DEVELOPMENT OF ADAPTIVE TRACKING METHODS WITH ENHANCED PERFORMANCE BASED ON DEEP LEARNING

SHUO ZHANG

A DISSERTATION SUBMITTED TO

THE FACULTY OF GRADUATE STUDIES

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN

MECHANICAL ENGINEERING

YORK UNIVERSITY

TORONTO, ONTARIO

DECEMBER 2024

Abstract

Adaptive object tracking aspires to locate the target incessantly in each frame with designated initial target location, which is an imperative yet demanding task in computer vision. Recent adaptive approaches strive to fuse global information of template and search region for achieving promising tracking performance. However, fusion of global information devastates some local details. Local information is essential for distinguishing the target from background regions. To address this problem, we present a novel tracker TGLC integrating a channel-aware convolution block and Transformer attention for global and local representation aggregation, and for channel information modeling. Experimental results demonstrate the superior tracking performance of TGLC. Ablation experiments further verify the effectiveness of multiple information aggregation for improving tracking performance.

Long-term tracking is a vital component in real-world tracking scenarios. Recently, one-stage long-term trackers achieve state-of-the-art tracking results due to more sufficient integration of search and template representations. These methods usually adopt an encoder for synchronous feature generation and interaction. Despite their high performance, the approaches tend to feed the encoder full input representations that are highly redundant during training. A novel algorithm MIMTracking is developed for tackling this problem. MIMTracking exploits an encoder and a decoder for masked image modeling during training. This design alleviates input redundancy and reduces the computational cost of the training process. The proposed MIMTracking achieves state-of-the-art tracking results on numerous datasets.

Addressing tracking challenges is an essential topic in real-world applications. Constantly varying appearance of targets brings tremendous challenges for object tracking, especially in background clutter scenarios. Current leading trackers attempt to introduce dynamic templates to encode changing target information. However, dynamic templates are obtained from intermediate frames that are not manually annotated. Therefore, these templates may contain a large amount of uninformative and irrelevant background noise due to imprecise tracking. To tackle the problem, a novel tracker ATPTrack is proposed for tracking. Particularly, ATPTrack develops an alternating token trimming method that prunes dynamic templates and search region progressively. Compared to merely trimming the search region, ATPTrack further reduces MACs by 11.5% with negligible performance drop of 0.3% by alternately pruning dynamic templates and search region.

Acknowledgements

I owe a special debt of gratitude to my supervisor, Prof. Dan Zhang. His encouragement, support and guidance have been the most critical component during my PhD journey in York University. His profound knowledge and critical insights have made substantial contribution to the quality and the depth of this research. His insightful feedback and encouragement have not only been invaluable to this dissertation but have also been crucial in shaping my character. His mentorship inspired me to strive for excellence in all my endeavors. I really want to thank him for being an exceptional supervisor and for making this achievement possible.

I would like to express my gratitude to my supervisory committee members, Prof. Aleksander Czekanski and Prof. George Z.H. Zhu, for their unwavering support and guidance throughout my PhD journey. They provided me with constructive feedback on my research topics, objectives, and methodologies, which helped me quickly grasp the essence of the research.

I am grateful to all my lab members, Qi Zou, Xueling Luo, and Behrad Jabarnejad for their invaluable assistance in discussing and troubleshooting problems for my research. I hope our collaboration will continue in the future.

I would like to thank my family members for their encouragement, understanding, and belief in me during this challenging phase of my life. Especially, I would like to extend my deepest gratitude to my husband, Qi Zou, for his love, patience, encouragement, and understanding for me. Thank you for believing in me and for your sacrifices to make sure that I had enough time to focus on my research. This dissertation could not have been accomplished without your support. Thank you for everything.

Finally, I would like to thank all my friends for their unconditional support and trust and all the people who helped me during this time. Thank you all. I wish you all a bright and prosperous future.

Contents

List of Tables

List of Figures

List of Nomenclatures

AO	Average overlap
AUC	Area under curve
BEiT	Bidirectional encoder representation from image transformers
CCB	Channel-aware convolution block
CF	Correlation filter
CN	Color names
ConvNets	Convolutional neural networks
Dec_i	The i-th layer of decoder
DL	Deep learning
DTP	Dynamic template pruning
Enc_i	The i-th layer of encoder
FLOPs	Floating-point operations per second
FPS	Frames per second
GIoU	Generalized intersection over union
GLCA	Global-local cross-attention
GLSA	Global-local self-attention
HOG	Histogram of oriented gradients
LCA	Long-term context attention
LN	Layer normalization
LSTM	Long short-term memory
MACs	Multiply-accumulate operations
MAE	Masked autoencoders

MHCA	Multi-head cross-attention
MHSA	Multi-head self-attention
MIM	Masked image modeling
MIMDET	Masked image modeling for detection
MLP	Multilayer perceptron
MSA	Multi-head self-attention
P	Precision
P_{norm}	Normalized precision
Params	Model parameters
RCNN	Region-based convolutional neural networks
RPN	Region proposal network
SE	Squeeze-and-excitation
SOTA	State-of-the-art
$SR_{0.5}$	Success rate under threshold 0.5
$SR_{0.75}$	Success rate under threshold 0.75
SRP	Search region pruning
SVM	Support vector machine
TGLC	Tracking with global local and channel information
UAV	Unmanned aerial vehicle
ViT	Vision Transformer
ViT-B	Vision Transformer base
ViT-L	Vision Transformer large

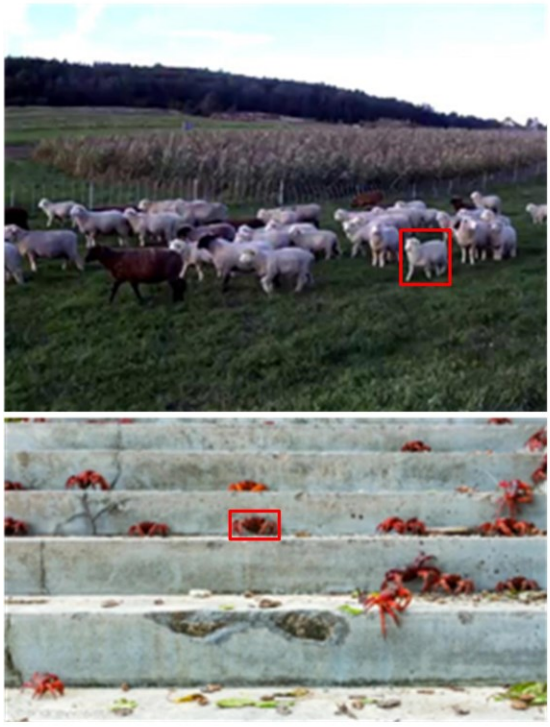
Chapter 1 Introduction

Visual object tracking is one of the most cutting-edge topics in robot vision and computer vision, which aims at locating the target continuously with given object position in the initial frame. Visual object tracking has been widely utilized in many application scenarios such as autonomous vehicles [1], human machine interaction [2], video surveillance [3] and robot perception [4], etc. As demonstrated in Figure 1-1, there are numerous tracking challenges, e.g., illumination variation, appearance variation, fast motion, motion blur, out-of-plane rotation, scale variation and occlusion, etc. [5]. These challenges render object tracking extremely demanding. Object tracking is divided into different categories according to different tracking requirements, such as short-term tracking, long-term tracking, real-time tracking and non-real-time tracking, etc. Multiple tracking algorithms are developed for specific types of targets, i.e., vehicle tracking, path tracking. These types of tracking methods are termed as specific target-oriented tracking approaches or application scenario-oriented tracking approaches.

Human tracking is one of the important applications for object tracking. Dadi et al. [6] propose a novel approach for face recognition and tracking by utilizing support vector machine and Gaussian mixture model, which can be used in recognizing and tracking human face for surveillance videos. Laaroussi et al. [7] develop a mean shift-based tracking method, which adopts a combination of foreground-weighted histogram and color features as the feature representation for the mean shift tracking model. Kumar et al. [8] present a human tracking and re-localization method based on convolutional neural networks, which achieves promising tracking performance.

(a) Motion blur

(b) Full occlusion



(c) Background clutter

(d) Out-of-view

Figure 1-1 Tracking challenge examples

Some tracking methods are designed specifically for tracking vehicles. In [9], the authors propose a new tracking method based on optimal unbiased finite memory filter for accurate tracking of the car ahead. This algorithm achieves better tracking performance than Kalman filter-based tracking approach. Dong et al. [10] present an accurate TLD (tracking learning detection) based tracking algorithm, which deploys fast retina key point feature to enhance the ability of the tracker for dealing with challenges. Qiu et al. [11] design a deep learning-based tracking method for mobile vehicles.

Object tracking can also be utilized in medical field such as surgical instrument tracking. Authors of [12] present a real-time tracking method by using spatiotemporal context information for medical instrument detection and tracking. The tracker is built by convolutional neural networks (ConvNets). Nwoye et al. [13] propose a novel surgical instrument tracking approach based on ConvNets and long short-term memory networks (LSTM), which utilizes exclusively weakly supervised tool presence label and achieves enhanced performance.

There are some drawbacks for the aforementioned tracking methods. These methods are developed only for one type of objects, so they have poor adaptability to other application scenarios. These approaches possess poor performance as well since they can only produce and exploit low-discriminative target features. These features are more likely to cause tracking failures. Furthermore, task-specific hyperparameters and other design parameters are not adaptive, which are required to be modified in different scenarios. Therefore, it is required to develop adaptive tracking algorithms for complicated tracking scenarios.

Adaptive tracking is presented to track objects of arbitrary classes, which possesses much stronger adaptability than specific target-oriented tracking. There are more difficulties in

designing adaptive tracking algorithms since these methods are required to generate and process more complicated features to be suitable for different types of objects and a variety of tracking scenarios. However, it is extremely meaningful and essential to develop adaptive approaches. The advantages are shown below:

(i) Once adaptive methods are developed, they can be employed to track arbitrary targets of any tracking scenarios without considering the category and appearance of the target. The only initialization is to determine the target region in the initial frame. After that, the target can be tracked automatically in the following frames. However, non-adaptive methods can only track a single type of targets in designated scenarios. They are not as practical as adaptive tracking approaches.

(ii) Some tracking scenarios have multiple target classes. For example, path tracking, vehicle tracking, and pedestrian tracking are three vital tasks in autonomous vehicles. To achieve this, three non-adaptive tracking approaches are required to be designed for three types of objects, respectively. However, merely one adaptive tracker is sufficient for handling three types of objects without modifying the hyper-parameters. This is practical and convenient for most application scenarios.

Adaptive tracking is the focus of this research. The performance of adaptive tracking methods is limited by many factors, which are demonstrated below and will be explored in this research:

(i) The existence of a series of tracking challenges makes it difficult for tracking methods to successfully track the target from the beginning to the end without any tracking failures. The ability to overcome as many challenges as possible is required in this research.

(ii) The complexity of tracking models is restricted by the tracking speed. This makes it difficult to further improve the tracking performance. The trade-off between tracking speed and performance is critical. It requires tracking methods to optimize the performance with minimal loss of speed.

As described in Figure 1-2, adaptive short-term tracking and adaptive tracking in real-world tracking scenarios are two vital problems of adaptive tracking. Furthermore, two key issues should be addressed for the application of adaptive tracking in real-world scenarios, especially in the field of robot vision. One is to enhance the long-term tracking performance of tracking methods. The other is to optimize the capability of tracking methods against complicated tracking challenges. Therefore, research on state-of-the-art adaptive tracking methods and respective study on the two significant issues are three focuses of this research. The three topics will be introduced in chapters 1.1, 1.2 and 1.3, respectively.

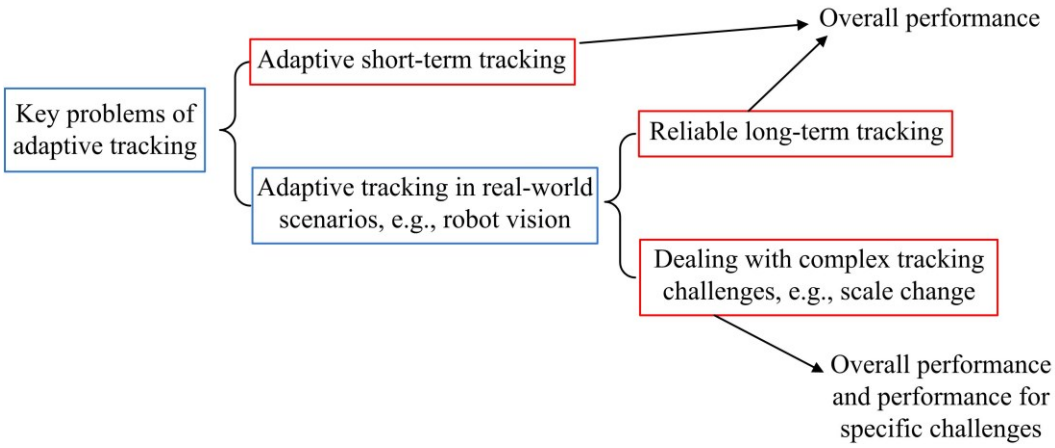


Figure 1-2 Key problems of adaptive tracking

1.1 Adaptive visual tracking

Adaptive object tracking methods are generally categorized into generative methods and discriminative methods [14]. Generative methods [15], [16], [17], [18] mainly analyze and model the object area in the current frame and find the area (the next object location) that

best resembles the model in the next frame. Discriminative methods [19], [20] are increasingly popular, which usually train classifiers to distinguish the object location from background regions. Discriminative methods achieve more superior tracking results. Their development experiences the transition from non-deep learning to deep learning. Correlation filter-based trackers [21], [22] are the most representative non-deep learning methods. These methods can perform real-time tracking even on CPU. However, these methods exhibit poor performance since they utilize hand-crafted features that are much less powerful than deep features generated by deep learning-based methods.

Siamese-based trackers gradually become the dominant deep learning approaches. Representative Siamese-based trackers usually consist of a shared backbone for feature extraction, a feature fusion module and one or more prediction head(s). Numerous feature fusion methods are employed for Siamese-based trackers, which are divided into two main categories, cross correlation-based methods and attention-based fusion approaches. Naive correlation is a simple method for correlation calculation, which is commonly used in previous Siamese trackers such as [23], [24], [25]. Depth-wise correlation calculates correlation of template and search image in a sliding window at each depth level [26]. Pixel-wise correlation [27], [28] computes feature connection in smaller areas than naive or depth-wise correlation, which better maintains the spatial features of the search region. Correlation-based methods solely establish locally linear similarity relations between search region and template, which are less discriminative and less informative. Therefore, trackers of this type are greatly limited in performance.

Recently, some researchers [29], [30] start to exploit attention-based networks to fuse features. These approaches focus more on specific features, achieving better tracking

results than correlation-based fusion methods. However, attention-based feature fusion methods [31] strive to build global associations of input feature maps without modeling connections between local representations. Locally detailed features are especially important when tracking challenges occur. For example, visible partial information can be utilized to locate the target when the target is partially occluded. Therefore, it is quite essential to simultaneously model the global and local representations for more accurate feature interaction. In addition, most existing feature fusion methods only exploit spatial information of feature maps, which fail to exploit channel representations. Lack of channel information results in low object localization ability.

Table 1-1 summarizes the representative adaptive trackers and their key attributes. DL and AUC denote separately deep learning-based and the area under curve metric on OTB100 dataset [32]. It can be seen from the table that deep-learning-based trackers outperform non-deep learning methods in terms of the AUC score. The reason is that deep learning trackers exploit deep semantic representations that are much stronger and more discriminative than hand-crafted features.

It also shows that almost all deep trackers only model local or global information in the feature fusion module without combining them. Channel information is not employed either in these methods. Thus, a novel feature fusion module incorporating multiple information should be developed for adaptive short-term tracking approaches. Fusion of extensive spatial and channel representations contributes to the improvement of the adaptability and the tracking results for adaptive short-term trackers.

Table 1-1 Representative adaptive tracking methods

Methods	DL	Year	AUC	Correlation or attention	Global or local	Channel	Real-time
KCF [21]	No	2015	0.477	Correlation filter	Hand-crafted features	No	Yes
Staple [22]	No	2016	0.579	Correlation filter	Hand-crafted features	No	Yes
SiamFC [23]	Yes	2016	0.582	Naive correlation	Local	No	Yes
SiamRPN [24]	Yes	2018	0.637	Naive correlation	Local	No	Yes
DaSiamRPN [25]	Yes	2018	-	Naive correlation	Local	No	Yes
SiamMask [26]	Yes	2019	-	Depth-wise correlation	Local	No	Yes
Alpha-Refine [28]	Yes	2021	-	Pixel-wise correlation	Local	No	Yes
TransT [29]	Yes	2021	0.694	Attention feature fusion	Global	No	Yes
Stark [30]	Yes	2021	0.681	Attention feature fusion	Global	No	Yes

1.2 Long-term object tracking

Adaptive long-term tracking is much more demanding than short-term tracking since long-term tracking encounters more tracking challenges. Therefore, long-term tracking is more prone to tracking failures. Long-term tracking approaches are also divided into non-deep learning methods and deep learning methods. TLD [20] is a classic long-term tracking algorithm, which integrates tracking, detection and learning processes. The tracking module is based on Median-Flow algorithm [33], which predicts the motion trajectories of multiple points inside of the bounding box. Then it utilizes the median of the most reliable

half of trajectories as the final prediction. Once the tracking module fails, the detector starts scanning the entire image and re-determines the position of the target. The learning block is employed to assess the tracking and detection results.

LT-FLO [34] employs boundary-based features to reduce the demand for image textures, which also improves its ability to adapt to illumination variation. Similar to TLD, the tracking failure detection and target redetection strategies are utilized to recover from tracking failures. Ma et al. [35] leverage correlation filters to estimate the location and scale of the object during long-term tracking. Random ferns are exploited to train a detector for relocating the object. It achieves real-time tracking. However, it is not capable of maintaining stable long-term tracking due to the limitations of correlation filters and the detection module. Liang et al. [36] propose a real-time long-term tracking method. This method builds a super pixel optical flow predictor for translation and scale estimation, an adaptive detector based on the kernelized correlation for more sophisticated prediction, and a corrector utilizing dual online SVMs for refinement of the tracking results. A re-detection module based on the particle filter [37] is presented to be integrated with a discriminative correlation filter for tracking result refinement in case of tracking failures. The reliability of the results is validated in the long-term tracking method [38].

Deep features possess stronger modeling ability for unknown categories of targets, which are suitable for adaptive tracking. A number of deep learning-based approaches are presented for long-term tracking. GlobalTrack [39] searches the target within the entire image. It utilizes a Query-Guided RPN for proposal generation, and a Query-Guided RCNN for top-proposal selection. This method is suitable for long-term tracking since the error cannot be accumulated due to the full-image search mechanism. A novel updating

method is proposed in [40] for tracking model update. It incorporates geometry, appearance and discriminative information. This method is beneficial to long-term tracking by updating the tracking model with the appropriate frames. Nevertheless, the tracking speed is affected by the time-consuming LSTM module. Dunnhofer et al. [41] perform long-time tracking by fusing two state-of-the-art short-term tracking approaches. Specifically, two target verifiers are utilized to estimate the target confidence from two tracking methods. Then a decision-making module is used to select the best tracking result. This combined method is more superior than two separate tracking methods.

The above-mentioned deep learning trackers belong to Siamese trackers. These methods are progressively replaced by Transformer-based one-stage tracking approaches due to earlier feature interactions and promising global modeling capability of Transformer trackers. Current state-of-the-art one-stage Transformer trackers [42], [43], [44] generally utilize an encoder for feature interaction. These top-notch methods feed the encoder with full representations of the template and search region. However, it is known that images have substantial amount of information redundancy, which makes it intractable to learn strong representations efficiently. Feeding full representations into the encoder results in heavy computational consumption as well as feature redundancy. Recent research [45] attempts to reduce the image redundancy by introducing masked image modeling (MIM) pre-training to vision. This work masks a high proportion of input patches and feeds the remaining visible patches into the encoder. This greatly alleviates the information redundancy of the input image and reduces pre-training time. Nevertheless, masked image modeling (MIM) is mostly utilized to reconstruct corrupted input images from visible

patches in the scope of image self-supervised pre-training. The pre-trained parameters can be transferred to other vision tasks for fine-tuning.

The early application of masked image modeling (MIM) on tracking also follows this recipe by transferring the masked autoencoders (MAE) [45] pre-trained ViT parameters to tracking models and fine-tuning these parameters for object tracking. Despite promising tracking performance, these tracking methods fail to fundamentally tackle the problem of image redundancy for tracking. Recently proposed MIM-based trackers [46], [47], [48], [49] are dedicated to developing new pre-training strategies specifically designed for object tracking. Similar to MAE, these trackers incorporate MIM into the pre-training of encoder-decoder type structures. The goal of the pre-training is to reconstruct the mask patches of template or search region. After pretraining, the decoder is discarded. The encoder is exploited to predict the tracking results with the help of a prediction head. However, these trackers are essentially the same as early MIM applications since they still strive to leverage MIM to obtain better pre-training parameters. None of them attempts to reduce the redundancy of image information during training or inference of tracking models. The success of MIM in the pre-training leads to a question: is it feasible to decrease the image redundancy in the process of training or inference to improve the long-term tracking performance with the aid of masked image modeling? This question will be answered in Chapter 4.

Table 1-2 demonstrates the representative long-term trackers regarding their performance, input features, backbone structure, etc. AUC represents the area under curve metric on the LaSOT long-term tracking dataset. DL denotes whether the tracking method is related to deep learning. NA indicates the item is not applicable for the specific method. From the

table, non-deep learning long-term trackers are poor in performance owing to the fact that these methods utilize hand-crafted features. Deep learning methods have transformed from Siamese frameworks with ResNet as the backbone to one-stage frameworks with the ViT encoder backbone. Full input features are fed into Siamese tracking frameworks during pretraining and training. In contrast, partial representations (e.g., 25% search tokens) are fed into one-stage frameworks in the process of self-supervised pretraining. However, full features are employed for training one-stage trackers. Therefore, more attempts should be made to reduce redundancy during training to further improve the long-term tracking results.

Table 1-2 Summary of adaptive long-term tracking methods

Methods	**DL**	**Year**	**AUC**	**Siamese or one-stage**	**Pretraining**	**Training**	**Backbone structure**
TLD [20]	No	2012	0.210	NA	NA	NA	NA
LT-FLO [34]	No	2013	-	NA	NA	NA	NA
PDCT [36]	No	2018	-	NA	NA	NA	NA
GlobalTrack [39]	Yes	2020	0.517	Siamese	Full features	Full features	ResNet-50
LTMU [40]	Yes	2020	0.572	Siamese	Full features	Full features	ResNet-50
τ [41]	Yes	2022	0.685	Siamese	Full features	Full features	ResNet-50
SimTrack [43]	Yes	2022	0.705	One-stage	Partial features	Full features	ViT-L encoder
OSTrack [44]	Yes	2022	0.711	One-stage	Partial features	Full features	ViT-B encoder
MAT [49]	Yes	2023	0.678	One-stage	Partial features	Full features	ViT-B encoder

1.3 Object tracking tailoring for various challenges

Most tracking approaches focus on improving overall tracking performance without developing specific strategies to address tracking challenges. Superior overall performance is fundamental to object tracking. However, tracking challenges (i.e., full occlusion, fast motion, illumination variation, etc.) are the most direct causes of tracking failures. Tracking challenges are prevalent in most tracking scenarios. Occlusion is one of the most extensively studied tracking challenges. Unlike occlusion, background clutter is less extensively studied. Background clutter indicates that multiple distractors in the scene resemble the target. It is one of the most challenging factors for adaptive object tracking [50]. Therefore, background clutter is the priority of this research.

Authors in [51] present a method to track the contour of the object in cluttered background by utilizing optical flow and edges. This approach can hardly achieve good tracking results when the background is severely cluttered. Panda et al. [52] strive to decrease the intensity and spatial resolution of images to mitigate the influence of distractors. This idea also weakens some significant representations of the target. Mueller et al. [53] propose to integrate the global context information into correlation filter-based tracking algorithms to cope with background clutter. This improves the baseline method substantially. LMCF [54] combines SVM and correlation filters for tracking, thereby possessing real-time performance and promising discriminative ability. LMCF proposes a multimodal target detection mechanism for background clutter. It indicates that the highest peak of the response map is not necessarily the target position. To tackle this problem, the detection mechanism redetects other peaks and utilizes the highest point of the second detection as the target position, thereby alleviating the impact of background clutter.

PGNet [55] is proposed to diminish the background involvement, where each pixel of the search region is matched with the whole template by similarity computation. The relations among neighboring pixels in the search region are not taken into account. On the contrary, KYS [56] increases background participation by introducing scene information. This approach categorizes and models three types of state vectors, including the target, distractors and the background. These state vectors are utilized throughout all the frames for reference. Thus, the object and distractors can be tracked simultaneously. This substantially reduces the misidentification between the target and distractors. However, KYS is not capable of accurately tracking the target in real-time when dense distractors appear or when more than one challenge arises, e.g., occlusion and background clutter. Nevertheless, dense distractors and other tracking challenges often appear simultaneously in real world scenarios, i.e., densely crowded pedestrians and vehicles at intersections. Tracking algorithms should be optimized to cope with these scenarios.

Afterwards, KeepTrack [57] and NeighbourTrack [58] are proposed to improve KYS [56] by developing more complicated tracking strategies. These two trackers also track the target and distractors simultaneously to tackle the problem of background clutter. The tracking performance is improved with the decrease of inference speed due to multiple tracking targets and complicated tracking frameworks. Moreover, the more distractors there are in the scene, the more time-consuming these trackers will be. Thus, these algorithms are not optimal when there are dense distractors in the scene.

Updating target appearance in a timely and accurate manner improves the discriminative capability of tracking algorithms under background clutter. The reason is that significant appearance variation of targets leads to mismatches between the target and distractors in

the case of background clutter. Therefore, appropriate update of target appearance contributes to coping with background clutter. Furthermore, no additional computation is required even in the presence of dense distractors. Therefore, timely target updating is suitable for handling dense distractors under background clutter.

To adapt to the appearance change of the target, previous trackers [59], [60] develop online-learning classifiers to enhance the tracker ability to distinguish the target from the background. The online classifiers are utilized to complement the offline classification results. The input features of online classifiers do not perform information interaction, thus lacking sufficient discriminative ability. Several Transformer-based trackers [30], [42], [61], [62] propose to adapt to target variation by introducing updateable dynamic templates. Representations of fixed and dynamic templates interact with features of search region, thereby generating more superior tracking results.

A recent method [63] increases the number of static templates and dynamic templates by multi-scale cropping two template frames. Varying target appearance and its spatial and temporal contexts are integrated to predict high-quality tracking results. Despite the high performance, this algorithm selects dynamic templates from intermediate frames without being labeled manually. The reliability of dynamic templates is not as high as that of fixed templates since fixed templates are annotated precisely. Therefore, target-related contents in dynamic templates might be obscured by a large amount of uninformative background noise. The increase in template number also causes longer processing time. Therefore, the background noise should be eliminated to highlight the target information and to reduce the running time of the tracking model. This is beneficial to the tracking model for dealing with background clutter.

Table 1-3 illustrates the representative tracking algorithms dealing with background clutter and the benefits and drawbacks of these methods. DL and AUC denote deep learning and the area under curve metric on OTB100 dataset. It is obvious that non deep learning trackers can not achieve high performance since they exploit hand-crafted features. Several deep learning approaches aim to track the target and background distractors simultaneously. These methods achieve superior results when dealing with background clutter. However, the tracking frameworks of these methods are excessively complicated. Additionally, the speed of the tracking models decreases with the increase of background distractors. These methods are not suitable for scenarios with dense distractors. ProContEXT [63] focuses on target updates. However, it introduces noises and increases complexity. Therefore, trackers should be improved to handle dense distractors under background clutter and strive to maintain real-time tracking.

Table 1-3 Representative tracking methods under background clutter

Methods	**DL**	**Year**	**AUC**	**Benefits**	**Drawbacks**
[51]	No	1996	-	Track target contour	Hand-crafted features
[52]	No	2011	-	Reduce image intensity and resolution	Weaken key target representations
CACF [53]	No	2017	0.598	Global context information	Hand-crafted features
LMCF [54]	Yes	2017	0.643	Multimodal target detection	Limited performance
PGNet [55]	Yes	2020	0.691	Less background noise	Without relations of neighboring pixels
KYS [56]	Yes	2020	0.695	Track the object and distractors simultaneously	More distractors, more time-consuming
KeepTrack [57]	Yes	2021	0.709	Track the object and distractors simultaneously	More distractors, more time-consuming
ProContEXT [63]	Yes	2023	-	Target appearance update, spatiotemporal contexts	Noises and reduced speed by dynamic templates

1.4 Research question and objectives

(A) Research question

According to above-mentioned research gaps, the research question is shown below.

How to develop adaptive tracking methods with enhanced performance for arbitrary objects in complicated tracking scenarios based on deep learning? Enhanced performance means higher success rate (AUC score) and precision; longer time for accurate tracking; and stronger ability to deal with tracking challenges.

(B) Research objectives

Objective 1

Develop a superior short-term tracking method integrating multiple information for category agnostic objects of complicated tracking scenarios by utilizing deep learning.

The aim is to enhance the adaptability and performance (e.g., AUC score, precision and normalized precision) of the tracking method for tracking arbitrary targets successfully in challenging sequences.

Objective 2

Develop an adaptive long-term tracking algorithm with improved tracking capability for arbitrary objects by incorporating masked image modeling (MIM) into the training process.

The goal is to reduce redundancy in the training process by feeding partial input representations for further improving the long-term tracking performance, and strive to maintain real-time long-term tracking.

Objective 3

Develop an adaptive tracking approach with admirable ability to cope with dense distractors under background clutter.

The goal is to improve the tracking method specifically to cope with background clutter (one of the most challenging scenarios) while still maintaining outstanding overall performance and real-time capability.

1.5 Research contributions

The main contributions of the dissertation are summarized as follows.

(i) A novel adaptive tracking algorithm (TGLC) is developed for the first research objective. A channel-aware convolution block (CCB) is applied to the feature fusion module to fully capture global and local information of the input features.

(ii) TGLC can also perceive channel information of feature representations. Both spatial and channel attentions are utilized to model essential information. TGLC achieves impressive tracking results with real-time running speed.

(iii) MIMTracking is proposed for the second research objective. It employs both encoder and decoder to fuse representations of template and search region more fully during training and inference.

(iv) Instead of feeding the encoder full representations, only partial search information is fed into the encoder for MIMTracking. This substantially alleviates the redundancy and urges the tracking network to learn useful information.

(v) A brand-new idea is explored by directly integrating masked image modeling into the training process of MIMTracking. This novel design further improves the long-term tracking performance.

(vi) ATPTrack is developed for the third objective. DTP and SRP blocks are proposed to prune dynamic templates and the search region alternately. After alternating pruning, target information is highlighted in both templates and search region.

(vii) A novel similarity ranking mechanism is exploited to select the most informative target-related tokens. This ensures the reliability of the information supplied by dynamic templates.

(viii) Experimental results indicate that ATPTrack demonstrates state-of-the-art tracking performance on multiple datasets, while substantially reducing computational consumption. ATPTrack also provides a new idea for dealing with distractors under background clutter.

1.6 Organization of the dissertation

The rest of the dissertation is organized as follows.

Chapter two reviews the advanced literature in related fields, e.g., adaptive tracking, target appearance update in visual tracking, multiple information integration, masked image modeling in vision and redundant token trimming.

Chapter three presents a novel tracking method dubbed TGLC for adaptive short-term tracking. This method develops an admirable feature fusion module for integrating global-local information and channel information. TGLC is compared with prevailing methods on popular datasets. The key components of TGLC are ablated respectively to determine their functionalities.

In chapter four, a masked image modeling enhanced tracking approach named MIMTracking is developed for long-term object tracking. Firstly, the overall tracking framework and key parts of MIMTracking are elaborated. Secondly, the implementation details of the tracker are described. Finally, a series of comparison and ablation experiments are conducted to assess the proposed tracking method.

Chapter five proposes an alternating token pruning tracker termed as ATPTrack for object tracking. This method highlights the target region for better handling tracking challenges especially background clutter with reduced computational complexity. The novel DTP and SRP blocks are first elaborated, following by the training and inference details. Then the tracking results are obtained from multiple benchmarks, accompanied by the performance under different tracking challenges.

In chapter six, the proposed two tracking algorithms (MIMTracking and ATPTrack) are evaluated on real-world video sequences captured by a camera. The tracking results are visualized frame by frame to determine the tracking performance of the proposed tracking algorithms.

Chapter seven concludes the research work and the main contributions of the dissertation. Finally, the limitations of the research and future improvements are discussed.

Chapter 2 Literature review

2.1 Adaptive tracking methods

Correlation filter (CF)-based trackers are dominant non-deep learning tracking approaches. The workflow of correlation filter-based tracking methods is summarized in Figure 2-1. It depicts two main steps for correlation filter related tracking methods, which are respectively target tracking and filter update.

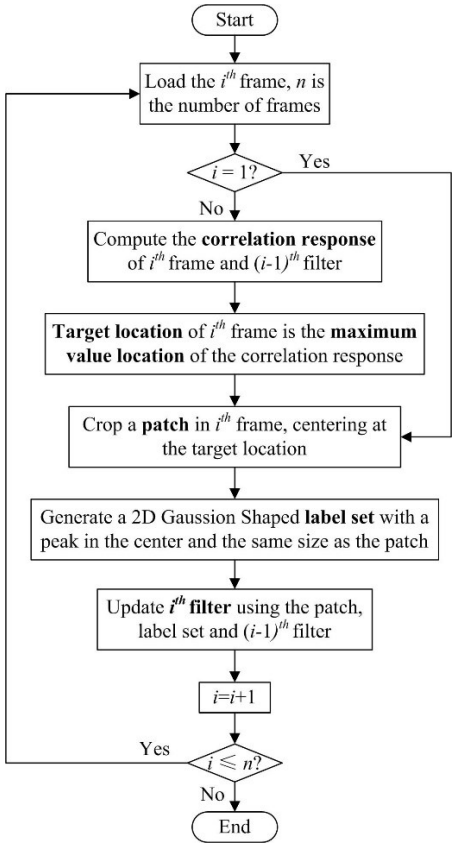


Figure 2-1 The workflow of CF-based trackers

MOSSE [64] is the first method that shows the efficiency and accuracy of correlation filter-based tracking methods. It also indicates the potential of correlation filters in tracking. Henriques et al. [65] propose a tracking-by-detection approach (CSK), which takes advantage of the circulant matrix theory and kernel trick. It accelerates the tracking process dramatically. Both MOSSE and CSK extract gray features for tracking. However, gray

features are sensitive to object appearance variation. In order to perform stronger tracking, KCF and DCF [21] replace gray features with HOG features. DCF is faster in tracking speed by using linear kernels, while KCF is more accurate with gaussian kernels. Danelljan et al. [66] utilize multi-channel color names (CN) to improve CSK. However, none of the above methods take into consideration the scale variation in object tracking. Subsequent research works such as SAMF [67] and DSST [68] adapt tracking to scale variation. DSST performs scale detection after completing translation detection. SAMF conducts scale detection and translation detection simultaneously. Subsequent studies in visual tracking [22], [69], [70] not only improve the tracking results but also maintain real-time tracking capability.

The above-mentioned correlation filter-based tracking methods utilize hand-crafted features. These preliminary low-level features constrain further improvement of the tracking performance. Deep learning accelerates the development of adaptive tracking algorithms due to the deployment of deep representations. Deep representations contain abundant semantic information, thereby enhancing the discriminative capability of the tracking methods for coping with more complicated tracking scenarios.

Deep learning-based tracking methods have experienced great development since they were proposed. Siamese-based tracking methods are gradually developed into the dominant tracking approaches. SiamFC [23] is the first Siamese-based approach. It mainly utilizes a shared fully convolutional network to learn the backbone features for exemplar image and candidate image. Then a similarity function is exploited to compare and generate the similarity map of two inputs. SiamFC is trained offline with large datasets, achieving promising tracking results.

Mainstream tracking methods can be divided into several categories according to the types of the prediction heads. Several methods such as ATOM [71] and DiMP [72] belong to IoU (Intersection over Union)-type regression, which employ IoU prediction for box refinement after a rough estimation. The architecture of ATOM contains an offline-trained object estimation component and an online-trained classification branch. The IoU predictor estimates the overlap between the object and the obtained bounding box. The online classifier can improve the discrimination ability for the object.

RPN-type tracking approaches [24], [73] integrate region proposal network (RPN) [74] into Siamese based tracking methods. These methods define many anchors and refine the anchors to perform accurate tracking. However, this type of methods only utilize positive samples (Overlapping between anchors and the object is larger than 0.6) for training of regression branch. It can hardly refine anchors with small overlap.

The above problem can be solved by anchor-free regression. Anchor-free regression does not require pre-defined anchors. Representative algorithms include SiamFC++ [75], Ocean [59], Alpha Refine [28] and RPT [60]. Instead of refining anchor boxes, anchor-free methods predict feature points or length parameters of the bounding boxes directly. Ocean [59] predicts distances of the object to the four edges of ground-truth boxes. This method also gives information about the object scale in each frame. Alpha Refine [28] is a refinement module that can considerably improve the tracking performance of base tracking methods. Its regression network predicts top-left corner and bottom-right corner of the bounding boxes.

The last category is mask-type regression. Popular algorithms consist of SiamMask [26], D3S [76] and DMB [77]. Mask-type tracking approaches have high tracking precision due

to pixel-wise estimation. These methods perform object tracking and segmentation simultaneously. The bounding boxes rotate freely according to the shape of the object.

In addition to tracking prediction heads, research on different backbones and feature fusion modules is also crucial for enhancing tracking methods. As mentioned in the previous chapter, Transformer-based algorithms [44], [78], [79] are becoming increasingly dominant due to their powerful global feature reasoning and interaction capabilities. Transformer related trackers experience a transition from separate to simultaneous representation generation and integration [43], [44]. After transition, a pretrained backbone is typically exploited to extract and interact representations synchronously from input images. A lightweight network is ultimately employed to estimate tracking results. This joint design delivers performance gains.

Pioneering Transformer-based tracking methods [29], [30] typically follow the Siamese tracking paradigm, which contain a CNN-based backbone for feature generation, a Transformer-based feature integration module and a prediction head for tracking result estimation. Self-attention and cross-attention in Transformer allow for better feature communication between the template and search region, thereby improving the tracking results significantly. SwinTrack [80] designs a Transformer only tracking framework to fully explore the potential of Transformer in visual tracking. It achieves prevailing tracking results on several well-known tracking benchmarks. AiATrack [81] develops an attention in attention mechanism to highlight meaningful attention weights and restrain unnecessary correlations. This mechanism is utilized to modify the existing self-attention and cross-attention modules and is applied to object tracking for leading tracking performance. A context-aware long-term context attention (LCA) module is presented in [62] for dealing

with target appearance variation, background clutter, etc. This greatly improves tracking results on multiple datasets.

One-stage trackers are recently proposed for unifying representation extraction and integration. MixFormer [42] designs a mixed attention module to build the tracking framework. This approach enables synchronous feature generation and interaction. Its multi-template online update mechanism also contributes to impressive tracking outcomes. Several trackers [43], [44], [46], [47], [49] are very relevant to our work, which employ vanilla ViT [82] encoder as the tracking backbone. These algorithms achieve superior tracking results. Authors in [44] propose a single-stream tracking method for visual object tracking. The template and search region are split into non-overlapping patches, followed by a linear projection process for mapping these patches into different tokens. Then a pretrained ViT encoder is utilized as the backbone for feature extraction and relation modeling. An early token elimination module is designed to remove background tokens in search region for alleviating the computational burden. SimTrack [43] develops a similar tracking framework to [44] by employing the ViT encoder as the tracking backbone. Uniquely, SimTrack crops and maps the central patch of the template image and concatenates it with all template and search tokens as the input of the backbone. This strategy improves the tracking performance.

DropMAE [46] extends the masked autoencoder (MAE) in matching-based tasks such as object tracking. It reconstructs the frames in videos by conducting spatial-attention dropout. DropMAE generates stronger pretraining parameters for tracking and segmentation. [47] and [49] develop novel self-supervised pretraining methods by reconstructing both the template and search region for visual tracking. These methods incorporate masked image

modeling into the pretraining process of the tracking model. After pretraining, the decoder is discarded. Therefore, only encoder is utilized for tracking model training and inference. ARTrack [78] and SeqTrack [83] consider visual object tracking as a sequence prediction process across all frames in a video. The coordinates of bounding boxes are obtained from a decoder rather than regular prediction heads. This idea aligns with the essence of object tracking. However, it is highly time-consuming due to complicated designs. GRM [79] develops a relation modeling approach for visual tracking. The token division block is proposed to categorize search region tokens into target-related tokens and background tokens, thereby enabling integration of key representations between the template and the search area. ROMTrack [84] proposes an object encoder for interacting features of the hybrid template, inherent template and the search area. It achieves state-of-the-art tracking performance on multiple datasets. The success of one-stage tracking approaches indicates that parallel feature generation and interaction contribute to the improvement of tracking performance.

