IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 8 October 2023, accepted 24 October 2023, date of publication 6 November 2023,
date of current version 14 December 2023.

*Digital Object Identifier 10.1109/ACCESS.2023.3330644*

RESEARCH ARTICLE

# Enhancing Single Object Tracking With a Hybrid Approach: Temporal Convolutional Networks, Attention Mechanisms, and Spatial–Temporal Memory

**PIMPA CHEEWAPRAKOBKIT[1,2], CHIH-YANG LIN[3], (Senior Member, IEEE), TIMOTHY K. SHIH[1], (Senior Member, IEEE), and AVIRMED ENKHBAT[1]**

[1]Department of Computer Science and Information Engineering, National Central University, Taoyuan 32001, Taiwan
[2]Department of Information Technology, Asia–Pacific International University, Saraburi 18180, Thailand
[3]Department of Mechanical Engineering, National Central University, Taoyuan 32001, Taiwan

Corresponding authors: Chih-Yang Lin (andrewlin@ncu.edu.tw) and Timothy K. Shih (timothykshih@gmail.com)

**ABSTRACT** Deep neural network-based tracking tasks have experienced significant advancements in recent years. However, these networks continue to face challenges in effectively adapting to appearance changes in both target and background, as well as linking objects after extended periods. The primary challenge in tracking lies in the frequent changes in a target's appearance throughout the tracking process, which can potentially reduce tracker robustness when faced with issues such as aspect ratio changes, occlusions, scale variations, and confusion from similar objects. To address this challenge, we propose a tracking architecture that combines a temporal convolutional network (TCN) and attention mechanism with spatial-temporal memory. The TCN component empowers the model to capture temporal dependencies, while the attention mechanism reduces computational complexity by focusing on crucial regions based on context. We leverage the target's historical information stored in the spatial-temporal memory network to guide the tracker in better adapting to target deformation. Our model attains a 67.5% average overlap (AO) on the GOT-10K dataset, a 72.1% success score (AUC) on OTB2015, a 65.8% success score (AUC) on UAV123, and achieves 59.0% accuracy on the VOT2018 dataset. These outcomes demonstrate the high effectiveness of our proposed tracker in tracking a single object.

**INDEX TERMS** Temporal convolutional networks, attention mechanism, spatial-temporal memory, single object tracking.

## I. INTRODUCTION

Computer vision research places significant importance on single object tracking due to its wide-ranging practical applications in computer interaction, monitoring, robotics, and autonomous driving. The objective of single object tracking is to locate and track a specific target in video sequences, given the initial target's position in the first frame. Despite considerable advancements in recent years, object tracking remains a complex challenge due to various factors, such as occlusion, distractors, motion, target deformation, similar objects, and background clutter. To obtain more accurate and robust tracking results, some state-of-the-art trackers employ a multi-stage tracking strategy involving additional tracking stages to achieve precise bounding box estimation. These trackers first identify the target's rough location and then fine-tune the results through additional tracking stages to achieve a more precise bounding box prediction.

While the others employ the template matching method, which involves identifying areas of the target template image

The associate editor coordinating the review of this manuscript and approving it for publication was Zhongyi Guo.

that match a search image. The representative of the template matching method is the Siamese tracker [1], which relies on a two-branch neural network that processes a template image and a search image using a convolutional neural network to generate a response map indicating the target's location. However, this method does not utilize spatial-temporal information and update the template, thus is hard to adapt to appearance changes during tracking, leading to tracking failures.

The incremental template update strategy [2] was proposed to maintain accurate and robust tracking of an object by continuously updating multiple templates that represent the object's appearance and environment. However, the approach may face limitations when tracking objects undergo significant changes in appearance or motion over time, as the incremental update strategy assumes that the changes in the object's appearance are gradual and can be captured by updating the existing templates. In scenarios with abrupt changes or occlusions, the approach may not be able to maintain accurate tracking. Additionally, the approach may not be suitable for tracking multiple objects simultaneously, as the templates would need to be updated for each object separately, leading to increased computational complexity.

The method for capturing the appearance and motion patterns of an object is implemented in the Temporal Convolutional Network (TCN) models [3]. TCN utilizes 1D convolutional layers with dilated filters to capture temporal dependencies across multiple time steps. It can also incorporate spatial information by treating the input data as a 1D sequence of feature vectors, where each vector corresponds to a spatiotemporal location in the input. This allows the network to capture both spatial and temporal dependencies in the data. However, TCN is designed to handle sequential data with fixed length and regular patterns, which means it may not be well-suited for tasks that require modeling complex dependencies [4].

Later on, Lai et al. [5] introduced a memory buffer network that can selectively store and retrieve relevant information about the target object, such as its appearance, motion, and context from past frames to improve object tracking performance. This approach utilizes the stored information to generate a set of candidate object locations and adaptively updates the tracking model over time. However, the method has some limitations. It can be computationally expensive and requires a large amount of training data to achieve high accuracy. Additionally, it is highly sensitive to the quality and quantity of input data, which can lead to overfitting or poor generalization to unseen data.

Inspired by prior works, this paper proposes a method that incorporates the Temporal Convolutional Network (TCN) with an attention mechanism to enhance its potential in modeling long-range patterns in videos. The proposed method also employs a historical network that relies on spatial-temporal information, replacing the conventional template-matching approach and eliminating the need for direct template updating. Furthermore, to ensure that the model captures the true target appearance instead of other interfering objects, a background label is added.

The main contributions of our work are as follows:

1) We introduce a tracking architecture that combines the Temporal Convolutional Network (TCN) and an attention mechanism with a spatial-temporal memory network for enhanced single-object tracking.

2) TCN is specifically designed to capture temporal dependencies in sequential data, enabling our model to handle temporal variations in object motion and appearance. Additionally, the attention mechanism is incorporated into the hidden layers of TCN to selectively focus on the most relevant parts of the input, reducing computational complexity and boosting the model's performance.

3) To ensure that the model captures the true target characteristics and not distractors, the background label is deliberately incorporated into the backbone network.

4) We utilize a spatial-temporal memory network that retains the past information of the target to guide the tracker in adjusting to the target's shape or movement.

5) We evaluate our proposed model on four benchmark datasets: GOT-10K, OTB2015, UAV123, and VOT2018.

The rest of this paper is organized as follows. Section II reviews the related work on object tracking. Section III presents the proposed method. Section IV illustrates the experimental results of the proposed method. Finally, section V presents the conclusions.

## II. RELATED WORKS

Extensive research has been conducted on single object trackers. The Siamese network structure [6] is one of the most popular. This approach involves combining two convolutional neural networks (CNNs) to achieve high accuracy while maintaining fast tracking speeds in real-time trackers. These trackers typically transform object tracking into a template matching problem by cropping a template image from the first frame during inference and matching it within the search image region in the current frame. They employ end-to-end training to acquire the object's feature representation. One major advantage of this method is the reduced need for online updates, which enables real-time tracking. However, most Siamese trackers [7] rely solely on the appearance features from the first frame and do not effectively leverage interframe information. Although this approach works well for tracking, it is weak in tracking objects that drift, especially in scenarios with cluttered backgrounds and occlusions, due to the lack of updating the target's appearance changes over time.

Lu et al. [8] introduced SiamFC (Siamese Fully Convolutional) to improve the architecture of Siamese networks by a fully convolutional neural network. SiamFC also incorporates a template suppression method to enhance the accuracy and robustness of the tracker. The template suppression technique removes background information from the template. This method is based on a segmentation network trained to identify foreground and background regions in the template.

