Transformer-Based Dual-Modal Visual Target Tracking Using Visible Light and Thermal Infrared

*Pengfei Lyu, Minxiang Wei
College of Energy and Power Engineering,
Nanjing University of Aeronautics and Astronautics
Nanjing, China
*lvpengfei, minxiangfwei@nuaa.edu.cn

Yuwei Wu
College of Mechanical and Electrical Engineering,
Nanjing University of Aeronautics and Astronautics
Nanjing, China
wyw826@nuaa.edu.cn

***Abstract*—Visual target tracking is an essential technology with numerous applications, including video surveillance, motion recognition, and autonomous driving. However, tracking accuracy can be affected in challenging scenarios, such as low-light conditions and occlusion, which make it difficult to extract effective tracking features from a single visible light image. On the other hand, infrared images can penetrate occlusion and are insensitive to light. However, tracking using only infrared images can be influenced by thermal crosstalk and lacks detailed texture information. Therefore, the RGB-T tracking method, which combines visible light and thermal infrared images, can significantly enhance the accuracy and robustness of object tracking in challenging scenarios, especially in autonomous driving. We propose a transformer-based fusion tracker that utilizes dual-modal information and combines test and training branches with target encoding for global reasoning across frames. The proposed method successfully achieves target tracking using visible light and thermal infrared images. The experimental results on the public benchmark show that the proposed tracker has higher overall performance and can meet the requirements for precise and robust tracking of vulnerable road users in autonomous driving tasks.**

I. Introduction

Visual target tracking is a computer vision task that aims to accurately locate and track objects in video frames. It is used in applications like surveillance, robotics, and autonomous driving. Visual target tracking is challenging under low illumination, occlusion, and rainy conditions because visible light images provide insufficient features for effective tracking [1]. These problems are inherent limitations of visible light images, and cannot be solved through network architecture alone. However, infrared images are insensitive to light, sensitive to temperature, and can penetrate situations such as smoke obstruction. Despite their advantages, they lack information such as fine textures. RGB-T tracking can address these shortcomings by fusing complementary information from visible light and thermal infrared images, leading to improved accuracy and robustness in object tracking, particularly in the context of autonomous driving. Fig. 1 illustrates the complementary advantages of the RGB and thermal infrared modalities.

RGB-T fusion tracking methods [2] can be categorized into pixel-level, feature-level, and decision-level fusion tracking based on the fusion stage. Pixel-level fusion tracking first fuses images of different modalities and then performs object tracking. However, this approach requires significant computation, which may impact the overall speed of the model. Feature-level fusion tracking extracts features from RGB and infrared images and fuses them based on designed fusion rules, resulting in improved recognition performance. Decision-level fusion tracking first tracks a single modality and then fuses the results to obtain the final result. Compared to pixel-level methods, feature-level and decision-level methods are more direct. The feature-level method requires solving the problem of feature extraction and fusion of visible light and infrared images, while the decision-level method requires selecting different trackers for tracking.

(a) *baginhand* - Visible and thermal infrared image pairs

(b) *GarageHover* - Visible and thermal infrared image pairs

Fig. 1. Explanation of complementary advantages of visible light and thermal infrared multimodal image tracking

In recent years, the field of RGBT tracking has seen significant advancements with the exploration and application of correlation filters. Several notable RGBT trackers have emerged, showcasing impressive results. For instance, Zhai et al. [3] have achieved consistent object localization by jointly learning correlation filters from different modalities with the inclusion of low-rank constraints. However, this approach relies solely on handcrafted object features, which may not

effectively represent the rich information of the object. In light of the widespread use of convolutional neural networks (CNNs) in RGB tracking, Zhang et al. [4] have taken a different approach by employing a fully convolutional Siamese network for RGBT tracking. This tracker has the capability to run at an impressive rate of approximately 30 frames per second. Nonetheless, it's worth noting that the correlation operation of Siamese-based trackers is a simple local linear matching process between the template and search area, which may have limitations in capturing complex object appearances and variations.

In this paper, we propose an end-to-end Transformer-based multi-modal visual object tracking method. This method utilizes complementary information from visible light and thermal infrared modalities to address the challenge of extracting meaningful features from visible light in low-light conditions. Our approach introduces a Transformer encoder-decoder architecture to handle global information, overcoming the limitations of CNN-based methods that rely on local receptive fields. The Transformer encoder utilizes the encoded concatenated features from the bi-modal input and outputs concatenated features with enhanced test features. The Transformer decoder leverages the concatenated features output from the Transformer encoder to predict model weights. The model weights, along with the enhanced test features, are then passed through linear layers and CNNs to output the final classification response map and regression bounding box, resulting in the tracking results. Qualitative and quantitative experimental results on publicly available datasets demonstrate that our proposed method outperforms several state-of-the-art algorithms in terms of tracking performance.

