DISS. ETH NO. 29277

TRACKING GENERIC OBJECTS IN VIDEOS

A dissertation submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH
(Dr. sc. ETH Zurich)

presented by

CHRISTOPH MAYER
Master of Science in Information Technology
and Electrical Engineering

born on 8 November 1991
citizen of Switzerland

accepted on the recommendation of

Prof. Dr. Luc Van Gool, examiner
Prof. Dr. Andrea Vedaldi, co-examiner
Prof. Dr. Haibin Ling, co-examiner
Dr. Martin Danelljan, co-examiner

2023

ABSTRACT

Visual object tracking is a fundamental problem in computer vision and finds its application in multiple tasks such as autonomous driving, robotics, surveillance, video understanding, and sports analysis. Generic Object Tracking (GOT) is a specialized tracking task that aims at tracking virtually any object in a video by using a user-specified bounding box that defines the target object in the initial video frame. Learning a target model, in order to track the target in each frame, from such sparse information proves extremely challenging. Especially in adverse tracking scenarios, where the target object is frequently occluded, goes out of view, or where distractors, visually similar objects as the target, are present. Thus, we tackle the problem of robust generic object tracking in videos even in challenging scenarios in this thesis.

First, we propose a novel tracking architecture that *keeps track* of distractor objects in order to continue tracking the target. We achieve this by learning an association network, that allows to propagate the identities of all target candidates from frame-to-frame. To tackle the problem of lacking ground-truth correspondences between distractor objects in visual tracking, we propose a training strategy that combines partial annotations with self-supervision.

Second, we introduce a Transformer-based target model predictor that produces the target model. The employed Transformer captures global relations with little inductive bias, allowing it thus to learn the prediction of powerful target models even for challenging sequences. We further extend the model predictor to estimate a second set of weights, which are applied for accurate bounding box regression.

Third, we propose the new visual tracking benchmark, AVisT, dedicated for tracking scenarios with adverse visibility. AVisT contains 18 diverse scenarios broadly grouped into five attributes with 42 object categories. The key contribution of AVisT are diverse and challeng-

ing scenarios, covering severe weather conditions, obstruction and adverse imaging effects, along with camouflage.

Finally, we propose the task of multi-object GOT, that benefits from a wider applicability than tracking only a single generic object per video, rendering it more attractive in real-world applications. To this end, we introduce a new large-scale GOT benchmark, LaGOT, containing multiple annotated target objects per sequence. Our benchmark allows researchers to tackle remaining challenges in GOT, aiming to increase robustness and reduce computation through joint tracking of multiple objects simultaneously. Furthermore, we propose a Transformer-based GOT tracker capable of joint processing of multiple objects through shared computation.

ZUSAMMENFASSUNG

Die visuelle Objektverfolgung ist ein grundlegendes Problem im Bereich des computerbasierten Sehens (Computer Vision) und findet ihre Anwendung in zahlreichen Aufgabenbereichen wie autonomes Fahren, Robotik, Überwachung, Videoverständnis und -bearbeitung, und Sportanalysen. Die Verfolgung generischer Objekte (Generic Object Tracking (GOT)) ist eine spezialisierte Verfolgungsaufgabe. GOT zielt darauf ab, praktisch jedes beliebige Objekt in einem Video zu verfolgen. Das Objekt muss allerdings zuvor, durch ein Rechteck im Anfangsbild des Videos, von einem Nutzer gekennzeichnet worden sein. Das Erlernen eines Zielobjektmodells, um das Zielobjekt in jedem Bild zu verfolgen, erweist sich bei solch spärlichen Informationen als äusserst schwierig. Dies gilt insbesondere für ungünstige Szenarien, in denen das Zielobjekt häufig verdeckt ist, aus dem Blickfeld verschwindet oder in denen visuell ähnliche Objekte wie das Zielobjekt vorhanden sind. Daher befassen wir uns in dieser Arbeit mit dem Problem der robusten Verfolgung von generischen Objekten in Videos, selbst in schwierigen Fällen.

Zunächst schlagen wir eine neuartige Verfolgungsmethode vor, die visuell ähnliche Objekte wie das Zielobjekt im Auge behält, um die Verfolgung des Zielobjekts zu verbessern. Wir erreichen dies durch das Erlernen eines Assoziationsnetzwerks, das es ermöglicht, die Kennzeichnung aller Zielobjekt-Kandidaten von Bild zu Bild weiterzugeben. Um das Problem fehlender gekennzeichneter Trainingsdaten zwischen den Zielobjekt-Kandidate bei der Objektverfolgung zu lösen, schlagen wir eine Trainingsstrategie vor, die teilweise gekennzeichnete Trainingsdaten mit Selbst-Überwachtem-Lernen kombiniert.

Zweitens führen wir einen Transformer-basierten Modellschätzer ein. Der eingesetzte Transformer erfasst globale Beziehungen mit geringer induktiver Verzerrung, so dass er die Vorhersage leistungs-

fähiger Zielobjektmodelle auch für schwierige Sequenzen erlernen kann. Ausserdem erweitern wir den Modellschätzer damit ein zweiter Satz von Gewichten generiert werden kann, die dann für eine genaue Regression des Zielobjekt-Begrenzungsrahmen verwendet werden kann.

Drittens schlagen wir den neuen visuellen Objekt-Verfolgungs-Benchmark AVisT vor, der sich auf Szenarien mit ungünstigen Sichtverhältnissen fokusiert. AVisT enthält 18 verschiedene Szenarien, die grob in fünf Attribute mit 42 Objektkategorien unterteilt sind. Der Hauptbeitrag von AVisT sind die vielfältigen und anspruchsvollen Szenarien, die schwierige Wetterbedingungen, Hindernisse, ungünstige Bildeffekte, und Tarnung umfassen.

Schliesslich schlagen wir die neue Aufgabe vor, mehrere generische Objekte im gleichen Video zu verfolgen. Diese Ausfgabe profitiert von einer breiteren Anwendbarkeit und macht die Aufgabe in realen Anwendungen attraktiver als die Verfolgung eines einzelnen generischen Objekts pro Video. Daher führen wir einen neuen gross angelegten Benchmark ein, LaGOT, der mehrere gekennzeichnete Zielobjekte pro Sequenz enthält. Unser Benchmark ermöglicht es den Forschern, die wichtigsten verbleibenden Herausforderungen im Bereich der generischen Objektverfolgung anzugehen, um die Robustheit zu erhöhen und den Rechenaufwand durch die gemeinsame Verfolgung mehrerer Objekte gleichzeitig zu reduzieren. Darüber hinaus schlagen wir einen auf Transformer basierenden Tracker vor, der die Verarbeitung mehrerer Objekte gleichzeitig durch gemeinsame Berechnung ermöglicht.

PUBLICATIONS

The following publications are included in part or as a whole in this thesis (* denotes equal contribution):

- Christoph Mayer, Martin Danelljan, Danda Pani Paudel, and Luc Van Gool. "Learning Target Candidate Association to Keep Track of What not to Track". In: *IEEE/CVF International Conference on Computer Vision (ICCV)* 2021.
- Christoph Mayer, Martin Danelljan, Goutam Bhat, Matthieu Paul, Danda Pani Paudel, Fisher Yu, and Luc Van Gool. "Transforming Model Prediction for Tracking". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 2022.
- Mubashir Noman*, Wafa H Al Ghallabi*, Daniya Kareem*, Christoph Mayer*, Akshay Dudhane, Martin Danelljan, Hisham Cholakkal, Salman Khan, Luc Van Gool, and Fahad Shahbaz Khan. "AVisT: A Benchmark for Visual Object Tracking in Adverse Visibility". In: *British Machine Vision Conference (BMVC)* 2022.
- Christoph Mayer, Martin Danelljan, Ming-Hsuan Yang, Vittorio Ferrari, Luc Van Gool, and Alina Kuznetsova. "Beyond SOT: It's Time to Track Multiple Generic Objects at Once". Under Review 2023.

Furthermore, the following publications were part of my Ph.D. research, but not included in this thesis.

- Matthieu Paul, Martin Danelljan, Christoph Mayer, and Luc Van Gool. "Robust Visual Tracking by Segmentation". In: *European Conference on Computer Vision (ECCV)* 2022.

- Matthieu Paul, Christoph Mayer, Luc Van Gool, and Radu Timofte. "Efficient Video Semantic Segmentation with Labels Propagation and Refinement". *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* 2020.

- Christoph Mayer, and Radu Timofte. "Adversarial Sampling for Active Learning". In: *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* 2020.

- Christoph Mayer, Matthieu Paul, and Radu Timofte. "Adversarial Feature Distribution Alignment for Semi-Supervised Learning". In: *Computer Vision and Image Understanding* 2021.

- Christoph Mayer, Radu Timofte, and Grégory Paul. "Towards Closing the Gap in Weakly Supervised Semantic Segmentation with DCNNs: Combining Local and Global Models". In: *Computer Vision and Image Understanding* 2021.

- Yawei Li, Shuhang Gu, Christoph Mayer, Luc Van Gool and Radu Timofte. "Group Sparsity: The Hinge between Filter Pruning and Decomposition for Network Compression". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* 2020.

- Dorothea Obwegeser, Radu Timofte, Christoph Mayer, Theodore Eliades, Michael M Bornstein, Marc A Schätzle, and Raphael Patcas. "Using artificial intelligence to determine the influence of dental aesthetics on facial attractiveness in comparison to other facial modifications". In: *European Journal of Orthodontics* 2022.

ACKNOWLEDGEMENTS

The journey of pursuing my PhD at ETH has been an extraordinary and unforgettable journey. I am deeply grateful for the exceptional guidance of my supervisors, the camaraderie of my colleagues, and the support of my family and friends.

First, I would like to thank Prof. Dr. Luc Van Gool for his supervision, guidance and for creating such an inspiring environment at the CVL. My deep gratitude goes to Dr. Martin Danelljan, whose mentorship, support, and supervision were absolutely invaluable. Without him, my PhD would have been a much tougher and definitely less joyful ride. To my favourite project head Dr. Danda Pani Paudel, I extend my gratitude for his great support and collaboration. Thanks to him, working on projects while conducting research during my PhD was more fun and less stressful.

I would like to thank my co-authors, Goutam, Matthieu, Yawei, Fisher and Radu for fruitful discussions and their help even during night sessions before paper deadlines. Moreover, I would like to thank my colleagues Alina, Ming-Hsuan, Vitto, Stefan and Paul for their supervision and collaboration during my internship at Google Research. Furthermore, I would like to express my appreciation to my thesis committee members Prof. Dr. Andrea Vedaldi and Prof. Dr. Haibin Ling for reviewing my thesis and for engaging at my defense.

Further, I am extremely grateful for having such wonderful colleagues at the lab. I thank Mo, Evan, Menelaos, Prune, Nikola, Goutam, Nico, Matthieu, Erik, Martin and Sergi for all the nice memories. I will always remember the adventurous sailing, funny skiing and the terrific conference trips.

Finally, I would like to thank my family Susanne, Peter, Kathrin, Emma and Mia for their understanding, support and for always cheering me up and making the more challenging moments less exhausting.

CONTENTS

1

INTRODUCTION

Human perception has developed highly sophisticated visual tracking capabilities that enable us to track with incredible accuracy and speed. Our visual system is highly adaptable and is able to rapidly adjust the gaze in response to changes in the visual environment. Moreover, human tracking abilities are not limited to simply following a target in a linear path. We are able to track in three dimensions, allowing us to accurately perceive depth and distance. Another key aspect of human tracking abilities is our deep understanding of the environment. By having a fundamental understanding of the physical world around us such as the principles of motion, humans can predict and anticipate the movements of different components in our surroundings. This allows us to track targets with incredible accuracy, even in complex and dynamic environments, and avoid collisions when navigating through crowded spaces.

While humans possess remarkable visual tracking capabilities, automating this task for machines is crucial for many applications. In practice, visual tracking can be applied in any field where the path of a target needs to be reconstructed over time. Thus, visual tracking has a wide range of practical applications across various domains. For instance, in the field of autonomous systems such as self-driving cars and drones [45, 122], visual tracking is used to detect and track objects like pedestrians, vehicles, and obstacles in real-time for safe navigation. In the domain of robotics, visual tracking is used to track the position and movement of robots or other objects in the environment to enable precise control, localization and manipulation [19, 95]. In 3D computer vision, tracking of key points is used to estimate camera pose, which in turn is used for 3D reconstruction and augmented reality applications [21, 86]. Industrial quality control also leverages tracking of automated processes using video

cameras for monitoring and identifying any anomalies in production lines. Similarly, the tracking of human and animal activities and behavior is used for understanding and analyzing their movements and poses, with applications in fields like sports, entertainment, and wildlife research [2, 100, 111]. Finally, tracking methods are used for semi-automatic annotation of videos [65], enabling improved video understanding for a wide range of applications.

Generic Object Tracking (GOT) is a specialized tracking task that involves tracking objects of virtually any category without prior knowledge of the object or the environment. This is typically achieved by defining the target object in the initial frame of the video sequence using a user-specified bounding box. The challenge of GOT is that the tracker needs to learn to track the target using only a single training example provided at test time, without any additional information about the target's movement or appearance. Additionally, the tracker needs to be able to operate in various domains with different lighting, resolution, and application scenarios. This makes GOT particularly challenging, as the tracker needs to be able to handle significant appearance changes of the target object, as well as adapt to different environments such as underwater or at microscopic scales.

GOT can be further complicated by objects that are visually similar to the target, known as distractor objects, as well as by background clutter and camouflage. Additionally, target objects are often in motion, occluded, or go out of view, making it difficult to learn their motion models and re-detect them once they reappear. These factors can significantly limit the tracking performance. The quality of video capturing also plays a crucial role in successful target tracking. Particularly harmful effects are camera motion, motion blur, and changes in scale, illumination, and view-point.

Due to the wide applicability of visual tracking, numerous trackers have been proposed to tackle the described challenges. Discriminative Correlation Filter (DCF)-based and Siamese trackers have shown great potential for robust tracking even in adverse tracking scenarios. DCF-based trackers formulate the target localization problem as an optimization problem and therefore require solving the said problem

during inference [10, 26, 27, 51, 52]. Due to their setup they allow to update the target model with prediction results of previous frames and can therefore track target objects that undergo severe appearance changes [28]. A downside of DCF-based trackers is that while they use deep learned feature representations, they are not readily end-to-end trainable [6]. Thus, Siamese trackers largely gained in popularity because they are end-to-end trainable and operate at a very high frame rate. However, Siamese trackers cannot as easily integrate previous predictions as DCF-based trackers. This characteristic makes Siamese trackers particularly vulnerable to severe appearance changes, to distractors or to sequences with heavy background clutter and camouflage [5, 68, 101, 115]. Furthermore, existing GOT methods focus on tracking a single object per video sequence instead of processing multiple targets at the same time. However, multi-object GOT has the potential of increasing the robustness of a tracker, by joint reasoning across all tracks to reduce the risk of confusing similar objects.

In this thesis, we aim at addressing several open challenges in visual object tracking. In Chapter 3, we propose a novel tracking architecture that *keeps track* of distractor objects to continue tracking the target. To achieve this, we first retrieve the target candidates that correspond to local maxima in the score map of the base tracker. Both, DCF-based and Siamese trackers produces such a score map. Then, we introduce a learned association network, that allows to propagate the identities of all target candidates from frame-to-frame.

Next, in Chapter 4, we introduce a powerful Transformer-based model prediction module, that reduces the need of specialized distractor mitigation approaches. The main idea is to replace optimization-based model predictors, that are typically employed in DCF-based trackers, with a Transformer. The Transformer is then learned in an end-to-end fashion to predict the target model for each frame individually. In contrast to DCF-based trackers, we further extend the model predictor to estimate a second set of weights that enable accurate bounding box regression.

Beside new tracking architectures, we introduce in Chapter 5, the new visual tracking benchmark AVisT. AVisT is dedicated for tracking scenarios with adverse visibility, such as severe weather or camouflage. Furthermore, we study the robustness of existing trackers in such challenging settings.

Lastly, in Chapter 6, we propose the multi-object GOT task that benefits from a wider applicability than single object tracking. The idea of this task is developing trackers that are capable of tracking multiple generic objects at the same time in each video sequence. To this end, we introduce a new large-scale GOT benchmark, LaGOT, containing multiple annotated target objects per sequence. Furthermore, we propose a Transformer-based GOT tracker that is capable of joint processing of multiple objects through shared computation.

2

RELATED WORK

We tackle the problem of GOT. In this chapter we provide a brief overview of the existing literature covering GOT datasets and benchmarks and the most popular tracking methodologies ranging from DCF-based trackers and Siamese trackers to Transformer-based trackers.

2.1 GENERIC OBJECT TRACKING DATASETS AND BENCHMARKS

In GOT there are specialized datasets and challenges that focus on short-term or long-term tracking. OTB [109] is one of the most popular short-term tracking datasets, containing 100 videos with an average length of 578 frames. Other similarly large short-term tracking datasets that focus on the impact of color information (TempleColor [70]) or on fast moving targets (NFS [42]). In contrast, GOT-10k [54] and TrackingNet [85] are large-scale short term tracking datasets that contain thousands of different videos (10k and 30k videos respectively). While most GOT datasets contain less than 100 different object classes, GOT-10k contains objects from 563 different class categories. In addition to all these tracking datasets, there exist dedicated benchmarks, such as the VOT series [60–64] associated with annual tracking challenge competitions. The VOT challenge mainly focuses on short-term tracking but contains several sub-challenges, where the VOTLT sub-challenge focuses on long-term tracking.

Short-term tracking focuses on accurate localization and bounding-box regression in videos where the objects is always at least partially visible [62, 109]. Thus, short-term trackers must track the target object reliably even in challenging scenarios where the target object undergoes sever appearance changes such as deformations or scale variations or when visually similar objects as the target are present. Other

challenging settings are low-resolution, fast-motion, background clutter, motion blur, viewpoint changes or camera motion [42, 85, 109]. In addition to all these challenges, long-term tracking includes sequences where the target object goes out-of-view, is fully or at least partially occluded [38, 84, 102]. Thus, beyond tracking the target object while it is visible, long-term trackers are required to re-detect the previously occluded target object as soon as it is visible again [60].

2.2 GENERIC OBJECT TRACKERS

Discriminative Correlation Filter-based Trackers: A successful GOT methodology is DCF-based tracking [10, 26, 27, 29, 51, 52]. Such methods learn a correlation filter from a set of training samples. The correlation filter is optimized to localize the target objects on the training samples by performing a circular sliding window operation. Due to the special structure of the formulated problem, the Fast Fourier Transform (FFT) can be used to efficiently obtain the correlation filter. Early DCF based trackers such as MOSSE [10] showed several limitations caused by the circular convolution, by the simple gray scale image features or by using a single scale. Thus, multiple follow up works tackled these problems [26, 27, 29]. The popularity of deep learning lead to astonishing improvements particularly in learning feature representations for effective target classification and for accurate target estimation [6, 8, 23, 25, 29].

More recent but still related GOT methods no longer used FFT-based algorithms to produce the target model but relied on alternative optimization strategies. ATOM [23] employed a two-layer Perceptron as target model and used Conjugate Gradient to solve the target classification problem. In order to train such models end-to-end, DiMP [6] introduced the idea of unrolling the iterative optimization algorithm for a fixed number of iterations and to integrate it in the tracking pipeline. In addition, ATOM [23] introduced a dedicated bounding box regression branch, making the multi-scale detection

strategy obsolete. PrDiMP [30] improves on the target state estimation of ATOM by applying a probabilistic regression formulation.

Siamese Trackers: Other related and very popular GOT methods are Siamese trackers that employ Siamese networks to learn a similarity metric [5, 16, 46, 47, 67, 68, 101, 104, 115, 124, 128, 133]. Siamese trackers are end-to-end trainable and typically achieve a high tracking efficiency. SiamFC [5] used a fully-convolutional deep network to learn a strong embedding in an offline phase. SiamRPN [68] and SiamRPN++ [67] introduced region proposal components that are extensively trained offline to regress more accurate bounding boxes. SiamCAR [47], SiamFC++ [115], SiamBAN [16], and Ocean [128] are anchor-free trackers that used separate branches for target object classification and bounding box regression.

Transformer-based Trackers: Recently Transformer-based trackers became very popular and powerful [14, 18, 41, 44, 106, 116, 117, 120, 121]. TrDiMP [106] and TrSiam [106] use traditional tracking architectures but integrated a Transformer encoder and decoder to produce refined template and search area features. TransT [14] introduced a feature fusion network with alternating self- and cross-attention blocks resulting in fused feature vectors that allow to localize the object and regress its bounding box. STARK [117] adopted the Transformer architecture of DETR [12]. Instead of fusing the template and search area features in the Transformer decoder, they are stacked and jointly processed by the encoder and the decoder. A single object query is fed into the the decoder and its output is fused with the encoder features in order to directly regress the target bounding box. MixFormer [18] and OSTrack [120] no longer employ classical backbones such as ResNets [50] to first produce visual features for tracking but uses one Transformer architecture [35, 108] to jointly extract template and search area features and for the feature fusion process. Eventually, shallow components produce the target classification scores and the bounding box.

3

LEARNING TARGET CANDIDATE ASSOCIATION TO KEEP TRACK OF WHAT NOT TO TRACK

In this chapter, we propose a novel tracking architecture that *keeps track* of distractor objects in order to continue tracking the target. To this end, we introduce a learned association network, allowing us to propagate the identities of all target candidates from frame-to-frame. To tackle the problem of lacking ground-truth correspondences between distractor objects in visual tracking, we propose a training strategy that combines partial annotations with self-supervision. The code and trained models are available at https://github.com/visionml/pytracking.

3.1 INTRODUCTION

Generic visual object tracking is one of the fundamental problems in computer vision. The task involves estimating the state of the target object in every frame of a video sequence, given only the initial target location. Most prior research has been devoted to the development of robust appearance models, used for locating the target object in each frame. The two currently dominating paradigms are Siamese networks [5, 67, 68] and discriminative appearance modules [6, 25]. While the former employs a template matching in a learned feature space, the latter constructs an appearance model through a discriminative learning formulation. Although these approaches have demonstrated promising performance in recent years, they are effectively limited by the quality and discriminative power of the appearance model.

As one of the most challenging factors, co-occurrence of distractor objects similar in appearance to the target is a common problem in real-world tracking applications [7, 112, 133]. Appearance-based

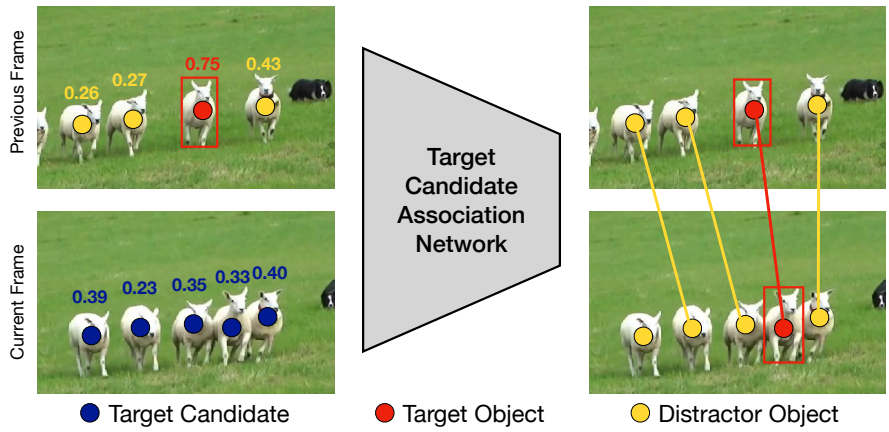


FIGURE 3.1: Visualization of the proposed target candidate association network used for tracking. For each target candidate (●) we extract a set of features such as score, position and appearance in order to associate candidates across frames. The proposed target association network then allows to associate these candidates (●) with the detected distractors (●) and the target object (●) of the previous frame. Lines connecting circles represent associations.

models struggle to identify the sought target in such cases, often leading to tracking failure. Moreover, the target object may undergo a drastic appearance change over time, further complicating the discrimination between target and distractor objects. In certain scenarios, e. g., as visualized in Fig. 3.1, it is even virtually impossible to unambiguously identify the target solely based on appearance information. Such circumstances can only be addressed by leveraging other cues during tracking, for instance the spatial relations between objects. We therefore set out to address problematic distractors by exploring such alternative cues.

We propose to actively keep track of distractor objects in order to ensure more robust target identification. To this end, we introduce a target candidate association network, that matches distractor objects as well as the target across frames. Our approach consists of a base appearance tracker, from which we extract target candidates in each

frame. Each candidate is encoded with a set of distinctive features, consisting of the target classifier score, location, and appearance. The encodings of all candidates are jointly processed by a graph-based candidate embedding network. From the resulting embeddings, we compute the association scores between all candidates in subsequent frames, allowing us to keep track of the target and distractor objects over time. In addition, we estimate a target detection confidence, used to increase the robustness of the target classifier.

While associating target candidates over time provides a powerful cue, learning such a matching network requires tackling a few key challenges. In particular, generic visual object tracking datasets only provide annotations of one object in each frame, i. e., the target. As a result, there is a lack of ground-truth annotations for associating distractors across frames. Moreover, the definition of a distractor is not universal and depends on the properties of the employed appearance model. We address these challenges by introducing an approach that allows our candidate matching network to learn from real tracker output. Our approach exploits the single target annotations in existing tracking datasets in combination with a self-supervised strategy. Furthermore, we actively mine the training dataset in order to retrieve rare and challenging cases, where the use of distractor association is important, in order to learn a more effective model.

Contributions: In summary, our contributions are as follows:

(i) We propose a method for target candidate association based on a learnable candidate matching network.

(ii) We develop an online object association method in order to propagate distractors and the target over time and introduce a sample confidence score to update the target classifier more effectively during inference.

(iii) We tackle the challenges with incomplete annotation by employing partial supervision, self-supervised learning, and sample-mining to train our association network.

(iv) We perform comprehensive experiments and ablative analyses by integrating our approach into the tracker SuperDiMP [6, 24, 30]. The resulting tracker *KeepTrack* sets a new state-of-the-art on six tracking datasets, obtaining an AUC of 67.1% on LaSOT [38] and 69.7% on UAV123 [84].

3.2 RELATED WORK

Discriminative appearance model based trackers [6, 23, 29, 43, 52, 125] aim to suppress distractors based on their appearance by integrating background information when learning the target classifier online. While often increasing robustness, the capacity of an online appearance model is still limited. A few works have therefore developed more dedicated strategies of handling distractors. Bhat *et al*. [7] combine an appearance based tracker and an RNN to propagate information about the scene across frames. It internally aims to track all regions in the scene by maintaining a learnable state representation. Other methods exploit the existence of distractors explicitly in the method formulation. DaSiamRPN [133] handles distractor objects by subtracting corresponding image features from the target template during online tracking. Xiao *et al*. [112] use the locations of distractors in the scene and employ hand crafted rules to classify image regions into background and target candidates on each frame. SiamRCNN [104] associates subsequent detections across frames using a hand-crafted association score to form short tracklets. In contrast, we introduce a learnable network that explicitly associates target candidates from frame-to-frame. Zhang *et al*. [129] propose a tracker inspired by the Multiple Object Tracking (MOT) philosophy of tracking by detection. They use the top-k predicted bounding boxes for each frame and link them between frames by using different features. In contrast, we omit any hand crafted features but fully learn to predict the associations using self-supervision.

Many online trackers [6, 23, 25] employ a memory to store previous predictions to fine-tune the tracker. Typically the oldest sample is

replaced in the memory and an age-based weight controls the contribution of each sample when updating the tracker online. Danelljan *et al*. [28] propose to learn the tracking model and the training sample weights jointly. LTMU [22] combines an appearance based tracker with a learned meta-updater. The goal of the meta-updater is to predict whether the employed online tracker is ready to be updated or not. In contrast, we use a learned target candidate association network to compute a confidence score and combine it with sample age to manage the tracker updates.

The object association problem naturally arises in MOT. The dominant paradigm in MOT is *tracking-by-detection* [4, 11, 98, 123, 126], where tracking is posed as the problem of associating object detections over time. The latter is typically formulated as a graph partitioning problem. Typically, these methods are non-causal and thus require the detections from all frames in the sequence. Furthermore, MOT typically focuses on a limited set of object classes [32], such as pedestrians, where strong object detectors are available. In comparison we aim at tracking different objects in different sequences solely defined by the initial frame. Furthermore, we lack ground truth correspondences of all distractor objects from frame to frame whereas the ground-truth correspondences of different objects in MOT datasets are typically provided [32]. Most importantly, we aim at associating target candidates that are defined by the tracker itself, while MOT methods associate all detections that correspond to one of the sought objects.

3.3 METHOD

In this section, we describe our tracking approach, which actively associates distractor objects and the sought target across multiple frames.

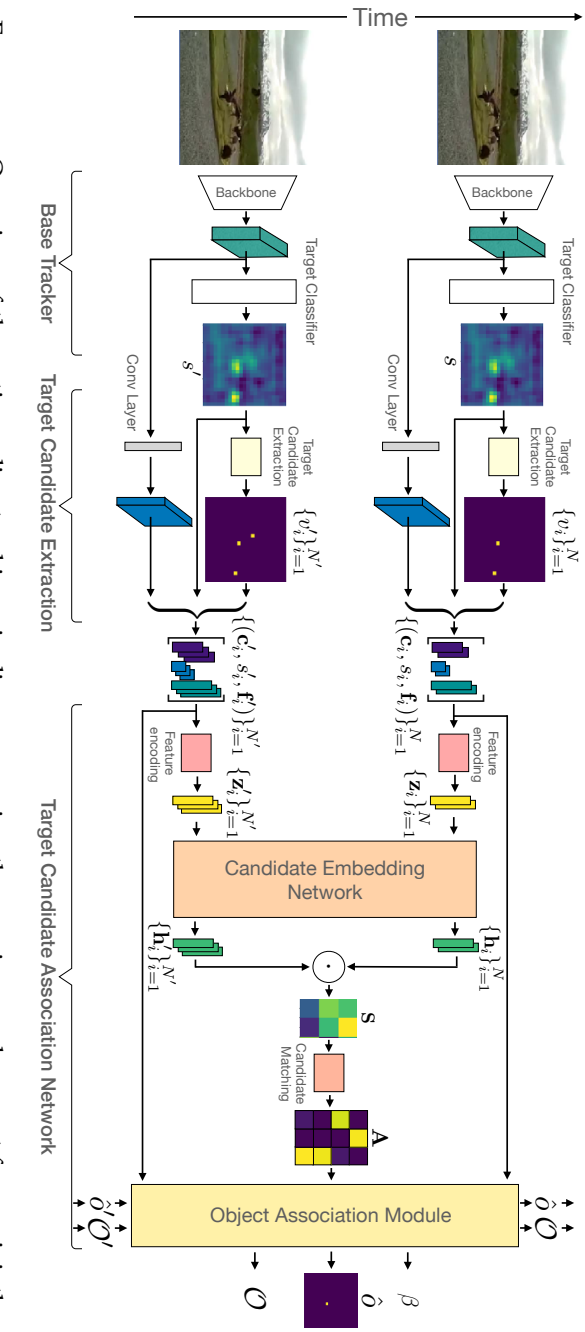


FIGURE 3.2: Overview of the entire online tracking pipeline, processing the previous and current frames jointly to predict the target object.

3.3.1 *Overview*

An overview of our tracking pipeline is shown in Fig. 3.2. We use a base tracker with a discriminative appearance model and internal memory. In particular, we adopt the SuperDiMP [24, 49] tracker, which employs the target classifier in DiMP [6] and the probabilistic bounding-box regression from [30], together with improved training settings.

We use the base tracker to predict the target score map s for the current frame and extract the target candidates v_i by finding locations in s with high target score. Then, we extract a set of features for each candidate. Namely: target classifier score s_i, location $\mathbf{c}_i$ in the image, and an appearance cue $\mathbf{f}_i$ based on the backbone features of the base tracker. Then, we encode this set of features into a single feature vector $\mathbf{z}_i$ for each candidate. We feed these representations and the equivalent ones of the previous frame – already extracted beforehand – into the candidate embedding network and process them together to obtain the enriched embeddings $\mathbf{h}_i$ for each candidate. These feature embeddings are used to compute the similarity matrix $\boldsymbol{S}$ and to estimate the candidate assignment matrix $\boldsymbol{A}$ between the two consecutive frames using an optimal matching strategy.

Once having the candidate-to-candidate assignment probabilities estimated, we build the set of currently visible objects in the scene $\mathcal{O}$ and associate them to the previously identified objects $\mathcal{O}'$, i. e., we determine which objects disappeared, newly appeared, or stayed visible and can be associated unambiguously. We then use this propagation strategy to reason about the target object $\hat{o}$ in the current frame. Additionally, we compute the target detection confidence β to manage the memory and control the sample weight, while updating the target classifier online.

3.3.2 *Problem Formulation*

Let the set of *target candidates*, which includes distractors and the sought target, be $\mathcal{V} = \{v_i\}_{i=1}^{N}$, where N denotes the number of candidates present in each frame. We define the target candidate sets $\mathcal{V}'$ and $\mathcal{V}$ corresponding to the previous and current frames, respectively. We formulate the problem of target candidate association across two subsequent frames as, finding the assignment matrix $\boldsymbol{A}$ between the two sets $\mathcal{V}'$ and $\mathcal{V}$. If the target candidate v_i' corresponds to v_j then $\boldsymbol{A}_{i,j} = 1$ and $\boldsymbol{A}_{i,j} = 0$ otherwise.

In practice, a match may not exist for every candidate. Therefore, we introduce the concept of dustbins, which is commonly used for graph matching [34, 93] to actively handle the non-matching vertices. The idea is to match the candidates without match to the dustbin on the missing side. Therefore, we augment the assignment matrix $\boldsymbol{A}$ by an additional row and column representing dustbins. It follows that a newly appearing candidate v_j – which is only present in the set $\mathcal{V}$ – leads to the entry $\boldsymbol{A}_{N'+1,j} = 1$. Similarly, a candidate v_i' that is no longer available in the set $\mathcal{V}$ results in $\boldsymbol{A}_{i,N+1} = 1$. To solve the assignment problem, we design a learnable approach that predicts the matrix $\boldsymbol{A}$. Our approach first extracts a representation of the target candidates, which is discussed below.

3.3.3 *Target Candidate Extraction*

Here, we describe how to detect and represent target candidates and propose a set of features and their encoding. We define the set of target candidates $\mathcal{V}$ as all unique coordinates $\mathbf{c}_i$ that correspond to a local maximum with minimal score in the target score map s. Thus, each target candidate v_i and its coordinate $\mathbf{c}_i$ need to fulfill the following two constraints,

$$\phi_{\max}(s, \mathbf{c}_i) = 1 \quad \text{and} \quad s(\mathbf{c}_i) \geq \tau, \tag{3.1}$$

where $\phi_{\max}$ returns 1 if the score at $\mathbf{c}_i$ is a local maximum of s or 0 otherwise, and τ denotes a threshold. This definition allows us to

build the sets $\mathcal{V}'$ and $\mathcal{V}$, by retrieving the local maxima of s' and s with sufficient score value. We use the max-pooling operation in a 5×5 local neighbourhood to retrieve the local maxima of s and set $\tau = 0.05$.

For each candidate we build a set of features inspired by two observations. First, we notice that the motion of the same objects from frame to frame is typically small and thus similar locations and similar distances between different objects. Therefore, the position $\mathbf{c}_i$ of a target candidate forms a strong cue. In addition, we observe only small changes in appearance for each object. Therefore, we use the target classifier score $s_i = s(\mathbf{c}_i)$ as another cue. In order to add a more discriminative appearance based feature $\mathbf{f}_i = \mathbf{f}(\mathbf{c}_i)$, we process the backbone features (used in the baseline tracker) with a single learnable convolution layer. Finally, we build a feature tuple for each target candidate as $(s_i, \mathbf{f}_i, \mathbf{c}_i)$. These features are combined in the following way,

$$\mathbf{z}_i = \mathbf{f}_i + \psi(s_i, \mathbf{c}_i), \quad \forall v_i \in \mathcal{V},$$

where ψ denotes a Multi-Layer Perceptron (MLP), that maps s and $\mathbf{c}$ to the same dimensional space as $\mathbf{f}_i$. This encoding permits jointly reasoning about appearance, target similarity, and position.

3.3.4 *Candidate Embedding Network*

In order to further enrich the encoded features and in particular to facilitate extracting features while being aware of neighbouring candidates, we employ a candidate embedding network. On an abstract level, our association problem bares similarities with the task of sparse feature matching. In order to incorporate information of neighbouring candidates, we thus take inspiration from recent advances in this area. In particular, we adopt the SuperGlue [93] architecture that establishes the current state-of-the-art in sparse feature matching. Its design allows to exchange information between different nodes, to handle occlusions, and to estimate the assignment of nodes across two images. In particular, the features of both frames translate to

nodes of a single complete graph with two types of directed edges: 1) self edges within the same frame and 2) cross edges connecting only nodes between the frames. The idea is then to exchange information either along self or cross edges.

The adopted architecture [93] uses a Graph Neural Network (GNN) with message passing that sends the messages in an alternating fashion across self or cross edges to produce a new feature representation for each node after every layer. Moreover, an attention mechanism computes the messages using self attention for self edges and cross attention for cross edges. After the last message passing layer a linear projection layer extracts the final feature representation $\mathbf{h}_i$ for each candidate v_i.