The foreground regions are then used to generate a new template that only contains the object of interest [10]. The SiamFC tracker has a limitation where it cannot adjust to appearance changes or learn a new template during the tracking process. This limitation causes problems with template drifting because the templates obtained during tracking are not always reliable, which can lead to the accumulation of small errors over time [9]. Then Li et al. [10] presented the SiamRPN architecture, which comprises two identical networks that share weights. One network is responsible for generating region proposals, while the other network predicts a similarity score between the target and the proposed regions. The combination of these two networks enables efficient and accurate tracking of the target object. While the SiamRPN architecture has demonstrated good performance in object tracking, it can be slow to track objects that are moving quickly. This is because SiamRPN needs to compute the similarity between the target object and the search region at every frame, which can be computationally expensive.

According to the limitation of previous works that struggle to capture the target's appearance changes over a long period, leading to suboptimal performance. To address this limitation, memory networks were introduced as a solution by selectively storing and retrieving relevant information about the target object from past frames, allowing them to effectively capture and model appearance changes over longer sequences of frames. Zhou et al. [11] introduced a memory network to store and update the features of the tracked object, which enables the model to handle occlusions and re-detections of the object. The memory network is designed to store the object's spatiotemporal features separately, which allows the model to distinguish between the appearance and motion features of the object. Although memory network has shown promise in improving the performance of various tasks, they can be computationally expensive and require large amounts of training data. In our approach, we integrated the memory network to capture target information from past frames through the learning process of TCN and attention mechanism. This integration enhances the model's ability to concentrate on the most relevant parts, allowing it to learn effectively even with limited training data and achieve high accuracy.

TCN is one of the powerful architectures introduced by Bai et al. [12], which utilizes causal convolutions and dilations to capture long-range temporal patterns for sequence modeling tasks. TCN has been successfully used in wind power prediction by Zhu et al. [13], training the model on a sequence of historical wind power data and weather forecast data. The results demonstrate excellent ability and prediction accuracy compared to 1D-Convolutional Neural Networks (CNNs), Long Short-Term Memory (LSTM) networks, and Gated Recurrent Units (GRUs). However, this approach is limited to short-term forecasting, specifically predicting wind power output 1-6 hours ahead. He et al. [4] employed TCN to detect anomalies by learning a temporal representation of the time series that captures the normal patterns and detects deviations from those patterns. The model was trained on a large dataset of labeled time series and is able to generalize to new and unseen time series. However, this approach requires a large amount of labeled data for training, and it can be computationally expensive, especially for longer or higher-dimensional time series features. This could be a drawback for users who have limited computational resources or require real-time anomaly detection.

## III. PROPOSED METHOD

The overview of our proposed architecture as illustrated in Figure 1 consists of four main components: a backbone network, a Temporal Convolutional Network and attention mechanism, a spatial-temporal memory network, and a prediction network. The backbone network is split into two branches: a historical branch (depicted in blue) and a search branch (depicted in orange). The historical branch takes both historical frames and their corresponding background label as inputs, while the search branch takes a single search frame representing the current frame.

### A. BACKBONE NETWORK

The backbone network comprises two branches: the historical branch and the search branch.

#### 1) HISTORICAL BRANCH

We adopt GoogLeNet as the backbone for feature extraction. To incorporate the historical branch, we utilize historical frames $T$ along with their corresponding background label $B$. The background label $B$ contains 1 and 0 pixels within the ground truth of the target and background, respectively. To fuse the information from the historical frames and background label, we extract the feature of historical frame $T_i$ (denoted as $\theta^m$). Then, we combine the first convolutional layer of $\theta^m$ with the background label map $B_i$ using the element-wise addition operation. This allows the network to focus on the regions corresponding to the target while suppressing the background. The combined sum of these two elements is subsequently fed into the following layers of $\theta^m$, producing $T$ historical feature maps. Each historical feature map is represented as $f_{t-i}$, where $f_{t-i} \in R^{CxHxW}$. In this context, $C$ stands for the number of channels, while $H$ and $W$ correspond to the height and width of the feature map, respectively.

#### 2) SEARCH BRANCH

The search branch takes a search frame as input and feeds it into the backbone network for feature extraction, resulting in an output feature map $f^s$, where $f^s \in R^{CxHxW}$

### B. TEMPORAL CONVOLUTIONAL NETWORK AND ATTENTION MECHANISM

All the historical feature maps $f_{t-i}$, generated by the backbone network, are concatenated to form the output, which then serves as input to the temporal convolutional network and attention mechanism. The TCN employs a hierarchical

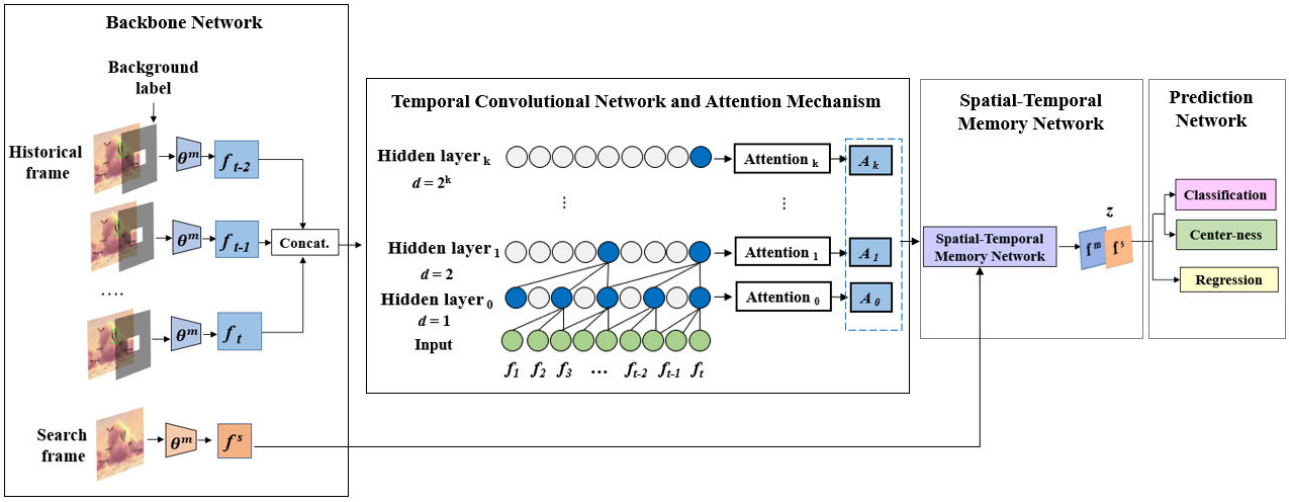


**FIGURE 1.** Overview of our proposed architecture.

convolutional architecture to capture long-term temporal patterns. The attention mechanism is integrated into the hidden layers of the TCN structure. Figure 1 illustrates an example of the temporal convolutional network and attention mechanism with one input layer and a kernel filter size of 3, using a dilation factor $d$ to represent the receptive field (depicted in blue cells) of the model. The attention layers' results (represented by vectors $A_0, A_1, \ldots, A_K$) are concatenated to form the output $f^m$.

The TCN structure incorporates causal convolution, dilated convolution, and residual connections [14].

#### 1) CAUSAL CONVOLUTIONS

In a causal convolutional layer, the output at each time step depends only on the input at the current and past time steps, and not on any future time steps. This ensures that the model can make predictions based solely on the past and not on the future. To maintain the same length between the input and hidden layers, zero padding is utilized in the hidden layers.