II. Related Work

A. RGB Tracking

Deep learning-based RGB tracking methods can be divided into two categories: classifier-based and siamese network-based. The former achieves object tracking through classification, while the latter performs tracking by comparing similarity. Recently, MDNet [5] combined domain learning and CNNs to learn a shared representation model for object tracking. The model consists of shared layers and K branches of fully connected layers, corresponding to K domains. Siamese network-based trackers aim to learn an appearance model that maximizes the distance between blocks of different objects and minimizes the distance between blocks of the same object. SiamFC [6] and SiamRPN [7] achieved real-time high tracking speeds due to their fully convolutional siamese networks. To further improve the discriminative ability of trackers, DaSiamRPN [8] incorporates potential distractors into the embedding space learning process and introduces a distractor-aware module. In addition, SiamRPN++ [9] and SiamDW [10] adopt deeper backbone networks such as ResNet [11] for feature extraction. SiamRPN++ replaces the up-channel cross-correlation layers in SiamRPN with lightweight deep cross-correlation layers. Meanwhile, it aggregates multi-layer features to further enhance the accuracy of the tracker. In SiamDW, the authors design a new module called Cropping-Inside Residual (CIR) unit, which helps build deeper and wider backbone networks.

B. RGBT Tracking

With the popularity of thermal infrared sensors and the proposal of RGBT234 [1] tracking benchmarks, RGBT tracking has attracted extensive attention. In recent years, the fusion tracking technology based on correlation filtering has also begun to attract the attention of researchers because of its good tracking performance and high efficiency. Wang et al. [12] proposed a soft consistency filter for visual tracking using visible light and thermal infrared data. This method employs soft consistency to account for collaboration and heterogeneity, enabling joint learning of correlation filters for visible light and thermal infrared spectra. Additionally, the computational time is significantly reduced by utilizing fast Fourier transform. Moreover, a weighted fusion mechanism is employed to compute the final response map during the inference phase. In addition, there are also tracking methods that are based on CNN feature fusion. For example, Zhang et al. [13] proposed a fusion tracking method based on the idea of MDNet [5], which utilizes a parallel structure to separately process visible light and thermal infrared images using two shallow CNNs. However, this method does not consider modal weights when combining visible light and thermal infrared features. In contrast, Li et al. [14] proposed a dual-stream fusion network that merges the most effective features generated by two sets of convolutional networks. This method uses one CNN in the dual-stream network for feature extraction from thermal images and another for processing visible light images. The designed FusionNet is used for adaptive fusion of the two modalities and noise reduction. However, this method may not meet real-time requirements and does not take into account the reliability of visible light and thermal infrared images. While these RGBT trackers have contributed to the advancement of RGBT tracking, they often neglect the nature of feature interaction during the learning process, which could potentially hinder further improvement in tracking performance.

C. Transformer mechanism

The Transformer [15] is introduced in natural language processing and later adapted for computer vision tasks. It has demonstrated strong global reasoning abilities due to its self-attention and cross-attention mechanisms in various computer vision tasks. Carion et al. [16] propose an end-to-end target detector called DETR, which uses the Transformer self-attention mechanism to directly output the final detection results, and successfully applies Transformer to the target detection task. Furthermore, attempts have been made to incorporate Transformer-based approaches into the field of visual tracking. Chen et al. [17] applied Transformer in visual tracking tasks, modeling the relationship between template frames and search frames. In TransT [17], a combination of self-attention and cross-attention modules in Transformer

is used to predict the target location, instead of traditional correlation operations as seen in most siamese network-based trackers. ToMP [18] is a tracker architecture that utilizes a Transformer-based prediction module, which captures global relationships with minimal inductive bias, leading to a more powerful target model prediction capability. This approach significantly enhances the expressive power of the tracking network, addressing the limitations of optimization-based methods. This study marks the beginning of our endeavors to integrate the Transformer into RGBT tracking algorithms.

III. The Proposed Algorithm

A. Overview of the Network Architecture

We propose a method for visual target tracking based on the fusion of visible and thermal infrared (RGB-T) modalities, as depicted in Fig. 2. The method consists of a test branch and a training branch, where the latter is responsible for feature extraction and initialization of the target tracking, and the former is updated over time to track the target. Both branches use the ResNet feature extraction network to extract features from the RGB and TIR modalities, respectively, and then fuse the features of the two modalities using *Concat* operation. The target state information in the training frame is encoded and fused with the depth image feature. Similarly, the test frame is also encoded to identify it as a test frame. The features from the training and test branches are jointly processed in the Transformer encoder, which generates enhanced features through cross-frame global reasoning. The Transformer decoder uses the output of the Transformer encoder to predict the target model weights. A linear layer is then applied to the predicted weights to produce target classification weights and bounding box regression weights. Finally, the target classification weights are utilized to localize the target in the enhanced test frame features, and the bounding box regression branch uses the enhanced test frame and bounding box regression weights for bounding box regression. The tracking process's main components are described in detail in the subsequent sections.

B. Target Encoding

While a target model can locate the center of the target in each frame, a tracker also needs to estimate the precise bounding box of the target. To achieve this, the model is extended by encoding both the target center position and size information as the target state encoding to provide richer input to the model predictor. Furthermore, the model predictor is also extended to estimate the weights of the bounding box regression network in addition to the target model weights. The resulting tracking architecture, as shown in Fig. 2, can better estimate the target's precise bounding box. In training frames, the target state information is encoded into target localization and target range encodings. The target localization encoding enables the model predictor to consider the target state information in training frames when predicting the target model, while the target range encoding contains information about the target bounding box. These two encoding methods are then fused with the deep image features.

