UNIVERSITY OF BIRMINGHAM

TOWARDS MORE FLEXIBLE AND EFFICIENT RGBD OBJECT TRACKING

by

JINYU YANG

A thesis submitted to the University of Birmingham for the degree of

DOCTOR OF PHILOSOPHY

Intelligent Robotics Lab

School of Computer Science

College of Engineering and Physical Sciences

University of Birmingham

September 2023

UNIVERSITY OF
BIRMINGHAM

University of Birmingham Research Archive

e-theses repository

This unpublished thesis/dissertation is copyright of the author and/or third parties. The intellectual property rights of the author or third parties in respect of this work are as defined by The Copyright Designs and Patents Act 1988 or as modified by any successor legislation.

Any use made of information contained in this thesis/dissertation must be in accordance with that legislation and must be properly acknowledged. Further distribution or reproduction in any format is prohibited without the permission of the copyright holder.

© Copyright by JINYU YANG, 2023

All Rights Reserved

Abstract

Object tracking is a fundamental task in the area of computer vision. Recently, RGBD (RGB+Depth) object tracking has gained lots of attention due to the development of depth cameras. Compared to RGB-only tracking, RGBD tracking opens more opportunities for accurate and robust object tracking in complex scenarios, such as background clutter and dark scenes, as depth clues are helpful on object and background interference, especially in color failed occasions.

However, the development of RGBD tracking severely lags behind its RGB counterparts and remains far from sufficient for real-world applications due to various challenges. On the one hand, current RGBD tracking is limited by 2D settings, which constrains RGBD tracking on 2D bounding box descriptions and neglects its potential for flexibility brought by depth information. On the other hand, the efficiency of RGBD trackers is ignored, which impedes the realistic applications of RGBD tracking.

This thesis addresses the aforementioned issues and contributes to more flexible and efficient RGBD object tracking. In particular, a series of works are presented to explore and demonstrate the power of RGBD tracking. The four main contributions of this thesis are:

- The first contribution of the thesis introduces a novel paradigm, *i.e.*, weakly-supervised RGBD video object segmentation, which achieves pixel-level RGBD object tracking under weak supervision. By exploring robust cross-modal fusion, the proposed FusedCDNet performs RGBD tracking on pixel level with only bounding box level supervision in both training and testing.

- The second contribution of this work introduces generic 3D object tracking in RGBD videos. Specifically, a novel *Track-it-in-3D* dataset is proposed with rotated 3D BBox annotation, which bridges the gap between RGBD tracking and point cloud tracking. Also, a strong baseline is given for generic 3D object tracking with color and depth fusion and 3D-level cross-correlation.

- The third contribution presents a study on the training-efficient tracking paradigm, which addresses the high training cost problem in RGBD tracking by applying a prompt learning paradigm. With the proposed cross-modal prompts, both the large-scale RGB knowledge from pre-trained large models and complementary information from depth sensors can be well explored. Moreover, the prompting framework is effective on different multi-modal object tracking tasks and its effectiveness is verified on different multi-modal tracking scenarios, including RGB+D, RGB+T, and RGB+Event tasks.

- Finally, we present an efficient and lightweight approach for RGBD tracking, which is the first study on efficient RGBD object tracking. With efficient modality-aware fusion and lightweight backbone, the proposed RGBD tracker EMT runs at a speed of over 100fps. We also provide on-board scenarios and newly defined overhead space for RGBD aerial tracking. In such a scenario, many more categories (34 classes) can be considered for multi-modal aerial tracking than existing aerial tracking datasets. The corresponding on-board tests demonstrate that the proposed EMT can achieve real-time tracking on edge platforms.

As outlined, it is concluded that, by fully exploiting depth clues for solving the problems in current RGBD tracking, more flexible and efficient RGBD object tracking can be achieved.

Declaration

Parts of this thesis have been included in the following published papers or submitted manuscripts:

1. **Jinyu Yang**†, Zhe Li†, Song Yan, Feng Zheng, Aleš Leonardis, Joni-Kristian Kämäräinen, Ling Shao. RGBD Object Tracking: An In-depth Review. *arXiv preprint arXiv:2203.14134, 2022*. [181] (Supporting Chapter 2.)

2. **Jinyu Yang**, Mingqi Gao, Feng Zheng, Xiantong Zhen, Rongrong Ji, Ling Shao, Aleš Leonardis. Weakly-supervised RGBD Video Object Segmentation. *IEEE TIP 2024*. (Supporting Chapter 3.)

3. **Jinyu Yang**, Zhongqun Zhang, Zhe Li, Hyung Jin Chang, Aleš Leonardis, Feng Zheng. Towards Generic 3D Tracking in RGBD Videos: Benchmark and Baseline. *ECCV 2022*. [182] (Supporting Chapter 4.)

4. **Jinyu Yang**, Zhe Li, Feng Zheng, Aleš Leonardis, Jingkuan Song. Prompting for Multi-Modal Tracking. *ACM Multimedia 2022*. [179] (Supporting Chapter 5.)

5. **Jinyu Yang**†, Shang Gao†, Zhe Li†, Feng Zheng, Aleš Leonardis. Resource-Efficient RGBD Aerial Tracking. *CVPR 2023*. [180] (Supporting Chapter 6.)

To my parents.

Acknowledgements

First and foremost, I would like to express my deepest appreciation to my supervisor, Aleš Leonardis, who gave me the opportunity to study in Birmingham and provided guidance through each stage of the process.

I would also like to thank my co-supervisor, Hyung Jin Chang, for inspiring my interest in the research field.

I would also like to thank my SUSTech supervisor, Feng Zheng, for providing scholarship during my PhD.

I am also thankful to my thesis group members, Iain Styles and Mohan Sridharan, for their valuable guidance throughout my studies.

I was fortunate to receive lots of help and thanks should also go to my workmates for their wonderful collaboration. It is my honor to work and cooperate with many great researchers, Song Yan, Mingqi Gao, Zhongqun Zhang, Zhe Li, Shang Gao, Fangjing Wang, and so on.

Finally, I would like to thank my family and friends for their unconditional support and trust.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Given the description (generally object center and size) of a target object in the first frame of a video, object tracking aims to track the target in the subsequent frames. It plays an important role in computer vision and has numerous practical applications, such as security and surveillance [61], unmanned aerial vehicles (UAVs) [43], autonomous driving [49, 116, 102], augmented reality [122], and autonomous robots [211, 141].

Generally, object tracking in computer vision involves several sub-topics, *e.g.*, single/multiple object tracking according to object numbers [119, 12, 197], 2D/3D/6D object tracking according to object description types [68, 196, 142], online/offline tracking according to data input formats [158, 6], and so on. In this thesis, we mainly focus on single object tracking, which aims to track an arbitrary (class-agnostic or semantics-agnostic) object in a given video. In other words, the supervision is only from the object description in the first frame, and we do not need to know or understand exactly what is being tracked. Thus, it is crucial to precisely track the object of interest, even in complex real-world scenarios. In fact, multiple difficulties exist and challenge tracking algorithms in a long time, *e.g.*, target appearance change, fast motion, target size change, partial/full occlusion, motion blur, target rotation, and so on.

To solve the above difficulties, in the past decades, numerous approaches are proposed

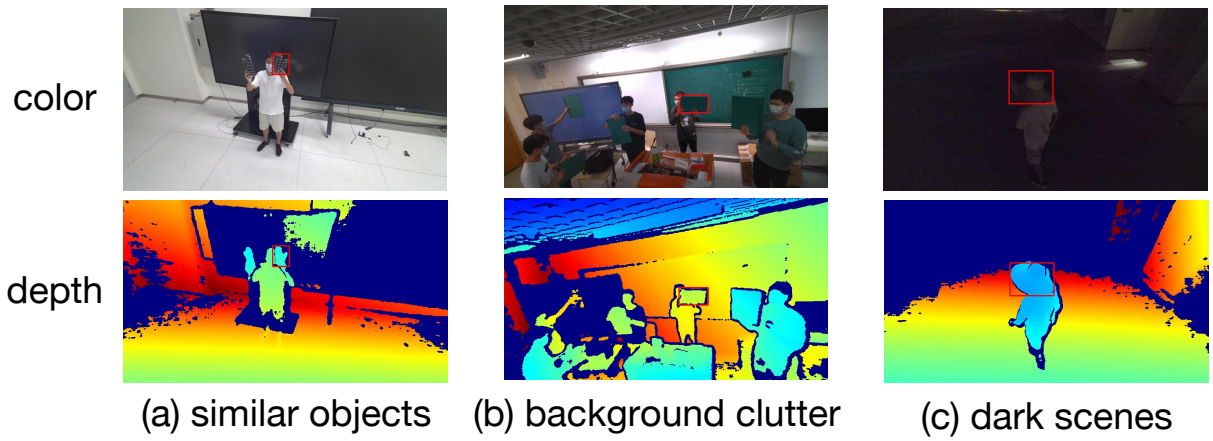


Figure 1.1: Depth favorable scenarios.

for accurate and robust object tracking. Notably, since 2016, the emergence of deep learning based (data-driven) trackers [6, 36, 198, 7, 210, 95, 96] promotes the development of object tracking. Correspondingly, large-scale datasets for object tracking are constructed to boost the performance of data-driven trackers, *e.g.*, GOT-10k [70], LaSOT [42], and TrackingNet [127]. Current advanced trackers can now well handle some challenges like target size change and target rotation. Nonetheless, it is noteworthy that the success of object tracking is limited in color-only area. The state-of-the-art trackers highly depend on the visible conditions and target appearance, leading to tracking failures on background clutters, illumination variance, and dark scenes. Therefore, other modalities are added to provide complementary information, including depth, thermal, and event information [175, 97, 160].

Among them, RGBD (RGB + Depth) object tracking is gaining momentum thanks to great popularity of accessible RGBD cameras, *e.g.*, *Microsoft Kinect V1/V2/Azure DK*, and *Intel RealSense D415/D435/D455*, which can provide synchronous color and depth video streams. Current settings for RGBD object tracking are as follows: given an input RGBD video sequence and the object bounding box in the first frame, it is required to give the object description in the whole video sequence. Then, RGBD tracking shows its effectiveness as depth can provide complementary cues for object tracking, especially in some complex scenarios. The advantages come from different aspects. On the one hand, the depth sensor is insensitive to illumination variation and dark scenes, which can strongly complement RGB domain information. For

example, as shown in Fig. 1.1, depth information is especially helpful in background clutter, dark scenes, and similar objects. On the other hand, a depth camera provides spatial information about objects and background, which is particularly useful in distinguishing objects from background interference.

1.2 Potential Applications

In the past decades, object tracking has attracted enormous attention due to its both academic and application potential. For RGBD object tracking, investigation of its potential is far from sufficient. Since RGBD tracking bridges the gap between 2D tracking and 3D tracking, which focus on target appearance representation learning and target spatial localisation, respectively, it has numerous potential applications. We here summarise the potential applications of RGBD object tracking.

All-day Surveillance. For security reasons, surveillance cameras have proliferated everywhere, including at airports, train stations, bus stations, concert halls and parks. Modern surveillance systems usually require the use of both visual object detection and visual tracking methods to locate suspicious objects and then track their traces. Then, based on the location and traces, you can identify where the object came from and where it went. However, most of them just capture natural light to track moving objects in a scene.

In fact, we spend almost half of our time in the dark, and these cameras don't work well in such dark scenes. Furthermore, ordinary systems with color sensors can be easily fooled using just pictures. Therefore, it is still unsafe for those surveillance systems that only use color information. Fortunately, RGBD tracking methods can successfully avoid such problems by integrating depth information. On the one hand, in dark scenes, depth information is obtained through active light, so it can handle such complex scenes well even at night. RGBD tracking systems, on the other hand, have depth information and therefore cannot be easily fooled by images. Therefore, the RGBD tracking proposed in this paper has the potential to further improve the robustness and performance of visual surveillance systems.

Robotics. Nowadays, robots are becoming more and more popular. Service robots will deliver boxes to people in hotels and answer questions in shopping malls. However, they can only perform simple operations and perform simple tasks because they are unable to perform advanced scene perception and understanding. For example, the interface between humans and these robots relies solely on color information or touch screens to understand human intentions.

Due to the machine's unfriendly and unrobust performance, humans are reluctant to use the machine and instead directly ask the waiter to get a response. In addition, robots can usually work in simple scenarios such as indoor flat ground, but cannot solve problems in complex outdoor conditions. The main problem is that perception in complex scenes is quite difficult because in most cases, robots are only equipped with cameras.

Visual navigation is far from established, as color information is heavily affected by changes in lighting conditions in a scene. Therefore, RGBD can handle these problems better because the depth information is constructed from active light and is more robust. It is more naturally used to exploit depth differences to segment foreground objects. As far as we know, 3D information is more conducive to robot navigation, especially tasks such as obstacle avoidance and feasible area judgment.

Human-Computer Interaction. Gesture control is a new way to interact with machines. For example, humans can play virtual computer games through hand movements or gestures. Gestures can also be defined as control commands, which can then also be used as remote controls to control the TV. In such real-world applications, trajectories are often the most important clues that allow machines to easily understand human intentions. The object tracking algorithm is the most important module for detecting these trajectories.

Typically, however, these pure color-based tracking techniques are very sensitive to lighting changes in complex scenes caused by television or computer screens. Therefore, human-computer interaction technology based on object tracking has not been truly applied to actual interaction scenarios. Fortunately, RGBD tracking can better handle lighting changes by incorporating depth information.

Therefore, HCI technology based on RGBD tracking will be more robust to the challenges

brought by changes in ambient light. XBOX 360 uses such technology and carries RGBD data to achieve friendly interaction between people and games. However, interaction in XBOX 360 is largely limited to human gestures or gestures. Model-free RGBD tracking has a wider application range, and any target can be used as a control device, greatly increasing the comfort and friendliness of interaction.

Unmanned Vehicles and Aerial Vehicles. Nowadays, drones have gradually become a very popular tool that facilitates daily life and are often used for aerial photography. You can take some beautiful photos from different angles that you have never taken before. And relying on drones, we produced a series of documentaries that provide a bird's-eye view of geography and humanities. In fact, drones have huge potential in the future for short-distance logistics transportation or package delivery applications in military scenarios.

However, most current drones are controlled through the drone's radio remote control or pre-written programs. This greatly limits the flexibility and autonomy of drones, thereby limiting the scope of applications. If the drone is equipped with more advanced sensing technology, it can automatically complete the tasks of obstacle avoidance and autonomous navigation.