3.3.5 *Candidate Matching*

To represent the similarities between candidates $v'_i \in \mathcal{V}'$ and $v_j \in \mathcal{V}$, we construct the similarity matrix $\boldsymbol{S}$. The sought similarity is measured using the scalar product: $\boldsymbol{S}_{i,j} = \langle \mathbf{h}'_i, \mathbf{h}_j \rangle$, for feature vectors $\mathbf{h}'_i$ and $\mathbf{h}_j$ corresponding to the candidates v'_i and v_j.

As previously introduced, we make use of the dustbin-concept [34, 93] to actively match candidates that miss their counterparts to the so-called dustbin. However, a dustbin is a virtual candidate without any feature representation $\mathbf{h}_i$. Thus, the similarity score is not directly predictable between candidates and the dustbin. A candidate corresponds to the dustbin, only if its similarity scores to all other candidates are sufficiently low. In this process, the similarity matrix $\boldsymbol{S}$ represents only an initial association prediction between candidates disregarding the dustbins. Note that a candidate corresponds either to an other candidate or to the dustbin in the other frame. When the candidates v'_i and v_j are matched, both constraints $\sum_{i=1}^{N'} \boldsymbol{A}_{i,j} = 1$ and $\sum_{j=1}^{N} \boldsymbol{A}_{i,j} = 1$ must be satisfied for one-to-one matching. These constraints however, do not apply for missing matches since multiple candidates may correspond to the same dustbin. Therefore, we make use of two new constraints for dustbins. These constraints

for dustbins read as follows: all candidates not matched to another candidate must be matched to the dustbins. Mathematically, this can be expressed as, $\sum_j \boldsymbol{A}_{N'+1,j} = N - M$ and $\sum_i \boldsymbol{A}_{i,N+1} = N' - M$, where $M = \sum_{(i \leq N', j \leq N)} \boldsymbol{A}_{i,j}$ represents the number of candidate-to-candidate matching. In order to solve the association problem, using the discussed constraints, we follow Sarlin *et al*. [93] and use the Sinkhorn [20, 96] based algorithm therein.

3.3.6 *Learning Candidate Association*

Training the embedding network that parameterizes the similarity matrix used for optimal matching requires ground truth assignments. Hence, in the domain of sparse keypoint matching, recent learning based approaches leverage large scale datasets [36, 93] such as MegaDepth [69] or ScanNet [21], that provide ground truth matches. However, in tracking such ground truth correspondences (between distractor objects) are not available. Only the target object and its location provide a single ground truth correspondence. Manually annotating correspondences for distracting candidates, identified by a tracker on video datasets, is expensive and may not be very useful. Instead, we propose a novel training approach that exploits, (i) partial supervision from the annotated target objects, and (ii) self-supervision by artificially mimicking the association problem. Our approach requires only the annotations that already exist in standard tracking datasets. The candidates for matching are obtained by running the base tracker on the given training dataset.

Partially Supervised Loss: For each pair of consecutive frames, we retrieve the two candidates corresponding to the annotated target, if available. This correspondence forms a partial supervision for a single correspondence while all other associations remain unknown. For the retrieved candidates v'_i and v_j, we define the association as a tuple $(l', l) = (i, j)$. Here, we also mimic the association for redetections and occlusions by occasionally excluding one of the corresponding candidates from $\mathcal{V}'$ or $\mathcal{V}$. We replace the excluded candidate by the

corresponding dustbin to form the correct association for supervision. More precisely, the simulated associations for redetection and occlusion are expressed as, $(l', l) = (N' + 1, j)$ and $(l', l) = (i, N + 1)$, respectively. The supervised loss, for each frame-pairs, is then given by the negative log-likelihood of the assignment probability,

$$L_{\text{sup}} = -\log \boldsymbol{A}_{l',l}. \tag{3.2}$$

Self-Supervised Loss: To facilitate the association of distractor candidates, we employ a self-supervision strategy. The proposed approach first extracts a set of candidates $\mathcal{V}'$ from any given frame. The corresponding candidates for matching, say $\mathcal{V}$, are identical to $\mathcal{V}'$ but we augment its features. Since the feature augmentation does not affect the associations, the initial ground-truth association set is given by $\mathcal{C} = \{(i, i)\}_{i=1}^{N}$. In order to create a more challenging learning problem, we simulate occlusions and redetections as described above for the partially supervised loss. Note that the simulated occlusions and redetections change the entries of $\mathcal{V}$, $\mathcal{V}'$, and $\mathcal{C}$. We make use of the same notations with slight abuse for simplicity. Our feature augmentation involves, randomly translating the location $\mathbf{c}_i$, increasing or decreasing the score s_i, and transforming the given image before extracting the visual features $\mathbf{f}_i$. Now, using the simulated ground-truth associations $\mathcal{C}$, our self-supervised loss is given by,

$$L_{\text{self}} = \sum_{(l',l)\in\mathcal{C}} -\log \boldsymbol{A}_{l',l}. \tag{3.3}$$

Finally, we combine both losses as $L_{\text{tot}} = L_{\text{sup}} + L_{\text{self}}$. It is important to note that the real training data is used only for the former loss function, whereas synthetic data is used only for the latter one.

Data Mining: Most frames contain a candidate corresponding to the target object and are thus applicable for supervised training. However, a majority of these frames are not very informative for training because they contain only a single candidate with high target classifier score and correspond to the target object. Conversely, the dataset contains adverse situations where associating the candidate

corresponding to the target object is very challenging. Such situations include sub-sequences with different number of candidates, with changes in appearance or large motion between frames. Thus, sub-sequences where the appearance model either fails and starts to track a distractor or when the tracker is no longer able to detect the target with sufficient confidence are valuable for training. However, such failure cases are rare even in large scale datasets. Similarly, we prefer frames with many target candidates when creating synthetic sub-sequences to simultaneously include candidate associations, re-detections and occlusions. Thus, we mine the training dataset using the dumped predictions of the base tracker to use more informative training samples.

Training Details: We first retrain the base tracker SuperDiMP without the learned discriminative loss parameters but keep everything else unchanged. We split the LaSOT training set into a *train-train* and a *train-val* set. We run the base tracker on all sequences and save the search region and score map for each frame on disk. We use the dumped data to mine the dataset and to extract the target candidates and its features. We freeze the weights of the base tracker during training of the proposed network and train for 15 epochs by sampling 6400 sub-sequences per epoch from the *train-train* split. We sample real or synthetic data equally likely. We use ADAM [58] with learning rate decay of 0.2 every 6*th* epoch with a learning rate of 0.0001. We use two GNN Layers and run 10 Sinkhorn iterations.

3.3.7 *Object Association*

This part focuses on using the estimated assignments (see Sec. 3.3.5) in order to determine the object correspondences during online tracking. An object corresponds either to the target or a distractor. The general idea is to keep track of every object present in each scene over time. We implement this idea with a database $\mathcal{O}$, where each entry corresponds to an object o that is visible in the current frame. Fig. 3.3 shows these objects as circles. An object disappears from the

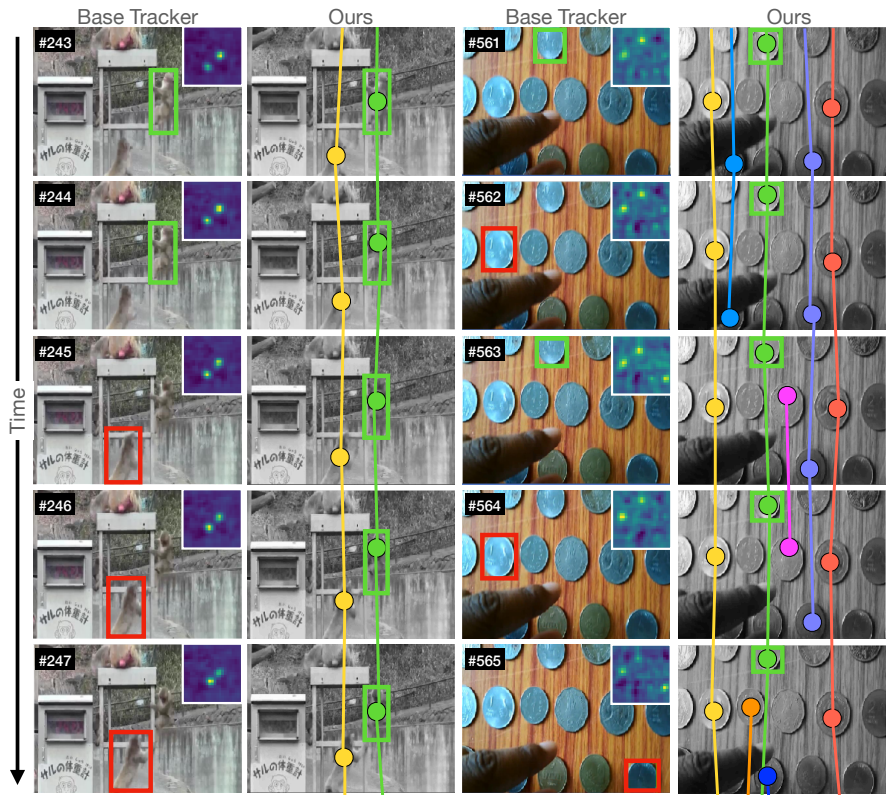


FIGURE 3.3: Visual comparison of the base tracker and our tracker. The bounding boxes represent the tracker result, green [■] indicates correct detections and red [■] refers to tracker failure. Each circle represents an object. Circles with the same color are connected to indicate that the object-ids are identical. If a target candidate cannot be matched with an existing object we add a new object (●, ●, ●). Similarly, we delete the object if no candidate corresponds to it anymore in the next frame (●, ●, ●).

scene if none of the current candidates is associated with it, e. g., in Fig. 3.3 the purple and pink objects (●, ●) no longer correspond to a candidate in the last frame. Then, we delete this object from the database. Similarly, we add a new object to the database if a new target candidate appears (●, ●, ●). When initializing a new object, we assign it a new object-id (not used previously) and the score s_i. In Fig. 3.3 object-ids are represented using colors. For objects that remain visible, we add the score s_i of the corresponding candidate to the history of scores of this object. Furthermore, we delete the old and create a new object if the candidate correspondence is ambiguous, i. e., the assignment probability is smaller than $\omega = 0.75$.

If associating the target object $\hat{o}$ across frames is unambiguous, it implies that one object has the same object-id as the initially provided object $\hat{o}_{\text{init}}$. Thus, we return this object as the selected target. However, in real world scenarios the target object gets occluded, leaves the scene or associating the target object is ambiguous. Then, none of the candidates corresponds to the sought target and we need to redetect. We redetect the target if the candidate with the highest target classifier score achieves a score that exceeds the threshold $\eta = 0.25$. We select the corresponding object as the target as long as no other candidate achieves a higher score in the current frame. Then, we switch to this candidate and declare it as target if its score is higher than any score in the history (of the currently selected object). Otherwise, we treat this object as a distractor for now, but if its score increases high enough, we will select it as the target object in the future. Please refer to the appendix Sec: 3.6.B.1 for the detailed algorithm.

3.3.8 *Memory Sample Confidence*

While updating the tracker online is often beneficial, it is disadvantageous if the training samples have a poor quality. Thus, we describe a memory sample confidence score, that we use to decide which sample to keep in the memory and which should be replaced when

employing a fixed size memory. In addition, we use the score to control the contribution of each training sample when updating the tracker online. In contrast, the base tracker replaces frames using a first-in-first out policy if the target was detected and weights samples during inference solely based on age.

First, we define the training samples in frame k as (x_k, y_k). We assume a memory size m that stores samples from frame $k \in \{1, \dots, t\}$, where t denotes the current frame number. The online loss then given by,

$$J(\theta) = \lambda R(\theta) + \sum_{k=1}^{t} \alpha_k \beta_k Q(\theta; x_k, y_k), \tag{3.4}$$

where Q denotes the data term, R the regularisation term, λ is a scalar and θ represents appearance model weights. The weights $\alpha_k \geq 0$ control the impact of the sample from frame k, i. e., a higher value increases the influence of the corresponding sample during training. We follow other appearance based trackers [6, 23] and use a learning parameter $\gamma \in [0, 1]$ in order to control the weights $\alpha_k = (1 - \gamma)\alpha_{k+1}$, such that older samples achieve a smaller value and their impact during training decreases. In addition, we propose a second set of weights β_k that represent the confidence of the tracker that the predicted label y_k is correct. Instead of removing the oldest samples to keep the memory fixed [6], we propose to drop the sample that achieves the smallest score $\alpha_k \beta_k$ which combines age and confidence. Thus, if $t > n$ we remove the sample at position $k = \text{argmin}_{1 \leq k \leq n} \alpha_k \beta_k$ by setting $\alpha_k = 0$. This means, that if all samples achieve similar confidence the oldest is replaced, or that if all samples are of similar age the least confident sample is replaced.

We describe the extraction of the confidence weights as,

$$\beta_t = \begin{cases} \sqrt{\sigma}, & \text{if } \hat{o} = \hat{o}_{\text{init}} \\ \sigma, & \text{otherwise,} \end{cases} \tag{3.5}$$

where $\sigma = \max_i s_i^t$ denotes the maximum value of the target classifier score map of frame t. For simplicity, we assume that $\sigma \in [0, 1]$.

The condition $\hat{o} = \hat{o}_{\text{init}}$ is fulfilled if the currently selected object is identical to the initially provided target object, i. e., both objects share the same object id. Then, it is very likely, that the selected object corresponds to the target object such that we increase the confidence using the square root function that increases values in the range $[0, 1)$. Hence, the described confidence score combines the confidence of the target classifier with the confidence of the object association module, but fully relies on the target classifier once the target is lost.

Inference details: We propose *KeepTrack* and the speed optimized *KeepTrackFast*. We use the SuperDiMP parameters for both trackers but increase the search area scale from 6 to 8 (from 352 to 480 in image space) for *KeepTrack*. For the fast version we keep the original scale but reduce the number of bounding box refinement steps from 10 to 3. In addition, we skip running the association module if only one target candidate with a high score is present in the previous and current frame. Overall, both trackers follow the target longer until it is lost such that small search areas occur frequently. Thus, we reset the search area to its previous size if it was drastically decreased before the target was lost, to facilitate redetections. Please refer to the appendix 3.6.B for more details.

3.4 EXPERIMENTS

We evaluate our proposed tracking architecture on seven benchmarks. Our approach is implemented in Python using PyTorch. On a single Nvidia GTX 2080Ti GPU, *KeepTrack* and *KeepTrackFast* achieve 18.3 and 29.6 FPS, respectively.

3.4.1 *Ablation Study*

We perform an extensive analysis of the proposed tracker, memory sample confidence, and training losses.

Target association network components: We evaluate the target candidate association network with different numbers of Sinkhorn [96]

Num GNN Layers	Num Sinkhorn iterations	NFS	UAV123	LaSOT	FPS
–	–	65.2	69.1	65.8	–
0	50	65.9	69.2	66.6	–
2	10	66.4	69.7	67.1	18.3
9	50	66.4	69.8	67.2	12.7

TABLE 3.1: Impact of each component of the Target Candidate Association Network in terms of AUC (%) on three datasets.

Memory Sample Confidence	Search area Adaptation	Target Candidate Association Network	NFS	UAV123	LaSOT
–	–	–	64.4	68.2	63.5
✓	–	–	64.7	68.0	65.0
✓	✓	–	65.2	69.1	65.8
✓	✓	✓	**66.4**	**69.7**	**67.1**

TABLE 3.2: Impact of each component in terms of AUC (%) on three datasets. The first row corresponds to our SuperDiMP baseline.

iterations and with different number of GNN layers of the embedding network or dropping it at all, see Tab. 3.1. We conclude, that using the target candidate association network even without any GNN layers outperforms the baseline on all three datasets. In addition, using either two or nine GNN layers improves the performance even further on all datasets. We achieve the best results when using nine GNN layers and 50 Sinkhorn iterations. However, using a large candidate embedding network and a high number of Sinkhorn iterations reduces the run-time of the tracker to 12.7 FPS. Hence, using only two GNN layers and 10 Sinkhorn iterations results in a negligible decrease of 0.1 on UAV123 and LaSOT but accelerates the run-time by 44%.

Online tracking components: We study the importance of memory

Loss	no TCA	L_{sup}	L_{self}	$L_{\text{sup}} + L_{\text{self}}$	$L_{\text{sup}} + L_{\text{self}}$
Data-mining	n.a.	✓	✓	-	✓
LaSOT, AUC (%)	65.8	66.0	66.9	66.8	**67.1**

TABLE 3.3: Analysis on LaSOT of association learning using different loss functions with and without data-mining.

Sample Replacement with conf. score	Online updating with conf. score	Conf. score threshold	LaSOT AUC (%)
–	–	–	63.5
✓	–	–	64.1
✓	✓	0.0	64.6
✓	✓	0.5	65.0

TABLE 3.4: Analysis of our memory weighting component on LaSOT.

sample confidence, the search area protocol, and target candidate association of our final method *KeepTrack*. In Tab. 3.2 we analyze the impact of successively adding each component, and report the average of five runs on the NFS [42], UAV123 [84] and LaSOT [38] datasets. The first row reports the results of the employed base tracker. First, we add the memory sample confidence approach (Sec. 3.3.8), observe similar performance on NFS and UAV but a significant improvement of 1.5% on LaSOT, demonstrating its potential for long-term tracking. With the added robustness, we next employ a larger search area and increase it if it was drastically shrank before the target was lost. This leads to a fair improvement on all datasets. Finally, we add the target candidate association network, which provides substantial performance improvements on all three datasets, with a +1.3% Area Under the Curve (AUC) on LaSOT. These results clearly demonstrate the power of the target candidate association network.

Training: In order to study the effect of the proposed training losses, we retrain the target candidate association network either

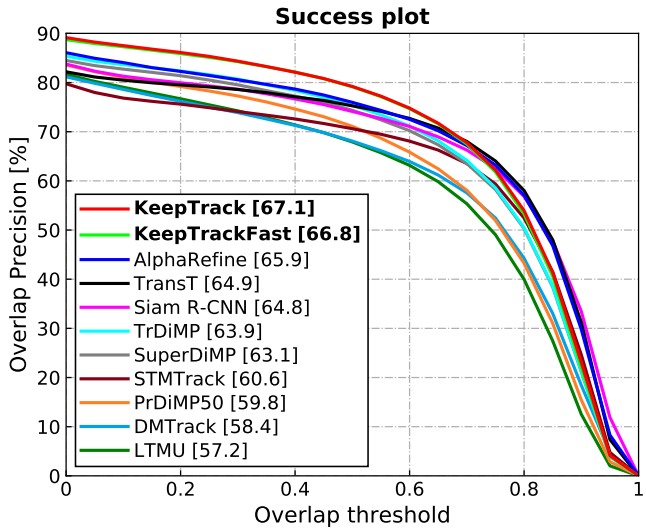
with only the partially supervised loss or only the self-supervised loss. We report the performance on LaSOT [38] in Tab. 3.3. The results show that each loss individually allows to train the network and to outperform the baseline without the target candidate association network (no TCA), whereas, combining both losses leads to the best tracking results. In addition, training the network with the combined loss but without data-mining decreases the tracking performance.

Memory management: We not only use the sample confidence to manage the memory but also to control the impact of samples when learning the target classifier online. In Tab. 3.4, we study the importance of each component by adding one after the other and report the results on LaSOT [38]. First, we use the sample confidence scores only to decide which sample to remove next from the memory. This, already improves the tracking performance. Reusing these weights when learning the target classifier as described in Eq. (3.4) increases the performance again. To suppress the impact of poor-quality samples during online learning, we ignore samples with a confidence score bellow 0.5. This leads to an improvement on LaSOT. The last row corresponds to the used setting in the final proposed tracker.

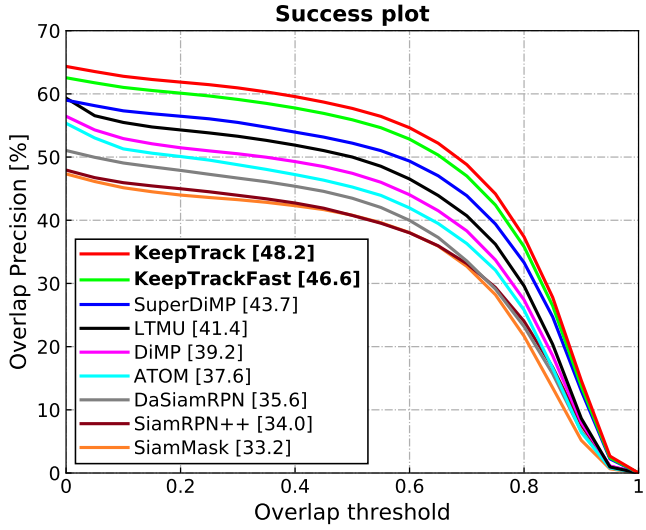
3.4.2 *State-of-the-art Comparison*

We compare our approach on seven tracking benchmarks. The same settings and parameters are used for all datasets. In order to ensure the significance of the results, we report the average over five runs on all datasets unless the evaluation protocol requires otherwise. We recompute the results of all trackers using the raw predictions if available or otherwise report the results given in the paper.

LaSOT [38]: First, we compare on the large-scale LaSOT dataset (280 test sequences with 2500 frames in average) to demonstrate the robustness and accuracy of the proposed tracker. The success plot in Fig. 3.4a shows the overlap precision OP_T as a function of the threshold T. Trackers are ranked w. r. t. their AUC score, denoted



(a) LaSOT [38]



(b) LaSOTExtSub [37]

FIGURE 3.4: Success plots, showing OP_T, on LaSOT [38] and LaSOTExtSub [37]. Our approach outperforms all other methods by a large margin in AUC, reported in the legend.

	Keep Track	**Keep Track Fast**	Alpha Refine [118]	TransT [14]	Siam R-CNN [104]	TrDiMP [106]	Super Dimp [24]	STM Track [41]	Pr DiMP [30]	DM Track [129]	LTMU [22]	DiMP [6]	Ocean [128]
Precision	70.2	70.0	68.0	69.0	68.4	66.3	65.3	63.3	60.8	59.7	57.2	56.7	56.6
Norm. Prec	77.2	77.0	73.2	73.8	72.2	73.0	72.2	69.3	68.8	66.9	66.2	65.0	65.1
Success (AUC)	67.1	66.8	65.3	64.9	64.8	63.9	63.1	60.6	59.8	58.4	57.2	56.9	56.0

TABLE 3.5: State-of-the-art comparison on the LaSOT [38] test set in terms of AUC score.

	KeepTrack	**KeepTrack Fast**	AlphaRefine [118]	LTMU [22]	TrDiMP [106]	KYS [7]	SuperDiMP [30]
Precision	70.2	70.0	68.0	66.5	61.4	64.0	65.3
Norm. Prec.	77.2	77.0	73.2	73.7	–	70.7	72.2
Success (AUC)	67.1	66.8	65.3	64.7	63.9	61.9	63.1

TABLE 3.6: Results on the LaSOT [38] test set. All trackers use the same base tracker SuperDiMP [24].

in the legend. Tab. 3.5 shows more results including precision and normalized precision. *KeepTrack* and *KeepTrackFast* outperform the recent trackers AlphaRefine [118], TransT [14] and TrDiMP [106] by a large margin and the base tracker SuperDiMP by 4.0% or 3.7% in AUC. The improvement in OP_T is most prominent for thresholds $T < 0.7$, demonstrating the superior robustness of our tracker. In Tab. 3.6, we further perform an apple-to-apple comparison with KYS [7], LTMU [22], AlphaRefine [118] and TrDiMP [106], where all methods use SuperDiMP as base tracker. We outperform the best method on each metric, achieving an AUC improvement of 1.8%.

LaSOTExtSub [37]: We evaluate our tracker on the recently published extension subset of LaSOT. LaSOTExtSub is meant for testing only and consists of 15 new classes with 10 sequences each. The sequences are long (2500 frames on average), rendering substantial challenges. Fig. 3.4b shows the success plot, that is averaged over 5 runs. All results, except ours and SuperDiMP, are obtained from [37], e. g., DaSiamRPN [133], SiamRPN++ [67] and SiamMask [107]. Our

	Keep Track	**Keep Track Fast**	LTMU [22]	Super DiMP [24]	Siam R-CNN [104]	TACT [17]	DM Track [129]	SPLT [119]	Global Track [53]	MBMD [127]	Siam FC+R [102]	TLD [55]
TPR	80.6	**82.7**	74.9	79.7	70.1	**80.9**	68.6	49.8	57.4	60.9	42.7	20.8
TNR	**81.2**	77.2	75.4	70.2	74.5	62.2	69.4	77.6	63.3	48.5	48.1	**89.5**
MaxGM	**80.9**	**79.9**	75.1	74.8	72.3	70.9	68.8	62.2	60.3	54.4	45.4	43.1

TABLE 3.7: Results on the OxUvALT [102] test set in terms of TPR, TNR, and the max geometric mean MaxGM of TPR and TNR.

	Keep Track	**Keep Track Fast**	LTMU [22]	Siam R-CNN [104]	Siam PGNet [71]	Siam RPN++ [67]	Super DiMP [24]	SPLT [119]	MBMD [127]	DaSiamLT [60, 133]
Precision	**73.8**	70.1	**71.0**	–	67.9	64.9	64.3	63.3	63.4	62.7
Recall	**70.4**	67.6	**67.2**	–	61.0	60.9	61.0	60.0	58.8	58.8
F1-Score	**72.0**	68.8	**69.0**	66.8	64.2	62.9	62.2	61.6	61.0	60.7

TABLE 3.8: Results on the VOT2018LT dataset [60] in terms of F1-Score, Precision and Recall.

method achieves superior results, outperforming LTMU by 6.8% and SuperDiMP by 3.5%.

OxUvALT [102]: The OxUvA long-term dataset contains 166 test videos with average length 3300 frames. Trackers are required to predict whether the target is present or absent in addition to the bounding box for each frame. Trackers are ranked by the Maximum Geometric Mean (MaxGM) of the True Positive Rate (TPR) and the True Negative Rate (TNR). We use the proposed confidence score and set the threshold for target presence using the separate dev. set. Tab. 3.7 shows the results on the test set, which are obtained through the evaluation server. *KeepTrack* sets the new state-of-the-art in terms of MaxGM by achieving an improvement of 5.8% compared to the previous best method and exceed the result of the base tracker SuperDiMP by 6.1%.

VOT2018LT [60]: Next, we evaluate our tracker on the 2018 edition of the VOT [62] long-term tracking challenge. We compare with the top methods in the challenge [60], as well as more recent methods. The

	Keep Track	**Keep Track Fast**	LT_DSE [61, 63]	LTMU_B [22, 61]	Mega track [61]	CLGS [61, 63]	RLT DiMP [61]	Super DiMP [24]	Siam DW_LT [61, 63]	ltMBNet [61]
Precision	**72.3**	70.6	71.5	70.1	70.3	**73.9**	65.7	67.6	69.7	64.9
Recall	**69.7**	68.0	67.7	68.1	67.1	61.9	68.4	66.3	63.6	51.4
F1-Score	**70.9**	69.3	**69.5**	69.1	68.7	67.4	67.0	66.9	66.5	57.4

TABLE 3.9: Results on the VOT2019LT [63]/VOT2020LT [61] dataset in terms of F1-Score, Precision and Recall.

	Keep Track	**Keep Track Fast**	CRACT [39]	TrDiMP [106]	TransT [14]	Super DiMP [24]	Pr DiMP [30]	STM Track [41]	Siam AttN [124]	Siam R-CNN [104]	KYS [7]	DiMP [6]
UAV123	**69.7**	**69.5**	66.4	67.5	69.1	68.1	68.0	64.7	65.0	64.9	–	65.3
OTB-100	70.9	71.2	**72.6**	71.1	69.4	70.1	69.6	**71.9**	71.2	70.1	69.5	68.4
NFS	**66.4**	65.3	62.5	**66.2**	65.7	64.7	63.5	–	–	63.9	63.5	62.0

TABLE 3.10: Comparison with state-of-the-art on the OTB-100 [109], NFS [42] and UAV123 [84] datasets in terms of AUC score.

dataset contains 35 videos with 4200 frames per sequence on average. Trackers are required to predict a confidence score that the target is present in addition to the bounding box for each frame. Trackers are ranked by the F1-score, evaluated for a range of confidence thresholds. As shown in Tab. 3.8, our tracker achieves the best results in all three metrics and outperforms the base tracker SuperDiMP by almost 10% in F1-score.

VOT2019LT [63]/VOT2020LT [61]: The dataset for both VOT [62] long-term tracking challenges contains 215,294 frames divided in 50 sequences. Trackers need to predict a confidence score that the target is present and the bounding box for each frame. Trackers are ranked by the F-score, evaluated for a range of confidence thresholds. We compare with the top methods in the challenge [61, 63], as well as more recent methods. As shown in Tab. 3.9, our tracker achieves the best result in terms of F-score and outperforms the base tracker SuperDiMP by 4.0% in F-score.

UAV123 [84]: This dataset contains 123 videos and is designed to assess trackers for UAV applications. It contains small objects, fast motions, and distractor objects. Tab. 3.10 shows the results, where the entries correspond to AUC for OP_T over Intersection over Union (IoU) thresholds T. Our method sets a new state-of-the-art with an AUC of 69.7%, exceeding the performance of the recent trackers TransT [14] and TrDiMP [106] by 0.6% and 2.2% in AUC.

OTB-100 [109]: For reference, we also evaluate our tracker on the OTB-100 dataset consisting of 100 sequences. Several trackers achieve tracking results over 70% in AUC, as shown in Tab. 3.10. So do *KeepTrack* and *KeepTrackFast* that perform similarly to the top methods, with a 0.8% and 1.1% AUC gain over the SuperDiMP baseline.

NFS [42]: Lastly, we report results on the 30 FPS version of the Need for Speed (NFS) dataset. It contains fast motions and challenging distractors. Tab. 3.10 shows that our approach sets a new state of the art on NFS.

3.5 CONCLUSION

We propose a novel tracking pipeline employing a learned target candidate association network in order to track both the target and distractor objects. This approach allows us to propagate the identities of all target candidates throughout the sequence. In addition, we propose a training strategy that combines partial annotations with self-supervision. We do so to compensate for lacking ground-truth correspondences between distractor objects in visual tracking. We conduct comprehensive experimental validation and analysis of our approach on seven generic object tracking benchmarks and set a new state-of-the-art on six.

3.6 APPENDICES

In the appendix, we first provide details about training in Sec. 3.6.A and about inference in Sec. 3.6.B. We then provide failure cases in Sec. 3.6.C and an attribute-wise comparison in Sec. 3.6.D.

3.6.A *Training*

First, we describe the training data generation and sample selection to train the network more effectively. Then, we provide additional details about the training procedure such as training in batches, augmentations and synthetic sample generation. Finally, we briefly summarize the employed network architecture.

3.6.A.1 *Data-Mining*

We use the LaSOT [38] training set to train our target candidate association network. In particular, we split the 1120 training sequences randomly into a *train-train* (1000 sequences) and a *train-val* (120 sequences) set. We run the base tracker on all sequences and store the target classifier score map and the search area on disk for each frame. During training, we use the score map and the search area to extract the target candidates and its features to provide the data to train the target candidate association network.

We observed that many sequences or sub-sequences contain mostly one target candidate with a high target classifier score. Thus, in this cases target candidate association is trivial and learning on these cases will be less effective. Conversely, tracking datasets contain sub-sequences that are very challenging (large motion or appearance changes or many distractors) such that trackers often fail. While these sub-sequences lead to a more effective training they are relatively rare such that we decide to actively search the training dataset.

First, we assign each frame to one of six different categories. We classify each frame based on four observations about the number of candidates, their target classifier score, if one of the target candidates

Name	Number of candidates	Is a candidate selected as target? $\max(s_i) \geq \eta$	Does the candidate with max score correspond to the target?	Does any candidate correspond to the target?	Num Frames	Ratio
D	1	✓	✓	✓	1.8M	67.9%
H	> 1	✓	✓	✓	498k	18.4%
G	> 1	x	✓	–	8k	0.3%
J	> 1	✓	x	x	76k	2.8%
K	> 1	✓	x	✓	42k	1.5%
other	–	–	–	–	243k	9.1%

TABLE 3.11: Categories and specifications for each frame in the training dataset used for data-mining.

is selected as target and if this selection is correct, see Tab. 3.11. A candidate corresponds to the annotated target object if the spatial distance between the candidate location and center coordinate of the target object is smaller than a threshold.

Assigning each frame to the proposed categories, we observe, that the dominant category is D (70%) that corresponds to frames with a single target candidate matching the annotated target object. However, we favour more challenging settings for training. In order to learn distractor associations using self supervision, we require frames with multiple detected target candidates. Category H (18.4%) corresponds to such frames where in addition the candidate with the highest target classifier score matches the annotated target object. Hence, the base tracker selects the correct candidate as target. Furthermore, category G corresponds to frames where the base tracker was no longer able to track the target because the target classifier score of the corresponding candidate fell bellow a threshold. We favour these frames during training in order to learn continue tracking the target even if the score is low.

Both categories J and K correspond to tracking failures of the base tracker. Whereas in K the correct target is detected but not selected, it is not detected in frames of category J. Thus, we aim to learn from

tracking failures in order to train the target candidate association network such that it learns to compensate for tracking failures of the base tracker and corrects them. In particular, frames of category K are important for training because the two candidates with highest target classifier score no longer match such that the network is forced to include other cues for matching. We use frames of category J because frames where the object is visible but not detected contain typically many distractors such that these frames are suitable to learn distractor associations using self-supervised learning.

To summarize, we select only frames with category H, K, J for self-supervised training and sample them with a ratio of 2 : 1 : 1 instead of 10 : 2 : 1 (ratio in the dataset). We ignore frames from category D during self-supervised training because we require frames with multiple target candidates. Furthermore, we select sub-sequences of two consecutive frames for partially supervised training. We choose challenging sub-sequences that either contain many distractors in each frame (HH, 350k) or sub-sequences where the base tracker fails and switches to track a distractor (HK, 1001) or where the base tracker is no longer able to identify the target with high confidence (HG, 1380). Again we change the sampling ratio from approximately 350 : 1 : 1 to 10 : 1 : 1 during training. We change the sampling ration in order to use failure cases more often during training than they occur in the training set.

3.6.a.2 *Training Data Preparation*

During training we use two different levels of augmentation. First, we augment all features of target candidate to enable self-supervised training with automatically produced ground truth correspondences. In addition, we use augmentation to improve generalization and overfitting of the network.

When creating artificial features we randomly scale each target classifier score, randomly jitter the candidate location within the search area and apply common image transformations to the original image before extracting the appearance based features for the

artificial candidates. In particular, we randomly jitter the brightness, blur the image and jitter the search area before cropping the image to the search area.

To reduce overfitting and improve the generalization, we randomly scale the target candidate scores for synthetic and real sub-sequences. Furthermore, we remove candidates from the sets $\mathcal{V}'$ and $\mathcal{V}$ randomly in order to simulate newly appearing or disappearing objects. Furthermore, to enable training in batches we require the same number of target candidates in each frame. Thus, we keep the five candidates with the highest target classifier score or add artificial peaks at random locations with a small score such that five candidates per frame are present. When computing the losses, we ignore these artificial candidates.

3.6.A.3 *Architecture Details*

We use the SuperDiMP tracker [24] as our base tracker. SuperDiMP employs the DiMP [6] target classifier and the probabilistic bounding-box regression of PrDiMP [30], together with improved training settings. It uses a ResNet-50 [50] pretrained network as backbone feature extractor. We freeze all parameters of SuperDiMP while training the target candidate association network. To produce the visual features for each target candidate, we use the third layer ResNet-50 features. In particular, we obtain a $29 \times 29 \times 1024$ feature map and feed it into a 2×2 convolutional layer which produces the $30 \times 30 \times 256$ feature map $\mathbf{f}$. Note, that the spatial resolution of the target classifier score and feature map agree such that extracting the appearance based features $\mathbf{f}_i$ for each target candidate v_i at location $\mathbf{c}_i$ is simplified.

Furthermore, we use a four layer MLP to project the target classifier score and location for each candidate in the same dimensional space as $\mathbf{f}_i$. We use the following MLP structure: $3 \rightarrow 32 \rightarrow 64 \rightarrow 128 \rightarrow 256$ with batch normalization. Before feeding the candidate locations into the MLP we normalize it according to the image size.

We follow Sarlin *et al*. [93] when designing the candidate embedding network. In particular, we use self and cross attention layers in an alternating fashion and employ two layers of each type. In addition, we append a 1×1 convolutional layer to the last cross attention layer. Again, we follow Sarlin *et al*. [93] for optimal matching and reuse their implementation of the Sinkhorn algorithm and run it for 10 iterations.

3.6.B *Inference*

In this section we provide the detailed algorithm that describes the object association module (Sec. 3.7 in the paper). Furthermore, we explain the idea of search area rescaling at occlusion and how it is implemented. We conclude with additional inference details.

3.6.B.1 *Object Association Module*

Here, we provide a detailed algorithm describing the object association module presented in the main paper, see Alg. 1. It contains target candidate to object association and the redetection logic to retrieve the target object after it was lost.

First, we will briefly explain the used notation. Each object can be modeled similar to a *class* in programming. Thus, each object o contains attributes that can be accesses using the "." notation. In particular $(o_j).s$ returns the score attribute s of object o_j. In total the object class contains two attribute: the list of scores s and the object-id id. Both setting and getting the attribute values is possible.