2.2 Target updates in visual tracking

Timely appearance update is essential for visual tracking approaches. Currently, there are two prevalent strategies for embedding updated target information into the tracking framework. One approach involves introducing a branch for online learning to optimize the tracking model outputs, thereby enhancing the model's discriminative capability between the target and background distractors. In this category, some trackers [60], [71] attempt to develop an online trained classifier to predict the classification maps for the target and background regions. The latter aggregates weighted online and offline classification results to obtain the final classification score. DiMP [72] updates model

parameters by utilizing both the target and background features during inference, which designs sophisticated objective functions and optimization modules to improve tracking results. Another work [59] proposes an anchor-free tracking architecture. It presents a lightweight online network to incorporate changing target appearance in the process of inference. Despite achieving promising tracking results, the aforementioned methods fail to interact input representations of the online module sufficiently, thus restricting the identifying capability of the online module.

The other strategy directly renews target appearance by updating the reference templates. UpdateNet [85] proposes a compact convolutional network specifically for Siamese tracker updating. It aspires to select the most suitable template for the next search region among the initial frame, intermediate frame, and the current frame. With the success of Transformer in the field of computer vision, the attention mechanism has been introduced into Siamese-type tracking algorithms to fuse features from different input branches. TransT-M [61] designs Transformer-based feature integration module to aggregate representations from search image and multiple templates. An IoU-related mechanism is presented to determine whether to update the dynamic template. Similar to [61], STARK [30] also adopts static and dynamic templates to acquire preliminary and more recent object information updated by a separate estimation branch. Although both methods show impressive performance, they require a complicated two-step strategy for training the main structure and the updating head, respectively.

The latest advancement in Transformer-based tracking seeks to unify feature extraction and integration, which improves tracking results by a large margin. MixFormer [42] introduces synchronized feature generation and interaction for the first time. It designs a

specific attention mechanism and a score block for template updating. However, the score block of MixFormer is trained after the main framework as well, which results in excess training time. To tackle this issue, TATrack [62] and ProContEXT [63] propose to exploit the built-in classification scores to determine dynamic template update. This strategy alleviates the training burden. Inspired by these two trackers, we also adopt built-in scores to renew dynamic templates. Differently, our tracker explores diverse update intervals and handles dynamic templates in a different way.

2.3 Integration of multiple information

Multiple information integration enhances the discriminative ability of visual models. BoTNet [86] proposes a Bottleneck Transformer (BoT) block to build a new backbone framework, which incorporates self-attention into the bottleneck block of ResNet. This method is capable of extracting global and local feature information and achieves state-of-the-art performance in object detection, instance segmentation and image classification. Xu et al. [87] present a co-scale image classification method, which utilizes depth-wise convolutions to build position encodings and relative position encodings for transformer attentions. A great number of serial and parallel modules are employed to estimate multiple scales. This proposed method outperforms existing attention-based or convolution-based classification approaches. CoAtNet [88] introduces an efficient combination of ConvNets and Transformer attentions, which retains superior generalization ability and model aptness. This algorithm realizes state-of-the-art classification performance. Conformer [89] develops a double-branched structure with convolution network and attention network in separate branches. Global feature representations and locally detailed features are communicated in a feature coupling block. This method leverages the advantages of both

convolution and Transformer attention, retaining exceptional performance for classification and detection. Mehta et al. [90] design a light-weight network combining CNNs and transformer for light mobile devices. This network learns global representations with time-efficient transformer and demonstrates admirable performance on ImageNet and object detection datasets.

Woo et al. [91] develop a network integrating channel attention and spatial attention mechanism, which is suitable for almost all ConvNets. It considerably improves the detection and classification ability. An advantageous algorithm is introduced in [92] for object detection of UAV images by integrating CNNs and transformer attention. The integrated algorithm achieves advanced detection performance. Zhang et al. [93] present an object detection method by adopting deep convolution network as backbone and attention mechanism for emphasizing important features. A cell segmentation approach is developed in [94] by fusing features of CNNs with representations from Transformer attention, which improves the segmentation mIoU score. U-Net Transformer [95] is proposed for precise medical image segmentation, which employs self-attention and cross-attention to enhance the conventional U-Net for establishing long-range relations. This method brings improvements to segmentation accuracy.

The aforementioned methods are capable of leveraging the advantages of attention and convolutions. Global and local feature representations are established for improving the performance of vision tasks. However, most approaches are developed for building a general backbone architecture for vision tasks, e.g., image classification, instance segmentation and object detection. Multiple information fusion can be attempted in visual object tracking for improving tracking performance. This will be explored in chapter 3.

2.4 Masked image modeling in vision

Masked image modeling (MIM) is capable of reconstructing corrupted images in the field of self-supervised pre-training [96]. BEiT [97] maps the input image into a series of representative tokens and randomly masks a percentage of patches before sending the rest to the backbone. The aim is to reconstruct the tokens of the input image. The pre-trained encoder is then fine-tuned for downstream tasks such as semantic segmentation. MAE [45] proposes an asymmetric framework for self-supervised pre-training. This method utilizes an extremely high masking rate of 75% and predicts the pixels of the masked patches rather than tokens. MAE achieves impressive fine-tuning and linear probing accuracy and demonstrates strong scaling performance as well. Another work in [98] feeds the encoder both the visible and masked patches for representation learning. This method manifests that simply designed structures are sufficient to learn admirable pre-training representations. MIMDET [99] exploits MIM strategy to train a detection model without the need for self-supervised reconstruction. It achieves prevailing detection performance. Huang et al. [100] integrate masked image modeling into hierarchical Vision Transformers (ViTs). This method develops a group window attention mechanism to alleviate the computational complexity of self-attention. It also proposes sparse convolution for processing incomplete input information in masked images. UM-MAE [101] presents a uniform masking strategy for MAE pre-training of hierarchical ViTs. Uniform sampling and secondary masking are two main components in UM-MAE. This method possesses promising transferability on downstream visual tasks and increases the efficiency of the pretraining process. ConvMAE [102] combines masked convolution and Transformers in masked autoencoder pretraining. It also proposes a block-wise masking strategy for

efficient pretraining. The fine-tuned model achieves superior object detection performance. LoMaR [103] proposes to perform local masked reconstruction within a tiny region for the purpose of reducing pretraining time. This reconstruction strategy has strong adaptability and flexibility to other self-supervised pretraining methods. MixMAE [104] presents a novel self-supervised pretraining approach by replacing the invisible patches of one image with patches of the other image in the same positions. Then it reconstructs both images from the combined image. This process improves the pretraining representations and efficiency. The attempts of masked image modeling in the field of tracking are elaborated in chapter 4.

2.5 Redundant token simplification

Token pruning has become increasingly important since the demand for speed in visual tasks expanded. Visual token trimming aims to identify tokens with strong semantics and discard uninformative background tokens, thereby reducing the processing time for visual models. Most token trimming algorithms are incorporated into Vision Transformer (ViT) [82] or its variants to mitigate the computational complexity as ViT is capable of processing sequences of varying lengths. PSViT [105] equips ViT with token pooling and weight sharing strategies to reduce spatial and layer-wise redundancy. Evo-ViT [106] develops a dual-branch token propagation approach. The class token is utilized to differentiate between relevant and irrelevant tokens. Only relevant tokens are fed into ViT with a symbolic token communicating with irrelevant tokens. DynamicViT [107] integrates an importance assessment scheme to ViT for eliminating inconsequential tokens. It reduces the FLOPs significantly. A-ViT [108] estimates a termination score for each token of every layer. A token is removed when its accumulated termination score surpasses

a certain threshold. STA [109] presents a plug-and-play token trimming approach for video Transformer, which alleviates temporal and spatial redundancy.

Token merging is similar to token pruning. The former proposes to merge uninformative tokens instead of removing them. ToMe [110] presents a matching-based approach to merge adjacent tokens with similar contents. It improves the throughput of current ViT methods and works off-the-shelf with no training required. Authors in [111] develop a token merging algorithm that identifies redundant tokens and represents the tokens with one token. This method can be embedded into the layers of ViT for accelerating the calculation. The token merging rates can also be controlled by the presented learnable gates. ToFu [112] integrates the advantages of token merging and token trimming. A novel merging algorithm is presented to improve the average merging method utilized in previous works. It also retains the norm distribution of tokens. A new token compression approach is proposed in [113] for large Transformer models (e.g., ViTs). It performs token pruning and token merging simultaneously with learnable compression rates for different layers. This method achieves impressive results with a significant drop in computational FLOPs. Despite a superior speed-performance balance, these token simplification algorithms are devised for a single image or a video sequence.

Tracking scenarios typically require a search area and one or more templates as the reference. Several trackers [44], [46], [63] propose to prune redundant tokens of the search area gradually, which clearly decreases the training and inference time. Nevertheless, the introduction of dynamic templates in [63] for considering target variation notably decreases the speed of the tracking model and brings uninformative noise. Therefore, appropriate pruning of both search area and dynamic templates will be an insightful research topic for

visual tracking. To the best of our knowledge, research on this topic is exceedingly limited. As demonstrated in charter 5, extensive research on the topic is conducted for object tracking.

2.6 Summary

The related works of adaptive visual tracking and existing problems of these approaches are discussed in this chapter. First, the development of adaptive tracking methods is thoroughly reviewed. Subsequently, a review of research in related areas (e.g., target appearance updates in tracking, multiple information fusion, masked image modeling and token simplification) is conducted to assess the current status of these research fields, containing the strengths and weaknesses. It aims to identify impactful directions for object tracking studied in the dissertation.

Chapter 3 Multiple information fusion for adaptive tracking

3.1 Methodology

3.1.1 Overall tracking framework of TGLC

TGLC develops a novel feature fusion module incorporating global long-range representations and locally detailed features of images. Channel information is also exploited in the feature fusion module to improve the tracking performance. The overall tracking framework of TGLC is shown in Figure 3-1, which consists of a shared backbone for feature extraction of template and search region, a feature fusion module to aggregate backbone features of two branches, and a key point prediction head for generating bounding boxes. The first four convolution blocks of ResNet50 [114] are exploited to build the backbone. In order to retain more detailed feature information, the stride of the down-sampling of the fourth block is set as 1. Additionally, a dilated convolution with a dilation rate of 2 is employed to replace the second convolution of the fourth block to obtain a wider field of view without increasing the computational cost. The accumulated stride of the backbone is 8. The template and search region are separately cropped from the first frame and current frame. Their backbone features are further extracted by the shared backbone.

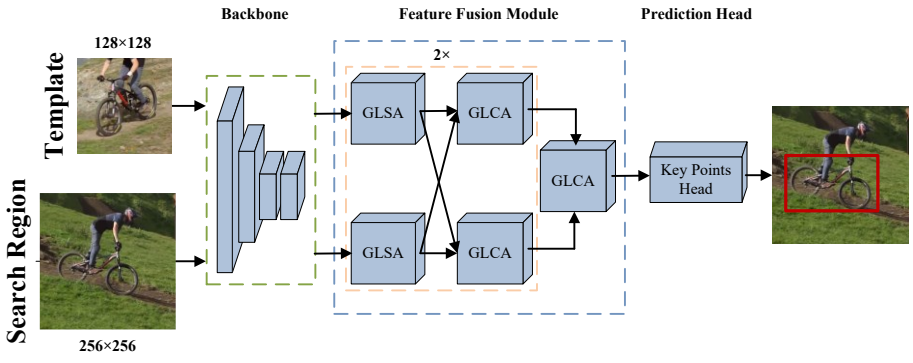


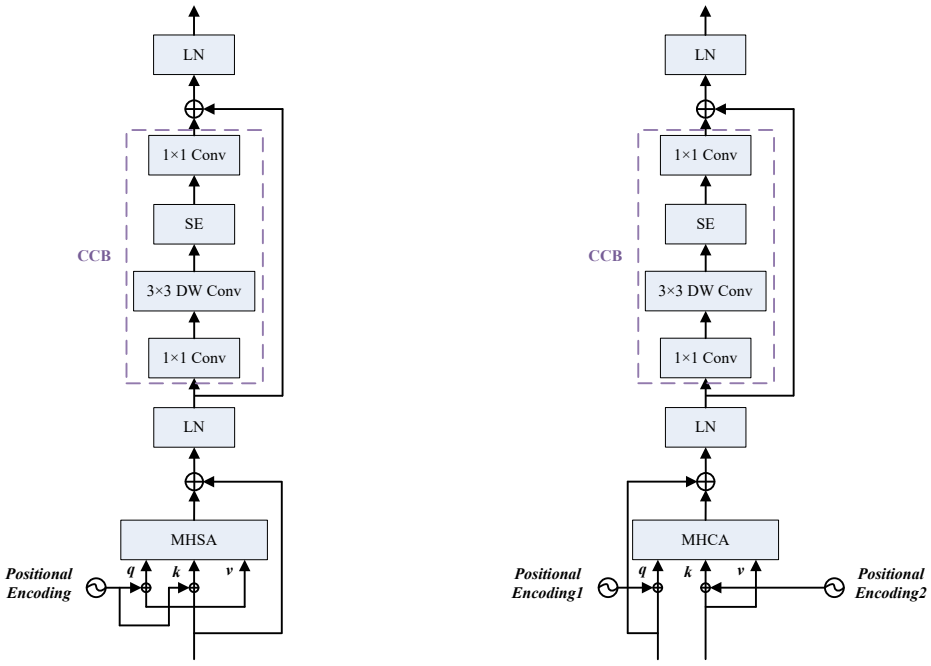
Figure 3-1 Structure of the proposed TGLC

3.1.2 Multiple feature fusion

$I_{t1} \in \mathbb{R}^{[C_1, \frac{H_t}{8}, \frac{W_t}{8}]}$ and $I_{s1} \in \mathbb{R}^{[C_1, \frac{H_s}{8}, \frac{W_s}{8}]}$ respectively represent the backbone features of template and search region, where C_1 is the channel number (1024). H_t, W_t, H_s and W_s denote separately heights and widths of the template and search region. These two backbone features are dimensionally reduced to 256 channels for the purpose of improving computational efficiency. Then they are flattened spatially, resulting in two new features, $I_{t2} \in \mathbb{R}^{[C_2, \frac{H_t}{8} \times \frac{W_t}{8}]}$ and $I_{s2} \in \mathbb{R}^{[C_2, \frac{H_s}{8} \times \frac{W_s}{8}]}$. C_2 is the new channel number (256). Flattening is necessary for Transformer attention since flattened feature maps are more computationally efficient than original feature maps for attentions. However, flattening operation will cause the loss of one dimension of images, which will lose the spatial connections among pixels. In order to solve this problem, positional encoding is incorporated and added to the original feature maps before flattening. Positional encoding generated by sine and cosine functions [115] is utilized in our feature fusion module. After dimension reduction and feature flattening, I_{t2} and I_{s2} are fed into the feature fusion module.

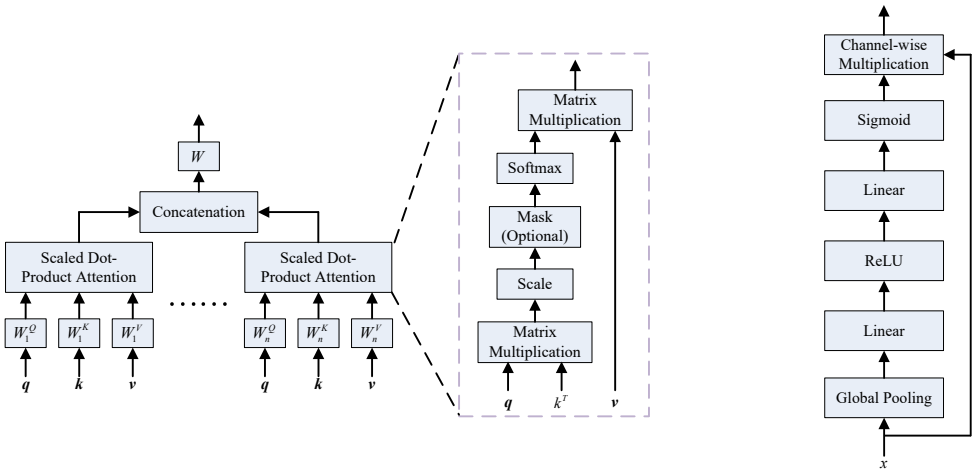
The feature fusion module is demonstrated in Figures 3-1, 3-2 and 3-3. Self-attention and cross-attention merely focus on global information of feature maps, ignoring the local detailed information. Local features are especially important when there are tracking challenges such as similar distractors. In order to tackle this issue, a channel-aware convolution block (CCB) is applied behind each multi-head self-attention and cross-attention. The block consists of a 1×1 convolution, a 3×3 depth-wise convolution, a squeeze-and-excitation layer (SE layer) [116] and a 1×1 convolution. Convolutions are capable of capturing local features of targets with better translation invariance and

generalization ability, which can be exploited to complement the deficiencies of transformer attentions.



(a) GLSA (b) GLCA

Figure 3-2 Structures of GLSA and GLCA



(a) MHSA and MHCA (b) SE in CCB

Figure 3-3 Structures of MHA and SE in CCB

Figures 3-2 and 3-3 show the detailed structures combining attention and convolutions in global-local self-attention (GLSA) and global-local cross-attention module (GLCA). The difference between GLSA and GLCA is that the query (q), key (k) and value (v) in GLSA are from a single branch, while they are from two branches in GLCA. The first block in GLSA and GLCA is multi-head attention, which includes multi-head self-attention (MHSA) and multi-head cross-attention (MHCA). The structures of MHSA and MHCA are identical, while the query (q), key (k) and value (v) follow the difference of GLSA and GLCA. Multi-head attention comes from scaled dot-product attention, the scaled dot-product attention is described in Eq. (3-1).

$$Att(q,k,v) = \text{softmax}(qk^T(d_k)^{(-\frac{1}{2})})v \tag{3-1}$$

where q,k,v are three matrices and $q \in \mathbb{R}^{n_q \times d_k}$, $k \in \mathbb{R}^{n \times d_k}$, $v \in \mathbb{R}^{n \times d_v}$. n_q and n are respectively the sequence length for the query and the key. d_k and d_v denote dimensions for the key and the value, $d_k = d_v$ in this case. T denotes transpose of a matrix. Softmax function is used to map all elements into $(0,1)$. For i^{th} element $f_i\big|_{i\in\Omega}$, the softmax value of f_i is shown in Eq. (3-2).

$$\text{softmax}(f_i) = \frac{\exp(f_i)}{\sum_{j\in\Omega}\exp(f_j)} \tag{3-2}$$

Multi-head attention exploits multiple scaled dot-product attentions to obtain information from numerous subspaces [31]. The calculation process is demonstrated in Figure 3-3(a). q,k,v are linearly transformed into different inputs for several single-head attentions by n sets of linear transformation matrices W_i^Q, W_i^K and W_i^V, $i \in n$, where n denotes the number of heads. Outputs from different heads are concatenated and then transformed by

another linear transformation matrix W. The process is calculated in Eq. (3-3) and Eq. (3-4).

$$MH\_Att(q,k,v) = \text{Concat}(Head_1,\ldots,Head_n)W \tag{3-3}$$

$$Head_i = Att(qW_i^Q, kW_i^K, vW_i^V) \tag{3-4}$$

The input of multi-head attention is added to the output to form a residual connection, which can alleviate gradient vanishing and network overfitting problems. There are two cases in MHSA of GLSA, which results from two input branches, the template features I_t and search region features I_s. MHSAs of search region branch and template branch in a residual form are illustrated in Eq. (3-5) and Eq. (3-6), respectively.

$$Y_s = I_s + MH\_Att(q = I_s + \Gamma_s, k = I_s + \Gamma_s, v = I_s) \tag{3-5}$$

$$Y_t = I_t + MH\_Att(q = I_t + \Gamma_t, k = I_t + \Gamma_t, v = I_t) \tag{3-6}$$

where Γ_s and Γ_t are position encodings for I_s and I_t. There are also two types of MHCAs in GLCA where the query is from one branch, the key and the value are from the other branch, which are demonstrated in Eq. (3-7) and Eq. (3-8).

$$Y_{s\_t} = I_s + MH\_Att(q = I_s + \Gamma_s, k = I_t + \Gamma_t, v = I_t) \tag{3-7}$$

$$Y_{t\_s} = I_t + MH\_Att(q = I_t + \Gamma_t, k = I_s + \Gamma_s, v = I_s) \tag{3-8}$$

Layer Normalization (LN) is utilized after multi-head attention for speeding up training and convergence of the model. Eq. (3-9) demonstrates the calculation process for LN, where x is the input with the mean μ and standard deviation δ. N and ε denote the number of channels of x and a small positive value to prevent a denominator of 0, respectively. α and β are learnable scaling parameters.

$$LN(x) = \alpha \cdot \frac{x - \mu}{\delta} + \beta \tag{3-9}$$

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{3-10}$$

$$\delta = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 + \varepsilon} \tag{3-11}$$

The next module is the channel-aware convolution block (CCB), which is employed to capture local feature information, complementing with the multi-head attentions that focus on global feature information. There is another superiority of this module, which is the ability to perceive channel information of the feature maps. This superiority is achieved by SE block shown in Figure 3-3(b). Convolutions and attentions can only exploit spatial information of the feature maps, which fail to leverage feature channel information. The use of SE block in CCB provides plentiful information of different channels for tracking models.

Assuming that x is the input of CCB, which is reshaped into a 3D feature map $\tilde{x}$ with shape of $[\tilde{C}, \tilde{H}, \tilde{W}]$. $\tilde{C}$, $\tilde{H}$ and $\tilde{W}$ denote channel number, height, and width of $\tilde{x}$, respectively. $\tilde{C} = 256$, $\tilde{H} = \tilde{W} = 16$ for template branch, $\tilde{H} = \tilde{W} = 32$ for search region branch. The first part of CCB is a convolution with kernel size of 1, which is utilized to expand the channel dimension from 256 to 1024. The convolution is followed by a batch normalization and a ReLU nonlinear activation function. This 1×1 convolution is depicted in Eq. (3-12).

$$h[n,k,l] = \sum_{m}\sum_{i=1}\sum_{j=1} K[m,n,i,j] \cdot \tilde{x}[m, k+i-1, l+j-1] \tag{3-12}$$

where K is the convolution kernel, h is the convolution output. $m = 256$, $n = 1024$ denote the number of input channels and output channels, respectively. $[k, l]$ and $\cdot$ represent the coordinates of any point in $\tilde{x}$ and multiplication. Batch normalization is similar to layer normalization, the difference is that N denotes batch size for batch normalization in Eq. (3-10) and Eq. (3-11). ReLU is defined as Eq. (3-13).

$$ReLU(x) = \max(0, x) \tag{3-13}$$

The second part of CCB is a 3×3 depth-wise convolution, which is demonstrated in Eq. (3-14). $\hat{K}$ denotes the convolution kernel with size of 3. The channels of the kernel $\hat{K}$, the feature map $\tilde{x}$ and the output map h have a one-to-one correspondence in depth-wise convolutions, which makes them simpler and more efficient than regular convolutions. A batch normalization and a ReLU activation function are used after the depth-wise convolution.

$$h[n, k, l] = \sum_{n} \sum_{i=0}^{2} \sum_{j=0}^{2} \hat{K}[n, i, j] \cdot \tilde{x}[n, k + i - 1, l + j - 1] \tag{3-14}$$

The third part of CCB is the SE block, as shown in Figure 3-3(b). Input x is first fed into a global average pooling module, as demonstrated by Eqs. (3-15), (3-16) and (3-17). This module calculates the average value y_t along two spatial dimensions W and H of x_t, where x_t is the feature map of t^{th} channel of x. y and C denote the output and the number of channels in x, respectively. Two linear layers (FC layers), W_1 and W_2, are employed after global pooling to build weights of channels, as shown in Eq. (3-18). $\Re$ is a ReLU activation function. $\text{Sigmoid}\left[W_2\left(\Re(W_1 y)\right)\right]$ is the final channel representation, which is

applied to the input x as channel attention weights by channel-wise multiplication. $\hat{y}$ is the output of SE block.

$$x = \left\{ x_t \middle|_{t\in\{1,2,\ldots,C\}} \right\} \tag{3-15}$$

$$y_t = \frac{1}{W \times H} \sum_{i=1}^{W} \sum_{j=1}^{H} x_t(i,j) \tag{3-16}$$

$$y = \left\{ y_t \middle|_{t\in\{1,2,\ldots,C\}} \right\} \tag{3-17}$$

$$\hat{y} = \text{Sigmoid}\left[W_2 \left(\Re(W_1 y) \right) \right] \cdot x \tag{3-18}$$

$$\text{Sigmoid}(t) = \frac{1}{1+\exp(-t)} \tag{3-19}$$

The final part of CCB is another 1×1 convolution, followed by a batch normalization layer. The channel number decreases from 1024 to 256. The input of CCB is added to the output to form a residual connection. Then, a layer normalization is utilized to the residual output of CCB, which generates the final output of the GLSA and GLCA.

3.1.3 Key point prediction head

The fused features are fed into the key point prediction head to generate the bounding boxes for the object. The prediction head is inspired by [60], which mainly predicts the required key points for construction of the bounding boxes. The main differences with [60] are two-folds. (1) Depth-wise correlation adopted in [60] is not utilized in our prediction head. (2) Only features from the last scale are exploited in our prediction head, while multiscale features are utilized in the prediction head of [60]. The detailed structure of the key point prediction head is demonstrated in Figure 3-4, which consists of a regression branch and a classification branch. Two 3×3 convolution blocks are employed for two branches, respectively. Each convolution block is composed of a 3×3 convolution shown in Eq. (3-

20), a batch normalization and a ReLU activation function. Eq. (3-20) resembles Eq. (3-14), while the difference is that channels of kernel $\bar{K}$ and input ζ are not necessarily corresponding. p and q denote respectively the number of input channels and that of output channels, which are both 256 and remain unchanged for all the components of the prediction head. For the regression branch, another 3×3 convolution block and a 1×1 convolution are used to predict coordinates of the first set of key points $kp1$. As shown in Eq. (3-21), the offset $(\Delta i, \Delta j)$ is calculated by $kp1$ and $base\_offset$, ($base\_offset$ is an initial set of points with values between -1 and 1).

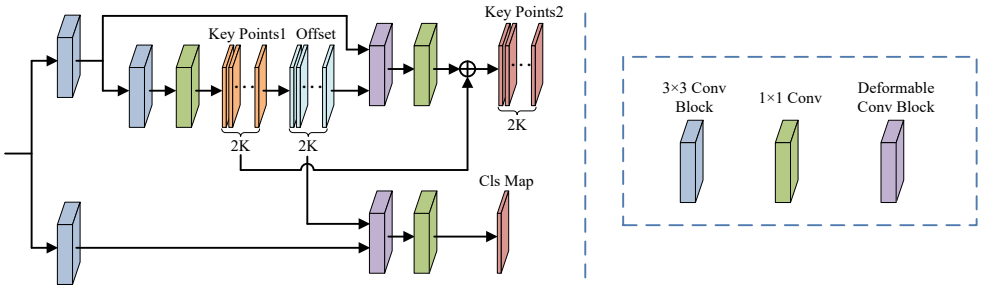


Figure 3-4 Structure of key point prediction head

$$h[q,k,l] = \sum_{p} \sum_{i=0}^{2} \sum_{j=0}^{2} \bar{K}[p,q,i,j] \cdot \zeta[p,k+i-1,l+j-1] \tag{3-20}$$

$$(\Delta i, \Delta j)\big|_{offset} = kp1 - base\_offset \tag{3-21}$$

Then the estimated offset $(\Delta i, \Delta j)$ and feature map $\hat{\zeta}$ are fed into a deformable convolution block, which includes a 3×3 deformable convolution shown in Eq. (3-22), a batch normalization and a ReLU activation. The offset is added to the regular sampling locations to expand the receptive fields of original convolutions, which is a superiority of deformable convolutions. Nevertheless, positions after the offset might not be integers, which is tackled by bilinear interpolation. The interpolation process is manifested by Eq.

(3-23), where $s_x = k + i - 1 + \Delta i$, $s_y = l + j - 1 + \Delta j$, denote the coordinates of a random non-integer position. $[t_x^z, t_y^z]$ denotes the coordinates of one of integer positions in $\hat{\zeta}$, where N represents a collection of all integer positions of $\hat{\zeta}$. Eq. (3-24) is utilized to calculate the g in Eq. (3-23).

$$\hat{h}[q,k,l] = \sum_{p} \sum_{i=0}^{2} \sum_{j=0}^{2} \hat{K}[p,q,i,j] \cdot \hat{\zeta}[p, k+i-1+\Delta i, l+j-1+\Delta j] \tag{3-22}$$

$$\hat{\zeta}[p, s_x, s_y] = \sum_{z \in N} g(s_x, t_x^z) \cdot g(s_y, t_y^z) \cdot \hat{\zeta}[p, t_x^z, t_y^z] \tag{3-23}$$

$$g(u,v) = \max\left(1 - |u - v|, 0\right) \tag{3-24}$$

Following the deformable convolution block, a 1×1 convolution is further exploited to map the number of output channels to $2K$. ($K = 9$, denotes the number of key points). The output of the 1×1 convolution is added to the first set of key points $kp1$ to obtain the second group of key points $kp2$. The classification branch also utilizes a deformable convolution block and a 1×1 convolution to estimate the final classification map (Cls Map), which is for foreground-background classification.

In order to be consistent with the training labels, the moment-based method [117] is utilized to transform the key points $kp1$ and $kp2$ into bounding boxes. A is assumed to be a set of predicted key points. The coordinates of the top-left corner (x_{tl}, y_{tl}) and bottom-right corner (x_{br}, y_{br}) of the bounding box are calculated by Eqs. (3-26) and (3-27), respectively. μ_x, δ_x, μ_y and δ_y are respectively the mean values and standard deviations for all x_n and y_n in A. β_x and β_y are two learnable parameters, which are used to adaptively adjust the scales of the transformed bounding boxes.

$$A=\left\{(x_n,y_n)\Big|_{n\in\{1,2,\ldots,K\}}\right\} \tag{3-25}$$

$$(x_{tl},y_{tl})=(\mu_x-\delta_x\cdot e^{\beta_x},\mu_y-\delta_y\cdot e^{\beta_y}) \tag{3-26}$$

$$(x_{br},y_{br})=(\mu_x+\delta_x\cdot e^{\beta_x},\mu_y+\delta_y\cdot e^{\beta_y}) \tag{3-27}$$

3.1.4 Loss function

As shown in Eq. (3-28), the weighted sum of generalized IoU (GIoU) loss [118] and binary cross entropy (BCE) loss is exploited to train the proposed tracker. μ and η are weights. y_{init_box} and y_{refine_box} are respectively corresponding bounding box coordinates of key point sets $kp1$ and $kp2$, y_{cls} is predicted classification map. $\hat{y}_{bbox}$ and $\hat{y}_{cls}$ are ground-truth bounding box coordinates and ground-truth classification label, respectively. μ and η are set as 5 and 10 in the training process.

$$Loss_{total}=\mu\left(Loss_{GIoU}(y_{init\_box},\hat{y}_{bbox})+Loss_{GIoU}(y_{refine\_box},\hat{y}_{bbox})\right)+\eta Loss_{BCE}(y_{cls},\hat{y}_{cls}) \tag{3-28}$$

GIoU loss is demonstrated in Eqs. (3-29) and (3-30), where $y,\hat{y}\in\mathbb{R}^n$, A^c is the area of the smallest enclosing box between y and $\hat{y}$. BCE loss is calculated in Eq. (3-31), where y and $\hat{y}$ are respectively the predicted output class probability and ground-truth class label.

$$Loss_{GIoU}(y,\hat{y})=1-GIoU(y,\hat{y}) \tag{3-29}$$

$$GIoU(y,\hat{y})=\frac{|y\cap\hat{y}|}{|y\cup\hat{y}|}-\frac{A^c-|y\cup\hat{y}|}{A^c} \tag{3-30}$$

$$Loss_{BCE}(y,\hat{y})=-\frac{1}{N}\sum_{i=1}^{N}\left[\hat{y}_i\log(y_i)+(1-\hat{y}_i)\log(1-y_i)\right] \tag{3-31}$$

As illustrated in Figure 3-5, any candidate position (x,y) in the predicted classification map $y_{cls}^{1\times H_f\times W_f}$ or regression maps $y_{init\_box}^{4\times H_f\times W_f}$, $y_{refine\_box}^{4\times H_f\times W_f}$ corresponds to a position (R_x,R_y) of

the search region. The relationship χ between the two positions is calculated in Eq. (3-32). s is the accumulated stride of the tracking framework. H_f, W_f, H and W are respectively the heights and widths of the predicted feature maps and the search region.

$$(R_x, R_y) = \left((x - \frac{W_f}{2}) \cdot s + \frac{W}{2}, (y - \frac{H_f}{2}) \cdot s + \frac{H}{2} \right) \tag{3-32}$$

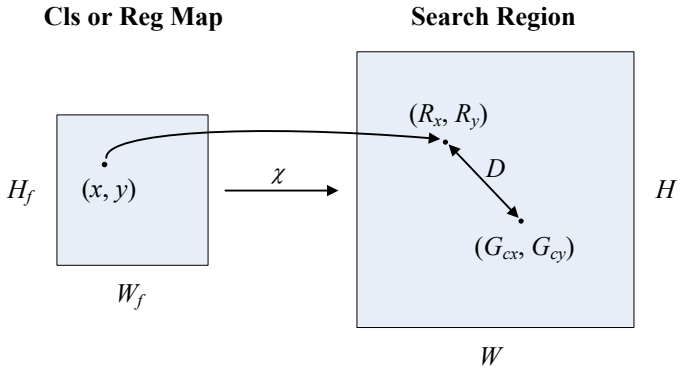


Figure 3-5 Mapping relation between output map and search region

As shown in Eq. (3-33), the distances between all $(R_x(i), R_y(j))$ and ground-truth box center (G_{cx}, G_{cy}) are calculated to determine the positive and negative sample positions in the initial regression output y_{init_box}. Positions closest to (G_{cx}, G_{cy}) after mapping are set as positive samples. The remaining positions are set as negative samples. Only positive sample positions are utilized to calculate initial bounding box loss. $\hat{y}_{bbox} \in \mathbb{R}^{4 \times H_f \times W_f}$ is the ground-truth label for y_{init_box} and y_{refine_box}, which is obtained by repeating the ground-truth coordinates $[G_{x1}, G_{y1}, G_{x2}, G_{y2}]$ $H_f \times W_f$ times.

$$\min_{\forall i \in [1, W_f] \wedge j \in [1, H_f]} \left\{ (R_x(i) - G_{cx})^2 + (R_y(j) - G_{cy})^2 \right\} \tag{3-33}$$

As depicted in Eq. (3-34), the Intersection over Unions ($IoUs$) between initial regression boxes y_{init_box} and ground-truth boxes $\hat{y}_{bbox}$ are calculated to determine the classification

label $\hat{y}_{cls}$ and positive and negative samples of classification output y_{cls} and refined regression output y_{refine_box}. $IoUs$ consists of $H_f W_f$ IoU_i, and $IoU_{\max}$ denotes the maximum IoU among $IoUs$. Each IoU_i corresponds to a label l_i, and the relation is demonstrated in Eq. (3-36). There is at least one label with the value of 1 even if there is no IoU greater than 0.5, which ensures that positive samples can be generated continuously for training. The collection of all labels forms the classification label $\hat{y}_{cls}$. All positive sample positions ($l_i = 1$) and negative sample positions ($l_i = 0$) are employed for calculating classification loss, while only positive sample positions ($l_i = 1$) are utilized to compute the refined bounding box loss for y_{refine_box}.

$$IoUs = \frac{\left| y_{init\_box} \bigcap \hat{y}_{bbox} \right|}{\left| y_{init\_box} \bigcup \hat{y}_{bbox} \right|} \in \mathbb{R}^{H_f \times W_f} \tag{3-34}$$

$$IoU_{\max} = \max\left\{ IoU_i \middle| IoU_i \in IoUs \right\} \tag{3-35}$$

$$l_i = \begin{cases} 0, & 0 \le IoU_i < 0.4 \\ 1, & IoU_i \ge 0.5 \\ 1, & IoU_i = IoU_{\max} \end{cases} \tag{3-36}$$

$$\hat{y}_{cls} = \left\{ l_i \middle| i \in [1, H_f W_f] \right\} \tag{3-37}$$

3.2 Comparison experiments

Implementation details. The proposed tracking TGLC is trained on the training sets of 4 publicly available datasets, GOT-10k [119], TrackingNet [120], LaSOT [50] and COCO [121]. The network is validated simultaneously on the validation set of GOT-10k every 10 epochs during training. The backbone network is pre-trained on ImageNet [122] to equip the network with preliminary feature extraction capability. Center jitter and scale jitter are

utilized for image augmentation. The center jitter factor and scale jitter factor are respectively set as 3.5 and 0.5 for the search region, and the two factors are set as 0 for the template. The network is trained on a workstation with a Nvidia RTX A6000 GPU and Intel Core i9 CPU. Ubuntu 20.04 is used as the operating system. During training, two frames within the same video in a dataset are sampled and cropped as template and search region, respectively. The template and search region are cropped into square patches with the sizes of 128 pixels and 256 pixels, respectively. The tracking network is trained for 250 epochs with 1000 iterations for one epoch by utilizing AdamW optimizer [123]. The batch size is set as 50. The learning rates of backbone and the remaining parameters are separately set as 10^{-5} and 10^{-4}. Both rates start to decay after 150 epochs with a weight decay factor of 10^{-4}. The summary of the implementation details is shown in Table 3-1.

Table 3-1 Implementation details

Settings	Values
Center jitter factor	3.5
Scale jitter factor	0.5
GPU	One Nvidia RTX A6000
CPU	Intel Core i9
The operating system	Ubuntu 20.04
Template size	128 × 128
Search region size	256 ×256
Epoch number	250
Iterations per epoch	1000
Batch size	50
Optimizer	AdamW
Learning rate of backbone	10^{-5}
Learning rate of other parameters	10^{-4}
Learning rate decay epoch	150
Weight decay factor	10^{-4}

Comparison experiments. The proposed TGLC is evaluated and compared with current state-of-the-art tracking algorithms on 5 widely used tracking datasets. The tracking results are shown below. All comparison methods mentioned on 5 datasets utilize a template to assist target localization during tracking.

GOT-10k [119]. GOT-10k dataset contains 560 common objects and 87 motion patterns, which utilizes 10000 and 180 video sequences respectively for training and testing. It provides completely different object categories for training and testing, which evaluates the performance of tracking methods to unknown targets. There are 3 metrics utilized to evaluate tracking performance on GOT-10k, which are respectively mAO, $mSR_{0.5}$ and $mSR_{0.75}$. Average overlap (AO) measures the average overlap of predicted bounding boxes and ground truth bounding boxes, which is equivalent to Success (AUC). The success rate (SR) denotes the percentage of frames with overlaps greater than the threshold 0.5 or 0.75. The mean average overlap (mAO) and mean success rate (mSR) are two class-balanced metrics.

The proposed tracking algorithm is compared with 16 state-of-the-art tracking methods, respectively. The corresponding results are shown in Table 3-2. The proposed method TGLC (Ours) outperforms existing tracking approaches on GOT-10k benchmark. The $mSR_{0.75}$ witnesses the largest improvements among them.

Table 3-2 Tracking results on GOT-10k

Tracking methods	mAO ↑	$mSR_{0.5}$ ↑	$mSR_{0.75}$ ↑
TGLC (Ours)	**0.679**	**0.800**	**0.607**
SiamGAT [124]	0.627	0.743	0.488
CMEDFL [125]	0.599	0.682	0.448
Ocean [59]	0.611	0.721	0.473
SiamFC++ [75]	0.595	0.695	0.479

KYS [56]	0.636	0.751	0.515
DCFST [126]	0.638	0.753	0.498
PrDiMP [127]	0.634	0.738	0.543
D3S [76]	0.597	0.676	0.462
DiMP [72]	0.611	0.717	0.492
SiamRPN++ [73]	0.517	0.616	0.325
UTT [128]	0.672	0.763	0.605
[129]	0.425	0.467	0.307
GFS-DCF [130]	0.464	0.482	0.162
AD-LSTM [131]	0.343	0.352	0.109
DTDU [132]	0.375	0.416	0.133
SiamFC [23]	0.348	0.353	0.098

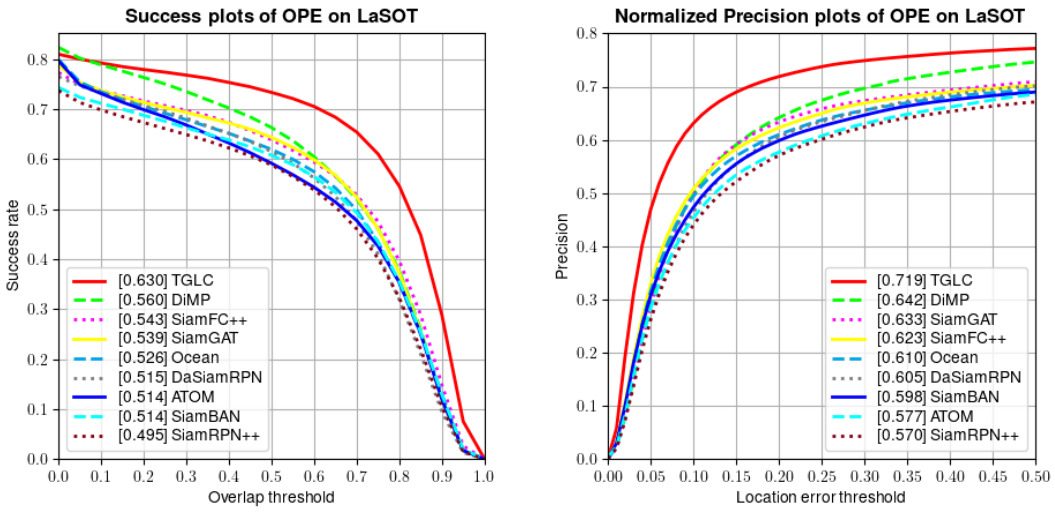
TrackingNet [120]. TrackingNet is currently the largest short-term video target tracking dataset, which consists of 30643 outdoor video sequences. It is divided into 12 sub-training sets and 1 test set. Three evaluation metrics, Precision, Normalized Precision and Success (AUC score) are used to evaluate the tracking performance. The proposed tracking algorithm is compared with 14 state-of-the-art tracking methods on TrackingNet dataset, and the results are illustrated in Table 3-3. It is obvious that the proposed tracking algorithm TGLC (Ours) surpasses all other tracking methods in terms of 3 evaluation metrics.