RGBD tracking aids in obstacle detection and feasible area identification. For example, a flying delivery robot might encounter a flock of birds, but it can easily use the RGBD tracking method to identify the birds' flight trajectories and avoid collisions. This is because depth information is useful for estimating the distance between a drone and birds or other moving targets.

Agriculture Perception. Compared with smart cities, industrial artificial intelligence, smart medical care, etc., smart agriculture is one of the most important fields in social production and life, but current research progress is lagging. Therefore, these autonomous and even smart agricultural technologies have very broad development potential.

Many specific tasks need to be performed, including fruit counting, quality assessment, insect tracking, weed identification, and yield prediction. To accomplish these tasks, the most important step is to detect and track objects in dense scenes, where there are many objects with very similar colors or textures in the background. For example, to implement the fruit counting

task, usually, humans will first send a robot equipped with a camera to take video of the orchard. A fruit detection algorithm will then be used to detect fruit in each frame of the video. The most important step is to use a tracking method to track the fruit and avoid double counting. However, ordinary cameras are actually severely affected by the challenging conditions in an orchard, such as shadows, light changes, and cluttered backgrounds. RGBD tracking methods handle these issues well because depth information is usually active and immune to such challenges.

1.3 Challenges

While the topic is exciting, there remain unsolved difficulties that impede the development of RGBD tracking. Firstly, general tracking challenges that exist in the RGBD tracking community are considered, such as fast motion, motion blur, occlusion, size change, and so on. These challenges also commonly appeared in RGB-only tracking scenarios, as real-world scenarios are always complicated. In addition to those challenges often discussed in ordinary single-modal tracking scenarios, we notice that there are two fundamental issues in the RGBD tracking area: more fine-grained object state representation, and the efficiency of both model's training and testing.

Then, we will mainly introduce the challenges from the following two aspects.

1.3.1 Limitations on Object Description

The purpose of object tracking is to determine the exact location of an object in a video and then generate an object bounding box to define the location of the object. This raw representation of position, which solely uses an axis-aligned bounding box, can often be used to track the movement of an object or the trajectory of a person. However, in many scenarios, bounding boxes are too primitive to achieve more fine-grained tasks, especially in depth favorable scenarios.

Mask generation. For example, in an embodied intelligence scenario, a robot needs to determine the precise boundaries of an obstacle through accurate perception to avoid it and keep safe. However, the bounding box output by ordinary tracking algorithms often focuses on the

center position of the target, leaving a considerable part of the background in the box. It is difficult for a robot based on this algorithm to accurately avoid certain obstacles. Although this kind of method does not affect the functional implementation of the general tracking algorithm, it is difficult to upgrade to another more refined scenario. In contrast, generating a target mask can solve this problem very well. Once the robot accurately determines the boundaries of obstacles, it can better plan its path to successfully avoid obstacles. Although masks are more useful, generating masks for objects is also more difficult. Compared with only locating the target, similarity calculation of image blocks is generally performed, while segmentation technology requires pixel-level similarity calculation, so it is more difficult. In addition, the probability that the background contains pixels similar to the target pixel is greater. However, introducing depth information will effectively avoid this confusion and will be beneficial to the generation of target masks.

3D bounding box generation. In image space, using merely a two-dimensional bounding box to define the position of an object loses a lot of structural and spatial information. This information is very critical in fields such as human-computer interaction. For example, we can still use the scenario of robot perception to explain this problem. Without three-dimensional information, it would be difficult for the robot to find the corresponding target in the scene, and it would also have no way to accurately perceive the three-dimensional structural information of the target. As a result, it is impossible to accurately grasp the corresponding target. Therefore, it is very important to accurately predict the three-dimensional information of objects in space. However, it is very challenging to recover a 3D bounding box directly from the two-dimensional image space. The inherent reason is that three-dimensional information is lost when forming the image.

1.3.2 Necessity on Model Efficiency

Involving additional depth information usually leads to cumbersome tracking models, especially for the data-driven deep trackers. Whether it is training or prediction phase, efficiency is a very important indicator to verify whether the built model has actual application value. In this

thesis, two very important challenges including training efficiency and prediction efficiency are considered.

Training efficiency. The efficiency of building trackers is also a very important issue that deserves to be solved. Generally speaking, efficiency is limited by two factors: the construction of the dataset and the training of the model. As mentioned before, fusing RGBD information can greatly improve tracking performance, especially when dealing with more complex scenes. However, it is very challenging to construct an RGBD dataset of a certain size to train multi-modal trackers. Compared with single-modal data sets, it is more difficult to build a platform that can collect RGBD scenes, and data annotation is also very labor-intensive. This severely limits the development of RGBD trackers, causing most existing trackers to be undertrained. Therefore, how to effectively train multi-modal trackers in the presence of insufficient data is critical.

Inference efficiency. As object tracking is a real-time application, tracking speed is an important metric, and, unfortunately, efficiency has long been ignored in RGBD tracking. Currently, most multi-modal trackers, such as RGBD trackers, suffer from high memory cost and low tracking speed, which severely limits their application scope. We note that researchers of multi-model trackers pay more attention to the performance of the models, so the speed of most RGBD trackers cannot meet real-time requirements. For example, even though participants in the VOT competition achieved good performance on hybrid datasets and several subtests, the speeds of their models were relatively low. Achieving a good balance between performance and speed remains a challenge for RGBD model design. This is also a problem that has to be solved shortly. In addition, real-world applications, especially on-board applications, such as drones and autonomous vehicles with weak computing power and storage capabilities, require lightweight architectures.

1.4 Contributions

The contributions made in this thesis are summarised below:

The first contribution is that a weakly-supervised RGBD video object segmentation is proposed to achieve pixel-level RGBD object tracking under weak supervision, in which the following three contributions are made: (1) Weakly-supervised RGBD video object segmentation (VOS) is proposed to achieve pixel-level tracking under the supervision of bounding boxes. (2) A benchmark dataset for weakly-supervised RGBD VOS - *DepthVOS* is proposed, which includes 350 challenging video sequences with bounding box and mask annotations. (3) *FusedCDNet* is proposed to perform pixel-level RGBD tracking with bounding box-level supervision in both training and testing. Experiments verify that *FusedCDNet* can handle various difficulties by successfully exploring robust cross-modal fusion and weakly-supervised training and prediction.

The second contribution is that generic 3D object tracking in RGBD videos is presented, in which the following contributions are made: (1) Generic 3D object tracking in RGBD videos is proposed, which aims to achieve class-agnostic 3D tracking in complex scenarios, bridging the gap between RGBD tracking and point cloud tracking. (2) We generate the benchmark *Track-it-in-3D*, which is the first benchmark for generic 3D object tracking. It contains challenging RGBD videos covering multiple tracking difficulties, with dense 3D BBox annotations and corresponding evaluation protocols provided. (3) We introduce a strong baseline, *TrackIt3D*, for generic 3D object tracking, which handles 3D tracking difficulties by RGBD fusion and 3D cross-correlation. Extensive experiments are executed on the proposed benchmark to facilitate future research.

The third contribution is that the high training cost problem in multi-modal tracking is addressed by applying a prompt learning paradigm. The contributions are three-fold: (1) A novel prompt learning paradigm is proposed for RGBD tracking, in which both the large-scale RGB knowledge from pre-trained models and the complementary information from depth sensors are effectively utilized. (2) We present a principled approach to cross-modal prompt configurations for various kinds of multi-modal tracking but without the inappropriate fine-tuning process. (3) We unify different multi-modal object tracking tasks into a prompting framework and conduct comprehensive experiments on different scenarios that demonstrate the

effectiveness of ProTrack. To the best of our knowledge, this is the first attempt at prompt learning in multi-modal tracking areas.

The last contribution is that a lightweight RGBD tracking approach is proposed for efficient RGBD tracking and both on-board scenarios and corresponding tests are provided. The contributions are three-fold: (1) We propose RGBD aerial tracking for newly defined overhead space (2m - 5m). Unlike previous aerial tracking, this task is more relevant to human life and has wider applications. (2) We construct a large-scale high-diversity benchmark for RGBD aerial tracking. The advantage is that many more categories (34 classes) can be considered than existing aerial tracking datasets. As far as we know, this is the first dataset that can test multi-modal aerial tracking models. (3) An efficient tracking baseline is proposed for RGBD aerial tracking, which is the first real-time tracker for efficient on-board multi-modal tracking. It performs better than classical UAV trackers and maintains comparable efficiency.

1.5 Thesis Structure

This thesis is divided into the following chapters:

Chapter 2 contains a literature review of relevant previous works on RGBD object tracking. Specifically, basic concepts and methods in the area of generic object tracking are introduced first. Then, the development of RGBD object tracking is presented. To be detailed, the taxonomy of the existing RGBD trackers from different perspectives, *i.e.*, depth feature extraction, depth usage, and RGBD fusion strategy. All existing RGBD object tracking benchmarks and evaluation metrics are reviewed as well. Finally, we also review some relevant topics, including video object segmentation, 3D tracking, and efficient tracking.

Chapter 3 mainly introduces a weakly-supervised paradigm to achieve video object segmentation in RGBD videos with only bounding box-level supervision. In detail, an RGBD video object tracking dataset is first proposed with per-frame mask annotation, which can be used for pixel-level tracking performance evaluation. Then, a novel method FusedCDNet is proposed for weakly-supervised RGBD video object segmentation, based on dedicated cross-modal fusion

and a three-stage weakly-supervised training paradigm. Finally, extensive experiments show that the proposed FusedCDNet yields state-of-the-art tracking performance under bounding box-level weak supervision.

Chapter 4 mainly addresses the 3D object tracking problem in RGBD videos. In particular, we annotate the rotated 3D bounding boxes for generic objects in generic scenarios in a newly constructed dataset. Corresponding evaluation protocols are given for such settings. Then, a strong baseline is given, which addresses the 3D tracking difficulties by color and point cloud fusion and dedicated 3D cross-correlation. Finally, extensive experiments demonstrate that the proposed baseline can outperform current point cloud-based trackers and overcome several challenges in 3D scenarios.

Chapter 5 presents a training-efficient tracker. Firstly, the gap between RGB-only and multi-modal tracking is analyzed. Then, a novel prompt learning mechanism is proposed for RGBD tracking, with combining the large-scale knowledge from color modality and the complementary information from depth modality. Finally. the effectiveness of the proposed ProTrack paradigm is also verified on other multi-modal tracking tasks, including RGBT and visible-event tracking tasks, resulting in state-of-the-art performance on three tasks and five benchmark datasets.

Chapter 6 proposes an efficient RGBD tracker, which is the first study on RGBD efficient tracker. Firstly, the EMT tracker is proposed with very early fusion and lightweight backbones, which can run at over 100fps on GPUs. Then, the application of efficient RGBD tracking is proposed as the RGBD aerial tracking task, which requires the tracker to run on edge platforms and address the challenges in overhead spaces. Correspondingly, a flight platform is built and a large-scale dataset is constructed. Finally, extensive experiments on the platform and dataset verify the effectiveness of the proposed method.

Chapter 7 presents the conclusions and future work.

The overview of the thesis structure is given in Figure 1.2.

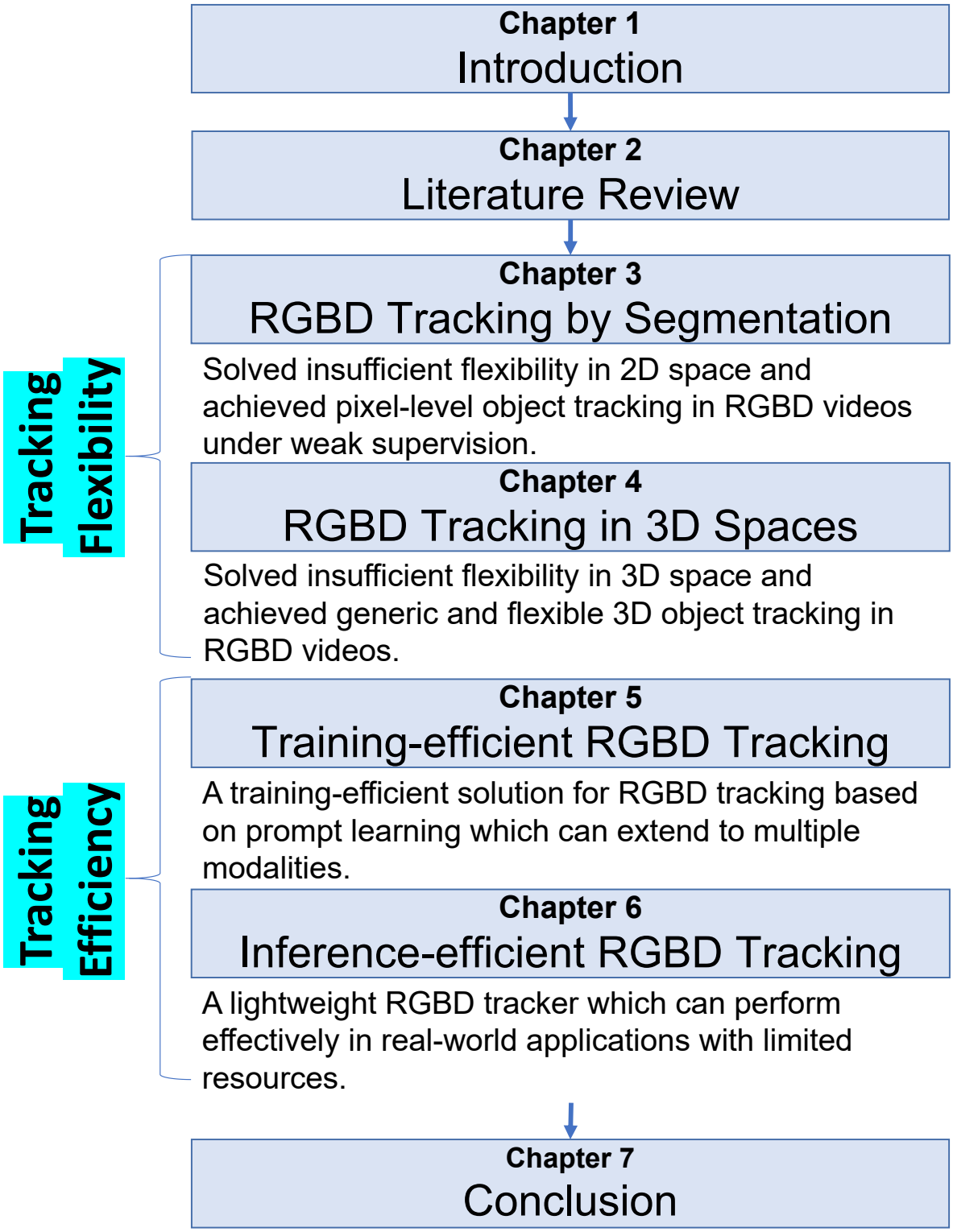


Figure 1.2: Thesis structure.