The algorithm requires the following inputs: the set of target candidates $\mathcal{V}$, the set of detected objects $\mathcal{O}'$ and the object selected as target $\hat{o}$ in the previous frame. First, we check if a target candidate matches with any previously detected object and verify that the assignment probability is higher than a threshold $\omega = 0.75$. If such a match exists, we associate the candidate to the object and append its target classifier score to the scores and add the object to the set of currently visible object $\mathcal{O}$. If a target candidate matches none of the

Algorithm 1 Object Association Algorithm.

Require: Set of target candidates $\mathcal{V}$
Require: Set of objects of previous frame $\mathcal{O}'$
Require: Target object $\hat{o}'$
$\mathcal{O} = \{\}$, $N = |\mathcal{V}|$
for $i = 1, \ldots, N$ **do**
    if matchOf$(v_i) \neq$ dustbin && $p(v_i) \geq \omega$ **then**
        $v'_j \leftarrow$ matchOf(v_i)
        $(o'_j).s \leftarrow$ concat$((o'_j).s, [s_i])$
        $o_i \leftarrow o'_j$
    else
        $o_i \leftarrow$ new Object(getNewId(), $[s_i]$)
    $\mathcal{O} \leftarrow \mathcal{O} \cup \{o_i\}$
if $\hat{o}' \neq$ none **and** $\hat{o}'$.id $\in \{o.\text{id} \mid o \in \mathcal{O}\}$ **then**
    $\hat{o} =$ getObjectById$(\mathcal{O}, \hat{o}'.\text{id})$
    for $i = 1, \ldots, N$ **do**
        if $\max(\hat{o}.s) < (o_i).s[-1]$ **then**
            $\hat{o} = o_i$
else
    $i = \text{argmax}_i\{(o_i).s[-1]) \mid o_i \in \mathcal{O}\}$
    if $(o_i).s[-1] \geq \eta$ **then**
        $\hat{o} = o_i$
    else
        $\hat{o} =$ none
return $\hat{o}$, $\mathcal{O}$

previously detected objects, we create a new object and add it to $\mathcal{O}$. Hence, previously detected objects that miss a matching candidate are not included in $\mathcal{O}$. Once, all target candidates are associated to an already existing or newly created object. We check if the object previously selected as target is still visible in the current scene and forms the new target $\hat{o}$. After the object was lost it is possible that the object selected as target is in fact a distractor. Thus, we select an other object as target if this other object achieves a higher target classifier score in the current frame than any score the currently selected object achieved in the past. Furthermore, if the object previously selected as target object is no longer visible, we try to redetect it by checking if the object with highest target classifier score in the current frame achieves a score higher than a threshold $\eta = 0.25$. If the score is high enough, we select this object as the target.

3.6.B.2 *Search Area Rescaling at Occlusion*

The target object often gets occluded or moves out-of-view in many tracking sequences. Shortly before the target is lost the tracker typically detects only a small part of the target object and estimates a smaller bounding box than in the frames before. The used base tracker SuperDiMP employs a search area that depends on the currently estimated bounding box size. Thus, a partially visible target object causes a small bounding box and search area. The problem of a small search area is that it complicates redetecting the target object, e. g., the target reappears at a slightly different location than it disappeared and if the object then reappears outside of the search area redetection is prohibited. Smaller search areas occur more frequently when using the target candidate association network because it allows to track the object longer until we declare it as lost.

Hence, we use a procedure to increase the search area if it decreased before the target object was lost. First, we store all search are resolutions during tracking in an list a as long as the object is detected. If the object was lost k frames ago, we compute the new search area by averaging the last k entries of a larger than the search

area at occlusion. We average at most over 30 previous search areas to compute the new one. If the target object was not redetected within these 30 frames with keep the search area fixed until redetection.

3.6.B.3 *Inference Details*

In contrast to training, we use all extracted target candidates to compute the candidate associations between consecutive frames. In order to save computations, we extract the candidates and features only for the current frame and cache the results such that they can be reused when computing the associations in the next frames.

KeepTrack Settings: We use the same settings as for SuperDiMP but increase the search area scale from 6 to 8 leading to a larger search are (from 352×352 to 480×480) and to a larger target score map (from 22×22 to 30×30). In addition, we employ the aforementioned search area rescaling at occlusion and skip running the target candidate association network if only one target candidates with high target classifier score is detected in the current and previous frame, in order to save computations.

KeepTrackFast Settings: We use the same settings as for SuperDiMP. In particular, we keep the search area scale and target score map resolution the same to achieve a faster run-time. In addition, we reduce the number of bounding box refinement steps from 10 to 3 which reduces the bounding box regression time significantly. Moreover, we double the target candidate extraction threshold τ to 0.1. This step ensures that we neglegt local maxima with low target classifier scores and thus leads to less frames with multiple detected candidates. Hence, *KeepTrackFast* runs the target candidate association network less often than *KeepTrack*.

3.6.C *Failure Cases*

While *KeepTrack* is particularly powerful when distractor objects appear in the scene, it also fails to track the target object in complex scenes, such as the examples shown in Fig. 3.5.

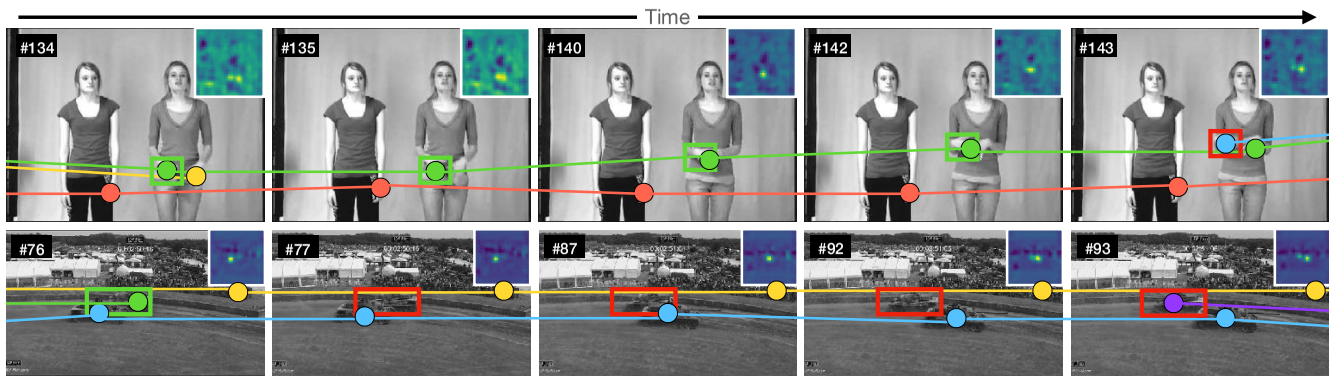


FIGURE 3.5: Failure Cases: a very challenging case is when a distractor crosses the target's location, since positional information is then of limited use. The box represents the ground truth bounding box of the target object, where green indicates the the selected target candidates corresponds to the sought target and red indicates that the tracker selected a candidate corresponding to a distractor object.

The top row shows such a challenging case, where the target object is the right hand of the person on the right (indicated by [■] or [■]). *KeepTrack* manages to individually track all hands (●,●,●) visible in the search area until frame number 134. In the next frame, both hands of the person are close and our tracker only detects one candidate for both hands (●). Thus, the tracker assigns the target id to the remaining candidate. The tracker detects two candidates (●,●) as soon as both hands move apart in frame 143. However, now it is unclear which hand is the sought target. If two objects approach each other it is unclear whether they cross each other or not. In this scenario positional information is of limited use. Hence, deeper understanding of the scene and the target object seems necessary to mitigate such failure cases.

The bottom row in Fig. 3.5 shows a similar failure case where again a distractor object (●) crosses the target's (●) location. This time, the tracker fails to extract the candidate corresponding to the target from frame number 77 on wards. The tracker detects that the target candidate previously assumed to represent the target has vanished but the remaining distractor object (●) achieves such a high target score that the tracker reconsiders its previous target selection and continues tracking the distractor object (●) instead. Thus, the tracker continues tracking the distractor object even if the a target candidate for the sought target (●) appears (frame number 93).

3.6.D *Attributes*

Tabs. 3.12 and 3.13 show the results of various trackers including *KeepTrack* and *KeepTrackFast* based on different sequence attributes. We observe that both trackers are superior to other trackers on UAV123 for most attributes. In particular, we outperform the runner-up by a large margin in terms of AUC on the sequences corresponding to the following attributes: *Aspect Ratio Change* (+2.5/2.6%), Full Occlusion (+1.8/1.9%), Partial Occlusion (+2.5/2.3%), Background Clutter (+1.5/1.3%), Illumination Variation (+1.7/1.5%), Similar Ob-

	Scale Variation	Aspect Ratio Change	Low Resolution	Fast Motion	Full Occlusion	Partial Occlusion	Total
ATOM [23]	63.0	61.9	49.5	62.7	46.2	58.1	64.2
DiMP50 [6]	63.8	62.8	50.9	62.7	47.5	59.7	65.3
STMTrack [41]	63.9	64.2	46.4	62.2	48.9	58.0	65.7
TrDiMP [106]	66.4	66.1	54.3	66.3	48.6	62.1	67.5
SuperDiMP [30]	66.6	66.4	54.9	65.1	52.0	63.5	67.7
PrDiMP50 [30]	66.8	66.3	55.2	65.3	53.6	63.5	68.0
TransT [14]	68.0	66.3	55.6	**67.4**	48.4	63.2	69.1
KeepTrackFast	**68.4**	**68.8**	**57.3**	67.2	**55.4**	**66.0**	**69.5**
KeepTrack	**68.7**	**68.9**	**57.0**	**68.0**	**55.5**	**65.8**	**69.7**

	Out-of-View	Background Clutter	Illumination Variation	Viewpoint Change	Camera Motion	Similar Object	Total
ATOM [23]	61.4	46.2	63.1	65.1	66.4	63.1	64.2
DiMP50 [6]	61.8	48.9	63.9	65.2	66.9	62.9	65.3
STMTrack [41]	**68.2**	46.2	61.9	70.2	67.5	58.0	65.7
TrDiMP [106]	66.3	45.1	61.5	70.0	68.3	64.9	67.5
SuperDiMP [30]	63.7	51.4	63.2	67.8	69.8	65.5	67.7
PrDiMP50 [30]	63.9	53.9	62.4	69.4	70.4	66.1	68.0
TransT [14]	**69.1**	44.1	62.6	**71.8**	70.5	65.3	69.1
KeepTrackFast	65.9	**55.4**	**65.6**	70.3	**71.2**	**67.9**	**69.5**
KeepTrack	66.8	**55.2**	**65.4**	**70.4**	**71.8**	**67.2**	**69.7**

TABLE 3.12: UAV123 attribute-based analysis in terms of AUC score. Each column corresponds to the results computed on all sequences in the dataset with the corresponding attribute.

ject (+1.8/1.1%). Especially, the superior performance on sequences with the attributes *Full Occlusion*, *Partial Occlusion*, *Background Clutter* and *Similar Object* clearly demonstrates that *KeepTrack* mitigates the harmful effect of distractors and allows to track the target object longer and more frequently than other trackers. In addition, Tab. 3.12 reveals that *KeepTrack* achieves the highest (red) or second-highest (blue) AUC on the sequences corresponding to each attribute except for the attribute *Out-of-View*.

The attribute-based analysis on LaSOT allows similar observations. In particular, *KeepTrack* and *KeepTrackFast* outperform all other

	Illumination Variation	Partial Occlusion	Deformation	Motion Blur	Camera Motion	Rotation	Background Clutter	Total
LTMU [22]	56.5	54.0	57.2	55.8	61.6	55.1	49.9	57.2
PrDiMP50 [30]	63.7	56.9	60.8	57.9	64.2	58.1	54.3	59.8
STMTrack [41]	65.2	57.1	64.0	55.3	63.3	60.1	54.1	60.6
SuperDiMP [30]	67.8	59.7	63.4	62.0	68.0	61.4	57.3	63.1
TrDiMP [106]	67.5	61.1	64.4	62.4	68.1	62.4	58.9	63.9
Siam R-CNN [104]	64.6	62.2	65.2	63.1	68.2	64.1	54.2	64.8
TransT [14]	65.2	62.0	**67.0**	63.0	67.2	64.3	57.9	64.9
AlphaRefine [118]	69.4	62.3	66.3	**65.2**	70.0	63.9	58.8	65.3
KeepTrackFast	**70.1**	**63.8**	66.2	65.0	**70.7**	**65.1**	**60.1**	**66.8**
KeepTrack	**69.7**	**64.1**	**67.0**	**66.7**	**71.0**	**65.3**	**61.2**	**67.1**

	Viewpoint Change	Scale Variation	Full Occlusion	Fast Motion	Out-of-View	Low Resolution	Aspect Ration Change	Total
LTMU [22]	56.7	57.1	49.9	44.0	52.7	51.4	55.1	57.2
PrDiMP50 [30]	59.2	59.4	51.3	48.4	55.3	53.5	58.6	59.8
STMTrack [41]	58.2	60.6	47.8	42.4	51.9	50.3	58.8	60.6
SuperDiMP [30]	63.4	62.9	54.1	50.7	59.0	56.4	61.6	63.1
TrDiMP [106]	62.8	63.4	56.4	53.0	60.7	58.1	62.3	63.9
Siam R-CNN [104]	65.3	64.5	55.3	51.5	62.2	57.1	63.4	64.8
TransT [14]	61.7	64.6	55.3	51.0	58.2	56.4	63.2	64.9
AlphaRefine [118]	63.1	65.4	57.4	53.6	61.1	58.6	64.1	65.3
KeepTrackFast	**67.6**	**66.6**	**59.2**	**57.1**	**63.4**	**62.0**	**65.6**	**66.8**
KeepTrack	**66.9**	**66.8**	**60.1**	**57.7**	**64.1**	**62.0**	**65.9**	**67.1**

TABLE 3.13: LaSOT attribute-based analysis. Each column corresponds to the results computed on all sequences in the dataset with the corresponding attribute.

trackers by a large margin in AUC on the sequences corresponding oth the following attributes: *Partial Occlusion* (+1.8/1.5%), *Background Clutter* (+2.3/1.2, *Viewpoint Change* (+1.6/2.3%), *Full Occlusion* (+2.7/1.8%), *Fast Motion* (4.1/3.5%), *Out-of-View* (+1.9/1.2%), *Low Resolution* (+3.4/3.4%). Moreover, the superior performance is even clearer when comparing to the base tracker SuperDiMP, e. g., and improvement of +6.0/5.2% for *Full Occlusion* or +7.0/6.4% for *Fast Motion*. *KeepTrack* achieves the highest AUC score for every attributed except two where *KeepTrackFast* achieves slightly higher scores. Again, the best performance on sequences with attributes such as *Background Clutter* or *Full Occlusion* clearly demonstrates

the effectiveness of our proposed target and distractor association strategy.

4

TRANSFORMING MODEL PREDICTION FOR TRACKING

In the previous chapter we proposed an target candidate association network to explicitly handle distractor objects. In contrast, we introduce in this chapter a more powerful Transformer-based model prediction module than the previously used one, reducing the need of further distractor mitigation approaches. The employed Transformer captures global relations with little inductive bias, allowing it thus to learn the prediction of more powerful target models. We further extend the model predictor to estimate a second set of weights that are applied for accurate bounding box regression. The resulting tracker relies on training and on test frame information in order to predict all weights trasductively. The code and trained models are available at https://github.com/visionml/pytracking.

4.1 INTRODUCTION

Generic visual object tracking is one of the fundamental problems in computer vision. The task involves estimating the state of the target object in every frame of a video sequence, given only the initial target location. One of the key problems in object tracking is learning to robustly detect the target object, given a scarce annotation. Among existing methods, DCF [6, 10, 23, 25, 43, 52, 78, 97] have achieved much success. These approaches learn a target model to localize the target in each frame, by minimizing a discriminative objective function. The target model, often set to a convolutional kernel, provides a compact and generalizable representation of the tracked object, leading to the popularity of DCFs.

The objective function in DCF integrates both foreground and background knowledge over the previous frames, providing effective

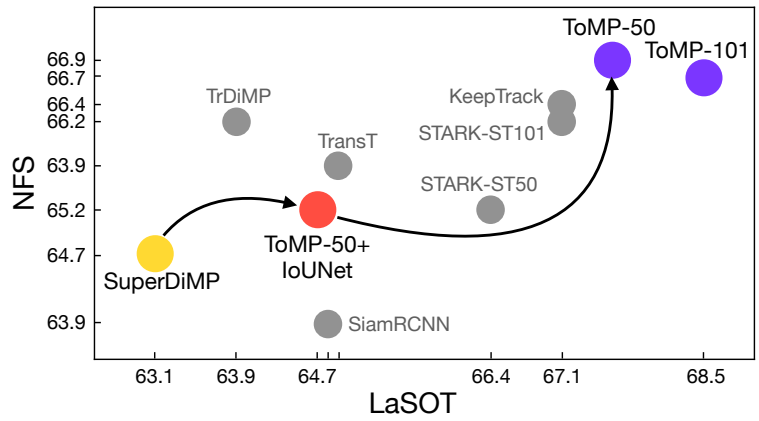


FIGURE 4.1: Performance improvements when transforming the model optimizer based tracker SuperDiMP [24] (●) step-by-step. First, we replace the model optimizer by a Transformer based model predictor (●). Secondly, we replace the probabilistic IoUNet by a new regressor and predict its weights with the same model predictor (●). The performance (success AUC) is reported on NFS [42] and LaSOT [38] and compared with recent trackers (●). ToMP-50 and ToMP-101 refer to the different employed backbones ResNet-50 [50] and ResNet-101 [50].

global reasoning when learning the model. However, it also imposes severe inductive bias on the predicted target model. Since the target model is obtained by solely minimizing an objective over the previous frames, the model predictor has limited flexibility. For instance, it cannot integrate any learned priors in the predicted target model. On the other hand, Transformers have also been shown to provide strong global reasoning across multiple frames, thanks to the use of self and cross attention. Consequently, Transformers have been applied to generic object tracking [14, 106, 117, 121] with considerable success.

In this work, we propose a novel tracking framework that aims at bridging the gap between DCF and Transformer based trackers. Our approach employs a compact target model for localizing the target, as in DCF. The weights of this model are however obtained using a Transformer-based model predictor, allowing us to learn more powerful target models, compared to DCFs. This is achieved by

introducing novel encodings of the target state, allowing the Transformer to effectively utilize this information. We further extend our model predictor to generate weights for a bounding box regressor network, in order to condition its predictions on the current target. Our proposed approach ToMP obtains significant improvement in tracking performance compared to state-of-the-art DCF-based methods, while also outperforming recent Transformer based trackers (see Fig. 4.1).

Contributions: In summary, our main contributions are the following:

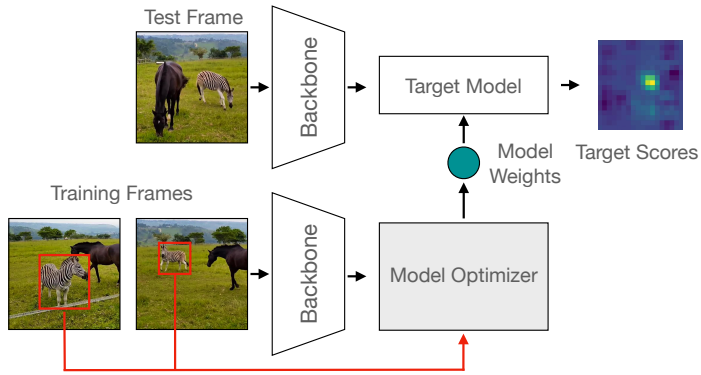
i) We propose a novel Transformer-based model prediction module in order to replace traditional optimization based model predictors.

ii) We extend the model predictor to estimate a second set of weights that are applied for bounding box regression.

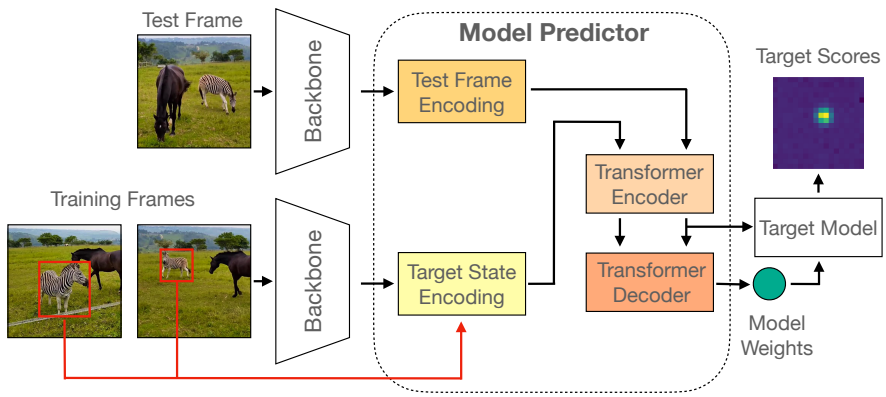
iii) We develop two novel encodings that incorporate target location and target extent allowing the Transformer-based model predictor to utilize this information.

iv) We propose a parallel two stage tracking procedure at test time to decouple target localization and bounding box regression in order to achieve robust and accurate target detection.

v) We perform a comprehensive set of ablation experiments to assess the contribution of each building block of our tracking pipeline and evaluate it on seven tracking benchmarks. The proposed tracker ToMP sets a new state of the art on three including LaSOT [38] where it achieves an AUC of 68.5% (see Fig. 4.1). In addition we show that our tracker ToMP outperforms other Transformer based trackers for every attribute of LaSOT [38].



(a) Tracker with optimization based model prediction.



(b) Proposed tracker with Transformer based model prediction.

FIGURE 4.2: Comparison between trackers that employ optimization based model prediction and our Transformer-based model prediction. The model optimizer [ ] in Fig. 4.2a is replaced by the model predictor in Fig. 4.2b that consists of the proposed modules [, , ,].

4.2 METHOD

In this work, we propose a Transformer-based target model prediction network for tracking called ToMP. We first revisit existing optimization based model predictors and discuss their limitations in Sec. 4.2.1. Next, we describe our Transformer-based model prediction approach in Sec. 4.2.2. We extend this approach to perform joint target classification and bounding box regression in Sec. 4.2.3. Finally, we detail our offline training procedure and online tracking pipeline in Sec. 4.2.4 and Sec. 4.2.5, respectively.

4.2.1 *Background*

One of the popular paradigms for visual object tracking is discriminative model prediction based tracking. These approaches, visualized in Fig. 4.2a, use a target model to localize the target object in the test frame. The weights (parameters) of this target model are obtained from the model optimizer, using the training frames and their annotation. While a variety of target models are used in the literature [6, 23, 57, 78, 97, 105, 130], discriminative trackers share a common base formulation to produce the target model weights. This involves solving an optimization problem such that the target model produces the desired target states $y_i \in \mathcal{Y}$ for the training samples $\mathcal{S}_{\text{train}} \in \{(x_i, y_i)\}_{i=1}^{m}$. Here, $x_i \in \mathcal{X}$ refers to a deep feature map of frame i and m denotes the total number of training frames. The optimization problem reads as follows,

$$w = \arg\min_{\tilde{w}} \sum_{(x,y)\in\mathcal{S}_{\text{train}}} f(h(\tilde{w}; x), y) + \lambda g(\tilde{w}). \tag{4.1}$$

Here, the objective consists of the residual function f which computes an error between the target model output $h(\tilde{w}; x)$ and the ground truth label y. $g(\tilde{w})$ denotes the regularization term weighted by a scalar λ, while w represents the optimal weights of the target model. Note that the training set $\mathcal{S}_{\text{train}}$ contains the annotated first frame, as

well as the previous tracked frames with the tracker's predictions being used as pseudo-labels.

Learning the target model by explicitly minimizing the objective of (4.1) provides a robust target model that can distinguish the target from the previously seen background. However, such a strategy suffers from notable limitations. The optimization based methods compute the target model using only limited information available in previously tracked frames. That is, they cannot integrate learned priors in the target model prediction so as to minimize *future* failures. Similarly, these methods typically lack the possibility to utilize the current test frame in a transductive manner when computing the model weights to improve tracking performance. The optimization based methods also require setting multiple optimizer hyperparameters, and can overfit/underfit on the training samples. Another limitation of optimization based trackers is their procedure that produces the discriminative features. Usually, the features provided to the target model are simply the extracted test features. Instead of reinforced features by using the target state information contained in the training frames. Extracting such enhanced features would allow reliable differentiation between the target and background regions in the test frame.

4.2.2 *Transformer-based Target Model Prediction*

In order to overcome the aforementioned limitations of optimization based target localization approaches, we propose to replace the model optimizer by a novel target model predictor based on Transformers (see Fig. 4.2b). Instead of explicitly minimizing an objective as stated in (4.1), our approach learns to directly predict the target model purely from data by end-to-end training. This allows the model predictor to integrate target specific priors in the predicted model so that it can focus on characteristic features of the target, in addition to the features that allow to differentiate the target from the *seen* background. Furthermore, our model predictor also utilizes

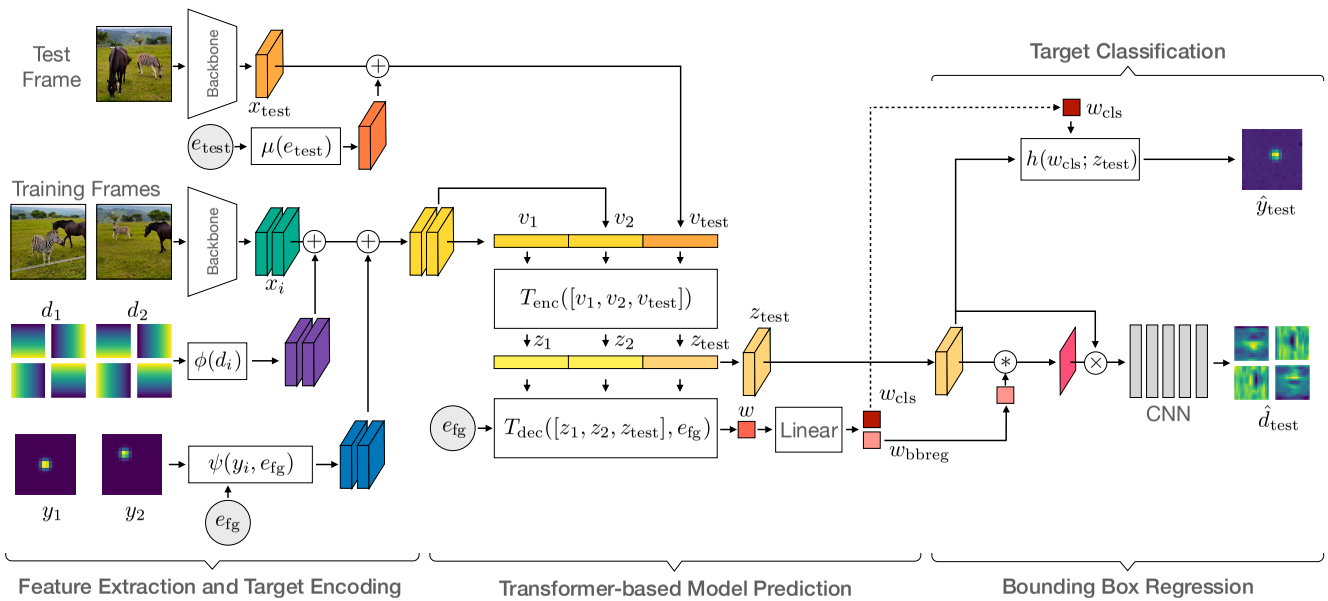


FIGURE 4.3: Overview of the entire ToMP tracking pipeline for joint model prediction. First, the training [■] and test [■] features are extracted using a backbone. Then the target location [■] and bounding box [■] encodings are added to the training features. For the test frame the test embedding is encoded [■] and added to the test features. The features are then concatenated and jointly processed by the Transformer-based model predictor that produces the weights used for target classification [■] and bounding box regression [■].

the current test frame features, in addition to the previous training features, to predict the target model in a transductive manner. As a result, the model predictor can utilize the current frame information to predict a more suitable target model. Finally, instead of applying the target model on a fixed feature space, defined by the pre-trained feature extractor, our approach can utilize the target information to dynamically construct a more discriminative feature space for every frame.

An overview of the proposed tracker employing the Transformer-based model prediction is shown in Fig. 4.2b. Similar to the optimization based trackers, it consists of a test and training branch. We first encode the target state information in the training frames and fuse it with the deep image features [■]. Similarly, we also add an encoding to the test frame in order to mark it as test frame [■]. The features from both the training and test branches are then jointly processed in the Transformer Encoder [■] that produces enhanced features by reasoning globally across frames. Next, the Transformer Decoder [■] predicts the target model weights [●] using the output of the Transformer Encoder. Finally, the predicted target model is applied on the enhanced test frame features to localize the target. Next, we describe the main components in our tracking pipeline.

Target Location Encoding: We propose a target location encoding that allows the model predictor to incorporate the target state information from the training frames, when predicting the target model. In particular, we use the embedding $e_{\text{fg}} \in \mathbb{R}^{1\times C}$ that represents *foreground*. Together with a Gaussian $y_i \in \mathbb{R}^{H\times W\times 1}$ centered at the target location, we define the target encoding function

$$\psi(y_i, e_{\text{fg}}) = y_i \cdot e_{\text{fg}}, \tag{4.2}$$

where "$\cdot$" denotes point-wise multiplication with broadcasting. Note, that $H_{\text{im}} = s \cdot H$ and $W_{\text{im}} = s \cdot W$ correspond to the spatial dimension of the image patch and s to the stride of the backbone network used

to extract the deep features $x \in \mathbb{R}^{H\times W\times C}$. Next, we combine the target encoding and the deep image features x as follows

$$v_i = x_i + \psi(y_i, e_{\text{fg}}). \tag{4.3}$$

This provides us the training frame features $v_i \in \mathbb{R}^{H\times W\times C}$ which contain encoded target state information. Similarly, we also add a test encoding to identify the features corresponding to the test frame as,

$$v_{\text{test}} = x_{\text{test}} + \mu(e_{\text{test}}), \tag{4.4}$$

where $\mu(\cdot)$ repeats the token e_{test} for each patch of x_{test}.

Transformer Encoder: We aim to predict our target model using the foreground and background information from both the training, as well as the test frames. To achieve this, we use a Transformer Encoder [12, 103] module to first jointly process the features from the training frames and the test frame. The Transformer Encoder serves two purposes in our approach. First, as described later, it computes the features used by the Transformer Decoder module to predict the target model. Secondly, inspired by STARK [117], our Transformer Encoder also outputs enhanced test frame features, which serve as the input to the target model when localizing the target.

Given multiple encoded training features $v_i \in \mathbb{R}^{H\times W\times C}$ and an encoded test feature $v_{\text{test}} \in \mathbb{R}^{H\times W\times C}$, we reshape the features to $\mathbb{R}^{(H\cdot W)\times C}$ and concatenate all m training features v_i and the test feature v_{test} along the first dimension. These concatenated features are then processed jointly in a Transformer Encoder

$$[z_1, \ldots, z_m, z_{\text{test}}] = T_{\text{enc}}([v_1, \ldots, v_m, v_{\text{test}}]). \tag{4.5}$$

The Transformer Encoder consists of multi-headed self-attention modules [103] that enable it to reason globally across a full frame and even across multiple training and test frames. In addition, the encoded target state identifies *foreground* and *background* regions and enables the Transformer to differentiate between both regions.

Transformer Decoder: The outputs of the Transformer Encoder (z_i and z_{test}) are used as inputs for the Transformer Decoder [12, 103] to predict the target model weights

$$w = T_{\text{dec}}([z_1, \dots, z_m, z_{\text{test}}], e_{\text{fg}}). \tag{4.6}$$

Note that the inputs z_i and z_{test} are obtained by jointly reasoning over the whole training and test samples, allowing us to predict a discriminative target model. We use the same learned *foreground* embedding e_{fg} as used for target state encoding as input query of the Transformer Decoder such that the Decoder predicts the target model weights.

Target Model: We use the DCF target model to obtain the target classification scores

$$h(w, z_{\text{test}}) = w * z_{\text{test}}. \tag{4.7}$$

Here, the weights of the convolution filter $w \in \mathbb{R}^{1\times C}$ are predicted by the Transformer Decoder. Note that the target model is applied on the output test features z_{test} of the Transformer Encoder. These features are obtained after joint processing of training and test frames, and thus support the target model to reliably localize the target.

4.2.3 *Joint Localization and Box Regression*

In the previous section, we presented our Transformer based architecture for predicting the target model. Although the target model can localize the object center in each frame, a tracker needs to also estimate an accurate bounding box of the target. DCF based trackers typically employ a dedicated bounding box regression network [23] for this task. While it is possible to follow a similar strategy, we decide to predict both models jointly since target localization and bounding box regression are related tasks that can benefit from one another. In order to achieve this, we extend our model as follows. First, instead of only using the target center location when generating the target state encoding, we also encode target size information to

provide a richer input to our model predictor. Secondly, we extend our model predictor to estimate weights for a bounding box regression network, in addition to the target model weights. The resulting tracking architecture is visualized in Fig. 4.3. Next, we describe each of these changes in detail.

Target Extent Encoding: In addition to the extracted deep image features x_i and the target location encoding $\psi(y_i, e_{\text{fg}})$, we add another encoding to incorporate information about the bounding box of the target. In order to encode the bounding box $b_i = \{b_i^x, b_i^y, b_i^w, b_i^h\}$ encompassing the target object in the training frame i, we adopt the *ltrb* representation [41, 99, 115, 121]. First, we map each location (j^x, j^y) on the feature map x_i back to the image domain using $(k^x, k^y) = (\lfloor \frac{s}{2} \rfloor + s \cdot j^x, \lfloor \frac{s}{2} \rfloor + s \cdot j^y)$. Then, we compute the normalized distance of each remapped location to the four sides of the bounding box b_i as follows,

$$\begin{aligned} l_i &= (k^x - b_i^x)/W_{\text{im}}, \quad r_i = (k^x - b_i^x - b_i^w)/W_{\text{im}}, \\ t_i &= (k^y - b_i^y)/H_{\text{im}}, \quad b_i = (k^y - b_i^y - b_i^h)/H_{\text{im}}, \end{aligned} \tag{4.8}$$

where $W_{\text{im}} = s \cdot W$ and $H_{\text{im}} = s \cdot H$. These four sides are used to produce the dense bounding box representation $d = (l, t, r, b)$, where $d \in \mathbb{R}^{H \times W \times 4}$. In this representation, we encode the bounding box using a MLP ϕ and thereby increase the number of dimensions from 4 to C before adding the obtained encoding to Eq. (4.3) such that

$$v_i = x_i + \psi(y_i, e_{\text{fg}}) + \phi(d_i). \tag{4.9}$$

Here, v_i is the resulting feature map which is used as input to the Transformer Encoder, see Fig. 4.3.

Model Prediction: We extend our architecture to predict weights for the target model, as well as bounding box regression. Concretely, we pass the output w of the Transformer Decoder through a linear layer to obtain the weights for bounding box regression w_{bbreg} and target classification w_{cls}. The weights w_{cls} are then directly used within the target model $h(w_{\text{cls}}; z_{\text{test}})$ as before. The weights w_{bbreg}, on the other hand, are used to condition the output test features z_{test} of

the Transformer Encoder with target information for bounding box regression, as explained next.

Bounding Box Regression: To make the encoder output features z_{test} target aware, we follow Yan *et al*. [117] and first compute an attention map $w_{\text{bbreg}} * z_{\text{test}}$ using the predicted weights w_{bbreg}. The attention weights are then multiplied point-wise with the test features z_{test} before feeding them into a Convolutional Neural Network (CNN). The last layer of the CNN uses an exponential activation function to produce the normalized bounding box prediction in the same *ltrb* representation as described in Eq. (4.8). In order to obtain the final bounding box estimation, we first extract the center location by applying the argmax$(\cdot)$ function on the target score map $\hat{y}_{\text{test}}$ predicted by the target model. Next, we query the dense bounding box prediction $\hat{d}_{\text{test}}$ at the center location of the target object to obtain the bounding box. We use two dedicated networks for target localization and bounding box regression in contrast to Yan *et al*. [117] that uses one network trying to predict both. This allows us as explained in Sec. 4.2.5 to decouple target localization from bounding box regression during tracking.

4.2.4 *Offline Training*

In this section, we describe the protocol to train the proposed tracker ToMP. Similar to recent end-to-end trained discriminative trackers [6, 30], we sample multiple training and test frames from a video sequence to form training sub-sequences. In particular, we use two training frames and one test frame. In contrast to recent Transformer based trackers [14, 117, 121] but similar to DCF based trackers [6, 23, 30], we keep the same spatial resolution for training and test frames. We pair each image I_i with the corresponding bounding box b_i. We use the target state of the training frames to encode target information and use the bounding box of the test frame only to supervise training by computing two losses based on the predicted bounding boxes and the derived center location of the target in the test frame.

We employ the target classification loss from DiMP [6] that consists of different losses for background and foreground regions. Further, we employ the generalized Intersection over Union loss [91] using the *ltrb* bounding box representation [99] to supervise bounding box regression

$$L_{\text{tot}} = \lambda_{\text{cls}} L_{\text{cls}}(\hat{y}, y) + \lambda_{\text{giou}} L_{\text{giou}}(\hat{d}, d), \tag{4.10}$$

where λ_{cls} and λ_{giou} are scalars weighting the contribution of each loss. Note that in contrast to FCOS [99] and related trackers [41] we omit an additional centerness loss since it would be redundant in addition to our classification loss that serves the same purpose. A detailed study examining the impact of centerness is available in the appendix.