The target localization encoding enables the model predictor to incorporate target state information from the training frame when predicting the target model. In this study, the target encoding function is defined using an embedding $e_{fg} \in R^{1\times C}$ representing the foreground and a Gaussian $v_i \in R^{H\times W\times 1}$ centered on the target position. The target encoding function is defined as follows:

$$\phi(v_i, e_{fg}) = v_i \cdot e_{fg}, \tag{1}$$

where $\cdot$ denotes pointwise multiplication with broadcasting. It is worth noting that $H_{im} = s \cdot H$ and $W_{im} = s \cdot W$ correspond to the spatial dimensions of image patches, where s is the stride of the backbone network used for extracting depth features $x \in R^{H\times W\times C}$. The embedding vector e_{fg} is a 256-dimensional learnable vector. Next, we combine the target encoding with the depth image feature x_i as follows:

$$y_i = x_i + \psi(v_i, e_{fg}), \tag{2}$$

which provides the training frame feature $y_i \in R^{H\times W\times C}$ that includes the encoded target state information.

To incorporate information about the target bounding box, in addition to the depth map feature x_i and target position encoding $\psi(y_i, e_{fg})$, we use a method called target range encoding. The *ltrb* representation is used to encode the bounding box $b_i = \{b_i^x, b_i^y, b_i^w, b_i^h\}$ containing the target object.

To implement this encoding method, we map each position (j^x, j^y) on the feature map x_i back to the image domain using $(k^x, k^y) = (\lfloor s/2 \rfloor + s\cdot j^x, \lfloor s/2 \rfloor + s\cdot j^y)$. Then, we compute the normalized distance from each remapped position to the four edges of the bounding box b_i using the following formulas:

$$\begin{cases} l_i = (k^x - b_i^x)/W_{im}, \\ r_i = (k^x - b_i^x - b_i^w)/W_{im}, \\ t_i = (k^y - b_i^y)/H_{im}, \\ b_i = (k^y - b_i^y - b_i^h)/H_{im}, \end{cases} \tag{3}$$

where $W_{im} = s \cdot W$ and $H_{im} = s \cdot H$. We use these four edges to generate a dense bounding box representation, i.e., $d = (l, t, r, b)$, where $d \in R^{H\times W\times 4}$. To further improve the performance of the target tracking method, we use a multilayer perceptron (MLP) to encode the bounding box, increasing its dimension from 4 to C, and then add the resulting encoding to formula 2. Let ϕ denote MLP, then y_i can be denoted as follows:

$$y_i = x_i + \psi(v_i, e_{fg}) + \phi(d_i), \tag{4}$$

where y_i is the obtained feature map used as the input to the Transformer encoder, as shown in Fig. 2. The MLP for target range encoding consists of three layers, with a structure of 4 $\rightarrow$ 64 $\rightarrow$ 256 $\rightarrow$ 256. Each layer contains three steps: linear projection, batch normalization, and ReLU activation (except for the last layer, which only contains linear projection). Using this MLP, the target range can be encoded as a 256-dimensional vector. This vector can be used as input to the