Chapter 2

Literature Review

This chapter will first give a broad view of generic object tracking, including some basic concepts and background information. Secondly, we specifically introduce the development of RGBD object tracking and give corresponding in-depth analysis. Then, works from related tasks, which are used to support the observations and solutions, are introduced.

Declaration: The literature review regarding RGBD object tracking has been public on arXiv [181], to benefit the research community.

2.1 Generic Object Tracking

Generic object tracking aims to track any object in a video sequence, providing the initial size and location of the target in the first frame. Early tracking methods primarily concentrated on representing and pursuing target features using techniques such as particle filters, Mean-shift, Kalman filters, and similar techniques.

Recent years have witnessed a great development of object tracking algorithms. Problems like object occlusion, deformation, and rotation, are still challenging in this field and have been extensively investigated by researchers [177, 74, 147]. A large amount of RGB trackers, especially short-term trackers, emerged to boost this community. Since 2013, Correlation Filter (CF) has been introduced to solve the template matching problem and occupied the mainstream thanks to its effectiveness and efficiency. Representative trackers include MOSSE, discriminative

correlation filter (DCF), kernelized correlation filter (KCF) [65], and so on.

Due to the popularity of deep neural networks in computer vision, researchers have the opportunity to explore and solve more challenges in more complex scenarios. Since 2016, deep learning-based (or data-driven) methods gradually prevail. Siamese network and deep correlation filter based trackers are very popular in object tracking [6, 36, 198, 7, 210, 95, 96, 118]. Progress in RGB tracking has been further boosted by the emergence of standard datasets and evaluation protocols. Correspondingly, large-scale datasets are proposed for model training and evaluation, *e.g.*, GOT-10K [70], LaSOT [42], and TrackingNet [127], which boosts the development of deep trackers. This field is also fueled by the annual VOT challenges [89, 90, 91, 87, 85, 86].

2.2 RGBD Object Tracking

Although significant progress has been made in RGB-based tracking, there are still tracking failures that are hard to be solved by color information [181]. Therefore, other modalities are added to provide complementary information, including depth, thermal, and event information [175, 97, 160]. Among them, RGBD (RGB + Depth) object tracking is gaining momentum in the past decade thanks to affordable advanced depth cameras, such as Microsoft Kinect and Intel RealSense. On the one hand, depth maps provide essential cues on occlusion reasoning and depth-based object segmentation [155, 15]. For example, CA3DMS [109] uses a context-aware 3D mean-shift to handle occlusion, and DM-DCF [79] proposes a depth-based segmentation to train a constrained Discriminative Correlation Filter (DCF). On the other hand, RGBD channels can sense both appearance and geometric components for better object-and-background separation. For example, OTR [80] uses both color and depth information to build a spatial reliability map and reconstruct an object 3D model. Therefore, exploring the depth cue is indeed helpful for multi-modal tracking.

Early RGBD trackers utilize direct heuristic extensions of RGB-based methods, which tend to extract hand-crafted features from depth maps to solve specific challenges [181]. For example,

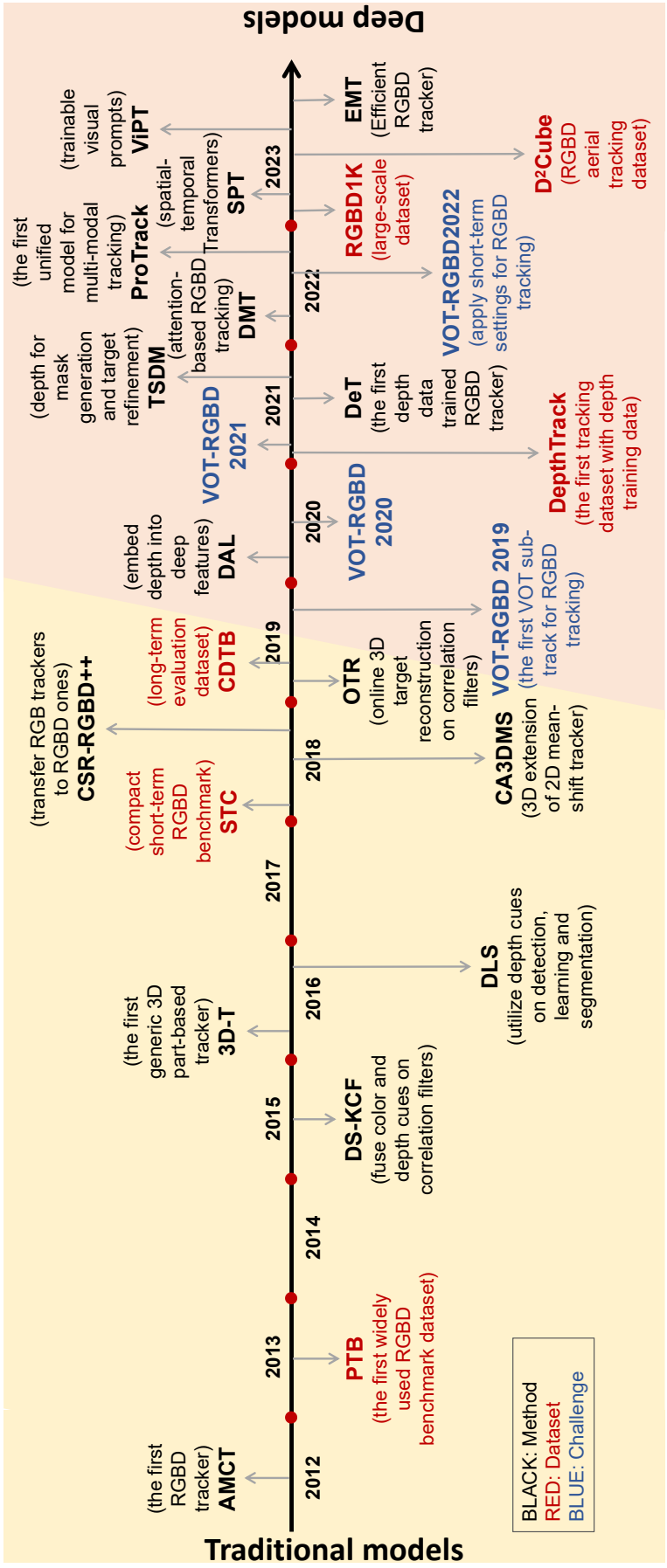


Figure 2.1: Chronology of RGBD object tracking.

PT [149] introduces a set of RGBD baseline trackers, including a traditional 2D tracker with additional depth HOG features, a 2D optical flow tracker, and a 3D point cloud tracker. Recently, deep networks are also introduced to RGBD tracking, but they are still straightforward extensions of RGB baselines [200, 175]. At the same time, the annual VOT challenge [87] has had a specific track for RGBD input since 2019. By providing a test set and evaluation protocols, RGBD tracking can gain more attention in the object tracking community. Until now, there have been multiple participant RGBD trackers in the VOT-RGBD challenges [87, 85, 86]. According to VOT reports, the VOT trackers show superior performance over traditional solutions and continuously improve the state-of-the-art. A brief chronology of RGBD tracking is shown in Fig. 2.1.

2.2.1 Models and Taxonomy

To systematically review RGBD trackers, we categorize the models into three main taxonomies: RGB and depth fusion paradigm, depth usage, and tracking framework [181]. We specifically focus on the tracker's participants in VOT RGBD tracking challenges. In the following, several representative models in each taxonomy will be described. Table 2.1 summarizes existing RGBD trackers.

Early/late RGBD fusion

The existing fusion strategies in RGBD tracking can be categorized into two main groups: early fusion and late fusion, which combine the RGB and depth streams in different stages.

Early fusion. Early fusion based models generally follow two strategies: RGB and depth images are first fed into independent feature extraction modules separately and then combined as a joint representation. Then the combined feature map is used to obtain a final prediction, such as DS-KCF [14], CSR_RGBD++ [78], ECO_TA [92]. Another strategy is integrating the depth channel with RGB channels to form a four-channel input, such as PT [149]. For early fusion, MCBT [157] combines optical flow, color, and depth information, which are simultaneously incorporated to predict the precise position. Camplani *et al.* [14] proposed DS-KCF which

Table 2.1: Statistics of RGBD tracking models. "CL/DL" indicates whether the tracker is a classical/deep-learning based method. "ST/LT" indicates short-term/long-term trackers. "Occ." indicates occlusion handling.

Method	Year	Publication	CL/DL	Framework	Backbone	Training data	ST/LT	Occ.	Code
AMCT [54]	2012	JDOS	CL	Condensation	-	-	ST		
PT [149]	2013	CVPR	CL	SVM	-	-	ST		
MCBT [157]	2014	Neurocomputing	CL		-	-	LT		
DS-KCF [14]	2015	BMVC	CL	KCF [65]	-	-	ST		✓
OL3DC [203]	2015	Neurocomputing	CL	SURF	-	-	LT	✓	
CDG [72]	2015	CAC	CL	SVM	-	-	LT	✓	
DOHR [38]	2015	FSKD	CL	Bayesian	-	-	LT	✓	
ISOD [25]	2015	Signal Processing	CL		-	-	LT	✓	
DS-KCF_shape [60]	2016	J.RTIP	CL	KCF [65]	-	-	ST		✓
3D-T [9]	2016	CVPR	CL	KCF [65]	-	-	LT	✓	
OAPF [125]	2016	CVIU	CL	Particle Filter	-	-	LT	✓	
DLS [3]	2016	ICPR	CL	KCF [65]	-	-	LT	✓	
ODIOT [202]	2017	Neural Processing Letters	CL	TLD	-	-	LT	✓	
ROTSL [120]	2017	ITEE	CL	Particle Filter	-	-	LT	✓	
STC [167]	2018	IEEE TCYB	CL	KCF [65]	-	-	ST		✓
CSR_RGBD++ [78]	2018	ECCVW	CL	CSR_DCF [114]	-	-	LT	✓	✓
DM-DCF [79]	2018	ICPR	CL	CSR_DCF [114]	-	-	ST		
SEOH [94]	2018	IEEE Access	CL	KCF [65]	-	-	LT	✓	
OACPF [191]	2018	IEEE Access	CL	Particle filter	-	-	LT	✓	
RT-KCF [193]	2018	CCDC	CL	KCF [65]	-	-	LT	✓	
CA3DMS [109]	2018	IEEE TMM	CL	MeanShift	-	-	LT	✓	✓
OTR [80]	2019	CVPR	CL	CSR_DCF [114]	-	-	LT	✓	✓
Depth-CCF [100]	2019	IOP	CL	DCF [83]	-	-	LT	✓	
ECO_TA [92]	2019	IEEE Sensors	CL	ECO [37]	-	-	ST	✓	
RGBD-OD [169]	2019	CIS	CL	PointNet [135]		Point Cloud	LT	✓	
H-FCN [76]	2019	Information Fusion	CL	ECO [37]	-	-	LT	✓	
3DMS [58]	2019	ICST	CL	Mean Shift	-	-	LT	✓	
WCO [108]	2020	IEEE Sensors	CL	DCF [83]	-	-	ST		
RF-CFF [161]	2020	Applied Soft Computing	DL	KCF [65]	VGG-Net	RGB pre-trained	LT	✓	
SiamOC [195]	2020	ICSP	DL	SiamDW [198]	ResNet-18	RGB pre-trained	LT	✓	
DAL [138]	2020	ICPR	DL	ATOM[36]	ResNet-18	RGB pre-trained	ST		✓
3s-RGBD [164]	2021	Neurocomputing	DL	SiamFC [6]	AlexNet	RGB pre-trained	LT	✓	
TSDM [200]	2021	ICPR	DL	SiamRPN++ [96]	ResNet-50	RGB pre-trained	ST		✓
DeT [175]	2021	ICCV	DL	DiMP [7]	ResNet-50	RGB+D	ST		✓
ProTrack [179]	2022	ACMMM	DL	STARK [173]	ResNet-50	RGB Pretrained	ST		✓
DMT [52]	2022	ECCVW	DL	DiMP [7]	ResNet-50	RGB+D	ST		✓
ViPT [207]	2023	CVPR	DL	OSTrack [188]	ViT	RGB+D	ST		✓
SPT [209]	2023	AAAI	DL	STARK [173]	Transformer	RGB+D	LT	✓	✓
ATCAIS19	2019	VOT-2019	DL	ATOM [36]	ResNet-18	RGB pre-trained	LT	✓	✓
SiamDW_D	2019	VOT-2019	DL	SiamDW [198]	ResNet-50	RGB pre-trained	LT	✓	✓
SiamM_Ds	2019	VOT-2019	DL	SiamMask [158]	ResNet-50	RGB pre-trained	LT	✓	✓
LTDSEd	2019	VOT-2019	DL	LT-DSE [87]	ResNet-50	RGB pre-trained	LT	✓	✓
ATCAIS20	2020	VOT-2020	DL	ATOM [36]	ResNet-18	RGB pre-trained	LT	✓	✓
CLGS_D	2020	VOT-2020	DL	SiamMask [158]	ResNet-50	RGB pre-trained	LT	✓	✓
DDiMP	2020	VOT-2020	DL	SuperDiMP[7]	ResNet-50	RGB pre-trained	ST		✓
Siam_LTD	2020	VOT-2020	DL	SiamRPN [95]	ResNet-50	RGB pre-trained	LT	✓	✓
sttc_rgbd	2021	VOT-2021	DL	STARK [173]	ResNet-50	RGB pre-trained	ST	✓	✓
STARK_RGBD	2021	VOT-2021	DL	STARK [173]	ResNet-50	RGB pre-trained	LT	✓	✓
SLMD	2021	VOT-2021	DL	PrDiMP [35]	ResNet-50	RGB pre-trained	LT		✓
TALGD	2021	VOT-2021	DL	SuperDiMP [7]	ResNet-50	RGB pre-trained	LT	✓	✓
DRefine	2021	VOT-2021	DL	SuperDiMP [7]	ResNet-50	RGB pre-trained	ST		✓
MixForRGBD	2022	VOT-2022	DL	MixFormer [33]	Transformer	RGB+D	ST		✓
SAMF	2022	VOT-2022	DL	MixFormer [33]	Transformer	RGB pre-trained	ST		✓
SBT_RGBD	2022	VOT-2022	DL	SBT & DeT [175]	Transformer	RGB+D	ST		✓

utilized HOG features for both color and depth maps. OAPF [125] employs multiple features, *e.g.* HoG, from color and depth streams to improve robustness against illumination changes and clutter and boost performance. Another example is DeT [175], which extracts deep depth features through an additional ResNet50 [77] branch. With fusing the color and depth features, it can use ATOM [36] or DiMP [7] tail part to perform the actual tracking. DMT [52] first researched how to fuse two modality features, they designed two modules for extracting shared information between dual modality inputs and augmenting shared information with the modality-specific features. Unlike the earlier fusion at the feature level, ProTrack [179] tries to fuse depth and color information before processing the images through the backbone network. They discussed that for the fusion of two modalities with similar physical meanings, the earlier fusion gets the better fusion effect [181].