Training Details: We train our tracker on the training splits of the LaSOT [38], GOT10k [54], Trackingnet [85] and MS-COCO [73] datasets. We sample 40k sub-sequences and train for 300 epochs on two Nvidia Titan RTX GPUs. We use ADAMW [76] with a learning rate of 0.0001 that we decay by a factor of 0.2 after 150 and 250 epochs and weight decay of 0.0001. We set $\lambda_{\text{cls}} = 100$ and $\lambda_{\text{giou}} = 1$. We construct a training sub-sequence by randomly sampling two training frames and a test frame from a 200 frame window within a video sequence. We then extract the image patches after randomly translating and scaling the image relative to the target bounding box. Moreover, we use random image flipping and color jittering for data augmentation. We set the spatial resolution of the target scores to 18×18 and set the search area scale factor to 5.0. Further training and architecture details are provided in the appendix.

4.2.5 *Online Tracking*

During tracking, we use the annotated first frame, as well as previously tracked frames as our training set $\mathcal{S}_{\text{train}}$. While we always keep the initial frame and its annotation, we include one previously tracked frame and replace it with the most recent frame that achieves

a target classifier confidence higher than a threshold. Hence, the training set $\mathcal{S}_{\text{train}}$ contains at most two frames.

We observed that incorporating previous tracking results in $\mathcal{S}_{\text{train}}$ improves the target localization considerably. However, including predicted bounding box estimations degrades the bounding box regression performance due to inaccurate predictions, see Sec. 4.3.1. Hence, we run the model predictor twice. First, we include intermediate predictions in $\mathcal{S}_{\text{train}}$ to obtain the classifier weights. In the second pass, we only use the annotated initial frame to predict the bounding box. Note that for efficiency both steps can be performed in parallel in a single forward pass. In particular, we reshape the feature map corresponding to two training and one test frame to a sequence and duplicate it. Then, we stack both in the batch dimension to process them jointly with the model predictor. To only allow attention between the initial frame with ground truth annotation and the test frame when predicting the model for bounding box regression, we make use of the so-called *key_padding_mask* that allows us to ignore certain keys when computing attention.

4.3 EXPERIMENTS

We evaluate our proposed tracking architecture ToMP on seven benchmarks. Our approach is based on PyTorch 1.7 and is developed within the PyTracking [24] framework. PyTracking is available under the GNU GPL 3.0 license. On a single Nvidia RTX 2080Ti GPU, ToMP-101 and ToMP-50 achieve 19.6 and 24.8 FPS and use a ResNet-101 [50] and ResNet-50 [50] as backbone respectively.

4.3.1 *Ablation Study*

We perform a comprehensive analysis of the proposed tracker. First, we analyze the contribution of the different proposed target state encodings and then examine the effect of different inference settings. Finally, we report the performance achieved when replacing the target

	e_{fg}	e_{bg}	e_{test}	$\phi(\cdot)$	$q_{\text{dec}} = e_{\text{fg}}$	LaSOT	NFS	OTB
①	✗	✗	✗	✓	n.a.	66.0	64.8	68.2
②	✓	✗	✗	✓	✓	67.1	**66.6**	**70.0**
③	✓	✓	✗	✓	✓	67.1	66.3	69.4
④	✓	✗	✓	✓	✓	**67.6**	**66.9**	**70.1**
⑤	✓	✓	✓	✓	✓	**67.4**	66.0	69.5
⑥	✓	✗	✓	✓	✗	66.0	66.2	69.9
⑦	✓	✗	✓	✗	✓	63.1	64.2	64.0

TABLE 4.1: For e_{fg}, e_{bg} and e_{test} learning the embedding is denoted by ✓ whereas ✗ means setting it to zero. Using the encoding $\phi(\cdot)$ is denoted by ✓ whereas ✗ refers to omitting it. For $q_{\text{dec}} = e_{\text{fg}}$ the symbol ✓ means sharing the learned embedding e_{fg} for encoding and querying the Decoder wheres ✗ means learning two separate embeddings for both tasks. (Our final model is in the 4^{th} row)..

classifier or the bounding box regressor of SuperDiMP with ours. All ablation experiments in this part use a ResNet-50 as backbone.

Target State Encoding: In order to analyze the effect of the different target state encodings we train different variants of our network and evaluate them on multiple datasets. The first five rows of Tab. 4.1 correspond to versions with different target location encodings. All other settings are kept the same. In addition to the foreground and test embedding, we include a learned background embedding (instead of setting $e_{\text{bg}} = 0$) to our analysis as follows: $\psi(y_i, e_{\text{fg}}, e_{\text{bg}}) = y_i \cdot e_{\text{fg}} + (1 - y_i) \cdot e_{\text{bg}}$. However, Tab. 4.1 shows (4^{th} vs. 5^{th} row) that adding such a learned background embedding decreases the tracking performance. We further observe that setting the foreground embedding $e_{\text{fg}} = 0$ (1^{st} row) and only relying on the target extent encoding $\phi(\cdot)$ still achieves high tracking performance but clearly lacks behind all other versions that include the foreground embedding. We conclude that using only the foreground encoding e_{fg} and the test encoding e_{test} leads to the best performance (4^{th} row).

Number of Decoder queries	Linear Layer	Decoder query q_{dec}	LaSOT	NFS	OTB
1	✓	$q_{\text{dec}} = e_{\text{fg}}$	**67.6**	**66.9**	**70.1**
2	✗	$q_{\text{dec}} \neq e_{\text{fg}}$	63.7	62.8	67.9

TABLE 4.2: Analysis of different model predictor architectures and its impact on the tracking performance in terms of success AUC.

Two Stage Model Prediction	Previous Tracking Results	Confidence Threshold η	LaSOT	NFS	OTB
✓	✓	0.85	67.3	66.9	**70.3**
✓	✓	0.90	**67.6**	**66.9**	70.1
✓	✓	0.95	67.4	66.0	69.8

TABLE 4.3: Analysis of different inference settings an of their impact on the tracking performance in terms of AUC of the success curve.

In the second part of Tab. 4.1 we choose the best settings for the target location encoding and remove either the target extent encoding $\phi(\cdot)$ or decouple the Transformer Decoder query from the foreground embedding e_{fg}. We observe that using a separate query (6^{th} row) decreases the overall performance. Similarly, we notice that incorporating target extent information via the proposed encoding is crucial. Otherwise, the performance drops significantly (7^{th} row).

Model Predictor: Since our model predictor estimates two different model weights, it seems natural to use two different Transformer queries: one to produce the target model weights and the other to obtain the bounding box regressor weights. However, this involves decoupling the query from the foreground embedding e_{fg} and the experiments in Tab. 4.2 show a significant performance drop for this case.

Inference Settings: During online tracking, we use the initial frame and its annotation as training frames. In addition, we include the

Two Stage Model Prediction	Previous Tracking Results	LaSOT	NFS	OTB
n.a.	✗	65.7	65.3	67.8
✓	✓	**67.6**	**66.9**	**70.1**
✗	✓	62.0	64.8	62.8

TABLE 4.4: Analysis of different inference settings an of their impact on the tracking performance in terms of success AUC.

Model Predictor	Bounding Box Regressor	LaSOT	NFS	UAV	LaSOT ExtSub
DiMP [6]	Prob. IoUNet [30]	63.1	64.8	**67.7**	43.7
ToMP	Prob. IoUNet [30]	**64.7**	**65.2**	65.0	**45.2**
ToMP	**ToMP**	**67.6**	**66.9**	**69.0**	**45.4**

TABLE 4.5: Impact of replacing DiMP [6] and the probabilistic IoUNet [30] with ToMP for localization and box regression.

most recent frame and its target prediction if the classifier confidence is above the threshold $\eta = 0.9$, see Tab. 4.3. Tab. 4.4 shows that including previous tracking results leads to higher tracking performance than using only the initial frame. Disabling the described two stage model prediction approach and predicting the weights of the target model and bounding box regressor at once decreases the tracking performance drastically (-5.6 AUC on LaSOT). The reason is the sensitivity of the bounding box predictor to inaccurate predicted boxes that are encoded and used for training.

Transforming Model Prediction Step-by-Step: Our model predictor can estimate model weights for the target model and bounding box regressor. In this part, we will transform an optimization based tracker step-by-step to assess the impact of each transformation step. Tab. 4.5 shows that replacing the model optimizer in SuperDiMP (1^{st} row) with our proposed model predictor to only predict the

	ToMP 101	**ToMP 50**	STARK ST101 [117]	Keep Track [83]	STARK ST50 [117]	Alpha Refine [118]	TransT [14]	Siam R-CNN [104]	Tr DiMP [106]	Super DiMP [24]	STM SAOT [132]	Track [41]	DTT [121]	Pr DiMP [30]
Precision	73.5	72.2	72.2	70.2	71.2	68.0	69.0	68.4	66.3	65.3	-	63.3	-	60.8
Norm. Prec	79.2	78.0	76.9	77.2	76.3	73.2	73.8	72.2	73.0	72.2	70.8	69.3	-	68.8
Success (AUC)	68.5	67.6	67.1	67.1	66.4	65.3	64.9	64.8	63.9	63.1	61.6	60.6	60.1	59.8

TABLE 4.6: Comparison on the LaSOT [38] test set ordered by AUC.

target model (2nd row) outperforms SuperDiMP on three out of four datasets. Our tracker ToMP that jointly predicts model weights for target localization and bounding box regression (3rd row) achieves the best performance on all four datasets. We conclude that predicting the weights of the target model improves the performance and likewise predicting the weights of the bounding box regressor. Note that we report the average over five runs for all trackers based on the probabilistic IoUNet due to its stochasticity.

4.3.2 *Comparison to the State of the Art*

We compare our tracker ToMP on seven tracking benchmarks. The same settings and parameters are used for all datasets. We recompute the metrics of all trackers using the raw predictions if available or otherwise report the results given in the respective paper.

LaSOT [38]: First, we compare ToMP on the large-scale LaSOT dataset (280 test sequences with 2500 frames on average). The success plot in Fig. 4.5a shows the overlap precision OP_T as a function of the threshold T. Trackers are ranked w. r. t. their AUC score, shown in the legend. Tab. 4.6 shows more results including precision and normalized precision for each tracker. Both versions of ToMP with different backbones outperform the recent trackers STARK [117], TransT [14], TrDiMP [106] and DTT [121] in AUC and sets a new state-of-the-art result. Note that even ToMP with ResNet-50 outperforms STARK-ST101 with ResNet-101 (67.6 vs 67.1). Fig. 4.4 shows the success AUC gain of ToMP compared to recent Transformer based

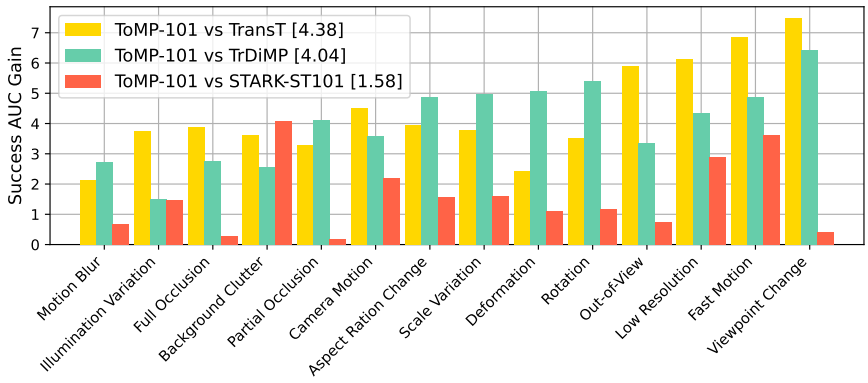
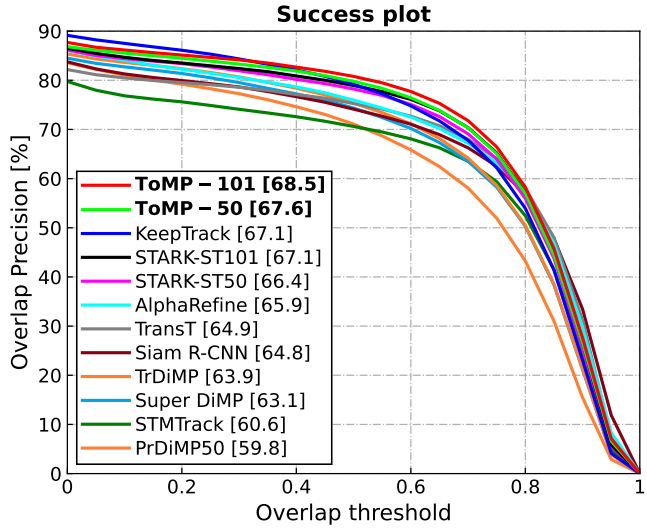


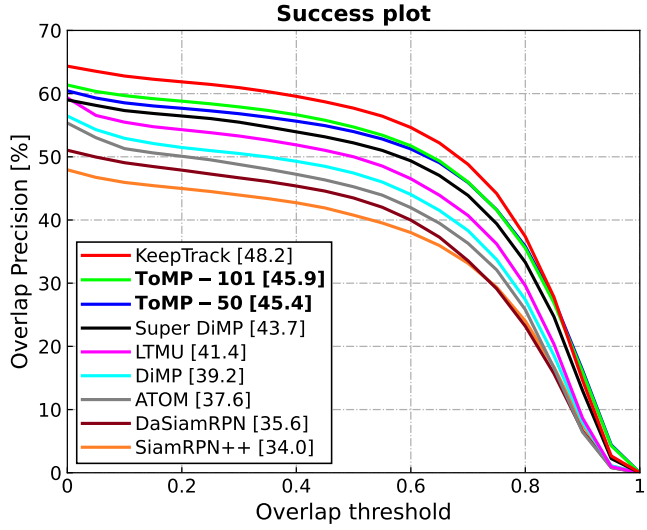
FIGURE 4.4: Per attribute analysis on LaSOT [38] between ToMP and recent Transformer based trackers. The bar heights correspond to the gain of our tracker and the legend shows the average gain.

trackers for different attributes annotated in LaSOT [38]. We want to highlight that ToMP outperforms TransT [14] and TrDiMP [106] on each attribute by more than one percent point. Similarly, ToMP achieves higher performance than STARK-ST101 for every attribute. It achieves the highest gain over STARK for *Background Clutter*, showing the disadvantage of using small templates instead of training frames with a large field of view that allow not only to leverage target, but also background information.

LaSOTExtSub [37]: This dataset is an extension of LaSOT. It only contains test sequences assigned to 15 new classes with 10 videos each. The sequences contain 2500 frames on average showing challenging tracking scenarios of small, fast moving objects with distractors present. Fig. 4.5b shows the success plot where the results of most trackers are obtained from [37], e. g., DaSiamRPN [133], SiamRPN++ [67], ATOM [23], DiMP [6] and LTMU [22]. ToMP exceeds the performance of all trackers except KeepTrack [83] that employs explicit distractor matching between frames. In particular, we outperform SuperDiMP [24] that uses a model optimizer (+2.2%).



(a) LaSOT [38]



(b) LaSOTExtSub [37]

FIGURE 4.5: Success plots, showing OP_T, on LaSOT [38] and LaSOTExtSub [37] and AUC is reported in the legend.

	ToMP 101	**ToMP 50**	STARK ST101 [117]	TransT [14]	STARK ST50 [117]	Siam R-CNN [104]	Alpha Refine [118]	STM Track [41]	DTT [121]	Tr DiMP [106]	Keep Track [83]	Super DiMP [24]	Pr DiMP [30]	Siam FC++ [115]
Precision	78.9	78.6	-	**80.3**	-	80.0	78.3	76.7	78.9	73.1	73.8	73.3	70.4	70.5
Norm. Prec	86.4	86.2	**86.9**	**86.7**	86.1	85.4	85.6	85.1	85.0	83.3	83.5	83.5	81.6	80.0
Success (AUC)	**81.5**	81.2	**82.0**	81.4	81.3	81.2	80.5	80.3	79.6	78.4	78.1	78.1	75.8	75.4

TABLE 4.7: Comparison on the TrackingNet [85] test set.

	ToMP 101	**ToMP 50**	Keep Track [83]	CRACT [39]	STARK ST101 [117]	TrDiMP [106]	TransT [14]	STARK ST50 [117]	Super DiMP [24]	Pr DiMP [30]	STM Track [41]	Siam AttN [124]	Siam R-CNN [104]	KYS [7]	DiMP [6]
UAV123	66.9	69.0	**69.7**	66.4	68.2	67.5	**69.1**	**69.1**	67.7	68.0	64.7	65.0	64.9	–	65.3
OTB-100	70.1	70.1	70.9	**72.6**	68.1	71.1	69.4	68.5	70.1	69.6	**71.9**	71.2	70.1	69.5	68.4
NFS	**66.7**	**66.9**	66.4	62.5	66.2	66.2	65.7	65.2	64.8	63.5	–	–	63.9	63.5	62.0

TABLE 4.8: Comparison with the state of the art on the OTB-100 [109], NFS [42] and UAV123 [84] datasets in terms of AUC score.

TrackingNet [85]: We evaluate ToMP on the large-scale TrackingNet dataset that contains 511 test sequences without publicly available ground-truth. An online evaluation server is used to obtain the tracking metrics shown in Tab. 4.7 by submitting the raw tracking results. Both versions of ToMP achieve competitive results close to the current state of the art. In particular, ToMP-101 achieves the second best performance in terms of AUC behind STARK [117], outperforming other Transformer based trackers such as TransT [14] and TrDiMP [106].

UAV123 [84]: The UAV dataset consists of 123 test videos that contain small objects, target occlusion, and distractors. Tab. 4.8 shows the achieved results in terms of success AUC. Again, ToMP achieves competitive results compared to the current state of the art achieved by KeepTrack [83].

OTB-100 [109]: We also report results on the OTB-100 dataset that contains 100 short sequences. Multiple trackers achieve results above 70% AUC. Among them are both versions of ToMP, see Tab. 4.8.

	ToMP 101	**ToMP 50**	STARK ST50 [117]	Super DiMP [24, 61]	STARK ST101 [117]	DPMT [61]	TRAT [61]	UPDT [8, 61]	DiMP [6, 61]	ATOM [23, 61]
Accuracy	0.453	0.453	0.478.	0.477	**0.481**	**0.492**	0.464	0.465	0.457	0.462
Robustness	**0.814**	0.789	**0.799**	0.728	0.775	0.745	0.744	0.755	0.734	0.734
EAO	**0.309**	0.297	**0.308**	0.305	0.303	0.303	0.280	0.278	0.274	0.271

TABLE 4.9: Comparison to the state of the art of bounding box only methods on VOT2020ST [61] in terms of EAO score.

ToMP achieve the same performance as SuperDiMP [24] but slightly higher results than TransT [14] and slightly lower than TrDiMP [106].

NFS [42]: We compete on the NFS dataset (30FPS version) containing 100 test videos. It contains fast motions and challenging sequences with distractors. Both versions of ToMP exceed the performance of the current best method KeepTrack [83] by +0.5% and +0.3%, see Tab. 4.8.

VOT2020 [61]: Finally, we evaluate on the 2020 edition of the Visual Object Tracking short-term challenge. We compare with the top methods in the challenge [61], as well as more recent methods. The dataset contains 60 videos annotated with segmentation masks. Since ToMP produces bounding boxes we only compare with trackers that produce the bounding boxes as well. The trackers are evaluated following the multi-start protocol and are ranked according to the EAO metric that is based on tracking accuracy and robustness, defined using IoU overlap and failure rate respectively. The results in Tab. 4.9 show that ToMP-101 achieves the best overall performance, with the highest robustness and competitive accuracy compared to previous methods.

4.3.3 *VOT2020 with AlphaRefine*

In contrast to previous years where the sequences in the VOT short-term challenge were annotated with bounding boxes [60, 63] the

	ToMP 101+AR	**ToMP 50 +AR**	RPT [61, 81]	STARK ST50+AR [117]	STARK ST101+AR [117]	Ocean Plus [15, 61]	Alpha Refine [61, 118]	AFOD [61]	LWTL [9, 61]	Fast Ocean [61]
EAO	0.497	0.496	**0.530**	**0.505**	0.497	0.491	0.482	0.472	0.463	0.461
Accuracy	0.750	0.754	0.700	**0.759**	**0.763**	0.685	0.754	0.713	0.719	0.693
Robustness	0.798	0.793	**0.869**	0.817	0.789	**0.842**	0.777	0.795	0.798	0.803

TABLE 4.10: Comparison to the state of the art of segmentation only methods on VOT2020ST [61] in terms of EAO score.

sequences of the more recent challenges contain segmentation mask annotations [61] of the target in each frame. Above we compared our method with methods that produce bounding boxes. Thus, in addition, we compare our method on the VOT2020 short-term challenge to methods that produce a segmentation mask in each frame. Since our method produces only a bounding box, we use AlphaRefine [118] that is able to produce a segmentation mask given the bounding box. Tab. 4.10 shows that our method achieves competitive results. In particular ToMP-101 achieves the same EAO (for more details on EAO we refer the reader to [61]) as STARK-ST101+AR [117] that employs AlphaRefine too. Nonetheless, RPT [81] achieves higher EAO than our tracker. In particular it scores a higher robustness but a lower accuracy than our trackers.

4.4 CONCLUSION

We propose a novel tracking architecture employing a Transformer-based model predictor. The model predictor estimates the weights of the compact DCF target model to localize the target in the test frame. In addition, the predictor produces a second set of weights used for precise bounding box regression. To achieve this, we develop two new modules that encode target location and its bounding box in the training features. We conduct comprehensive experimental validation and analysis of ToMP on several challenging datasets, and set a new state of the art on three.

4.5 APPENDICES

In the appendix, we first provide details about training, model architecture and inference in Sec. 4.5.A. Further, we report visual results such as a comparison to state-of-the-art trackers, a comparison of different model predictors and failure cases of our tracker in Sec. 4.5.B. Finally, we provide an attribute-wise comparison in Sec. 4.5.C.

4.5.A *Training, Architecture and Inference*

First, we provide additional details about the training followed by a detailed description of the architectures employed and finally we provide further inference details.

4.5.A.1 *Training and Architecture Details*

For training we produce the target states y by using a Gaussian with standard deviation $1/4$ relative to the base target size and by setting $\tau = 0.05$ to differentiate between foreground and background regions in the corresponding classification loss l_{cls} adopted from DiMP [6]. For the model predictor we extract features with a stride of 16 from the third block of the ResNet that we use as backbone. We initialize the backbone with the official weights obtained by training the backbone on ImageNet [33] and freeze the batch norm statistics during training. Since we use a channel dimension of 256 for the Transformer and the ResNet features have 1024 channels we employ an single convolutional layer to decrease the number of channels before feeding the features into the Transformer Encoder. The Transformer Encoder consists of layers containing multi-headed self attention and a feed-forward network. We use eight heads and a hidden dimension of 2048 for the feed-forward network. Furthermore, we use Dropout with probability 0.1 and layer normalization. The Transformer settings are adopted from DETR [12]. The predicted target model weights for classification and bounding box regression consist of a single 1×1 filter with 256 channels. The bounding box

training frames	NFS	OTB	UAV	LaSOT	LaSOTExtSub	Speed [FPS]
1 initial	65.3	67.8	68.7	65.7	43.7	**26.2**
1 initial + 1 recent	66.9	70.1	69.0	**67.6**	**45.4**	**24.8**
2 initial + 1 recent	**67.6**	**70.5**	67.2	**68.0**	**45.4**	20.5
1 initial + 2 recent	66.7	**70.8**	**69.4**	**67.6**	44.4	21.8
1 initial + 3 recent	66.8	**70.5**	**69.2**	**67.6**	44.2	17.6
1 initial + 4 recent	**67.2**	70.1	68.2	67.3	**44.7**	13.2
1 initial + 5 recent	66.8	70.1	69.1	67.2	43.9	11.3

TABLE 4.11: Comparison of different number of training samples in success AUC.

regression CNN consists of four convolution-instance-normalization-ReLU layers and a final convolution layer, followed by an exponential activation. The MLP for target extent encoding ϕ consists of three layers ($4 \rightarrow 64 \rightarrow 256 \rightarrow 256$) where each layer consists of a linear projection, batch normalization and ReLU activation except the last that only consist of a linear projection. The region-encoding tokens e_{fg} and e_{test} are 256 dimensional learnable embeddings.

4.5.A.2 *Inference Details*

In order to decide whether a previous tracking result should be used for training of not we use the maximal value of the target score map produced by the target model as discussed in the main paper. Furthermore, we follow SuperDiMP [24] and enter in the *target not found* state if the maximal value of the target score map is bellow 0.25. Moreover, we use the same spatial resolution of the target scores of 18×18 and the same search area scale factor of 5.0 during inference and training.

Furthermore, we study the effect of using more than two training frames stored in the sample memory. Instead of using only one initial and one recent training frame to predict the network weights we test the impact of increasing the number of recent training frames and

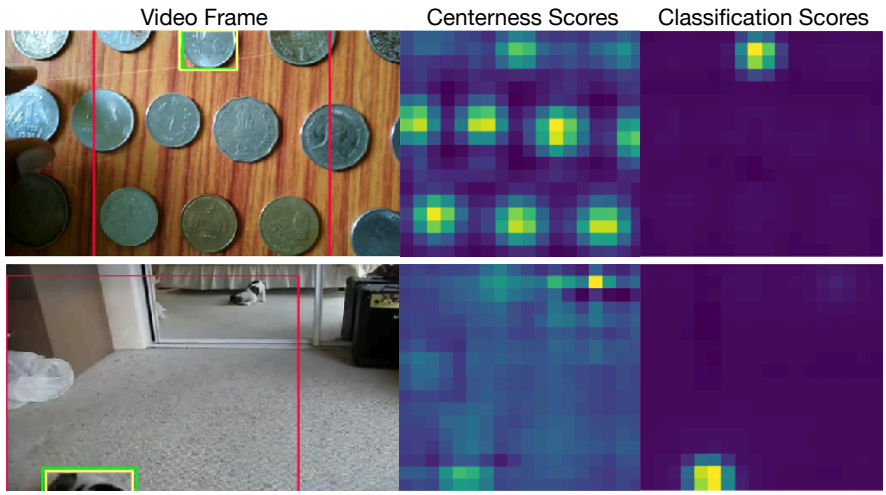


FIGURE 4.6: Visual Comparison between centerness and classification scores.

	$L_{\text{centerness}}$	NFS	OTB	UAV	LaSOT	LaSOTExtSub
Classification	✗	**66.9**	**70.1**	**69.0**	**67.6**	**45.4**
Classification	✓	**65.8**	**69.2**	67.3	**67.9**	**45.5**
Centerness	✓	62.7	66.3	67.4	64.4	41.3
Classification · Centerness	✓	63.7	67.8	**68.7**	65.8	45.3

TABLE 4.12: Impact of centerness scores on training and inference.

of using multiple initial training frames. We increase the number of initial training frames with ground truth bounding box annotations using an augmentation (vertical flipping and random translation). Tab. 4.11 shows the results for different combinations of multiple initial and recent training frames. Note, that we use the same network weights for all experiments trained with one initial and one recent recent frame in all cases. We observer that using more training frames can improve the tracking performance but decreases the run-time. Furthermore, we observe that the tracker greatly benefits from including at least one recent frame for training.

4.5.A.3 *Centerness*

Our proposed bounding box regression component is inspired by FCOS [99] but in contrast to FCOS we omit an auxiliary centerness branch. The classification head of FCOS is trained to predict a high score for almost every region inside the bounding box. The centerness branch is therefore needed to identify the center location of the object, used to select the bounding box offsets. In contrast, our classification branch is directly trained to accurately locate the object's center. The additional centerness branch is therefore redundant. Nonetheless, we train our best model with a centerness head and $L_{\text{centerness}}$ and report the results in Tab. 4.12 (2^{nd}-4^{th} rows). The 1^{st} row shows the performance when omitting centerness for training. We achieve comparable results when using the model trained with centerness but applying only the classification scores to localize the target (2^{nd} row). Using only the centerness scores decreases the performance (3^{rd} row) because centerness often fails to identify the target among distractors (see Fig. 4.6). Finally, we follow FCOS and multiply the classification and centerness scores point-wise to retrieve the target object (4^{th} row). We conclude that omitting the centerness branch for training and during inference to localize the target achieves the best tracking performance.

4.5.B *Visual Results*

In this part we provide visual results of our tracker. First, we show three frames of different sequences where our tracker outperforms the state of the art. Secondly, we compare the produced target score map of our tracker with score maps obtained by optimization based model prediction. Finally, we show some failure cases of our tracker.

4.5.B.1 *Visual Comparison to the State of the Art*

Fig. 4.7 shows three frames of eight different LaSOT [38] sequences where each frame contains the ground truth annotation of the target

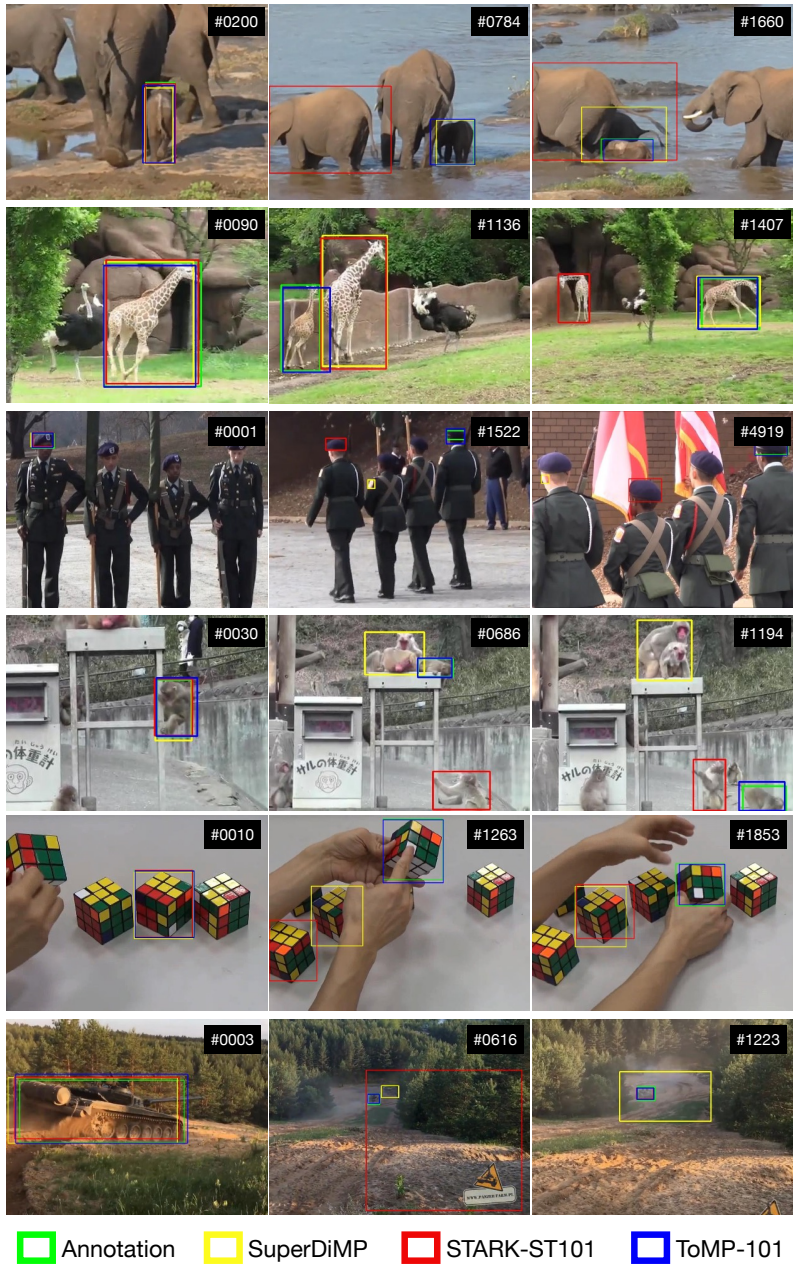


FIGURE 4.7: Visual comparison of different trackers (ToMP-101, SuperDiMP [24] and STARK-ST101 [117]) on different LaSOT [38] sequences.

object and the predictions of three different trackers: SuperDiMP [24], STARK-ST101 [117] and ToMP-101. We observe that our tracker produces in most sequences more robust and in some more accurate bounding box predictions than the related methods. In particular it achieves solid robustness for scenarios where distractors are present but the target object is at least partially visible and not undergoing a full occlusion.

4.5.B.2 *Target Model Prediction*

Fig. 4.8 shows the target score maps produced by the target model when using two different model predictors for three different sequences. In detail we compare the target score map produced by SuperDiMP [24] that adopts the DiMP [6] model predictor with optimized settings. In particular it uses a slightly smaller search area factor of 6 instead of 5 and a target score resolution of 22 instead of 18. Note, that our tracker uses 5 and 18 similar to DiMP [6] as stated Sec. 4.5.A.2. We observe that our model predictor leads to much cleaner and unambiguous target localization than DiMP. While the former often produces multiple local maxima for distractors, our methods is able to almost fully suppress these. An important design choice that enables this is the transductive model weight and test feature prediction produced by our Transformer based model predictor. However, the cleaner score maps come with the risk, that once the target is lost and a distractor is tracked instead recovering is less likely since our tracker effectively suppresses distractors. Similarly, our method learns to produce a score map containing a Gaussian such that overall the maximum score values are higher than by SuperDiMP. Thus, we chose a relatively high threshold to decide whether to use a previous prediction as training sample or not.

4.5.B.3 *Failure Cases*

Fig. 4.9 shows failure cases of our tracker. In particular, it shows three frames of four different LaSOT [38] sequences containing the ground

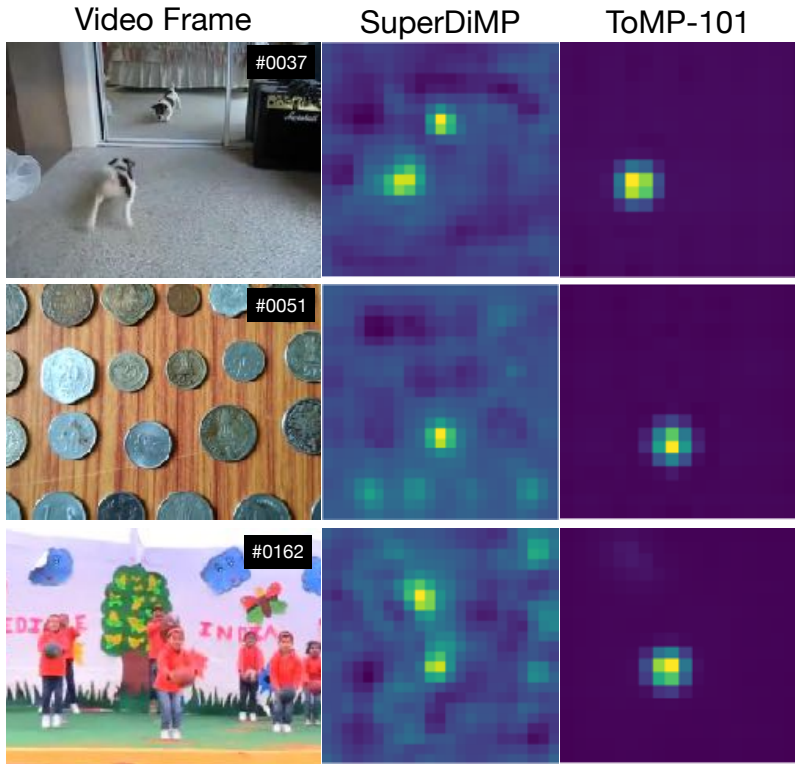


FIGURE 4.8: Visual comparison of the target score maps resulting from different model predictors.

truth annotations and the predicted bounding boxes of our tracker using a ResNet-101 [50] as backbone. To summarize, our tracker typically fails if object similar to the targets so called distractors are present. While the sole presence of distractors typically does not lead to tracking failure, our tracker shows difficulties in sequences where the target is occluded and distractors are present (1st and 3rd row). Instead of detecting that the target is occluded the tracker starts to track a distractor instead. Another challenging scenario are sequences where the target and a distractor approach each other (2nd row in Fig. 4.9) or one occludes the other (4th row in Fig. 4.9). The model then detects only a single object instead of two in both

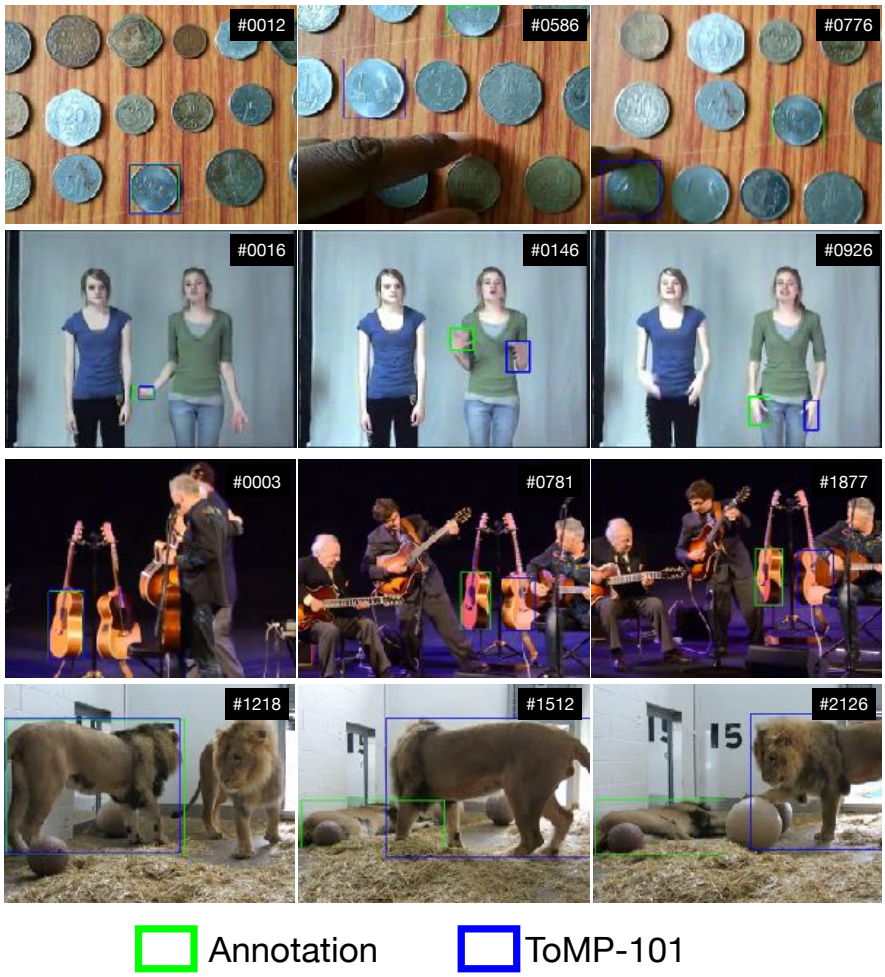


FIGURE 4.9: Visualization of failure cases of our tracker.

scenarios. Once they diverge again and the tracker detects two objects it typically fails to reliably differentiate between the target and the distractor.