#### 2) DILATED CONVOLUTIONS

The dilated convolutions are employed to capture long-term dependencies and model large receptive fields. In a dilated convolutional layer, the filter is applied over a wider range than its original size by skipping input values with a specified step size, thus effectively expanding the receptive field. This is necessary because relying solely on causal convolutions would require an excessively deep network, which can be computationally expensive. The receptive field size $y$ is determined by the formula in (1).

$$y = \sum_{n=1}^{n} (k-1) \text{ x } d_n + 1, \tag{1}$$

where $n$ represents the number of hidden layers, and $k$ denotes the kernel size. The expansion factor of the $n^{th}$ hidden layer, denoted by $d_n$, is calculated based on the formula $d_n = 2^{n-1}$. The dilated causal convolution is shown in Figure 2.

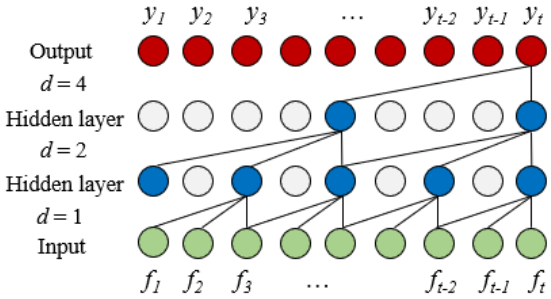


**FIGURE 2.** The dilated causal convolution.

#### 3) RESIDUAL CONNECTIONS

Residual connections have demonstrated remarkable effectiveness in training deep networks by allowing information to be transmitted across layers. In a residual network, skip connections are used throughout to accelerate the training process and circumvent issues such as gradient explosion or vanishing, despite the network being extremely deep. A residual connection comprises two branches, with the first branch containing two layers of dilated causal convolution with weight normalization and dropout layers, following ReLU activation. Meanwhile, the second branch is a shortcut that directly connects the input to the output of the convolutional layers by using 1D convolution to ensure that the dimensions of the output from both branches are equal and can be added together, as shown in Figure 3.

### C. ATTENTION MECHANISM

The attention mechanism [15] has proven to be a successful technique in both machine learning and natural language

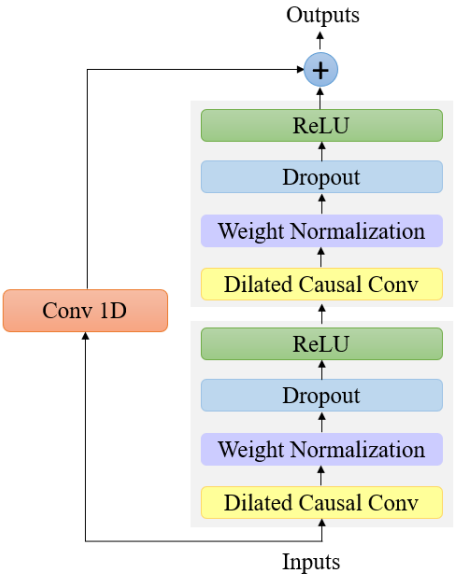


**FIGURE 3.** Residual connections.

processing. It relies on the context to determine which part of the data is more important than another. We incorporated this technique into the TCN structure to improve its performance. Figure 4 presents the attention mechanism flowchart, a constituent of the 'Temporal Convolution Network and Attention Mechanism' as depicted in Figure 1.

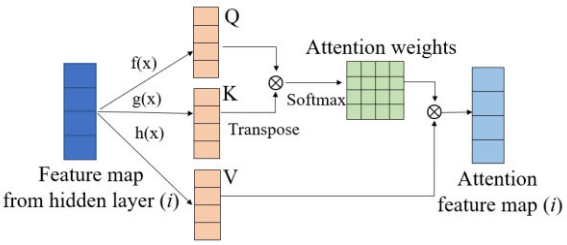


**FIGURE 4.** The attention mechanism flowchart.

To achieve this, we employ linear transformations denoted as $f$, $g$, and $h$ to map the feature map from the hidden layer $(i)$ of the TCN to three distinct vectors: key $(K)$, query $(Q)$, and value $(V)$. Following this, the attention weight is determined through matrix multiplication of $QK^T$, which calculates the dot product for each combination of queries and keys. This product is then divided by $\sqrt{d_k}$. The derived attention weight undergoes normalization using the softmax function and is subsequently multiplied with $V$ to derive the attention feature map. The attention function can be expressed in (2).

$$\boldsymbol{Attention}_i\,(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = \boldsymbol{Softmax}\left(\frac{\boldsymbol{Q}\cdot\boldsymbol{K}^{\boldsymbol{T}}}{\sqrt{\boldsymbol{d_k}}}\right)\cdot\boldsymbol{V} \quad (2)$$

where $T$ denotes the transpose matrix, $d_k$ represents the feature dimension of $K$, and $i$ represents the number of hidden layers in the TCN. The symbol $\otimes$ in the attention mechanism flowchart represents matrix multiplication.

### D. SPATIAL-TEMPORAL MEMORY NETWORK

The purpose of the spatial-temporal memory network [16] is to retrieve target information from historical frames and uses this information to generate a soft weight map $w$ in order to create a fused feature map $z$. This $z$ is then used to classify the target and distinguish it from the background in the current search frame, as well as to predict the target's location within the search frame. The spatial-temporal memory network is shown in Figure 5. To begin the process of the spatial-temporal memory network, the feature map $f^m$ and $f^s$ are first reshaped to a new dimensionality of 512, where $f^m$ is the output from the temporal convolutional network and attention mechanism, and $f^s$ is the output from the search branch in the backbone network.

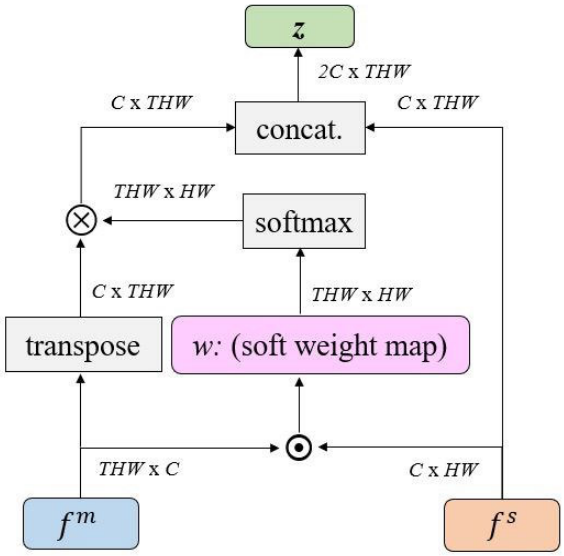


**FIGURE 5.** The spatial-temporal memory network.

The spatial-temporal memory network computes the similarities between every pixel of $f^m \in R^{THW\times C}$ and $f^s \in R^{C\times HW}$ to obtain a soft weight map $w \in R^{THW\times WH}$, where $T$ is the number of historical frames, and $C$, $H$, $W$ represent the number of channels, the height, and the width of the feature map, respectively.

To ensure proper scaling, we normalize $w$ using the softmax function [17]. The formula for one element $w_{ij}$ is shown in (3) as an example.

$$w_{ij} = \frac{exp\left[\left(f_i^m \odot f_j^s\right)/\sqrt{C}\right]}{\sum exp\left[\left(f_i^m \odot f_j^s\right)/\sqrt{C}\right]} \quad (3)$$

In this context, the variable $i$ represents the index of each pixel on $f^m$, while $j$ corresponds to the index of each pixel on $f^s$. The symbol $\odot$ represents the vector dot-product operation.

