR (2024).
[18] H. Chen, R. Tao, H. Zhang, Y. Wang, X. Li, W. Ye, J. Wang, G. Hu, M. Savvides, Conv-adapter: Exploring parameter efficient transfer learning for convnets, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 1551–1561.
[19] W.G.C. Bandara, V.M. Patel, Attention prompt tuning: Parameter-efficient adaptation of pre-trained models for spatiotemporal modeling, 2024, arXiv preprint arXiv:2403.06978.
[20] J.O. Zhang, A. Sax, A. Zamir, L. Guibas, J. Malik, Side-tuning: a baseline for network adaptation via additive side networks, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16, Springer, 2020, pp. 698–714.
[21] B. Yan, H. Peng, J. Fu, D. Wang, H. Lu, Learning spatio-temporal transformer for visual tracking, in: CVPR, 2021, pp. 10448–10457.
[22] J. Yang, Z. Li, F. Zheng, A. Leonardis, J. Song, Prompting for multi-modal tracking, in: ACMMM, 2022, pp. 3492–3500.
[23] C. Li, W. Xue, Y. Jia, Z. Qu, B. Luo, J. Tang, D. Sun, LasHeR: A large-scale high-diversity benchmark for RGBT tracking, IEEE TIP 31 (2021) 392–404.
[24] C. Li, X. Liang, Y. Lu, N. Zhao, J. Tang, RGB-T object tracking: Benchmark and baseline, PR 96 (2019) 106977.
[25] J. Kaplan, S. McCandlish, T. Henighan, T.B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, D. Amodei, Scaling laws for neural language models, 2020, arXiv preprint arXiv:2001.08361.