4.5.C *Attributes*

To support the attribute based analysis in the main paper, where we compared the performance of our tracker with other Transformer based trackers, we provide the detailed analysis for multiple trackers and ToMP in Tab. 4.13. ToMP-101 achieves the best performance on all but three. It achieves the second best results for *Motion Blur* behind KeepTrack [83] and similar to AlphaRefine [118]. Further ToMP-101 achieves the third best for *Full Occlusion* behind KeepTrack [83] and ToMP-50. Similarly it scores third for *Illumination Variation* behind KeepTrack [83] and AlphaRefine [118]. We further observe, that discriminative model prediction based methods such as TrDiMP [106], SuperDiMP [24], AlphaRefine [118], KeepTrack [83] and ToMP all outperform STARK [117] on the attribute *Background Clutter* showing the advantage of using full training samples during tracking instead of cropped templates that mainly cover the centered target.

	Illumination Variation	Partial Occlusion	Deformation	Motion Blur	Camera Motion	Rotation	Background Clutter	Total
LTMU	56.5	54.0	57.2	55.8	61.6	55.1	49.9	57.2
PrDiMP50	63.7	56.9	60.8	57.9	64.2	58.1	54.3	59.8
STMTrack	65.2	57.1	64.0	55.3	63.3	60.1	54.1	60.6
SuperDiMP	67.8	59.7	63.4	62.0	68.0	61.4	57.3	63.1
TrDiMP	67.5	61.1	64.4	62.4	68.1	62.4	58.9	63.9
Siam R-CNN	64.6	62.2	65.2	63.1	68.2	64.1	54.2	64.8
TransT	65.2	62.0	67.0	63.0	67.2	64.3	57.9	64.9
AlphaRefine	**69.4**	62.3	66.3	**65.2**	70.0	63.9	58.8	65.3
STARK-ST50	66.8	64.3	66.9	62.9	69.0	66.1	57.3	66.4
STARK-ST101	67.5	**65.1**	68.3	64.5	69.5	66.6	57.4	67.1
KeepTrack	**69.7**	64.1	67.0	**66.7**	**71.0**	65.3	**61.2**	67.1
ToMP-50	66.8	64.9	**68.5**	64.6	70.2	**67.3**	59.1	**67.6**
ToMP-101	69.0	**65.3**	**69.4**	**65.2**	**71.7**	**67.8**	**61.5**	**68.5**

	Viewpoint Change	Scale Variation	Full Occlusion	Fast Motion	Out-of-View	Low Resolution	Aspect Ration Change	Total
LTMU	56.7	57.1	49.9	44.0	52.7	51.4	55.1	57.2
PrDiMP50	59.2	59.4	51.3	48.4	55.3	53.5	58.6	59.8
STMTrack	58.2	60.6	47.8	42.4	51.9	50.3	58.8	60.6
SuperDiMP	63.4	62.9	54.1	50.7	59.0	56.4	61.6	63.1
TrDiMP	62.8	63.4	56.4	53.0	60.7	58.1	62.3	63.9
Siam R-CNN	65.3	64.5	55.3	51.5	62.2	57.1	63.4	64.8
TransT	61.7	64.6	55.3	51.0	58.2	56.4	63.2	64.9
AlphaRefine	63.1	65.4	57.4	53.6	61.1	58.6	64.1	65.3
STARK-ST50	67.8	66.1	58.7	53.8	62.1	59.4	64.9	66.4
STARK-ST101	**68.8**	66.8	58.9	54.2	63.3	59.6	65.6	67.1
KeepTrack	66.9	66.8	**60.1**	**57.7**	**64.1**	**62.0**	65.9	67.1
ToMP-50	67.2	**67.5**	**59.3**	56.1	**63.7**	61.1	**66.5**	**67.6**
ToMP-101	**69.2**	**68.4**	59.1	**57.9**	**64.1**	**62.5**	**67.2**	**68.5**

TABLE 4.13: LaSOT [38] attribute-based analysis. Each column corresponds to the results computed on all sequences in the dataset with the corresponding attribute.

5

AVIST: A BENCHMARK FOR VISUAL OBJECT TRACKING IN ADVERSE VISIBILITY

In the previous two chapters, we introduced two new methods that address the problem of visual object tracking in adverse tracking scenarios where the target is occluded, goes out-of-view or where similar objects as the target are present. Beside these scenarios that complicate robust tracking other factors such as adverse visibility can greatly impact the tracking quality. Thus, we introduce in this chapter the new visual tracking benchmark AVisT dedicated for tracking scenarios with adverse visibility. AVisT comprises 120 challenging sequences, spanning 18 diverse scenarios broadly grouped into five attributes with 42 object categories. The key contribution of AVisT are diverse and challenging scenarios covering sever weather conditions such as, dense fog, heavy rain and sandstorm; obstruction effects including, fire, sun glare and splashing water; adverse imaging effects such as, low-light; target effects including, small targets and distractor objects along with camouflage. Our dataset along with the complete tracking performance evaluation is available at https://github.com/visionml/pytracking.

5.1 INTRODUCTION

Visual object tracking is one of the fundamental problems in computer vision, where the objective is to estimate the target state and trajectory in an image sequence, provided only its initial location. The target object is not known a priori and is not constrained to be from a specific object class. Therefore, the main challenge is to accurately learn the appearance of the target object in unconstrained real-world scenarios.

Datasets	OTB-100 [109]	UAV123 [84]	GOT-10k [54]	TrackingNet [85]	LaSOT [38]	AVisT
Best Tracker	TrDiMP	MixF-22k	MixF-1k	MixFL-22k	MixFL-22k	MixFL-22k
Performance	71.1	70.4	71.2	83.9	70.1	56.0

TABLE 5.1: Tracking performance (AUC score) achieved by the top-performing trackers on existing datasets and AVisT. MixF-1k refers to MixFormer-1k, MixF-22k refers to MixFormer-22k and MixFL-22k to MixFormerLarge-22k Compared to existing datasets such as, LaSOT and TrackingNet, the performance achieved on AVisT is significantly lower highlighting the challenging nature of the proposed dataset.

Recent years have witnessed a significant progress in the field of visual tracking with a plethora of trackers introduced in the literature. One of the major contributing factors towards these recent advances in tracking is the introduction of several benchmarks [38, 42, 54, 85, 109]. OTB [109] was one of the first large-scale datasets, containing 100 videos. Afterwards, tracking benchmarks are introduced to evaluate different aspects of tracking such as, the impact of color information [70], fast target motion [42] as well as tracking in aerial imagery [84]. More recently, the tracking community has focused on constructing datasets [38, 54, 85] with large-scale training splits, to benefit from task-specific deep learning. Among these, GOT-10K [54] comprises a large collection of shorter videos, whereas LaSOT [38] focuses on longer sequences. Moreover, there exist dedicated benchmarks, such as the VOT series [62] associated with annual tracking challenge competitions.

While all of these datasets have greatly benefited the tracking research, they no longer pose the same difficulty as before to the current state-of-the-art trackers, due to the rapid progress in the field. Most notably, the state-of-the-art trackers now achieve AUC scores of above 70% also on LaSOT (see Tab. 5.1), which is one of the most difficult established datasets. On the other hand, since the introduction of OTB in 2013 [110], the existence of highly difficult

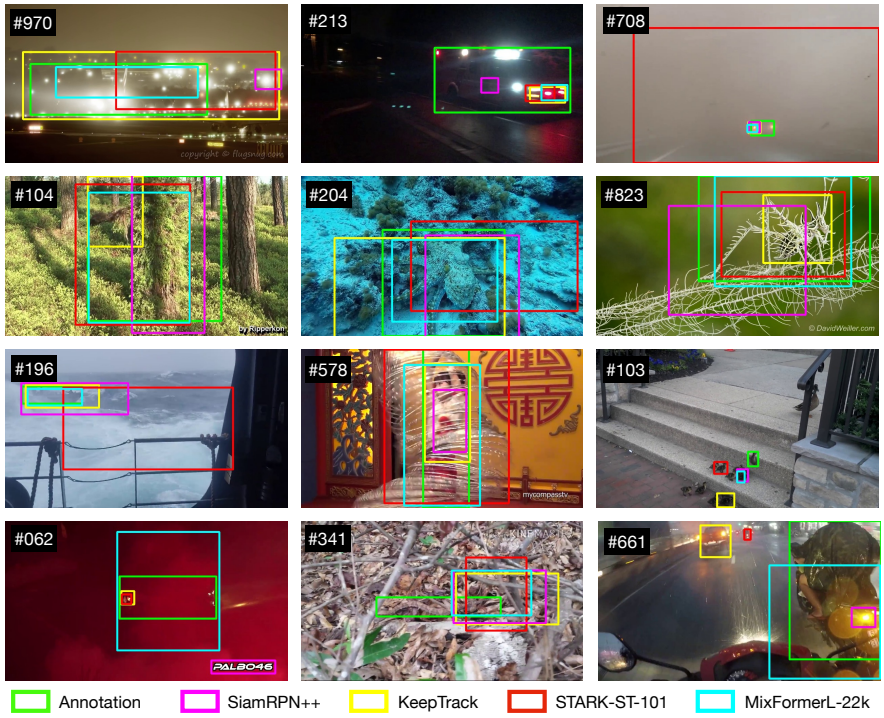


FIGURE 5.1: AVisT comprises challenging diverse tracking scenarios with adverse visibility. The diverse scenarios cover adverse weather conditions, including dense fog, heavy rain and sandstorm; obstruction effects such as, fire and sun glare; illumination effects; target effects, including distractor objects and small targets; along with camouflage. Here, we show individual frames of some representative sequences and visualize with different colors the ground truth annotations and the predicted bounding boxes of four different trackers. The trackers belong to different tracking paradigms: KeepTrack [83] (Discriminative Classifier), SiamRPN++ [67] (Siamese), STARK-ST-101 [117] and MixFormerL-22k [18] (Transformer).

tracking benchmarks has been vital, in order to challenge researchers to designing ever more robust and accurate trackers, applicable for increasingly diverse scenarios. In this work, we therefore set out to develop a new, highly challenging dataset, in order to promote further progress in the visual tracking field.

We believe that one of the main reasons that the aforementioned datasets do not pose sufficient challenge to new trackers is that diverse scenarios such as, adverse visibility due to weather conditions, camouflage and illumination effects are underrepresented. In practice, robust handling of adverse visibility is essential in many applications. For instance, autonomous driving applications require the target to be tracked under all weather conditions, such as heavy rain, dense fog, and sandstorms. Similarly, rescue missions involving drones require robust and accurate object tracking in adverse scenarios, such as fire, smoke, and strong winds. Further, wildlife conservation often relies on monitoring different animal populations in their natural habitats, where many animal species are difficult to distinguish from the surrounding environments due to their camouflaged appearance.
Contributions: We propose AVisT, a benchmark for visual object tracking in diverse scenarios with adverse visibility. AVisT better accommodates the difficult conditions encountered in the aforementioned real-world applications, while being severely challenging even for the most recent trackers (see Tab. 5.1). Our dataset comprises 120 challenging sequences, spanning 18 diverse scenarios and 42 object categories. The scenarios cover adverse weather conditions, including heavy rain, dense fog, and hurricane; obstruction effects such as, splashing water, fire, sun glare, and smoke; adverse imaging effects; target effects such as, fast motion and small target; along with camouflage. The proposed AVisT is densely annotated with accurate bounding boxes following a thorough quality control. Moreover, every frame is annotated with flags for occlusion, partial occlusion, out of view, and extreme visibility.

We evaluate 17 popular trackers on AVisT, including the most recent state-of-the-art methods. The best method, MixFormer-22k [18] which employs an ImageNet-22K pre-trained backbone achieves an

AUC score of only 56.0%, demonstrating the challenging nature of AVisT. We further analyze the performance of different trackers across attributes, which can provide valuable insights for specific applications. For instance, we note that ImageNet-22K pre-training is important for improved performance on the weather conditions attribute. Fig. 5.1 shows a qualitative comparison of recent trackers belonging to different tracking paradigms: discriminative classifiers, Siamese networks and Transformers.

5.2 THE AVIST BENCHMARK

5.2.1 *Scenarios and Attributes*

Our AVisT offers a dedicated dataset that covers a variety of adverse scenarios highly relevant to real-world applications. Importantly, AVisT poses additional challenges to the tracker design due to adverse visibility. To this end, our AVisT covers a wide range of 18 diverse scenarios: rain, fog, hurricane, fire, sun glare, low-light, archival videos, fast motion, distractor objects, occlusion, snow, sandstorm, tornado, smoke, splashing water, camouflage, small objects and deformation. These diverse scenarios are broadly categorized into five attributes: *weather conditions*, *obstruction effects*, *imaging effects*, *target effects* and *camouflage*. A short description of each scenario and their partitioning into attributes are presented in Tab. 5.2. The frequency of each scenario and attribute is visualized in Fig. 5.2. Next, we describe the included attributes.

Weather Conditions: While most existing benchmarks, such as LaSOT comprises sequences acquired under normal weather conditions, the tracking problem becomes more challenging in adverse weather scenarios, which often lead to bad visibility. In our dataset, the diverse adverse scenarios caused by weather conditions are: rain, fog, hurricane, snow, sandstorm and tornado. Accurately capturing the target appearance information in such extreme scenarios (see Fig. 5.3) is crucial and poses additional challenges to the tracking model.

Attribute	Scenario	Description
Weather Conditions	Rain	Heavy rain that compromises the visibility of the target.
	Snow	Heavy snowfall or snow conditions, affecting target visibility.
	Fog	Dense fog that severely affects target visibility.
	Sandstorm	Dense sand and dust in the air, severely impairing target visibility.
	Hurricane	Severe winds accompanied by rain and lightning.
	Tornado	Presence of a tornado, hampering target visibility.
Obstruction Effects	Occlusion	The target is occluded by another object or background structures.
	Splashing water	Splashing water in front of or on the target.
	Fire	Fire obstructing the target as well as causing lighting variation.
	Smoke	Dense smoke obstructing the view of the target.
	Sun glare	Sun glare effects that reduces the visibility of the target.
Imaging Effects	Low-light	Poor scene lighting conditions.
	Archival	Monochrome archival videos of poor quality.
Target Effects	Fast motion	The target motion is greater than the target size.
	Small target	In at least one frame, the target box is smaller than 500 pixels.
	Distractor objects	Presence of several objects that are visually similar to the target.
	Deformation	The target undergoes shape changes during tracking.
Camouflage	Camouflage	The target appearance is very similar to the surrounding background.

TABLE 5.2: A brief description of 18 different adverse scenarios, grouped into five broader attributes (weather conditions, obstruction effects, imaging effects, target effects and camouflage), in the proposed AVisT dataset.

Obstruction Effects: Apart from difficult weather conditions, there are several real-world obstructions that pose additional challenges to the tracker. These obstructions can be caused by occlusion as well as natural phenomenon such as, fire, smoke, sun glare and splashing water. Our dataset covers all these diverse settings of obstructions (see Fig. 5.3).

Imaging Effects: Challenging imaging conditions such as, low-light, night-time and archival monochrome videos causes losing the natural color of the target, and thereby pose difficulties to the tracking model. Our AVisT dataset comprises a set of challenging sequences covering these imaging effects (see Fig. 5.3).

Target Effects: In addition to the aforementioned adverse scene scenarios, there are several target-related challenges in the real-world. The proposed AVisT dataset includes various target effects such as,

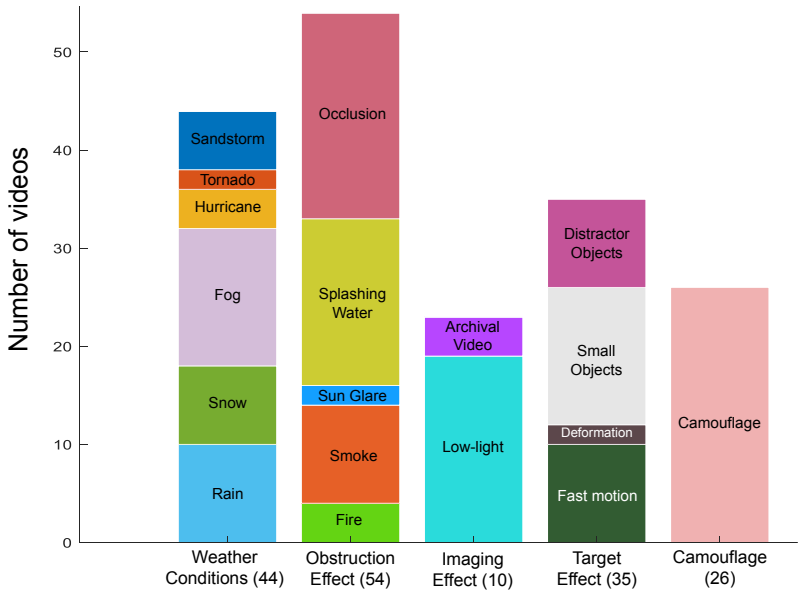


FIGURE 5.2: Distribution of image sequences with respect to scenarios and attributes in the proposed AVisT benchmark.

fast motion, small objects, deformations, and distractor objects (see Fig. 5.3).

Camouflage: Camouflage aims to conceal the object by making it blend into the background appearance. Most animal species utilize camouflage to various degrees, with some even changing their camouflage with the seasons. This cryptic coloration makes the target hard to distinguish from the surroundings. Compared to most existing tracking benchmarks, our dataset comprises a dedicated set of camouflage sequences that pose difficulties to state-of-the-art trackers (see Fig. 5.3).

5.2.2 *Data Collection*

As discussed earlier, AVisT aims to provide a benchmark for evaluating visual trackers under diverse scenarios with adverse visibility,

such as severe weather conditions, image, obstruction and target effects as well as camouflage. In addition, AVisT strives to achieve diversity with respect to target object classes (42 object categories in our dataset). With this objective, we first collect a large pool of around 400 videos from Youtube covering the 18 diverse scenarios with adverse visibility. We filter out unrelated contents in each video and retain the relevant clip for tracking. We then annotate the target object in the first frame of each trimmed video. Next, we qualitatively analyze two recent representative trackers, KeepTrack [83] and STARK [117], on these image sequences. We select a set of 120 sequences which are highly challenging for both these recent representative trackers. We note that other trackers such as, ToMP [82], and MixFormer-1k [18], which perform similar to KeepTrack and STARK on LaSOT [38], also perform similarly on our AVisT dataset. Therefore, the 120 videos we select are generally challenging and not specific to these two representative trackers. Among the initially large pool of around 400 videos, some of the camouflage sequences are overlapping with the MoCA dataset [66]. However, we re-annotate those camouflage sequences since the MoCA dataset was originally proposed for camouflage object detection and hence does not provide dense frame-level annotations required for visual object tracking.

To summarize, our AVisT dataset comprises 120 challenging videos from YouTube under the Creative Commons licence. All these 120 videos belong to at least one of the diverse scenarios, with a total of 80k annotated video frames. The frame-rates of these videos ranges from 24 to 30 Frames per Second (FPS) and the average sequence length is 664 frames (i.e., 22.2 seconds with 30 FPS). The shortest sequence in our dataset has 99 frames (3.3 seconds with 30 FPS), while the longest one has 3113 frames (103.7 seconds with 30 FPS).

5.2.3 *Dataset Annotation*

After data collection, the next step is to obtain high-quality annotations for all the sequences to ensure an accurate benchmarking

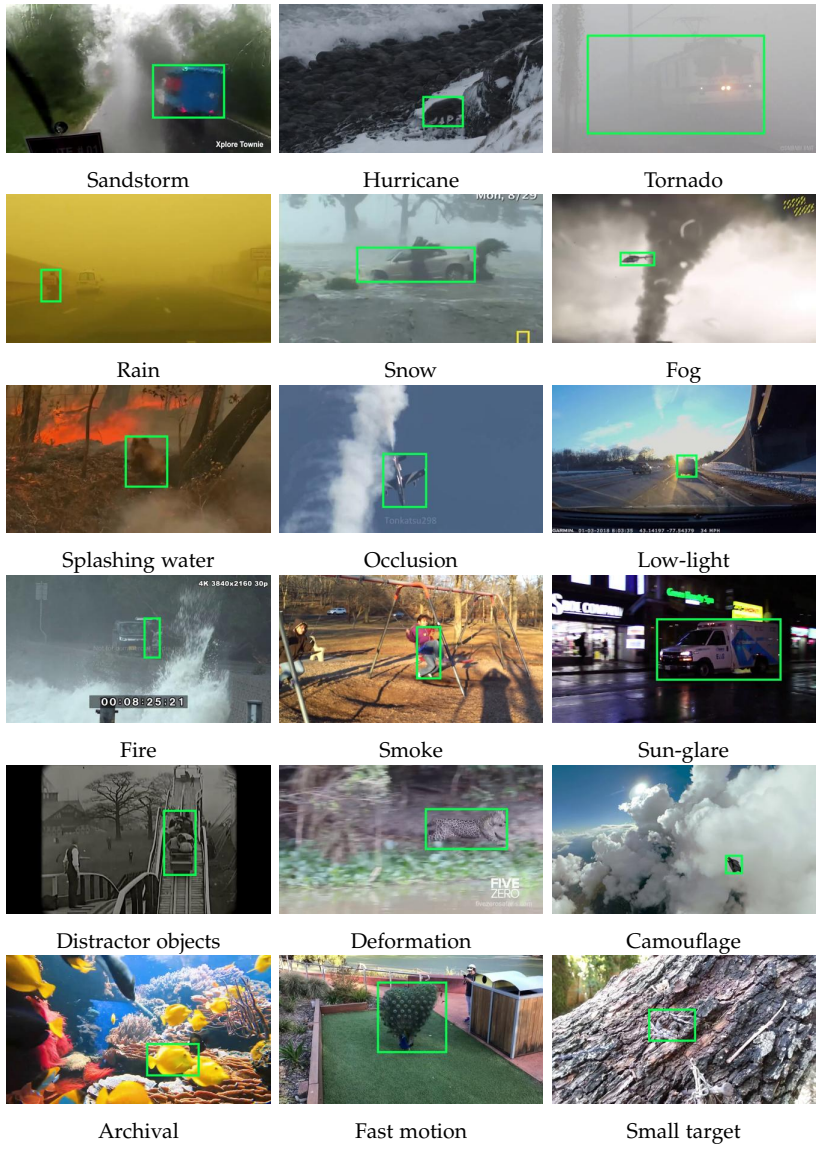


FIGURE 5.3: Video frames corresponding to different attributes namely Rain, Snow, Fog, Sandstorm, Hurricane, Tornado, Fire, Smoke, Sun glare, Splashing water, Occlusion, Low-light, Archival video, Fast motion, Small target, Distractor objects, Deformation, Camouflage.

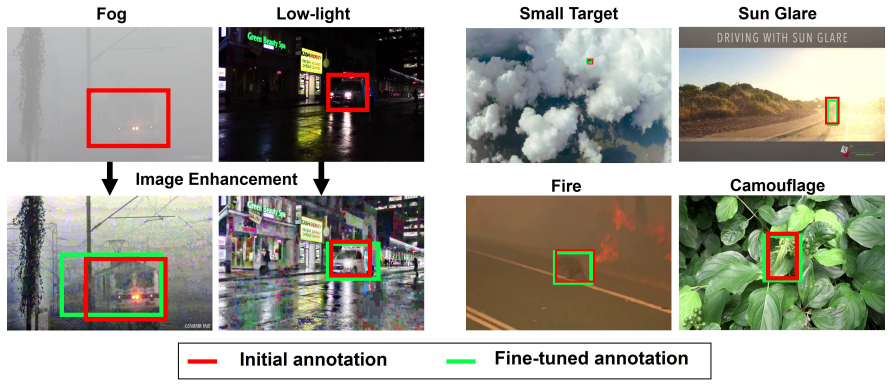


FIGURE 5.4: Example frames from diverse scenarios highlighting the complexity of the annotation process. In case of severe visibility, we employ image enhancement techniques to better distinguish the target aiding in improved annotations. Here, the example frames of different scenarios show that fine-tuned annotations (green color) better fit the target object region, compared to the initial annotations (red color).

of visual trackers. To obtain consistent annotations, we standardize a protocol that ensures high-quality annotations for the proposed AVisT dataset. During the annotation process, a video is processed by two teams, where the labeling team, comprising typically three members, manually draws the target object's bounding box as the tightest axis-aligned rectangle that fits the target in each frame of a video that has a specified tracking target. Afterwards, the validation team reviews the annotation results with either unanimously agreeing on the annotation results or returning it back to the labeling team for revising the annotation.

Quality Control: In case of diverse scenarios where the target object suffers from extreme visibility issues (e. g., dense fog and low-light), the annotation process becomes further challenging. Therefore, we employ standard image enhancement techniques, including contrast limited adaptive histogram equalization, histogram equalization, gamma correction, brightness and contrast adjustment, and white balance, to distinguish the boundary of the target and improve the

annotation quality (see Fig. 5.4 for fog and low-light examples). Furthermore, we also utilize the relative displacement of the target between consecutive frames to estimate its boundaries. As discussed earlier, we also improve the annotation quality by making separate teams for the process of labeling and validation. The labeling team manually annotates and cross checks the annotated video samples. Afterwards, the validation team verifies the quality of the annotated videos and points out any possible mistakes in the annotations which are then corrected by the respective team members. The annotation teams meticulously examine the annotations and often revise them in order to enhance the quality of the annotation. In the initial phase of validation, around 24% of the original annotations were corrected. Additionally, several frames underwent more than three revisions. Fig. 5.4 presents example frames where the initial annotations are fine-tuned, leading to improved annotation quality.

Our AVisT benchmark comprises different flags, where a frame can be labeled with full occlusion, partial occlusion, out-of-view and extreme visibility. A full occlusion flag is set when the target object is fully occluded by another object, such that the original pixel values are hard to be recovered. The partial occlusion flag implies that the target object is partially visible. In such a case, we annotate only the visible part of the target in the frame. The out-of-view flag refers to the case where the target object is not in the camera field-of-view. Here, we set a dedicated flag indicating that the target object is out-of-view in the frame. The extreme visibility flag refers to the cases of dense fog and extreme low-light where the target is suffering from severe visibility issues and is hard to identify for human eyes. We observe that image enhancement techniques helps in improving the annotation process in such cases.

5.3 EXPERIMENTS

5.3.1 *Evaluated Trackers*

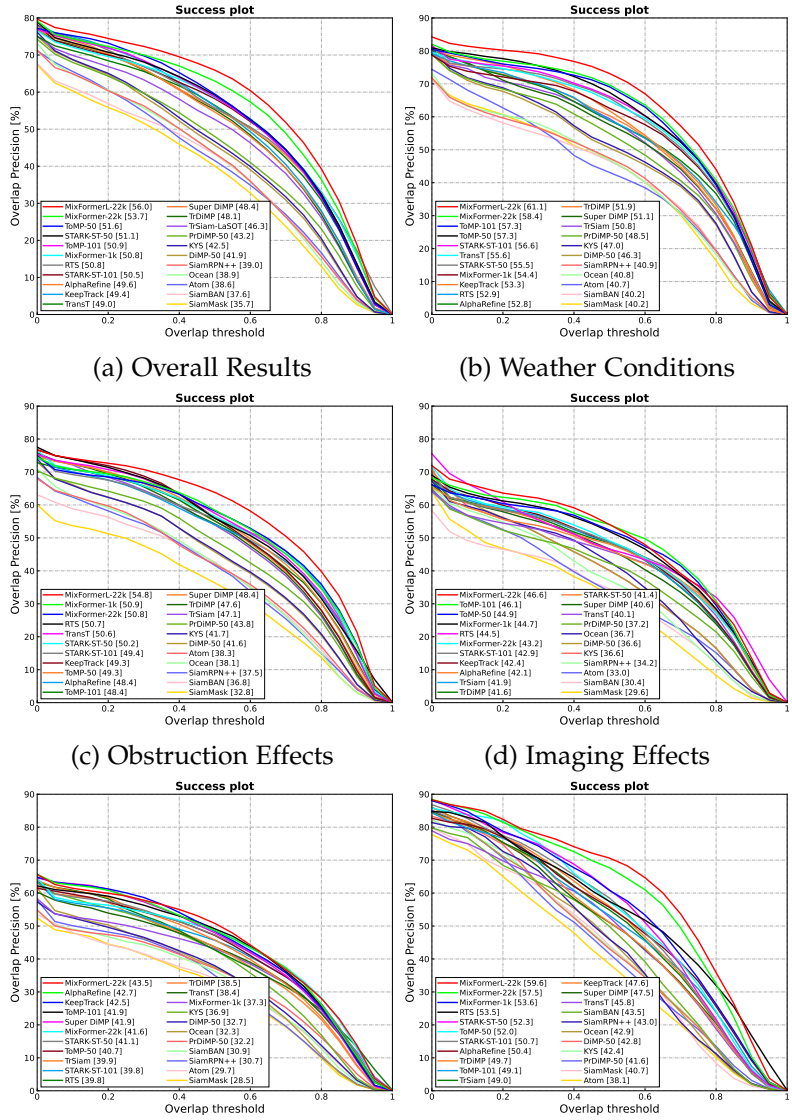
The field of generic tracking has greatly progressed in recent years with the development of various approaches. Siamese trackers employ a deep network to extract a target template that is matched with the features of the current video frame in order to localize the target therein. In contrast, trackers based on a discriminative classifier learn the weights of a convolutional kernel that allows to differentiate between the target object (foreground) and background regions in the current video frame. More recently, Transformer-based trackers have emerged that use self and cross attention layers to combine template and search frame information to extract discriminative features to localize the target. To analyse our AVisT, we evaluate high-performance and popular trackers that we briefly summarize bellow.

Siamese:: SiamRPN++ [67] employs a region proposal network to detect the target and to produce accurate bounding boxes. SiamMask [107] proposes an auxiliary binary segmentation loss and produces a segmentation mask and a bounding box for the target. SiamBAN [16] employs a box adaptive network that fuses multi-scale features to robustly localize targets of various scales. Ocean [128] employs an object aware anchor free network for target classification and bounding box regression.

Discriminative Classifiers:: Atom [23] uses an online trained two-layer fully convolutional neural network for target classification and employs a target estimation branch based on overlap maximization. DiMP [6] adopts the target estimation component of Atom but proposes an end-to-end learnable optimization-based model predictor that produces discriminative filter weights to localize the target. PrDiMP [30] and SuperDiMP [24] employ a probabilistic regression formulation. KeepTrack [83] uses SuperDiMP as a base tracker and employs a target candidate association network on top to reliably identify the target among distractor objects. Similarly, KYS [7] em-

Framework	Name	Backbone	Online Update	Venue	Success (AUC)	OP50	OP75
Siamese	SiamMask [107]	ResNet-50	✗	CVPR 2019	35.75	40.06	18.45
	SiamRPN++ [67]	ResNet-50	✗	CVPR 2019	39.01	43.48	21.18
	SiamBAN [16]	ResNet-50	✗	CVPR 2020	37.58	43.22	21.73
	Ocean [128]	ResNet-50	✓	ECCV 2020	38.89	43.60	20.47
Discriminative Classifier	Atom [23]	ResNet-18	✓	CVPR 2019	38.61	41.51	22.17
	DiMP-18 [6]	ResNet-18	✓	ICCV 2019	40.55	44.07	23.67
	DiMP-50 [6]	ResNet-50	✓	ICCV 2019	41.91	45.67	25.95
	PrDiMP-18 [30]	ResNet-18	✓	CVPR 2020	41.65	45.80	27.20
	PrDiMP-50 [30]	ResNet-50	✓	CVPR 2020	43.25	48.02	28.70
	Super DiMP [24]	ResNet-50	✓	CVPR 2020	48.39	54.61	33.99
	KYS [7]	ResNet-50	✓	ECCV 2020	42.53	46.67	26.83
	KeepTrack [83]	ResNet-50	✓	ICCV 2021	49.44	56.25	37.75
	AlphaRefine [118]	ResNet-50	✓	CVPR 2021	49.63	55.65	38.17
	RTS [89]	ResNet-50	✓	ECCV 2022	50.81	55.69	38.89
Transformer	TrSiam [106]	ResNet-50	✓	CVPR 2021	47.82	54.84	33.04
	TrDiMP [106]	ResNet-50	✓	CVPR 2021	48.14	55.26	33.77
	TransT [14]	ResNet-50	✗	CVPR 2021	49.03	56.43	37.19
	STARK-ST-50 [117]	ResNet-50	✓	ICCV 2021	51.11	59.20	39.07
	STARK-ST-101 [117]	ResNet-101	✓	ICCV 2021	50.50	58.23	38.97
	ToMP-50 [82]	ResNet-50	✓	CVPR 2022	51.60	59.47	38.87
	ToMP-101 [82]	ResNet-101	✓	CVPR 2022	50.90	58.77	38.42
	MixFormer-1k [18]	MAM	✓	CVPR 2022	50.83	58.56	39.30
	MixFormer-22k [18]	MAM	✓	CVPR 2022	53.72	62.98	43.02
	MixFormerL-22k [18]	MAM	✓	CVPR 2022	**55.99**	**65.92**	**46.34**

TABLE 5.3: Comparison of different trackers in terms of AUC score on AVisT. Other than MixFormerL-22k and MixFormer-22k, the evaluated trackers utilize backbones trained on ImageNet-1K dataset. Among existing trackers, STARK-ST-50 and ToMP-50 achieve comparable AUC scores. Both MixFormerL-22k and MixFormer-22k utilizing backbone with ImageNet-22K pre-training obtain improved tracking performance. In addition to AUC score, we also report performance at OP50 and OP75.



(a) Overall Results (b) Weather Conditions

(c) Obstruction Effects (d) Imaging Effects

(e) Target Effects (f) Camouflage

FIGURE 5.5: Comparisons in terms of success plots on AVisT. The AUC scores are given in the legend. In all cases, the recent MixFormer with a stronger backbone with ImageNet-22k pre-training achieves better performance. Best viewed zoomed in.

ploys DiMP-50 as baseline tracker but propagates dense localized state vectors that encode target, background or distractor information allowing to localized the target more robustly. In contrast, AlphaRefine [118] refines the preliminary bounding box generated by the base tracker SuperDiMP to increase the tracking accuracy.

Transformers:: TrDiMP [106] and TrSiam [106] employ a Transformer to enhance the extracted template and search features that are then used in a discriminative classification or Siamese setting. In contrast, TransT [14] directly extracts discriminative features using a feature fusion module performing self and cross attention operations. STARK [117] jointly fuses template and search features using a Transformer encoder consisting of self attention operations and uses a Transformer decoder together with an object query to predict the target state. ToMP [82] is inspired by DiMP but replaces the online-optimization based model predictor with a Transformer that predicts discriminative filter weights. In contrast to aforementioned trackers that use a feature extractor followed by a feature fusion module, MixFormer [18] only uses a mixed attention based backbone that allows to directly extract discriminative features.

5.3.2 *Evaluation Results*

Evaluation Metric: Following LaSOT [38], we evaluate the tracking performance using the One Pass Evaluation (OPE) [109], assessing the success score of different tracking methods. Success is calculated as the IoU of the ground truth bounding box and the tracking result. The AUC, which ranges from 0 to 1, is used to rank the trackers. Additionally, we present the results in terms of normalized precision plot in the suppl. material. We normalize the precision as in [85]. The precision is calculated by comparing the pixel distance between the ground-truth bounding box and the tracking result.

Quantitative Results: We perform comprehensive evaluations for each of the 120 videos in AVisT. Each tracker is evaluated with publicly available trained weights. Tab. 5.3 shows the performance,

in terms of AUC score, of trackers for the different frameworks. Among the trackers belonging to the Siamese-based framework, SiamRPN++ [67], SiamBAN [16] and Ocean [128] achieve AUC scores of 39.01, 37.58 and 38.89, respectively. Within the discriminative classifier framework, KeepTrack [83] and AlphaRefine [118] achieve comparable performance with AUC score of 49.44 and 49.63, respectively. Among existing Transformer-based methods employing backbones pre-trained on ImageNet-1k, STARK-ST-50 [117] and ToMP-50 [82] achieve obtain similar AUC scores of 51.11 and 51.60. The recently introduced MixFormer-1k [18] achieves AUC score of 50.83. A significant improvement in tracking performance is obtained when using ImageNet-22k pre-trained backbone in MixFormer [18], with MixFormerL-22k achieving AUC score of 56.0. We also report the overall success plot in Fig. 5.5a.