Late fusion. Late fusion based models generally process both modalities simultaneously, and the independent models for RGB stream and depth stream are built to make decisions [181]. For example, Xiao *et al.* proposed STC [167] which fuses two single-modal trackers through weighted maps. CDG [72] uses depth gradient information to extract depth motion models and weights the results from RGB and depth models. RT-KCF [193] fuses response maps instead of features to get a good performance. RF-CFF [161] focuses on fusing tracking results from RGB and depth images, in which objects are tracked separately in RGB and depth images using the correlation filter. Then results are adaptively fused.

2D/3D depth usage

Since depth maps provide appearance descriptions and geometry information for tracked objects, some methods treat depth maps in 2D and 3D structures, respectively [181].

2D usage. In 2D view, a depth map naturally provides a texture-free segmentation between foreground and background, so it is common to use depth cues for object segmentation. In CSR_RGBD++ [78], a depth-augmented foreground segmentation is formulated by graph cut to obtain a foreground mask in the target update. DM-DCF [79] extracts the depth-based segmentation masks to train a constrained DCF. More recently, DeT [175] utilizes colormaps of

depth maps to transfer the depth information into colored ones [181].

3D usage. Depth information provides the 3D spatial description of objects. Representative trackers include OTR [80], CA3DMS [109] and 3D-T [9]. Among them, 3D-T [9] is the first 3D part-based tracker, which exploits parts to preserve temporal structural information and helps in particle pruning. CA3DMS [109] finds the 2D adjacent objects are separated in 3D space. They propose a 3D extension of the classical mean-shift tracker, which uses a model adaption strategy and an occlusion handling strategy guided by 3D information. OTR [80] implements online 3D object construction to learn a robust view-specific discriminative correlation filter (DCF), extending the 2D tracking structure to 3D representation. The 3D construction benefits the tracking performance from two aspects: generation of spatial description for the constrained 2D DCF learning; and 3D pose estimation based on point clouds to localize the object after heavy occlusion [181].

Mixed usage. Hybrid trackers jointly use 2D and 3D models to combine 2D appearance features and 3D spatial features. For example, DLS [3] simultaneously builds two target models: a 2D appearance model built upon the features extracted from both color and depth frames, and a 3D distribution model built according to the point cloud distribution on the target surface. The depth histogram is used to adaptively segment the target in depth frames and project the point cloud into a depth image patch. SEOH [94] employs the spatial continuity in depth values for scale estimation, and a part-based model updating strategy to deal with occlusion. Another representative is TSDM [200], consisting of an RGB tracking core and two assistant modules. First, the core is SiamRPN++ [96], which takes an image pair (template and mask images) as input. Then, a mask-generator module utilizes depth information to generate a mask image for a candidate search image. Finally, a depth-refiner module cuts out non-target areas from the original outputs and gives a more compact and precise mask [181].

Heuristic/deep models

Depth information provides useful cues, *i.e.*, boundary cues, and helps to identify object characteristics. Over the past several years, many traditional trackers with handcrafted features have

been designed by using these specific cues. For example, MCBT [157] uses the depth mean and variance in the target region to measure the difference between candidates and templates. PT [149] proposes a series of baseline trackers with handcrafted features, including a traditional 2D image patch-based tracker, a 3D point cloud-based tracker, and a low-level optical flow-based tracker. STC [167] first uses depth HOG features, and then the RGB and depth features are separately used in KCF to find the target position in a global layer. As shown in Table 2.1, there are many methods specially designed for occlusion handling since depth cues straightforwardly indicate the target locations. ISOD [25] exploits depth information obtained from binocular video data to detect occlusion, which prevents improper appearance model updating during occlusions. Meshgi *et al.* [125] proposed an occlusion-aware particle filter framework that employs a probabilistic model with a latent variable to represent an occlusion flag. Liu *et al.* [109] proposed a context-aware 3D Meanshift method, which compares depth differences between the target and the occluder, and between the nearby 3D point and the occluder to detect and recover from tracking failures caused by full occlusions. CSR_RGBD++ [78] sets multiple assumptions in the occlusion recovery stage (the positions are similar before disappearing and after disappearing, the target speed remains constant) to recapture the object [181].

However, due to the limited-expression ability of handcrafted features, deep neural networks are introduced to RGBD tracking. Generally, deep learning-based models follow two principles: 1) Trackers integrate RGB features extracted by pre-trained deep neural networks with handcrafted depth features into heuristic tracking frameworks [138, 200]. 2) Trackers are trained on both RGB and depth data jointly to obtain deep depth features as well as deep RGB features. However, up to now, almost all trackers follow the first principle and remain using the deep features extracted by pre-trained models on RGB datasets. Some representative models are briefly introduced here. In addition to the Siamese tracking network, SiamOC [195] provides two modules to consider both depth histogram characteristics and movement smoothness, simultaneously. The underlying assumption is that different objects have different depth histogram characteristics. DAL [138] embeds depth information into deep RGB features through the reformulation of a deep discriminative correlation filter (DCF). TSDM [200] equips

SiamRPN++ [96] with two assistant depth-related modules. Until 2021, the trainable RGBD tracker DeT [175] is first proposed by duplicating a separate feature extraction branch for depth colormaps [181].

VOT participants

Since 2019, there have been 4 VOT-RGBD challenges [87, 85, 86, 88] held annually, consisting of 13 participating trackers in total. Most participants in VOT just provide a tracking model and brief introduction without specific publications yet they show outstanding performance in multi-challenges. Here we review representative VOT participants.

2019 Winner: SiamDW-D is a long-term tracker that addresses the problems of target appearance variations and frequent target loss. It contains three parts, *i.e.*, the main tracker, a re-detection module, and a multi-template matching module. The main tracker is based on [198], and further equips with an online updating model [128, 36]. The re-detection module is triggered when the main tracker is not confident in its predictions. The multi-template matching module is to output a more reliable estimation when the tracking results are unreliable with history templates. Moreover, depth information is used to estimate the disappearance of target objects.

2020 Winner: ATCAIS combines both instance segmentation and depth information for accurate tracking. It is based on ATOM [36] and the HTC instance segmentation method [20], which is retrained in a category-agnostic manner. The instance segmentation results are used to detect background distractors and to refine the target bounding boxes to prevent drifting. The depth value is used to detect the target occlusion or disappearance and redetect the target.

2021 Winner: STARK_RGBD is a tracker combining STARK [173] and SuperDiMP [7]. Here STRAK variant DeiT [151] is used to strengthen the features of STARK. To better handle the appearance change of the target, DeiT combines with the SuperDiMP model. Specifically, when the STARK tracker's confidence is low or the prediction of STARK suddenly strays away, SuperDiMP takes over the tracking process, providing an appearance adaptive result. In addition, a refinement module based on AlphaRefine [172] is applied to the final output of the

Table 2.2: Comparison of existing RGBD tracking datasets. "LT/ST" denotes long-term or short-term sequences.

Dataset	Publ.	LT/ST	#Seq.	#Frame	#Avg.Len.	#Attr.	#Split	Scenario	Sensor	Resolution
PTB [149]	CVPR	ST	100	20,332	203	5	-	indoor	Kinect	640×480
STC [167]	TCYB	ST	36	9,195	255	12	-	indoor & outdoor	Xtion	640×480
CDTB [113]	ICCV	LT	80	101,956	1274	13	-	indoor & outdoor	Kinect; Basler	960×540
DepthTrack [175]	ICCV	LT	200	294,600	1473	15	150/50	indoor & outdoor	RealSense	640×360
RGBD1K [209]	AAAI	LT	1,050	2,503,400	2,384	15	1,000/50	indoor & outdoor	ZED	640×360

whole tracking system to further boost the quality of box estimation.

2022 Winner: MixForRGBD is built based on MixFormer [33], in which the online template is updated at regular intervals when the prediction confidence is larger than the preset threshold. The original depth map is transferred to colormap and then input into the backbone networks, which extract and fuse the features of the template and search area between the two modalities. Furthermore, an element-wise max operation is performed to merge the two modalities, and a simple post-processing strategy is added to penalize large displacements.

Besides the annual winners, other participants also gave promising solutions for RGBD tracking. In DDiMP [85], depth information is utilized to prevent the target scale from changing too quickly. As targets cannot have very large displacements in two consecutive frames, SiamM_Ds [87] obtains the average depth value in the candidate patches as constraints to determine the target location. TALGD [86] uses depth images for occlusion or disappearance reasoning and target retrieval. CLGS-D [87] uses depth maps to filter region proposals. DRefine [86] fuses RGB and depth information jointly in the input. Note that some participants do not use depth information indeed, such as sttc_rgbd and STARK_RGBD, while they still keep high performance [181].

2.2.2 Datasets and Evaluation Metrics

Datasets

Early datasets for RGBD object tracking only contain a few sequences for evaluation. For example, a small-scale dataset called BoBoT-D [54] was proposed in 2012, consisting of five RGBD video sequences captured by Microsoft Kinect v1.0. In [157], four videos were captured and utilized for evaluation with four different objects (Book, Face, Inno, and TeaCan). These video sequences represent different challenges, such as occlusion, rotation, illumination change,

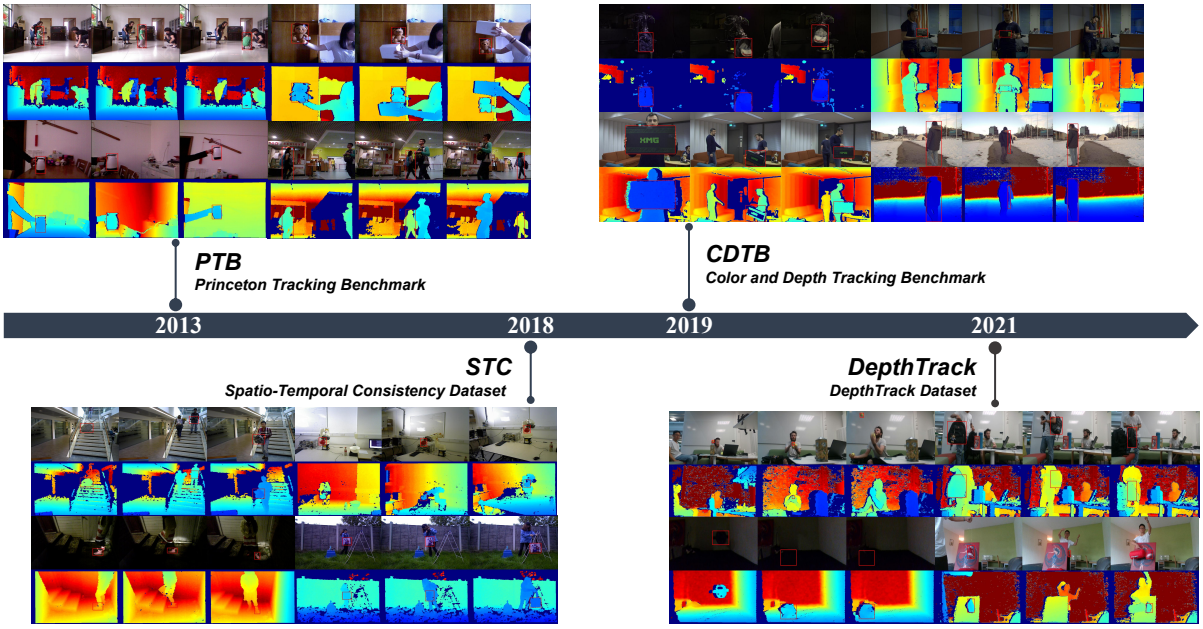


Figure 2.2: Samples in different RGBD tracking datasets.

shape variation of a flexible object, and small targets, and are manually annotated by the groundtruth every five frames [181].

From 2013, benchmark datasets have appeared for wide use. Up to now, there are four datasets designed and widely used for RGBD object tracking. Table 2.2 compares the four representative RGBD tracking datasets. Fig. 2.2 gives chronology and examples of these datasets [181]. The details of each dataset are given as follows:

Princeton Tracking Benchmark (PTB) [149]. As the first dataset designed for RGBD object tracking, PTB contains 100 RGBD video clips recorded by Microsoft Kinect v1.0. There are only 3 types: human, animal, and rigid object, with more than half of the sequences, are for people tracking. For a fair comparison, PTB withholds groundtruth for 95 videos and hosts an online evaluation server to allow new result submissions. The remaining 5 videos are public for tracker validation before submitting their results. It is worth noting that the RGB and depth channels are poorly calibrated in PTB. In approximately 14% of sequences, the RGB and D channels are not synchronized and approximately 8% are miss-aligned, which was fixed by [9].

Spatial-Temporal Consistency (STC) dataset [167]. It was proposed in 2018 to address the drawbacks of the PTB dataset. It is designed to increase the data diversity with a compact

size of only 36 sequences for short-term RGBD tracker evaluation. STC constrains the dataset mostly to indoor scenarios and there are only a few low-light outdoor video clips. Although the STC dataset is the smallest dataset among the RGBD tracking family, it is annotated with 13 attributes. It uses two kinds of evaluation metrics imported from OTB [163] and VOT protocols [89].