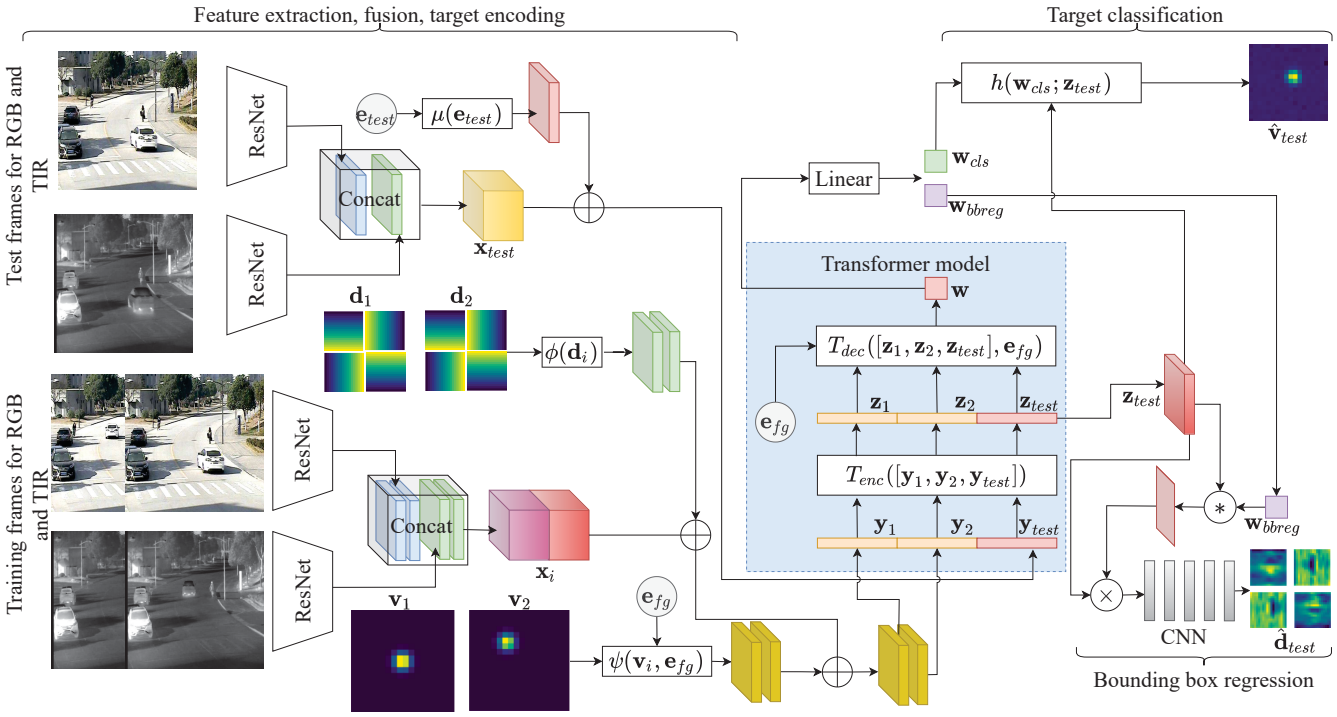

Fig. 2. Transformer-based dual-modal fusion visual target tracking network architecture

target tracking method for tracking and position estimation of the target.

We also introduce a test encoding scheme to identify the features corresponding to the test frames, which is formulated as follows:

$$y_{test} = x_{test} + \mu(e_{test}), \tag{5}$$

where $\mu(\cdot)$ encodes the patch e_{test} onto each token of x_{test}. A token refers to a discrete unit obtained by segmenting the input sequence, with each unit representing a specific meaning or information. The patch e_{test} is represented by a 256-dimensional learnable embedding vector. The next section introduces a Transformer-based architecture for predicting the target model.

C. Transformer model architecture

The Transformer model utilized in this paper comprises an encoder and a decoder. The encoder comprises N encoder layers, and the decoder is likewise composed of M decoder layers. Each encoder layer and decoder layer contains several sub-layers that receive the output of the previous layer as input and generate their output. By adding more encoder and decoder layers, a more profound Transformer network can be created to enhance the model's performance further. The architecture of the Transformer encoder and decoder is demonstrated in Fig. 3.

The encoder layer consists of two sub-layers, namely the multi-head self-attention mechanism (MHA) and the fully connected feed-forward network (FFN). The MHA learns the dependencies between different positions in the input sequence, while the FFN performs non-linear transformations and modeling of features at each position. To alleviate the gradient vanishing problem caused by the depth of the model, residual connections and layer normalization are applied to each sub-layer. Specifically, the output of each sub-layer is added element-wise to its input, and then passed through a layer normalization operation, which can be represented as $\text{LayerNorm}(\mathbf{X} + \text{Sublayer}(\mathbf{X}))$. Here, Sublayer represents the function of the sub-layer, LayerNorm is the layer normalization operation, and $+$ denotes the residual connection.

The structure of the decoder layer is similar to that of the encoder layer, consisting of three sub-layers. The first sub-layer is a multi-head self-attention mechanism, the second sub-layer is a multi-head attention mechanism, and the third sub-layer is a fully connected feed-forward network (FFN). The structures of the multi-head attention mechanism sub-layer and the multi-head self-attention mechanism sub-layer are the same, but the inputs are different. Like the encoder layer, each sub-layer adopts residual connections and is followed by a layer normalization operation. To prevent the model from attending to subsequent positions at the current position, the original Transformer introduced a masking mechanism to modify the self-attention sub-layer of the decoder layer. As the method used in this study only performs single-frame prediction, where the input of the decoder is from the start token encoding, the first sub-layer of the decoder layer is the same as the first sub-layer of the encoder layer.

The encoder processes the concatenated training and test features $\mathbf{Y} \in \mathbb{R}^{N \times d}$ using the multi-head self-attention mechanism. This mechanism allows for interaction between features at different positions and captures global information. The

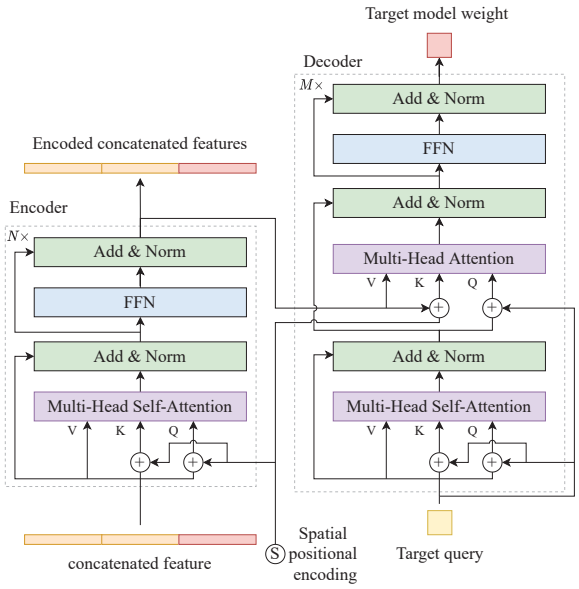

Fig. 3. Transformer model architecture

query, key, and value matrices $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{n \times d}$ have n heads and a feature dimension of d. The output $\mathbf{Z}$ of the multi-head self-attention mechanism can be obtained using the following equation:

$$\hat{\mathbf{Z}} = \text{MHA}(\mathbf{Y}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_n)\mathbf{W}^O, \quad (6)$$

where head_i refers to the i-th head, $\mathbf{W}^O \in \mathbb{R}^{d \times d}$ is the output's linear transformation matrix, and Concat concatenates the outputs of all the heads. Each head performs a scaled dot-product attention operation, and its calculation can be expressed as follows:

$$\text{head}_i = \text{Att}(\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i) = \text{Softmax}(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{d_k}})\mathbf{V}_i, \quad (7)$$

where d_k is one dimension of the key matrix, and $\mathbf{Q}_i$, $\mathbf{K}_i$, and $\mathbf{V}_i$ are the query, key, and value matrices of the i-th head obtained by linear transformations of $\mathbf{Y}$.