Attribute-based Comparison: In Figs. 5.5b-5.5f, we further evaluate the trackers on the five attributes in AVisT. We observe that a stronger backbone along with large-scale ImageNet-22k pre-training typically helps achieve better results (MixFormerL-22k [18]). However, the performance varies among attributes when using a tracker with same backbone that is either pre-trained on ImageNet-22k or ImageNet-1k (MixFormer-1k and MixFormer-22k). Further, the results also vary among attributes when using trackers all employing ImageNet-1k pre-trained backbones. For instance, MixFormer [18] significantly improves when using a backbone with ImageNet-22k pre-training on weather conditions attribute, compared to the same tracker and backbone but with ImageNet-1k pre-training (MixFormer-1k: 54.4 vs. MixFormer-22k: 58.4). However, we observe no improvement in performance possibly due to this large-scale pre-training when moving from MixFormer-1k to MixFormer-22k for obstruction effects. In case of imaging effects, ToMP-101 [82] with ImageNet-1k pre-trained backbone achieves an AUC score of 46.1, which is even comparable to that of MixFormerL-22k using a stronger backbone along with ImageNet-22k pre-training. For target effects, we observe MixFormer-1k to struggle compared to ToMP [82], KeepTrack [83]

and AlphaRefine [118]. To summarize, we observe that among recent trackers utilizing ImageNet-1k pre-trained backbones, no single method achieves better performance against its counterparts on all attributes. The comparisons further highlight the scope in designing a novel tracking mechanism which could tackle the diverse range of scenarios and attributes comprising sequences captured in real-world adverse conditions. More results are in suppl. material.

5.4 CONCLUSION

We introduce a new benchmark, AVisT, for visual tracking in diverse scenarios with adverse visibility. AVisT comprises 120 challenging videos, covering 18 diverse scenarios with 42 classes. These diverse scenarios are further grouped into five attributes. We evaluate a variety of recent Siamese, discriminative classifiers and Transformer-based trackers. Our experiments show that even the most recent Transformer-based tracker using a heavy ImageNet-22k backbone achieves an AUC score of only 56.0%, thereby highlighting the challenging nature of AVisT. We further analyze trackers based on attributes observing the need to design novel solutions that achieve favorable performance on real-world adverse tracking conditions.

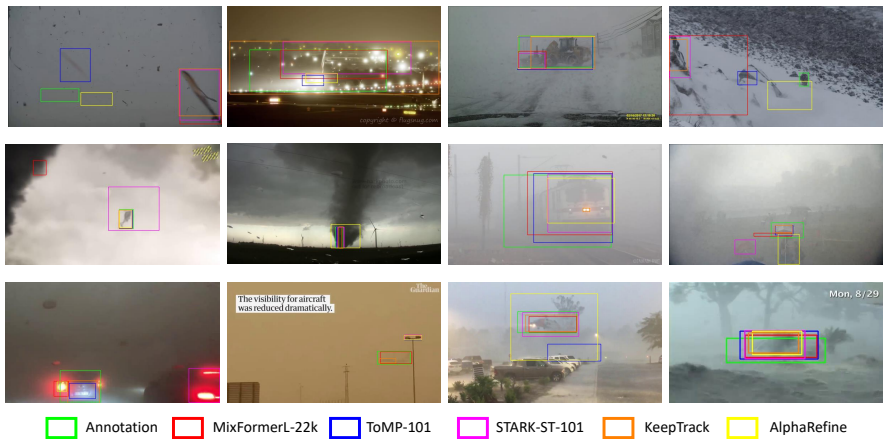


FIGURE 5.6: Here, we show frames of some representative sequences of **Weather Condition** attribute and visualize with different colors the ground truth annotations and the predicted bounding boxes of five different trackers: MixFormerL-22k [18], ToMP-101 [82], STARK-ST-101 [117], KeepTrack [83], and AlphaRefine [118]

5.5 APPENDICES

5.5.A *Visual Comparison per Attributes*

Figs. 5.6-5.10 shows the visual results of the trackers on attribute specific sequences from AVisT. From the visual results it is clear that the compared trackers faces major difficulty in intense lightning conditions e. g. Fig. 5.6 (first row second column) Fig. 5.8 (second row second column), extreme visibility conditions e. g. Fig. 5.6 (second row last column), Fig. 5.7 (second row second and third column), object deformation e. g. Fig. 5.9 (first row first column), distractor objects e. g. Fig. 5.9 (last row), small target e. g. Fig. 5.9 (first row last column), camouflage sequences e. g. Fig. 5.10 (second row second column).

FIGURE 5.7: Here, we show frames of some representative sequences of **Obstruction Effects** attribute and visualize with different colors the ground truth annotations and the predicted bounding boxes of five different trackers: MixFormerL-22k [18], ToMP-101 [82], STARK-ST-101 [117], KeepTrack [83], and AlphaRefine [118]

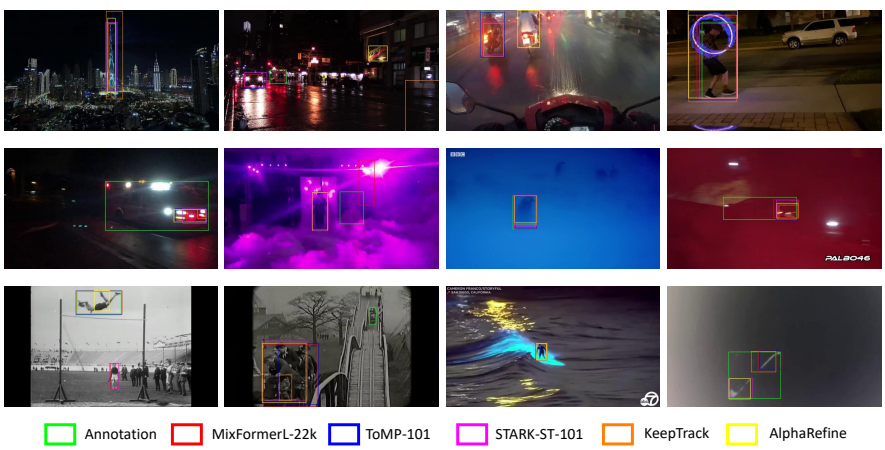


FIGURE 5.8: Here, we show frames of some representative sequences of **Imaging Effects** attribute and visualize with different colors the ground truth annotations and the predicted bounding boxes of five different trackers: MixFormerL-22k [18], ToMP-101 [82], STARK-ST-101 [117], KeepTrack [83], and AlphaRefine [118]

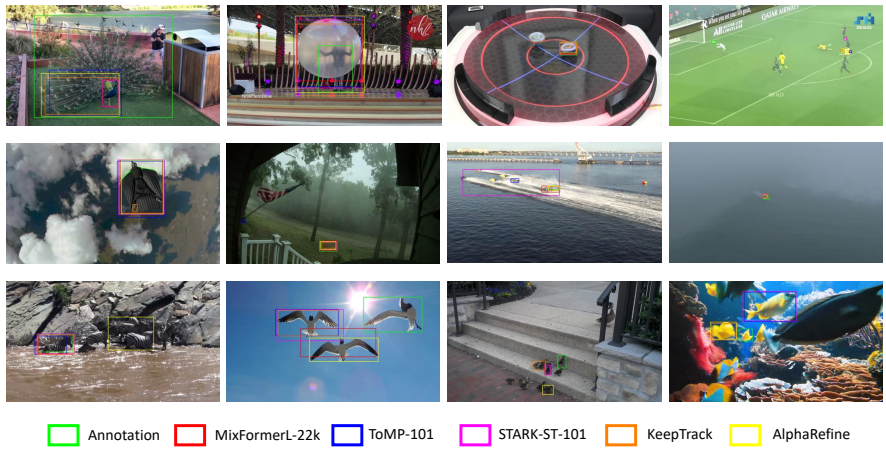


FIGURE 5.9: Here, we show frames of some representative sequences of **Target Effects** attribute and visualize with different colors the ground truth annotations and the predicted bounding boxes of five different trackers: MixFormerL-22k [18], ToMP-101 [82], STARK-ST-101 [117], KeepTrack [83], and AlphaRefine [118]

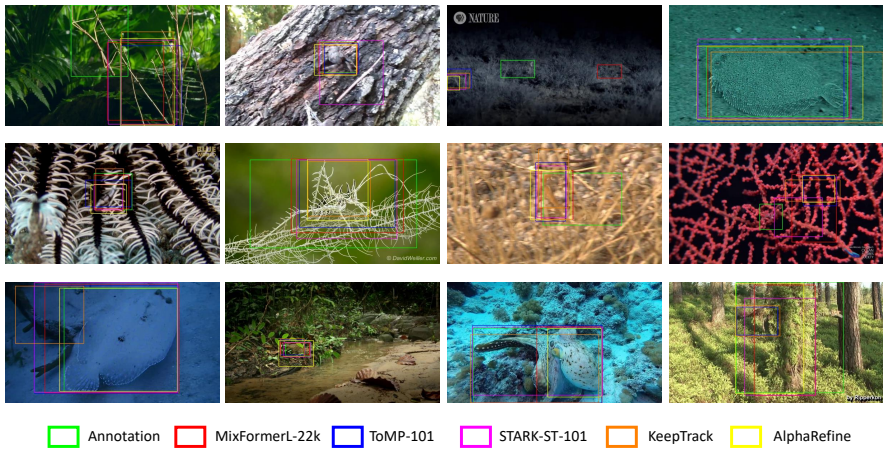


FIGURE 5.10: Here, we show frames of some representative sequences of **Camouflage** attribute and visualize with different colors the ground truth annotations and the predicted bounding boxes of five different trackers: MixFormerL-22k [18], ToMP-101 [82], STARK-ST-101 [117], KeepTrack [83], and AlphaRefine [118]

6

BEYOND SOT: IT'S TIME TO TRACK MULTIPLE GENERIC OBJECTS AT ONCE

In the previous chapters, we focused on tracking a single generic object per video sequence. However, multi-object GOT benefits from a wider applicability, rendering it more attractive in real-world applications. We attribute the lack of research interest into this problem to the absence of suitable benchmarks. In this chapter, we introduce a new large-scale GOT benchmark, LaGOT, containing multiple annotated target objects per sequence. Our benchmark allows researchers to tackle remaining challenges in GOT, aiming to increase robustness and reduce computation through joint tracking of multiple objects simultaneously. Furthermore, we propose a Transformer-based GOT tracker capable of joint processing of multiple objects through shared computation. Our benchmark, code, and trained models will be made publicly available at https://github.com/visionml/pytracking.

6.1 INTRODUCTION

Visual object tracking is a fundamental problem in computer vision. Over the years the research effort has been directed mainly to two different task definitions: Generic Object Tracking [5, 10, 25, 52, 62, 67, 109] and Multiple Object Tracking [13, 32, 45, 98, 122, 123, 126]. MOT aims at detecting and tracking all objects defined in a class category list, see Fig. 6.1. Hence, MOT methods can only detect and track objects of known class categories whereas all other objects are ignored. MOT methods are unable to track generic objects solely defined by a user-specified bounding box at test time. In contrast, GOT focuses on the scenario where a priori information about the object's appearance is unknown. Thus, the target model of the object's

appearance must be learned at test time from a single user-specified bounding box in the initial frame, see Fig. 6.1.

While GOT has a long history of active research, the problem of tracking multiple generic objects in the same video sequence has been largely ignored by the community. So far GOT methods and benchmarks focused on tracking a single object per video such that the term *Single Object Tracking (SOT)* was introduced. However, the task of GOT is not limited to tracking a single object. In fact, the ability to track multiple generic objects is desired in many real-world applications, such as surveillance, video understanding, semi-automatic video annotation, and industrial quality control. A method that jointly tracks multiple objects can achieve substantial reduction in computational cost through shared elements, compared to running a separate instance of a SOT method for each object. Moreover, processing multiple targets at the same time has the potential of increasing the robustness of the tracker by joint reasoning.

To facilitate the work on tracking multiple generic objects, we introduce the new multi-object GOT benchmark LaGOT. It provides up to 10 user-specified generic objects in the initial frame visible through the large part of the video. The target objects in one video may correspond to completely different and previously unseen class categories. Our benchmark features challenging characteristics such as fast moving objects, frequent occlusions, presence of distractors, camera motion, and camouflage. In total LaGOT contains 588k annotated objects of 102 different class categories and an average track length of 75 seconds.

Tracking multiple target objects in the same video poses key challenges and research questions that are typically overlooked by SOT methods. A multi-object GOT method needs to jointly produce multiple target models using the first-frame annotations. This allows the tracker to exploit annotations of potential distractors to improve the robustness of each target model. Furthermore, a joint localization step opens the opportunity for global reasoning across all tracks to reduce the risk of confusing similar objects. Furthermore, operating on multiple local search area [18, 82, 117] is no longer feasible for a

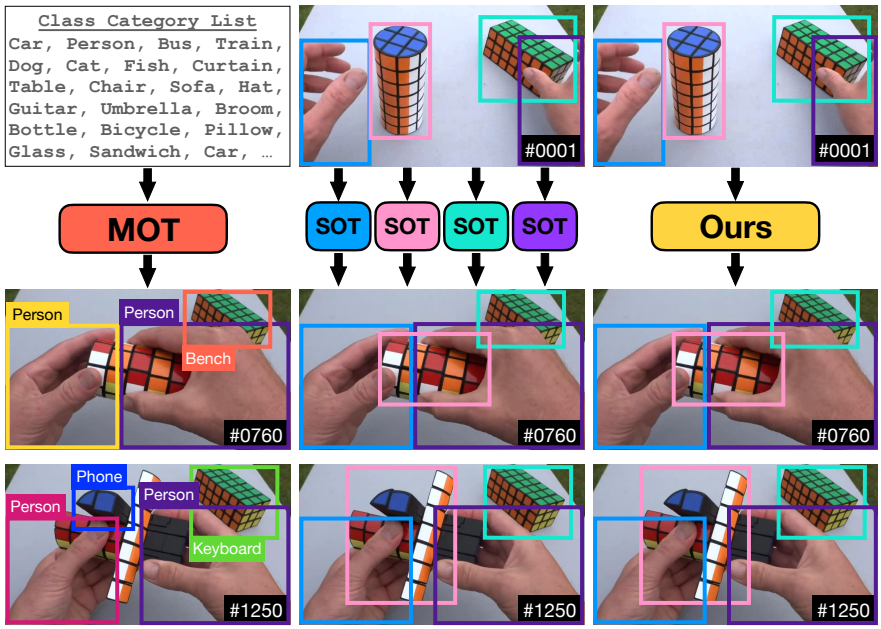


FIGURE 6.1: Multiple Object trackers (MOT) track all the objects corresponding to classes in a *predefined* category list, while all other objects are ignored. Single Object Tracking (SOT) methods focus on tracking only a single user-specified object per video. Thus, when encountered with multiple objects, such methods must resort to independent tracking of each object. This leads to a directly linear increase in computation. Our tracker can track multiple *generic* objects jointly that are defined via user-specified bounding boxes, leading to the opportunity of computational savings and to exploit inter-object information for improved robustness. The box colors correspond to track IDs.

multi-object GOT method because it is inefficient and complicates re-detecting lost objects.

We tackle these challenges by introducing a new multiple object GOT tracker. In order to track all desired target objects at once it operates globally on the entire frame. Furthermore, we propose a new generic multiple object encoding that allows us to encode multiple targets within the same training sample. We achieve this by learning a fixed size pool of different object embeddings, each representing a different target. Thus, we query the proposed model predictor with these object embeddings to produce all target models. In addition, we introduce a Feature Pyramidal Network (FPN) to increase the overall tracking accuracy while operating on full-frame inputs.

Contributions: Our main contributions are:

(i) We propose a novel large-scale multi-object GOT evaluation benchmark, LaGOT. It provides multiple annotated objects per frame with an average of 2.8 tracks sequence.

(ii) We further evaluate several baselines on LaGOT, including one MOT and six SOT methods. We assess their quality by using GOT and MOT metrics.

(iii) We develop a new baseline, TaMOs, a GOT tracker that tracks multiple generic objects at the same time efficiently. To achieve this, we propose a new multi-object encoding, introduce an FPN and apply the tracker globally on the entire video frame. TaMOs demonstrates almost constant run-time when increasing the number of targets and operates at an over $4\times$ faster run-time compared to the SOT baselines when tracking 10 objects.

(iv) We analyze TaMOs by assessing the impact of its different components using multiple benchmarks. Furthermore, TaMOs outperforms all baselines on LaGOT, while achieving excellent results on popular SOT benchmarks.

6.2 LAGOT BENCHMARK

In this section we first introduce the multi-object GOT task and discuss its differences to other object tracking tasks. Then, we introduce our new benchmark LaGOT.

6.2.1 *Multi-object GOT Task*

Multi-object GOT is the task of tracking multiple generic target objects in a video sequence. The target objects are defined by user-specified bounding boxes in the initial frame of the video. Thus, the target objects are generic in the sense that their class category is unknown and there might be no object of the same category in the training data, see Fig. 6.1.

Multi-object GOT vs. SOT: SOT requires to track only a single target object defined by the user [38, 62, 109], whereas multi-object GOT focuses on tracking multiple user-specified generic target objects in the same video.

Multi-object GOT vs. MOT: Multi-object GOT is a fundamentally different problem than MOT: **(i)** The MOT task requires to track all objects of known categories, whereas for multi-object GOT target objects in each video are defined by user-specified boxes. Consequently, multi-object GOT is a one-shot problem where the target objects are unknown at training time and are only available during inference. In contrast, MOT methods track all objects corresponding to the categories defined at training time. **(ii)** For the multi-object GOT task an object-id switch is equivalent to a complete failure since the user-specified object is no longer recoverable [79, 109]. Conversely, for MOT methods object-id switches are considered less problematic and are penalized less drastically by the MOT metrics [77].

Multi-object GOT vs. GMOT: Generic Multi Object Tracking (GMOT) focuses on tracking multiple objects of a single generic object class in each video. The class is defined by a single user-specified bounding box in the initial video frame [3, 40]. Thus, in contrast to multi-object

Dataset	TAO val [31]	GMOT-40 [3]	LaSOT val [38]	LaGOT
Task	MOT	GMOT	SOT	GOT
Object Definition	class list	1 box	1 box	n boxes
Num Classes per Video	≥ 1	1	1	≥ 1
Tracking Metrics	Track-mAP	MOTA/IDF1	Success AUC	F1-Score
Num Classes	302	30	70	102
Num Videos	988	40	280	294
Avg Video Length (num frames)	1010	240	2430	2258
Avg Track Length (num annos)	21	133	702	2106
Avg Tracks per Video	5.55	50.65	1	2.88
Num Annotations	115k	486k	680k	588k
Annotation Frequency	1 FPS	24-30 FPS	30 FPS	10 FPS

TABLE 6.1: Comparison of LaGOT with existing benchmarks that focus on related tasks to multi-object GOT.

GOT, a GMOT method is unable to track multiple objects of different categories in the same video.

6.2.2 *LaGOT*

Benchmark Construction: To build a challenging long-term multi-object GOT benchmark we require objects that are visible for a significant amount of time and at the same time are difficult to track. LaSOT [38] contains such diverse and relatively long videos (2430 frames or 81 seconds on average) with challenging tracking scenarios including fast moving objects, camera motion, various object sizes, frequent object occlusions, scale changes, motion blur, camouflage and objects that go out of view or change their appearance. LaSOT provides annotations for a single object in each video but typically multiple objects are present throughout the full sequence, which is desirable for long-term tracking scenarios. Thus, instead of collecting new videos, we used the popular LaSOT evaluation set and add new annotations for multiple objects in each sequence.

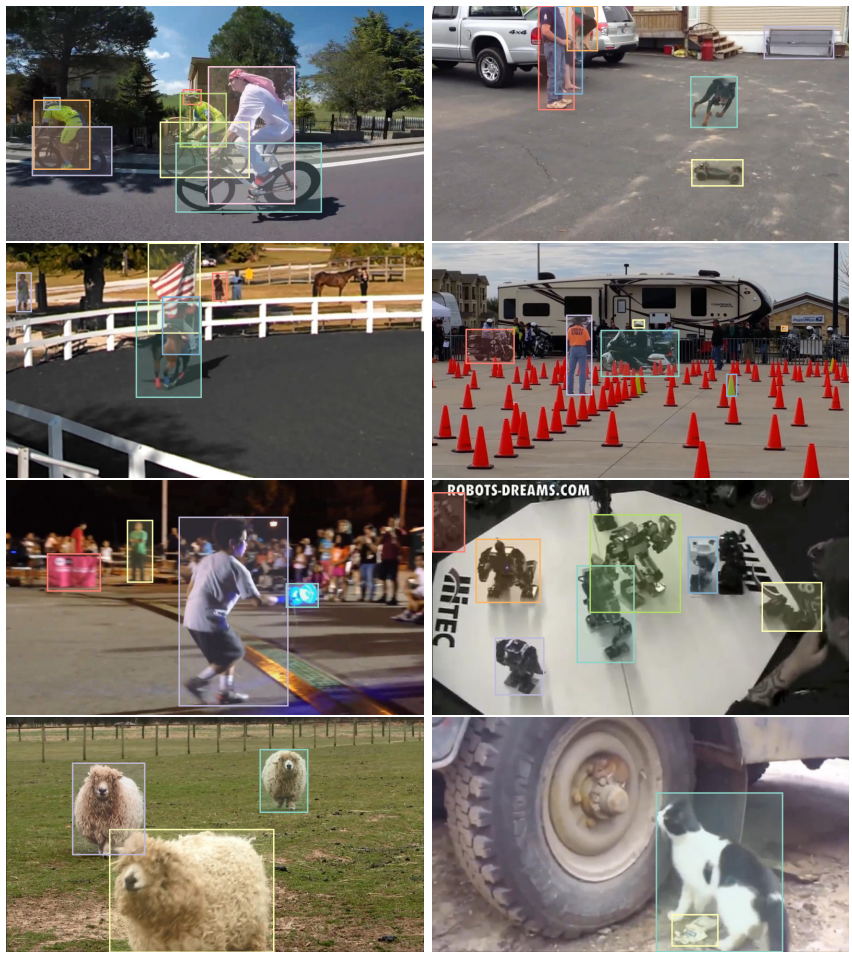


FIGURE 6.2: Examples of the annotated objects in the video sequences of our LaGOT dataset. The objects are annotated at 10 FPS. Notice the diversity of the annotated media as well as the complexity of the scenes.

Another large-scale video dataset we considered is TAO [31]. However, compared to LaSOT, TAO contains shorter videos with an average of 33 seconds and its outdoor and road sequences mainly focus on pedestrians and vehicles (60% of all objects in TAO). While the indoor sequences contain rarer object categories, they are often static and are only visible for a short time. Furthermore, TAO contains only sparse annotations (1 FPS). For all these reasons, we used LaSOT instead of TAO to build our benchmark. In contrast to TAO, GMOT-40 [3] contains dense annotations, but each video contains only annotated objects of the same class. Furthermore, GMOT-40 consists of only 40 short sequences (avg 240 frames or 8 seconds) rendering only 10 different object classes, see Tab. 6.1. Thus, GMOT-40 is unsuitable to serve as a multi-object GOT benchmark.

Annotation Protocol: First, we inspect all 280 sequences in LaSOT and identify in each video challenging target objects that play an active role and meet the previously specified criteria. Next, we entrust professional annotators to annotate the selected objects in all sequences on every third frame, leading to an annotation frequency of 10 FPS. They use an interactive annotation tool which incorporates an object tracker to speed up the annotation process [65]. A group of researchers verifies the newly obtained annotations and sends low-quality annotations back for correction until all annotations meet our high quality standards. Finally, we post-process the annotations to construct the final tracks. First, we remove all tracks shorter than 4 seconds. Second, we define the starting frame by manually selecting the earliest frame where as many annotated objects as possible are clearly visible. Third, it is not always possible to unambiguously associate all object identities over time due to occlusions and out-of-view events. Hence, we either remove ambiguous annotations or cut these videos into multiple sub-sequences, where the objects association is clear. Following this protocol guarantees a high annotation quality, see Fig. 6.2 for annotated example frames.

Statistics: Our benchmark LaGOT has 294 videos with 837 tracks leading to over 588'000 annotated objects. Thus, we almost triple

the number of tracks compared to the original LaSOT validation set (and the corresponding evaluation time from 381 to 975 min). Furthermore, we add 31 additional generic object classes, e. g. pool queue, propeller, tires or fabric bag. Overall our benchmark contains 10× more class categories than GMOT-40. The average track length of LaGOT is 2106 frames (702 annotated frames), which is 3× longer than in TAO, and almost 10× longer than in GMOT-40. We detail the evaluation protocol in Sec. 6.4.

Annotation Frequency: According to Valmadre *et al*. [102] it is more effective to spend a fixed annotation budget on many videos with sparse annotations than on fewer videos with dense annotations. Thus, we annotate every third frame to reduce the overall annotation cost. To analyze the difference between 10s and 30 FPS annotations, we evaluate five recent trackers on the tracks borrowed from LaSOT, where 30 FPS annotations are available. The mean relative error of the success rate AUC is only 0.237%. This shows that 10 FPS is sufficient on large-scale datasets such as LaSOT and LaGOT, leading to only minor score deviations.

6.3 METHOD

In this section we present our tracker TaMOs, which employs a Transformer to jointly model and track a set of arbitrary objects defined in the first frame of a video. We start from ToMP [82], a recent Transformer-based generic single object tracker. In Sec. 6.3.1 we introduce the proposed Transformer-based multi-object tracking architecture and in Sec. 6.3.2 we discuss the used training protocol.

6.3.1 *Generic Multi-Object Tracker - Overview*

An overview of the proposed generic multi-object tracker TaMOs is presented in Fig. 6.3. First, unlike original ToMP, our tracker operates on the full train and test images instead of crops. The target object encoder uses a pool of learnable object embeddings to encode the lo-

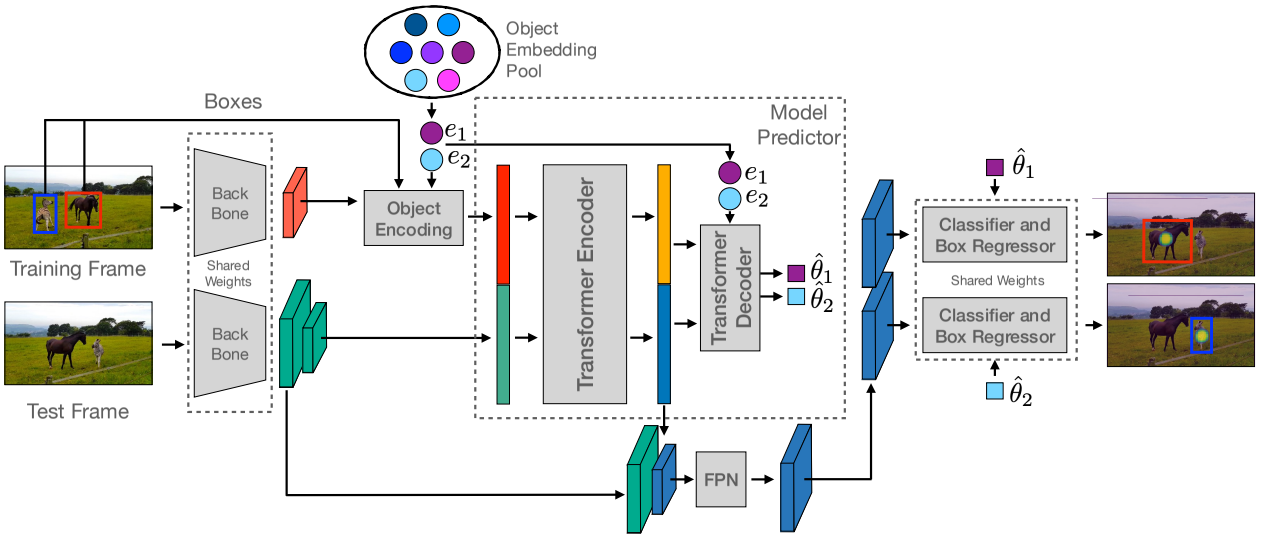


FIGURE 6.3: Overview of our tracker TaMOs for joint tracking of multiple targets. First, we extract features from training and test frames. All objects in the training frame are encoded jointly with a multi-object encoding and passed to the model predictor together with the training frame features. The model predictor produces target models $\hat{\theta}_i$ together with enhanced test features. We apply an FPN on the enhanced output features to generate higher resolution test features. Finally, we predict the bounding box of each target by applying the target model $\hat{\theta}_i$ via the classifier and box regressor for each target.

cation and extent of each target object within a single shared feature map (Sec. 6.3.1.1). The randomly sampled object embedding then represents a particular target in the entire video sequence: we use the object embedding to condition the model predictor to produce the target model that localizes the target object in the test frame (Sec. 6.3.1.2). Since operating on the entire video frame increases the computational cost of the Transformer operations, we are limited to a certain feature resolution. To track small objects we propose an FPN-based feature fusion of the test frame features produced by the Transformer with the higher resolution backbone features. We adopt the correlation filter based target localization and bounding box regression mechanism of ToMP but apply both on the higher resolution FPN features instead of the output features of the Transformer (Sec. 6.3.1.3).

6.3.1.1 *Generic Multiple Object Encoding*

To track a growing number of target objects efficiently, we propose a novel object encoding that allows to encode multiple objects in a shared feature map without requiring multiple templates. In addition, we process the entire video frame instead of a separate search region for each target.

In particular, we extend the single object encoding formulation of ToMP to be applicable for multiple objects. The idea is to replace the foreground embedding with multiple object embeddings, each representing a different target object. Thus, we create a pool $E \in \mathbb{R}^{m \times c}$ of $m \geq n$ object embeddings $e_i \in \mathbb{R}^{1 \times c}$. Then, we sample for each target object a random object embedding from the pool E without replacement. Next, we combine the object embeddings with the Gaussian score map $y_i \in \mathbb{R}^{h \times w \times 1}$ that represents the center location of the target object i and the LTRB [99, 115] bounding box encoding $b_i^{\text{ltrb}} \in \mathbb{R}^{h \times w \times 4}$. The final encoding is thus:

$$f_{\text{train}}^{\text{enc}} = f_{\text{train}} + \sum_{i=0}^{n} e_i \cdot y_i + \sum_{i=0}^{n} e_i \cdot \phi\left(b_i^{\text{ltrb}}\right), \tag{6.1}$$

where ϕ is a MLP and $n \leq m$ is the number of tracked objects. Note, that in contrast to the object encoding step in ToMP, we not only use the object embedding to encode the Gaussian score map but also the bounding box representation. The object embeddings e_i are learned during training such that the model is able to disentangle the shared feature representation and can identify each object in the training and test features. Note, that the products in Eq. (6.1) employ multiplications with broadcasting across every dimension whereas the latter uses channel-wise multiplication with broadcasting across the spatial dimensions.

6.3.1.2 *Joint Model Prediction*

Now that the target object locations and extents are embedded in the training features, we require a model predictor to produce a target model for each encoded object. The target models are then used to localize the targets in the test frame and to regress their bounding boxes. In order to easily associate the different targets over time, we require a model predictor that can be conditioned on the targets, or in our case on the different object embeddings e_i. Furthermore, the model needs to be able to produce all target models jointly to increase the efficiency.

We extend the aforementioned single target model predictor of ToMP by keeping the Transformer encoder unchanged but by modifying the Transformer decoder. In particular, we query the Transformer decoder with multiple object embeddings e_i at the same time instead of a single foreground embedding,

$$[\hat{\theta}_1, \dots \hat{\theta}_n] = T_{\text{dec}}([\tilde{h}_{\text{train}}, \tilde{h}_{\text{test}}], [e_1, \dots e_n]) \,. \tag{6.2}$$

Here, $\hat{\theta}_i$ is the target model and n is the number of target objects encoded in the training frame.

6.3.1.3 *Target Localization and Box Regression*

We use the generated target models to localize the targets and to regress their bounding boxes. We produce a correlation filter for

target classification and adopt the bounding box regression branch of ToMP [82]. But instead of applying the target classifier and box regressor on the low-resolution test features h_{test} of the Transformer encoder, we use high resolution features generated with an FPN:

$$z^{\text{low}}, z^{\text{high}} = \psi(h_{\text{test}}, f_{\text{test}}^{\text{high}}), \tag{6.3}$$

where $f_{\text{test}}^{\text{high}} \in \mathbb{R}^{2h \times 2w \times c}$ are the high resolution test features extracted at an earlier stage of the backbone, and $\psi(\cdot)$ denotes the FPN. The high-resolution multi-channel score map can then be obtained as follows

$$\hat{y}_i^{\text{high}} = w_i^{\text{cls}}(\hat{\theta}_i) * z^{\text{high}}, \quad 0 \leq i < n, \tag{6.4}$$

where $w_i^{\text{cls}}(\hat{\theta}_i)$ refers to the discriminative correlation filter for the target object i obtained from the model predictor $\hat{\theta}_i$. Similarly we obtain the multi-channel bounding box regression maps $\hat{b}_i^{\text{high}}$. Note, that during inference we only use the high-resolution score and bounding box prediction maps. During training we apply all target models directly on the Transformer encoder features h_{test}, similar to ToMP, but also on the low- and high-resolution FPN feature maps $z^{\text{low}}, z^{\text{high}}$. We empirically observed better training performance when applying the losses on each instead of only on the high resolution outputs.

6.3.2 *Training*

During training we employ a classification and a bounding box regression loss. We compute both losses for the predictions obtained by processing each FPN feature map (low-res and high-res) as well as the output test features h_{test} of the Transformer encoder. The classification loss is

$$L_{\text{cls}} = \sum_{i=0}^{n} L_{\text{focal}}(\hat{y}_i, y) + \sum_{j=n}^{m} L_{\text{focal}}(\hat{y}_j, 0), \tag{6.5}$$

Here we assume that the first n object embeddings e_i were used to encode the n objects marked in the training frame whereas the

remaining $m - n$ object embeddings were not used to encode any objects. Thus, we require that the resulting score maps $\hat{y}_j$ that correspond to an unused object embedding e_j produce low scores everywhere (second sum in Eq. (6.5)). This step tightly couples the object encoding and decoding. Omitting this term not only decreases the overall performance but slows down the training progress.

In contrast to classification we enforce the bounding box regression loss only for the predictions that actually correspond to an encoded object and ignore those corresponding to unused object embeddings. The regression loss is given by

$$L_{\text{bbreg}} = \sum_{i=0}^{n} L_{\text{GIoU}} \left(\hat{b}_i^{\text{ltrb}}, \hat{b}_i^{\text{ltrb}} \right), \tag{6.6}$$

where L_{GIoU} denotes the generalized IoU-Loss [91]. The overall training loss is then defined as

$$L_{\text{tot}} = \lambda_{\text{cls}} L_{\text{cls}}(\hat{y}, y) + \lambda_{\text{bbreg}} \cdot L_{\text{bbreg}}(\hat{b}^{\text{ltrb}}, b^{\text{ltrb}}) \tag{6.7}$$

where λ_{cls} and λ_{bbreg} are scalars weighting the contribution of each loss component.

Training Details: We randomly sample an image pair consisting of one training and one test frame from a training video. Since our tracker operates on full frames, we retain the full training and testing frames. These are re-scaled and padded to a resolution of 384×576. As we use the feature maps with stride 16 for both the ResNet-50 [50] and SwinBase [75] backbones, this results in an extracted feature and score map resolution of 24×36. For ResNet-50 we use pretrained weights on ImageNet-1k and for SwinBase on ImageNet-22k.

We train our tracker on the training splits of LaSOT [38], GOT10k [54], TrackingNet [85], MS-COCO [73], ImageNet-Vid [92], TAO [31], and YoutubeVOS [114]. Note, that we remove all videos from the TAO training set that overlap with the evaluation set of LaSOT. We randomly sample for each epoch 40k image pairs with equal probability from all datasets. In order to leverage SOT datasets and training all object embeddings e_i equally, we assign random object ids to

all objects in the sampled training pair, Note, that both SOT and MOT datasets are crucial to train the proposed tracker. Without MOT datasets the tracker is unable to learn multiple target models a the same time and avoiding SOT datasets leads to inferior tracking quality. We train the tracker for 300 epochs on 4 Nvidia A100 GPUs using ADAMW [76] with a learning rate of 0.0001 that we decay after 150 and 250 epochs by a factor of 0.2. Our method is implemented using PyTracking [24].

Inference: During inference we adopt the simple memory updating approach described in [82]. In particular, updating the memory refers to adding a second dynamic training frame using predicted box annotations. We replace the second training frame (update the memory) if the maximal value in each score map is above the threshold of $\tau = 0.85$.

6.4 EXPERIMENTS

To illustrate the challenges of our proposed GOT benchmark, we evaluate several recent trackers that serve as baselines along with our proposed tracker on our GOT benchmark LaGOT, which requires tracking multiple objects in the same video (Sec. 6.4.1). Furthermore, we compare our method to recent trackers on several SOT benchmarks (Sec. 6.4.2). Finally, we present an ablation study, evaluating the impact of different components of our tracker.

6.4.1 *State-of-the-Art Evaluation on LaGOT*

We evaluate our tracker with a ResNet-50 and a SwinBase backbone as well as 6 SOT (SuperDiMP [24], KeepTrack [83], TransT [14], STARK [117], ToMP [82], and MixFormer [18]) and one MOT (QDTrack [88]) tracker on LaGOT.