After transposing $f^m$, we proceed to multiply $f^m$ with $w$. This is because $f^m$ stores all historical information related to the target. By assigning weights to each element of $f^m$, the model is enhanced to selectively retrieve the most relevant target information stored in $f^m$ based on the current search frame. The output is a feature map that has the same size as $f^s$. To generate the fused feature map $z$, we concatenate the output from the multiplication of $f^m$ and $w$ with $f^s$. The equation for this process can be shown in (4). The symbol $\bigotimes$ represents the matrix multiplication, where $i$ represents the index of the element of $z$, and $T$ represents the transpose of $f^m$.

$$z_i = \boldsymbol{concat}\left(f_i^s, (f^m)_i^T \otimes w\right) \tag{4}$$

### E. PREDICTION NETWORK

The prediction network comprises two primary branches: a classification branch and a regression branch. The classification branch is further divided into two sections: the first is devoted to predicting class confidence, while the other focuses on determining the centerness of the object. To enhance the classification accuracy between the target object and the background, we introduce the regression network. This network supplies pertinent information that provides relevant information to enhance the classification branch.

#### 1) THE REGRESSION BRANCH

The fused feature map, denoted as $z$, serves as the input for the prediction network. The regression branch utilizes anchor-free regression to predict the object's location on the image at a pixel level. It achieves this by considering each pixel in the feature map $z$ as *training samples*, without relying on predefined anchor boxes, following a similar approach as [31]. If the location $(x, y)$ of a pixel falls within the boundaries of the ground-truth bounding box of the target object, it is classified as a positive sample with a label that corresponds to the class of the ground-truth label. Otherwise, it is treated as a negative sample, with is assigned to 0 (representing the background class)

Along with the classification label, we also generate a 4D vector $t^* = (l^*, t^*, r^*, b^*)$ representing the regression targets associated with a particular location. In this context, $(l^*, t^*, r^*, b^*)$ denote the distances from the location $(x, y)$ to the four edges of the bounding box, as illustrated in Figure 6. If the location $(x, y)$ aligns with the ground-truth bounding box $B$, we can define the training regression targets for that distinct location as follows:

$$\begin{aligned} l^* &= x - x_0, \quad & t^* &= y - y_0 \\ r^* &= x_1 - x, \quad & b^* &= y_1 - y \end{aligned} \tag{5}$$

For an input image, the bounding boxes corresponding to the ground-truth are represented as $B = (x_0, y_0, x_1, y_1, c) \in R^4 \mathrm{x}\{1, 2, \ldots, C\}$. In this notation, the coordinate pairs $(x_0, y_0)$ and $(x_1, y_1)$ indicate the positions of the top-left and bottom-right corners of the bounding box, respectively.

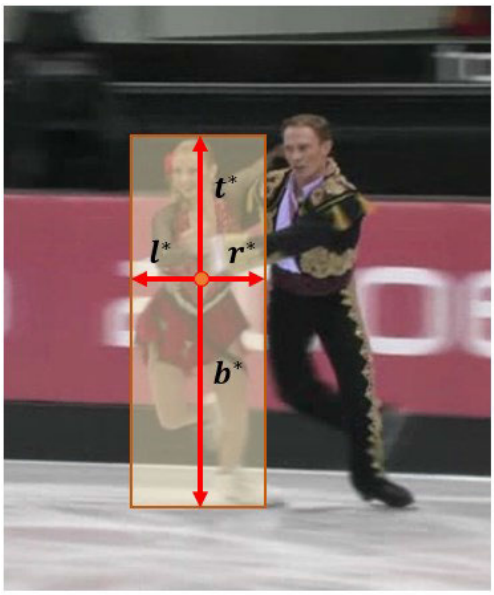


**FIGURE 6.** An example of a 4D vector $(l^*, t^*, r^*, b^*)$ representing the regression target.

The variable $c$ corresponds to the class label to which the object within the bounding box belongs, and $C$ signifies the total number of available classes.

#### 2) THE CLASSIFICATION BRANCH

As mentioned previously that the classification branch consists of two parts: one for predicting the class confidence and another for estimating the center-ness of the object, both utilizing the fused feature map $z$ as an input for the prediction network. To condense the output's dimensionality to 1D, we employ a linear convolutional layer with a $1 \times 1$ kernel. This is then followed by a sigmoid function to compute the predicted classification confidence.

The second part focuses on predicting the center-ness [33] of the object. Based on our observations, many low-quality predictions for bounding boxes stem from locations distant from the object's center. To address this problem, we introduce an auxiliary branch composed of a single-layer ($1 \times 1$ convolutional layer) This works concurrently with the classification branch (as depicted in Figure 1) and is specifically designed to estimate the center-ness of a given location. The center-ness value signifies the normalized distance between the location and the target object's center. For a given location with regression targets $t^* = (l^*, t^*, r^*, b^*)$, the center-ness target is defined as follows:

$$centerness^* = \sqrt{\frac{\min(l^*, r^*)}{\max(l^*, r^*)} x \frac{\min(t^*, b^*)}{\max(t^*, b^*)}} \tag{6}$$

The center-ness is a value confined within the range of 0 to 1. It is trained through binary cross-entropy loss, which is subsequently incorporated into the aggregate loss function, as shown in (7). The final score is computed by

multiplying the predicted center-ness with the corresponding predicted classification score. As a result, the center-ness can effectively assign reduced weights to bounding boxes that are situated further from an object's center. Consequently, these lower-quality bounding boxes are excluded during the final non-maximum suppression step, significantly bolstering the detection performance.

#### 3) LOSS FUNCTION

To optimize a training object, we determine the training loss function as follows:

$$L\left(\{p_{x,y}\},\{t_{x,y}\}\right)=\frac{1}{N_{pos}}\sum_{x,y}L_{cls}\left(p_{x,y},c^{*}_{x,y}\right)+\frac{\lambda}{N_{pos}}\sum_{x,y}1_{\left\{c^{*}_{x,y}>0\right\}}L_{reg}\left(t_{x,y},t^{*}_{x,y}\right) \tag{7}$$

The training loss function [33] consists of two components: the focal loss [32], denoted as $L_{cls}$, and the IoU loss [34] of the bounding box, represented by $L_{reg}$. The term $L_{cls}$ is defined based on the predicted classification score $p_{x,y}$ for class $c^{*}_{x,y}$ as determined in (8), while $L_{reg}$ is defined in (9) based on the regression target $t_{x,y}$ (predicted bounding box) and the ground-truth bounding box $t^{*}_{x,y}$. In the training loss function of (7), $N_{pos}$ indicates the number of positive samples. The balance weight for $L_{reg}$, denoted as $\lambda$, is assigned a value of 1. The sum is computed across all locations on the feature map $Z_i$. The indicator function $1_{\{c^{*}_{x,y}>0\}}$ is used, assigning a value of 1 to $c^{*}_{x,y}$ if the location $(x, y)$ is considered a positive sample, and 0 if it is regarded as a negative sample.

$$L_{cls}\left(p_{x,y},c^{*}_{x,y}\right)=-\propto\left(1-p_{x,y}\right)^{\gamma}\log\left(p_{x,y}\right) \tag{8}$$

The hyperparameters $\propto$ and $\gamma$ necessitate tuning based on our evaluation criteria. Typically, $\propto$ is set within the interval [0,1], while $\gamma$ lies within the interval [0,5]. In the context of this study, we have specifically designated $\propto = 0.25$ and $\gamma = 2$.

$$L_{reg}=-\sum_{i}ln\left(IoU\left(t,t^{*}\right)\right) \tag{9}$$

$$IoU=\frac{Intersection(t,t^{*})}{Union(t,t^{*})} \tag{10}$$

The IoU loss defined in (9) and (10) quantifies the discrepancy between the predicted bounding box $t*$ and the ground truth bounding box $t$. This offers an evaluation of the alignment accuracy between the predicted bounding box and the ground truth.