The encoder's fully connected feed-forward network sublayer is a non-linear transformation used to enhance the model's expressive power. Let $\hat{\mathbf{Z}} \in \mathbb{R}^{N \times d}$ denote the output of the multi-head self-attention mechanism. The output $\mathbf{Z}$ of the fully connected feed-forward network sublayer can be calculated using the following equation:

$$\mathbf{Z} = \text{FFN}(\hat{\mathbf{Z}}) = \text{ReLU}(\hat{\mathbf{Z}}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \quad (8)$$

where $\mathbf{W}_1$, $\mathbf{W}_2$, $\mathbf{b}_1$, and $\mathbf{b}_2$ are the weights and biases of the fully connected layer, respectively. This sublayer maps the features $\hat{\mathbf{z}}$ at each position to a new set of features $\mathbf{z}$.

The multi-head self-attention sublayer in the decoder has the same structure as that in the encoder, but it takes $\hat{\mathbf{E}} \in \mathbb{R}^{1 \times d}$ as input, passes it through a multi-head self-attention mechanism sublayer, and outputs $\mathbf{E} \in \mathbb{R}^{1 \times d}$. This output is then concatenated with the encoder's output, i.e., object feature vectors $\mathbf{Z} \in \mathbb{R}^{N \times d}$, and fed into another multi-head attention mechanism sublayer. After obtaining the output of the multi-head attention mechanism, the model further processes it through a feed-forward network (FFN), which has the same structure as that in the encoder layer. The output of the FFN is a matrix $\mathbf{w} \in \mathbb{R}^{1 \times d}$ which is the obtained target model.

D. Transformer-based target model prediction

This study aims to utilize the foreground and background information of both training and test frames to predict the target model. For this purpose, the Transformer encoder module explained above, is employed to jointly process the features of both training and test frames. The Transformer encoder has two main functions in our approach. Firstly, it computes the features required by the Transformer decoder module to predict the target model. Secondly, it generates improved test frame features that are utilized as input to the target model for detecting the target.

To achieve the goal, multiple encoded training features $y_i \in R^{H \times W \times C}$ and an encoded test feature $y_{test} \in R^{H \times W \times C}$ are reshaped into $R^{(H \cdot W) \times C}$ and concatenated along the first dimension. The concatenated features, which consist of all m training features y_i and the test feature y_{test}, are jointly processed in the Transformer encoder as follows:

$$[z_1, \ldots, z_m, z_{test}] = T_{enc}([y_1, \ldots, y_m, y_{test}]). \quad (9)$$

The Transformer encoder module includes multi-head self-attention modules, enabling it to perform global reasoning across the entire frame, even across multiple training and test frames. Moreover, the encoded target state can differentiate between foreground and background regions, allowing the Transformer to distinguish between these two regions.

The Transformer encoder's output, namely z_i and z_{test}, is utilized as an input to the Transformer decoder to predict the target model weights, denoted as $\mathbf{w}$:

$$\mathbf{w} = T_{dec}([z_1, \ldots, z_m, z_{test}], e_{fg}). \quad (10)$$

It is important to note that the inputs z_i and z_{test} are obtained through joint inference over the entire training and test samples, allowing for the prediction of a discriminative target model in this study. Additionally, we use the same learned foreground embedding e_{fg} as the input query to the Transformer decoder when predicting the target model weights $\mathbf{w}$.

E. Regression and Classification

The output $\mathbf{w}$ from the Transformer decoder is passed through a linear layer that uses a 1×1 convolutional kernel with 256 channels. This process yields two sets of weights: w_{bbreg} for bounding box regression and w_{cls} for target classification.To obtain the response map, we utilize the discriminative correlation filter method, which is expressed as:

$$h(w_{cls}; z_{test}) = w_{cls} * z_{test}. \quad (11)$$

For bounding box regression, we utilize the weight w_{bbreg} to conditionally constrain the output test feature z_{test} of the Transformer encoder with target information. This is achieved by applying w_{bbreg} as a weight matrix to z_{test}, resulting in a

refined feature representation used for bounding box regression. To imbue target awareness into the output feature z_{test} of the encoder, we take the following steps. First, we compute the attention map $w_{bbreg} * z_{test}$ using the predicted weight w_{bbreg}. Next, we point-wise multiply the attention weight with the test feature z_{test} and input the resulting product into a five-layer convolutional neural network. The network carries out bounding box prediction on the final convolutional layer and utilizes the exponential activation function to ensure a positive number output. This number is then normalized to achieve the same *ltrb* representation of the bounding box prediction, as illustrated in Equation 3. To obtain the final bounding box estimate, we use the $\arg\max(\cdot)$ function on the predicted target score map $\hat{v}_{test}$ to extract the center position and then query the dense bounding box prediction $\hat{d}_{test}$ at the center position of the target object to obtain the bounding box. Two dedicated networks are employed for target localization and bounding box regression to decouple these two tasks during tracking.