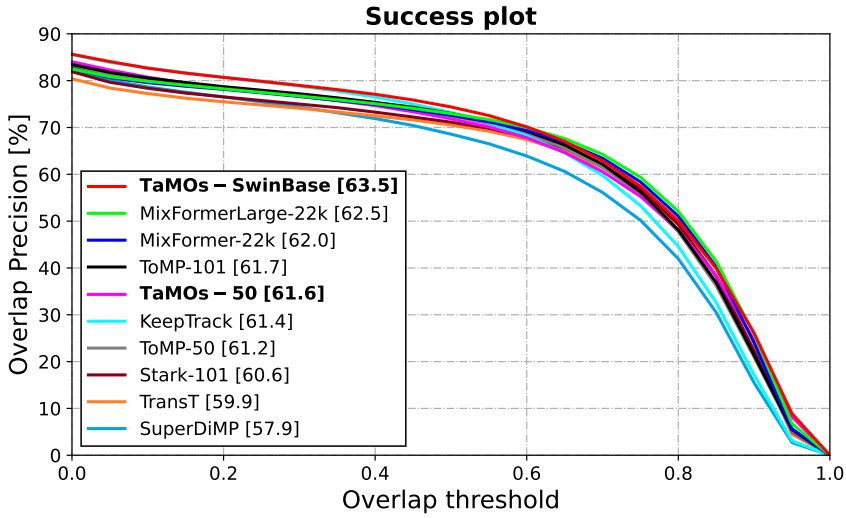
Metrics: We measure the performance of a tracker in the OPE setting. We employ the standard GOT Success rate AUC metric [37, 38, 42, 84, 85, 87, 109]. However, success rate does not account for false

positive predictions when a target gets occluded or is out of view. While this is not a big issue in standard SOT datasets, where the target object is present in the vast majority of frames, it becomes vital in long-term tracking. In LaGOT objects are more frequently invisible due to occlusions or are out-of-view. To capture this aspect, we employ the VOTLT [60, 79] metric that penalizes false positives. It computes the IoU-weighted precision-recall curve and ranks the trackers according to their F1-score.

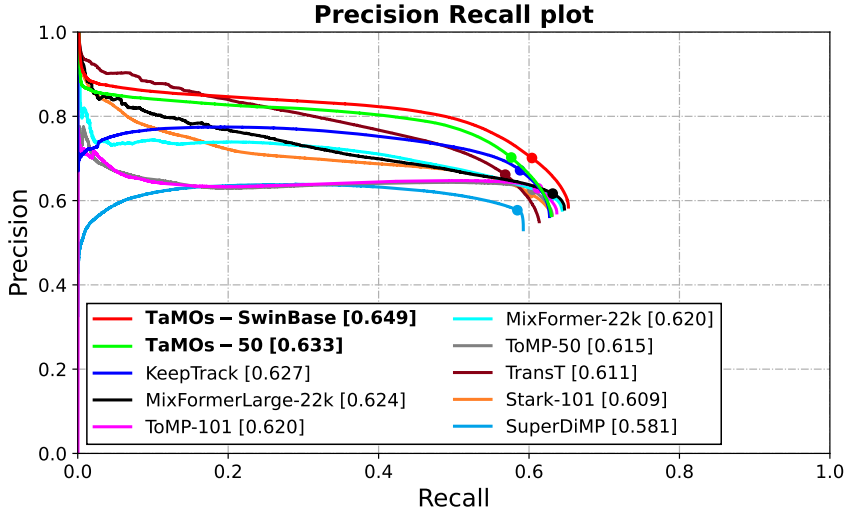
6.4.1.1 *Comparison to SOT Methods*

SOT methods are designed to track a user-specified object defined by a bounding box in each video. However, SOT trackers are limited to track only a single target at once. Thus, multiple instances of the same tracker need to be run in parallel to track multiple objects in the same sequence leading to a linearly increasing run-time, see Fig. 6.1.

Results: Fig. 6.4a shows the success rate of all trackers on LaGOT. We observe that SOT trackers perform well on LaGOT. However, our multi-object tracker TaMOs achieves the best AUC, even outperforming the state-of-the-art SOT tracker MixFormerLarge-22k [18]. We further observe that TaMOs is as robust as KeepTrack [83] ($T < 0.4$), where the gap to the remaining trackers is particularly prominent. This demonstrates the potential of a global multiple object GOT method. Fig. 6.4b shows the *tracking* Precision-Recall curve on LaGOT. Both versions of TaMOs outperform all other SOT trackers. The highly robust object presence scores predicted by our tracker lead to a superior precision at all recall rates > 0.2. Moreover, our approach achieves the best maximal recall and outperforms all previous methods in VOTLT by 2.2 points. This demonstrates that joint tracking of multiple objects and global search benefit the object localization and identification capabilities of the tracker. For further insights we show MOT metrics on LaGOT in Tab. 6.3. Our tracker achieves the best results for every MOT metric and outperforms MixFormerLarge-22k by 5.7 points in MOTA.



(a) Success Plot



(b) Precision-Recall

FIGURE 6.4: Success plot, showing OP_T, on LaGOT (AUC is reported in the legend). Tracking Precision-Recall curve on LaGOT – VOTLT is reported in the legend (the highest F1-score).

	1 Object	2 Objects	5 Objects	10 Objects
ToMP-50	34.7	17.4	7.0	3.4
TaMOs-50	19.2	17.9	16.3	13.9

TABLE 6.2: Run-time analysis (in FPS) between our baseline model ToMP and our tracker TaMOs.

Run-Time Analysis: We evaluate the run-time on a single A100 GPU. Tab. 6.2 reports a run-time analysis of our tracker TaMOs compared to ToMP, with both employing a ResNet-50 backbone. While TaMOs is slower than ToMP for a single object, due to the higher resolution required for full-frame tracking, our approach already reaches an advantage for 2 concurrent objects. As ToMP needs to run a separate independent tracker for each new object, our approach achieves a $4\times$ speedup for 10 concurrent objects. Furthermore, the analysis demonstrates that TaMOs achieves almost a constant run-time even when increasing the number of targets. TaMOs-SwinBase achieve 13.1 FPS for a single object and 9.3 FPS when jointly tracking 10 objects.

6.4.1.2 *Comparison to MOT Methods*

MOT methods are designed to track multiple objects in a video sequence and are thus used as baselines for LaGOT. However, there is no straightforward way to adapt an MOT tracker to the multi-object GOT task, due to their radically different inputs. In MOT the targets are defined via a list of classes. Thus, an MOT tracker cannot easily be used to track arbitrary objects defined by a user-specified bounding box in the initial frame. In order to be able to track generic objects MOT methods need to be trained on large vocabulary datasets. Then, we can greedily match the detected tracks with the user-defined bounding boxes on the initial frame to track user-specified objects. However, if a user-specified object corresponds to an unknown class the tracker is unable to track this object. We evaluate QDTrack [88] that is trained on LVIS [48] and TAO as MOT baseline.

		F1-Score	Success	HOTA	MOTA	IDF1	OWTA
GOT	**TaMOs-SwinBase**	**0.649**	**63.5**	**62.5**	**58.3**	**75.2**	**69.4**
	TaMOs-50	**0.633**	61.5	60.4	**53.2**	72.6	67.6
SOT	MixFormerLarge-22k	0.624	**62.5**	**61.7**	52.6	**74.6**	**69.3**
	ToMP-101	0.620	61.7	60.4	52.4	74.3	67.9
	STARK-101	0.609	60.6	59.9	49.8	73.2	67.0
	TransT	0.611	59.9	58.2	47.3	71.5	66.0
	KeepTrack	0.627	61.4	59.3	51.7	74.0	66.4
	SuperDiMP	0.581	57.9	56.3	43.3	69.8	64.1
MOT	QDTrack	0.189	19.4	22.1	-119.7	16.2	36.6

TABLE 6.3: Comparison of GOT and MOT metrics on LaGOT.

Results: QDTrack achieves a success rate AUC of 19.4 and an F1-score of 0.189 and thus performs inferior compared to all other trackers. QDTrack is simply not robust enough and fails to track unknown generic objects. To further explore the limitations of MOT methods in our setting, we evaluate a version 'QDTrack-Oracle', where we select the track ID that maximizes the AUC score on LaGOT. Even with such oracle information, it achieves a success rate AUC (29.1) and an F1-score (0.333) far inferior to any evaluated SOT baseline. In addition we evaluate QDTrack using all its predicted tracks with MOT metrics, see Tab. 6.3. However, QDTrack tracks multiple background objects that are not annotated in LaGOT leading to many False Positives (FPs). Thus, traditional MOT tracking metrics such as MOTA, HOTA and IDF1 are unsuitable to evaluate QDTrack on LaGOT. Instead, we concentrate on the OWTA metric [74] that focuses on Detection Recall (DetRe) and Association Accuracy (AssA) and thus ignores FPs. QDTrack achieves 36.6, which is still the lowest OWTA score among all trackers. QDTrack achieves a low DetRe (46.7 vs. 70.2 of TaMOs) because it is unable to detect target objects that appear rarely or are absent in TAO or LVIS. Furthermore, it achieves

Method	Venue	LaSOT [38]			TrackingNet [85]		
		Prec	N-Prec	Succ	Prec	N-Prec	Succ
TaMOs-SwinBase		**77.8**	79.3	70.2	**84.2**	**88.7**	**84.4**
TaMOs-50		75.0	77.2	67.9	82.0	87.2	82.7
SwinTrack [72]	NIPS'22	76.5	—	**71.3**	82.0	—	**84.0**
Unicorn [116]	ECCV'22	74.1	76.6	68.5	82.2	86.4	83.0
AiATrack [44]	ECCV'22	73.8	79.4	69.0	80.4	87.8	82.7
OSTrack [120]	ECCV'22	**77.6**	**81.1**	**71.1**	**83.2**	88.5	**83.9**
RTS [89]	ECCV'22	73.7	76.2	69.7	79.4	86.0	81.6
MixFormer [18]	CVPR'22	76.3	**79.9**	70.1	83.1	**88.9**	83.9
ToMP [82]	CVPR'22	73.5	79.2	68.5	78.9	86.4	81.5
GTELT [131]	CVPR'22	73.2	75.9	67.7	81.6	86.7	82.5
UTT [80]	CVPR'22	67.2	—	64.6	77.0	—	79.7
SBT [113]	CVPR'22	71.1	—	66.7	—	87.5	82.2
KeepTrack [83]	ICCV'21	70.2	77.2	67.1	73.8	83.5	78.1
STARK [117]	ICCV'21	72.2	77.0	67.1	—	86.9	82.0
TransT [14]	CVPR'21	69.0	73.8	64.9	80.3	86.7	81.4
TrDiMP [106]	CVPR'21	66.3	73.0	63.9	73.1	83.3	78.4
SiamRCNN [104]	CVPR'20	68.4	72.2	64.8	80.0	85.4	81.2
SuperDiMP [30]	CVPR'20	65.3	72.2	63.1	73.3	83.5	78.1

TABLE 6.4: State of the art comparison on SOT datasets.

a low AssA (29.0 vs. 69.0 of TaMOs), demonstrating that QDTrack's association mechanism is not robust enough for the challenging scenarios contained in LaGOT, see appendix for more details.

6.4.2 *State-of-the-Art Comparison on SOT Datasets*

While TaMOs is built to track multiple user-specified objects in a video it can as well track only a single generic object. Thus, we evaluate TaMOs on popular large-scale SOT benchmarks. We deploy

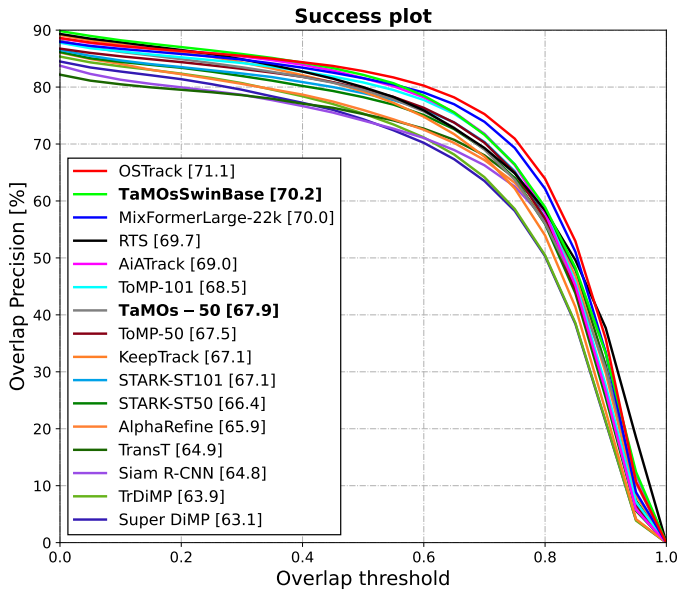


FIGURE 6.5: Success plot, showing OP_T, on LaSOT [38] (AUC is reported in the legend).

the very same tracker in these settings, without altering its weights nor any hyper-parameters.

LaSOT [38]: This large-scale dataset consists of 280 test sequences with 2500 frames on average. Tab. 6.4 shows a comparison to recent SOT trackers. While primarily designed to cope with multiple objects, our tracker achieves the highest precision and the third highest success rate AUC. In additions, Fig. 6.5 shows the corresponding success plot. We observe that our tracker is the most robust ($T < 0.3$) and that both MixFormerLarge-22k [18] and OSTrack [120] can regress more accurate bounding boxes ($0.5 < T < 0.9$). Note, that neither MixFormer, SwinTrack nor OSTrack operate on the entire video frame, but rely on a local search area to produce such high tracking accuracy. Thus, these results show the great potential of applying trackers *globally*. Furthermore, we do not employ any motion priors, such as

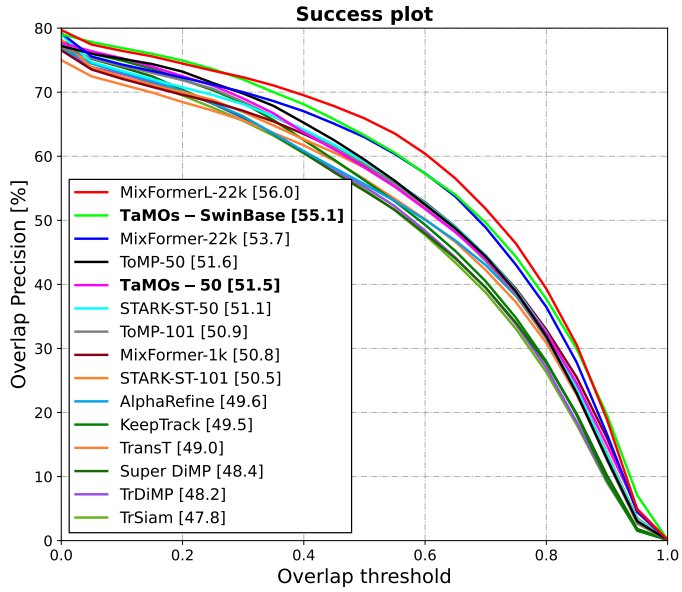


FIGURE 6.6: Success plot, showing OP_T, on AVisT [87] (AUC is reported in the legend).

search area selection [6, 18, 82, 120] or spatial windowing [67, 68], considered standard in tracking.

TrackingNet [85]: This large-scale dataset consists of 511 test sequences and an evaluation server is used to evaluate the tracking predictions. Tab. 6.4 shows that our tracker with SwinBase sets the new state of the art on TrackingNet in terms of success rate and precision AUC. Similarly, our tracker with ResNet-50 achieves the best results among all trackers using that backbone. Again, these results demonstrate the great potential of multi-object GOT for SOT.

AVisT [87]: AVisT is an evaluation benchmark that contains 120 sequences containing tracking scenarios in adverse visibility conditions. Fig. 6.6 shows that our tracker achieves excellent results with a success AUC of 55.1. This result shows that TaMOs is able to track generic single objects even in visually challenging settings. The best

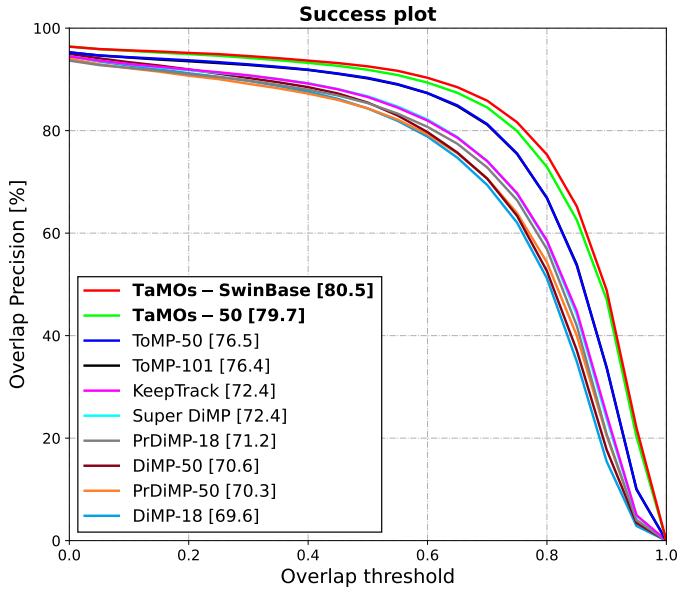


FIGURE 6.7: Success plot, showing OP_T, on ImagenetVID [92] (AUC is reported in the legend).

tracker MixFormerLarge-22k [18] is able to regress more accurate bounding boxes ($0.3 < T < 0.9$), as it relies on small search area selection to ensure high-resolution features. In contrast, our approach is capable of jointly tracking multiple objects.

6.4.3 *Comparison on Video Object Detection Datasets*

In order to validate the proposed multiple object GOT tracker not only on LaGOT but also on another multiple object dataset, we modify ImageNetVID [92].

ImageNetVID [92]: We perform the following adaptations in order to apply GOT methods on the video object detection dataset ImageNetVID. First, we remove all tracks that are not present in the first frame. Then, we use the remaining tracks to produce the bounding

Gaussian Encoding	LTRB Encoding	Object Embedding Pool size m	LaSOT AUC	LaGOT AUC	LaGOT F1
✓	✗	10	58.3	54.0	0.552
✗	✓	10	66.3	60.2	0.620
✓	✓	10	**67.2**	**61.6**	**0.633**
✓	✓	15	65.7	60.0	0.617
✓	✓	20	65.7	58.9	0.603
✓	✓	50	63.1	57.4	0.587

TABLE 6.5: Analysis of different object encoding settings. All tested configurations are not employing the FPN.

box annotations of the first frame. For simplicity we remove the 11 sequences where no track is visible in the first frame. This results in 544 sequences with 938 tracks and 1.7 tracks on average per video. Fig. 6.7 shows the success plot on the resulting GOT dataset. We observe that all trackers achieve relatively high AUC mostly differing in bounding box accuracy. Both versions of our tracker outperform the baselines ToMP-50 and ToMP-101 [82]. In particular, we notice the superior bounding box accuracy of our tracker compared to ToMP. To summarize we observe a similar ranking between trackers on ImageNetVID and the proposed LaGOT dataset. However, LaGOT is more challenging due to the higher average track number (2.9 vs. 1.7) and the much longer sequence length (2258 vs. 312) that leads more frequently to occlusions and out-of-view events.

6.4.4 *Ablation Study*

Generic Multiple Object Encoding: Tab. 6.5 shows the effect of the Gaussian score map encoding, the LTRB bounding box encoding and the total number of object embeddings m stored in the pool E. The first two rows in Tab. 6.5 show that the LTRB encoding is more important than the Gaussian encoding (as removing LTRB decreases

Backbone	FPN	Memory Update	LaSOT AUC	LaGOT AUC	LaGOT F1-Score
Resnet-50	✗	✓	67.2	60.4	0.621
Resnet-50	✓	✗	66.0	60.2	0.620
Resnet-50	✓	✓	**67.9**	**61.6**	**0.633**
SwinBase	✗	✓	69.5	62.4	0.643
SwinBase	✓	✗	67.9	62.1	0.636
SwinBase	✓	✓	**70.2**	**63.5**	**0.649**

TABLE 6.6: Architecture and memory update analysis.

all results more significantly). Another key factor is the number of different object embeddings, that sets an upper limit on the number of objects that can be tracked. LaGOT requires at least 10 embeddings and our tracker achieves the best results when using a pool size of 10. Increasing the number of embeddings decreases the overall tracking performance.

Architecture: Tab. 6.6 shows that using SwinBase increases the tracking performance on LaSOT and LaGOT. Similarly, adding an FPN improves the results. Thus, we employ an FPN and report all results for both backbones.

Inference: During inference we update the memory by adding a second dynamic training frame. Since the ground truth bounding box is not available, we use the predicted box as annotation. We replace the dynamic training frame (update the memory) if the maximal value in each target score map is above the threshold of $\tau = 0.85$. The results in Tab. 6.6 show that adding a second training frame improves the results on both LaGOT and LaSOT.

6.5 CONCLUSION

We propose a novel multiple object GOT tracking benchmark, LaGOT, that allows to evaluate GOT methods that can jointly track multiple targets in the same sequence. We demonstrate that the proposed

task and benchmark are challenging for existing SOT and MOT trackers. We further propose a Transformer-based tracker capable of processing multiple targets at the same time. Our approach integrates a novel generic multi object encoding and an FPN in order to achieve full frame tracking. Our method outperforms recent trackers on the LaGOT benchmark, while operating $4\times$ faster than the SOT baseline when tracking 10 objects. Lastly, our approach also achieves excellent results on large-scale SOT benchmarks.

6.6 APPENDICES

In the appendix, we first give an overview of the different task definitions and the corresponding abbreviations used in the paper and appendix. Next, we describe the details of the model architecture and training in Sec. 6.8. Then, we provide more insights into the experiments presented in the main paper and provide additional results on less popular tracking datasets in Sec. 6.9. Then, we show visual results between the baseline and our tracker on multiple sequences of the proposed datasets including failure cases 6.10. Finally, we provide additional insights about our dataset and compare it to datasets of related tasks in Sec. 6.11.

6.7 GLOSSARY

In this Section we will briefly summarize the different task definitions behind the individual abbreviations:

GOT: Generic Object Tracking refers to the task of tracking potentially multiple user-defined target objects of arbitrary classes specified by a user-specified bounding box in the initial video frame.

SOT: Singe Object Tracking is the same task as GOT but focuses on the setting where only a single generic object needs to be tracked.

Multi-Object GOT: The same as GOT but emphasizes that multiple-objects need to be tracked. We use multi-object GOT because GOT is in other research works sometimes used interchangeably with SOT.

MOT: Multi Object Tracking is completely different from the tasks listed above because it requires a class category list to detect and track all objects corresponding to the defined class categories.

GMOT: Generic Multi Object Tracking is the same as MOT but instead of using a class category list to define the target objects, a single user-specified box shows an example object of the target class category. Thus, all objects that belong to the same class as the user-specified example need to be detected and tracked.

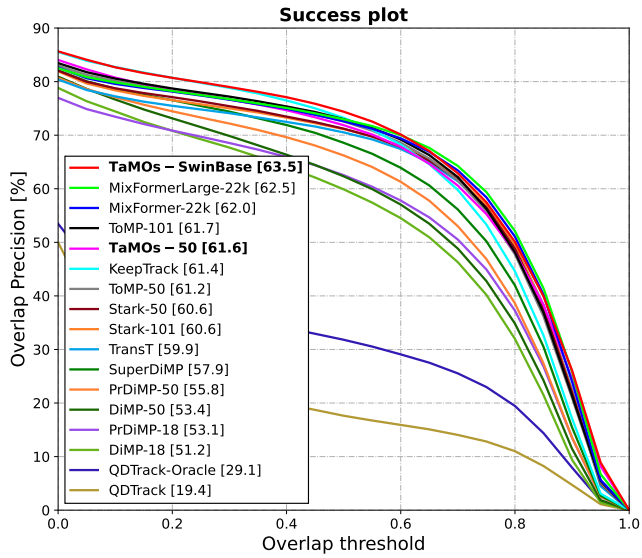
6.8 MODEL ARCHITECTURE AND TRAINING DETAILS

Architecture: We extract backbone features either from the Resnet-50 or the SwinBase backbone. For both backbones we extract the features corresponding to the blocks with stride 8 and 16. We only use the features with stride 16 for object encoding and feed these features into the model predictor. For both backbones we use a linear layer to decrease the number of channels from 1024 to 256 or 512 to 256 respectively. Thus we use 256 dimensional object embeddings e_i and a MLP to project the LTRB bounding box encoding map from 4 to 256 channels. Since the model predictor produces 256 dimensional convolutional filters we require the same number of channels for the FPN output features. In particular we use a two layer FPN that uses as input the enhanced Transformer encoder output features corresponding to the test frame as well as the aforementioned high resolution backbone test features. The high resolution input features have either 512 or 256 channels for the Resnet-50 or the SwinBase backbone respectively. Thus, we adapt the FPN accordingly depending on the used backbone.

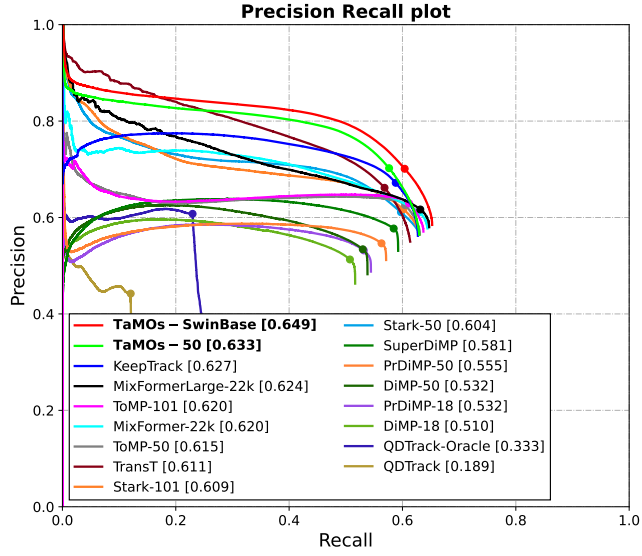
Training Details: We use a fixed size Gaussian when producing the score map encoding for each object where $\sigma = 0.25$. Furthermore, we use gradient norm clipping with the parameter 0.1 in order to stabilize training. In addition, we set the loss weighting parameters to $\lambda_{\text{cls}} = 100$ and to $\lambda_{\text{bbreg}} = 1$. We train all models on four A100 GPUs with a batch size of 4×12 or 4×6 depending on the used backbone.

6.9 EXPERIMENTS

We provide more detailed results to complement the comparison shown in the main paper. In addition we provide result for the LaSOTExt [37] dataset in order to assess the performance of our tracker on sequences containing small objects.



(a) Success Plot



(b) Precision-Recall

FIGURE 6.8: Success plot, showing OP_T, on LaGOT (AUC is reported in the legend). Tracking Precision-Recall curve on LaGOT – VOTLT is reported in the legend (the highest F1-score).

		HOTA	DetA	AssA	DetRe	DetPr	AssRe	AssPr	LocA	OWTA	MOTA	IDSW	IDF1
GOT	**TaMOs-SwinBase**	62.5	57.5	69.0	70.2	70.0	76.6	76.4	84.5	69.4	58.3	5962	75.2
	TaMOs-50	60.4	54.9	67.5	68.2	67.9	75.2	75.0	84.2	67.6	53.2	7273	72.6
SOT	MixformerLarge-22k	61.7	54.1	71.1	67.8	67.5	78.1	77.9	85.0	69.3	52.6	2788	74.636
	Mixformer-22k	61.6	54.4	70.4	67.8	67.6	77.4	77.3	84.5	69.0	53.9	2969	75.0
	ToMP-101	60.4	53.2	69.3	66.7	66.4	76.7	76.6	83.9	67.9	52.4	2297	74.3
	ToMP-50	60.0	52.6	69.0	66.2	66.0	76.6	76.5	83.8	67.5	51.5	2042	74.0
	STARK-ST-101	59.9	52.2	69.3	66.1	65.8	76.5	76.4	84.2	67.6	49.8	3212	73.2
	STARK-ST-50	59.7	52.2	68.9	65.9	65.7	76.1	76.0	84.0	67.3	49.9	3689	73.2
	TransT	58.2	50.6	67.5	64.8	64.5	75.0	74.9	84.4	66.0	47.3	2058	71.5
	KeepTrack	59.3	52.6	67.3	65.7	65.5	74.8	74.7	82.4	66.4	51.7	2045	74.0
	SuperDiMP	56.3	48.5	65.9	62.4	62.2	73.7	73.6	82.1	64.1	43.3	1748	69.8
	PrDiMP-50	53.4	45.7	63.0	59.8	59.6	71.2	71.1	81.4	61.3	38.2	2023	67.2
	PrDiMP-18	51.4	43.1	62.0	57.5	57.3	70.2	70.1	81.4	59.6	32.3	1770	63.6
	DiMP-50	51.2	42.7	62.0	56.8	56.6	69.9	69.8	80.1	59.3	30.1	1722	62.6
	DiMP-18	48.9	40.4	59.8	54.5	54.3	67.8	67.7	79.5	57.0	25.1	1647	60.1
MOT	QDTrack	22.1	17.2	29.0	46.7	20.7	30.3	79.8	81.9	36.6	-119.7	18612	16.2

TABLE 6.7: Comparison of different trackers using MOT metrics on LaGOT.

6.9.A *LaGOT*

To complement the results shown in the main paper, we report in Fig. 6.8 and Tab. 6.7 results for additional trackers and different variants, such as using a different backbone or different hyperparameters. In Tab. 6.7 we report additional MOT sub-metrics and statistics on LaGOT. In general we conclude, that using larger backbones especially if they are pretrained on Imagenet-22k leads to the best results. Furthermore, we observe that the MOT method QDTrack is not competitive with GOT methods. In particular, we observe that QDTrack achieves the lowest OWTA score that depends on the DetRe and the AssA scores. QDTrack scores the lowest DetRe among all methods since it is not readily able to track target objects of arbitrary classes. While this is an expected limitation we further observe that QDTrack achieves by far the lowest AssA caused by the poor Association Recall (AssRe) of 30.3, whereas even small GOT method DiMP-18 achieves a AssRe of 67.7.

Method	Venue	LaSOTExt [37] Prec	N-Prec	Succ
TaMOs-SwinBase		**58.0**	57.8	**49.2**
TaMOs-Resnet-50		54.1	55.0	46.7
AiATrack [44]	ECCV'22	54.7	58.8	49.0
OSTrack [120]	ECCV'22	**57.6**	**61.3**	**50.5**
ToMP-101 [82]	CVPR'22	52.6	58.1	45.9
ToMP-50 [82]	CVPR'22	51.9	57.6	45.4
GTELT [131]	CVPR'22	52.4	54.2	45.0
KeepTrack [83]	ICCV'21	54.7	**61.7**	48.2
SuperDiMP [30]	CVPR'20	49.0	56.3	43.7
LTMU [22]	CVPR'20	45.4	53.6	41.4
DiMP [6]	ICCV'19	43.2	49.6	39.2
ATOM [23]	CVPR'19	41.2	49.6	37.6

TABLE 6.8: Comparison to the state of the art on LaSOTExt [37].

Backbone	FPN	Zoom	LaSOTExt AUC	UAV123 AUC
Resnet-50	✗	✗	41.3	56.2
Resnet-50	✓	✗	43.1	58.2
Resnet-50	✓	✓	**46.7**	**64.2**
SwinBase	✗	✗	43.9	56.5
SwinBase	✓	✗	44.6	57.3
SwinBase	✓	✓	**49.2**	**66.2**

TABLE 6.9: Analysis of the FPN and the zooming mechanism on LaSOTExt [37] and UAV123 [84].

6.9.B *LaSOTExt*

Since our tracker always operates on the full frame without the help of a local search region, tracking small objects is challenging. Thus,

we integrated an FPN in our tracker to improve the tracking accuracy. To analyze our tracker on small objects we run it on LaSOTExt [37] and UAV123 [84]. Tab. 6.9 shows that including an FPN improves the tracking results on both datasets but is more effective when using a Resnet-50 as backbone.

To track small objects a high feature map resolution is desirable. To better cope with extremely small objects, found in some SOT benchmarks, we add a simple zooming mechanism. In particular, when the target is smaller that 30×30 pixels, we crop a region of the image that ensures this minimal target size when up-scaled to the input-resolution of 384×576. Tab. 6.9 clearly shows that using such a zooming mechanism improves the results on LaSOTExt and UAV123 considerably, due to the presence of extremely small objects in these datasets.

Tab. 6.8 shows that our tracker with FPN and zooming achieves competitive results on LaSOTExt. In particular it achieves the highest precision and the second highest success AUC only being outperformed by OSTrack [120].

6.10 VISUAL RESULTS

Visual Comparison to the State of the Art: We show visualizations of the tracking results of the baseline (ToMP-101) and our proposed tracker (TaMOs-SwinBase) on four different sequences of the proposed LaGOT benchmark in Fig. 6.9. The first frame specifies the target objects annotated with bounding boxes that should be tracked in the video. The other frames show predictions of both trackers. The results on the first and third sequences demonstrate that our tracker can re-detect occluded objects quickly whereas a search area based tracker is not able to re-detect the targets if they reappear outside of the search area. The second and fourth sequences show the superior robustness of our tracker. It is able to distinguish similarly looking objects better without confusing their ids. For more visual results we refer the reader to the mp4-videos submitted alongside this doc-

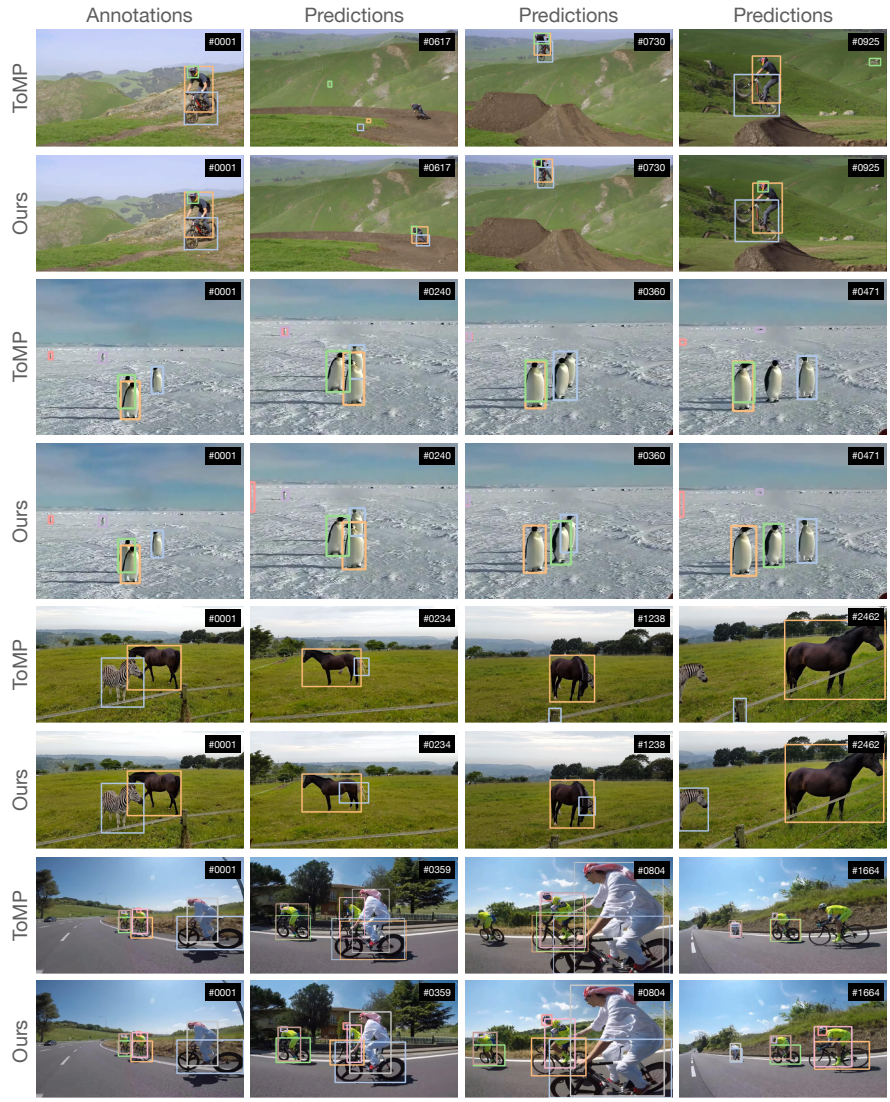


FIGURE 6.9: Visual comparison between the proposed tracker (Ours-SwinBase) and the baseline ToMP-101 on different LaGOT sequences.