Color and Depth Tracking Benchmark (CDTB) [113]. It includes 80 video sequences for long-term tracking with an average video length of 1274 frames. With long-term settings, objects are possibly fully occluded or out-of-view for a long duration and thus, it can be used to evaluate the re-detection performance. Note that the CDTB is the only dataset acquired by multiple color-and-depth sensors, which guarantees its diversity of realistic depth signals. Indoor, as well as outdoor scenarios, are covered to extend the tracking domains [181].

DepthTrack dataset [175]. It consists of 200 sequences, which is the currently largest and most diverse dataset for RGBD tracking. Specifically, it includes the most diverse object types (46 categories in 50 test sequences), scenarios (indoors and outdoors with 15 attributes), and video length (varying from 143 to 3816 frames). Until now, it is the first and the only public RGBD tracking dataset divided into training and test sets. In addition, with long-term tracking settings, the DepthTrack dataset is dedicated to exploring the depth-related power to assist in tracking challenges [181].

RGBD1K dataset [209]. To promote the rapid development of RGB-D tracking, RGBD1K dataset is proposed, which contains 1,050 sequences (~2.5M frames). In RGBD1K, 1,000 videos are for training and 50 videos for testing. While, only the first 600 frames of the training sequences are annotated.

Evaluation metrics

Although there are only four datasets, their evaluation protocols are different. In this section, we give representative evaluation metrics for RGBD object tracking in detail, *i.e.*, Success Rate (SR), Pr-Re (Precision-Recall), and F-score [181].

Success Rate (SR). Inspired by PASCAL VOC challenge [66], for t-th frame, the overlap ratio

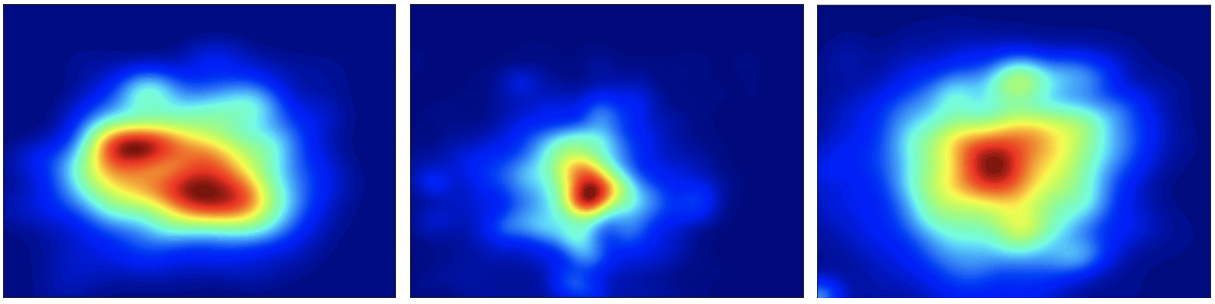
r_t between the predicted bounding box A_t and the groundtruth bounding box G_t is:

$$r_t = \begin{cases} \frac{area(A_t \cap G_t)}{area(A_t \cup G_t)} & both\ A_t\ and\ G_t\ exist \\ 1 & neither\ A_t\ or\ G_t\ exists \\ -1 & otherwise \end{cases} \tag{2.1}$$

Then, a minimum overlapping area ratio r can be used to decide whether the output is correct. Thus, the average success rate R of each tracker is defined as follows:

$$R = \frac{1}{N}\sum_{t=1}^{N} u_t, \qquad where\ u_t = \begin{cases} 1\ if\ r_t > r \\ 0\ otherwise \end{cases} \tag{2.2}$$

where u_t denotes whether the output bounding box of the t-th frame is acceptable, and N is the number of frames.



(a) STC (b) CDTB (c) DepthTrack

Figure 2.3: Target center distributions in RGBD datasets.

Precision-Recall (Pr-Re) & F-score. According to the settings of the VOT challenges [87, 85], the most popular metric in RGBD tracking is the precision-recall and F-score [115]. As the video length over different datasets and sequences varies dramatically, there are frame-based and sequence-based evaluation protocols. At frame t, θ_t is a prediction confidence score and τ_θ is a classification threshold. If the predicted confidence score θ_t is not below τ_θ, $A_t(\tau_\theta)$ is used to denote the corresponding prediction. Otherwise, the output is an empty set and we set $A_t(\tau_\theta) = \varnothing$. Thus, $\Omega(A_t(\tau_\theta), G_t)$ can be used to indicate the intersection-over-union (IoU) between the prediction result $A_t(\tau_\theta)$ and the groundtruth G_t. Then, the *frame-based evaluation*

is as follows:

$$
\begin{aligned}
Pr(\tau_\theta) &= \frac{1}{N_p} \sum_{A_t(\tau_\theta)\neq\varnothing} \Omega(A_t(\tau_\theta), G_t), \\
Re(\tau_\theta) &= \frac{1}{N_g} \sum_{G_t\neq\varnothing} \Omega(A_t(\tau_\theta), G_t), \\
F(\tau_\theta) &= \frac{2Re(\tau_\theta)Pr(\tau_\theta)}{Re(\tau_\theta) + Pr(\tau_\theta)},
\end{aligned}
\tag{2.3}
$$

where $F(\tau_\theta)$, $Pr(\tau_\theta)$ and $Re(\tau_\theta)$ denote the F-score metric, the precision (Pr) and the recall (Re) over all frames, respectively. N_p denotes the number of frames in which the target is predicted visible, and N_g denotes the number of frames in which the target is indeed visible [181].

For *sequence-based evaluation*, the precision-recall over all sequences is as follows:

$$
\begin{aligned}
Pr(\tau_\theta) &= \frac{1}{M} \sum_{i=1}^{M} Pr^i(\tau_\theta), \\
Re(\tau_\theta) &= \frac{1}{M} \sum_{i=1}^{M} Re^i(\tau_\theta),
\end{aligned}
\tag{2.4}
$$

where $Pr^i(\tau_\theta)$ and $Re^i(\tau_\theta)$ denote the precision and recall metrics for i-th sequence among M test videos. F-score is obtained in the same way as Eq. 2.3.

Annotations and attributes

All datasets are given with per-frame axis-aligned bounding box annotation. Fig. 2.3 illustrates the bounding box distributions for STC, CDTB, and DepthTrack datasets. The PTB dataset is not included since its groundtruth is not available. As shown, there are high discrepancies between bounding box distributions of different datasets.

In the field of object tracking, "attribute" defines possible challenging factors and different characteristics of each sequence. Existing datasets propose various attributes from various perspectives as follows [181].

1) PTB dataset evaluates trackers on *target type* (human/animal/rigid), *target size* (large/small), *movement* (slow/fast), *occlusion* (yes/no), and *motion type* (active/passive).

2) STC dataset annotates the attributes per sequence, including *Illumination Variation* (IV), *Color/Depth Distribution Variation* (CDV/DDV), *Surrounding Depth/Color Clutter* (SDC/SCC), *Background Color/Shape Camouflages* (BCC/BSC), *Partial Occlusion* (PO), and *Depth/Scale Variation* (DV/SV).

3) CDTB and DepthTrack datasets share the most attributes, including the common challenges that appeared in RGB-based tracking: *Aspect Change* (AC), *Fast Motion* (FM), *Non-rigid Deformation* (ND), *Full/Partial Occlusion* (FO/PO), *Out-of-plane Rotation* (OP), *Out-of-frame* (OF), *Size Change* (SC), and *Similar Objects* (SO), and depth-related challenges: *Dark Scene* (DS), *Depth Change* (DC), and *Reflective Targets* (RT).

4) In particular, the DepthTrack dataset considers two more challenges, such as *Background Clutter* (BC) and *Camera Motion* (CM) which are indirectly related to depth scenarios but challenging to RGBD tracking. Notably, RGBD1K also involves the two attributes and results in the same attribute settings as DepthTrack.

2.3 Other Related Topics

2.3.1 Video Object Segmentation

Video Object Segmentation (VOS) aims to separate the target (region of interest) from the background in a video sequence, which can be seen as a kind of more fine-grained object tracking. Compared to object tracking, VOS achieves more detailed target descriptions by pixel-level masks, rather than the bounding boxes used in object tracking.

Given an input video and the target object annotated in the first frame, Semi-supervised Video Object Segmentation (SVOS) aims to track and segment the target from all subsequent frames [50]. Unlike other video segmentation techniques, such as Video Instance Segmentation (VIS) [183] or Video Semantic Segmentation (VSS) [159], which only focuses on the fixed-size of object categories, Semi-supervised VOS has a more flexible problem setting, where the objects with any categories can be segmented once they are annotated in the first frame. Thus, Semi-supervised VOS has recently drawn much attention in the community. To facilitate progress

in this field, several benchmarks have been proposed, such as DAVIS [131] and YouTube-VOS [170]. They focus on RGB videos only and form many challenges for training and evaluation.

With the problem setting, early works leverage online fine-tuning [10, 152, 123, 57], optical flow [73, 27, 165], or feature matching [190, 129, 153] for SVOS. Specifically, they consider the first frame annotations as the template. During segmentation, the template is used to fine-tune network parameters or to match the current frame features directly. Despite achieving good results, the early works only take the first and previous frames into account. However, the intermediate frames also contain valuable clues, which are first utilized in STM [130] and then raise a trend of memory-based SVOS. In such methods, both the first and intermediate frames are considered as the template. In this way, more object changes can be captured to facilitate subsequent frame segmentation. Recently, memory-based methods have dominated SVOS benchmarks, in both segmentation accuracy and efficiency. The well-designed benchmarks and models promote the progressive development of SVOS and its application in reality. However, current RGB-only VOS models are limited by the visible conditions. In poor visible conditions, *e.g.*, foggy weather, or dark scenes, VOS models will face severe performance degradation as they can only rely on the target and background appearance.

2.3.2 3D Object Tracking

In 3D tracking, the task is defined as getting a 3D BBox in a video sequence given the object template of the first frame [182]. In general, 3D single object tracking is still constrained by tracking on raw point clouds. SC3D [56] extends the 2D to 3D Siamese tracker on point clouds for the first time, in which exhaustive search is used to generate candidates. P2B [137] is proposed to solve the drawbacks of SC3D by importing VoteNet to construct the point-based correlation. Also, the 3D region proposal network (RPN) is utilised to obtain the object proposals. However, the ambiguities among part-aware features weaken the tracking performance severely. After that, BAT is [201] proposed to directly infer the BBox by box-aware feature enhancement, which is the first to use box information. Recent works make multiple attempts with the image prior[212],

multi-level features[162], or transformers [32] to handle these problems, but the performances remain low with only point cloud provided. On the other hand, current RGBD tracking follows 2D BBox settings [85, 87, 86], while there were works devoted to predicting the 2D BBox in 3D view. In 2016, Bibi et al. developed 3D-T [9] which used 3D BBox with particle filter for RGBD tracking. In 2018, OTR [80] generated a 3D BBox to model appearance changes during out-of-plane rotation. However they only generated incomplete 3D BBoxes in a rough level and served for 2D predictions. Regarding the datasets, in 3D tracking, LiDAR is the most popular sensor due to distant view and insensitivity to ambient light variations. The commonly used benchmarks on the 3D tracking task are *KITTI* [55] and *NuScenes* [11]. KITTI contains 21 outdoor scenes and 8 types of targets. NuScenes is more challenging, containing 1000 driving scenes across 23 object classes with annotated 3D BBoxes. With respect to their volume, the data diversity remains poor with focusing on driving scenarios and restraining methods to track objects in point clouds [182].

2.3.3 Efficient Object Tracking

Object tracking initially gained lots of attention due to the increase in demand for real-time applications, *e.g.*, surveillance systems, autonomous driving, and human-computer interfaces [180]. Trackers based on correlation filters [65, 37] use hand-designed features to enable real-time operation on the CPU. Although they are fast, their reliance on handcrafted features greatly hinders their performance compared to more sophisticated methods. While increasing tracking performance is gaining attention, trackers' speed and real-time nature are gradually diminished. In recent years, the popularity of deep learning-based trackers [158, 210, 47] has significantly improved the performance of visual trackers thanks to deeper networks and newer architectures, *e.g.*, Transformers. However, these advancements often come at the expense of more expensive models. Thus, how to design a lightweight network for efficient tracking is of concern. This requires the trackers to, not only be accurate and robust but also action in real time under the hard computational constraints on limited hardware. For example, LightTrack [174] utilizes the Neural Architecture Search (NAS) paradigm to find a lightweight and efficient Siamese tracking

architecture. However, unlike color-based object tracking, multi-modal tracking efficiency is unexplored [180].

Chapter 3

RGBD Object Tracking by Segmentation

In order to increase the fine-grained output of the multi-modal tracking algorithm, this chapter will first introduce a new algorithm that achieves the output of the target mask by better incorporating depth information.

Declaration: The materials of this section have been organized as a journal paper, which was accepted by TIP.