F. Loss Function

In the proposed method, there are two sub-tasks, and therefore, the loss function consists of a classification loss and a regression loss as follows:

$$L_{tot} = \lambda_{cls} L_{cls}(\hat{v}, v) + \lambda_{giou} L_{giou}(\hat{d}, d), \tag{12}$$

where λ_{cls} and λ_{giou} are scalar weights that balance the contribution of each loss. In this work, we set them to $\lambda_{cls} = 100$ and $\lambda_{giou} = 1$.

The target classification loss used in this study is a variant of hinge loss. It penalizes the model for misclassifying samples during classification and helps improve decision boundary learning. The loss function is given by

$$L_{cls}(s, z) = \begin{cases} s - z, & z > \tau, \\ max(0, s), & z \leq \tau, \end{cases} \tag{13}$$

where the threshold τ defines the target and background regions based on the label's confidence value z. If $z > \tau$, the predicted confidence score s and label z difference is calculated for the target region. For the background region where $z \leq \tau$, only positive confidence values are penalized.

The Generalized Intersection over Union (GIoU) loss function is used in this study to supervise bounding box regression with *ltrb* bounding box representation [19]. This function, denoted as L_{giou}, is a more accurate measure of similarity between bounding boxes, thereby guiding model training more effectively. The GIoU loss function is defined as follows:

$$L_{giou} = 1 - \text{GIoU}(B_{pred}, B_{gt}), \tag{14}$$

where B_{pred} and B_{gt} represent the predicted and ground-truth bounding boxes, respectively.

IV. Implementation Details

A. Training Details

We sample multiple training and test frames from video sequences to form training subsequences. Specifically, we use two training frames and one test frame. The training and test frames are kept at the same resolution and resized to $288 \times 288 \times 3$ after being sampled and cropped from the dataset. Each image I_i is paired with its corresponding bounding box b_i. The object's target state is encoded using the training frames, and only the bounding boxes from the test frames are used to supervise the training by computing two loss functions based on the predicted bounding boxes and the target object's center position in the test frame.

The feature extraction backbone network used in this study is a modified ResNet-50 network that is initialized with pre-trained weights on the ImageNet dataset to improve training speed and model performance. During training, we generate the target states v by using a Gaussian function with a standard deviation of $1/4$ relative to the base target size, and set $\tau = 0.05$ to differentiate between foreground and background regions in the corresponding classification loss l_{cls}. The model predictor extracts features with a stride of $s = 16$ from the third block of ResNet. As the input channel dimension of the Transformer used in this study is 256 and ResNet features have 1024 channels, the channel number doubles after fusing the two modal features. Therefore, before inputting features to the Transformer encoder, a convolutional layer is used to reduce the number of channels. The Transformer encoder consists of multiple layers, each containing multi-head self-attention and a feed-forward network. We employ 8 heads and 2048 hidden dimensions as the feed-forward network, and use dropout with a probability of 0.1 and layer normalization. The proposed tracker is trained on the training set split of the LasHeR dataset [20] by sampling 40k subsequences and performing training for 300 iterations. The ADAMW optimizer with a learning rate of 0.0001 is used for optimization.

B. Tracking Details

The target model in this study generates the expected target state $y_i \in \mathcal{Y}$ for the training samples $S_{train} \in (x_i, y_i)_{i=1}^{m}$, where $x_i \in \chi$ represents the depth feature map of the i-th frame, and $m = 2$ is the total number of training frames. During the tracking process, we use the annotated first frame and the previously tracked frames as the training set S_{train}. Although the initial frame and its annotations are always kept, we also use a previously tracked frame and replace it with the latest frame. According to the ablation experiment results in this paper, the replacement condition is that the tracking method works best when the target classifier confidence score is above a threshold of 0.85.

Incorporating the previous tracking results into S_{train} significantly improves target localization. However, including bounding box estimates with predictions reduces the performance of bounding box regression due to inaccurate predictions. Therefore, we run the model predictor twice. First, this study includes intermediate predictions in S_{train} to obtain classifier weights. In the second run, we only use the annotated initial frame to predict the bounding boxes.

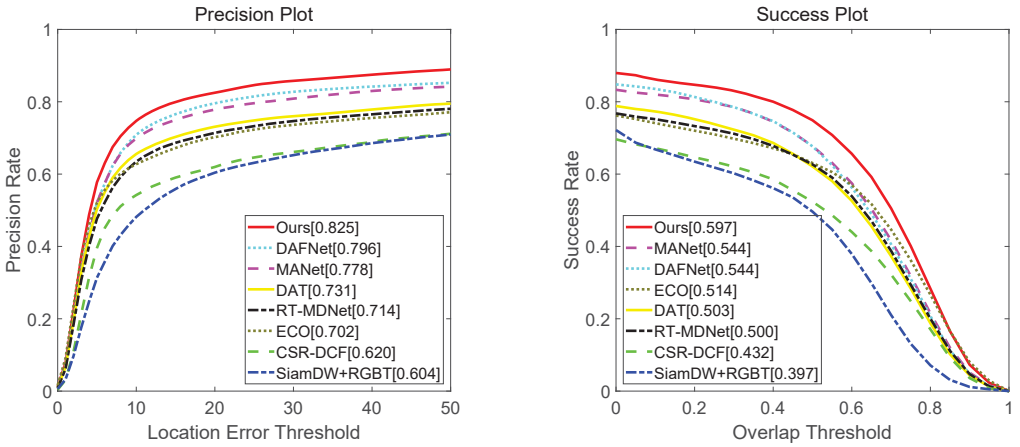

Fig. 4. Performance evaluation on RGBT234 in terms of success and precision plots