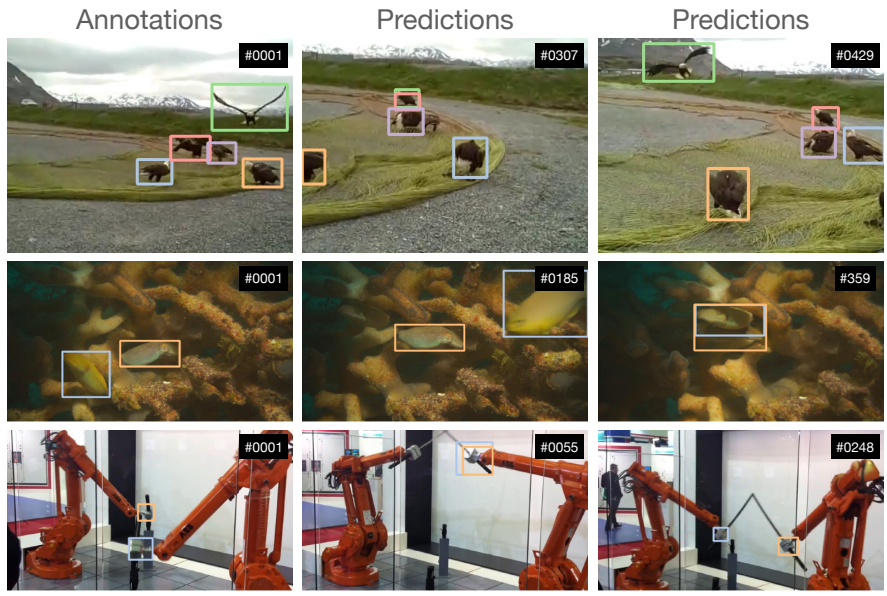


FIGURE 6.10: Visual examples of failure cases of the proposed tracker (Ours-SwinBase) on different LaGOT sequences.

ument. Each video shows the predictions of the proposed tracker TaMOs-SwinBase on the proposed LaGOT benchmark. Please note that we always produce a bounding box for visualization independent of its confidence score.

Failure Cases: Fig. 6.10 shows typical failure cases of the proposed tracker on three different sequences of the proposed LaGOT benchmark. Particularly challenging are videos that contain multiple visually similar objects since our tracker does not employ any motion model but rather tracks the objects via the learned appearance from the first frame. Another failure case occurs when the target object is no longer visible such that our tracker might start to track a visually similar distractor instead. However, once the target reappears our globally operating tracker is usually able to re-detect it. Lastly, if multiple visually similar objects need to be tracked our tracker might fail

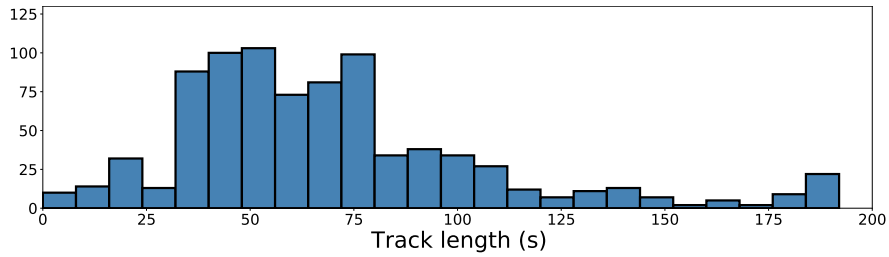


FIGURE 6.11: Track lengths distribution of the LaGOT benchmark.

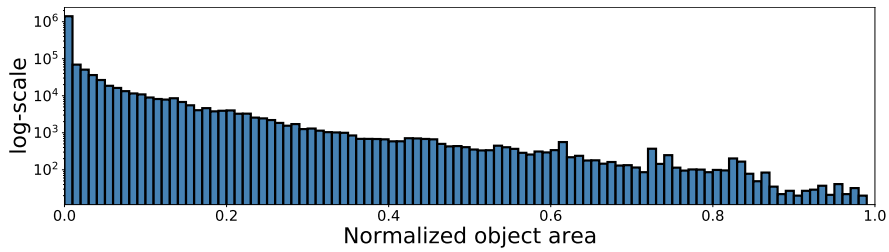


FIGURE 6.12: Object size distribution of the LaGOT benchmark.

to distinguish these objects such that it produces multiple bounding boxes with different ids for the same object.

6.11 DATASETS

Below we provide additional details about our annotated dataset, such as examples of new classes and various statistics, as well as an extensive comparison to existing datasets that focus on related tasks.

6.11.A *Insights*

Fig. 6.11 shows the distribution of the track lengths in seconds for all tracks in the proposed benchmark LaGOT. We observe that most tracks are between 30 and 110 seconds long. Furthermore, Fig. 6.12 shows the size distribution of the annotated objects in the dataset. We conclude that various sizes are present in the dataset but large

Dataset	Num Classes	Num Videos	Avg Video length (num frames)	Avg Tracks per Video	Avg Track Length (num boxes)	Avg Track Length (s)	Avg Instance per frame	Video FPS	Annotation FPS
YouTubeVOS [114]	91	474	135	1.74	27	4.5	1.64	30 FPS	6 FPS
Davis17 [90]	-	30	67	1.97	67	2.8	1.97	24 FPS	24 FPS
ImageNetVID* [33]	30	555	317	2.35	208	7	1.58	30 FPS	30 FPS
TAO* [31]	302	988	1010	5.55	21	21	3.31	30 FPS	1 FPS
BDD100k [122]	11	200	198	94.21	26	5	11.8	30 FPS	5 FPS
MOT15 [32]*	1	11	500	45.5	75	3	8	2.5-30 FPS	2.5-30 FPS
MOT16 [32]*	1	7	760	74	273	10	38	14-30 FPS	14-30 FPS
MOT20 [32]*	1	4	2233	583	572	23	150	25 FPS	25 FPS
GMOT-40 [3]	10	40	240	50.65	133	5.3	26.6	24-30 FPS	24-30 FPS
TrackingNet [85]	27	511	442	1	442	15	1	30 FPS	30 FPS
UAV123 [84]	8	123	915	1	915	28	1	30 FPS	30 FPS
OTB-100 [109]	16	100	590	1	590	20	1	30 FPS	30 FPS
NFS-30 [42]	15	100	479	1	479	14	1	30 FPS	30 FPS
GOT10k [54]	84	420	150	1	150	15	1	10 FPS	10 FPS
OxUvA [102]	8	200	4198	1	60	140	1	30 FPS	1 FPS
LaSOT [37]	71	280	2430	1	2430	81	1	30 FPS	30 FPS
LaGOT	102	291	2258	2.88	702	70	2.36	30 FPS	10 FPS

TABLE 6.10: Comparison of LaGOT and the existing datasets. Statistics is provided for test or validation set for the datasets for which test set annotations are hidden. * For MOT15-20 we report stats on the train set.

objects are rare than small ones. Further, the distribution shows that the targets are not visible in a large amount of video frames indicated by an object area of zero.

During the annotation process, we added 31 new classes: *rotor*, *fish*, *backpack*, *motor*, *wheel*, *garbage*, *drum*, *accordion*, *super-mario*, *hockey puck*, *hockey stick*, *kite-tail*, *ball*, *crown*, *stick*, *spiderweb*, *head*, *banner*,*face*, *bench*, *tissue-bag*, *para glider*, *star-patch*, *shadow*, *bucket*, *helicopter*, *sonic*, *hero*, *ninja-turtle*, *reflection*, *rider*.

6.11.B *Comparison*

We provide a detailed comparison of related existing datasets in Tab. 6.10. We divide the table into Video Object Segmentation (VOS), Video Object Detection, Multiple Object Tracking (MOT), Generic

Multiple Object Tracking (GMOT) and Single Object Tracking (SOT) datasets.

The length of VOS sequences is much shorter than in our LaGOT benchmark (2.8s/4.5s vs 70s). Similarly the video object detection dataset ImagenetVID contains shorter sequences (7s vs. 70s), fewer classes (30 vs 102) and a smaller number of average tracks per sequence (2.35 vs 2.88) than LaGOT. MOT datasets typically focus on fewer classes, contain shorter sequences or are annotated at low frame rates only. TAO contains many more classes than typical MOT datasets but provides annotations only at 1 FPS leading to a much lower average number of annotated frames per track than LaGOT (21 vs. 702). The GMOT-40 dataset contains fewer classes, fewer videos, shorter sequences and provides due to its task only annotations of one particular object class per sequence compared to LaGOT. In contrast to SOT datasets that provide only a single annotated object per sequence, LaGOT provides on average 2.88 tracks per sequence. Furthermore, it contains longer sequences than most listed SOT datasets. Overall LaGOT enables to properly evaluate the robustness and accuracy of multiple object GOT methods. A key factor are the multiple annotated tracks per sequence at a high frame rate and the relatively long sequences.

7

CONCLUSION

In this thesis, we tackled the problem of generic visual object tracking in videos. Next, we will briefly summarize our contributions, discuss limitations and possible directions for future work.

7.1 SUMMARY OF CONTRIBUTIONS

In Chapter 3, we proposed a novel tracking pipeline employing a learned target candidate association network in order to track both the target and distractor objects. This approach allows us to propagate the identities of all target candidates throughout the sequence. In addition, we propose a training strategy that combines partial annotations with self-supervision. We do so to compensate for lacking correspondence annotations between distractor objects in visual tracking. We showed that the resulting tracker KeepTrack outperformed the state of the art at that time. KeepTrack achieved by far the most robust tracking results.

In Chapter 4, we introduced ToMP a novel tracker employing a Transformer-based model predictor. The model predictor estimates the weights of the DCF-based target model to localize the target and a second set of weights for precise bounding box regression. To achieve this, we developed a new module that produce a target-specific positional encoding that is combined with the backbone features. We showed that our tracker ToMP is more robust and regresses more accurate bounding boxes than the baseline method SuperDiMP.

Beside the new tracking methods proposed in the previous cßhapters, we introduced a new benchmark in Chapter 5, that focuses on visual tracking in diverse scenarios with adverse visibility. The benchmark AVisT covers 18 diverse scenarios with 42 classes. These diverse scenarios are further grouped into five attributes. We evaluated a

variety of recent Siamese, discriminative classifiers and Transformer-based trackers on AVisT. Our experiments showed that even the most recent Transformer-based tracker achieves a relatively low score, thereby highlighting the challenging nature of AVisT.

Finally, in Chapter 6, we proposed a multiple object GOT benchmark, that allows to evaluate GOT methods that can jointly track multiple targets in the same video. In addition, we detailed a Transformer-based tracker that is capable of processing multiple target objects at the same time. Our approach integrates a novel generic multi object encoding and an FPN in order to achieve full frame tracking. Our method outperforms recent trackers on the proposed LaGOT benchmark, while operating 4× faster than the SOT baseline when tracking 10 objects. Lastly, we demonstrated, that our approach also achieves excellent results on large-scale SOT benchmarks.

7.2 DISCUSSION, LIMITATIONS AND FUTURE WORK

7.2.1 *KeepTrack*

KeepTrack uses the score map of a base tracker to extract target candidates. Then, it matches these candidates with the target object and the distractors of the previous frame in order to identify the target in the current frame. Thus, the main limitations are the small time horizon and the selected features used for matching.

Matching across Multiple Frames: KeepTrack matches target candidates only between two frames. However, matching candidates over multiple past frames would be beneficial, especially in uncertain scenarios. Furthermore, keeping all possible assignment probabilities between target candidates of adjacent frames, would allow to form a graph over the whole video sequence. Then, a dynamic programming algorithm could be applied to find the best fitting target candidate by taking the entire history into account [104].

Candidate Features: Currently, the visual features used for matching correspond to the features queried at the predicted center location

of each target candidate. However, having access to a bounding box for each target candidate could allow to extract more discriminative visual features, for example by using Region of Interest (ROI) pooling. Furthermore, the bounding box itself could be used as a feature for matching.

7.2.2 *ToMP*

ToMP replaces the optimization based model predictor of DCF-based trackers with a Transformer. Despite several advantages, doing so results in the following limitations.

Memory Consumption: The Transformer encoder consist of self-attention layers that computes similarity matrices between multiple training and test frame features. This leads to a large memory footprint that impacts training and inference run-time. Thus, in future work this limitation should be addressed by evaluating alternatives such as [56, 59, 94] aiming at decreasing the memory burden.

Distractors: Another limiting factor of ToMP arises from challenging tracking sequences. In particular, distractors that are present while the target is occluded, is a typical failure scenario of ToMP, since it is lacking explicit distractor handling as in KeepTrack.

7.2.3 *AVisT*

AVisT focuses on adverse visibility scenarios with the goal of examining visual trackers on diverse scenes. Even though, AVisT puts a special focus on challenging and diverse scenarios, it has the following limitations.

Domain Diversity: AVisT mainly focuses on tracking objects in adverse visibility of natural scenes. To further increase the richness of GOT benchmarks, sequences from other domains could be added such as bio-medical videos captured via microscopes or other devices, corrupted sequences from old black and white or early color movies,

artificially generated videos from video games [38] or blockbuster movies.

Rare Object Classes: AVisT contains objects of 42 different classes that can be summarized as pedestrians, animals and vehicles. In order to better benchmark the capability of existing trackers to track objects of unknown class categories, sequences with rarer class categories could be added. Possible examples are microorganisms, spaceships, or fictional creatures [1, 38, 54].

7.2.4 *LaGOT*

In contrast to SOT datasets, LaGOT provides annotations for multiple objects in the same video sequence. The video sequences are borrowed from LaSOT [38] but new annotations were added. Thus, LaGOT has the following limitations.

Initial Annotations: Following the setup in existing GOT and Video Object Segmentation (VOS) benchmarks, LaGOT contains a single initial frame where all target objects are specified. However, the multi-object GOT task could be further generalized by allowing that target objects can appear at any time. Thus, whenever they appear, they should be tracked along with the previously specified targets.

Video Diversity: Currently, all the LaGOT sequences are borrowed from the evaluation set of LaSOT. Thus, the multi-object GOT benchmark could be further improved by adding new sequences or by collecting videos from other datasets with complementary characteristics.

Training Data: LaGOT is an evaluation benchmark and therefore does not provided any training data. Instead, LaGOT requires that the multi-object GOT tracking capabilities are trained by using datasets that focus on different tasks. Particularly useful are datasets where multi-object annotations are available (MOT, VOS, VIS). Thus, providing a training dataset could further improve the interest in this task and improve the performance of resulting multi-object GOT trackers.

7.2.5 *TaMOs*

TaMOs allows to jointly track multiple generic objects specified by user-provided bounding boxes in the initial frame of the video sequence. The introduced multi object encoding module and the modified Transformer-based model predictor allow to jointly produce multiple target models. Since TaMOs is the first method that can track multiple generic targets in the same video it has a few limitations.

Global Reasoning: While the Transformer-based model predictor jointly produces all target models, TaMOs avoids any explicit matching or global reasoning over all target object tracklets. Hence, future work could try to overcome this limitation by developing a more explicit matching approach, that allows to match different target objects over time. The matching methods could be similar to KeepTrack or the association methods employed in MOT trackers [4, 11, 88].

Tracking newly appearing Objects: TaMOs is able to track multiple generic objects at the same time but requires that all objects are annotated and visible in the initial frame. Thus, an extension of TaMOs could allow to integrate and track objects appearing at arbitrary times. This would not only allow to track new target objects on the fly but would also enable to track distractor objects. Together with the aforementioned global reasoning, this approach has the potential to increase the robustness of the tracker against distractors.

Number of Targets: Currently, the number of target objects, that can be tracked, is limited by the pool-size of the object embeddings. While it is possible to learn a larger pool-size, it is cumbersome. Thus, an interesting direction for future research would be to generate an arbitrary number of object embeddings on the fly, such that any number of target objects can be tracked.

Tracking Small Objects: Furthermore, we proposed to use an FPN to regress more accurate bounding boxes for small objects and show that adding such an FPN helps. However, as in object detection, tracking extremely small objects is challenging due to the limited feature resolution when processing the full frame.

BIBLIOGRAPHY

1. Alawode, B., Guo, Y., Ummar, M., Werghi, N., Dias, J., Mian, A. & Javed, S. *UTB180: A High-quality Benchmark for Underwater Tracking* in *Proceedings of the Asian Conference on Computer Vision (ACCV)* (2022), 3326.
2. Andriluka, M., Pishchulin, L., Gehler, P. & Schiele, B. *2D Human Pose Estimation: New Benchmark and State of the Art Analysis* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014).
3. Bai, H., Cheng, W., Chu, P., Liu, J., Zhang, K. & Ling, H. *GMOT-40: A Benchmark for Generic Multiple Object Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), 6719.
4. Bergmann, P., Meinhardt, T. & Leal-Taixe, L. *Tracking Without Bells and Whistles* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2019).
5. Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A. & Torr, P. H. *Fully-Convolutional Siamese Networks for Object Tracking* in *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)* (2016).
6. Bhat, G., Danelljan, M., Gool, L. V. & Timofte, R. *Learning Discriminative Model Prediction for Tracking* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2019).
7. Bhat, G., Danelljan, M., Van Gool, L. & Timofte, R. *Know Your Surroundings: Exploiting Scene Information for Object Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2020).

8. Bhat, G., Johnander, J., Danelljan, M., Khan, F. S. & Felsberg, M. *Unveiling the Power of Deep Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2018).

9. Bhat, G., Lawin, F. J., Danelljan, M., Robinson, A., Felsberg, M., Gool, L. V. & Timofte, R. *Learning What to Learn for Video Object Segmentation* in *Proceedings of the European Conference on Computer Vision ECCV* (2020).

10. Bolme, D. S., Beveridge, J. R., Draper, B. A. & Lui, Y. M. *Visual object tracking using adaptive correlation filters* in *CVPR* (2010).

11. Braso, G. & Leal-Taixe, L. *Learning a Neural Solver for Multiple Object Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

12. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A. & Zagoruyko, S. *End-to-end object detection with transformers* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2020), 213.

13. Chang, M.-F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D. & Hays, J. *Argoverse: 3D Tracking and Forecasting With Rich Maps* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

14. Chen, X., Yan, B., Zhu, J., Wang, D., Yang, X. & Lu, H. *Transformer Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021).

15. Chen, Y., Xu, J., Yu, J., Wang, Q., Yoo, B. & Han, J. J. *AFOD: Adaptive Focused Discriminative Segmentation Tracker* in *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)* (2020).

16. Chen, Z., Zhong, B., Li, G., Zhang, S. & Ji, R. *Siamese Box Adaptive Network for Visual Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

17. Choi, J., Kwon, J. & Lee, K. M. *Visual Tracking by TridentAlign and Context Embedding* in *Proceedings of the Asian Conference on Computer Vision (ACCV)* (2020).

18. Cui, Y., Jiang, C., Wang, L. & Wu, G. *MixFormer: End-to-End Tracking With Iterative Mixed Attention* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 13608.

19. Cummins, M. & Newman, P. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research* **27**, 647 (2008).

20. Cuturi, M. *Sinkhorn Distances: Lightspeed Computation of Optimal Transport* in *Advances in Neural Information Processing Systems (NeurIPS)* (2013).

21. Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T. & Niessner, M. *ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

22. Dai, K., Zhang, Y., Wang, D., Li, J., Lu, H. & Yang, X. *High-Performance Long-Term Tracking With Meta-Updater* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

23. Danelljan, M., Bhat, G., Khan, F. S. & Felsberg, M. *ATOM: Accurate Tracking by Overlap Maximization* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

24. Danelljan, M., Bhat, G. & Mayer, C. *PyTracking: Visual tracking library based on PyTorch.* https://github.com/visionml/pytracking. Accessed: 1/07/2022. 2019.

25. Danelljan, M., Bhat, G., Shahbaz Khan, F. & Felsberg, M. *ECO: Efficient Convolution Operators for Tracking* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

26. Danelljan, M., Häger, G., Khan, F. S. & Felsberg, M. *Learning Spatially Regularized Correlation Filters for Visual Tracking* in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (2015).

27. Danelljan, M., Häger, G., Shahbaz Khan, F. & Felsberg, M. *Accurate Scale Estimation for Robust Visual Tracking* in *Proceedings of the British Machine Vision Conference* (BMVA Press, 2014).

28. Danelljan, M., Häger, G., Shahbaz Khan, F. & Felsberg, M. *Adaptive Decontamination of the Training Set: A Unified Formulation for Discriminative Visual Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

29. Danelljan, M., Robinson, A., Shahbaz Khan, F. & Felsberg, M. *Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2016).

30. Danelljan, M., Van Gool, L. & Timofte, R. *Probabilistic Regression for Visual Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

31. Dave, A., Khurana, T., Tokmakov, P., Schmid, C. & Ramanan, D. *TAO: A Large-Scale Benchmark for Tracking Any Object* in *Proceedings of the European Conference on Computer Vision (ECCV)* (Springer International Publishing, 2020), 436.

32. Dendorfer, P., Osep, A., Milan, A., Schindler, K., Cremers, D., Reid, I., Roth, S. & Leal-Taixé, L. MOTChallenge: A Benchmark for Single-camera Multiple Target Tracking. *International Journal of Computer Vision (IJCV)* **129**, 1 (2020).

33. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. & Fei-Fei, L. *ImageNet: A large-scale hierarchical image database* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2009).

34. DeTone, D., Malisiewicz, T. & Rabinovich, A. *SuperPoint: Self-Supervised Interest Point Detection and Description* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2018).

35. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J. & Houlsby, N. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale* in *International Conference on Learning Representations* (2021).

36. Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A. & Sattler, T. *D2-Net: A Trainable CNN for Joint Description and Detection of Local Features* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

37. Fan, H., Bai, H., Lin, L., Yang, F., Chu, P., Deng, G., Yu, S., Huang, M., Liu, J., Xu, Y., *et al.* Lasot: A high-quality large-scale single object tracking benchmark. *International Journal of Computer Vision (IJCV)* **129**, 439 (2021).

38. Fan, H., Lin, L., Yang, F., Chu, P., Deng, G., Yu, S., Bai, H., Xu, Y., Liao, C. & Ling, H. *LaSOT: A High-Quality Benchmark for Large-Scale Single Object Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

39. Fan, H. & Ling, H. CRACT: Cascaded Regression-Align-Classification for Robust Visual Tracking. *arXiv preprint arXiv:2011.12483* (2020).

40. Ferryman, J. & Shahrokni, A. *PETS2009: Dataset and challenge* in *2009 Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance* (2009), 1.

41. Fu, Z., Liu, Q., Fu, Z. & Wang, Y. *STMTrack: Template-Free Visual Tracking With Space-Time Memory Networks* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021).

42. Galoogahi, H. K., Fagg, A., Huang, C., Ramanan, D. & Lucey, S. *Need for Speed: A Benchmark for Higher Frame Rate Object Tracking* in *ICCV* (2017).

43. Galoogahi, H. K., Fagg, A. & Lucey, S. *Learning Background-Aware Correlation Filters for Visual Tracking* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2017).

44. Gao, S., Zhou, C., Ma, C., Wang, X. & Yuan, J. *AiATrack: Attention in Attention for Transformer Visual Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2022), 146.

45. Geiger, A., Lenz, P., Stiller, C. & Urtasun, R. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research* **32**, 1231 (2013).

46. Guo, D., Shao, Y., Cui, Y., Wang, Z., Zhang, L. & Shen, C. *Graph Attention Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021).

47. Guo, D., Wang, J., Cui, Y., Wang, Z. & Chen, S. *SiamCAR: Siamese Fully Convolutional Classification and Regression for Visual Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

48. Gupta, A., Dollar, P. & Girshick, R. *LVIS: A Dataset for Large Vocabulary Instance Segmentation* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

49. Gustafsson, F. K., Danelljan, M., Timofte, R. & Schön, T. B. *How to Train Your Energy-Based Model for Regression* in *Proceedings of the British Machine Vision Conference (BMVC)* (2020).

50. He, K., Zhang, X., Ren, S. & Sun, J. *Deep residual learning for image recognition* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

51. Henriques, J., Caseiro, R., Martins, P. & Batista, J. *Exploiting the Circulant Structure of Tracking-by-Detection with Kernels* in *Computer Vision – ECCV 2012* (2012), 702.

52. Henriques, J. F., Caseiro, R., Martins, P. & Batista, J. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **37**, 583 (2015).

53. Huang, L., Zhao, X. & Huang, K. *Globaltrack: A simple and strong baseline for long-term tracking* in *Proceedings of the Conference on Artificial Intelligence (AAAI)* (2020).

54. Huang, L., Zhao, X. & Huang, K. GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **43**, 1562 (2021).

55. Kalal, Z., Mikolajczyk, K. & Matas, J. Tracking-Learning-Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **34**, 1409 (2012).

56. Katharopoulos, A., Vyas, A., Pappas, N. & Fleuret, F. *Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention* in *Proceedings of the International Conference on Machine Learning (ICML)* (2020), 5156.

57. Kenan, D., Dong, W., Huchuan, L., Chong, S. & Jianhua, L. *Visual Tracking via Adaptive Spatially-Regularized Correlation Filters* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

58. Kingma, D. P. & Ba, J. *Adam: A Method for Stochastic Optimization* in *Proceedings of the International Conference on Learning Representations (ICLR)* (2014).

59. Kitaev, N., Kaiser, L. & Levskaya, A. *Reformer: The Efficient Transformer* in *Proceedings of the International Conference on Learning Representations (ICLR)* (2020).

60. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pfugfelder, R., Zajc, L. C., Vojir, T., Bhat, G., Lukezic, A., Eldesokey, A., Fernandez, G. & et al. *The sixth Visual Object Tracking VOT2018 challenge results* in *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)* (2018).

61. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Kämäräinen, J.-K., Danelljan, M., Zajc, L. Č., Lukežič, A., Drbohlav, O., He, L., Zhang, Y., Yan, S., Yang, J., Fernández, G. & et al. *The Eighth Visual Object Tracking VOT2020 Challenge Results* in *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)* (2020).

62. Kristan, M., Matas, J., Leonardis, A., Vojir, T., Pflugfelder, R., Fernandez, G., Nebehay, G., Porikli, F. & Čehovin, L. A Novel Performance Evaluation Methodology for Single-Target Trackers. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **38**, 2137 (2016).

63. Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Pflugfelder, R., Kämäräinen, J.-K., Cehovin Zajc, L., Drbohlav, O., Lukezic, A., Berg, A., Eldesokey, A., Käpylä, J., Fernández, G. & et al. *The Seventh Visual Object Tracking VOT2019 Challenge Results* in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (2019).

64. Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Pflugfelder, R., Kämäräinen, J.-K., Chang, H. J., Danelljan, M., Cehovin, L., Lukežič, A., Drbohlav, O., Käpylä, J., Häger, G., Yan, S., Yang, J., Zhang, Z., Fernández, G. & et al. *The Ninth Visual Object Tracking VOT2021 Challenge Results* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops* (2021), 2711.

65. Kuznetsova, A., Talati, A., Luo, Y., Simmons, K. & Ferrari, V. *Efficient video annotation with visual interpolation and frame selection guidance* in *IEEE Winter Conference on Applications of*

Computer Vision, WACV 2021, Waikoloa, HI, USA, January 3-8, 2021 (2021).

66. Lamdouar, H., Yang, C., Xie, W. & Zisserman, A. *Betrayed by Motion: Camouflaged Object Discovery via Motion Segmentation* in *Proceedings of the Asian Conference on Computer Vision (ACCV)* (2020).

67. Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J. & Yan, J. *SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

68. Li, B., Yan, J., Wu, W., Zhu, Z. & Hu, X. *High Performance Visual Tracking With Siamese Region Proposal Network* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

69. Li, Z. & Snavely, N. *MegaDepth: Learning Single-View Depth Prediction From Internet Photos* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

70. Liang, P., Blasch, E. & Ling, H. Encoding Color Information for Visual Tracking: Algorithms and Benchmark. *IEEE Transactions on Image Processing* **24**, 5630 (2015).

71. Liao, B., Wang, C., Wang, Y., Wang, Y. & Yin, J. *PG-Net: Pixel to Global Matching Network for Visual Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2020).

72. Lin, L., Fan, H., Zhang, Z., Xu, Y. & Ling, H. *SwinTrack: A Simple and Strong Baseline for Transformer Tracking* in *Advances in Neural Information Processing Systems* (eds Oh, A. H., Agarwal, A., Belgrave, D. & Cho, K.) (2022).

73. Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P. & Zitnick, C. L. *Microsoft COCO: Common Objects in Context* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2014).

74. Liu, Y., Zulfikar, I. E., Luiten, J., Dave, A., Ramanan, D., Leibe, B., Ošep, A. & Leal-Taixé, L. *Opening Up Open World Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 19045.

75. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S. & Guo, B. *Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 10012.

76. Loshchilov, I. & Hutter, F. *Decoupled Weight Decay Regularization* in *Proceedings of the International Conference on Learning Representations (ICLR)* (2019).

77. Luiten, J., Osep, A., Dendorfer, P., Torr, P. H. S., Geiger, A., Leal-Taixé, L. & Leibe, B. HOTA: A Higher Order Metric for Evaluating Multi-Object Tracking. *International Journal of Computer Vision (IJCV)* **129**, 548 (2021).

78. Lukezic, A., Vojir, T., Zajc, L. C., Matas, J. & Kristan, M. Discriminative Correlation Filter Tracker with Channel and Spatial Reliability. *International Journal of Computer Vision (IJCV)* **126**, 671 (2018).

79. Lukežič, A., Zajc, L. Č., Vojíř, T., Matas, J. & Kristan, M. *Now you see me: evaluating performance in long-term visual tracking* 2018.

80. Ma, F., Shou, M. Z., Zhu, L., Fan, H., Xu, Y., Yang, Y. & Yan, Z. *Unified Transformer Tracker for Object Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 8781.

81. Ma, Z., Wang, L., Zhang, H., Lu, W. & Yin, J. *RPT: Learning Point Set Representation for Siamese Visual Tracking* in *Proceedings of the European Conference on Computer Vision Workshops (ECCVW)* (2020).

82. Mayer, C., Danelljan, M., Bhat, G., Paul, M., Paudel, D. P., Yu, F. & Van Gool, L. *Transforming Model Prediction for Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 8731.

83. Mayer, C., Danelljan, M., Paudel, D. P. & Van Gool, L. *Learning Target Candidate Association To Keep Track of What Not To Track* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 13444.

84. Mueller, M., Smith, N. & Ghanem, B. *A Benchmark and Simulator for UAV Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2016).

85. Müller, M., Bibi, A., Giancola, S., Al-Subaihi, S. & Ghanem, B. *TrackingNet: A Large-Scale Dataset and Benchmark for Object Tracking in the Wild* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2018).

86. Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S. & Fitzgibbon, A. *KinectFusion: Real-time dense surface mapping and tracking* in *2011 10th IEEE International Symposium on Mixed and Augmented Reality* (2011), 127.

87. Noman, M., Ghallabi, W. H. A., Kareem, D., Mayer, C., Dudhane, A., Danelljan, M., Cholakkal, H., Khan, S., Gool, L. V. & Khan, F. S. *AVisT: A Benchmark for Visual Object Tracking in Adverse Visibility* in *33rd British Machine Vision Conference BMVC* (2022).

88. Pang, J., Qiu, L., Li, X., Chen, H., Li, Q., Darrell, T. & Yu, F. *Quasi-Dense Similarity Learning for Multiple Object Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), 164.

89. Paul, M., Danelljan, M., Mayer, C. & Van Gool, L. *Robust Visual Tracking by Segmentation* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2022), 571.

90. Pont-Tuset, J., Perazzi, F., Caelles, S., Arbelaez, P., Sorkine-Hornung, A. & Gool, L. V. The 2017 DAVIS Challenge on Video Object Segmentation. *CoRR* **abs/1704.00675** (2017).

91. Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I. & Savarese, S. *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 (CVPR)* (2019).

92. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C. & Fei-Fei, L. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)* **115**, 211 (2015).

93. Sarlin, P.-E., DeTone, D., Malisiewicz, T. & Rabinovich, A. *SuperGlue: Learning Feature Matching With Graph Neural Networks* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

94. Shen, Z., Zhang, M., Zhao, H., Yi, S. & Li, H. *Efficient Attention: Attention With Linear Complexities* in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)* (2021), 3531.

95. Siegwart, R., Nourbakhsh, I. R. & Scaramuzza, D. *Introduction to autonomous mobile robots* (MIT press, 2011).

96. Sinkhorn, R. & Knopp, P. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics* **21**, 343 (1967).

97. Sun, C., Wang, D., Lu, H. & Yang, M. *Correlation Tracking via Joint Discrimination and Reliability Learning* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2018).

98. Tang, S., Andriluka, M., Andres, B. & Schiele, B. *Multiple People Tracking by Lifted Multicut and Person Re-Identification* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

99. Tian, Z., Shen, C., Chen, H. & He, T. *FCOS: Fully Convolutional One-Stage Object Detection* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2019).

100. Tuia, D., Kellenberger, B., Beery, S., Costelloe, B. R., Zuffi, S., Risse, B., Mathis, A., Mathis, M. W., van Langevelde, F., Burghardt, T., *et al.* Perspectives in machine learning for wildlife conservation. *Nature communications* **13**, 792 (2022).

101. Valmadre, J., Bertinetto, L., Henriques, J., Vedaldi, A. & Torr, P. H. S. *End-To-End Representation Learning for Correlation Filter Based Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2017).

102. Valmadre, J., Bertinetto, L., Henriques, J. F., Tao, R., Vedaldi, A., Smeulders, A. W., Torr, P. H. & Gavves, E. *Long-term Tracking in the Wild: a Benchmark* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2018).

103. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł. & Polosukhin, I. *Attention is All you Need* in *Advances in Neural Information Processing Systems (NeurIPS)* (2017).

104. Voigtlaender, P., Luiten, J., Torr, P. H. & Leibe, B. *Siam R-CNN: Visual Tracking by Re-Detection* in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

105. Wang, G., Luo, C., Sun, X., Xiong, Z. & Zeng, W. *Tracking by Instance Detection: A Meta-Learning Approach* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

106. Wang, N., Zhou, W., Wang, J. & Li, H. *Transformer Meets Tracker: Exploiting Temporal Context for Robust Visual Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021).

107. Wang, Q., Zhang, L., Bertinetto, L., Hu, W. & Torr, P. H. *Fast Online Object Tracking and Segmentation: A Unifying Approach* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019).

108. Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L. & Zhang, L. *CvT: Introducing Convolutions to Vision Transformers* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 22.

109. Wu, Y., Lim, J. & Yang, M.-H. Object Tracking Benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* **37**, 1834 (2015).

110. Wu, Y., Lim, J. & Yang, M.-H. *Online Object Tracking: A Benchmark* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2013).

111. Xiao, B., Wu, H. & Wei, Y. *Simple Baselines for Human Pose Estimation and Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2018).

112. Xiao, J., Qiao, L., Stolkin, R. & Leonardis, A. *Distractor-Supported Single Target Tracking in Extremely Cluttered Scenes* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2016).

113. Xie, F., Wang, C., Wang, G., Cao, Y., Yang, W. & Zeng, W. *Correlation-Aware Deep Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 8751.

114. Xu, N., Yang, L., Fan, Y., Yue, D., Liang, Y., Yang, J. & Huang, T. *YouTube-VOS: A Large-Scale Video Object Segmentation Benchmark* 2018.

115. Xu, Y., Wang, Z., Li, Z., Yuan, Y. & Yu, G. *Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines* in *Proceedings of the Conference on Artificial Intelligence (AAAI)* (2020).

116. Yan, B., Jiang, Y., Sun, P., Wang, D., Yuan, Z., Luo, P. & Lu, H. *Towards Grand Unification of Object Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (Springer International Publishing, 2022), 733.

117. Yan, B., Peng, H., Fu, J., Wang, D. & Lu, H. *Learning Spatio-Temporal Transformer for Visual Tracking* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 10448.

118. Yan, B., Zhang, X., Wang, D., Lu, H. & Yang, X. *Alpha-Refine: Boosting Tracking Performance by Precise Bounding Box Estimation* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021).

119. Yan, B., Zhao, H., Wang, D., Lu, H. & Yang, X. *'Skimming-Perusal' Tracking: A Framework for Real-Time and Robust Long-Term Tracking* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2019).

120. Ye, B., Chang, H., Ma, B., Shan, S. & Chen, X. *Joint Feature Learning and Relation Modeling for Tracking: A One-Stream Framework* in *Proceedings of the European Conference on Computer Vision (ECCV)* (Springer Nature Switzerland, 2022), 341.

121. Yu, B., Tang, M., Zheng, L., Zhu, G., Wang, J., Feng, H., Feng, X. & Lu, H. *High-Performance Discriminative Tracking With Transformers* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 9856.

122. Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V. & Darrell, T. *BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

123. Yu, Q., Medioni, G. & Cohen, I. *Multiple target tracking using spatio-temporal markov chain monte carlo data association* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2007).

124. Yu, Y., Xiong, Y., Huang, W. & Scott, M. R. *Deformable Siamese Attention Networks for Visual Object Tracking* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020).

125. Zhang, J., Ma, S. & Sclaroff, S. *MEEM: robust tracking via multiple experts using entropy minimization* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2014).

126. Zhang, L., Li, Y. & Nevatia, R. *Global data association for multi-object tracking using network flows* in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2008).

127. Zhang, Y., Wang, L., Wang, D., Qi, J. & Lu, H. Learning regression and verification networks for long-term visual tracking. *International Journal of Computer Vision (IJCV)* **129**, 2536 (2021).

128. Zhang, Z., Peng, H., Fu, J., Li, B. & Hu, W. *Ocean: Object-Aware Anchor-Free Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2020).

129. Zhang, Z., Zhong, B., Zhang, S., Tang, Z., Liu, X. & Zhang, Z. *Distractor-Aware Fast Tracking via Dynamic Convolutions and MOT Philosophy* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021).

130. Zheng, L., Tang, M., Chen, Y., Wang, J. & Lu, H. *Learning Feature Embeddings for Discriminant Model Based Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2020).

131. Zhou, Z., Chen, J., Pei, W., Mao, K., Wang, H. & He, Z. *Global Tracking via Ensemble of Local Trackers* in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), 8761.

132. Zhou, Z., Pei, W., Li, X., Wang, H., Zheng, F. & He, Z. *Saliency-Associated Object Tracking* in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), 9866.

133. Zhu, Z., Wang, Q., Bo, L., Wu, W., Yan, J. & Hu, W. *Distractor-aware Siamese Networks for Visual Object Tracking* in *Proceedings of the European Conference on Computer Vision (ECCV)* (2018).