Table 3-3 Tracking performance comparison on TrackingNet

Tracking methods	AUC ↑	Norm Precision ↑	Precision ↑
TGLC (Ours)	**0.808**	**0.859**	**0.790**
UTT [128]	0.797	-	0.770
DualMN [133]	0.778	0.832	0.728
SiamAttn [134]	0.752	0.817	-
TGAN [135]	0.768	0.824	0.710
CMEDFL [125]	0.713	0.761	0.651
SiamFC++ [75]	0.754	0.800	0.705
KYS [56]	0.740	0.800	0.688

DCFST [126]	0.752	0.809	0.700
PrDiMP [127]	0.758	0.816	0.704
CGACD [136]	0.711	0.800	0.693
D3S [76]	0.728	0.768	0.664
DiMP [72]	0.740	0.801	0.687
SiamRPN++ [73]	0.733	0.800	0.694
SiamFC [23]	0.571	0.663	0.533

LaSOT [50]. LaSOT is a large-scale long-term tracking benchmark, which consists of two subsets. The main subset contains 1400 video sequences from 70 types of targets. The extended subset has 150 videos from 15 categories of objects. The dataset is divided into a training set and a test set. As shown in Figure 3-6, the proposed tracking method TGLC (Ours) achieves much better success rate and normalized precision than current tracking methods including DiMP [72], SiamFC++ [75], SiamGAT [124], Ocean [59], DaSiamRPN [25], ATOM [71], SiamBAN [137], and SiamRPN++ [73]. This indicates that the proposed approach retains better long-term tracking performance than listed tracking approaches.

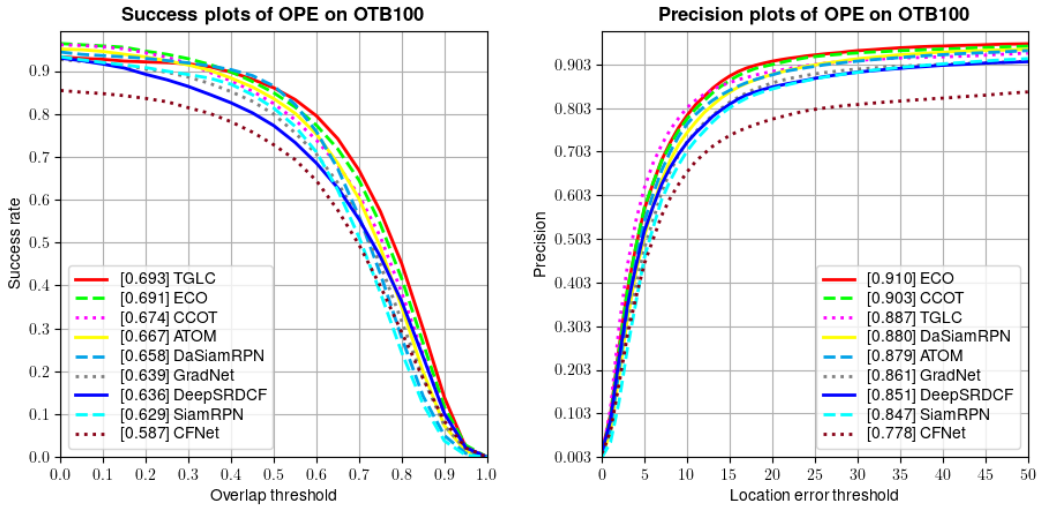


(a) Success plot (b) Normalized precision plot

Figure 3-6 Success plot and normalized precision plot on LaSOT dataset

OTB100 [32]. OTB100 consists of 100 challenging sequences for visual object tracking.

Success rate (AUC) and Precision are utilized to measure the tracking performance of tracking methods, as shown in Figure 3-7. The proposed method TGLC (Ours) outperforms the existing tracking methods in terms of success rate. Success rate is more important than precision as success rate takes into account scales and centers of predicted boxes simultaneously. The listed methods include ECO [70], CCOT [138], ATOM [71], DaSiamRPN [25], GradNet [139], DeepSRDCF [140], SiamRPN [24] and CFNet [141].

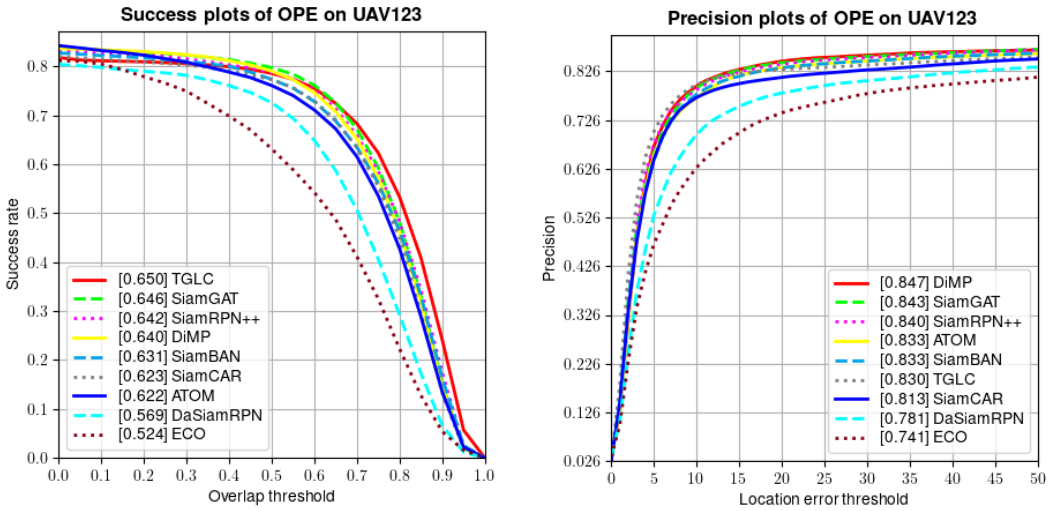


(a) Success plot (b) Precision plot

Figure 3-7 Success plot and precision plot on OTB100 dataset

UAV123 [142]. UAV123 is a dedicated tracking dataset with videos taken from unmanned aerial vehicles (UAVs), which is used for evaluating tracking performance by 123 videos captured from drone views. The comparison results are shown in Figure 3-8. The proposed tracking method TGLC (Ours) achieves a success rate of 0.650, which is competitive and better than listed superior tracking approaches, e.g., SiamGAT [124], SiamRPN++ [73], DiMP [72], SiamBAN [137], SiamCAR [143], ATOM [71], DaSiamRPN [25] and ECO [70]. The precision of TGLC is lower than superior methods, e.g., DiMP. However, precision is less convincing compared to success rate (AUC). Therefore, the more

comprehensive AUC is mainly utilized to assess the overall performance of trackers.



(a) Success plot (b) Precision plot

Figure 3-8 Success plot and precision plot on UAV123 dataset

Seven representative sequences of LaSOT dataset are selected for visualizing the tracking performance of the proposed tracking method TGLC (Ours) and six superior tracking approaches, (DaSiamRPN [25], GlobalTrack [39], SiamBAN [137], Ocean [59], SiamFC++ [75], ATOM [71]), as shown in Figure 3-9. Sequences bird-17 and person-5 face the problem of background clutter. Green and red boxes represent the ground-truth bounding box and the proposed tracking result, respectively. Boxes of other colors denote tracking results of the six existing tracking methods. It is evident that the proposed method has the largest overlap area with the ground truth when other algorithms demonstrate tracking drifts or failures. Partial occlusion occurs in sequence bus-2, and the proposed tracking method can roughly estimate the closest target area with the ground truth even the object is partially occluded. Furthermore, the proposed algorithm displays the maximum similarity with the ground-truth results as well when dealing with other challenges, e.g., illumination variation (guitar-3), out-of-view (horse-1), fast motion (yoyo-7), scale change

and deformation of the object (kangaroo-11). Visualization of the tracking results further confirms the effectiveness of our proposed tracking algorithm TGLC.

Figure 3-9 Visualization of tracking results on LaSOT

3.3 Ablation experiments

A series of ablation experiments are performed to evaluate the effect of different modules on tracking performance. The corresponding results are shown in Table 3-4 and Table 3-5.

Backbone ablation. Backbone is used for feature extraction of template and search region, and backbone is a vital module for object tracking. Swin-Tiny [144] and ResNet [114] are respectively utilized to build the backbone. Under the same condition (Backbone + complete feature fusion module), ResNet50 achieves an AUC of 0.63, outperforming Swin-Tiny (0.595) by 5.9% and ResNet101 (0.623) by 1.1%. Additionally, ResNet50 is the most efficient among the three backbone options. ResNet50 is utilized as the final backbone since it achieves the best tracking performance in terms of AUC and tracking speed.

Feature fusion ablation. SE block and CCB block of the proposed feature fusion module are developed to capture the channel information and local feature information of feature maps. In order to explore the effectiveness of the SE block and CCB block, three ablation experiments with or without these two blocks are conducted. An AUC of 0.602 is obtained when removing the SE block and CCB block, which means that pure global information for feature fusion is not able to achieve optimal tracking performance. Gains of 3.3% and 1.3% are achieved when CCB block and SE block are respectively added to the feature fusion module, which proves that global-local information fusion and channel information can substantially improve the tracking performance.

Table 3-4 Ablation study on LaSOT dataset

Backbones	SE block	CCB block	AUC	Norm precision
Swin-Tiny	✓	✓	0.595	0.677
ResNet101	✓	✓	0.623	0.709

ResNet50	×	✓	0.622	0.707
ResNet50	×	×	0.602	0.695
ResNet50	✓	✓	**0.630**	**0.719**

Adaptability of CCB block. The proposed CCB block is incorporated with original TransT [29] for further exploring its adaptability and superiority. In order to achieve this, the proposed GLSA and GLCA are separately employed to replace the ECA and CFA modules in TransT for feature fusion. Other components and training details remain the same as the original TransT. The AUC scores of three benchmarks are demonstrated in Table 3-5.

TransT_N4 and TransT_N2 denote respectively original TransT with 4 and 2 feature fusion layers. TransT_CCB_N4 and TransT_CCB_N2 represent the modified models with 4 and 2 feature fusion layers, separately. It can be seen from the table that the modified models clearly improve the tracking performance on both levels (N2 or N4). Surprisingly, the smaller model TransT_CCB_N2 performs almost on par with TransT_N4 on LaSOT and TrackingNet benchmarks. It further suggests that integration of global local representations and channel information facilitates the feature fusion of two branches. The proposed CCB block can also be integrated into other pure Transformer-based tracking methods to introduce local representations and channel features.

Table 3-5 Performance comparison between TransT_CCB and TransT at different levels

Methods	LaSOT	TrackingNet	OTB100
TransT_N4 [29]	0.649	0.814	0.694
TransT_CCB_N4 (Ours)	**0.654**	0.820	**0.707**
TransT_N2 [29]	0.642	0.809	0.681
TransT_CCB_N2 (Ours)	0.648	**0.821**	0.684

3.4 Summary

The proposed short-term tracking method TGLC is elaborated in chapter three. First, the main components of TGLC including the feature fusion module and the prediction head are introduced thoroughly. Second, TGLC is compared with the current state-of-the-art methods on multiple tracking benchmarks. The ablation experiments are also conducted to validate the performance of the core modules in TGLC.

Chapter 4 Masked image modeling enhanced long-term tracking

4.1 Methodology

4.1.1 Overall tracking framework of MIMTracking

A novel tracking method dubbed MIMTracking is proposed specifically for long-term object tracking in this chapter. The presented MIMTracking incorporates masked image modeling into tracking and learns discriminative representations for more accurate tracking. The proposed MIMTracking framework is illustrated in Figure 4-1. Template and search region are regions of interest cropped from the 1st and current image, respectively. Patch embedding module is adopted to transform 2D images into 1D patch (token) embeddings. Embedding of learnable sine-cosine position information [99] enables encoding of patch embeddings of template and search region. This is quite essential for masked image modeling (MIM) since neither masked tokens nor visible tokens can restore their positions in the original image once they are shuffled without position encodings. MIM is mainly utilized in the search region by randomly occluding a certain proportion of tokens, which is achieved by random masking [45]. After random masking, the unoccluded search tokens and template tokens are concatenated and then fed into the encoder of Vision Transformer [82]. The masked search tokens are reserved for supplementing the representations of the search region in subsequent steps.

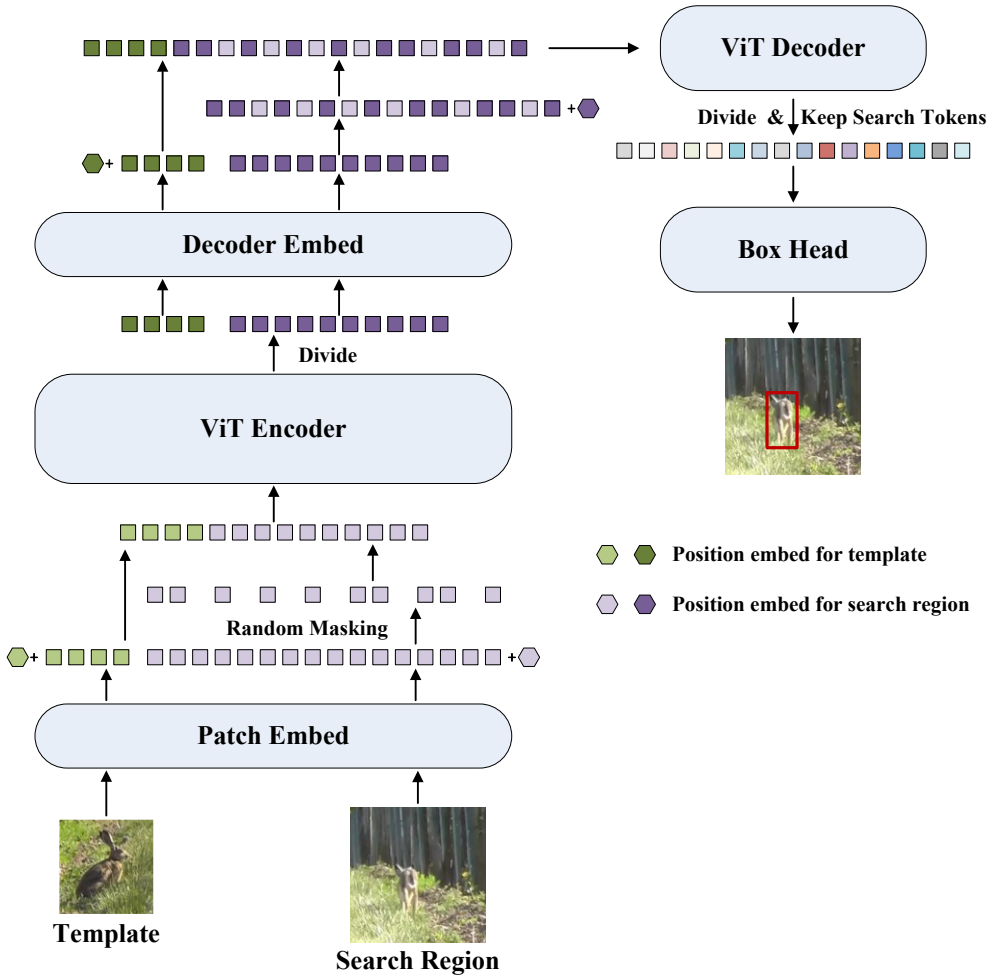


Figure 4-1 The pipeline of the proposed MIMTracking

The encoded latent representations are further partitioned into template representations and search representations. A linear decoder embedding block is employed to map the template and search representations to decoder latent space with a dimension of 512. The above-mentioned masked search tokens are subsequently transformed into decoder latent space. Then the transformed search representations and masked search tokens are concatenated and rearranged according to the original token order. Position encodings are again added to template representations and rearranged search tokens before concatenating them. A compact decoder with blocks taken from Vision Transformer [82] is employed to model holistic latent representations for search region and to further facilitate the information

incorporation of template and search region. Therefore, more discriminative features are generated for tracking prediction head. The bounding box of the target is eventually obtained by the box prediction head.

Patch embed. Patch embedding block is exploited to generate the patch (token) embeddings of the template $t \in \mathbb{R}^{H_t \times W_t \times 3}$ and the search region $s \in \mathbb{R}^{H_s \times W_s \times 3}$. As depicted in Eq. (4-1) and Eq. (4-2), this shared patch embedding layer first utilizes a 2D convolution layer to transform the template and search region into 2D feature maps with a channel size $q = 768$. K is the convolution kernel. The kernel size and the stride of this convolution layer are both 16. Then the feature maps are flattened in the spatial dimension. Therefore, non-overlapping 1D token embeddings $\bar{H}_t$ and $\bar{H}_s$ of two inputs are produced for subsequent encoder module. The number of tokens for template and search region are respectively $\frac{H_t}{16} \times \frac{W_t}{16}$ and $\frac{H_s}{16} \times \frac{W_s}{16}$.

$$h_s[q,k,l] = conv(s) = \sum_{p} \sum_{i=0}^{15} \sum_{j=0}^{15} K[p,q,i,j] \cdot s[p,k+i-1,l+j-1] \tag{4-1}$$

$$h_t[q,k,l] = conv(t) = \sum_{p} \sum_{i=0}^{15} \sum_{j=0}^{15} K[p,q,i,j] \cdot t[p,k+i-1,l+j-1] \tag{4-2}$$

$$\bar{H}_s^{(\frac{H_s W_s}{16^2}, 768)} = flatten\left[conv(s)\right] \tag{4-3}$$

$$\bar{H}_t^{(\frac{H_t W_t}{16^2}, 768)} = flatten\left[conv(t)\right] \tag{4-4}$$

Random masking. Inspired by [45], random masking is performed after position encoding of template and search region. Search tokens are randomly masked according to the uniform distribution. Remaining tokens are fed into the encoder. A sampling ratio is utilized to determine how many tokens are kept. Masked tokens and their indices are preserved for

restoring their original positions on the search token sequences. Random masking strategy alleviates the computational burden of the tracking model by only feeding partial input tokens into the heavy ViT encoder. It also stimulates the potential of the tracking model to pay attention to more detailed representations.

ViT encoder and decoder. After random masking, the remaining search tokens and template tokens are concatenated and input into the ViT encoder for feature representation generation and interaction. Encoded representations and masked token embeddings are restored in sequence and fed into ViT decoder for further information exchange. In order to maximize the potential of MIM on tracking and to benefit from self-supervised pretraining that has recently been proved to be very effective, the architecture of encoder and decoder follows the asymmetric design of MAE [45].

The encoder and decoder consist of M and N ViT blocks, respectively. As described in Eq. (4-5) and Eq. (4-6), each ViT block is composed of two residual blocks with multi-head self-attention (MSA) and multilayer perceptron (MLP) as the main part for each residual block, separately. LayerNorm [145] is employed at the beginning of each residual block. M and N are respectively set to 12 and 4 in this case. The embedding dimensions of the encoder and decoder are 768 and 512, separately.

$$\tilde{x} = MSA(LN(x)) + x \tag{4-5}$$

$$y = MLP(LN(\tilde{x})) + \tilde{x} \tag{4-6}$$

Multi-head self-attention generally performs multiple self-attention operations simultaneously and combines multiple attention outputs for acquiring diverse representations. Self-attention is elaborated in Eq. (4-7) for better understanding.

$$SelfAtt(q,k,v) = \text{softmax}(qk^T(d)^{(-\frac{1}{2})})v \tag{4-7}$$

where $q = xL_q \in \mathbb{R}^{n\times d}$, $k = xL_k \in \mathbb{R}^{n\times d}$ and $v = xL_v \in \mathbb{R}^{n\times d}$ denote the query, key and value for the self-attention operation. n and d respectively represent sequence length and dimension. x and three matrices L_q, L_k and L_v denote the input of self-attention and three linear transformation matrices, separately.

Box head. The search and template representations are fully integrated and interacted through the encoder and decoder. Then the search representations are detached from the output concatenated representations and act as the input for a bounding box head for tracking result prediction. The box head is inspired by the prediction head of OSTrack [44]. Three parallel branches with Z($Z = 4$) standard Convolution-BatchNorm-ReLU blocks and one convolution mapping layer for every branch are exploited to estimate bounding box center (x', y'), center offset (x_{offset}, y_{offset}) and bounding box size (w, h), respectively. Specifically, the location of the maximum value of the first predicted feature map serves as the bounding box center. The values corresponding to this max value location in the output maps of the other two branches are respectively the center offset and bounding box size. The center offset is added to the bounding box center for alleviating the deviation of the center position prediction due to various reasons, which is illustrated in Eq. (4-8).

$$(c_x, c_y, w, h) = (x' + x_{offset}, y' + y_{offset}, w, h) \tag{4-8}$$

where (c_x, c_y) is the compensated center position coordinates of the predicted bounding box. Therefore, the final bounding box of the target can be determined by Eq. (4-8). A simplified flowchart of the proposed MIMTracking is shown in Figure 4-2.

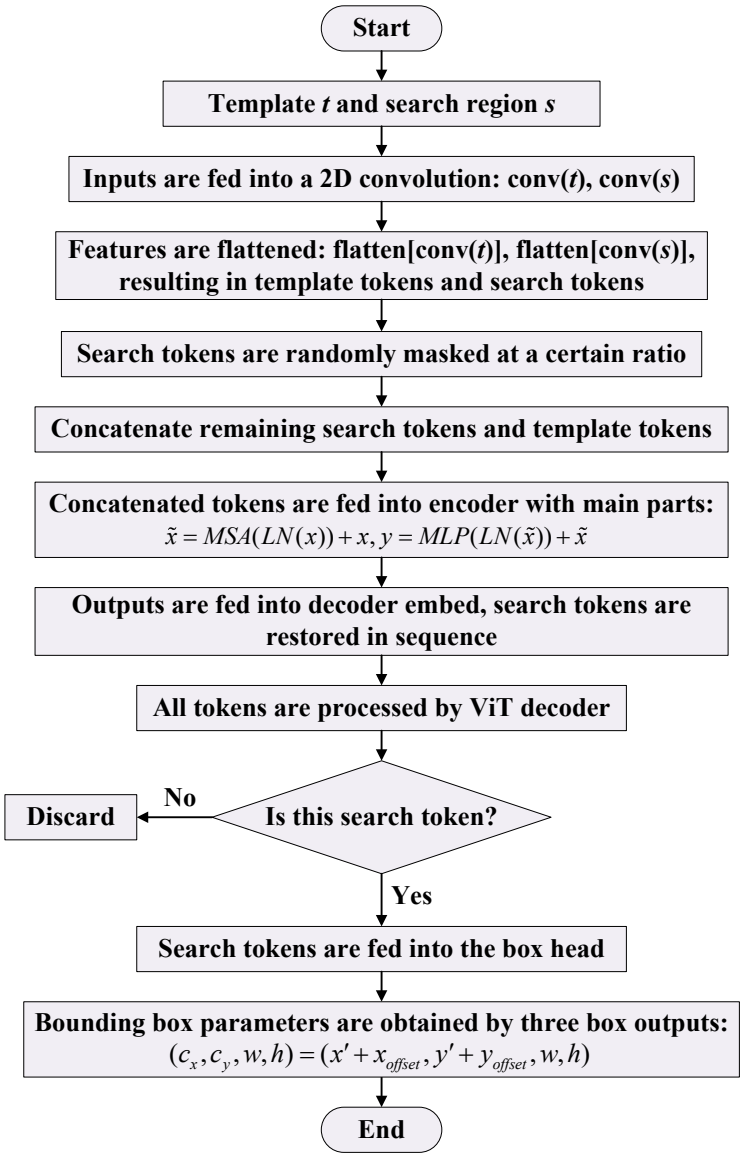


Figure 4-2 Flowchart of the proposed MIMTracking

4.1.2 Objective function

The overall objective function of the training process consists of a focal loss variant [146] for classification of foreground and background, a generalized intersection over union (GIoU) loss [118] and a ℓ_1 loss for precise bounding box adjustment. This weighted objective function is demonstrated in Eq. (4-9).

$$Obj_{overall} = \alpha Obj_{GIoU}(y_{bbox}, \hat{y}_{bbox}) + \beta Obj_{\ell_1}(y_{bbox}, \hat{y}_{bbox}) + \gamma Obj_{focal}(y_{cls}, \hat{y}_{cls}) \quad (4\text{-}9)$$

where y_{bbox} and $\hat{y}_{bbox}$ are respectively the predicted and ground-truth bounding box coordinates. y_{cls} and $\hat{y}_{cls}$ denote predicted classification feature map and ground-truth classification label, respectively. $\alpha = 2$, $\beta = 5$, and $\gamma = 1$ represent the weights of three sub-objective functions, separately.

4.2 Comparison experiments

4.2.1 Implementation details

Tracking framework. Two tracking variants are developed in this work, which are respectively MIMTracking and MIMTracking-L. The structure of MIMTracking is elaborated in Section 4.1.1. Similar to MIMTracking, MIMTracking-L adopts Vanilla ViT-Large as the backbone. 24 and 4 ViT blocks are utilized for building the encoder and decoder of MIMTracking-L, separately. The remaining backbone settings are the same as ViT-L. The template size and search region size for both variants are 192×192 and 384×384, respectively. The MAE [45] pretrained ViT-B and ViT-L weights are separately utilized to initialize the encoder and decoder parameters of the proposed MIMTracking and MIMTracking-L. The box head is initialized by Xavier uniform initialization [147] and is trained from scratch for bounding box estimation.

Training. The proposed MIMTracking is implemented by reasonable version matching of resources (Python 3.8.13, Pytorch 1.10.0, TorchVision 0.11.0) on Ubuntu 20.04. The training hardware contains a Nvidia RTX A6000 GPU and an Intel Core i9 CPU. The proposed MIMTracking is fine-tuned on the training sets of 4 publicly available datasets, GOT-10k [119], TrackingNet [120], LaSOT [50] and COCO [121]. In order to follow the protocol of GOT-10k evaluation, the tracking results of GOT-10k are generated by the tracking model trained only on GOT-10k dataset. Brightness jittering and horizontal flip

are employed for image augmentation. The presented MIMTracking is trained for 600 epochs in total with 60000 samples for one epoch by utilizing AdamW optimizer [123]. The batch size is only 32 as a single GPU is exploited during training. The initial learning rates of the backbone (including the encoder and decoder) and the prediction head are separately 10^{-5} and 10^{-4}, which start to decay after 300 epochs by a factor of 10. MIMTracking-L follows the training recipe of MIMTracking, except using a smaller batch size (16) due to computational resource limits.

Inference. The inference process is unidirectional without any parameter update. A template (cropped from first frame) and a search region (cropped from current frame) with the same sizes as the training process serve as the inputs of the inference process. The parameters of the last epoch are utilized for tracking result estimation during inference process. The remaining inference settings are the same as OSTrack [44]. Detailed implementation is summarized in Table 4-1.

Table 4-1 Implementation details

Settings	Values
Backbone of MIMTracking	Vanilla ViT-Base
Backbone of MIMTracking-L	Vanilla ViT-Large
Center jitter factor	4.5
Scale jitter factor	0.5
GPU	One Nvidia RTX A6000
CPU	Intel Core i9
The operating system	Ubuntu 20.04
Python	3.8.13
Pytorch	1.10.0
TorchVision	0.11.0
Template size	192×192
Search region size	384×384
Epoch number	600

Samples per epoch	60000
Batch size of MIMTracking	32
Batch size of MIMTracking-L	16
Optimizer	AdamW
Learning rate of backbone	10^{-5}
Learning rate of other parameters	10^{-4}
Learning rate decay epoch	300
Weight decay factor	10^{-4}

4.2.2 Comparison with top methods

The presented trackers MIMTracking and MIMTracking-L are compared with current state-of-the-art long-term and short-term trackers on five widely used tracking datasets. The tracking results are shown in Table 4-2, Table 4-3 and Table 4-4. The results of MIMTracking and MIMTracking-L are obtained under a training sampling ratio of 0.5 and inference sampling ratio of 1, unless otherwise specified.

Table 4-2 Comparisons on LaSOT and LaSOText benchmarks

Trackers	LaSOT			$\text{LaSOT}_{\text{ext}}$		
	AUC	P_{norm}	P	AUC	P_{norm}	P
SiamRPN++ [73]	49.6	56.9	49.1	34.0	41.6	39.6
DiMP [72]	56.9	65.0	56.7	39.2	47.6	45.1
LTMU [40]	57.2	-	57.2	41.4	49.9	47.3
SiamFC++ [75]	54.4	62.3	54.7	-	-	-
Ocean [59]	56.0	65.1	56.6	-	-	-
PrDiMP [127]	59.8	68.8	60.8	-	-	-
TrDiMP [148]	63.9	-	61.4	-	-	-
AutoMatch [149]	58.3	-	59.9	-	-	-
TransT [29]	64.9	73.8	69.0	-	-	-
KeepTrack [57]	67.1	77.2	70.2	48.2	-	-
SwinTrack [80]	69.6	78.6	74.1	47.6	58.2	54.1
STARK [30]	67.1	77.0	-	-	-	-
AiATrack [81]	69.0	79.4	73.8	47.7	55.6	55.4

SwinV2-B [48]	70.0	-	-	-	-	-
SwinV2-L [48]	70.7	-	-	-	-	-
MAT [49]	67.8	77.3	-	-	-	-
GdaTFT [150]	64.3	68.0	68.7	-	-	-
TATrack [62]	69.4	78.2	74.1	-	-	-
SimTrack-B [43]	69.3	78.5	-	-	-	-
SimTrack-L [43]	70.5	79.7	-	-	-	-
OSTrack384 [44]	71.1	81.1	77.6	50.5	61.3	57.6
MixFormer [42]	69.2	78.7	74.7	-	-	-
MixFormer-L [42]	70.1	79.9	76.3	-	-	-
CTTrack [47]	67.8	77.8	74.0	-	-	-
MIMTracking	72.0	81.4	78.4	51.5	62.1	59.0
MIMTracking-L	72.9	82.5	79.6	53.4	64.0	60.8

Notes:

1. The top two results are illustrated in red and blue fonts.

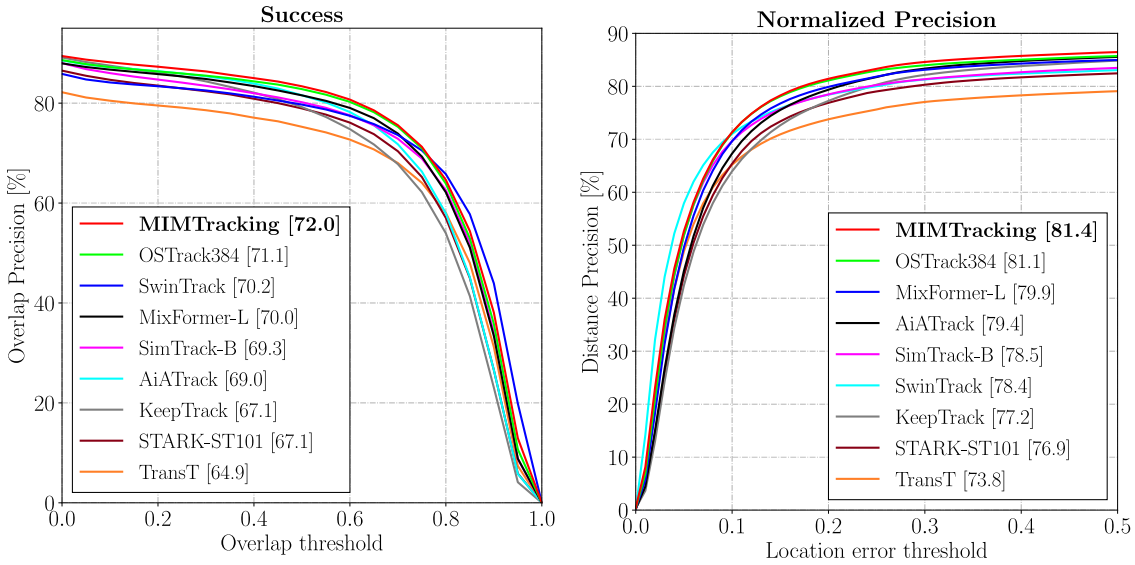
2. MIMTracking and MIMTracking-L denote respectively the proposed tracking methods with ViT-B and ViT-L backbones.

3. It suggests that the proposed MIMTracking outperforms almost all the state-of-the-art trackers on these benchmarks.

4. All evaluation metric scores (AUC, P_{norm}, P, AO, $SR_{0.5}$ and $SR_{0.75}$) from this point onward are in the form of percentages (%), unless otherwise specified.

LaSOT [50]. LaSOT provides 1120 and 280 sequences respectively used for training and testing tracking methods. It is developed especially for long-term tracking. As shown in Table 4-2, MIMTracking achieves an AUC of 72.0%, which outperforms all state-of-the-art trackers. Surprisingly, MIMTracking with medium-sized ViT-B is capable of surpassing large-sized MixFormer-L and SimTrack-L. It is also worth noting that both SimTrack and OSTrack exploit the encoder of Vanilla ViT as the backbone for tracking. The proposed method MIMTracking employs both encoder and decoder to realize masked

image modeling. This suggests that masked image modeling is able to stimulate the tracking potential of Vanilla ViT. The success and normalized precision plots are demonstrated in Figure 4-3 for visualizing tracking performance more intuitively, which yield the same conclusion. Additionally, MIMTracking-L denotes the proposed tracking method with the ViT-L model. It further surpasses MIMTracking in terms of all three metrics.



(a) Success plot (b) Normalized precision plot

Figure 4-3 Success and normalized precision plots on LaSOT dataset

LaSOT$_{ext}$ [151]. LaSOT$_{ext}$ is a newly introduced large-scale long-term tracking benchmark for complementing the original LaSOT dataset. It contains 150 accurately annotated long sequences from 15 new categories collecting beyond ImageNet [122]. The proposed MIMTracking is assessed and compared with state-of-the-art tracking methods on LaSOT$_{ext}$ benchmark. The results are shown in Table 4-2. MIMTracking improves the AUC to 51.5%, outperforming the previous state-of-the-art OSTrack by 2%. The normalized precision and precision increase by 1.3% and 2.4%, respectively. Furthermore,

our large model MIMTracking-L demonstrates promising scalability by outperforming the base model MIMTracking by 3.7% in terms of AUC.

GOT-10k [119]. GOT-10k is composed of a training set with 10000 videos and a test set with 180 video sequences. Non-overlapping target categories between training and test sets allow for evaluating the adaptability of trackers to targets of unknown classes. It can be seen from Table 4-3 that MIMTracking performs on par with the top algorithms OSTrack and SwinV2. 1.7% gain is achieved in $SR_{0.75}$ by MIMTracking compared with OSTrack. MIMTracking-L further boosts the tracking performance in terms of three metrics.

TrackingNet [120]. TrackingNet utilizes 30132 and 511 video sequences for training and testing, separately. Similar to GOT-10k, the tracking results of TrackingNet should be uploaded to the evaluation server for performance assessment. As demonstrated in Table 4-3, MIMTracking exceeds MixFormer-L and OSTrack in terms of three evaluation metrics. A new state-of-the-art AUC (85.0%) is set by our large model MIMTracking-L on TrackingNet benchmark.

Table 4-3 Comparisons on TrackingNet and GOT-10k benchmarks

Trackers	GOT-10k			TrackingNet		
	AO	$SR_{0.5}$	$SR_{0.75}$	AUC	P_{norm}	P
SiamRPN++ [73]	51.7	61.6	32.5	73.3	80.0	69.4
DiMP [72]	61.1	71.7	49.2	74.0	80.1	68.7
SiamFC++ [75]	59.5	69.5	47.9	75.4	80.0	70.5
Ocean [59]	61.1	72.1	47.3	-	-	-
PrDiMP [127]	63.4	73.8	54.3	75.8	81.6	70.4
TrDiMP [148]	67.1	77.7	58.3	78.4	83.3	73.1
AutoMatch [149]	65.2	76.6	54.3	76.0	-	72.6
TransT [29]	67.1	76.8	60.9	81.4	86.7	80.3
SwinTrack [80]	69.4	78.0	64.3	82.5	87.0	80.4
STARK [30]	68.8	78.1	64.1	82.0	86.9	-

AiATrack [81]	69.6	80.0	63.2	82.7	87.8	80.4
SwinV2-B [48]	70.8	-	-	82.0	-	-
SwinV2-L [48]	72.9	-	-	82.5	-	-
MAT [49]	67.7	78.4	-	81.9	86.8	-
GdaTFT [150]	65.0	77.8	53.7	77.8	83.5	75.4
TATrack [62]	73.0	83.3	68.5	83.5	88.3	81.8
SimTrack-B [43]	68.6	78.9	62.4	82.3	86.5	-
SimTrack-L [43]	69.8	78.8	66.0	83.4	87.4	-
OSTrack384 [44]	73.7	83.2	70.8	83.9	88.5	83.2
MixFormer [42]	70.7	80.0	67.8	83.1	88.1	81.6
MixFormer-L [42]	-	-	-	83.9	88.9	83.1
CTTrack [47]	71.3	80.7	70.3	82.5	87.1	80.3
MIMTracking	73.7	83.1	72.0	84.5	89.0	84.0
MIMTracking-L	74.9	84.0	73.5	85.0	89.0	84.8

Notes:

1. The top two results are illustrated in red and blue fonts.

2. MIMTracking and MIMTracking-L denote respectively the proposed tracking methods with ViT-B and ViT-L backbones.

TNL2K [152]. TNL2K is a recently proposed tracking dataset that equips tracking sequences with language descriptions. 1300 and 700 videos are respectively utilized for training and evaluation purposes. The results of several state-of-the-art trackers are illustrated in Table 4-4. The results imply that our base model MIMTracking surpasses all the state-of-the-art methods in terms of AUC and Precision. The large model MIMTracking-L further increases the AUC to 57.3%.

Table 4-4 Results on TNL2K benchmark

Methods	AUC	P
DiMP [72]	44.7	43.4
Ocean [59]	38.4	37.7
AutoMatch [149]	47.2	43.5

TransT [29]	50.7	51.7
SwinTrack [80]	55.9	57.1
SimTrack-L [43]	55.6	55.7
MAT [49]	51.3	-
OSTrack384 [44]	55.9	-
MIMTracking	56.3	58.3
MIMTracking-L	57.3	59.7

Note: The best two results are denoted by red and blue fonts.

4.3 Ablation experiments

The proposed MIMTracking is ablated regarding the key design elements. The impact of masked image modeling (MIM) on the tracking model is explored qualitatively and quantitatively.

Sampling ratio. Sampling ratio is designed in the random masking module to control the proportion of retained search tokens. The training sampling ratio and inference sampling ratio denote respectively the token retention ratio during training and inference processes. Extensive experiments are conducted on LaSOT benchmark regarding different training and inference sampling ratios. The results are depicted in Table 4-5. Four training sampling ratios (1, 0.75, 0.5, 0.25) are utilized to train four tracking models, respectively. Each model is tested under a range of inference sampling ratios not lower than the corresponding training sampling ratio.

From Table 4-5, tracking results are improved as the inference sampling ratio grows under the training sampling ratio 0.75 or 0.5. This is reasonable as more search tokens are encoded when the inference sampling ratio increases. Surprisingly, a lower training sampling ratio of 0.5 gains slightly better results than a higher ratio 0.75 under the same inference sampling ratio, e.g., 0.75. However, the results drop considerably when the training sampling ratio declines to 0.25. This indicates that moderately low training

sampling ratios, e.g., 0.5, can greatly reduce feature redundancy and improve the localization ability of the tracking method. However, exceptionally low training sampling ratios will corrupt the key feature information, thereby reducing the target localization capability. It suggests that moderate training sampling ratio 0.5 works best for our case. MIM is further removed from the tracking model to validate the effectiveness of masked image modeling for tracking. To this end, both the training sampling ratio and inference sampling ratio are set to 1. The results in Table 4-5 suggest that removing MIM decreases the tracking performance. The reason is that masked image modeling (MIM) is capable of urging the tracking network to learn more useful and more discriminative representations. Removing MIM leads to redundant input information, which is not conducive to improving the global reasoning ability of the tracking algorithm.