Background. Video Object Segmentation (VOS) is the problem of performing pixel-wise classification of target objects in a video sequence. As an extension of bounding box based object tracking [112, 158], VOS is especially helpful for tracking non-rectangular or non-rigid targets, which are common in real-world applications. Thus, it can be widely used in applications like video surveillance [59], robotics [31], and autonomous driving [144][117]. However, many difficult cases in color view, including complicated backgrounds and different lighting conditions, remain challenging for VOS. For example, it is difficult to distinguish the objects under background clutter and dark scenes. One way to overcome these challenges is to employ depth information, which provides complementary spatial information for RGB videos. In RGBD domain, RGBD tracking is gaining increasing attention [87, 85, 86] since depth camera is introduced into computer vision tasks. Depth information is utilized to solve tracking failures appearing in color-view videos, *e.g.* background distractors, or target deformation. Such a depth-based solution has the potential to achieve pixel-level tracking in more complicated

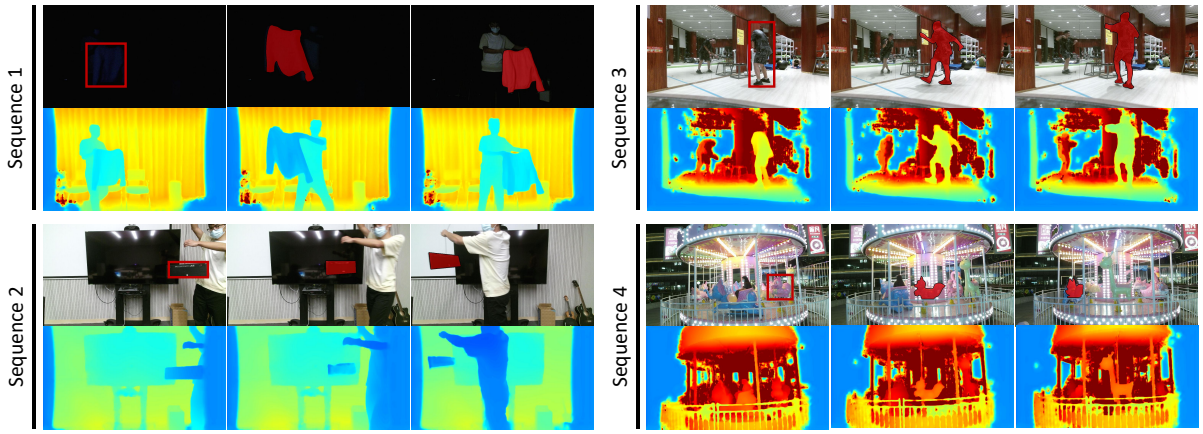


Figure 3.1: Example sequences from our dataset *DepthVOS* and corresponding annotations. Video sequences consist of multiple challenges for RGBD VOS, *e.g.*, (1) dark scenes and deformable objects; (2) background clutter; (3) similar targets and deformation; (4) target rotation and fast motion. Bounding box and mask annotations are given in the RGB modality.

environments [21][171]. However, to the best of our knowledge, there is no such flexible and fine-grained pixel-level general object tracking, *i.e.* VOS, in the RGBD domain.

3.1 Preliminaries

3.1.1 Motivation

Early works observed that depth information is able to reduce the feature ambiguity that appeared in the RGB domain for accurate object segmentation, especially in complex scenarios. As can be seen from the above, the RGBD VOS task has potential wider applications [48], but cannot be implemented simply and directly. In fact, to this end, we still have to address several following challenges, including: 1) Existing RGBD video datasets are mainly for RGBD object tracking, the content and settings are not designed for RGBD video object segmentation. 2) Segmentation mask annotation is high-cost and time-consuming. Compared to object bounding box annotations, pixel-wise mask ones are far more expensive, requiring ~15× more time [106]. Therefore, box annotations are more pervasive than pixel-wise annotations. 3) The depth value is not fully explored to assist segmentation and tracking, especially how to extract and fuse the depth feature in multi-modality mode is still an open problem. In fact, current RGBD trackers

are simply extended from classical RGB trackers, in which the RGB and depth fusion in deep neural networks is preliminary and straightforward [138].

Aiming to solve the aforementioned problems, we propose a novel task of weakly-supervised video object segmentation in RGBD videos. Through training the network with a weak supervision paradigm, a new view to solve the VOS task in the RGBD area is provided. In detail, we define the setting of weakly-supervised RGBD VOS as an RGBD VOS model trained merely under bounding box supervision is used to predict the pixel-level mask of objects as the output with only the bounding box level initialization in test phases. Therefore, the tracker is supervised by only bounding boxes in the same way as RGBD tracking, while it is required to retain pixel-accurate segmentation masks.

To boost this topic, we generate a new benchmark named *DepthVOS*, which provides 350 RGBD videos in total, with annotations of bounding boxes and segmentation masks. In the DepthVOS dataset, bounding boxes are designed mainly to supervise the tracker, while the segmentation masks are provided for comparison and evaluation. As shown in Fig. 6.6, different challenges in RGBD VOS are included. Moreover, we develop a novel framework based on RGBD fusion, named Fused Color-Depth Network (*FusedCDNet*), which is a novel attempt to perform VOS in multiple modalities. Experiments validate its effectiveness due to 1) RGBD fusion provides a robust description in some RGB-failed scenarios, *e.g.*, low illumination and background clutter and enhances the RGB-only features with cross-modal information; 2) weakly-supervised training overcomes the high-cost labeling, and 3) weakly-supervised prediction to obtain more accurate descriptions enhances the practicality of our proposed model in RGBD VOS scenarios.

Our contributions can be concluded as follows:

- A new task - weakly-supervised RGBD video object segmentation (VOS) - is defined as pixel-level tracking under the supervision of bounding boxes. To our knowledge, this is the first attempt to apply a weakly-supervised paradigm on VOS in the RGBD domain.
- A new benchmark dataset for weakly-supervised RGBD VOS - *DepthVOS* - is proposed, which includes 350 challenging video sequences with bounding box and mask annotations.

- A new method - *FusedCDNet* - performs multi-modal VOS with bounding box level supervision in both training and testing. Experiments verify that *FusedCDNet* can handle various VOS difficulties by successfully exploring robust cross-modal fusion and weakly-supervised training and prediction.

3.1.2 Problem Formulation

Generally, VOS models are required for accurate target localization and fine-grained target description. Obviously, RGBD VOS has the potential to be widely used, but the main challenge of VOS is that mask annotations are difficult to collect. Therefore, to overcome the high cost of labeling, we define our weakly-supervised VOS learning mechanism with bounding box annotations, as follows:

$$Training : [I, D, B] \rightarrow [Model], \tag{3.1}$$

where I and D represent the color and depth frames from the RGBD videos, respectively. B indicates the bounding boxes of corresponding targets in the first frame. Furthermore, to increase ease of use, we formulate our weakly-supervised VOS inference mechanism, with only a bounding box provided in the first frame, as follows:

$$Model : [I_1, D_1, B_1; I_t, D_t] \rightarrow [M_t], \tag{3.2}$$

where $[I_1, D_1, B_1]$ denotes the template and M_t is the generated binary mask for frame t. The definition of our weakly-supervised VOS problem is illustrated in Fig. 3.2, in which we expect a high-level output from very weak one-shot supervision in video tasks.

3.1.3 Comparison with Related Tasks

We here show the differences between our proposed task and closely related ones.

RGBD object tracking. This is to track an arbitrary object with a bounding box from an RGBD video, given only the object's location in the first frame. As an extension of RGBD

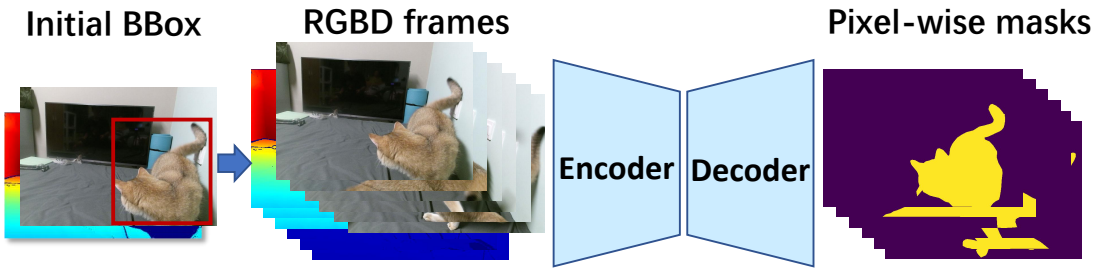


Figure 3.2: Illustration of the proposed task. Weakly-supervised RGBD VOS requires only a bounding box of the region of interest for initialization and results in pixel-wise mask description.

tracking, we retain all the settings of current RGBD object tracking despite the output. With the same level of supervision in training and inference, we require a high-level prediction (pixel-accurate mask).

Semi-supervised VOS. This is the closest related task that gives the segmentation mask in the first frame and asks the algorithm to predict the object segmentation. Unfortunately, there is no dataset dedicated to semi-supervised VOS in RGBD domains as it requires pixel-level supervision. Our proposed task bridges this gap by training under weak supervision.

RGBD video saliency detection. It can be viewed as an unsupervised VOS task, but it is a pity that both unsupervised VOS and video saliency detection are at very preliminary stages in RGBD communities. Thus, our work aims to develop the RGBD video analysis to a fine-grained level.

Semi-supervised video object segmentation. Semi-supervised video object segmentation (VOS) is to classify all pixels in a video sequence into foreground and background, given the ground truth mask of the object in the first frame. Early works mainly apply semantic segmentation methods to handle the VOS challenges with online fine-tuning, leading to low computation speed [10, 132, 69]. Recent works focus on importing memory bank and target-specific information to segmentation networks, which brings improvements in efficiency [130, 67]. Specifically, tracking by segmentation methods emerged recently, which are viewed as the combination of semi-supervised video object segmentation and object tracking. Segmentation trackers like [158, 8, 112] show a good trade-off between accuracy and speed, while there are no such methods designed in the RGBD community.

RGBD visual object tracking. Early RGBD trackers are based on handcrafted depth features

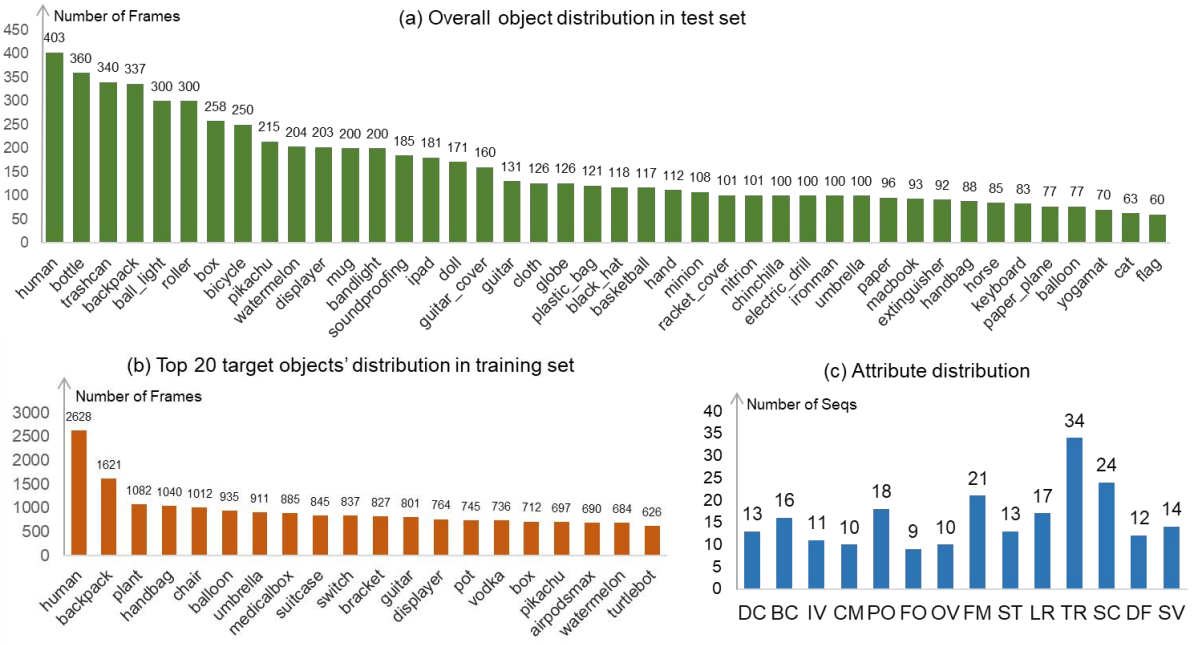


Figure 3.3: Overall distribution of our proposed *DepthVOS*. (a) Distribution of objects in our test set; (b) Distribution of the top 20 frequently appeared objects in our training set; (c) Distribution of attributes over the video sequences in the test set.

to handle challenges like occlusion [149, 109, 78]. So far, most successful RGBD trackers are based on RGB trackers and use depth information as an auxiliary to improve performance. Until very recently, there emerged deep RGBD trackers. for example, DAL [138] embedded the depth feature into RGB deep features through reformulation; DeT [175] achieves end-to-end training network on RGBD data with duplicated depth branches. However, RGBD trackers remain on predicting bounding boxes, which impedes precise mask output as the RGB tracking community did.

Weakly-supervised segmentation. Weak supervision is introduced into semantic segmentation to handle the deficiency of annotated segmentation data. The common types of weak supervision include bounding boxes, scribbles, and points. Our work is most related to bounding box supervised segmentation. For bounding box-level supervision, pseudo segmentation masks are generated by GrabCut [145] and MCG [133] proposals for training in early works [34, 82]. Recent work applies multiple instance learning (MIL) loss to best use the bounding box information [206]. In addition, generating a segmentation mask from weak supervision, *e.g.*, the bounding box, is also gaining attention as a sub-task [103, 199, 184].

3.2 DepthVOS: Dataset for Weakly-supervised RGBD VOS

3.2.1 Dataset Background

There exist datasets for video object segmentation or RGBD tracking, but none of them has been specifically designed for RGBD VOS. Here we review the related RGBD video datasets. Existing widely used datasets for RGBD object tracking, including *Princeton Tracking Benchmark* (PTB) [149], *Spatio-Temporal Consistency* dataset (STC) [167], *Color and Depth Tracking Benchmark* (CDTB) [113] and *DepthTrack* [175], provide per-frame axis-aligned bounding boxes covering target object only. All of them give test data for evaluation, while DepthTrack is the only one providing a training dataset with 150 video sequences, which indicates that large-scale training datasets are still lacking in the RGBD domain. There are much less pixel-accurate RGBD video datasets compared with the RGB ones. Specifically, unlike RGB communities, the RGBD video object segmentation remains unexplored. Here we review the related RGBD datasets with object mask annotations. The *SUN3D* [166] has 415 sequences captured from 254 different spaces, which is for place-centric scene understanding. The *SBM-RGBD dataset* [13] is built to evaluate background modeling methods for moving object detection in RGBD videos. This dataset consists of 33 videos of indoor visual data captured in video surveillance and smart environment scenarios. It was chosen to address the challenge of moving object detection due to the presence of static background in the videos. So far, there are no datasets designed for RGBD VOS challenges with pixel-wise annotations.

3.2.2 Data Acquisition

To facilitate the study on RGBD VOS, we aim to provide a diverse set of groundtruthed synchronized color and depth sequences specifically for RGBD VOS challenges. Our dataset, namely *DepthVOS*, consists of 350 videos (~56k frames), selected to cover a wide range of challenges for object segmentation in RGBD videos. For the test set, videos are mostly captured with *Microsoft Kinect V2* and *Intel Realsense SR300 & D455* by ourselves at $30 fps$ and cropped to 640×360 spatial resolution. Depth images are recorded at 16 bits. To make the dataset more diverse and

Table 3.1: Attributes and corresponding description.