## IV. EXPERIMENTAL RESULTS

Our proposed method uses Python 3.6 and PyTorch 1.8.0. It achieves a speed of 25 frames per second (FPS). Our model is implemented on an Intel(R) Xeon(R) CPU E5-2698 v4 @ 2.20GHz with a single GPU and 32GB of memory, while other methods utilize 4 GPUs for their approaches [18], [14], [29].

### A. TRAINING DATASET

We employed the GOT-10k datasets [19] for our training, which involved training for 20 epochs using the stochastic gradient descent (SGD) optimizer with a batch size of 10. The learning rate was progressively increased from 0.01 to 0.08 during training. Our model was configured with a historical frame count ($T$) set to 3. For each frame within a training sample, we generated a 289 × 289 pixel square image patch, which served as the input for the model.

### B. COMPARISON

In our research, we explored three distinct approaches. Initially, we employed a baseline architecture as a starting point. In the second approach, we integrated a Temporal Convolutional Network (TCN) between the backbone network and the spatial-temporal memory network. Finally, in the third approach, we combined a TCN and an attention mechanism (TCN + Attention) and placed them between the backbone network and the spatial-temporal memory network. We trained our model using the entire training set from the GOT-10k dataset. Subsequently, we evaluated its performance on the GOT-10k testing set, and the results can be found in Table 1. In terms of the average overlap (AO) metric, it measures the average intersection over union (IoU) between the predicted bounding boxes and the ground truth across all frames in the sequence. Additionally, the success rate (SR) metric is assessed at thresholds of 0.50 and 0.75, measuring the percentage of frames where the predicted bounding box overlaps with the ground truth by a certain threshold.

**TABLE 1. Comparisons of three trackers on the GOT-10k dataset.**

| Trackers | AO | $SR_{0.50}$ | $SR_{0.75}$ | FPS |
|---|---|---|---|---|
| Baseline | 0.554 | 0.632 | 0.437 | 28 |
| TCN | 0.608 | 0.706 | 0.534 | 27 |
| TCN + Attention | 0.675 | 0.788 | 0.587 | 25 |

Based on the findings in Table 1, we can conclude that our proposed method, which integrates both the Temporal Convolutional Network and attention mechanism between the backbone network and the spatial-temporal memory network, outperformed the baseline and TCN methods in terms of tracker performance. However, this method had a lower frame per second (FPS) rate compared to the baseline and TCN trackers. To further evaluate our model, we tested it on a sequence video with a complex scene, and the results are illustrated in Figure 7.

In Figure 7(a), the bird is the target object in frame #86, and it is located at the center of the frame. The baseline method tracks the entire body of the bird, while the TCN and TCN+Attention methods closely track the object in proximity to the ground truth. In frame #130, when the birds move into a cloud, the TCN and TCN+Attention methods outperform the baseline method. In frame #191, when the objects emerge from the cloud, the TCN+Attention method

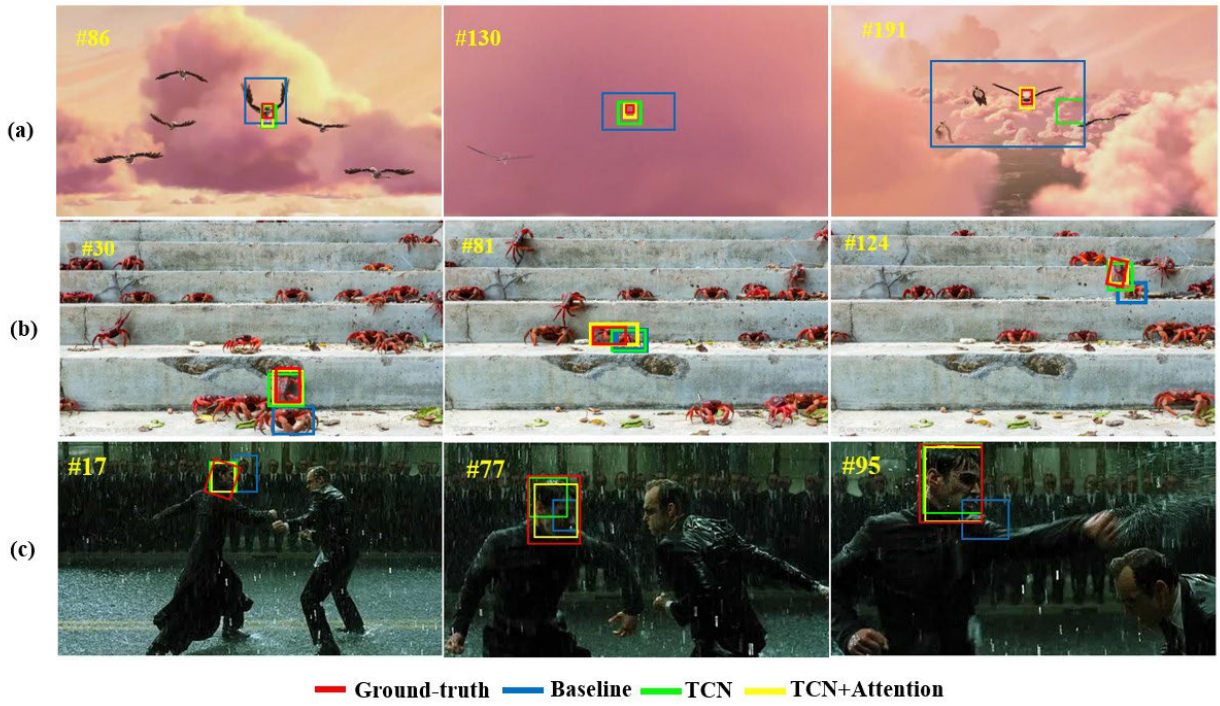


**FIGURE 7.** Visualized comparisons of our methods.

continues to track the target accurately, while the baseline and TCN methods deviate from the ground truth.

In Figure 7(b), in frame #30, the target crab traverses among other crabs, encountering an obstruction caused by a larger crab. Notably, the baseline method erroneously tracks the larger crab as the target, whereas the TCN and TCN+Attention methods maintain close and accurate tracking of the intended target.

In frame #81, a crab swiftly passes by the target crab, and only the TCN+Attention method successfully maintains tracking of the intended target. Subsequently, in frame #124, a crab resembling the target momentarily appears in close proximity to the target's location. Here, both the TCN and TCN+Attention methods consistently exhibit precise target tracking, while the baseline method exhibits deviations from the ground truth.

Moving on to Figure 7(c), where the challenge lies in tracking a face within a fast-paced battle scene characterized by a dark background and a group of individuals resembling the target. It is evident that the TCN+Attention method exhibits superior performance in closely tracking the target face, aligning well with the ground truth. The video available at https://drive.google.com/drive/folders/1ePyPF85nSgrDYkeoORzK6UG0oZMEwLDo?usp=sharing

Furthermore, we evaluated the performance of our method by comparing it with state-of-the-art approaches. The comparison results on the GOT-10k, OTB2015, UAV123, and VOT2018 datasets are shown in Table 2 to Table 5, respectively.

The GOT-10k dataset [19] is a large benchmark that comprises 10,000 videos, with 180 videos designated for the test set. This dataset is designed for generic object tracking, encompassing not only visual object tracking but also other related tasks such as visual object detection and semantic segmentation. To assess the performance of our approach, we conducted a comprehensive comparison with state-of-the-art methods, using the metrics of Average Overlap (AO) and Success Rate (SR) at IoU thresholds of 0.50 and 0.75. In Table 2, we present our tracker's performance, which achieved an AO score of 67.5%. It falls slightly below the RPformer and RANformer tracker. Notably, other methodologies were trained on larger and more diverse datasets. The datasets and their corresponding sizes, including LaSOT (227 GB), GOT-10k (66 GB), TrackingNet (2.1 TB), COCO (25 GB), ImageNet (150 GB), and Youtube-VOS (130 GB). Our approach distinguishes itself by being trained solely on a training set of the GOT-10k dataset. This result highlights the efficiency and effectiveness of our tracking model. For the graphical representation of the comparative performance across various tracking methods on the GOT-10k dataset, please refer to Figure 8.