Table 4-5 Effect of training and inference sampling ratios on tracking performance

Training sampling ratio	Inference sampling ratio	AUC	P_{norm}	P
1	1	71.5	81.1	77.8
0.75	0.75	71.3	81.3	77.5
	1	71.8	81.4	78.3
0.5	0.5	70.7	80.9	76.6
	0.75	71.5	81.3	77.8
	1	72.0	81.4	78.4
0.25	0.25	68.8	79.2	73.8
	0.5	70.0	80.1	75.5
	0.75	70.2	79.9	75.7
	1	70.0	79.5	75.7

Notes:

1. Results are obtained from LaSOT dataset.

2. The best results are shown in red.

Search region size. As shown in Table 4-6, the impact of search region size is explored on three benchmarks in terms of AUC (%). The first two rows denote separately the proposed MIMTracking with search sizes of 384 and 256. AUC increases by 3.6% and 3.1% respectively on LaSOT and TNL2K datasets when search region size increases from 256 to 384. This is because that a larger search size increases the probability that the target exists. Table 4-6 also indicates that MIMTracking outperforms the previous state-of-the-art OSTrack on all three datasets for either search region size.

Table 4-6 Impact of search region size and decoder depth

Model	Search size	Decoder depth	LaSOT	TNL2K	TrackingNet
MIMTracking	384	4	72.0	56.3	84.5
MIMTracking-256	256	4	69.5	54.6	84.1
MIMTracking	384	1	71.8	56.1	84.5
OSTrack256 [44]	256	0	69.1	54.3	83.1
OSTrack384 [44]	384	0	71.1	55.9	83.9

Notes:

1. AUC scores (%) are evaluated on LaSOT, TNL2K, and TrackingNet benchmarks.

2. Top results are highlighted in red.

Decoder depth. Decoder depth is ablated as well on three benchmarks. The results are shown in the first and third rows of Table 4-6. There is a slight AUC improvement (+0.2%) on LaSOT and TNL2K datasets with decoder depth changing from 1 to 4. Compared to OSTrack with only encoder, MIMTracking builds the encoder and decoder required by masked image modeling (MIM). The superior results of MIMTracking on three benchmarks prove that MIM can improve tracking performance.

Elaborate analysis of MIMTracking. Table 4-7 demonstrates the computational cost (MACs), model parameters (Params), running speed (FPS) and AUC (%) under various

combinations of model variants, inference sampling ratios and decoder depths. Increase in search region size brings around 3.6% gain in AUC with a drop of 8.3% in running speed. Reducing the inference sampling ratio from 1 to 0.5 results in a notable drop in computational cost (33.3%), which leads to an AUC decline of 1.8% as well. Additionally, decoder depth reduction also has a substantial impact on reducing computational cost and model parameters. Prominently, the proposed MIMTracking with a search size of 384, a decoder depth of 4 and an inference sampling ratio of 1, achieves a superior AUC of 72.0% on LaSOT while still running in real time (83.0 FPS). The large model MIMTracking-L improves the AUC to 72.9%, while the inference speed drops dramatically. Therefore, MIMTracking achieves a better trade-off between tracking accuracy and speed.

Table 4-7 Elaborate analysis of MIMTracking

Model variants	Inference sampling ratio	Decoder depth	MACs (G)	Params (M)	FPS	AUC
MIMTracking-384	0.5	4	49.3	103.7	96.2	70.7
	0.75	4	61.6	103.7	93.7	71.5
	1	4	73.9	103.7	83.0	72.0
	0.75	1	54.8	94.3	115.5	71.1
	1	1	67.1	94.3	96.5	71.8
MIMTracking-256	0.5	4	21.9	103.7	94.5	-
	0.75	4	27.4	103.7	92.2	69.3
	1	4	32.8	103.7	90.5	69.5
	0.75	1	24.3	94.3	113.8	-
	1	1	29.8	94.3	111.4	-
MIMTracking-L-384	0.5	4	143.3	321.4	34.9	-
	0.75	4	186.9	321.4	25.9	-
	1	4	230.4	321.4	20.8	72.9

Notes:

1. The table shows the impact of different model variants, inference sampling ratios and decoder depths on computational cost (MACs), model parameters (Params), running speed (FPS) and AUC (%) on LaSOT benchmark.

2. The speeds are measured by a single RTX A6000 GPU.

Visualization of attention weights. In this section, the encoder and decoder attention weights are visualized for exploring the emphases of different layers of MIMTracking. As depicted in Figure 4-4, attention weights of three encoder layers (Enc_1, Enc_6, Enc_12) and three decoder layers (Dec_1, Dec_3, Dec_4) are visualized for the search region during the inference process.

It is obvious that the encoder layers do not focus on the target region, while the decoder layers gradually highlight the target region. This is not a coincidence as the encoder is trained to process partial discrete search tokens. There is a possibility that the target-related tokens are masked out. Therefore, it is not a wise choice to focus on localizing the precise target region in the encoder with incomplete search region information. Alternatively, the encoder is mainly responsible for establishing connections between the remaining search tokens and the template tokens, thus learning discriminative latent representations. The representations and missing search information serve as the input of the decoder, thereby making the decoder the most suitable one for target region localization. This explains how masked image modeling (MIM) works internally.

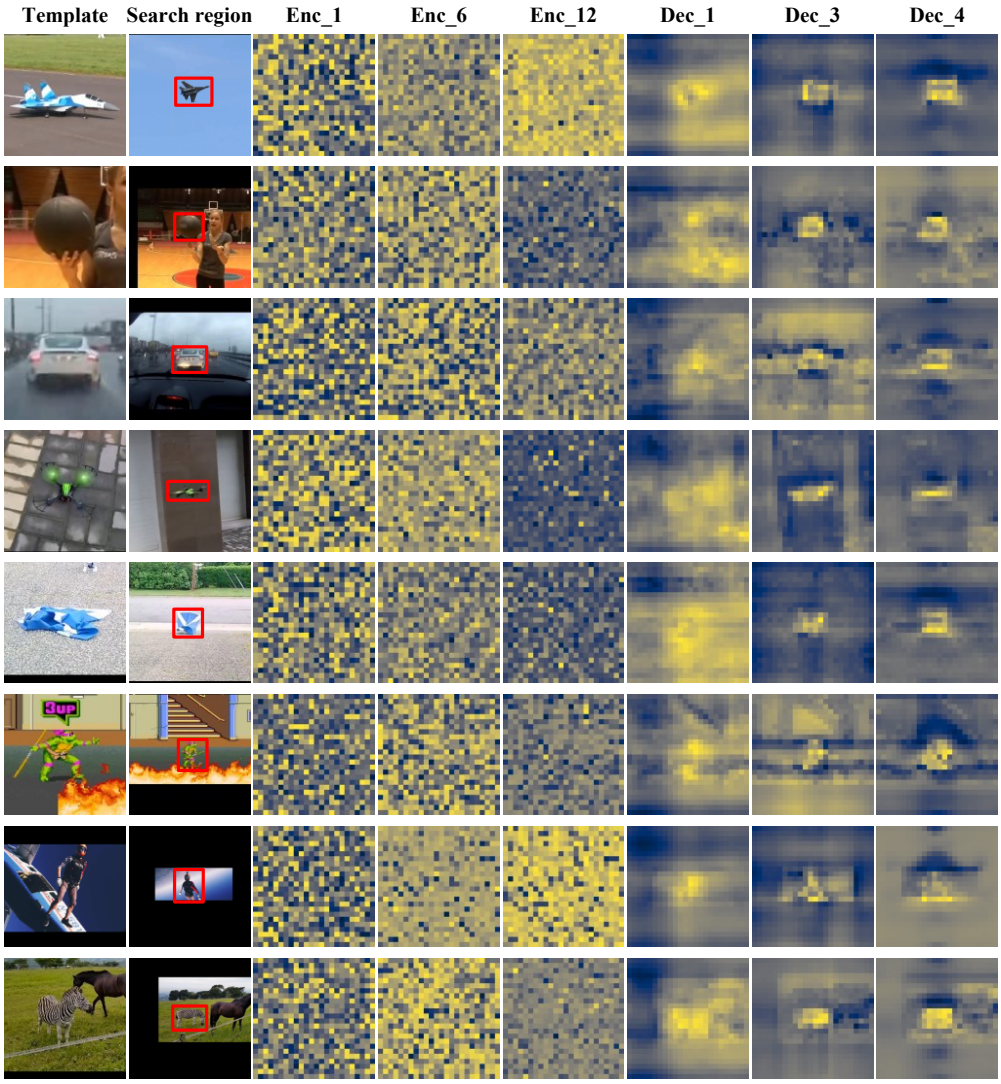


Figure 4-4 Visualization of attention weights of encoder and decoder on LaSOT

Visualization under different sampling ratios. Figure 4-5 demonstrates the attention maps of the decoder of MIMTracking under several inference sampling ratios on LaSOT benchmark. It can be seen from the figure that the pixels of background regions are more uniform and smoother when the sampling ratio is 1.0, thus causing less interference to the prediction of target area. However, when the sampling ratio is lower than 1.0, the prediction of both background and object regions becomes less accurate. This further validates the effect of the sampling ratio in the ablation study.

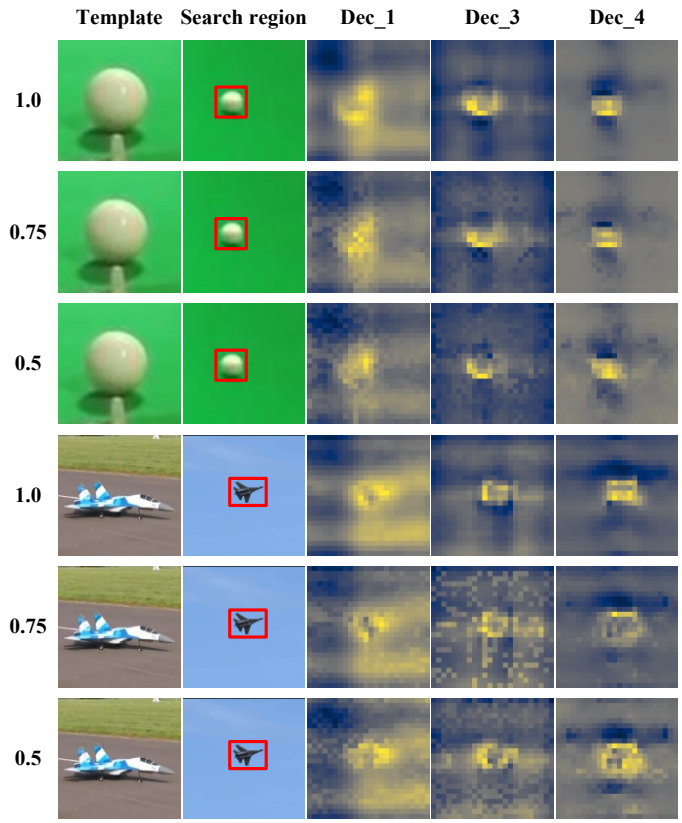


Figure 4-5 Visualization of decoder attentions under several inference sampling ratios

Comprehensive attribute analysis. Figure 4-6 describes the comprehensive attribute analysis on LaSOT dataset. This analysis compares all the state-of-the-art tracking approaches in terms of 14 attributes and combines the results into one figure. The 14 attributes consist of camera motion, viewpoint change, rotation, deformation, full occlusion, partial occlusion, illumination variation, out-of-view, scale variation, background clutter, motion blur, aspect ratio change, low resolution and fast motion. The explanation of the attributes is described in Table 4-8. It can be seen that fast motion, full occlusion and background clutter are three most challenging attributes since they demonstrate the lowest AUC scores. Our presented MIMTracking achieves the most superior comprehensive performance on all attributes.

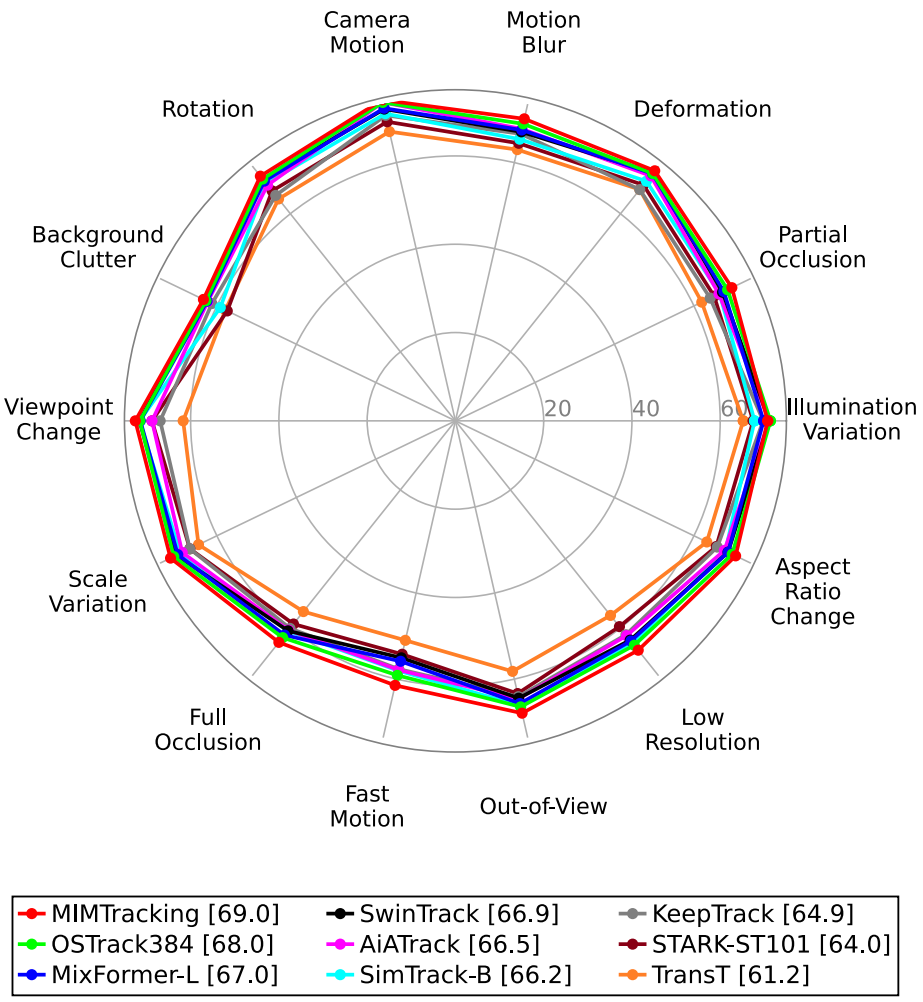


Figure 4-6 Comprehensive attribute analysis

Table 4-8 Attribute interpretation on LaSOT dataset

Attributes	Interpretation
Aspect ratio change	The aspect ratio of the bounding box exceeds the rage [0.5, 2]
Background clutter	The content of the target is similar to that of the background
Camera motion	The camera starts moving suddenly
Deformation	The object deforms in the tracking process
Fast motion	The movement of the target is beyond the range of the bounding box
Full occlusion	The object is completely occluded by the background
Illumination variation	The illumination condition varies in target area
Low resolution	The bounding box size in at least one frame is less than 1000 pixels
Motion blur	The target is blurred due to the movement of the camera or the object

Out-of-view	The object is entirely outside the frame
Partial occlusion	The object is partially occluded by the background
Rotation	The object rotates in different frames
Scale variation	The bounding box ratio is beyond the range of [0.5, 2]
Viewpoint change	The object is affected by the viewpoint variation substantially

Separate attribute analysis. Attribute-level comparisons are performed on LaSOT benchmark for the proposed MIMTracking. The comparison results are illustrated in Figures 4-7, 4-8 and 4-9. From Figure 4-7, the proposed MIMTracking outperforms the state-of-the-art methods regarding the success rate (AUC score) in terms of almost all attributes. It also demonstrates that MIMTracking achieves exceptionally superior performance on several challenging attributes, e.g., fast motion, full occlusion, low resolution, motion blur and out-of-view. This suggests that the proposed MIMTracking has the strength in tracking unclear or incomplete objects. This can be explained by the masked image modeling (MIM) strategy. MIMTracking is trained with partial search tokens, so it is naturally capable of handling incomplete and even fully occluded targets.

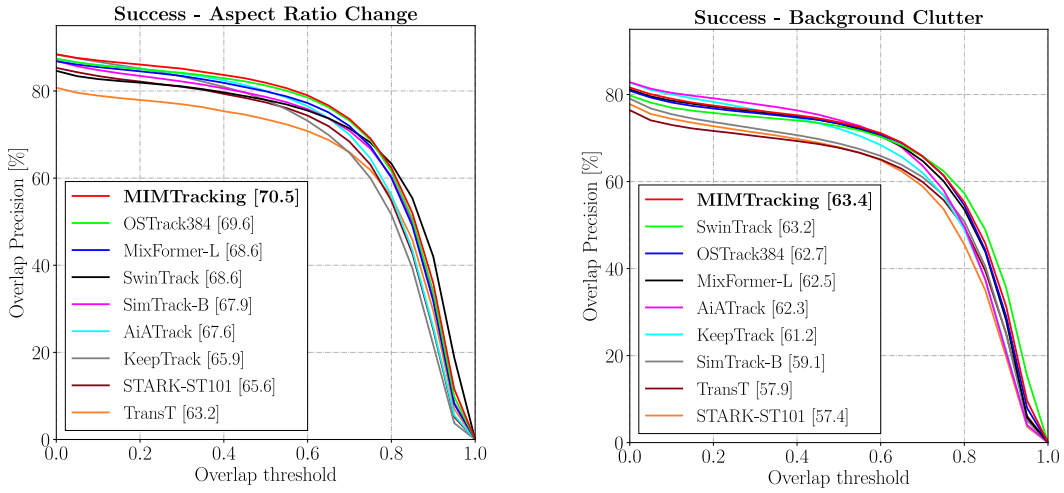


Figure 4-7 Attribute-level comparisons regarding success rate

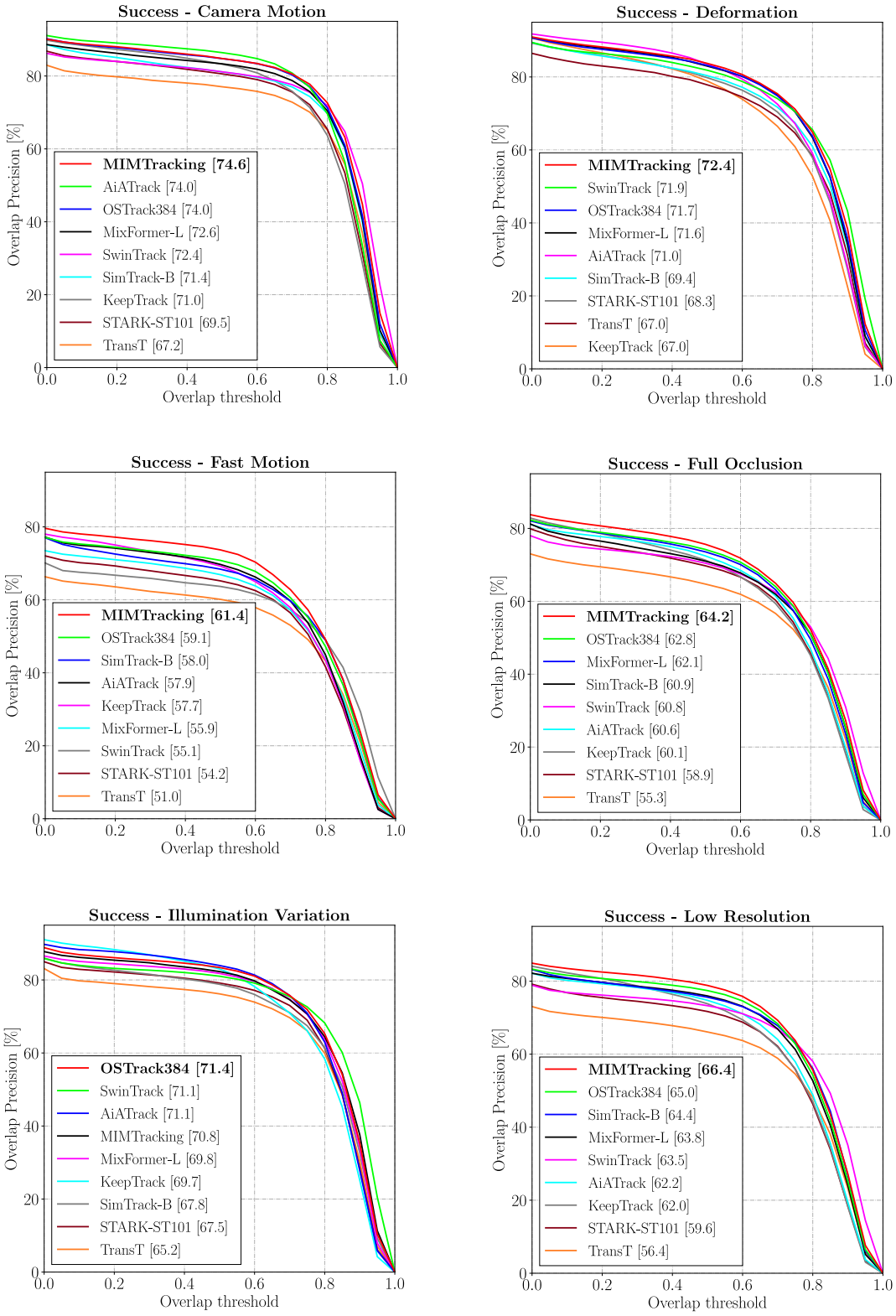


Figure 4-7 Attribute-level comparisons regarding success rate (continued)

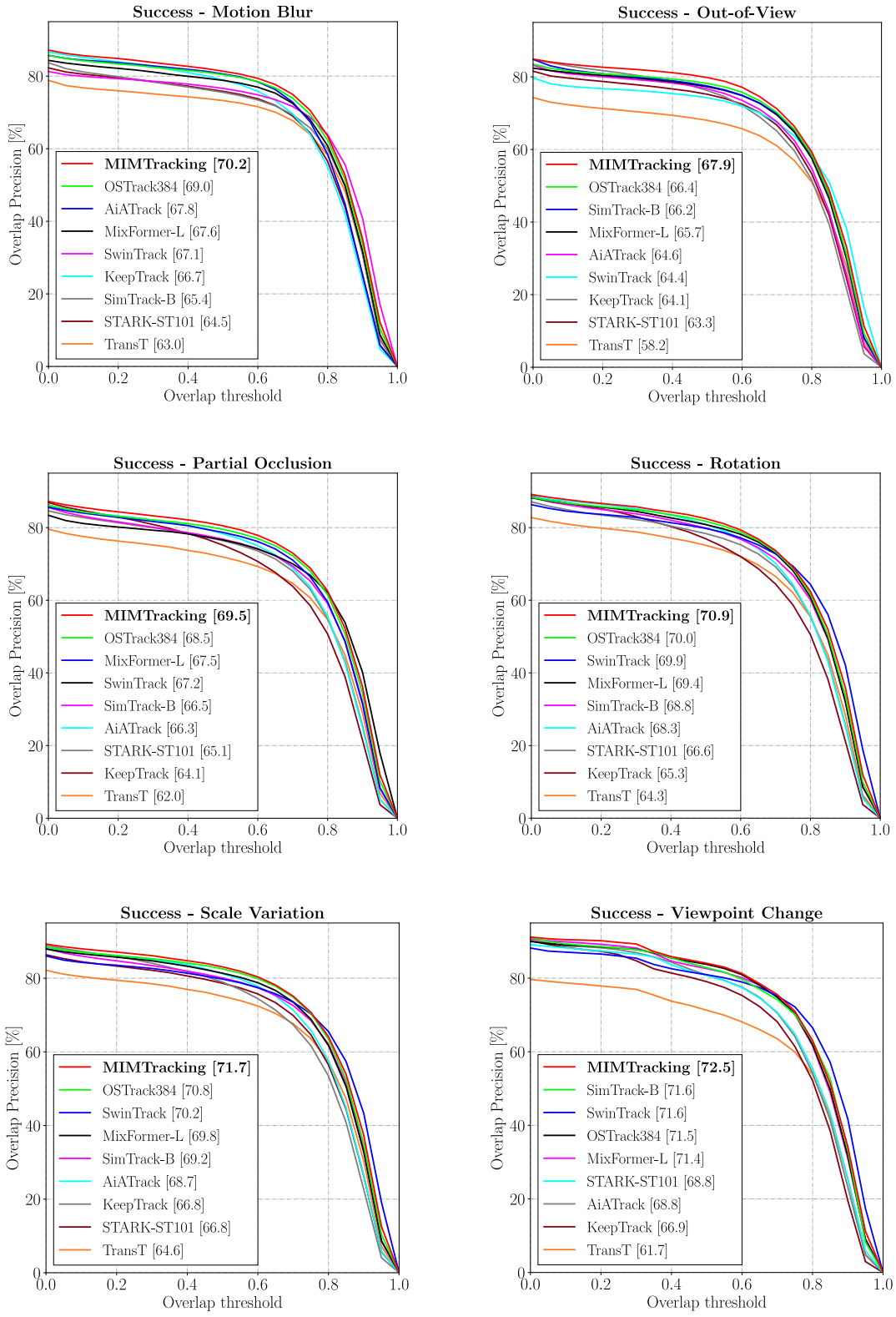


Figure 4-7 Attribute-level comparisons regarding success rate (continued)

Figure 4-8 and Figure 4-9 illustrate the attribute-level comparisons in terms of the precision and the normalized precision on LaSOT dataset. Similar to the success (AUC) results, the precision plots and normalized precision plots also demonstrate the superior tracking performance of the presented long-term tracker MIMTracking. It also suggests that MIMTracking significantly outperforms state-of-the-art tracking approaches on several extremely challenging attributes such as fast motion, full occlusion, low resolution, motion blur and out-of-view.

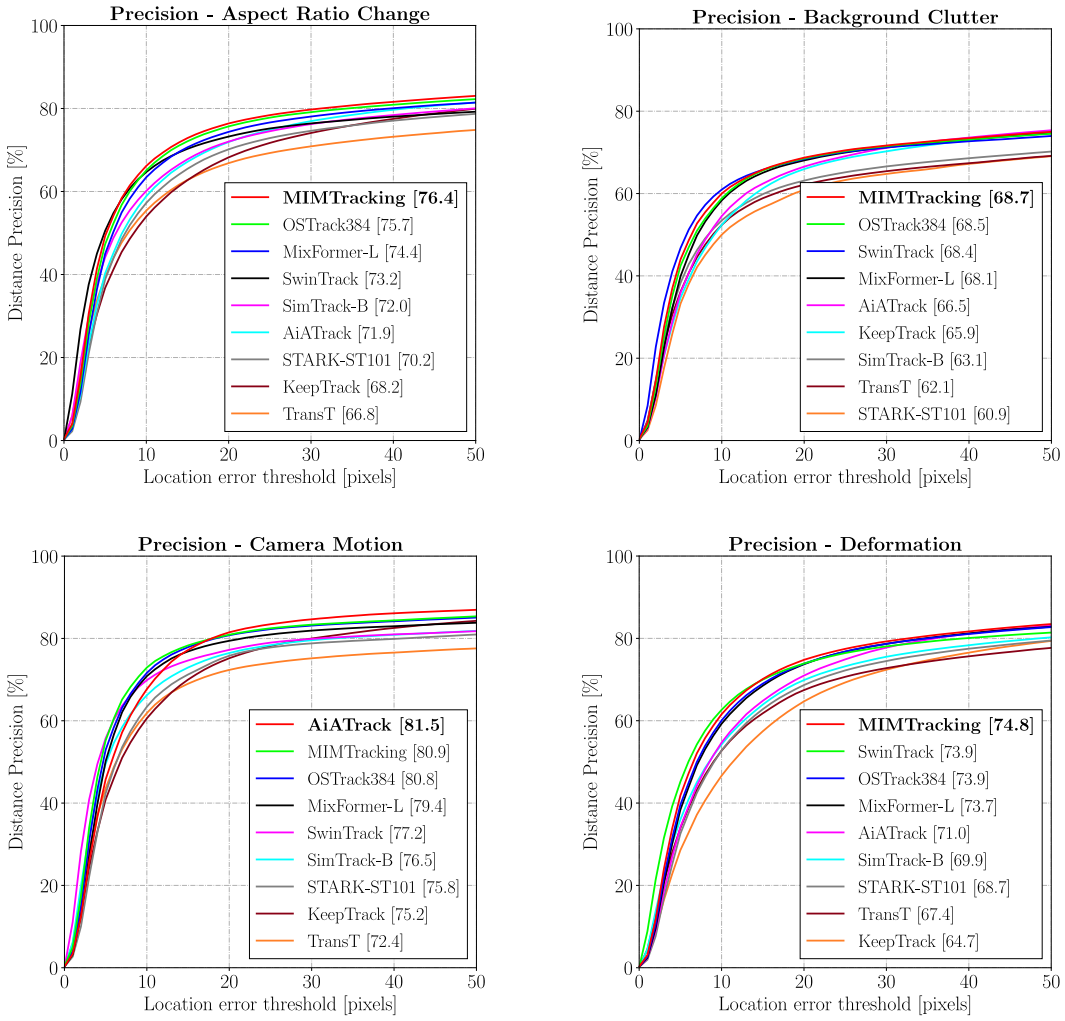


Figure 4-8 Attribute-level comparisons regarding precision

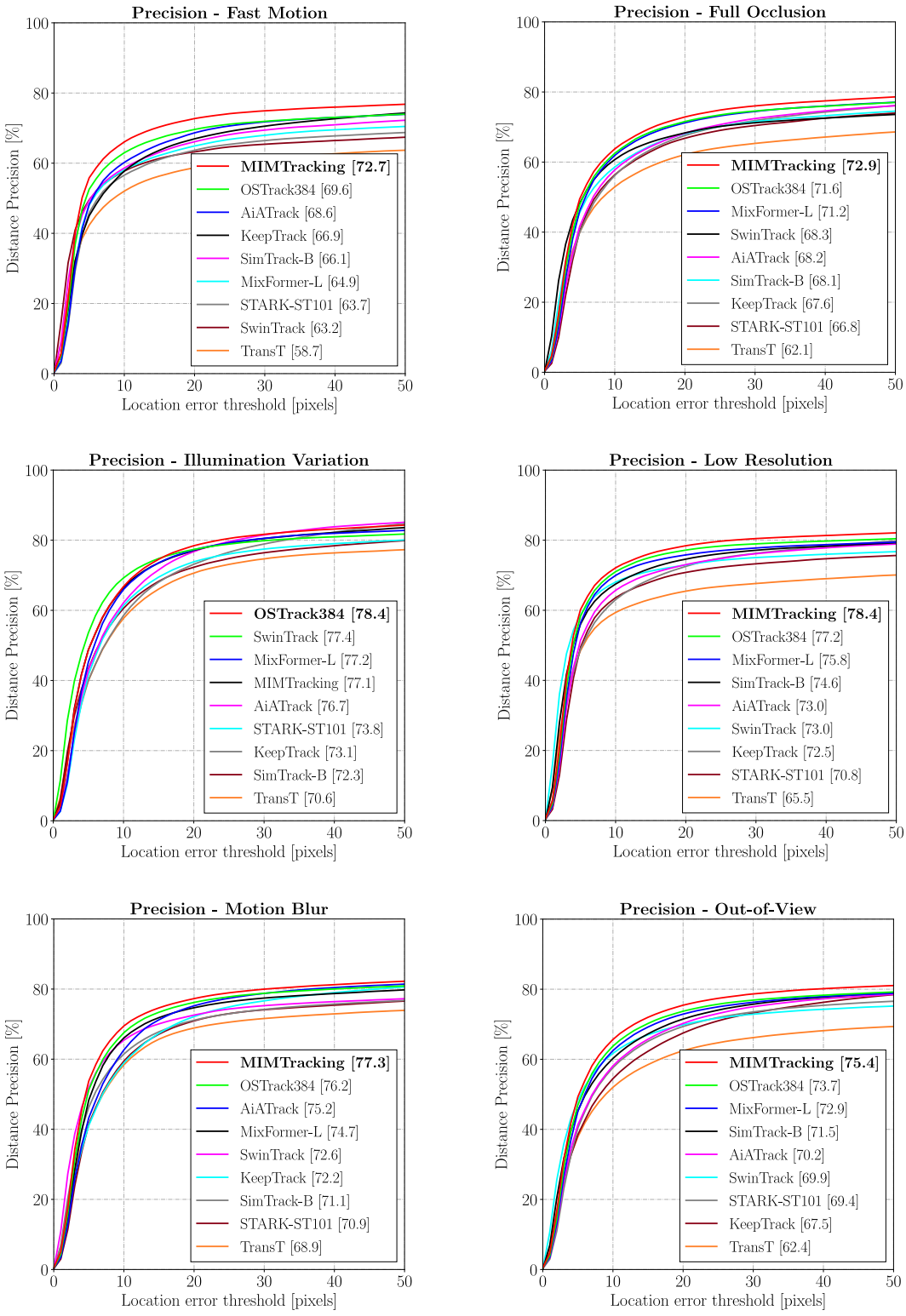


Figure 4-8 Attribute-level comparisons regarding precision (continued)

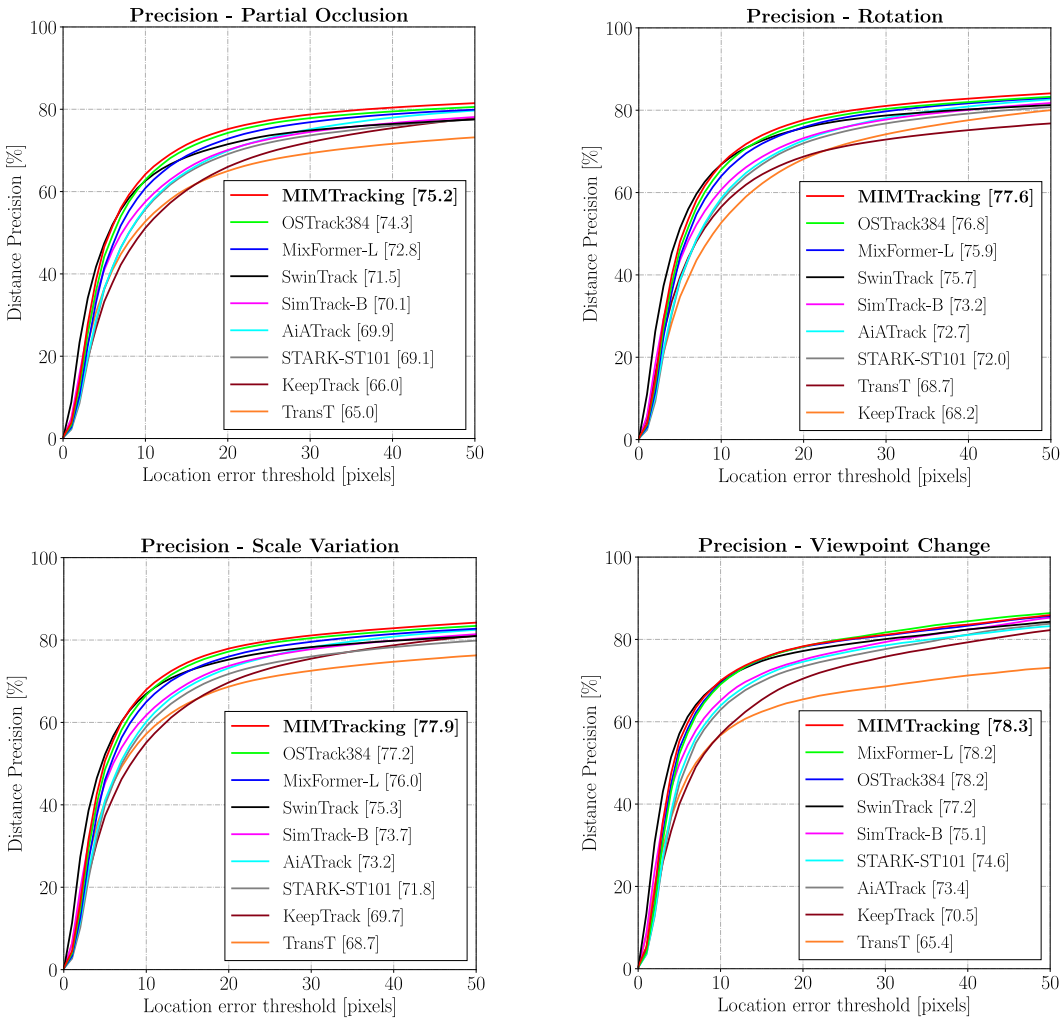


Figure 4-8 Attribute-level comparisons regarding precision (continued)

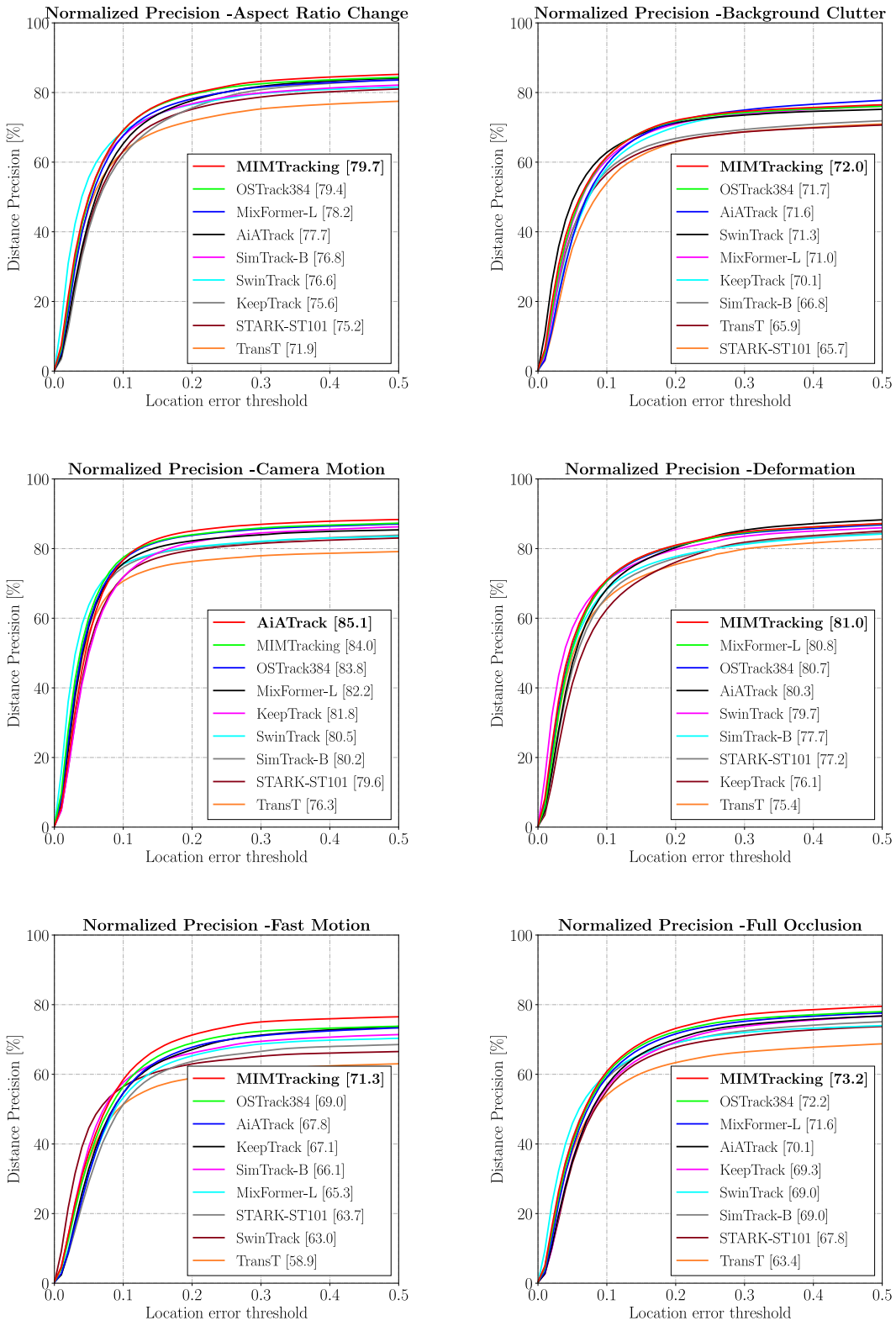


Figure 4-9 Attribute-level comparisons regarding normalized precision

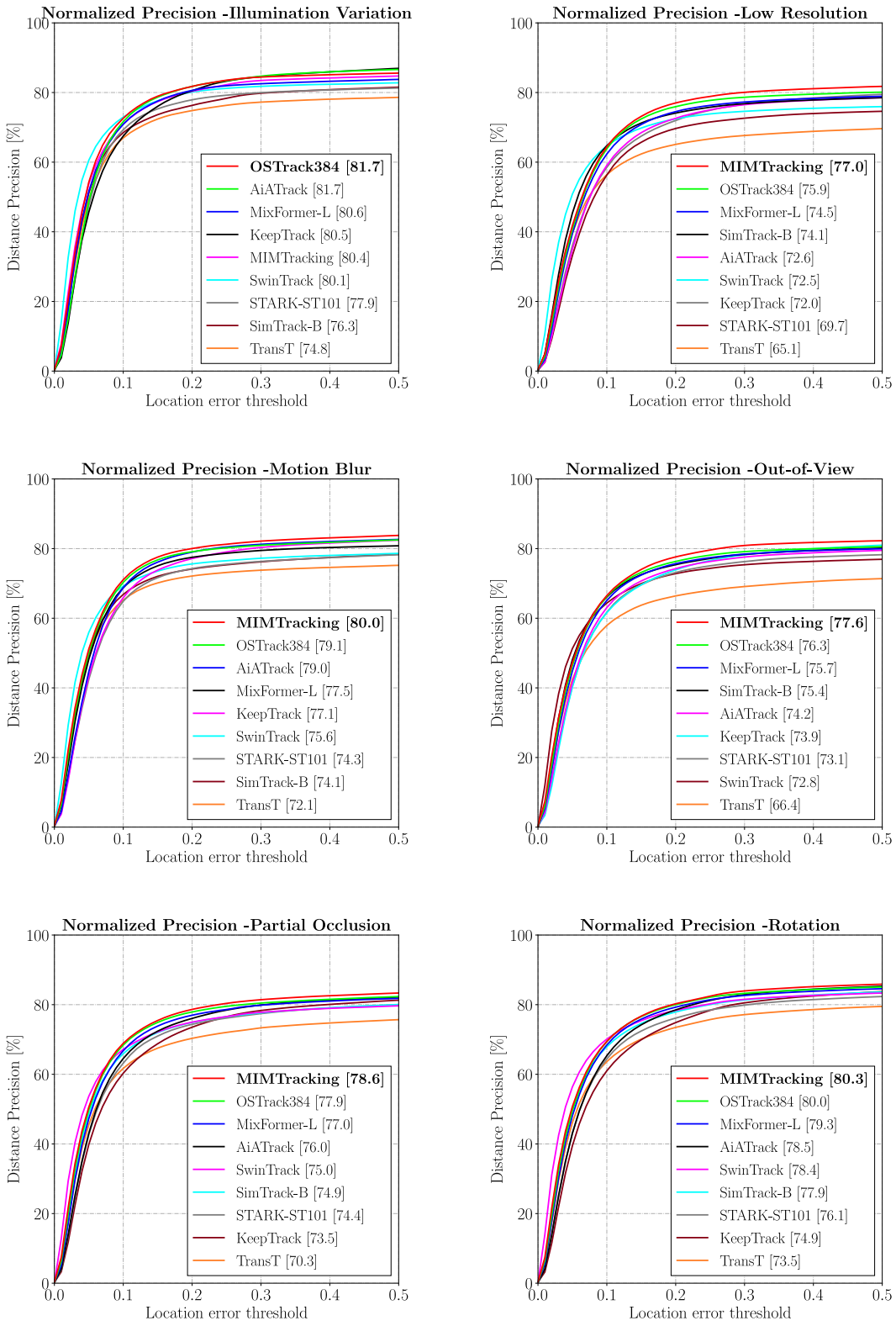


Figure 4-9 Attribute-level comparisons regarding normalized precision (continued)

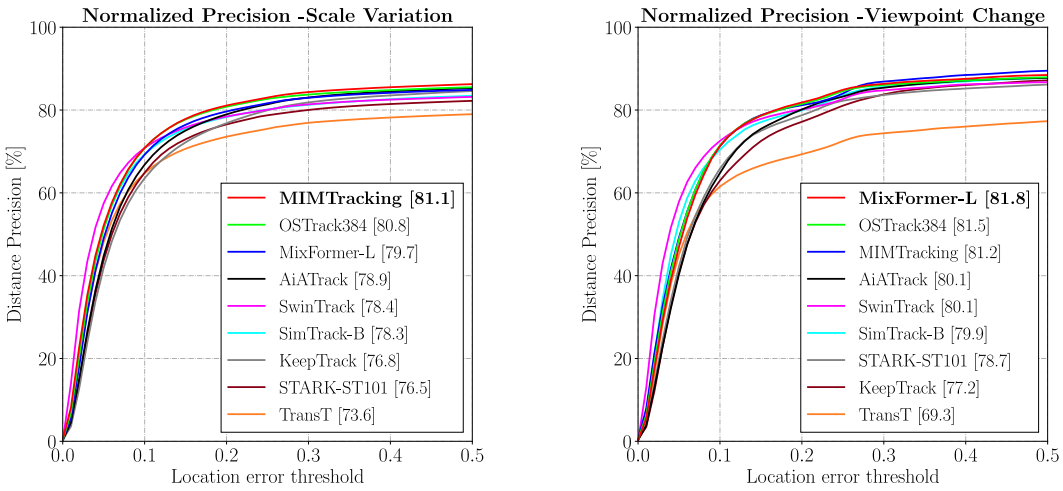


Figure 4-9 Attribute-level comparisons regarding normalized precision (continued)

4.4 Summary

This chapter introduces the presented long-term tracking method MIMTracking. Firstly, the important modules and the loss function of the tracking framework are explained in sequence. Secondly, a series of comparison experiments are conducted on multiple datasets to assess the proposed long-term tracking approach MIMTracking. Several ablation experiments are performed to evaluate the impact of the sampling ratio, search region size, etc. on tracking performance. In addition, attribute-level comparisons are also conducted for identifying the strengths and weaknesses of MIMTracking.