Type	Abb.	Attribute	Description
Depth-related	DS	Dark Scene	The light is too low to distinguish the target.
	BC	Background Clutter	There are distractors around the target object.
	IV	Illumination Variation	There are illumination changes.
Video-based	CM	Camera Motion	The camera moves/shakes during video capturing.
	FO	Full Occlusion	The object is fully occluded.
	OV	Out of View	Object is partially or fully leave the view.
	FM	Fast Motion	The average per-frame object motion is larger than 20 pixels.
	ST	Similar Targets	There are similar objects that appeared in the view.
	TR	Target Rotation	Target rotates in-plane or out-of-plane.
	SV	Scale Variation	Ratio of target area is smaller than 50%.
	PO	Partial Occlusion	Object is partially occluded.
Segmentation	LR	Low Resolution	The ratio of the object area to the image size is lower than 10%.
	SC	Shape Complexity	The object has a complex boundary (*e.g.* irregular or jagged).
	DF	Deformation	The object is non-rigid or deformable.

better evaluate the generalization ability of different methods, we also select and clip 5 video sequences that are challenging to our task from existing CDTB [113] and DepthTrack [175] as test sequences. Therefore, our test set contains 50 sequences with 6712 frames accompanied by densely annotated, pixel-accurate, and per-frame groundtruth segmentation mask of the target object. Corresponding bounding boxes of targets are also provided. For the training set, to our best knowledge, only DepthTrack provides training data in a volume of 150 sequences which is not suitable for VOS. Thus, in DepthVOS, we provide a training set manually captured for VOS challenges with per-frame box-level annotations, resulting in 300 training sequences. Some examples of annotated data are shown in Fig. 6.6. Since our goal is generic object segmentation in RGBD videos, the types of objects specified in the test set may not even appear in the training set. We give the full object distribution in the test set and the top 20 object classes in the training set in Fig. 6.9 to validate the diversity of the proposed dataset.

3.2.3 Challenges

In addition, to analyse how different kinds of challenges influence the performance, we annotate all the test videos with 14 attributes, including Dark Scene (DS), Background Clutter (BC), Illumination Variation (IV), Camera Motion (CM), Partial Occlusion (PO), Out of View (OV), Full Occlusion (FO), Target Rotation (TR), Similar Targets (ST), Low Resolution (LR), Scale Variation (SV), Fast Motion (FM), Shape Complexity (SC), and Deformation (DF). As the attributes are not exclusive, a sequence can be annotated with multiple attributes. The distribution of attributes in our test set is shown in Fig. 6.9(c). Except for the common challenges in video object analysis, *e.g.* TR and PO, there are attributes dedicated to depth-based scenarios, *i.e.*, DS, BC, and IV, which can be handled by depth information provided. In addition, SC, DF, and LR are especially challenging to the segmentation task to measure the segmentation accuracy. The description for each attribute is given in Table 6.2.

3.2.4 Evaluation Metrics

Following [131], two popular metrics are adopted for evaluation - *Region Similarity* ($\mathcal{J}$) and *Contour Accuracy* ($\mathcal{F}$). $\mathcal{J}$ is to measure the region-based similarity of segmented masks, which is defined as the Intersection-over-Union (IoU) of the output segmentation (M) and the groundtruth mask (G). We employ the Jaccard index as:

$$J = \frac{|M \cap G|}{|M \cup G|}. \tag{3.3}$$

$\mathcal{F}$ is to evaluate the accuracy of contour prediction by calculating the contour-based F-score. The segmented mask M and groundtruth G can be treated as a set of closed contour regions $c(M)$ and $c(G)$. Thus, the contour-based precision P_c and recall R_c can be calculated between the contour points via a bipartite graph matching to be robust to small inaccuracies, as proposed in [41]. F-measure is used to obtain a good trade-off between precision and recall, defined as:

$$F = \frac{2P_cR_c}{P_c + R_c}. \tag{3.4}$$

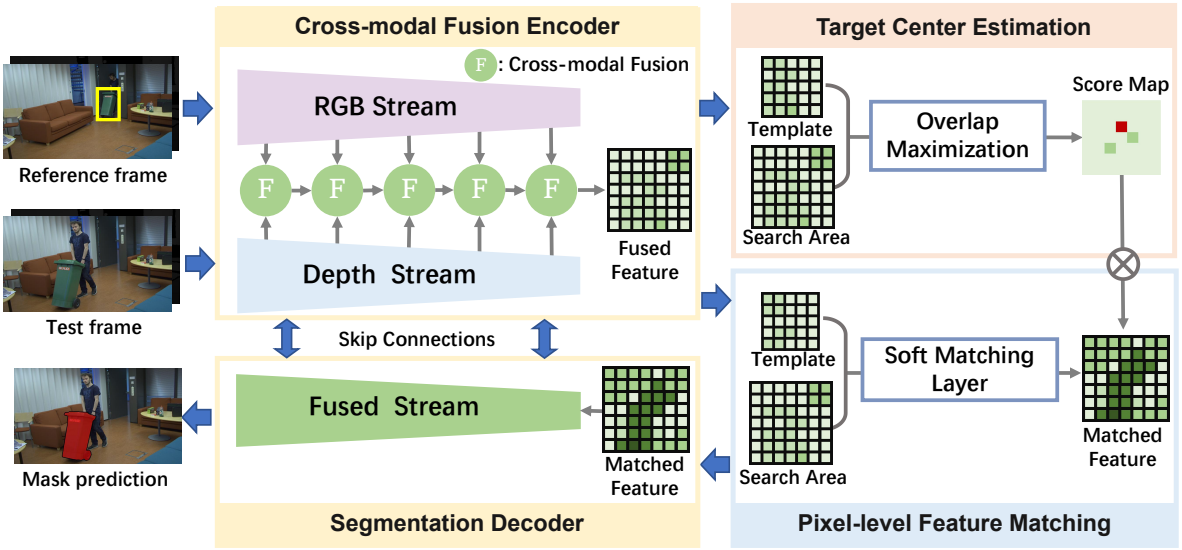


Figure 3.4: Overall framework of the proposed *FusedCDNet*. Our network consists of four main components. The cross-modal fusion encoder is to obtain the cross-modal features. The target center estimation module predicts the target center using the fused feature and produces a score map mixed with pixel-level feature matching. Finally, the matched features are fed into the decoder to output the predicted mask.

We here follow the setting in DAVIS [131] with approximating the bipartite matching via morphology operators.

3.3 FusedCDNet: Achieving Pixel-level Tracking under Weak Supervision

In this section, we propose a novel baseline for the problem of segmenting an object of interest in RGBD videos, given only its bounding box in the first frame, under weak supervision. First, we describe the pipeline, including the cross-modal feature extractor for RGBD fusion, the modules for robust target localization and foreground feature matching, and the segmentation decoder to generate the mask output. Sec. 3.3.2 gives the training procedure, and Sec. 3.3.3 gives the inference details.

3.3.1 Architecture

Overview. Fig. 3.4 shows the framework of the proposed Fused Color-Depth Network (Fused-CDNet) for RGBD VOS, which is built based on U-Net [143] structure. The reference and test frames are cropped into the template and search area first and fed into corresponding backbone networks to extract hierarchical features. Five multi-level features are extracted in RGB and depth sub-networks, *i.e.*, $F_I = \{f_I^i, i = 1, 2, 3, 4, 5\}$ and $F_D = \{f_D^i, i = 1, 2, 3, 4, 5\}$, which enables to learn the individual features for each modality. Here we design a cross-modal fusion module to fully exploit joint feature representation of color and depth modalities for each layer, *i.e.*, $F_S = \{f_S^i, i = 1, 2, 3, 4, 5\}$. The fused feature in the template and search area are utilised for target center estimation to get a candidate region for pixel-accurate prediction. The fused feature (f_S^5) of the template and the search area are also separately fed into the pixel-level feature matching module to extract the foreground and background information in the search area to obtain the matched feature. Finally, the matched feature is concatenated and upsampled with a segmentation decoder, in which features in the first four levels of the encoder are integrated into the decoder via skip connections.

Cross-modal fusion encoder. This module is to effectively fuse the cross-modal features from both color and depth modalities to learn a fused representation. To conveniently process the color and depth information, we first normalize the cross-modal features from the backbone networks as follows. We use a 1×1 convolutional layer to reduce their dimensionality. The hierarchical features in the subnetworks, *i.e.* F_I and F_D, are then fed into a 3×3 convolutional layer with Sigmoid activation, after what we can get the normalized feature maps, $H_I = \{h_I^i, i = 1, 2, 3, 4, 5\}$ and $H_D = \{h_D^i, i = 1, 2, 3, 4, 5\}$. According to [205], the normalized feature maps can be seen as the attention maps for different modalities. Thus, we enhance the hierarchical features in this way:

$$\hat{f}_I^i = f_I^i + f_I^i \otimes h_I^i, \ \hat{f}_D^i = f_D^i + f_D^i \otimes h_D^i, \tag{3.5}$$

in which $\hat{f}_I^i$ and $\hat{f}_D^i$ denote the enhanced color and depth features, for the i-th layer, respectively. Here $\otimes$ denotes the element-wise multiplication operation.

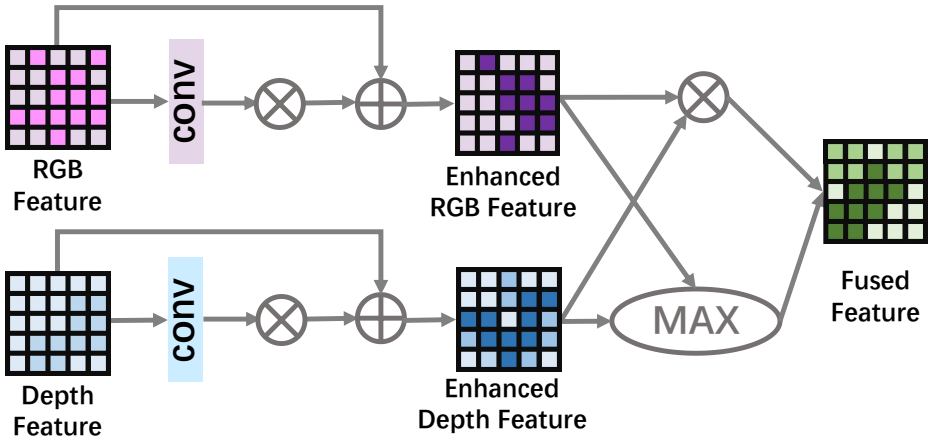


Figure 3.5: Calculation of cross-modal fused features.

After obtaining the enhanced features $\hat{f}_I^i$ and $\hat{f}_D^i$, we calculate element-wise multiplication and maximization and then concatenate them to obtain a strong fused RGBD feature. The fusion can be formulated as:

$$f_S^i = \hat{f}_I^i \otimes \hat{f}_D^i + Max(\hat{f}_I^i, \hat{f}_D^i). \tag{3.6}$$

This module effectively exploits the correlations from multiple modalities. The calculation process of f_S^i is shown in Fig. 3.5. Besides, the fused feature f_S^i is propagated to the next layer to integrate the cross-level information. Finally, the last-layer fused feature f_S^5 is fed into target center estimation and multi-stream feature matching modules.

Target center estimation. This module aims to determine the target location given the template and the search area features. Here we only use the fused feature f_S^5 to estimate the target center with a popular discriminative correlation filter formulation [36]. Given the fused feature of the target in the reference frame $f_S^5(T)$, this module is trained to predict the IoU between the target and a set of candidate boxes on the test frame. By maximizing the predicted IoU, the module refines the candidates to get the final score map R. Maximum of R is considered as the target center, which indicates the candidate region for multi-stream feature matching.

Pixel-level feature matching. While the target center estimation module provides accurate target center outputs, it lacks the ability to precisely distinguish the target object from background distractors pixel by pixel. Thus, we perform pixel-level feature matching as follows. Given a

target template T, this module is to precisely distinguish the target object from background distractors pixel by pixel. In this module, we also use the fused feature f_S^5 from the template and search area. Once the template feature map $f_S^5(T)$ is calculated, we collect foreground features m_F and background features m_B by partitioning pixels belonging to foreground and background, according to the supervision from the reference:

$$m_F = \{f(T)^j : j \in g(T)\},\ m_B = \{f(T)^j : j \notin g(T)\} \tag{3.7}$$

Hereby $g(T)$ is the set of pixels belonging to the foreground as indicated by the mask supervision downsampled to the same size of $f(T)$. To obtain the fused information, f_S^5 is fed into the pixel-level feature matching module. The fused feature f_S^5 is matched with the m_F and m_B via the soft matching layer [69], with which we can obtain the foreground and background matching scores for each pixel. The soft matching layer is to compute a matching score matrix that measures the similarity between the search feature map and $\{m_F, m_B\}$. We compute the pairwise similarity scores with a cosine function. Then we choose the highest 5 similarity scores which are upsampled and normalized into a predicted foreground probability via the softmax operation. Here the foreground probability P and target center estimation map R are mixed via the mixing layer, where we do element-wise multiplication to give a precise matched feature map for upsampling. Fig. 3.6 shows the details of our matching and mixing, in which only the foreground similarity is introduced for illustration.

Segmentation decoder. The matched features of the search area will be fed into the segmentation decoder to generate the final masks. The original features, *i.e.*, $F_S = \{f_S^i, i = 1, 2, 3, 4, 5\}$ from the encoder, are integrated into the decoder via skip connections.