The OTB2015 [28] is a popular visual tracking benchmark comprising 100 video sequences used for evaluating tracking method performance. In our experiment, we conducted a comparative analysis between our proposed approach and several state-of-the-art tracking methods. Table 3 presents comparisons of success (AUC) and precision scores on OTB2015. Notably, our method outperforms others with an

**TABLE 2.** The comparisons on GOT-10k dataset.

| Trackers | AO ↑ | $SR_{0.50}$ ↑ | $SR_{0.75}$ ↑ | Train dataset |
|---|---|---|---|---|
| SiamMASK [21] | 0.514 | 0.587 | 0.366 | COCO, ImageNet, YouTube-VOS |
| SiamDEPU [30] | 0.519 | 0.620 | 0.329 | ImageNet, GOT-10k |
| ATOM [22] | 0.556 | 0.634 | 0.402 | LaSOT, TrackingNet, COCO |
| SiamTPN [37] | 0.576 | 0.686 | 0.441 | COCO, LaSOT, GOT-10k, TrackingNet |
| DiMP-18 [23] | 0.579 | 0.672 | 0.446 | LaSOT, TrackingNet, GOT-10k, COCO |
| D3S [24] | 0.597 | 0.676 | 0.462 | Youtube-VOS |
| DASTSiam [37] | 0.601 | 0.698 | 0.493 | LaSOT, ImageNet |
| TRAST [18] | 0.604 | 0.708 | 0.469 | GOT-10k, ImageNet |
| ASCF[49] | 0.614 | 0.696 | 0.526 | GOT-10k |
| SiamLA[47] | 0.619 | 0.724 | 0.510 | Youtube-BB, ImageNet, GOT-10k, COCO |
| ALT [36] | 0.622 | 0.732 | 0.501 | COCO, GOT-10k, LaSOT |
| Ours | 0.675 | 0.788 | 0.587 | GOT-10k |
| RPformer [46] | 0.694 | 0.795 | 0.642 | COCO, GOT-10k, LaSOT |
| RANformer [26] | **0.706** | **0.813** | **0.664** | COCO, GOT-10k, LaSOT |

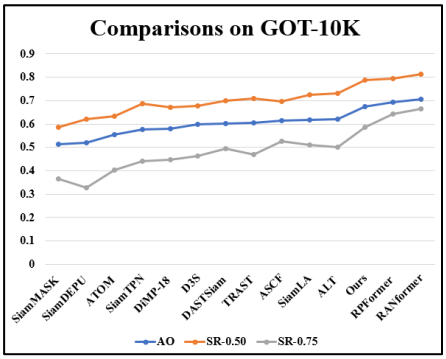


**FIGURE 8.** The comparison graph on GOT-10k.

**TABLE 3.** The comparisons on OTB2015 dataset.

| Trackers | Success | Precision |
|---|---|---|
| ULAST [41] | 0.610 | 0.811 |
| Spiking SiamFC++ [40] | 0.644 | 0.854 |
| DaHCF [42] | 0.670 | 0.911 |
| MEGTCF [44] | 0.678 | 0.914 |
| AFCSCF [43] | 0.681 | 0.915 |
| RPformer [46] | 0.682 | 0.891 |
| DHPR [45] | 0.691 | 0.916 |
| ALT [36] | 0.692 | 0.908 |
| RANformer [26] | 0.692 | - |
| SiamTPN [37] | 0.702 | 0.902 |
| SiamLA [47] | 0.709 | 0.929 |
| Ours | **0.721** | **0.935** |

**TABLE 4.** The comparisons on UAV123 dataset.

| Trackers | Success | Precision |
|---|---|---|
| ASTCA [38] | 0.481 | 0.687 |
| MRCF [20] | 0.485 | 0.666 |
| MEGTCF [44] | 0.502 | 0.721 |
| DHPR [45] | 0.514 | 0.741 |
| Spiking SiamFC++ [40] | 0.578 | 0.744 |
| SiamCAR [15] | 0.614 | 0.760 |
| SiamBAN [6] | 0.631 | 0.833 |
| SiamTPN [37] | 0.636 | 0.823 |
| UAST [50] | 0.645 | 0.860 |
| SiamGAT [39] | 0.646 | 0.843 |
| ALT [36] | 0.652 | 0.871 |
| Ours | 0.658 | **0.874** |
| RPformer [46] | 0.669 | - |
| RANformer [26] | **0.686** | - |

**TABLE 5.** Comparisons on VOT2018 dataset.

| Trackers | A ↑ | R ↓ | EAO ↑ |
|---|---|---|---|
| SiamSNN [25] | 0.460 | 0.860 | 0.176 |
| Siamese-RPN [10] | 0.490 | 0.460 | 0.244 |
| DHPR [45] | 0.495 | 0.304 | 0.274 |
| SiamFC [8] | 0.503 | 0.585 | 0.188 |
| MEGTCF [44] | 0.505 | 0.314 | 0.278 |
| UpdateNet [27] | 0.518 | 0.454 | 0.244 |
| DaSiamRPN+Att [28] | 0.536 | **0.144** | 0.097 |
| C-RPN [29] | 0.550 | 0.320 | 0.273 |
| BCS [31] | 0.556 | 0.318 | 0.304 |
| Spiking SiamFC++ [40] | 0.556 | 0.445 | 0.255 |
| Ours | 0.590 | 0.281 | 0.380 |
| RPformer [46] | 0.648 | 0.158 | **0.491** |
| RANformer [26] | **0.709** | 0.156 | 0.481 |

impressive AUC score of 72.1%, the highest among the tracking methods compared.

The UAV123 [51] dataset comprises 123 video sequences captured by a low-altitude unmanned aerial vehicle (UAV). In contrast to other benchmark datasets, this dataset contains many small objects, along with tracking sequences that have several distractor objects and prolonged occlusions. Table 4 displays a success (AUC) score of 65.8% achieved by our proposed tracker. It falls slightly below the RPformer and RANformer tracker.

The VOT2018 dataset [21] consists of 60 videos designed for visual object tracking, where the objective is to track an object in a video sequence based on its initial location. We employ standard metrics to evaluate the methods, specifically in terms of the accuracy (A) metric, which measures the percentage of frames where the predicted bounding box overlaps with the ground truth by a certain threshold. The robustness (R) metric assesses the percentage of videos in which the tracker successfully tracks the object until the end of the sequence. The expected average overlap (EAO) metric, a combination of A and R, provides an overall measure of the tracker's performance in terms of accuracy and robustness.

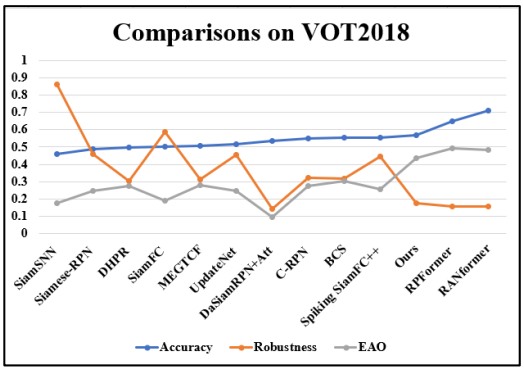


**FIGURE 9.** The comparison graph on VOT2018.

The results presented in Table 5 indicate that our method achieved an accuracy score of 59.0%, which is higher than the most tracking methods compared, except for the RPformer and RANformer tracker. The graphical representation of this comparison on VOT2018 is shown in Figure 9.