Chapter 5 Alternating pruning tracking under background clutter

5.1 Methodology

5.1.1 Alternating token pruning track

ATPTrack develops an alternating token pruning algorithm to simplify both dynamic templates and the search area. It highlights updated target appearance and further speeds the tracking model. In this section, the key components of the proposed tracker ATPTrack are elaborated. Figure 5-1(a) shows the overall framework of ATPTrack. Following the baseline method [63], two fixed templates $f_1, f_2 \in \mathbb{R}^{H_f \times W_f \times 3}$ and two dynamic templates $d_1, d_2 \in \mathbb{R}^{H_d \times W_d \times 3}$ are respectively cropped from the initial frame and an intermediate frame at two different proportions and then resized into the same template size 192 × 192. The search region $s \in \mathbb{R}^{H_s \times W_s \times 3}$ is cropped from the current frame and resized into the search size 384 × 384. Patch embedding layer is applied to divide the templates and the search region into non-overlapping patches with the size of 16 × 16.

Specifically, a convolution operation is separately utilized to each input template and search region to obtain all template and search representations. All representations are flattened and positionally encoded, resulting in fixed template tokens $f_1^t, f_2^t \in \mathbb{R}^{\frac{H_f}{16} \times \frac{W_f}{16} \times C}$, dynamic template tokens $d_1^t, d_2^t \in \mathbb{R}^{\frac{H_d}{16} \times \frac{W_d}{16} \times C}$ and search region tokens $s^t \in \mathbb{R}^{\frac{H_s}{16} \times \frac{W_s}{16} \times C}$. The embedding dimension C is 768. Three types of tokens are concatenated and then serve as the input of the first ViT block. Several ViT blocks are replaced by our proposed dynamic template pruning (DTP) or search region pruning (SRP) blocks for the purpose of alternating token pruning. Multi-head self-attention is a pivotal component in ViT, DTP

and SRP blocks, which is primarily responsible for feature interaction and association modeling among three types of input tokens.

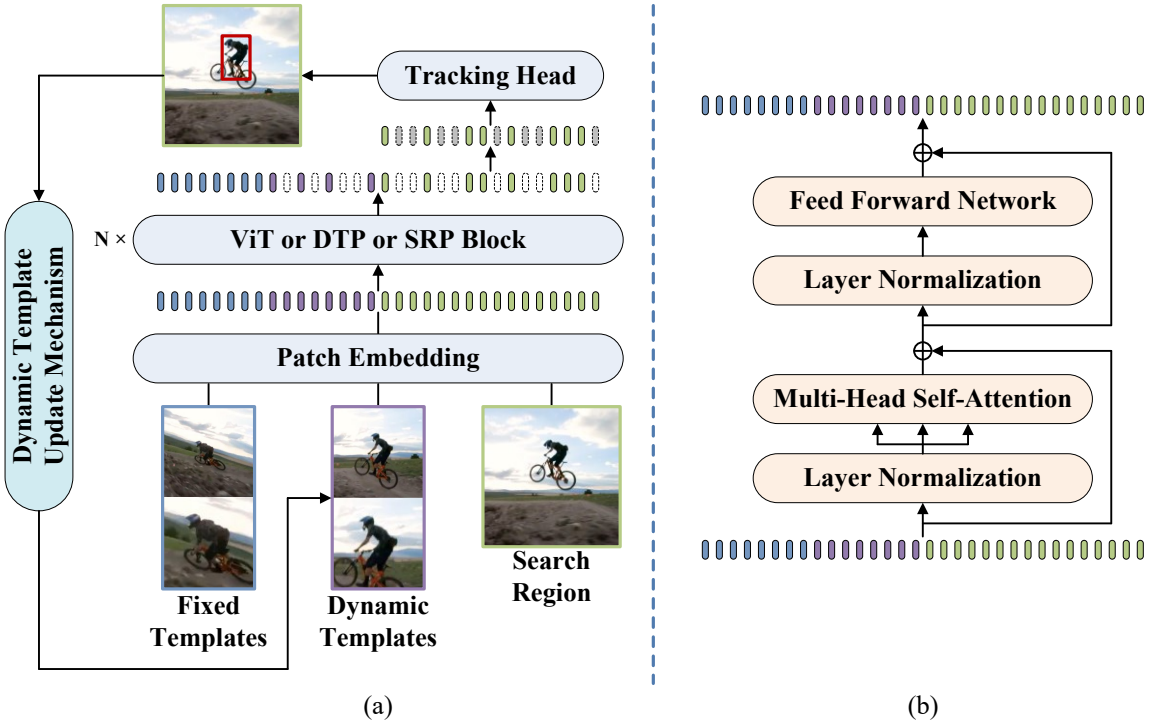


Figure 5-1 (a) The overall structure of ATPTrack (b) The architecture of ViT block

Attention relation modeling. Self-attention forms the foundational design of multi-head self-attention. For simplicity, self-attention is exploited to explain the associations of input tokens. Self-attention of concatenated tokens is described in Eq. (5-1).

$$
\begin{aligned}
SA(Q,K,V) &= \text{softmax}(\frac{QK^T}{\sqrt{d_k}})\cdot V \\
&= \text{softmax}(\frac{[Q_{f_1};Q_{f_2};Q_{d_1};Q_{d_2};Q_s][K_{f_1};K_{f_2};K_{d_1};K_{d_2};K_s]^T}{\sqrt{d_k}})\cdot \begin{bmatrix} V_{f_1} \\ V_{f_2} \\ V_{d_1} \\ V_{d_2} \\ V_s \end{bmatrix} \\
&= \text{softmax}(\frac{\begin{bmatrix} Q_{f_1}K_{f_1}^T & Q_{f_1}K_{f_2}^T & Q_{f_1}K_{d_1}^T & Q_{f_1}K_{d_2}^T & Q_{f_1}K_s^T \\ Q_{f_2}K_{f_1}^T & Q_{f_2}K_{f_2}^T & Q_{f_2}K_{d_1}^T & Q_{f_2}K_{d_2}^T & Q_{f_2}K_s^T \\ Q_{d_1}K_{f_1}^T & Q_{d_1}K_{f_2}^T & Q_{d_1}K_{d_1}^T & Q_{d_1}K_{d_2}^T & Q_{d_1}K_s^T \\ Q_{d_2}K_{f_1}^T & Q_{d_2}K_{f_2}^T & Q_{d_2}K_{d_1}^T & Q_{d_2}K_{d_2}^T & Q_{d_2}K_s^T \\ Q_sK_{f_1}^T & Q_sK_{f_2}^T & Q_sK_{d_1}^T & Q_sK_{d_2}^T & Q_sK_s^T \end{bmatrix}}{\sqrt{d_k}})\cdot \begin{bmatrix} V_{f_1} \\ V_{f_2} \\ V_{d_1} \\ V_{d_2} \\ V_s \end{bmatrix}
\end{aligned}
\tag{5-1}
$$

where Q,K,V denote respectively the query, the key and the value in self-attention. The three items are separately generated by applying three linear transformations to the concatenated tokens $[f_1^t, f_2^t, d_1^t, d_2^t, s^t]$. The superscript t is omitted for all concatenated tokens in Eq. (5-1) for the sake of simplicity. T and d_k represent the transpose of a matrix and a scaling adjustment factor, respectively.

It is obvious from Eq. (5-1) that the relations of diverse input tokens are established by the self-attention operation. For example, $Q_{f_1}K_{f_1}^T$, $Q_{d_1}K_{d_2}^T$, etc. denote associations of the same type of tokens. $Q_{f_1}K_{d_2}^T$, $Q_{f_2}K_s^T$, etc. show the relations of different types of tokens. Multi-head self-attention contains multiple parallel self-attention mechanisms that focus on various perspectives, thereby building stronger dependencies among inputs. Therefore, complicated latent representations and semantic information can be modeled for the target by multiple attention layers.

DTP block. As shown in Figure 5-2, DTP block is developed for dynamic template trimming. Motivated by [44], [63], DTP block aggregates a novel similarity ranking scheme into the original ViT block for progressive template trimming. After multi-head self-attention, the attention weights are fed into the similarity ranking scheme for selecting important dynamic template tokens. From Eq. (5-2), D_1 denotes the mean token similarity set of two fixed templates f_1, f_2 and the first dynamic template d_1. f_1^c and f_2^c represent the central tokens of two fixed templates, separately. n is the number of tokens in dynamic templates d_1 or d_2. D_2 in Eq. (5-3) shows the mean token similarities of two fixed templates and the second dynamic template d_2. Both D_1 and D_2 are ranked in descending order and the indices of a certain proportion of tokens with high similarities are recorded in Eq. (5-4) and Eq. (5-5). Template retention ratio is utilized to determine the proportion of retained tokens for each dynamic template.

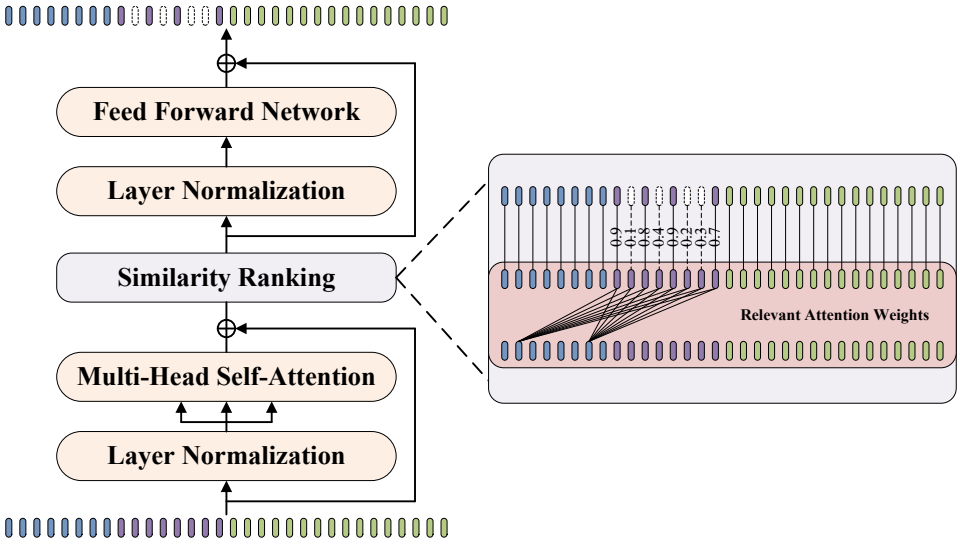


Figure 5-2 The structure of the dynamic template pruning (DTP) block

$$D_1 = \left\{ \frac{1}{2\sqrt{d_k}} \left(Q_{f_1^c} K_{d_1}^T(i) + Q_{f_2^c} K_{d_1}^T(i) \right) \Big|_{i \in \{1,2,\ldots\ldots,n\}} \right\} \tag{5-2}$$

$$D_2 = \left\{ \frac{1}{2\sqrt{d_k}} \left(Q_{f_1^c} K_{d_2}^T(j) + Q_{f_2^c} K_{d_2}^T(j) \right) \Big|_{j \in \{1,2,\ldots\ldots,n\}} \right\} \tag{5-3}$$

$$idx_{d_1} = \text{Index}_{\text{Retention}} \left(\text{Ranking}_{\downarrow}(D_1) \right) \tag{5-4}$$

$$idx_{d_2} = \text{Index}_{\text{Retention}} \left(\text{Ranking}_{\downarrow}(D_2) \right) \tag{5-5}$$

As illustrated in Eq. (5-6), dynamic template tokens with indices idx_{d_1} or idx_{d_2} are kept and concatenated with all fixed template tokens f_1^t, f_2^t and search area tokens s^t. The resulting C_1 serves as the input of following layers (layer normalization and feed forward network). So far, the first step of alternating token pruning is completed. The original ViT blocks with block numbers 3, 6, and 9 are replaced by the proposed DTP blocks. A large number of uninformative dynamic template tokens are dropped gradually, thereby ensuring the dependability of remaining dynamic tokens and improving the efficiency.

$$C_1 = \text{Concatenate}\left(f_1^t, f_2^t, d_1^t(idx_{d_1}), d_2^t(idx_{d_2}), s^t \right) \tag{5-6}$$

SRP block. Following the DTP block, SRP block is responsible for search region pruning. Figure 5-3 demonstrates the structure of the SRP block. The similarity ranking mechanism in our SRP block takes into account the attention weights of two fixed templates and the search region. $Q_{f_1^c} K_s^T$ and $Q_{f_2^c} K_s^T$ of Eq. (5-7) denote separately the attention weights of two central tokens f_1^c, f_2^c and search region tokens *s*. *q* is the number of search region tokens. A mean similarity score is computed for each search token by averaging the two attention weights. Similarity scores of all search tokens are sorted in descending order in Eq. (5-8). The high-score search tokens are selected and concatenated with fixed template tokens and the trimmed dynamic tokens in Eq. (5-9). The search retention ratio is utilized to control the number of retained search tokens.

To achieve alternating pruning of templates and the search area, the locations of SRP blocks are set to 4, 7 and 10. A pair of adjacent DTP and SRP blocks completes an alternating token pruning. A total of 3 alternating pruning operations are implemented in the backbone. Therefore, the DTP, SRP, and ViT blocks are arranged in combinations of 3, 3, 6 to form a diverse 12-stage backbone. The target features are highlighted and interacted progressively for all templates and search region.

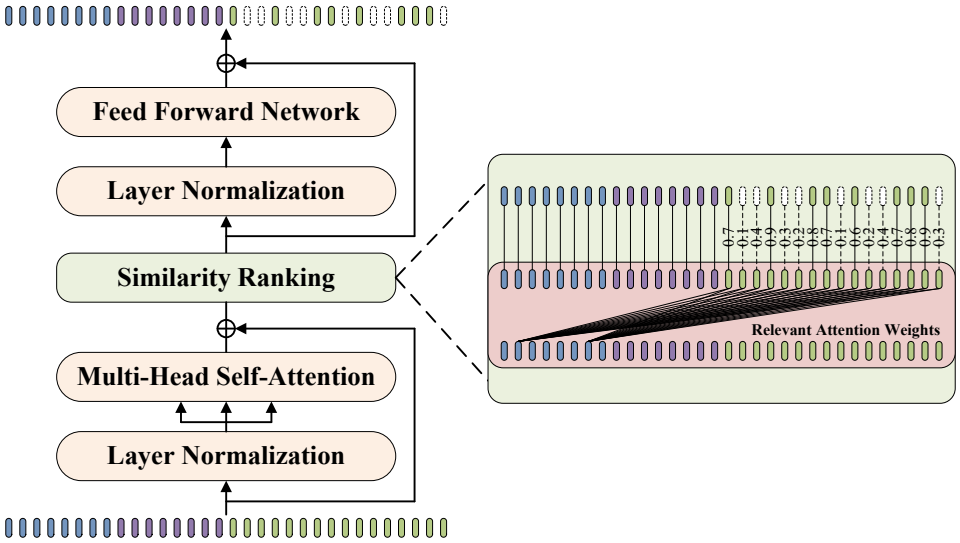


Figure 5-3 The structure of the search region pruning (SRP) block

$$S=\left\{\frac{1}{2\sqrt{d_k}}\left(Q_{f_1^c}K_s^T(k)+Q_{f_2^c}K_s^T(k)\right)\Big|_{k\in\{1,2,\ldots\ldots,q\}}\right\} \tag{5-7}$$

$$idx_s=\mathrm{Index}_{\mathrm{Retention}}\left(\mathrm{Ranking}_{\downarrow}(S)\right) \tag{5-8}$$

$$C_2=\mathrm{Concatenate}\left(f_1^t,f_2^t,d_1^t(idx_{d_1}),d_2^t(idx_{d_2}),s^t(idx_s)\right) \tag{5-9}$$

Tracking result prediction. The tracking results (bounding boxes) are predicted by a tracking head. The output tokens of the backbone that belong to the search region are extracted with zero-padding applied to the pruned positions, leading to a full-length one-

dimensional feature tensor. This tensor is converted into a two-dimensional feature map to fit with the tracking head.

The tracking head employs three identical fully convolutional structures for predicting the classification map, size map and offset map of the target, respectively. Each structure of the tracking head consists of four sets of Conv-BN-ReLU layers. The location possessing the highest score in the classification map is utilized as the target center. The target size and central offset of this location are extracted from the size map and offset map, separately. Therefore, the target center and target size are respectively considered as the bounding box center and size. The central offset is added to the coordinates of the bounding box center to correct errors due to resolution changes.

Template update mechanism. Timely updating of dynamic templates contributes to enhancing the robustness and discriminative capability of the tracking model. Three random frames with certain intervals are sampled from a video sequence and respectively serve as the fixed, dynamic templates and search region during training. This process updates all templates and search region for each iteration. Thus, there is no specific update strategy in the training process.

The dynamic template update mechanism is embedded into the inference process. The maximum value of the classification map is regarded as the reliability score to determine the likelihood of the current frame being utilized to update the dynamic templates. Template update interval has a significant impact on tracking performance. Extensive experiments are conducted to seek for the optimal update interval. After reaching the optimal interval, the reliability score is utilized to determine whether the current frame can

act as dynamic templates. If the reliability score surpasses a threshold, this frame is employed to update the dynamic templates.

The baseline [63] leverages fixed templates repeatedly as dynamic templates until the appearance of dynamic templates. This is redundant since fixed templates are identical to dynamic templates at the beginning of tracking. Differently, the proposed ATPTrack merely utilizes fixed templates to reduce identical tokens before obtaining dynamic templates. Furthermore, ATPTrack trims dynamic templates frequently to improve the inference speed.

5.1.2 Training objective

The objective function of ATPTrack is applied to the predicted classification map and regression output (bounding box coordinates) to train a precise tracking network. For classification, the focal function [146] is employed to compute the difference between the estimated classification map and the ground-truth map, and to minimize this difference. For regression, the L_1 loss and GIoU loss [118] are separately leveraged to reduce the coordinate discrepancy and to maximize the overlapping area between the predicted and the ground-truth bounding boxes. The total weighted loss is described in Eq. (5-10).

$$L_{total} = L_{cls}(y_{cls}, \hat{y}_{cls}) + \mu_1 L_{giou}(y_{box}, \hat{y}_{box}) + \mu_2 L_1(y_{box}, \hat{y}_{box}) \tag{5-10}$$

where y_{cls} and y_{box} denote respectively the predicted classification map and bounding box coordinates. $\hat{y}_{cls}$ and $\hat{y}_{box}$ indicate the corresponding ground-truth annotations of the classification and regression outputs, separately. Two weighting coefficients μ_1 and μ_2 are equal to 2 and 5, respectively.

5.2 Comparison experiments

5.2.1 Implementation details

Tracking model. The proposed ATPTrack exploits Vanilla ViT-Base [82] as the backbone. 6 out of 12 ViT blocks of the backbone are replaced by our developed DTP and SRP blocks for the purpose of alternating token pruning. DTP blocks are distributed at the positions of the 3rd, 6th, and 9th layers of the backbone. SRP blocks are arranged at three positions of the backbone: the 4th, 7th, and 10th stages, separately. The sizes of fixed templates, dynamic templates and the search area are 192×192, 192×192 and 384×384, respectively. The backbone of ATPTrack is initialized by the DropMAE [46] pretrained ViT-B weights. Additionally, ATPTrack is implemented with Python 3.8.13 and Pytorch 1.10.0.

Training. The tracking model is trained with the aid of the Intel Core i9 CPU and two RTX A6000 GPUs. The training splits of four popular datasets are leveraged to train our ATPTrack, which are respectively LaSOT [50], TrackingNet [120], COCO [121] and GOT-10k [119]. ATPTrack that is trained on these four datasets can be tested on most datasets except GOT-10k following GOT-10k evaluation protocol. Therefore, to evaluate the results of GOT-10k benchmark, the tracking model should be trained only on GOT-10k training set. Moreover, brightness jittering and horizontal flip are utilized for image augmentation. The tracking model is trained with AdamW optimizer. The overall batch size is 32 with each GPU processing 16 image pairs. The initial learning ratios of the backbone and other parts are respectively set to 2×10^{-5} and 2×10^{-4} with a weight decay factor 10^{-4}. The total epoch number for training is 300 with 60000 sample pairs every epoch. The learning ratios drop by a factor of 10 after 240 epochs. The GOT-10k model is trained for 100 epochs with learning ratios decreasing after 80 epochs.

Inference. The update interval of dynamic templates is 50 and the threshold for template update is 0.7 during inference. Template and search retention ratios are both 0.7 during training and inference after extensive experiments. The inference speed of the tracking model is obtained from one RTX A6000 GPU. Table 5-1 enumerates the implementation details for clearer demonstration.

Table 5-1 Implementation details

Settings	Values
Backbone of ATPTrack	Vanilla ViT-Base
DTP block locations	Layers 3^{rd}, 6^{th}, and 9^{th}
SRP block locations	Layers 4^{th}, 7^{th}, and 10^{th}
Center jitter factor	4.5
Scale jitter factor	0.5
GPU	Two Nvidia RTX A6000
CPU	Intel Core i9
The operating system	Ubuntu 20.04
Python	3.8.13
Pytorch	1.10.0
Number of fixed templates	2
Number of dynamic templates	2
Fixed template size	192×192
Dynamic template size	192×192
Search region size	384×384
Epoch number	300
Samples per epoch	60000
Batch size	32
Dynamic template update interval	50
Dynamic template update threshold	0.7
Template retention ratio	0.7
Search retention ratio	0.7
Optimizer	AdamW
Learning rate of backbone	2×10^{-5}
Learning rate of other parameters	2×10^{-4}

Learning rate decay epoch	240
Weight decay factor	10^{-4}

5.2.2 Comparison with SOTA methods

The proposed tracker ATPTrack is evaluated on five commonly utilized benchmarks. The tracking performance is assessed by comparing ATPTrack with state-of-the-art trackers. The results are depicted in Table 5-2, Table 5-3 and Table 5-4.

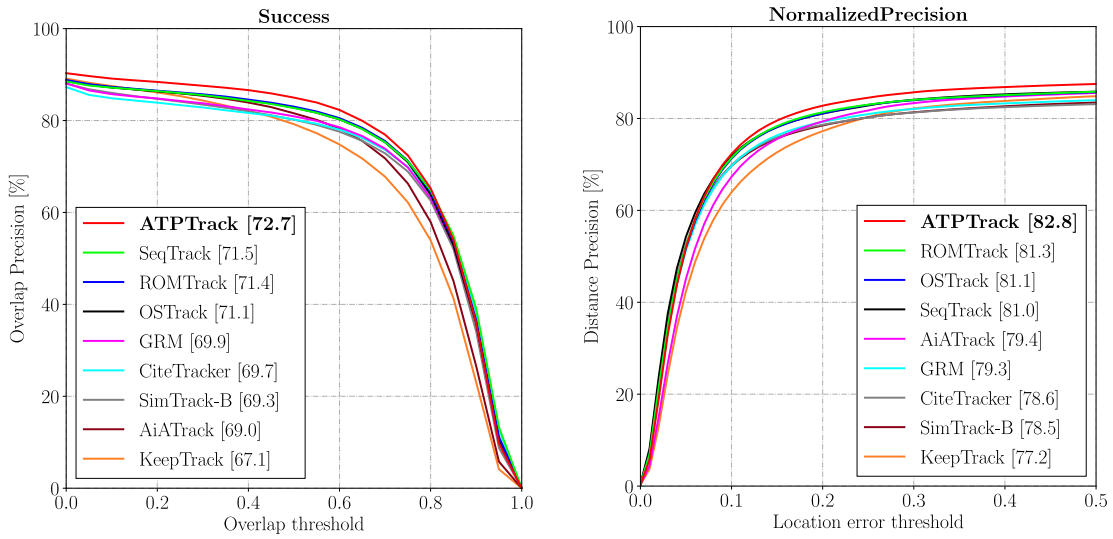
Table 5-2 Overall performance comparison on LaSOT and $\text{LaSOT}_{\text{ext}}$

Trackers	LaSOT			$\text{LaSOT}_{\text{ext}}$		
	AUC	P_{norm}	P	AUC	P_{norm}	P
SiamRPN++ [73]	49.6	56.9	49.1	34.0	41.6	39.6
DiMP [72]	56.9	65.0	56.7	39.2	47.6	45.1
LTMU [40]	57.2	-	57.2	41.4	49.9	47.3
TrDiMP [148]	63.9	-	61.4	-	-	-
TransT [29]	64.9	73.8	69.0	-	-	-
KeepTrack [57]	67.1	77.2	70.2	48.2	-	-
SwinTrack [80]	69.6	78.6	74.1	47.6	58.2	54.1
STARK [30]	67.1	77.0	-	-	-	-
AiATrack [81]	69.0	79.4	73.8	47.7	55.6	55.4
MixFormer [42]	69.2	78.7	74.7	-	-	-
SimTrack [43]	70.5	79.7	-	-	-	-
SwinV2 [48]	70.7	-	-	-	-	-
OSTrack [44]	71.1	81.1	77.6	50.5	61.3	57.6
MAT [49]	67.8	77.3	-	-	-	-
TATrack [62]	69.4	78.2	74.1	-	-	-
GRM [79]	69.9	79.3	75.8	-	-	-
SeqTrack [83]	71.5	81.1	77.8	50.5	61.6	57.5
DropTrack [46]	71.8	81.8	78.1	52.7	63.9	60.2
ROMTrack [84]	71.4	81.4	78.2	51.3	62.4	58.6
F-BDMTrack [153]	72.0	81.5	77.7	50.8	61.3	57.8
CiteTracker [154]	69.7	78.6	75.7	-	-	-
MixViT [155]	70.4	80.4	76.7	-	-	-

ATPTrack	72.7	82.8	79.3	52.9	63.9	60.3

Note: The red and blue numbers are the top two ranked.

LaSOT [50]. LaSOT is responsible for large-scale long-term object tracking, which adopts 1120 and 280 videos for training and testing, separately. Table 5-2 summarizes the tracking results of different tracking approaches on LaSOT dataset. Clearly, the proposed ATPTrack achieves state-of-the-art AUC of 72.7% on LaSOT, exceeding top-performance tracker F-BDMTrack [153] by 1.0%. This suggests that ATPTrack possesses superior long-term tracking capability. The success and normalized precision plots are shown in Figure 5-4.



(a) Success plot (b) Normalized precision plot

Figure 5-4 Success and normalized precision plots of ATPTrack on LaSOT

LaSOT$_{ext}$ [151]. LaSOT$_{ext}$ extends LaSOT benchmark by using challenging sequences. 150 video sequences from 15 new categories are leveraged to assess tracking methods in a more demanding way. ATPTrack is compared with the state-of-the-art (SOTA) tracking algorithms. The results in Table 5-2 indicate that the tracking performance of ATPTrack is on par with the SOTA trackers on LaSOT$_{ext}$.

TrackingNet [120]. TrackingNet covers large-scale scenarios for visual object tracking. It provides over 30000 videos for model training and 511 sequences for performance evaluation of tracking methods. The tracking results are assessed on the online server. As depicted in Table 5-3, our ATPTrack achieves an AUC of 85.2% on TrackingNet. It outperforms the state-of-the-art trackers in terms of all three metrics.

GOT-10k [119]. GOT-10k provides videos with diverse target categories and motion patterns. The training and test sets of GOT-10k contain respectively 10000 and 180 sequences. To verify the adaptability of tracking methods, sequences of different target classes are utilized for model training and testing. The comparison results are shown in Table 5-3. ATPTrack obtains competitive tracking results compared with current promising trackers.

Table 5-3 Overall performance comparison on TrackingNet and GOT-10k

Trackers	GOT-10k			TrackingNet		
	AO	$SR_{0.5}$	$SR_{0.75}$	AUC	P_{norm}	P
SiamRPN++ [73]	51.7	61.6	32.5	73.3	80.0	69.4
DiMP [72]	61.1	71.7	49.2	74.0	80.1	68.7
TrDiMP [148]	67.1	77.7	58.3	78.4	83.3	73.1
TransT [29]	67.1	76.8	60.9	81.4	86.7	80.3
SwinTrack [80]	69.4	78.0	64.3	82.5	87.0	80.4
STARK [30]	68.8	78.1	64.1	82.0	86.9	-
AiATrack [81]	69.6	80.0	63.2	82.7	87.8	80.4
MixFormer [42]	70.7	80.0	67.8	83.1	88.1	81.6
SimTrack [43]	69.8	78.8	66.0	83.4	87.4	-
SwinV2 [48]	72.9	-	-	82.5	-	-
OSTrack [44]	73.7	83.2	70.8	83.9	88.5	83.2
MAT [49]	67.7	78.4	-	81.9	86.8	-
TATrack [62]	73.0	83.3	68.5	83.5	88.3	81.8
GRM [79]	73.4	82.9	70.4	84.0	88.7	83.3
SeqTrack [83]	74.5	84.3	71.4	83.9	88.8	83.6

ProContEXT [63]	74.6	84.7	72.9	84.6	89.2	83.8
DropTrack [46]	75.9	86.8	72.0	84.1	88.9	-
ROMTrack [84]	74.2	84.3	72.4	84.1	89.0	83.7
F-BDMTrack [153]	75.4	84.3	72.9	84.5	89.0	84.0
CiteTracker [154]	74.7	84.3	73.0	84.5	89.0	84.2
MixViT [155]	74.3	84.1	73.0	84.5	89.1	83.7
ATPTrack	75.8	86.6	72.2	85.2	89.6	84.4

Note: The red and blue numbers are the top two ranked.

TNL2K [152]. TNL2K provides 1300 and 700 sequences separately for training and testing with language explanations serving as additional annotations for joint visual language tracking. The proposed ATPTrack is evaluated and compared with the state-of-the-art approaches on TNL2K. The results are illustrated in Table 5-4. It shows that ATPTrack improves the AUC to 59.6%, surpassing the SOTA tracker F-BDMTrack by 3.1%. This further demonstrates the admirable tracking performance of ATPTrack.

Table 5-4 Tracking performance comparison on TNL2K dataset

Methods	AUC	P
DiMP [72]	44.7	43.4
TransT [29]	50.7	51.7
SwinTrack [80]	55.9	57.1
SimTrack [43]	55.6	55.7
OSTrack [44]	55.9	-
MAT [49]	51.3	-
SeqTrack [83]	56.4	-
DropTrack [46]	56.9	57.9
F-BDMTrack [153]	57.8	59.4
CiteTracker [154]	57.7	59.6
ATPTrack	59.6	61.9

Note: The top two results are demonstrated in red and blue fonts.

5.3 Ablation experiments

Comprehensive ablation experiments are conducted on tracking benchmarks to evaluate our ATPTrack regarding its crucial components.

Template retention ratio ablation. Template retention ratio and search retention ratio denote respectively the proportion of retained tokens for each dynamic template and search region. Search retention ratio is empirically set to 0.7 based on the success of [44], [63] in search region trimming. Multiple attempts are conducted to determine the optimal retention ratio for dynamic templates. Specifically, several fixed retention ratios (0.5, 0.6, 0.7, 0.8, 0.9, 1.0) are separately exploited to train and evaluate the tracking model on TNL2K and LaSOT benchmarks. The results are illustrated in Table 5-5.

From Table 5-5, the template retention ratio of 0.7 achieves optimal trade-off between tracking performance and running speed. The retention ratio of 0.8 slightly surpasses 0.7 on TNL2K, but is much less competitive on LaSOT dataset. Moreover, the computational cost of 0.8 is much higher than 0.7. A retention ratio of 1.0 indicates the removal of dynamic template pruning. Therefore, merely the search area is trimmed during tracking. A retention ratio less than 1.0 (e.g., 0.7) denotes alternating pruning of dynamic templates and search region. It is observed that pruning both dynamic templates and search region in a ratio of 0.7 significantly reduces the MACs by 11.5% with a slight decrease (0.3%) in AUC compared with only pruning the search region. As a result, 0.7 is ultimately selected as the template retention ratio for dynamic template trimming.

Table 5-5 Effect of the template retention ratio on tracking results, computational costs (MACs) and running speeds (FPS)

Retention ratios	TNL2K			LaSOT			MACs (G)	FPS
	AUC	P_{norm}	P	AUC	P_{norm}	P		
0.5	58.8	75.2	60.6	71.3	80.8	76.8	71.1	87.9
0.6	59.3	76.1	61.3	72.3	82.2	78.4	73.2	84.6
0.7	59.6	76.6	61.9	72.7	82.8	79.3	75.5	82.1
0.8	59.7	76.7	62.2	72.4	82.6	78.9	78.3	79.3
0.9	59.3	76.3	61.6	71.8	81.6	77.8	81.6	76.6
1.0	59.8	76.9	62.3	72.9	83.2	79.8	85.3	75.8

Notes:

1. The best three results are highlighted in red, blue and green fonts, respectively.

2. All speeds are measured on one RTX A6000 GPU.

Adaptive retention ratio. The adaptive retention ratio is designed as well for dynamic template pruning. This is used for comparison with the fixed retention ratios mentioned in the previous section.

The adaptive retention ratio (*arr*) is defined in Eq. (5-11). $H_{gt\_box} \times W_{gt\_box}$ denotes the area of the ground-truth bounding box for current dynamic templates. $H_d \times W_d$ represents the area of current dynamic templates. λ is an amplification factor utilized to expand the proportion of the bounding box, thereby reducing the template pruning errors caused by over-trimming. The variation in $H_{gt\_box} \times W_{gt\_box}$ is due to the updating of dynamic templates, resulting in an adaptive retention ratio. *arr* is utilized to replace the fixed retention ratio in three corresponding blocks.

$$arr = \sqrt[3]{\frac{H_{gt\_box} \times W_{gt\_box}}{H_d \times W_d} \times \lambda} \tag{5-11}$$

Table 5-6 demonstrates the effect of the amplification factor λ on tracking performance. It

is clear that 2.0 achieves the best performance trade-off on two datasets. However, the performance of optimal adaptive retention ratio is still lower than the fixed retention ratio 0.7. It suggests that the fixed retention ratio 0.7 is more suitable for dynamic template pruning.

Table 5-6 Effect of the amplification factor on tracking performance on TNL2K and LaSOT benchmarks

Amplification factor	TNL2K			LaSOT		
	AUC	P_{norm}	P	AUC	P_{norm}	P
1.0	**59.2**	75.6	**61.2**	71.9	81.4	77.4
2.0	**59.2**	75.6	**61.2**	**72.2**	**81.9**	**77.9**
3.0	**59.2**	**75.7**	**61.2**	72.0	81.7	77.6
4.0	**59.2**	**75.7**	**61.2**	71.9	81.5	77.5

Update interval ablation. Update interval is employed to control the speed of dynamic template update. An appropriate interval is capable of updating the target information to the extent possible while reducing noise. Multiple update intervals are attempted during inference. The results are shown in Table 5-7. It can be seen that 50 is the optimal update interval as 50 achieves superior performance both on TNL2K and LaSOT benchmarks. A larger interval of 100 leads to marginally stronger long-term tracking performance (0.2 AUC increase on LaSOT), while decreasing short-term tracking ability substantially (0.9 AUC drop on TNL2K). Therefore, 50 is utilized as the update interval of dynamic templates.

Table 5-7 Impact of the dynamic template update interval on tracking performance on TNL2K and LaSOT benchmarks

Update intervals	TNL2K			LaSOT		
	AUC	P_{norm}	P	AUC	P_{norm}	P
100	58.7	75.4	60.5	**72.9**	**83.0**	**79.7**
50	**59.6**	**76.6**	**61.9**	72.7	82.8	79.3
25	59.5	76.4	**61.9**	72.2	82.5	78.8
1	56.0	70.3	54.8	66.4	73.4	68.7

Pruning location ablation. Another set of ablation experiments is conducted to determine the pruning locations of dynamic templates and search region. Three possible position matching options are designed in Table 5-8 for alternating token pruning. SRP and DTP blocks are respectively responsible for trimming the search region and dynamic templates. The results indicate that the first combination ([4, 7, 10] for SRP locations and [3, 6, 9] for DTP locations) integrates the advantages of tracking speed and performance. This option achieves the highest AUC 72.7% on LaSOT while still running in real time (82.1 FPS). Thus, SRP blocks are located in the 4^{th}, 7^{th}, and 10^{th} layers of the backbone. DTP blocks are distributed in the 3^{rd}, 6^{th} and 9^{th} layers. Figure 5-5 demonstrates the arrangement for three types of blocks in ATPTrack.