V. Experimental Results

This section presents the experimental results of the proposed method on the RGBT234 dataset [1] in comparison with other RGBT tracking methods. Furthermore, the performance of the trackers is analyzed on various attributes. Finally, the effectiveness of the proposed tracker is qualitatively verified through visualization results of different trackers. Experiments were conducted on hardware including Intel i9-10900K 3.70 GHz CPU and Nvidia GeForce RTX 3090 GPU, with software environments including Ubuntu 20.04 64-bit operating system, PyTorch, and Python.

A. Evaluation on RGBT234

The comparisons were conducted based on the one pass evaluation (OPE) rule with Success Rate (SR) and Precision Rate (PR) metrics. The performance of various trackers, including DAFNet [21], RT-MDNet [22], DAT [23], ECO [24], CSR-DCF [25], MANet [26], SiamDW+RGBT [10], were compared. As shown in Fig. 4, it can be observed that the proposed tracker achieves the best performance among all the trackers on the RGBT234 dataset with PR/SR reaching 82.5%/59.7%, respectively. Specifically, compared to DAFNet, which is the second strongest tracker in SR, the performance of the proposed tracker is improved by 4.3%. Furthermore, compared to the second-ranked tracker MANet, the proposed tracker improves PR by about 2.9%. These results demonstrate that the proposed tracker is highly competitive.

B. Quantitative

The performance of tracking methods is observed in test video sequences, and the method's effectiveness is analyzed through image descriptions. Fig. 5 showcases the visualization results of our method and five other trackers, namely MDNet [27], SGT [28], SOWP [29], ECO [24], and SOWP+RGBT [29], in various challenging scenarios. Our method's results are depicted by red bounding boxes, while the other five methods' results are represented by bounding boxes of different colors. The yellow number present in the upper right corner of each image represents the frame number of the image in the sequence. Objects being tracked in test video sequences, including cyclists (sequence *bikeman*), cars (sequence *car41*) and electric bicycles (sequence *elecbike*), which are crucial targets or vulnerable road users in autonomous driving scenarios. The visualization results clearly demonstrate that the proposed method outperforms these state-of-the-art methods and meets the requirements of driving scenarios.

TABLE I. EFFECT OF η ON THE AREA UNDER THE SUCCESS RATE CURVE IN RGBT234

η	0.75	0.8	0.85	0.9	0.95
AUC	59.1%	59.4%	**59.7%**	59.5%	59.3%

C. Ablation Study

To determine whether previous tracking results were used as training frames, we used the maximum value of the target score map generated by the target model as the basis. Specifically, if the confidence value of the sample is higher than a certain threshold η, the sample is selected. We selected different values of η, including 0.75, 0.8, 0.85, 0.9, and 0.95. The area under the success rate curve was used as the evaluation metric to determine the optimal η value. The experimental results are shown in Table I. On RGBT234, the experimental results show that selecting a threshold of 0.85 can achieve good performance. In addition, if the maximum value of the target score map is less than 0.25, it is considered that the target is not found. During inference and training, we used the same target score map spatial resolution (18×18) and search region scaling factor (5.0). In conclusion, selecting the appropriate threshold to obtain corresponding training frames is very important in the proposed target tracking method and can significantly affect tracking performance.

VI. Conclusion

In this work, we have addressed the challenge of limited visible light information in low-light conditions by fully

Fig. 5. Visualization of tracking results of different methods on three sequences of RGBT234. The first, third, and fifth lines represent the visible light modality of the sequences *bikeman*, *car41*, and *elecbike*, respectively. The second, fourth, and sixth lines represent the thermal infrared modality of the sequences *bikeman*, *car41*, and *elecbike*, respectively.

exploiting the supplementary information provided by thermal infrared. Specifically, we have proposed a dual-modal visual target tracking framework based on Transformers that effectively utilizes both visible light and thermal infrared. Our proposed tracker utilizes a Transformer encoder-decoder structure to improve the performance of traditional convolutional models. We have used a modified ResNet-50 as the backbone network to extract features. By fusing the training and test branches of the dual-modality information, we have generated enhanced features that leverage global reasoning through the Transformer encoder, combined with target encoding information from both branches. We have employed the Transformer decoder to predict the target model weights, which are then acted upon by a linear layer to generate target classification and bounding box regression weights. Finally, we have utilized the enhanced test frame and target classification weights to achieve object localization, while the bounding box regression weights are used to achieve bounding box regression. The proposed method has achieved state-of-the-art performance in both quantitative and qualitative analysis on various publicly available datasets.