### C. ABLATION EXPERIMENT

We conducted an ablation study in which we evaluated our model's performance under two different training scenarios: one using 20% of the GOT-10k and COCO datasets and the other using only 40% of the GOT-10k dataset's training set. Subsequently, we tested the model on the GOT-10k dataset, and the results are presented in Table 6. Our findings clearly demonstrate that both the TCN and the attention mechanism play pivotal roles in enhancing overall accuracy.

**TABLE 6.** Comparing the performance between training with 20% of the GOT-10k and COCO datasets and training with 40% of the GOT-10k dataset.

| Trackers | Training 20% of the GOT-10k and COCO | | | Training 40% of the GOT-10k | | |
|---|---|---|---|---|---|---|
| | AO | $SR_{0.50}$ | $SR_{0.75}$ | AO | $SR_{0.50}$ | $SR_{0.75}$ |
| Base model | 0.529 | 0.615 | 0.416 | 0.532 | 0.626 | 0.402 |
| TCN | 0.580 | 0.688 | 0.471 | 0.592 | 0.699 | 0.497 |
| TCN+Attention | 0.611 | 0.712 | 0.535 | 0.630 | 0.737 | 0.529 |

## V. CONCLUSION

We presented an architecture for enhancing single object tracking that combines Temporal Convolutional Network (TCN) and attention mechanism with spatial-temporal memory. This method is designed to capture complex temporal patterns with long-range dependencies. By utilizing spatial-temporal memory based on historical data, this approach eliminates the need for direct template updating and conventional template matching methods. Moreover, to ensure that the model accurately captures the target's true characteristics, we incorporated a background label during feature extraction. Our method focuses on streamlining the training process when working with restricted datasets due to the constraints imposed by computational resources. We conducted experiments across various benchmark datasets, including GOT-10k, OTB2015, UAV123, and VOT2018.

Despite its capabilities, our method faces challenges when confronted with some intricate real-world situations. These situations include rapid motion sequences, closely resembling objects, and instances of occlusion, which can adversely affect the efficacy of our tracking approach. In future research endeavors to enhance tracking performance, we may consider incorporating versatile and diverse template features. Moreover, given unrestricted hardware resources, evaluating our tracker on expansive datasets, like LaSOT and TrackingNet, will offer a comprehensive perspective on its resilience and flexibility across diverse tracking situations.

## ACKNOWLEDGMENT

This research was supported by the National Science and Technology Council (NSTC), Taiwan, under Grants NSTC 111-2221-E-155-039-MY3 and NSTC 112-2918-I-008-008.

## REFERENCES

[1] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6667–6676.

[2] Q. Xie, K. Liu, A. Zhiyong, L. Wang, Y. Li, and Z. Xiang, "A novel incremental multi-template update strategy for robust object tracking," *IEEE Access*, vol. 8, pp. 162668–162682, 2020, doi: 10.1109/ACCESS.2020.3021786.

[3] J. Fan, K. Zhang, Y. Huang, Y. Zhu, and B. Chen, "Parallel spatio-temporal attention-based TCN for multivariate time series prediction," *Neural Comput. Appl.*, vol. 35, no. 18, pp. 13109–13118, May 2021, doi: 10.1007/s00521-021-05958-z.

[4] Y. He and J. Zhao, "Temporal convolutional networks for anomaly detection in time series," *J. Phys., Conf. Ser.*, vol. 1213, no. 4, pp. 1–6, Jun. 2019, doi: 10.1088/1742-6596/1213/4/042050.

[5] Z. Lai, E. Lu, and W. Xie, "MAST: A memory-augmented self-supervised tracker," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6478–6487.

[6] H. Gao and C. Hu, "A new approach of template matching and localization based on the guidance of feature points," in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Aug. 2018, pp. 548–553.

[7] T. Shi, D. Wang, and H. Ren, "Triplet network template for Siamese trackers," *IEEE Access*, vol. 9, pp. 44426–44435, 2021, doi: 10.1109/ACCESS.2021.3066294.

[8] H. Lu, X. Ren, and M. Tong, "Object tracking algorithm of fully-convolutional Siamese networks using the templates with suppressed background information," in *Proc. 26th IEEE Int. Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2021, pp. 1–6.

[9] W. R. Tan and S.-H. Lai, "I-Siam: Improving Siamese tracker with distractors suppression and long-term strategies," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 55–63.

[10] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with Siamese region proposal network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 8971–8980.

[11] Z. Zhou, X. Li, T. Zhang, H. Wang, and Z. He, "Object tracking via spatial–temporal memory network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 5, pp. 2976–2989, May 2022, doi: 10.1109/TCSVT.2021.3094645.

[12] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.

[13] R. Zhu, W. Liao, and Y. Wang, "Short-term prediction for wind power based on temporal convolutional network," *Energy Rep.*, vol. 6, pp. 424–429, Dec. 2020, doi: 10.1016/j.egyr.2020.11.219.

[14] P. Lara-Benítez, M. Carranza-García, J. M. Luna-Romera, and J. C. Riquelme, "Temporal convolutional networks applied to energy-related time series forecasting," *Appl. Sci.*, vol. 10, no. 7, p. 2322, Mar. 2020, doi: 10.3390/app10072322.