Table 5-8 Impact of different alternating pruning locations on tracking results, computational costs (MACs) and running speeds (FPS)

SRP block locations	DTP block locations	TNL2K			LaSOT			MACs (G)	FPS
		AUC	P_{norm}	P	AUC	P_{norm}	P		
[4, 7, 10]	[3, 6, 9]	59.6	76.6	61.9	**72.7**	**82.8**	**79.3**	75.5	82.1
[3, 6, 9]	[4, 7, 10]	59.2	76.3	61.4	71.9	81.9	77.9	**74.1**	**84.5**
[4, 7, 10]	[5, 8, 11]	**59.8**	**76.9**	**62.4**	72.3	82.5	78.6	78.1	79.9

Note: All speeds are measured on one RTX A6000 GPU.

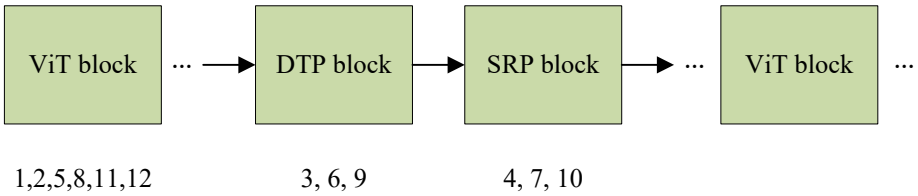


Figure 5-5 Block arrangement for ATPTrack

Attention weights visualization. Figure 5-6 depicts the attention weights of six encoder layers for the proposed ATPTrack. Each red bounding box emphasizes the target region of the search region. It can be seen that shallow attention layers mainly replicate feature maps with limited target-related features. Deep layers incorporate complicated semantic information and generate discriminative representations of the target. It also shows that the final layer highlights the edge information of the target area, e.g., corner points. This suggests that ATPTrack can accurately perceive the target region.

Token pruning visualization. In this section, the alternating token pruning process for two dynamic templates and the search area is visualized in Figure 5-7. Dyn1 and Dyn2 denote two dynamic templates in each case. Targets are highlighted in red boxes. Background tokens are progressively discarded from dynamic templates and the search region, resulting in discriminative target regions and improved running speed. It is also worth mentioning that a small portion of target regions of dynamic templates are identified by the network as redundant areas. This is similar to mask image modeling which occludes partial target and background contents, thereby enhancing the global information inference capability of the tracking network.

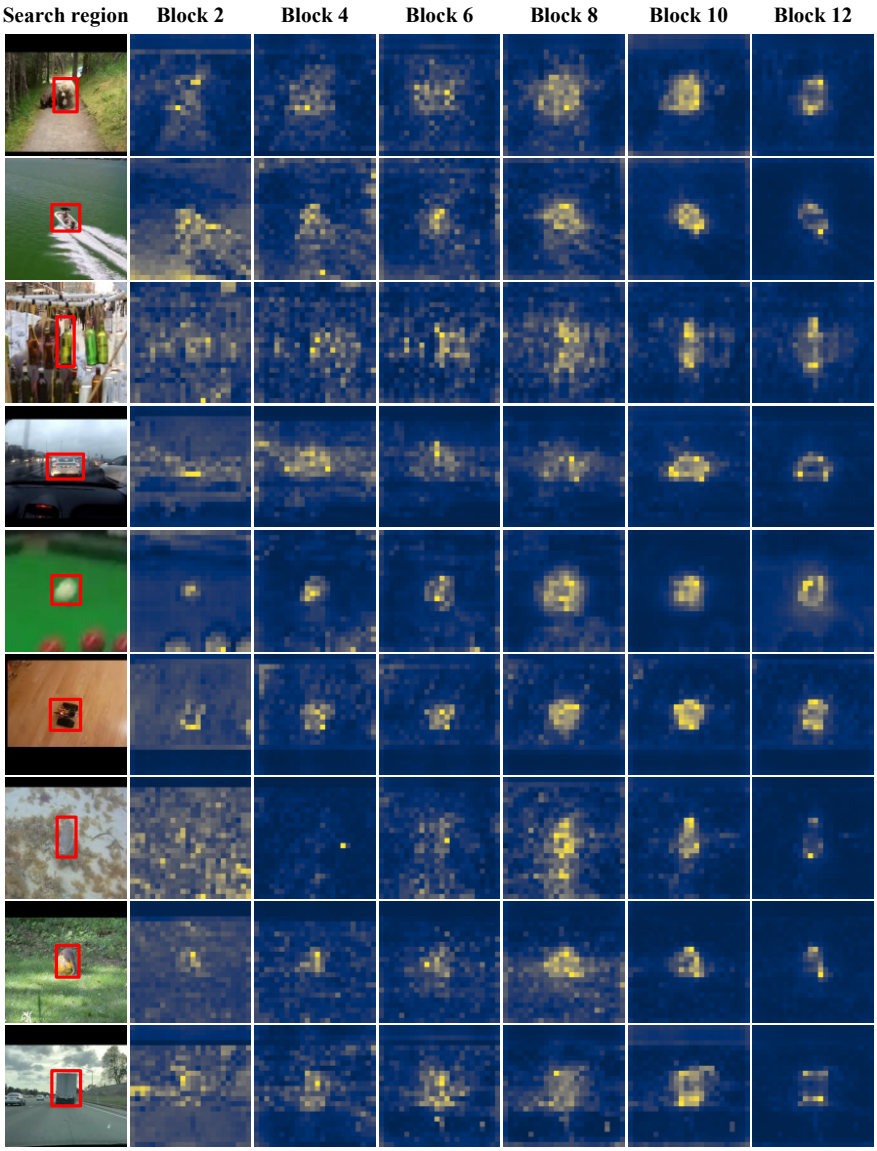


Figure 5-6 Visualization of attention weights on LaSOT

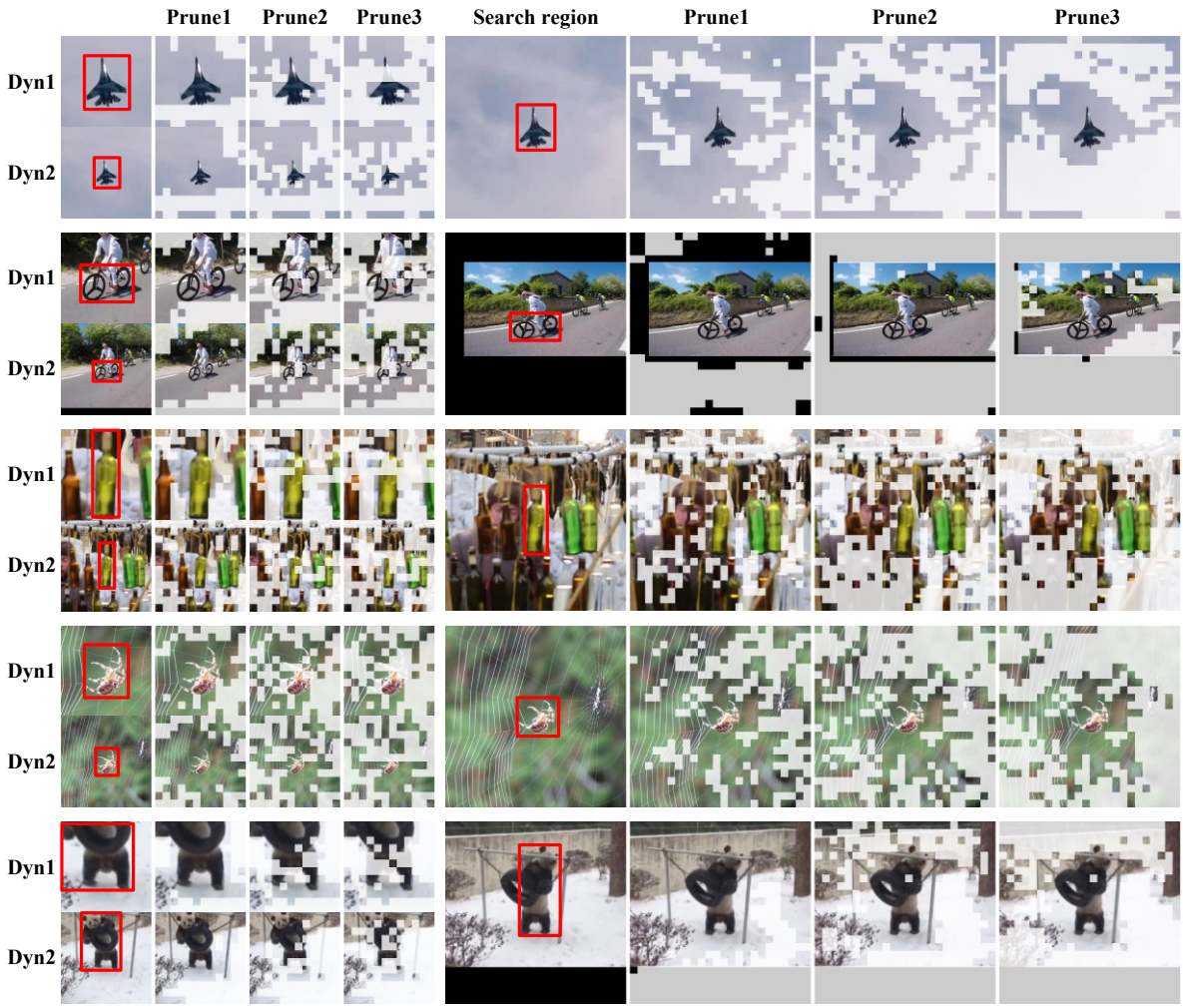


Figure 5-7 Visualization of token pruning for dynamic templates and search area

Comprehensive attribute analysis. Figure 5-8 conducts the comprehensive analysis of 14 attributes on LaSOT for the proposed ATPTrack. It is obvious that our ATPTrack outperforms current state-of-the-art tracking methods by a large margin. For instance, ATPTrack achieves an AUC of 70.2%, outperforming SeqTrack by 2.5%.

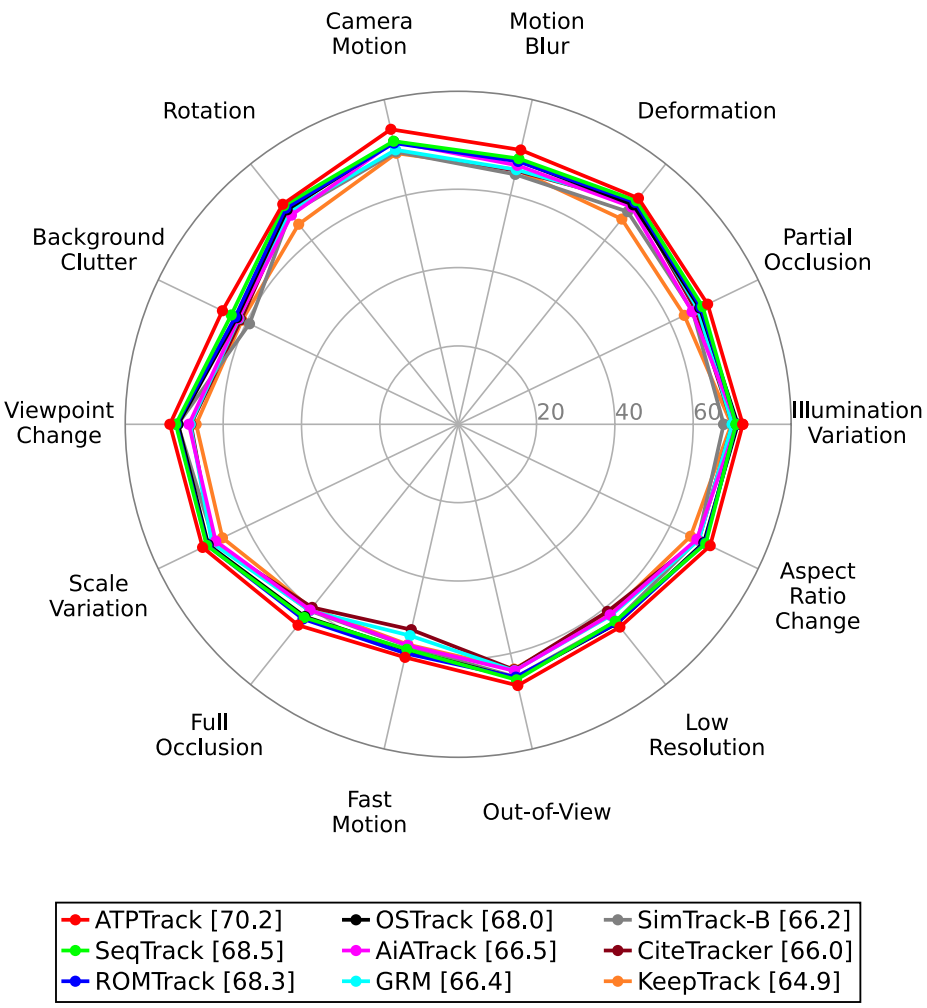


Figure 5-8 Comprehensive attribute analysis for ATPTrack

Separate attribute analysis. Separate attribute analysis is performed on LaSOT benchmark to evaluate the performance of ATPTrack on 14 attributes. The success rate plots are described in Figure 5-9. It can be seen that ATPTrack surpasses current top-performing tracking algorithms in terms of the success rate (AUC score) for all attributes. Background clutter is the focus in this chapter. Our ATPTrack obtains a success rate of 66.8% in background clutter, exceeding SeqTrack by 3.9%. ATPTrack also performs well on camera motion, full occlusion, motion blur, etc.

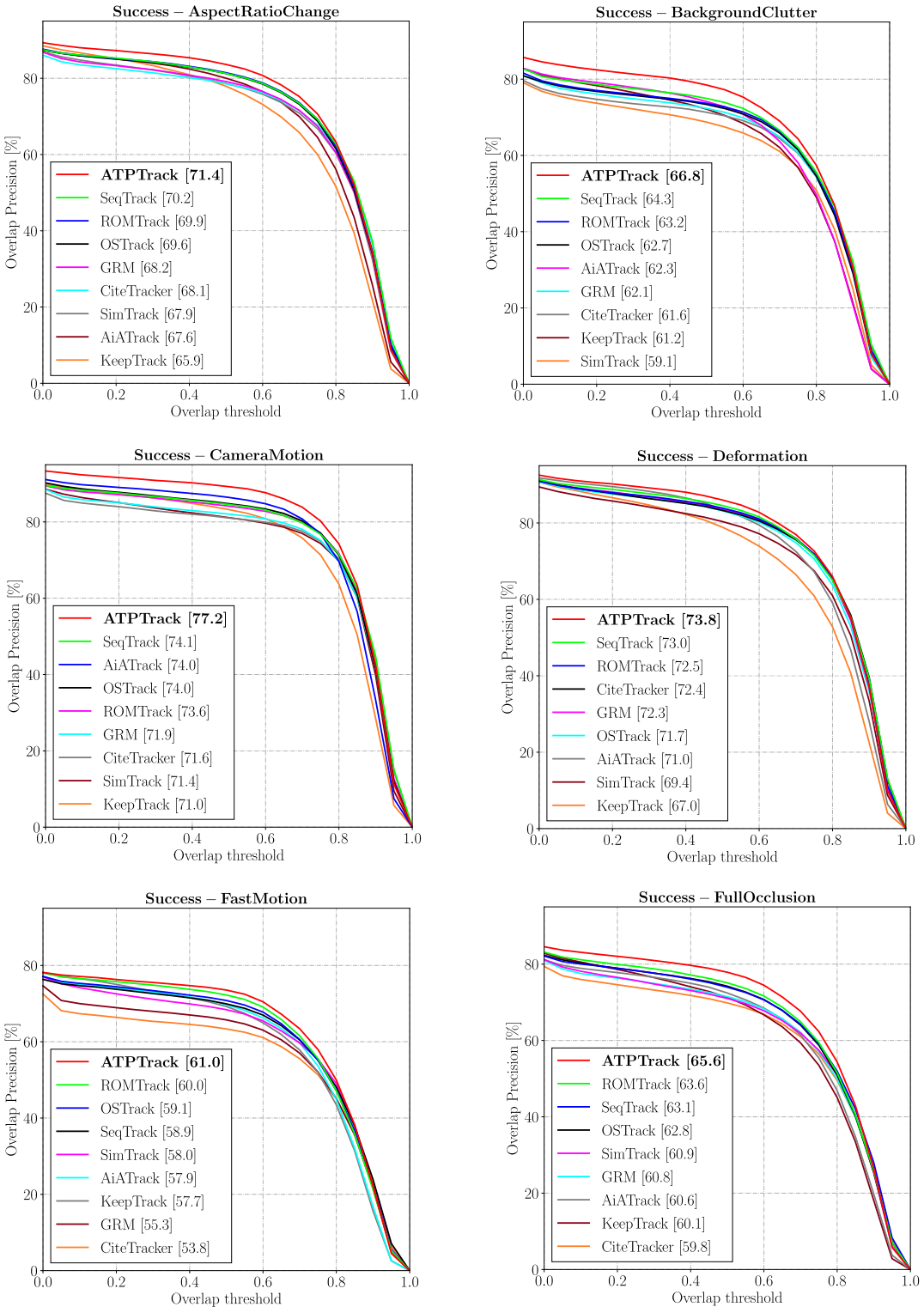


Figure 5-9 Attribute-level comparisons in terms of success rate

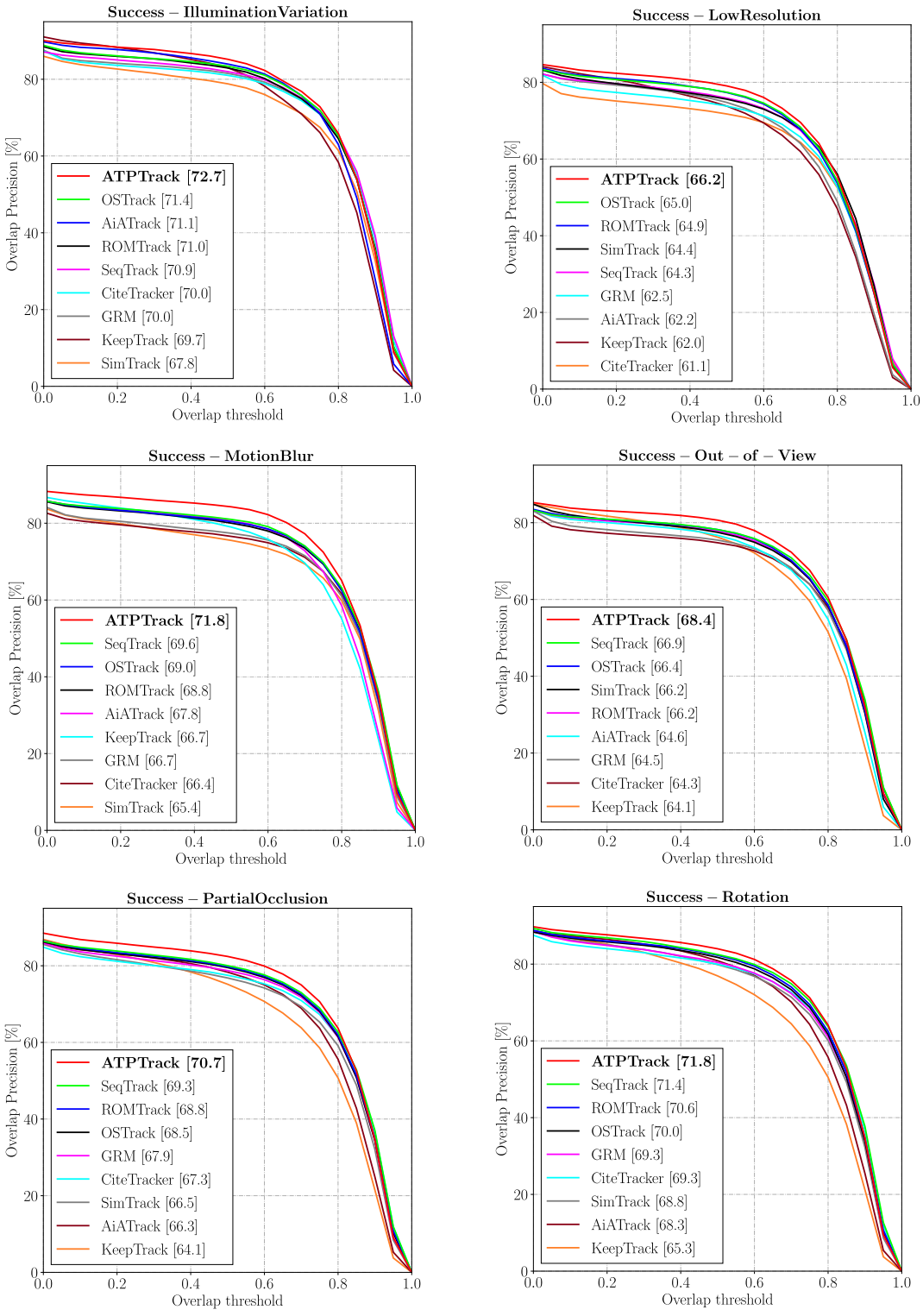


Figure 5-9 Attribute-level comparisons in terms of success rate (continued)

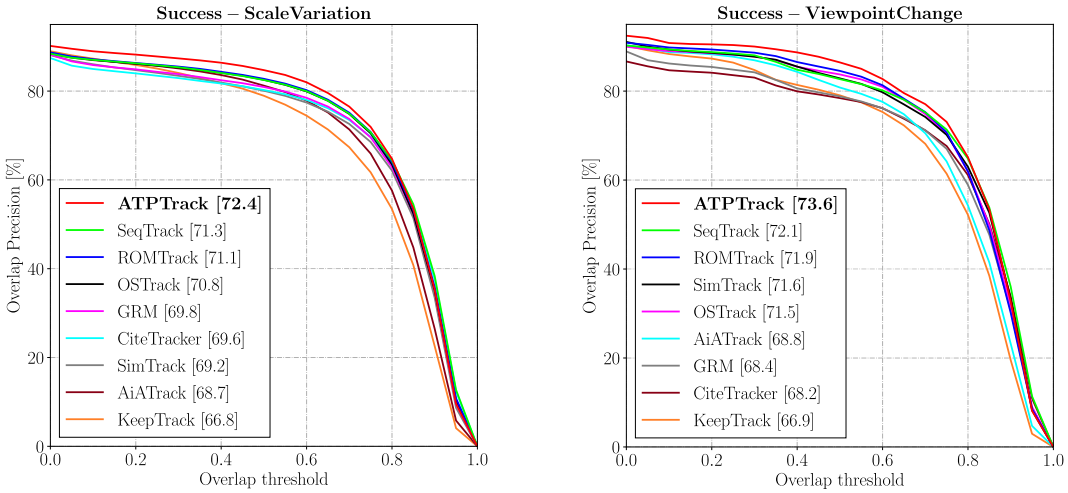


Figure 5-9 Attribute-level comparisons in terms of success rate (continued)

The precision and normalized precision plots are demonstrated in Figures 5-10 and 5-11. The proposed ATPTrack achieves a precision of 72.7% and a normalized precision of 75.7% with regard to background clutter, substantially outperforming the current prevailing approaches on LaSOT dataset. This suggests that alternating pruning of dynamic templates and the search region is capable of highlighting the target and therefore distinguishing the distractors, thereby better coping with background clutter.

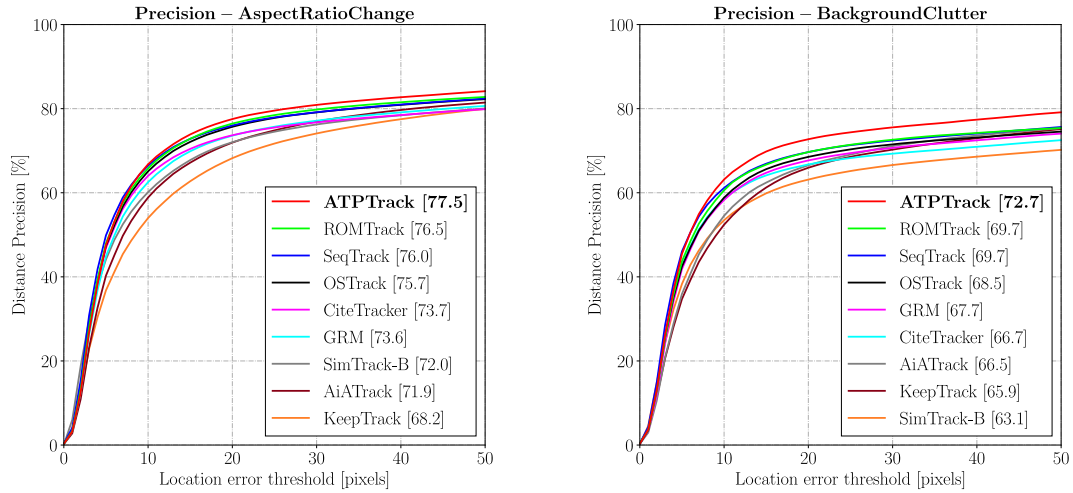


Figure 5-10 Attribute-level comparisons in terms of precision

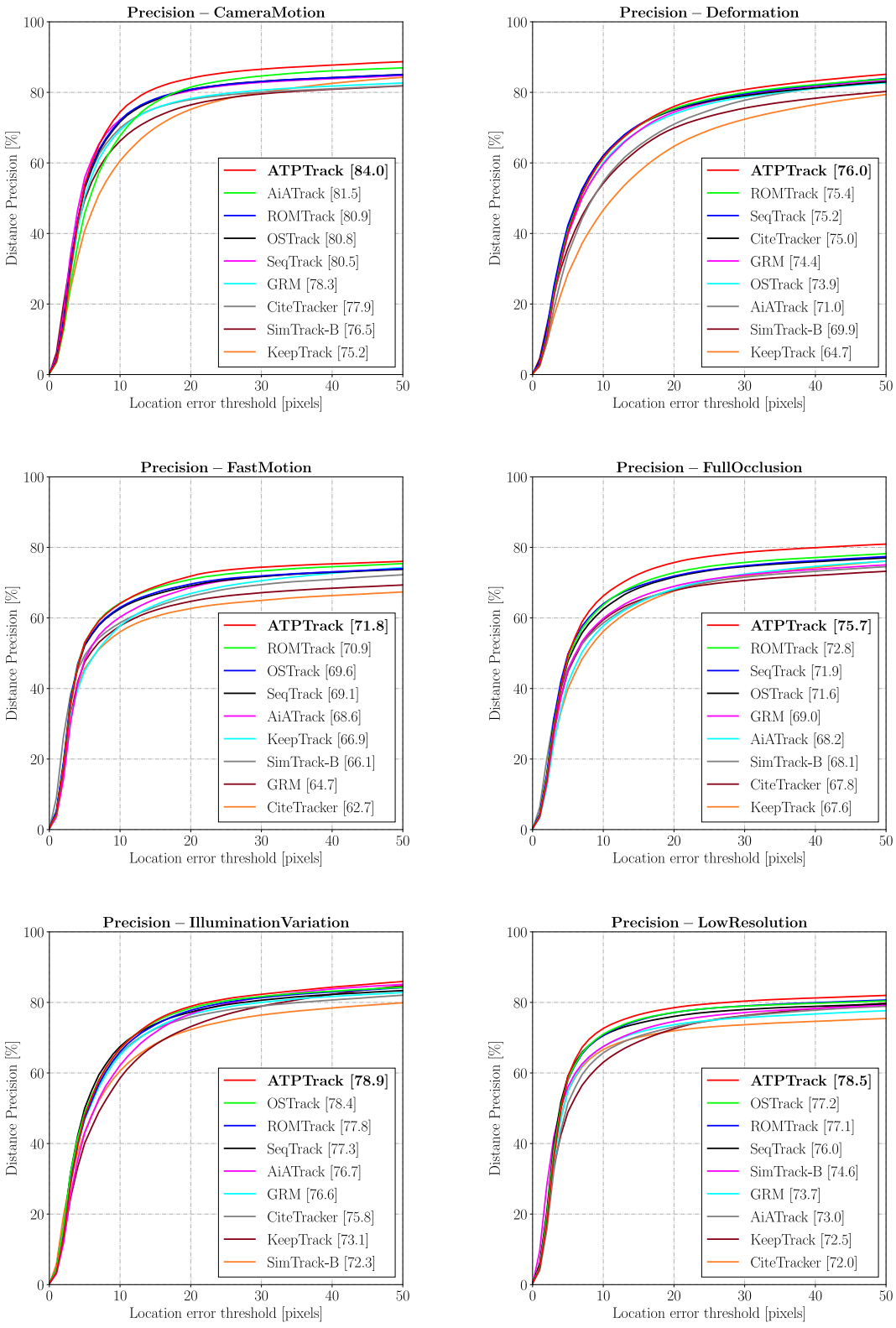


Figure 5-10 Attribute-level comparisons in terms of precision (continued)

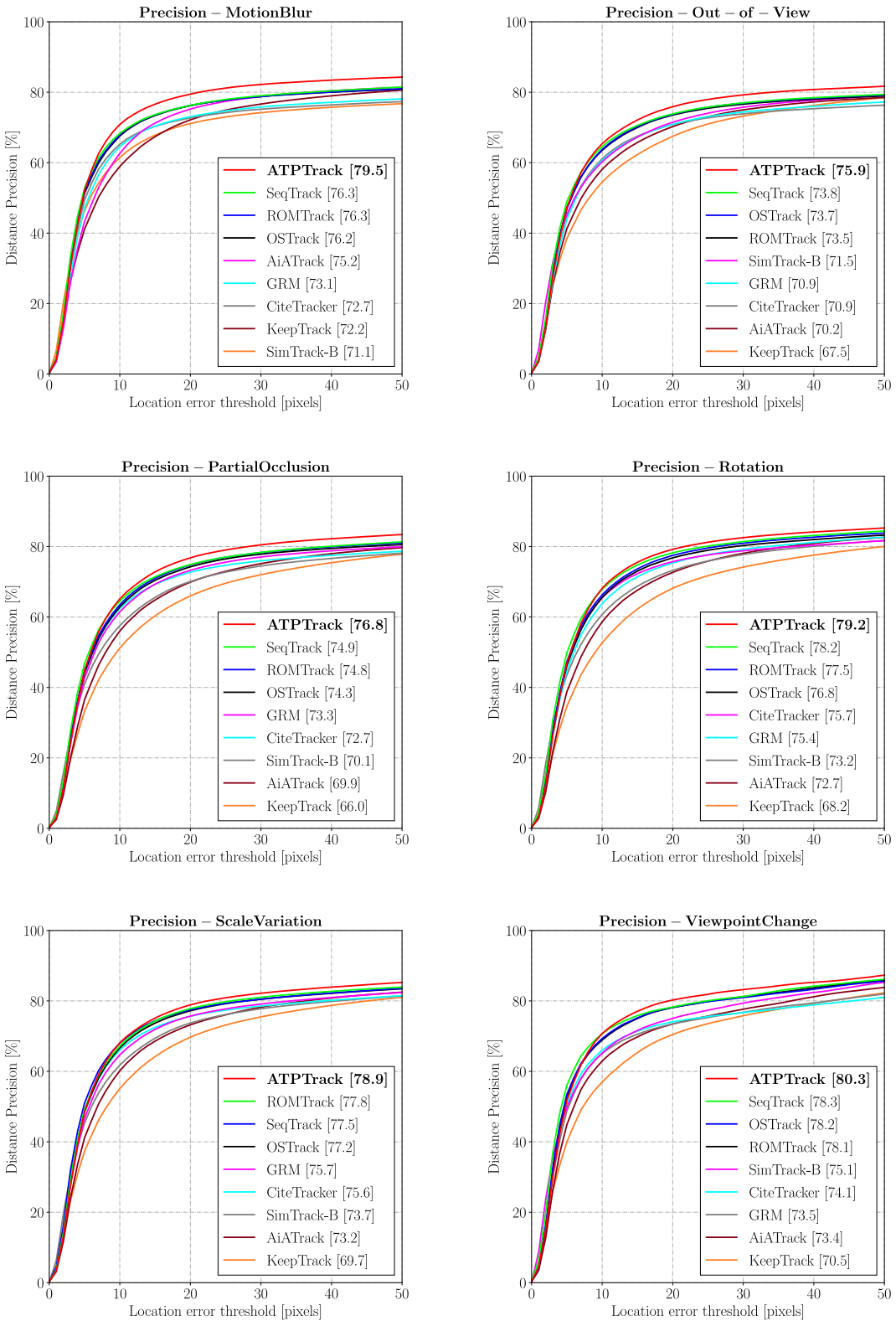


Figure 5-10 Attribute-level comparisons in terms of precision (continued)

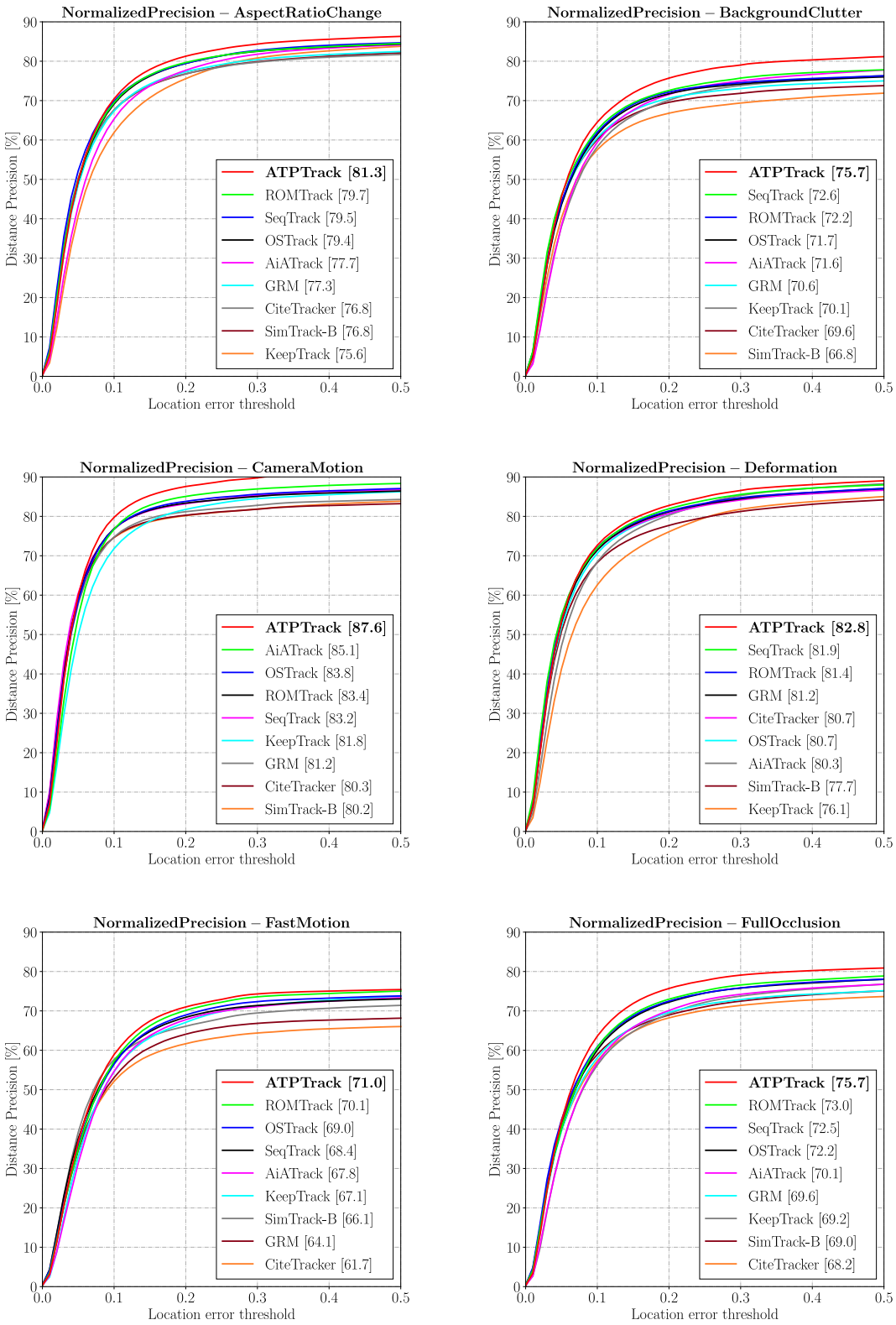


Figure 5-11 Attribute-level comparisons in terms of normalized precision

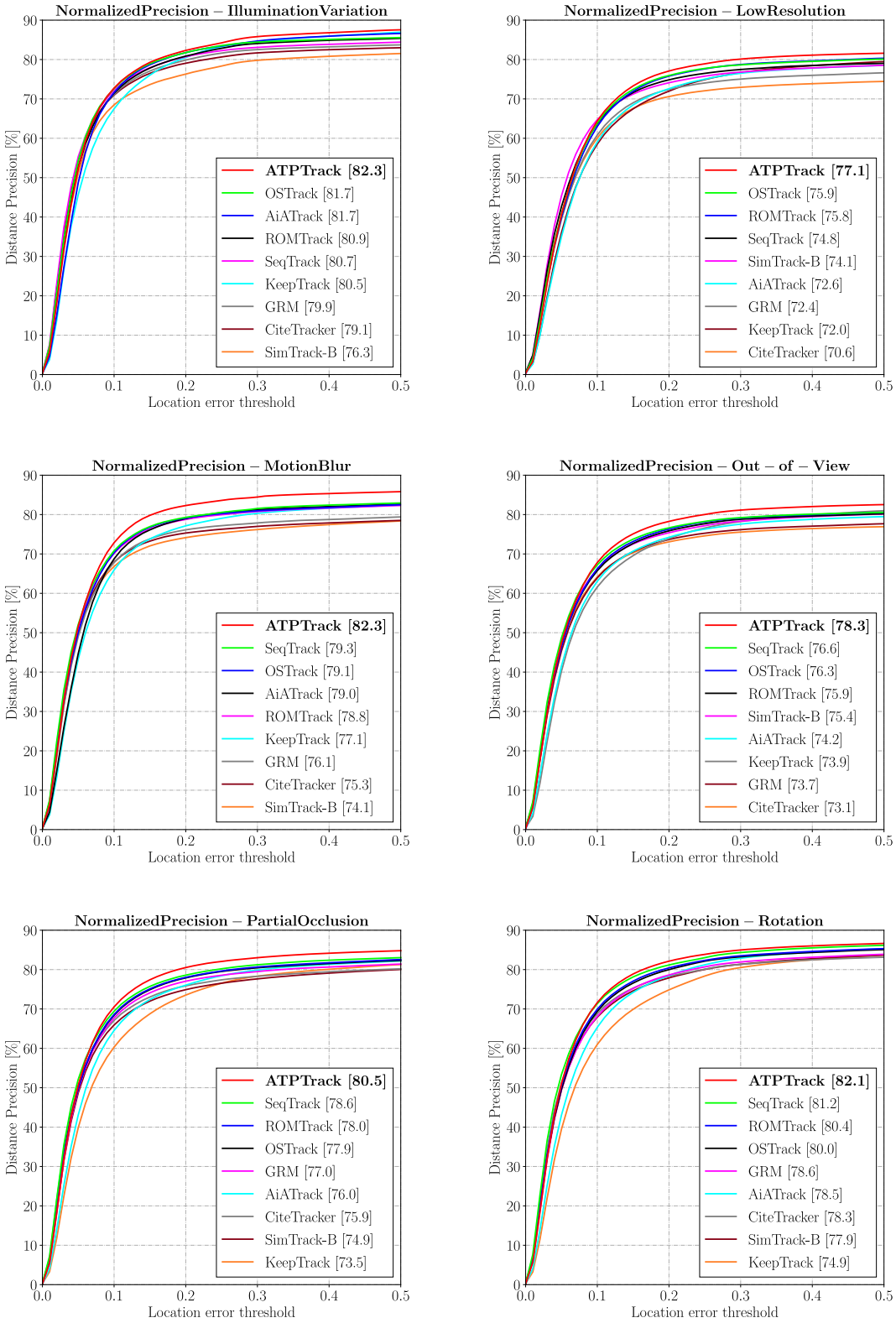


Figure 5-11 Attribute-level comparisons in terms of normalized precision (continued)

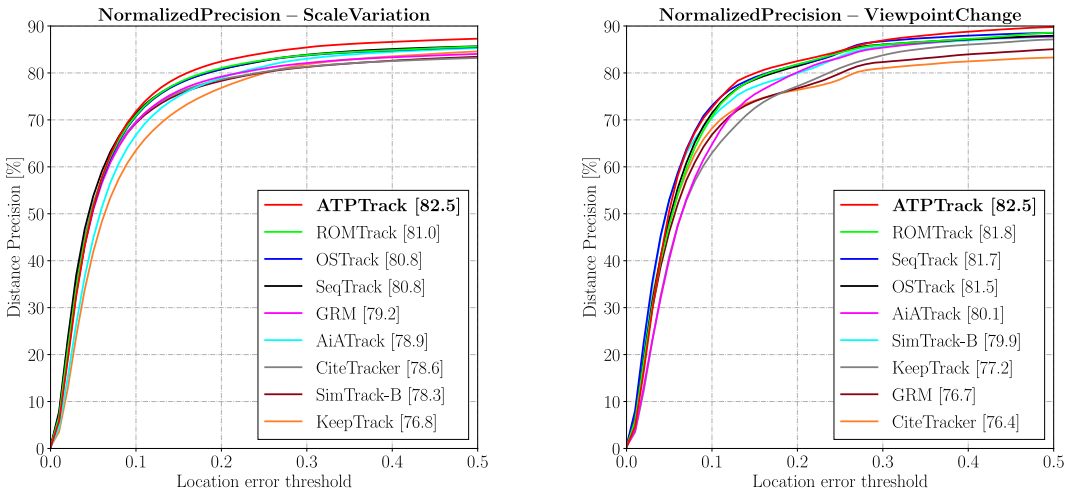


Figure 5-11 Attribute-level comparisons in terms of normalized precision (continued)

Box comparison among top trackers. As described in Figure 5-12, the bounding boxes generated by state-of-the-art tracking approaches are visualized for more explicit result comparison. The video frames are obtained from LaSOT benchmark. The results of our proposed tracker ATPTrack are shown in red bounding boxes. Green bounding boxes denote ground truth labels. Three leading methods ROMTrack, OSTrack and SeqTrack are represented by blue, yellow and purple bounding boxes, respectively. It is obvious that our predicted bounding boxes are the most similar to the ground truth bounding boxes in terms of the box location and size in the frames. The selected sequences contain a variety of similar distractors and occlusions. It suggests that ATPTrack can track the target accurately in the presence of background clutter and occlusion.