ACKNOWLEDGMENT

Thanks for the Nanjing University of Aeronautics and Astronautics PhD international academic communication funding.

REFERENCES

[1] C. Li, X. Liang, Y. Lu, N. Zhao, and J. Tang, "Rgb-t object tracking: Benchmark and baseline," *Pattern Recognition*, vol. 96, p. 106977, 2019.

[2] L. Zhang, M. Danelljan, A. Gonzalez-Garcia, J. Van De Weijer, and F. Shahbaz Khan, "Multi-modal fusion for end-to-end rgb-t tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, Seoul, Korea (South), October 27-28, 2019, pp. 2252–2261.

[3] S. Zhai, P. Shao, X. Liang, and X. Wang, "Fast rgb-t tracking via cross-modal correlation filters," *Neurocomputing*, vol. 334, pp. 172–181, 2019.

[4] X. Zhang, P. Ye, S. Peng, J. Liu, K. Gong, and G. Xiao, "Siamft: An rgb-infrared fusion tracking method via fully convolutional siamese networks," *IEEE Access*, vol. 7, pp. 122 122–122 133, 2019.

[5] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, Las Vegas, NV, USA, June 27–30, 2016, pp. 4293–4302.

[6] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *Proceedings of the European Conference on Computer Vision*, Amsterdam, The Netherlands, October 8–10 and 15–16, 2016, pp. 850–865.

[7] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, USA, June 18–22, 2018, pp. 8971–8980.

[8] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware siamese networks for visual object tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany, September 8–14, 2018, pp. 101–117.

[9] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 16–20, 2019, pp. 4282–4291.

[10] Z. Zhang and H. Peng, "Deeper and wider siamese networks for real-time visual tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 16–20, 2019, pp. 4591–4600.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 27–30, 2016, pp. 770–778.

[12] Y. Wang, C. Li, and J. Tang, "Learning soft-consistent correlation filters for rgb-t object tracking," in *Pattern Recognition and Computer Vision: First Chinese Conference, PRCV 2018, Proceedings, Part IV 1*. Guangzhou, China: Springer, November 23–26, 2018, pp. 295–306.

[13] X. Zhang, X. Zhang, X. Du, X. Zhou, and J. Yin, "Learning multi-domain convolutional network for rgb-t visual tracking," in *11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*. Beijing, China: IEEE, October 13–15, 2018, pp. 1–6.

[14] C. Li, X. Wu, N. Zhao, X. Cao, and J. Tang, "Fusing two-stream convolutional neural networks for rgb-t object tracking," *Neurocomputing*, vol. 281, pp. 78–85, 2018.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[16] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision–ECCV 2020: 16th European Conference*. Glasgow, UK: Springer, August 23–28, 2020, pp. 213–229.

[17] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, "Transformer tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, vVirtual (Online), June 19–25, 2021, pp. 8126–8135.

[18] C. Mayer, M. Danelljan, G. Bhat, M. Paul, D. P. Paudel, F. Yu, and L. Van Gool, "Transforming model prediction for tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, New Orleans, LA, USA, June 18–24, 2022, pp. 8731–8740.

[19] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 16–20, 2019, pp. 658–666.

[20] C. Li, W. Xue, Y. Jia, Z. Qu, B. Luo, J. Tang, and D. Sun, "Lasher: A large-scale high-diversity benchmark for rgbt tracking," *IEEE Transactions on Image Processing*, vol. 31, pp. 392–404, 2021.

[21] Y. Gao, C. Li, Y. Zhu, J. Tang, T. He, and F. Wang, "Deep adaptive fusion network for high performance rgbt tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, Seoul, Korea (South), October 27-28, 2019, pp. 91–99.

[22] I. Jung, J. Son, M. Baek, and B. Han, "Real-time mdnet," in *Proceedings of the European conference on computer vision (ECCV)*, Munich, Germany, September 8-14, 2018, pp. 83–98.

[23] S. Pu, Y. Song, C. Ma, H. Zhang, and M.-H. Yang, "Deep attentive tracking via reciprocative learning," *Advances in neural information processing systems*, vol. 31, 2018.

[24] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, "Eco: Efficient convolution operators for tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, July 21–26, 2017, pp. 6638–6646.

[25] A. Lukezic, T. Vojir, L. ˇCehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, HI, USA, July 21-26, 2017, pp. 6309–6318.

[26] C. Long Li, A. Lu, A. Hua Zheng, Z. Tu, and J. Tang, "Multi-adapter rgbt tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, Seoul, Korea (South), October 27-28, 2019, pp. 2262–2270.

[27] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, USA, June 27–30, 2016, pp. 4293–4302.

[28] C. Li, N. Zhao, Y. Lu, C. Zhu, and J. Tang, "Weighted sparse representation regularized graph learning for rgb-t object tracking," in *Proceedings of the 25th ACM international conference on Multimedia*, Mountain View, CA, USA, October 23-27, 2017, pp. 1856–1864.

[29] H.-U. Kim, D.-Y. Lee, J.-Y. Sim, and C.-S. Kim, "Sowp: Spatially ordered and weighted patch descriptor for visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, December 7-13, 2015, pp. 3011–3019.