[15] A. Vaswani, "Attention is all you need," in *Proc. Int. Conf. Neur. Inf. Proc. Syst. (NIPS)*, Jun. 2017, pp. 6000–6010.
[16] Z. Fu, Q. Liu, Z. Fu, and Y. Wang, "STMTrack: Template-free visual tracking with space-time memory networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13769–13778.
[17] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7794–7803.
[18] M. Dunnhofer, N. Martinel, and C. Micheloni, "Tracking-by-trackers with a distilled and reinforced model," in *Proc. Asian Conf. Comput. Vis. (ACCV)*, 2020, pp. 631–650.
[19] L. Huang, X. Zhao, and K. Huang, "GOT-10k: A large high-diversity benchmark for generic object tracking in the wild," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1562–1577, May 2021, doi: 10.1109/TPAMI.2019.2957464.
[20] J. Ye, C. Fu, F. Lin, F. Ding, S. An, and G. Lu, "Multi-regularized correlation filter for UAV tracking and self-localization," *IEEE Trans. Ind. Electron.*, vol. 69, no. 6, pp. 6004–6014, Jun. 2022.
[21] F. Chen, F. Zhang, and X. Wang, "Two stages for visual object tracking," in *Proc. Int. Conf. Intell. Comput., Autom. Appl. (ICAA)*, Jun. 2021, pp. 165–170.
[22] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ATOM: Accurate tracking by overlap maximization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4655–4664.
[23] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, "Learning discriminative model prediction for tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6181–6190.
[24] A. Lukezic, J. Matas, and M. Kristan, "D3S—A discriminative single shot segmentation tracker," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7131–7140.
[25] Y. Luo, M. Xu, C. Yuan, X. Cao, L. Zhang, Y. Xu, T. Wang, and Q. Feng, "SiamSNN: Siamese spiking neural networks for energy-efficient object tracking," in *Proc. Int. Conf. Neural Netw.*, 2021, pp. 182–194.
[26] L. Zhang, A. Gonzalez-Garcia, J. Van De Weijer, M. Danelljan, and F. S. Khan, "Learning the model update for Siamese trackers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4009–4018.
[27] S. Jia, C. Ma, Y. Song, and X. Yang, "Robust tracking against adversarial attacks," in *Proc. Comput. Vis. (ECCV)*, 2020, pp. 69–84.
[28] H. Fan and H. Ling, "Siamese cascaded region proposal networks for real-time visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7944–7953.
[29] M. Zhang, K. Van Beeck, and T. Goedemé, "Object tracking with multiple dynamic templates updating," in *Proc. Int. Conf. Image Vis. Comput. (IVCNZ)*, 2022, pp. 144–158.
[30] H. Dong, J. Jiao, and Y. Bai, "Bounding-box centralization for improving SiamFC++," in *Proc. Asian Conf. Artif. Intell. Technol. (ACAIT)*, Oct. 2021, pp. 196–203.
[31] Z. Zhang, H. Peng, J. Fu, B. Li, and W. Hu, "Ocean: Object-aware anchor-free tracking," in *Proc. Comput. Vis. (ECCV)*, 2020, pp. 771–787.
[32] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2999–3007.
[33] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9626–9635.
[34] J. Yu, Y. Jiang, Z. Wang, Z. Cao, and T. Huang, "Unitbox: An advanced object detection network," in *Proc. 24th ACM Int. Conf. Multimedia*, Oct. 2016, pp. 516–520.
[35] D. Xing, N. Evangeliou, A. Tsoukalas, and A. Tzes, "Siamese transformer pyramid networks for real-time UAV tracking," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2022, pp. 2139–2148.
[36] D. Yuan, X. Chang, Z. Li, and Z. He, "Learning adaptive spatial–temporal context-aware correlation filters for UAV tracking," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 18, no. 3, pp. 1–18, 2022, doi: 10.1145/3486678.
[37] D. Guo, Y. Shao, Y. Cui, Z. Wang, L. Zhang, and C. Shen, "Graph attention tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 9543–9552.
[38] S. Xiang, T. Zhang, S. Jiang, Y. Han, Y. Zhang, C. Du, X. Guo, L. Yu, Y. Shi, and Y. Hao, "Spiking SiamFC++: Deep spiking neural network for object tracking," 2022, *arXiv:2209.12010*.
[39] Q. Shen, L. Qiao, J. Guo, P. Li, X. Li, B. Li, W. Feng, W. Gan, W. Wu, and W. Ouyang, "Unsupervised learning of accurate Siamese tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2022, pp. 10–8101.
[40] J. Zhang, Y. Liu, H. Liu, J. Wang, and Y. Zhang, "Distractor-aware visual tracking using hierarchical correlation filters adaptive selection," *Appl. Intell.*, vol. 52, no. 6, pp. 6129–6147, 2022.
[41] S. Ma, L. Zhang, Z. Hou, X. Yang, L. Pu, and X. Zhao, "Robust visual tracking via adaptive feature channel selection," *Int. J. Intell. Syst.*, vol. 37, no. 10, pp. 6951–6977, Oct. 2022.
[42] S. Ma, Z. Zhao, Z. Hou, L. Zhang, X. Yang, and L. Pu, "Correlation filters based on multi-expert and game theory for visual object tracking," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–14, 2022.
[43] S. Ma, B. Zhao, Z. Hou, W. Yu, L. Pu, and L. Zhang, "Robust visual object tracking based on feature channel weighting and game theory," *Int. J. Intell. Syst.*, vol. 2023, pp. 1–19, Jul. 2023, doi: 10.1155/2023/6731717.
[44] F. Gu, J. Lu, and C. Cai, "RPformer: A robust parallel transformer for visual tracking in complex scenes," *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–14, 2022, doi: 10.1109/TIM.2022.3170972.
[45] D. Yuan, X. Chang, Q. Liu, Y. Yang, D. Wang, M. Shu, Z. He, and G. Shi, "Active learning for deep visual tracking," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, May 10, 2023, doi: 10.1109/TNNLS.2023.3266837.
[46] F. Gu, J. Lu, and C. Cai, "A robust attention-enhanced network with transformer for visual tracking," *Multimedia Tools Appl.*, vol. 82, no. 26, pp. 40761–40782, Nov. 2023, doi: 10.1007/s11042-023-15168-5.

**PIMPA CHEEWAPRAKOBKIT** received the B.Sc. degree in computer science from Burapha University, Thailand, and the M.S. degree in computer science and information systems from the National Institute of Development Administration, Thailand. She is currently pursuing the Ph.D. degree in computer science and information engineering from National Central University, Taiwan. She is an Assistant Professor with the Information Technology Department, Asia–Pacific International University, Thailand. Her current research interests include computer vision, object tracking, big data analysis, deep learning, object detection, and recognition.

**CHIH-YANG LIN** (Senior Member, IEEE) is currently with the Department of Mechanical Engineering, National Central University, Taoyuan, Taiwan. Previously, he was the Dean of the International Academy, the Chief of the Global Affairs Office, and a member of the Department of Electrical Engineering, Yuan-Ze University, Taoyuan. His current research interests include computer vision, machine learning, deep learning, image processing, big data analysis, and the design of surveillance systems. He has been recognized as an IET Fellow and has contributed to over 200 papers that have been featured in a wide range of international conferences and journals. Throughout his academic career, he has received multiple accolades, including the Best Paper Awards from the Pacific-Rim Conference on Multimedia (PCM), in 2008; the Best Paper Awards and an Excellent Paper Award from the IPPR Conference on Computer Vision, Graphics, and Image Processing, in 2009, 2013, and 2019; the Best Paper Award from the Sixth International Visual Informatics Conference, in 2019 (IVIC'19), and the Best Paper Award from the Second International Conference on Broadband Communications, Wireless Sensors, and Powering, in 2020. He also has served in several leadership positions for various international conferences, taking on responsibilities, such as the Program Chair, the Session Chair, the Publication Chair, the Publicity Chair, and the Workshop Organizer for events, such as AHFE, ICCE, ACCV, IEEE Multimedia Big Data, ACM IH&MMSec, APSIPA, and CVGIP. Additionally, he serves as a Regular Reviewer for esteemed journals, including IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON MULTIMEDIA, IEEE ACCESS, and several other distinguished Elsevier publications.

**TIMOTHY K. SHIH** (Senior Member, IEEE) is currently a Distinguished Professor with National Central University (NCU), Taiwan. In NCU, he was the Vice Dean of the College of EECS and the Founding Director of the Innovative AI Research Center. He was the Dean of the College of Computer Science, Asia University, Taiwan, and the Chairperson of the CSIE Department, Tamkang University, Taiwan. He is a fellow of the Institution of Engineering and Technology (IET). He was the Founding Chairman Emeritus of the IET Taipei Local Network. In addition, he is a Senior Member of ACM. He was the Founder and the Co-Editor-in-Chief of *International Journal of Distance Education Technologies*, USA. He was an Associate Editor of IEEE COMPUTING. He was an Associate Editor of IEEE TRANSACTIONS ON LEARNING TECHNOLOGIES, *ACM Transactions on Internet Technology*, and IEEE TRANSACTIONS ON MULTIMEDIA. He was the Conference Co-Chair of the 2004 IEEE International Conference on Multimedia and Expo (ICME'2004). He has received many research awards, including the Research Award from the National Science Council of Taiwan, the IIAS Research Award from Germany, the HSSS Award from Greece, the Brandon Hall Award from USA, the 2015 Google MOOC Focused Research Award, and several best paper awards from international conferences. He was named the 2014 Outstanding Alumnus by Santa Clara University. For more information: (http://tshih.minelab.tw).

**AVIRMED ENKHBAT** received the B.S. degree in computer science and a M.S. degree in applied sciences and engineering from the National University of Mongolia, Mongolia, in 2011 and 2016, respectively. He is currently pursuing the Ph.D. degree in computer science and information engineering with National Central University (NCU), Taoyuan, Taiwan. His current research interests include computer vision, human–computer interaction, and gesture recognition.