Figure 5-12 Bounding box comparison among top trackers

5.4 Summary

This chapter proposes a novel tracking approach termed as ATPTrack for addressing the problem of background clutter in real-world tracking scenarios. ATPTrack develops an alternating token trimming algorithm to emphasize the target region and to reduce the computational complexity. Extensive experiments are performed to assess the overall tracking performance of ATPTrack and its capability for handling diverse tracking challenges.

Chapter 6 Applications in real world scenarios of proposed methods

6.1 Application of MIMTracking

A series of videos are captured from a 70mai M300 dash camera with the frame rate of 30 frames per second (fps) and an iPhone 13 with the frame rate of 30 fps to validate the tracking performance of the presented long-term tracker MIMTracking. The tracking results (bounding boxes) of the selected frames in 8 representative videos are visualized in Figures 6-1, 6-2, 6-3, 6-4, 6-5, 6-6, 6-7 and 6-8. The target of the first frame in each video is labelled with a bounding box. Then the target is tracked by our MIMTracking in remaining frames. Target categories include 3D printed items, vehicles, book, and a drink box. These videos contain more than 3000 frames to be suitable for long-term tracking. There are multiple tracking challenges in these videos, e.g., scale variation, low resolution, occlusion, viewpoint change, rotation, illumination variation, etc. Therefore, these videos can be utilized to assess our long-term tracker MIMTracking.

It can be seen from the results that MIMTracking is capable of coping with extensive tracking challenges. From Figure 6-3, the drink box disappears in frame 210 and reappears in frame 250. The proposed tracker loses the drink box in frame 210 and relocates it in frame 250. This suggests that our tracking algorithm can recover from tracking failures automatically. Automatic recovery from tracking failures is an important evaluation metric for long-term tracking methods. Our MIMTracking is also capable of tracking tiny targets such as vehicles in dashcam videos. This allows it to be applied to autonomous driving. Furthermore, MIMTracking performs well in challenging scenarios, e.g., target rotation, scale variation, occlusion, etc.

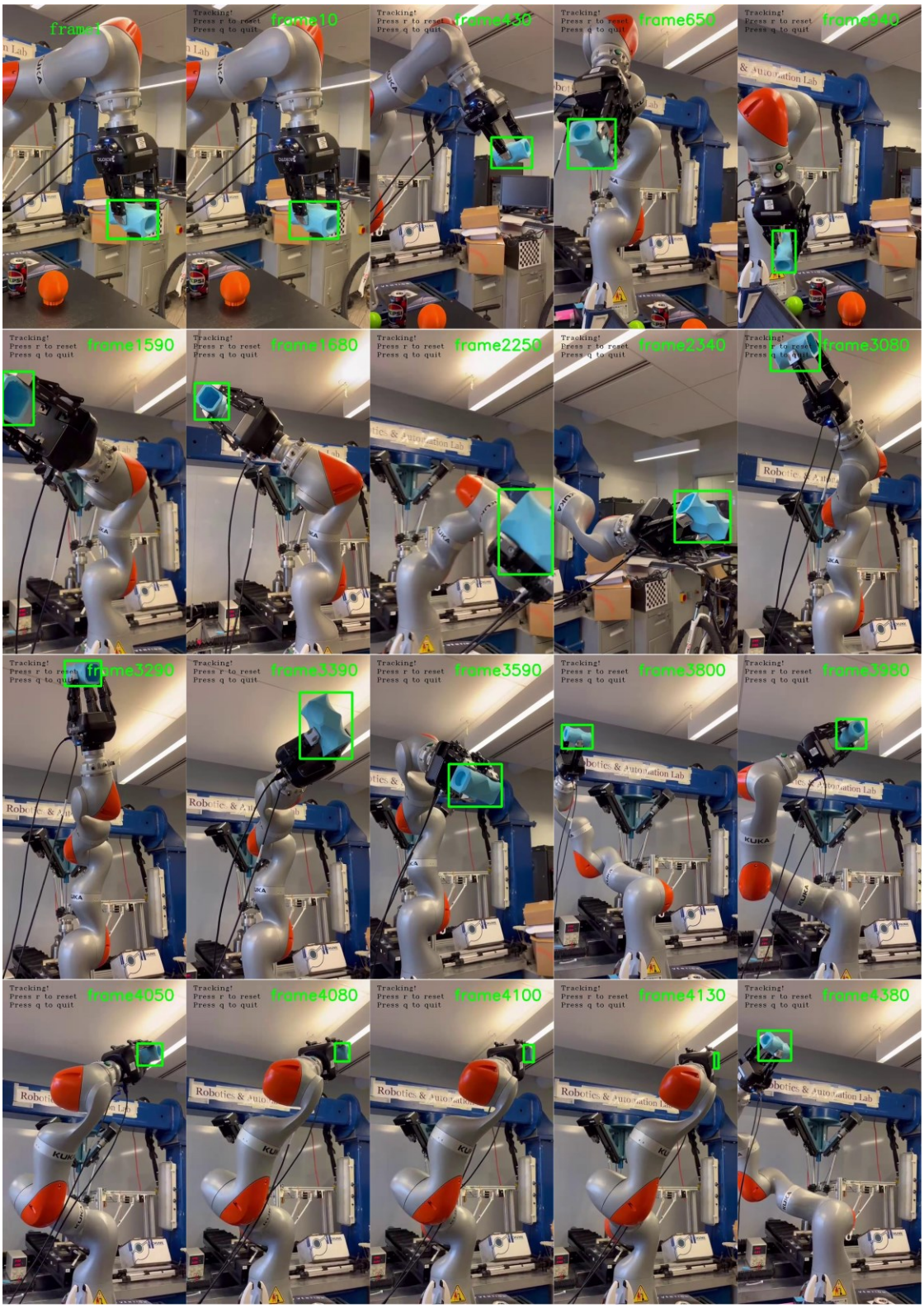


Figure 6-1 Tracking results for a blue object

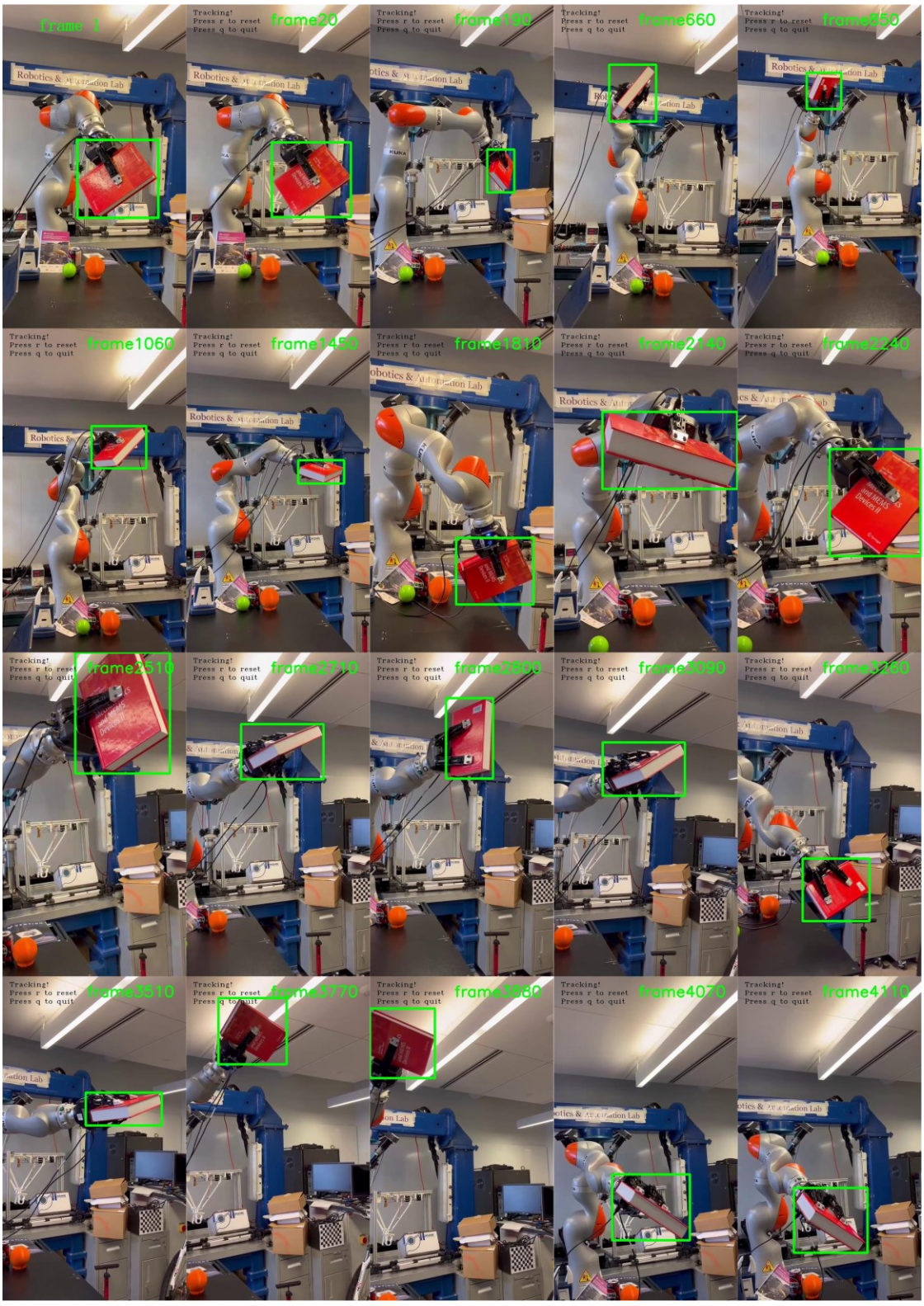


Figure 6-2 Tracking results for a book

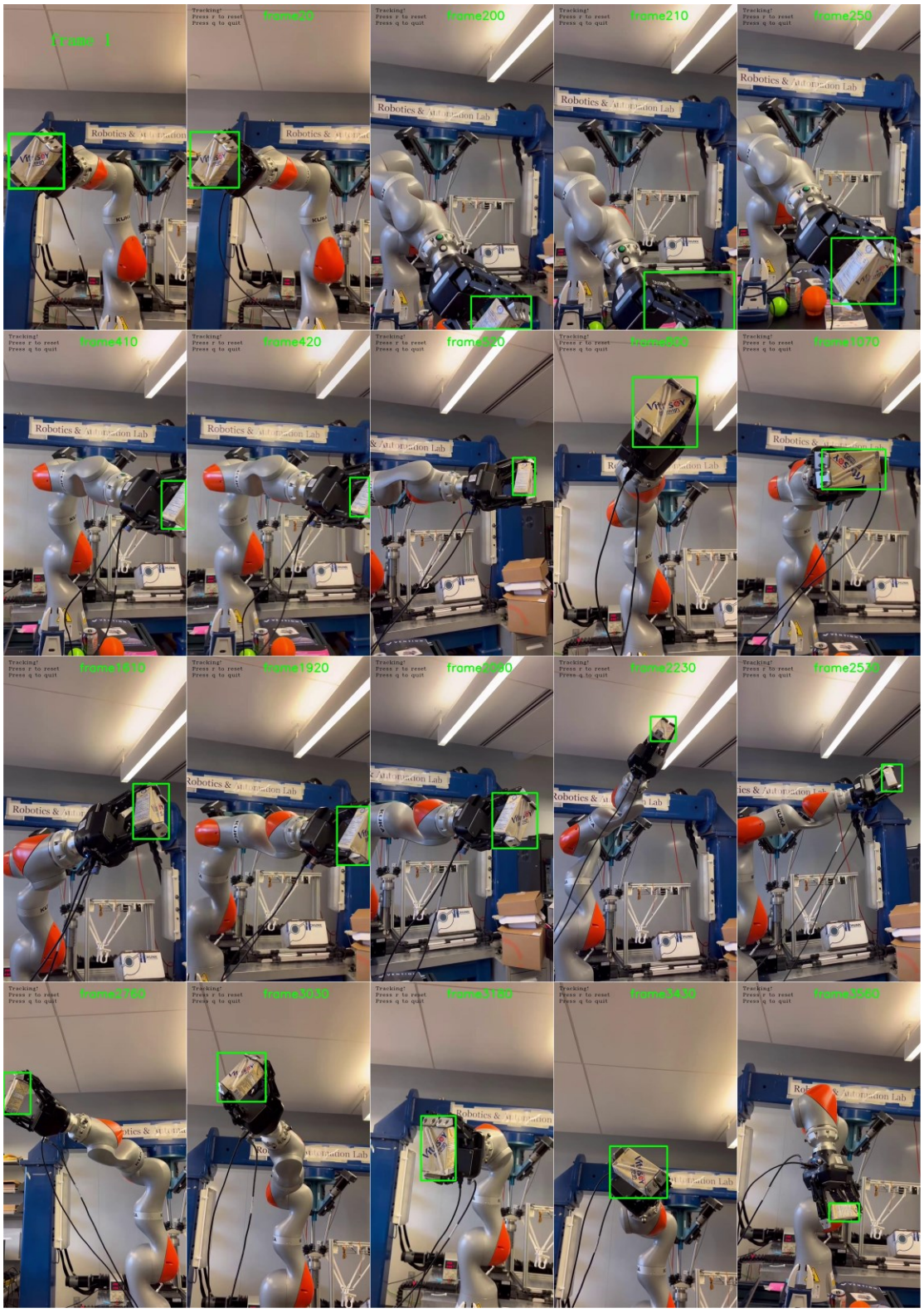


Figure 6-3 Tracking results for a drink box

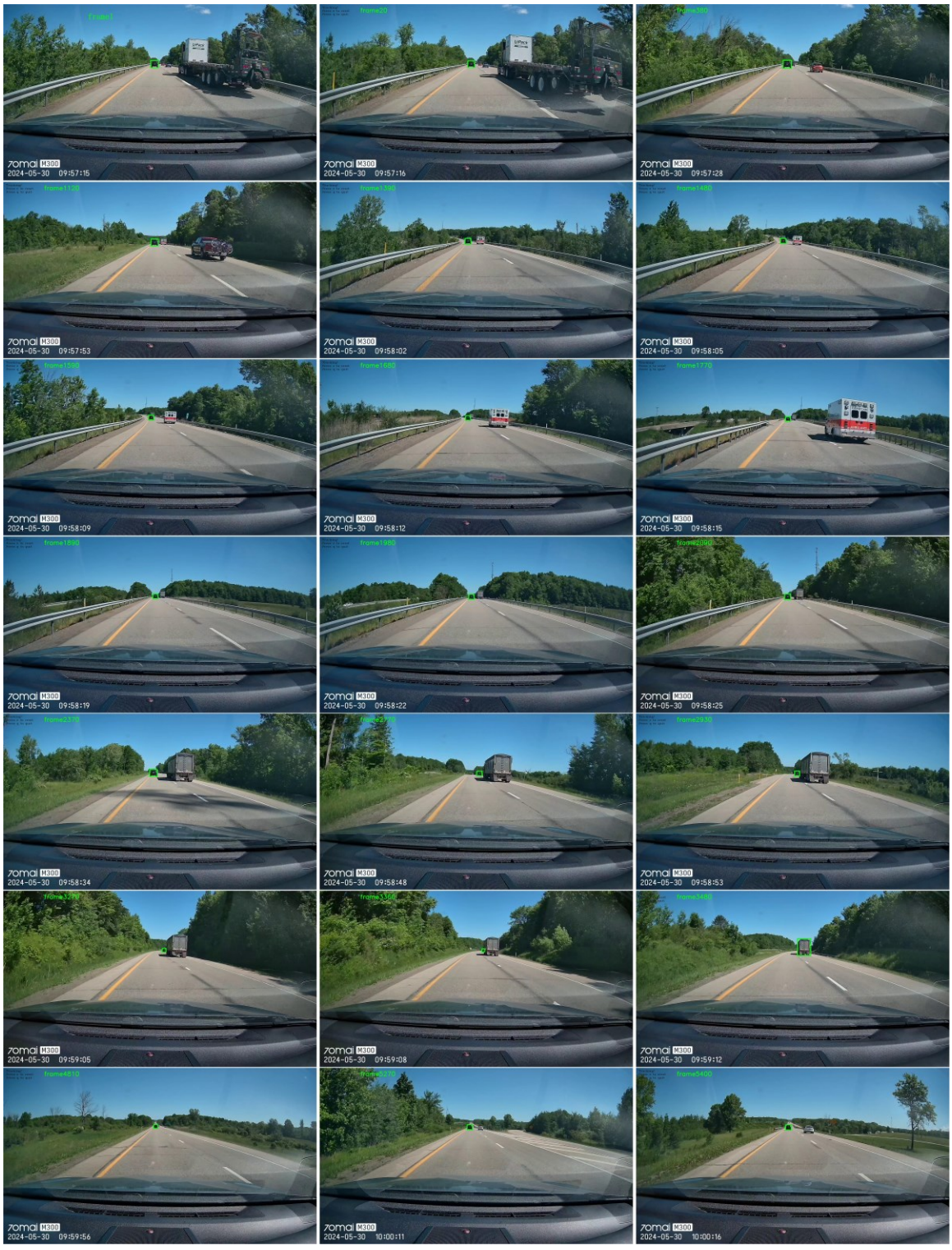


Figure 6-4 Tracking results for vehicle 1

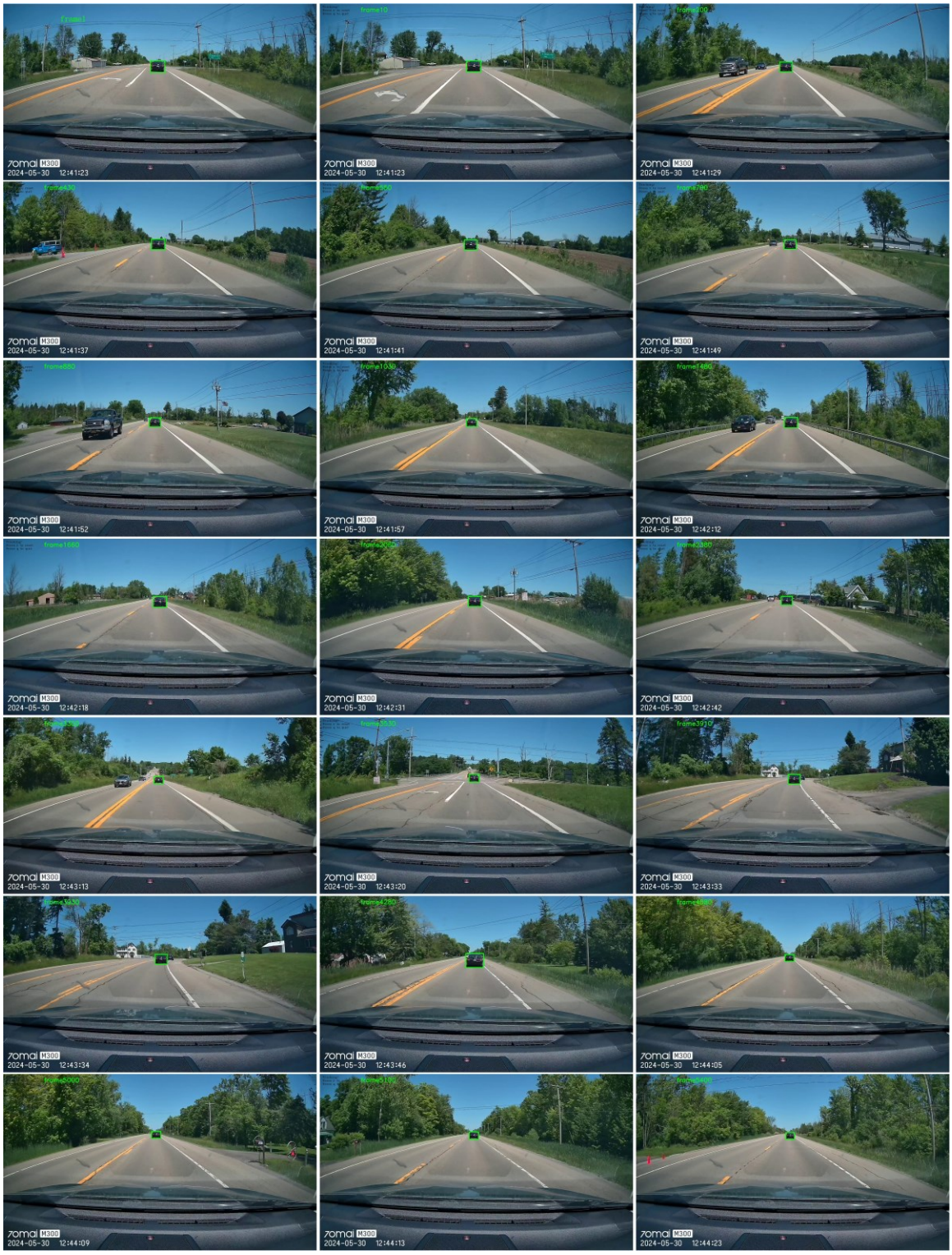


Figure 6-5 Tracking results for vehicle 2

Figure 6-6 Tracking results for vehicle 3

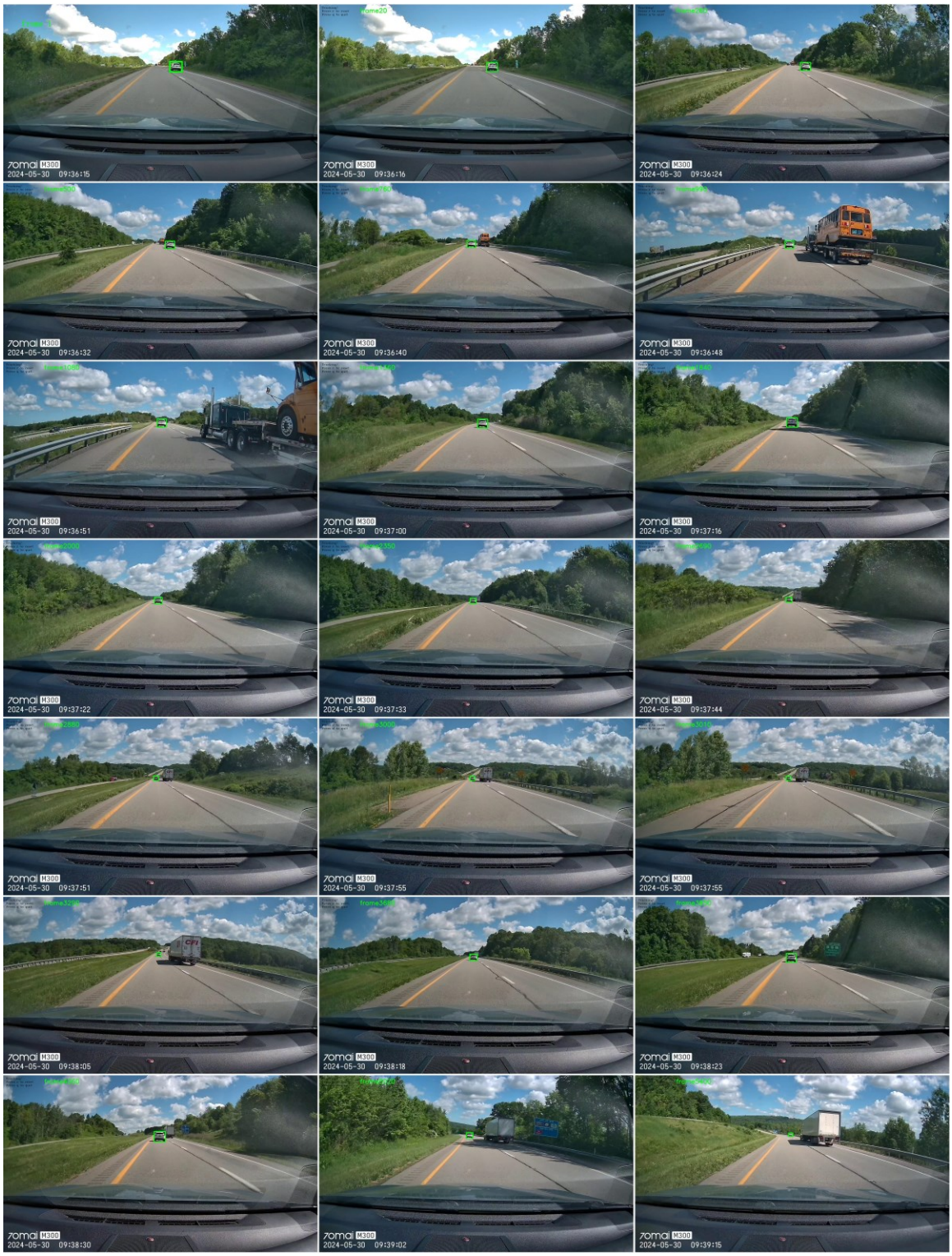


Figure 6-7 Tracking results for vehicle 4

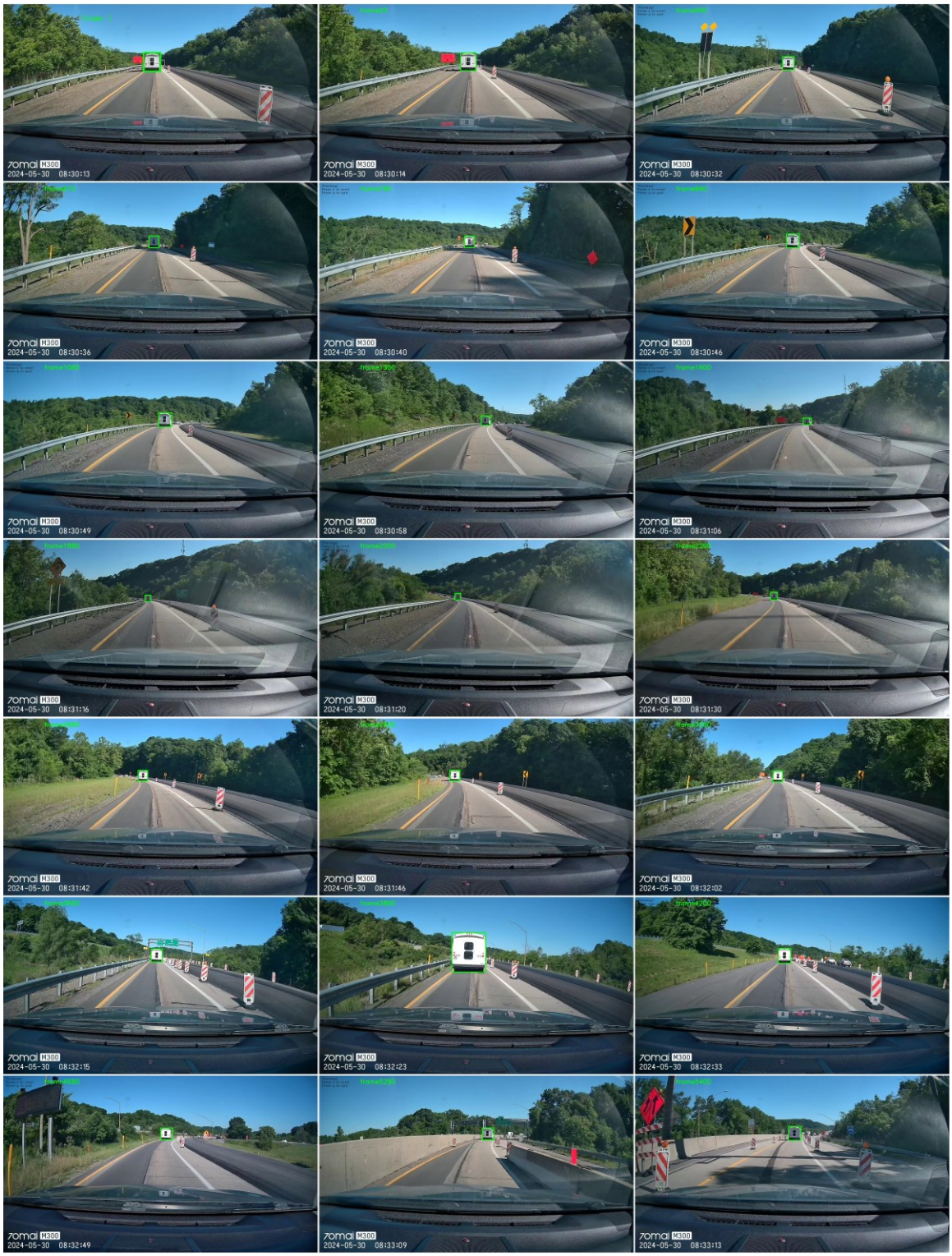


Figure 6-8 Tracking results for vehicle 5

6.2 Application of ATPTrack

ATPTrack is evaluated on multiple challenging video sequences captured from extensive tracking scenarios. The camera used to capture videos is an iPhone 13 with the frame rate of 30 fps. 9 videos are exploited to test the proposed ATPTrack. The results are visualized in Figures 6-9 to 6-17. The first frame of each video is annotated with a bounding box. The target in the bounding box is tracked by the proposed tracker ATPTrack. There are a wide range of targets, including 3D printed object, shopping cart, green ball, human body, human head, laser tracker device, orange ball, peach, and tool case. The 9 videos involve multiple common tracking challenges, i.e., background clutter, full occlusion, partial occlusion, out-of-view, scale variation, camera motion, rotation, etc.

ATPTrack can effectively tackle the problem of background clutter. In Figure 6-10, there are several identical shopping carts in frames 850, 870, 1280, 1660, 1790 and 1990. The target is occluded partially when background distractors appear. This makes it exceptionally demanding to locate the target accurately. However, ATPTrack is still able to distinguish the target from similar distractors in these challenging frames. Additionally, ATPTrack is capable of handling dense distractors under background clutter. In Figures 6-12 and 6-13, ATPTrack is employed to track the human body and human head in densely populated supermarkets. It is obvious that ATPTrack can locate the human body and human head precisely in crowded scenarios, as demonstrated in frames 1190, 1950, 2400, 2470 of Figure 6-12 and frames 910, 1990, 2020, 2100, 2930 of Figure 6-13. Moreover, ATPTrack performs effectively as well in other tracking challenges, e.g., occlusion, out-of-view, scale change, etc.

Figure 6-9 Tracking results for blue object 2

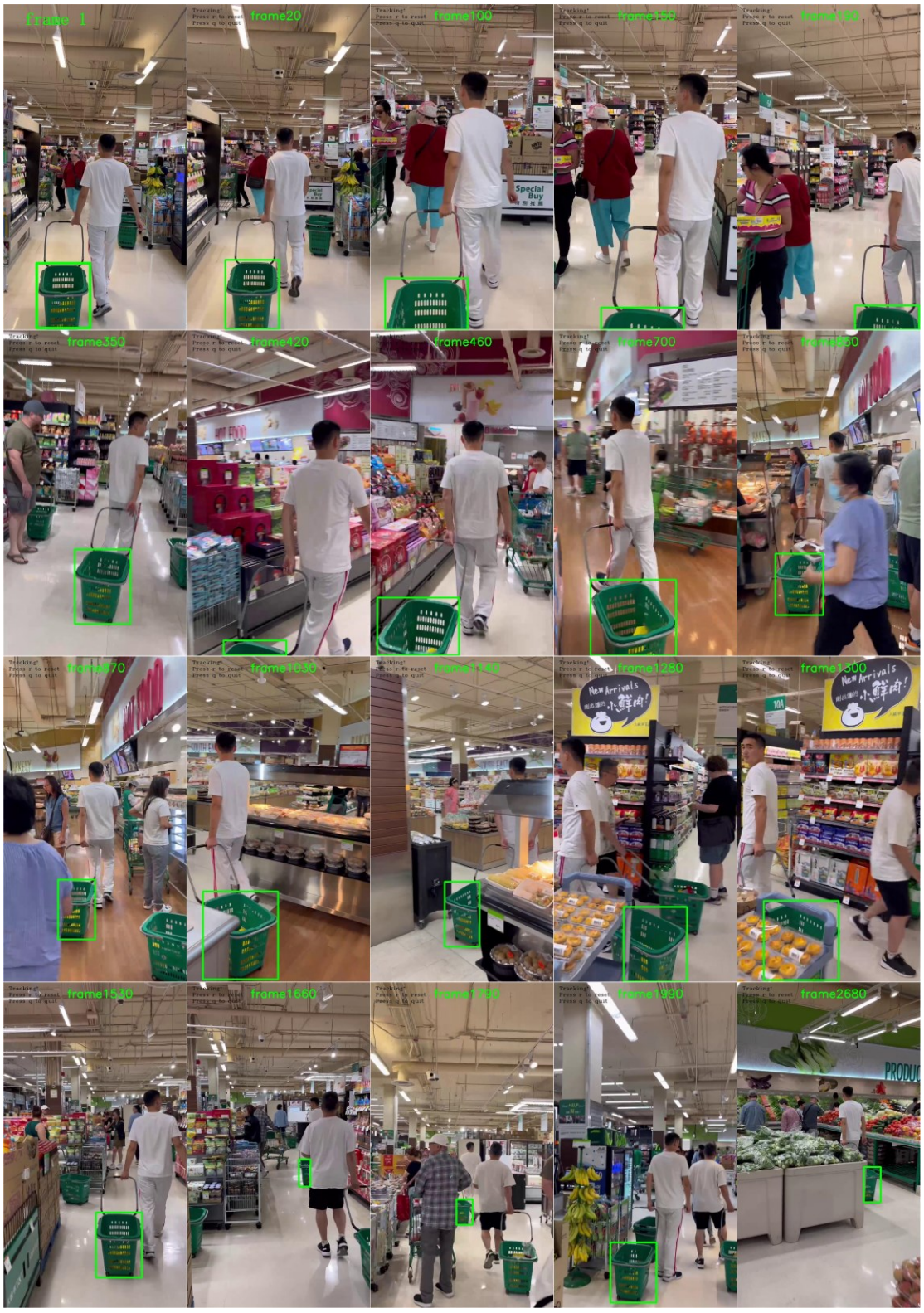


Figure 6-10 Tracking results for a shopping cart

Figure 6-11 Tracking results for a green ball

Figure 6-12 Tracking results for the human body

Figure 6-13 Tracking results for the human head

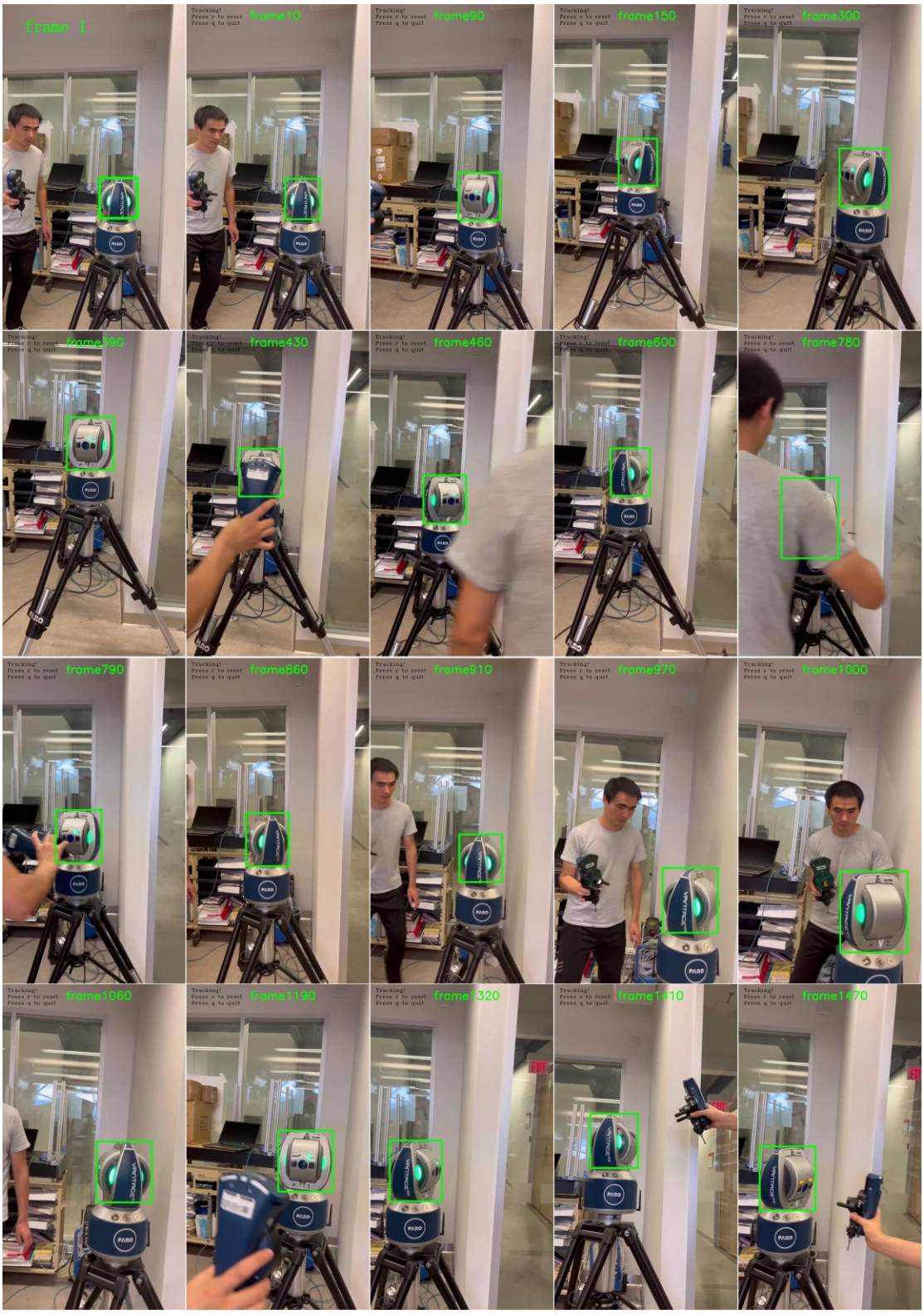


Figure 6-14 Tracking results for a FARO laser tracker

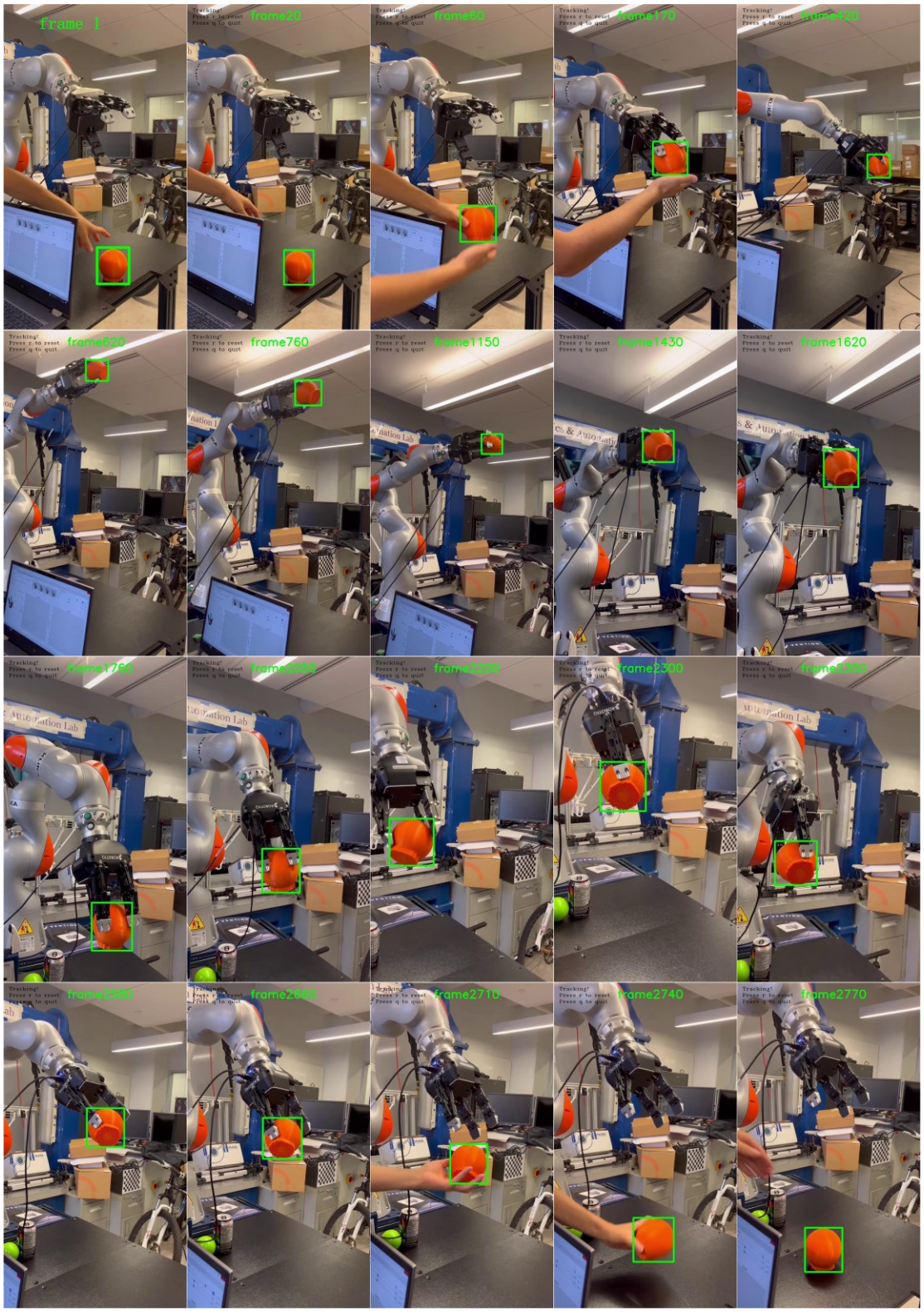


Figure 6-15 Tracking results for an orange ball

Figure 6-16 Tracking results for a peach

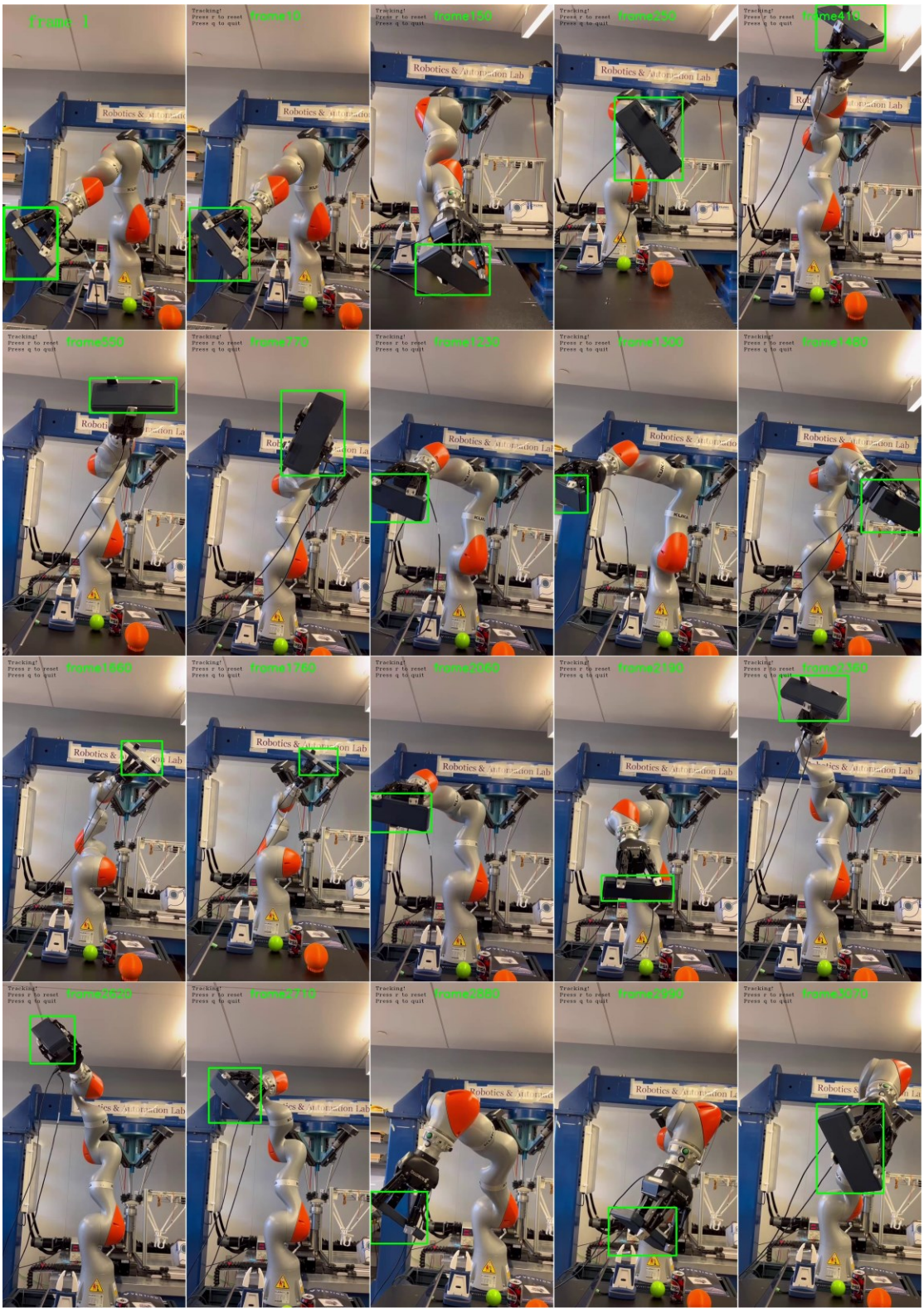


Figure 6-17 Tracking results for a tool case

6.3 Summary

In this chapter, the proposed tracking methods MIMTracking and ATPTrack are evaluated in real world tracking scenarios. A number of videos are captured from a dash camera and a smart phone at a frame rate of 30 fps. These videos contain challenging scenes for validating the long-term tracking capability of MIMTracking and the ability of ATPTrack for dealing with background clutter. The tracking results are visualized with bounding boxes for a more intuitive representation.

Chapter 7 Conclusion and future work

7.1 Conclusions

In the dissertation, three methods are developed for adaptive short-term tracking, long-term tracking and object tracking coping with background clutter, respectively. Three trackers are assessed on widely utilized tracking benchmarks for comparison with state-of-the-art trackers. Additionally, the latter two methods are evaluated on a series of videos recorded by a mobile phone for validating their performance in real-world applications.

(A) TGLC is proposed to fully combine global-local feature representations as well as channel information for short-term tracking. This end-to-end tracking method is capable of accurately locating the target and predicting immensely appropriate bounding boxes according to the dimension and shape of the target. Extensive experiments are conducted on five widely recognized datasets, i.e., GOT-10k, TrackingNet, LaSOT, OTB100 and UAV123. The results depict that TGLC achieves competitive tracking performance compared with state-of-the-art trackers while still running in real-time. Visualization of the tracking results on LaSOT further demonstrates the capability of the proposed tracking method to cope with tracking challenges, e.g., illumination variation and deformation of the target.

(B) A neat end-to-end tracker MIMTracking is proposed for long-term object tracking. Unlike current MIM-based trackers that integrate masked image modeling into the pre-training process of the tracking models, a brand-new idea is developed by incorporating MIM directly into the training process of the tracking model. Specifically, only partial search tokens are fed into the encoder during training, which reduces the redundancy of the input sequences as well as yielding more detailed representations. Extensive comparison

and ablation experiments are performed on numerous benchmarks. The results demonstrate that MIMTracking achieves leading tracking performance while still maintaining real-time speed. This suggests that MIM can better stimulate the potential of ViT for long-term target tracking.

(C) Background clutter acts as a primary factor of tracking failures. ATPTrack is developed to deal with background clutter. Dynamic templates update the target representations, but inevitably introduce noise and increase computational complexity. Therefore, an alternating token pruning method is proposed in ATPTrack to trim redundant tokens for dynamic templates and the search region. After progressive pruning, discriminative target representations are gradually highlighted with decreased background noise and improved model inference speed. This contributes to eliminating distractors under background clutter. Extensive experimental results demonstrate that the proposed ATPTrack outperforms current state-of-the-art tracking methods on multiple tracking datasets with significantly reduced computational costs.

7.2 Contributions of the dissertation

7.2.1 TGLC for the first research objective

In this work, a novel tracking algorithm (TGLC) is developed to tackle the problems mentioned in adaptive visual tracking. This tracking method is composed of a shared backbone for feature extraction, a novel feature fusion module and a key point prediction head. The proposed feature fusion module consists of a number of global-local self-attention (GLSA) and global-local cross-attention (GLCA) modules. Each GLSA or GLCA contains a self-attention or cross-attention module for global information capturing, a channel-aware convolution block (CCB) for modeling local feature information and

channel information. This feature fusion method can improve the target localization ability and bounding box prediction capability of the proposed tracking algorithm. The main contributions of the proposed TGLC are shown below.

(i) A channel-aware convolution block (CCB) is applied behind each multi-head self-attention and cross-attention in the feature fusion module. This design fully captures global and local information from two feature maps.

(ii) Another superiority of the proposed feature fusion module is its ability to perceive channel information of feature maps. Both spatial attention and channel attention are applied to feature maps for modeling essential spatial information and channel information.

(iii) The proposed tracker (TGLC) achieves competitive and even stronger tracking performance compared with state-of-the-art tracking methods on several widely utilized tracking datasets. The tracking speed of the proposed tracker is approximately 50 frames per second, which is suitable for real-time applications.

7.2.2 MIMTracking for the second research objective

Motivated by masked image modeling and in pursuit of a neat solution to reduce redundancy for tracking during training, an end-to-end one-stage tracking method MIMTracking is proposed for long-term object tracking. There are several main contributions for the proposed MIMTracking.

(i) The presented MIMTracking employs both encoder and decoder to fuse representations of template and search region more fully during training and inference.

(ii) Instead of feeding the encoder full representations, only fragmentary, discrete

search embeddings serve as the input of the encoder for MIMTracking. This substantially alleviates the redundancy and urges the tracking network to learn useful information.

(iii) In this work, a brand-new idea is explored by directly integrating masked image modeling into the training process of MIMTracking. Instead of utilizing MIM to reconstruct the incomplete input sequences, our MIMTracking directly leverages MIM to generate tracking results (bounding boxes of the target). Surprisingly, this novel design brings further tracking performance gain. This suggests that MIM enables ViT to reach the potential of global representation reasoning by feeding only a subset of input embeddings, which helps improve the long-term tracking performance.

(iv) Extensive experiments are conducted on several commonly used short-term and long-term tracking benchmarks to assess our presented MIMTracking. The results demonstrate that MIMTracking outperforms the state-of-the-art trackers on different datasets while still running in real time.

7.2.3 ATPTrack for the third research objective

To better cope with tracking challenges, i.e., background clutter, a novel tracker ATPTrack is developed for object tracking. The main contributions of ATPTrack are shown as follows.

(i) DTP and SRP blocks are proposed to prune dynamic templates and the search region alternately. After alternating pruning, target tokens are highlighted in both dynamic templates and search region. A large number of background tokens are gradually discarded by interweavingly distributed DTP and SRP blocks. Thus, the processing time of the proposed method is reduced notably due to trimming of both

dynamic templates and search region. This also improves the capability of ATPTrack against disturbances.

(ii) A novel similarity ranking mechanism is exploited to select the most informative target-related tokens. The baseline method ProContEXT utilizes center tokens of both fixed and dynamic templates to calculate target similarity with search tokens. This is adventurous as dynamic templates are less reliable due to potentially imprecise tracking results. However, our developed similarity ranking module merely exploits center tokens of fixed templates for similarity calculation since fixed templates are accurately annotated. In this case, dynamic templates not only provide renewed target appearance for search region but operate under the supervision of accurate fixed templates, thereby ensuring the reliability of the information supplied by dynamic templates.

(iii) Experimental results indicate that ATPTrack demonstrates state-of-the-art tracking performance on multiple datasets, while substantially reducing computational consumption. ATPTrack also provides a new idea for dealing with distractors under background clutter. It achieves promising tracking performance for handling challenges, i.e., background clutter, occlusion, etc. It is hoped that the proposed ATPTrack can perform as a strong baseline for exploring the tracking potential under tracking challenges.

7.3 Future work

Although the proposed tracking methods (TGLC, MIMTracking and ATPTrack) achieve promising results, there are still some limitations. The future work is striving to improve these methods to overcome the limitations.

(i) Improvements of the short-term tracking method TGLC

There are two drawbacks for the proposed short-term tracking method TGLC. The first drawback is that only features of the last feature extraction stage are exploited for fusion. To achieve better multiple information fusion, multi-scale representations from different extraction stages can be employed for feature fusion. This is a possible improvement for the proposed TGLC.

Secondly, the proposed key point prediction head focuses on several sparse representation points of the target. These scattered sparse points are not capable of representing the target completely. Therefore, a segmentation branch can be developed to be integrated with the key point head for dense prediction of the target. This can perform a pixel-wise estimation, thereby enhancing the short-term tracking approach TGLC.

(ii) Enhancement of the long-term tracker MIMTracking

MIMTracking employs masked image modeling to enhance the tracking capability of ViT. However, ViT is time-consuming due to the global information processing in the self-attention mechanism. To address this problem, some methods, i.e., Swin Transformer attempt to restrict the computation of self-attention to a local region. Although these approaches alleviate computational costs substantially, they cannot process sequences of any length. The concatenated token sequence from the template and the search region is not processable for the improved approaches. The future work is to adapt the local ViTs to MIMTracking for improving the tracking speed.

Masked modeling is not fully explored in the proposed MIMTracking. Merely random regions of the search region are masked in MIMTracking. All template tokens are visible to the encoder and decoder during training. The future modification will apply masks to

the template. Furthermore, masked modeling can also be attempted in the inference process to reduce the redundancy.

(iii) Improvements of ATPTrack

Future research priorities for ATPTrack are twofold. The first is the possibility of replacing the token pruning with token merging. The reason is that token pruning highlights the core tokens by discarding irrelevant tokens. This may result in complete loss of information. Differently, token merging emphasizes the key tokens by merging redundant tokens. In this case, the coherence of tokens can be maintained to a certain extent. The difficulty is that more sophisticated token padding strategies should be proposed for token merging in tracking. Additionally, merging is less computationally efficient than pruning.

The second research priority is to develop more flexible retention ratios for dynamic templates and the search region to further boost the inference speed of the proposed ATPTrack. Adaptive retention ratios are attempted as an ablation study in chapter 5. However, these adaptive ratios are not as precise as fixed ratios. The reason is that the bounding boxes of dynamic templates are not as accurate as bounding boxes of fixed templates. Thus, the adaptive ratios are not sufficiently reliable since they only consider the proportions of bounding box areas in dynamic templates. In the future, more reliable adaptive retention ratios should be developed by taking into account target proportions, background influence, weight distribution of the adaptive ratios, etc.

In the future, the proposed three tracking methods (TGLC, MIMTracking and ATPTrack) have a wide range of application scenarios. Potential applications consist of several aspects.

(i) **Automatic driving**. Visual object tracking is widely utilized in the field of automatic driving due to the superior capability of adaptive tracking methods. The

proposed methods are trained across multiple driving scenarios. Therefore, they can be employed in scenarios such as path tracking, vehicle tracking, pedestrian tracking, etc.

(ii) **Robotics**. The proposed tracking methods are evaluated on various real-world video sequences of robot vision in chapter 6. The results demonstrate that the proposed methods are capable of coping with extensive tracking challenges in robot-related scenarios. Therefore, the presented methods MIMTracking and ATPTrack can be exploited for object tracking of common robots, e.g., handling robots, human following robots, pick-and-place robots, etc.

(iii) **Human-machine intelligent interaction**. Visual object tracking approaches bridge the gap between human beings and machines. Hand or finger tracking enables non-verbal communication between machines and operators. In addition, human tracking and motion trajectory analysis based on tracking results ensure safe operation of machines in shared spaces. The proposed tracking methods can be applied to efficient human-machine interaction in the future.

(iv) **Public safety**. Video surveillance serves as a critical component in the field of public safety. Visual object tracking enhances video surveillance by equipping surveillance systems with intelligence. Tracking algorithms can be utilized to track suspicious people, i.e., criminals in public areas (airports, subway stations, etc.). The proposed approaches (TGLC, MIMTracking and ATPTrack) can be incorporated into the surveillance systems for accurate tracking of unusual or suspicious objects.

Bibliography

[1] A. Faisal, M. Kamruzzaman, T. Yigitcanlar, and G. Currie, "Understanding autonomous vehicles," *J Transp Land Use*, vol. 12, no. 1, pp. 45–72, 2019.

[2] V. Suma, "Computer vision for human-machine interaction-review," *Journal of trends in Computer Science and Smart technology (TCSST)*, vol. 1, no. 02, pp. 131–139, 2019.

[3] K. A. Joshi and D. G. Thakore, "A survey on moving object detection and tracking in video surveillance system," *International Journal of Soft Computing and Engineering*, vol. 2, no. 3, pp. 44–48, 2012.

[4] S.-Y. Chen, "Kalman filter for robot vision: a survey," *IEEE Transactions on industrial electronics*, vol. 59, no. 11, pp. 4409–4420, 2011.

[5] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2411–2418.

[6] H. S. Dadi, G. K. M. Pillutla, and M. L. Makkena, "Face recognition and human tracking using GMM, HOG and SVM in surveillance videos," *Annals of Data Science*, vol. 5, no. 2, pp. 157–179, 2018.

[7] K. Laaroussi, A. Saaidi, M. Masrar, and K. Satori, "Human tracking using joint color-texture features and foreground-weighted histogram," *Multimed Tools Appl*, vol. 77, no. 11, pp. 13947–13981, 2018.

[8] N. Kumar and N. Sukavanam, "An improved CNN framework for detecting and tracking human body in unconstraint environment," *Knowl Based Syst*, vol. 193, p. 105198, 2020.

[9] I. H. Choi, J. M. Pak, C. K. Ahn, Y. H. Mo, M. T. Lim, and M. K. Song, "New preceding vehicle tracking algorithm based on optimal unbiased finite memory filter," *Measurement*, vol. 73, pp. 262–274, 2015.

[10] E. Dong, M. Deng, J. Tong, C. Jia, and S. Du, "Moving vehicle tracking based on improved tracking–learning–detection algorithm," *IET Computer Vision*, vol. 13, no. 8, pp. 730–741, 2019.

[11] S. Qiu, K. Cheng, L. Cui, D. Zhou, and Q. Guo, "A moving vehicle tracking algorithm based on deep learning," *J Ambient Intell Humaniz Comput*, pp. 1–7, 2020.

[12] Z. Zhao, Z. Chen, S. Voros, and X. Cheng, "Real-time tracking of surgical instruments based on spatio-temporal context and deep learning," *Computer Assisted Surgery*, vol. 24, no. sup1, pp. 20–29, 2019.

[13] C. I. Nwoye, D. Mutter, J. Marescaux, and N. Padoy, "Weakly supervised convolutional LSTM approach for tool tracking in laparoscopic videos," *Int J Comput Assist Radiol Surg*, vol. 14, pp. 1059–1067, 2019.

[14] Z. Chen, Z. Hong, and D. Tao, "An experimental survey on correlation filter-based tracking," *arXiv preprint arXiv:1509.05520*, 2015.

[15] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via structured multi-task sparse learning," *Int J Comput Vis*, vol. 101, no. 2, pp. 367–383, 2013.

[16] L. Sevilla-Lara and E. Learned-Miller, "Distribution fields for tracking," in *2012 IEEE Conference on computer vision and pattern recognition*, IEEE, 2012, pp. 1910–1917.

[17] S. Kwak, W. Nam, B. Han, and J. H. Han, “Learning occlusion with likelihoods for visual tracking,” in *2011 international conference on computer vision*, IEEE, 2011, pp. 1551–1558.

[18] T. Vojir, J. Noskova, and J. Matas, “Robust scale-adaptive mean-shift for tracking,” *Pattern Recognit Lett*, vol. 49, pp. 250–258, 2014.

[19] S. Hare *et al.*, “Struck: Structured output tracking with kernels,” *IEEE Trans Pattern Anal Mach Intell*, vol. 38, no. 10, pp. 2096–2109, 2015.

[20] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” *IEEE Trans Pattern Anal Mach Intell*, vol. 34, no. 7, pp. 1409–1422, 2011.

[21] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filters,” *IEEE Trans Pattern Anal Mach Intell*, vol. 37, no. 3, pp. 583–596, 2014.

[22] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, “Staple: Complementary learners for real-time tracking,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1401–1409.

[23] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, “Fully-convolutional siamese networks for object tracking,” in *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II 14*, Springer, 2016, pp. 850–865.

[24] 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*, 2018, pp. 8971–8980.

[25] 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)*, 2018, pp. 101–117.

[26] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast online object tracking and segmentation: A unifying approach," in *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, 2019, pp. 1328–1338.

[27] Z. Wang, J. Xu, L. Liu, F. Zhu, and L. Shao, "Ranet: Ranking attention network for fast video object segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3978–3987.

[28] B. Yan, X. Zhang, D. Wang, H. Lu, and X. Yang, "Alpha-refine: Boosting tracking performance by precise bounding box estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 5289–5298.

[29] 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*, 2021, pp. 8126–8135.

[30] B. Yan, H. Peng, J. Fu, D. Wang, and H. Lu, "Learning spatio-temporal transformer for visual tracking," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10448–10457.

[31] A. Vaswani *et al.*, "Attention is all you need," *Adv Neural Inf Process Syst*, vol. 30, 2017.

[32] Y. Wu, J. Lim, and M.-H. Yang, "Object Tracking Benchmark," *IEEE Trans Pattern Anal Mach Intell*, vol. 37, no. 9, pp. 1834–1848, 2015, doi: 10.1109/TPAMI.2014.2388226.

[33] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: Automatic detection of tracking failures," in *2010 20th international conference on pattern recognition*, IEEE, 2010, pp. 2756–2759.

[34] K. Lebeda, S. Hadfield, J. Matas, and R. Bowden, "Long-term tracking through failure cases," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2013, pp. 153–160.

[35] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 5388–5396.

[36] N. Liang, G. Wu, W. Kang, Z. Wang, and D. D. Feng, "Real-time long-term tracking with prediction-detection-correction," *IEEE Trans Multimedia*, vol. 20, no. 9, pp. 2289–2302, 2018.

[37] Y. Boers and J. N. Driessen, "Particle filter based detection for tracking," in *Proceedings of the 2001 American Control Conference.(Cat. No. 01CH37148)*, IEEE, 2001, pp. 4393–4397.

[38] N. Wang, W. Zhou, and H. Li, "Reliable re-detection for long-term tracking," *IEEE transactions on circuits and systems for video technology*, vol. 29, no. 3, pp. 730–743, 2018.

[39] L. Huang, X. Zhao, and K. Huang, "Globaltrack: A simple and strong baseline for long-term tracking," in *Proceedings of the AAAI conference on artificial intelligence*, 2020, pp. 11037–11044.

[40] K. Dai, Y. Zhang, D. Wang, J. Li, H. Lu, and X. Yang, "High-performance long-term tracking with meta-updater," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6298–6307.

[41] M. Dunnhofer, K. Simonato, and C. Micheloni, "Combining complementary trackers for enhanced long-term visual object tracking," *Image Vis Comput*, vol. 122, p. 104448, 2022.

[42] Y. Cui, C. Jiang, L. Wang, and G. Wu, "Mixformer: End-to-end tracking with iterative mixed attention," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 13608–13618.

[43] B. Chen *et al.*, "Backbone is all your need: A simplified architecture for visual object tracking," in *European Conference on Computer Vision*, Springer, 2022, pp. 375–392.

[44] B. Ye, H. Chang, B. Ma, S. Shan, and X. Chen, "Joint feature learning and relation modeling for tracking: A one-stream framework," in *European Conference on Computer Vision*, Springer, 2022, pp. 341–357.

[45] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16000–16009.

[46] Q. Wu, T. Yang, Z. Liu, B. Wu, Y. Shan, and A. B. Chan, "Dropmae: Masked autoencoders with spatial-attention dropout for tracking tasks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 14561–14571.

[47] Z. Song, R. Luo, J. Yu, Y.-P. P. Chen, and W. Yang, “Compact transformer tracker with correlative masked modeling,” in *Proceedings of the AAAI conference on artificial intelligence*, 2023, pp. 2321–2329.

[48] Z. Xie, Z. Geng, J. Hu, Z. Zhang, H. Hu, and Y. Cao, “Revealing the dark secrets of masked image modeling,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 14475–14485.

[49] H. Zhao, D. Wang, and H. Lu, “Representation learning for visual object tracking by masked appearance transfer,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18696–18705.

[50] H. Fan *et al.*, “Lasot: A high-quality benchmark for large-scale single object tracking,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 5374–5383.

[51] Y. Mae, Y. Shirai, J. Miura, and Y. Kuno, “Object tracking in cluttered background based on optical flow and edges,” in *Proceedings of 13th International Conference on Pattern Recognition*, IEEE, 1996, pp. 196–200.

[52] D. K. Panda and S. Meher, “Robust real-time object tracking under background clutter,” in *2011 International Conference on Image Information Processing*, IEEE, 2011, pp. 1–6.

[53] M. Mueller, N. Smith, and B. Ghanem, “Context-aware correlation filter tracking,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1396–1404.

[54] M. Wang, Y. Liu, and Z. Huang, “Large margin object tracking with circulant feature maps,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4021–4029.

[55] B. Liao, C. Wang, Y. Wang, Y. Wang, and J. Yin, “Pg-net: Pixel to global matching network for visual tracking,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXII 16*, Springer, 2020, pp. 429–444.

[56] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, “Know your surroundings: Exploiting scene information for object tracking,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, Springer, 2020, pp. 205–221.

[57] C. Mayer, M. Danelljan, D. P. Paudel, and L. Van Gool, “Learning target candidate association to keep track of what not to track,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 13444–13454.

[58] Y.-H. Chen *et al.*, “NeighborTrack: Single object tracking by bipartite matching with neighbor tracklets and its applications to sports,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 5139–5148.

[59] Z. Zhang, H. Peng, J. Fu, B. Li, and W. Hu, “Ocean: Object-aware anchor-free tracking,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXI 16*, Springer, 2020, pp. 771–787.

[60] Z. Ma, L. Wang, H. Zhang, W. Lu, and J. Yin, “Rpt: Learning point set representation for siamese visual tracking,” in *Computer Vision–ECCV 2020*

Workshops: Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16, Springer, 2020, pp. 653–665.

[61] X. Chen, B. Yan, J. Zhu, H. Lu, X. Ruan, and D. Wang, "High-performance transformer tracking," *IEEE Trans Pattern Anal Mach Intell*, vol. 45, no. 7, pp. 8507–8523, 2022.

[62] K. He, C. Zhang, S. Xie, Z. Li, and Z. Wang, "Target-aware tracking with long-term context attention," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, pp. 773–780.

[63] J.-P. Lan *et al.*, "Procontext: Exploring progressive context transformer for tracking," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2023, pp. 1–5.

[64] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *2010 IEEE computer society conference on computer vision and pattern recognition*, IEEE, 2010, pp. 2544–2550.

[65] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part IV 12*, Springer, 2012, pp. 702–715.

[66] M. Danelljan, F. Shahbaz Khan, M. Felsberg, and J. Van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1090–1097.

[67] Y. Li and J. Zhu, “A scale adaptive kernel correlation filter tracker with feature integration,” in *Computer Vision-ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part II 13*, Springer, 2015, pp. 254–265.

[68] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, “Discriminative scale space tracking,” *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 8, pp. 1561–1575, 2016.

[69] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, “Learning spatially regularized correlation filters for visual tracking,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4310–4318.

[70] 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*, 2017, pp. 6638–6646.

[71] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, “Atom: Accurate tracking by overlap maximization,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4660–4669.

[72] G. Bhat, M. Danelljan, L. Van Gool, and R. Timofte, “Learning discriminative model prediction for tracking,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 6182–6191.

[73] 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*, 2019, pp. 4282–4291.

[74] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 6, pp. 1137–1149, 2016.

[75] Y. Xu, Z. Wang, Z. Li, Y. Yuan, and G. Yu, "SiamFC++: Towards robust and accurate visual tracking with target estimation guidelines," in *Proceedings of the AAAI conference on artificial intelligence*, 2020, pp. 12549–12556.

[76] A. Lukezic, J. Matas, and M. Kristan, "D3s-a discriminative single shot segmentation tracker," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7133–7142.

[77] F. Xie, W. Yang, K. Zhang, B. Liu, G. Wang, and W. Zuo, "Learning spatio-appearance memory network for high-performance visual tracking," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 2678–2687.

[78] X. Wei, Y. Bai, Y. Zheng, D. Shi, and Y. Gong, "Autoregressive visual tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9697–9706.

[79] S. Gao, C. Zhou, and J. Zhang, "Generalized relation modeling for transformer tracking," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18686–18695.

[80] L. Lin, H. Fan, Z. Zhang, Y. Xu, and H. Ling, "Swintrack: A simple and strong baseline for transformer tracking," *Adv Neural Inf Process Syst*, vol. 35, pp. 16743–16754, 2022.

[81] S. Gao, C. Zhou, C. Ma, X. Wang, and J. Yuan, "Aiatrack: Attention in attention for transformer visual tracking," in *European Conference on Computer Vision*, Springer, 2022, pp. 146–164.

[82] D. Alexey, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv: 2010.11929*, 2020.

[83] X. Chen, H. Peng, D. Wang, H. Lu, and H. Hu, "Seqtrack: Sequence to sequence learning for visual object tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2023, pp. 14572–14581.

[84] Y. Cai, J. Liu, J. Tang, and G. Wu, "Robust object modeling for visual tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9589–9600.

[85] L. Zhang, A. Gonzalez-Garcia, J. Van De Weijer, M. Danelljan, and F. S. Khan, "Learning the model update for siamese trackers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 4010–4019.

[86] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani, "Bottleneck transformers for visual recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 16519–16529.

[87] W. Xu, Y. Xu, T. Chang, and Z. Tu, "Co-scale conv-attentional image transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9981–9990.

[88] Z. Dai, H. Liu, Q. V Le, and M. Tan, "Coatnet: Marrying convolution and attention for all data sizes," *Adv Neural Inf Process Syst*, vol. 34, pp. 3965–3977, 2021.

[89] Z. Peng *et al.*, "Conformer: Local features coupling global representations for visual recognition," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 367–376.

[90] S. Mehta and M. Rastegari, "Mobilevit: light-weight, general-purpose, and mobile-friendly vision transformer," *arXiv preprint arXiv:2110.02178*, 2021.

[91] S. Woo, J. Park, J.-Y. Lee, and I. S. Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 3–19.

[92] W. F. Hendria, Q. T. Phan, F. Adzaka, and C. Jeong, "Combining transformer and CNN for object detection in UAV imagery," *ICT Express*, vol. 9, no. 2, pp. 258–263, 2023.

[93] Y. Zhang, Y. Chen, C. Huang, and M. Gao, "Object detection network based on feature fusion and attention mechanism," *Future Internet*, vol. 11, no. 1, p. 9, 2019.

[94] D. Pandey, P. Gupta, S. Bhattacharya, A. Sinha, and R. Agarwal, "Transformer assisted convolutional network for cell instance segmentation," *arXiv preprint arXiv:2110.02270*, 2021.

[95] O. Petit, N. Thome, C. Rambour, L. Themyr, T. Collins, and L. Soler, "U-net transformer: Self and cross attention for medical image segmentation," in *Machine Learning in Medical Imaging: 12th International Workshop, MLMI 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, September 27, 2021, Proceedings 12*, Springer, 2021, pp. 267–276.

[96] C. Zhang, C. Zhang, J. Song, J. S. K. Yi, K. Zhang, and I. S. Kweon, "A survey on masked autoencoder for self-supervised learning in vision and beyond," *arXiv preprint arXiv:2208.00173*, 2022.

[97] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," *arXiv preprint arXiv:2106.08254*, 2021.

[98] Z. Xie *et al.*, “Simmim: A simple framework for masked image modeling,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 9653–9663.

[99] Y. Fang, S. Yang, S. Wang, Y. Ge, Y. Shan, and X. Wang, “Unleashing vanilla vision transformer with masked image modeling for object detection,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 6244–6253.

[100] L. Huang, S. You, M. Zheng, F. Wang, C. Qian, and T. Yamasaki, “Green hierarchical vision transformer for masked image modeling,” *Adv Neural Inf Process Syst*, vol. 35, pp. 19997–20010, 2022.

[101] X. Li, W. Wang, L. Yang, and J. Yang, “Uniform masking: Enabling mae pre-training for pyramid-based vision transformers with locality,” *arXiv preprint arXiv:2205.10063*, 2022.

[102] P. Gao, T. Ma, H. Li, Z. Lin, J. Dai, and Y. Qiao, “Convmae: Masked convolution meets masked autoencoders,” *arXiv preprint arXiv:2205.03892*, 2022.

[103] J. Chen, M. Hu, B. Li, and M. Elhoseiny, “Efficient self-supervised vision pretraining with local masked reconstruction,” *arXiv preprint arXiv:2206.00790*, 2022.

[104] J. Liu, X. Huang, J. Zheng, Y. Liu, and H. Li, “MixMAE: Mixed and masked autoencoder for efficient pretraining of hierarchical vision transformers,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 6252–6261.

[105] B. Chen *et al.*, “Psvit: Better vision transformer via token pooling and attention sharing,” *arXiv preprint arXiv:2108.03428*, 2021.

[106] Y. Xu *et al.*, “Evo-vit: Slow-fast token evolution for dynamic vision transformer,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022, pp. 2964–2972.

[107] Y. Rao, W. Zhao, B. Liu, J. Lu, J. Zhou, and C.-J. Hsieh, “Dynamicvit: Efficient vision transformers with dynamic token sparsification,” *Adv Neural Inf Process Syst*, vol. 34, pp. 13937–13949, 2021.

[108] H. Yin, A. Vahdat, J. M. Alvarez, A. Mallya, J. Kautz, and P. Molchanov, “A-vit: Adaptive tokens for efficient vision transformer,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10809–10818.

[109] S. Ding, P. Zhao, X. Zhang, R. Qian, H. Xiong, and Q. Tian, “Prune spatio-temporal tokens by semantic-aware temporal accumulation,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 16945–16956.

[110] D. Bolya, C.-Y. Fu, X. Dai, P. Zhang, C. Feichtenhofer, and J. Hoffman, “Token merging: Your vit but faster,” *arXiv preprint arXiv:2210.09461*, 2022.

[111] Z. Feng and S. Zhang, “Efficient vision transformer via token merger,” *IEEE Transactions on Image Processing*, 2023.

[112] M. Kim, S. Gao, Y.-C. Hsu, Y. Shen, and H. Jin, “Token fusion: Bridging the gap between token pruning and token merging,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1383–1392.

[113] M. Chen *et al.*, "Diffrate: Differentiable compression rate for efficient vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 17164–17174.

[114] 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*, 2016, pp. 770–778.

[115] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*, Springer, 2020, pp. 213–229.

[116] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[117] Z. Yang, S. Liu, H. Hu, L. Wang, and S. Lin, "Reppoints: Point set representation for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9657–9666.

[118] 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*, 2019, pp. 658–666.

[119] 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, 2019.

[120] M. Muller, A. Bibi, S. Giancola, S. Alsubaihi, and B. Ghanem, "Trackingnet: A large-scale dataset and benchmark for object tracking in the wild," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 300–317.

[121] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, pp. 740–755.

[122] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, Ieee, 2009, pp. 248–255.

[123] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[124] D. Guo, Y. Shao, Y. Cui, Z. Wang, L. Zhang, and C. Shen, "Graph attention tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 9543–9552.

[125] H. Yu *et al.*, "Learning dynamic compact memory embedding for deformable visual object tracking," *IEEE Trans Neural Netw Learn Syst*, vol. 35, no. 4, pp. 5656–5670, 2022.

[126] L. Zheng, M. Tang, Y. Chen, J. Wang, and H. Lu, "Learning feature embeddings for discriminant model based tracking," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XV 16*, Springer, 2020, pp. 759–775.

[127] M. Danelljan, L. Van Gool, and R. Timofte, “Probabilistic regression for visual tracking,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 7183–7192.

[128] F. Ma *et al.*, “Unified transformer tracker for object tracking,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 8781–8790.

[129] H. Zhang, Z. Zhang, J. Zhang, Y. Zhao, and M. Gao, “Online bionic visual siamese tracking based on mixed time-event triggering mechanism,” *Multimed Tools Appl*, vol. 82, no. 10, pp. 15199–15222, 2023.

[130] S. Javed *et al.*, “A novel algorithm based on a common subspace fusion for visual object tracking,” *IEEE Access*, vol. 10, pp. 24690–24703, 2022.

[131] H. Zhang, J. Liang, J. Zhang, T. Zhang, Y. Lin, and Y. Wang, “Attention-driven memory network for online visual tracking,” *IEEE Trans Neural Netw Learn Syst*, 2023.

[132] J. Liu, Y. Wang, X. Huang, and Y. Su, “Tracking by dynamic template: Dual update mechanism,” *J Vis Commun Image Represent*, vol. 84, p. 103456, 2022.

[133] J. Wang, H. Zhang, J. Zhang, M. Miao, and J. Zhang, “Dual-branch memory network for visual object tracking,” in *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, Springer, 2022, pp. 646–658.

[134] Y. Yu, Y. Xiong, W. Huang, and M. R. Scott, “Deformable siamese attention networks for visual object tracking,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6728–6737.

[135] K. Yang, H. Zhang, D. Zhou, and L. Liu, “TGAN: A simple model update strategy for visual tracking via template-guidance attention network,” *Neural Networks*, vol. 144, pp. 61–74, 2021.

[136] F. Du, P. Liu, W. Zhao, and X. Tang, “Correlation-guided attention for corner detection based visual tracking,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6836–6845.

[137] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, “Siamese box adaptive network for visual tracking,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6668–6677.

[138] M. Danelljan, A. Robinson, F. Shahbaz Khan, and M. Felsberg, “Beyond correlation filters: Learning continuous convolution operators for visual tracking,” in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, Springer, 2016, pp. 472–488.

[139] P. Li, B. Chen, W. Ouyang, D. Wang, X. Yang, and H. Lu, “GradNet: Gradient-guided network for visual object tracking,” in *Proceedings of the IEEE/CVF International conference on computer vision*, 2019, pp. 6162–6171.

[140] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, “Convolutional features for correlation filter based visual tracking,” in *Proceedings of the IEEE international conference on computer vision workshops*, 2015, pp. 58–66.

[141] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, “End-to-end representation learning for correlation filter based tracking,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2805–2813.

[142] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for UAV tracking," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, Springer, 2016, pp. 445–461.

[143] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "SiamCAR: Siamese fully convolutional classification and regression for visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6269–6277.

[144] Z. Liu *et al.*, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10012–10022.

[145] J. L. Ba, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[146] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 734–750.

[147] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, JMLR Workshop and Conference Proceedings, 2010, pp. 249–256.

[148] N. Wang, W. Zhou, J. Wang, and H. Li, "Transformer meets tracker: Exploiting temporal context for robust visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 1571–1580.

[149] Z. Zhang, Y. Liu, X. Wang, B. Li, and W. Hu, “Learn to match: Automatic matching network design for visual tracking,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 13339–13348.

[150] Y. Liang, Q. Li, and F. Long, “Global dilated attention and target focusing network for robust tracking,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023, pp. 1549–1557.

[151] H. Fan *et al.*, “Lasot: A high-quality large-scale single object tracking benchmark,” *Int J Comput Vis*, vol. 129, pp. 439–461, 2021.

[152] X. Wang *et al.*, “Towards more flexible and accurate object tracking with natural language: Algorithms and benchmark,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13763–13773.

[153] D. Yang, J. He, Y. Ma, Q. Yu, and T. Zhang, “Foreground-background distribution modeling transformer for visual object tracking,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 10117–10127.

[154] X. Li, Y. Huang, Z. He, Y. Wang, H. Lu, and M.-H. Yang, “Citetracker: Correlating image and text for visual tracking,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 9974–9983.

[155] Y. Cui, C. Jiang, G. Wu, and L. Wang, “MixFormer: End-to-End Tracking With Iterative Mixed Attention,” *IEEE Trans Pattern Anal Mach Intell*, vol. 46, no. 6, pp. 4129–4146, 2024, doi: 10.1109/TPAMI.2024.3349519.