3.3.2 Training

In this section, we describe our training details. Under weak supervision, an effective method to enhance supervision information is to use pseudo-labels. Here, with bounding box-level supervision, we design a three-stage training strategy to gradually improve the quality of pseudo-

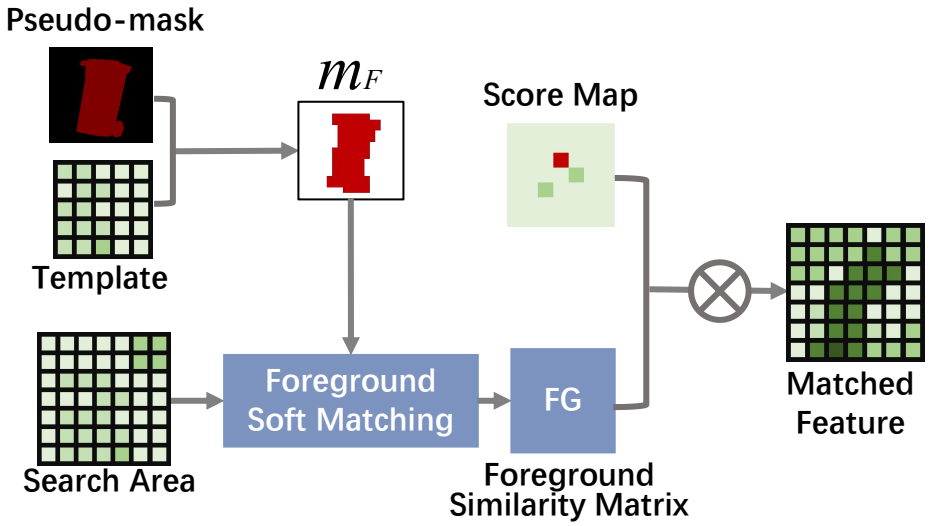


Figure 3.6: Matching and mixing modules.

labels by incrementally adding the modules. The key to achieving such separable and multi-stage training is: the output features of the encoder and input features of the decoder are of the same structure and resolution. This step allows both maps produced by target center estimation and pixel-level feature matching can be seamlessly incorporated into the encoder output. In detail, the encoder output is exactly the same as the decoder input in the first stage, which is modified by the maps of a reference image and further refined in the third stage. Fig. 3.7 shows our training details.

Stage one: generating pseudo masks. In the first stage, the encoder and the decoder will be trained by using single images under the bounding box supervision. Once trained, they can be explored to generate pseudo masks for objects within bounding boxes which can be used in the next training stages. Assume Ω denotes the spatial domain, and Ω_O and Ω_I define the area outside and inside the bounding box, respectively. As for the mask M produced by our model, we can use $\theta(p) \in [0,1]^{\Omega}$ to denote the probabilities of pixel p, where 0 and 1 represent background and foreground, respectively. Following [81], we consider three aspects of learning the encoder and decoder, including the certainty outside the box, the uncertainty within the box, and the global size constraint.

First, since we know that all pixels outside the box belong to the background, the quantity $L_O = -\sum_{p\in\Omega_O} log(1-\theta(p))$ is required to be minimised. In addition, we can observe that,

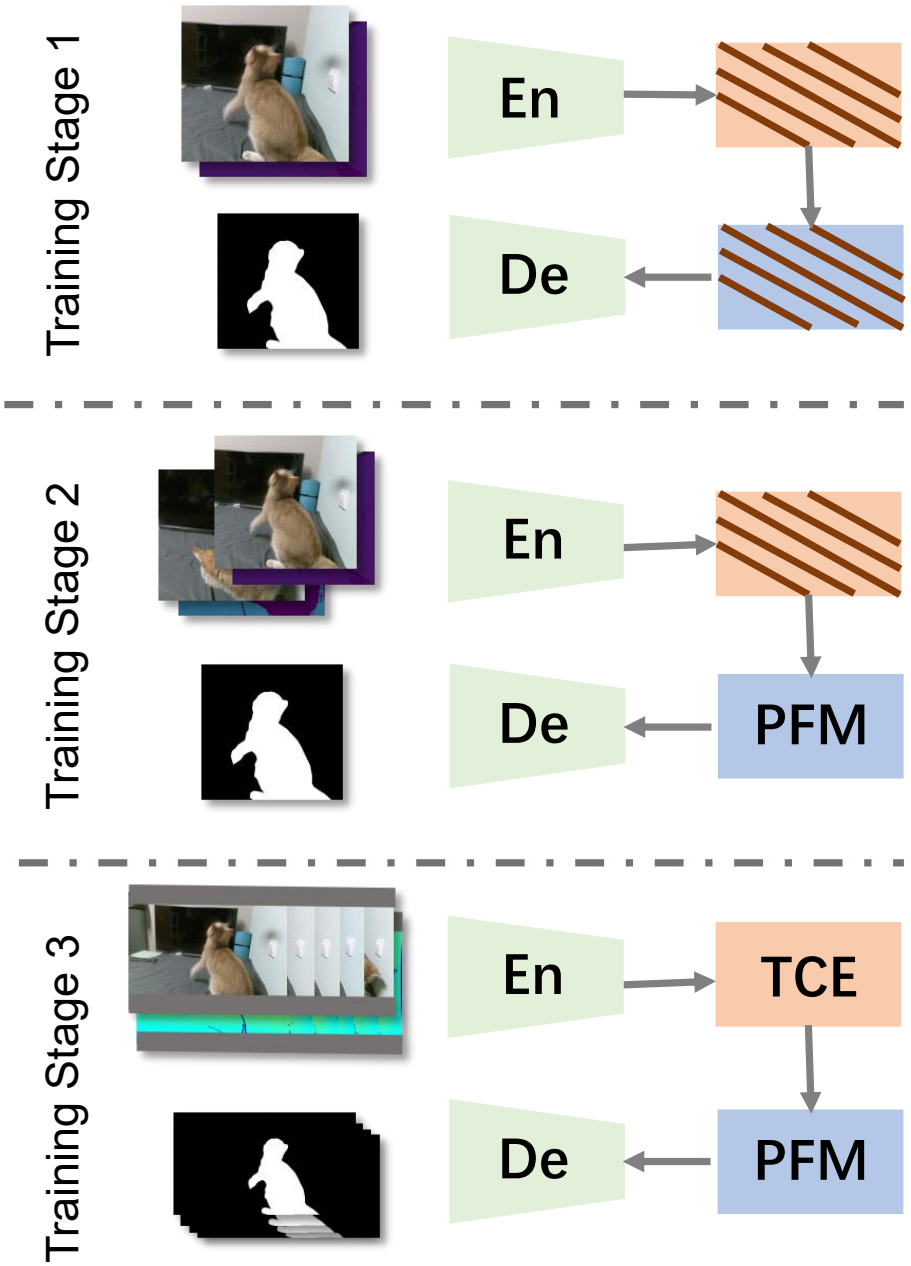


Figure 3.7: Overview of our three-stage training procedure supervised by bounding boxes. In different stages, we provide different kinds of training data to train the individual parts. "En" and "De" denote the encoder and decoder, respectively. "PFM" denotes pixel-level feature matching module, and "TCE" denotes target center estimation module. The modules marked with slashes (/) will be omitted from training at this stage.

generally, each horizontal or vertical line inside the box will cross at least one pixel of the target region. Thus, a tightness prior [93] that the sum of the softmax probabilities for each segment should be greater or equal to its width can be applied. Then, we can apply the first condition $\sum_{p\in s_l}\theta(p) > w, \forall s_l \in S$, where $S = \{s_l\}$ is the set of segments of width w parallel to the sides of the bounding boxes. Third, a region-size constraint can be exploited $\epsilon|\Omega_I| \leq \sum_{p\in\Omega}\theta(p) \leq |\Omega_I|$, where ϵ is a small fraction of the box belonging to the target region. Therefore, the overall objective function in the first stage can be defined as follows:

$$\begin{aligned} \min \quad & L_O = -\sum_{p\in\Omega_O} log(1-\theta(p)), \\ & s.t. \ \ \sum_{p\in s_l}\theta(p) > w, \forall s_l \in S, \\ & s.t. \ \ \epsilon|\Omega_I| \leq \sum_{p\in\Omega}\theta(p) \leq |\Omega_I|. \end{aligned} \tag{3.8}$$

For implementation details, we refer to [81]. The pseudo masks generated by the current encoder and decoder may not be accurate, but better than boxes.

Stage two: learning from references. As we already have the training set with bounding boxes and rough pseudo labels, we release the pixel-level feature matching module to train its parameters. In this stage, the pair of images is randomly selected from an RGBD video sequence in the training set. Being fed the RGBD frame with pseudo-labels, the pixel-level feature matching module will learn the soft matching layer to match the template feature and search area feature. The pixel-wise loss L_{seg} is used to train our models, which is defined as:

$$L_{seg} = L_c(M, E) \tag{3.9}$$

where the binary cross-entropy loss for L_c calculation is used. M is the prediction mask and E is the pseudo-label. The highlighted feature would be the part similar to the reference. After this stage, the fused features are enhanced to be more informative, therefore they can give more reliable pseudo-labels. To generate masks of higher quality for the next training, we randomly select two images of the same target in our training dataset and one will be treated as the reference with a pseudo mask. Then, we run the current models to produce the new mask for another

image until all images have new masks.

Stage three: enhancing by accurate localisation. The purpose of the VOS task is to model the appearance of the target of interest and variations in the scene. Thus, to identify the location of a target under huge variations is important. Therefore, in stage three, we release all modules to train and improve the target discriminative ability. The model is required to locate the object and model the similarities between the current target and the reference. Specifically, an L_2-norm classification error L_{box} in [36] will also be considered. Thus, the overall loss function can be formulated as:

$$L_{total} = L_{seg} + L_{box}. \tag{3.10}$$

Compared with the fully supervised training all masks in a video sequence have been carefully annotated, and our training strategy significantly achieves a weakly-supervised training. Therefore, we needn't use the precise masks from RGBD videos and successfully avoid the labor-intensive labeling work.

3.3.3 Inference

Given a video of both RGB and depth images, the target is generally defined in the first frame and treated as the template. As illustrated in Eq. 3.2, the task is to segment the target in the subsequent frames.

Target initialization. By default, the target is initialized using a bounding box in the first frame. Thus, all pixels inside will be regarded as positive for the target, and then the pseudo-labels generated using our encoder and decoder are treated as the supervision of our model for the following tracking.

Model initialization. Then, RGB and depth images of the first frame will be fed into the encoder. The fused feature $f_S^5(T)$ will be used to initiate the model of target center estimation. According to the annotations, by partitioning pixels belonging to foreground and background, m_F and m_B will be calculated and saved for the matching task.

In the test phase, the RGB and depth images will first be cropped according to the bounding

Table 3.2: Ablation study for key component analysis.

En-De	PFM	TCE	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$
✓			0.528	0.464	0.496
✓	✓		0.653	0.685	0.669
✓	✓	✓	**0.717**	**0.696**	**0.707**

Table 3.3: Ablation study on different target initialization.

Initialization	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$
GT bounding box	0.627	0.596	0.612
GT mask	0.713	0.738	0.725
Pseudo-labels (default)	**0.717**	**0.696**	**0.707**

box in the previous frame. Next, they will be fed into the encoder of our architecture to extract features. Then, by default, the fused feature f_S^5 will be used to estimate the target center score map R. At the same time, by using the features of the template, the corresponding foreground probability P will be used to modify the feature maps. Finally, the output of the decoder will be considered as the final prediction map by default.

3.4 Experiments

3.4.1 Experimental Setup

All experiments are run on a single NVIDIA Tesla V100 GPU with 32GB memory. We use Res2Net-50 [53] as the backbone network, and the encoder for depth has a single input channel. In the training phase, we adopt the Adam optimizer, and the initial learning rate is set to $1e^{-4}$. The network was trained for 40 epochs in every training stage, with 1000 iterations per epoch and 0.2 decay on learning rate every 20 epochs.

3.4.2 Ablation Study

Effectiveness of key components. To investigate the effectiveness of different components, we test them by incrementally adding one at a time. Thus, we have three types of evaluations,

Table 3.4: Ablation study on different input modalities.

Input	$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$
RGB only	0.616	0.640	0.628
Depth only	0.494	0.506	0.500
RGBD (default)	**0.717**	**0.696**	**0.707**

including "En-De"[1], "PFM" and "TCE". We report $\mathcal{J}\&\mathcal{F}$ on our test set in Table 3.2. As shown, the original encoder-decoder can only give a prediction in a single image, which results in very low performance. With pixel-level feature matching, the VOS performance improves by a large margin of 17.3%. By further adding target center estimation modules, we can see that our entire model can reach the best. However, "TCE" does not provide much improvement compared to "PFM", meaning that the reference template plays a more significant role in the VOS task.

Impact of target initialization. Here we analyse the impact of bounding box initialization in the first frame. By default, the pseudo-labels generated using our "En-De" module are treated as the supervision in testing as Sec. 3.3.3. In our experiment, we also test annotations including groundtruth masks and bounding boxes. Table 3.3 shows the compared results. The performance using pseudo-labels is very close to that of directly using groundtruth mask. While the performance of FusedCDNet significantly drops from 70.7% to 61.2% without the pseudo masks, which demonstrates the necessity of our target initialization.

Impact of different modalities. Besides the fused features as in Sec. 3.3.1, we also evaluate the output of original RGB and depth modalities separately. To achieve this, by removing the fusion operation, we modify the encoder to extract unimodal features. Then, each model will be fed only one type of data and retrained accordingly. From Table 3.4, we can see our fused model outperforms others significantly, demonstrating the effectiveness of our cross-modal fusion.

Quality of pseudo-labels. To investigate the performance of pseudo masks, the examples of the generated pseudo-labels after different training stages are shown in Fig. 3.8. Generally speaking, the pseudo-labels from the first stage (2nd row) contain more pixels from the background because no reference is provided in this stage. In contrast, the quality of masks after

[1]"En-De" denotes the encoder-decoder.

Table 3.5: Quantitative comparison of the different VOS methods. **Bold** denotes our method and results.

Method	Initialization		$\mathcal{J}$	$\mathcal{F}$	$\mathcal{J}\&\mathcal{F}$
	BBox	Mask			
RGMP[129]		✓	0.523	0.528	0.525
STM[130]		✓	0.671	0.697	0.684
SiamMask[158]		✓	0.598	0.574	0.586
D3S[112]		✓	0.533	0.481	0.507
LWL[8]		✓	0.702	0.710	0.705
AOT [185]		✓	0.730	0.764	0.747
CFBI+ [186]		✓	0.684	0.713	0.698
EMVOS [28]		✓	0.717	0.735	0.726
HMMN [146]		✓	0.731	0.755	0.743
RDE [101]		✓	0.687	0.716	0.702
RMNet [168]		✓	0.578	0.564	0.571
STCN[26]		✓	0.727	0.753	0.740
TBD [29]		✓	0.683	0.707	0.695
RGMP[129]	✓		0.388	0.360	0.374
STM[130]	✓		0.607	0.620	0.614
SiamMask[158]	✓		0.597	0.598	0.598
D3S[112]	✓		0.539	0.497	0.518
LWL[8]	✓		0.694	0.708	0.701
AOT [185]	✓		0.579	0.523	0.551
CFBI+ [186]	✓		0.548	0.536	0.542
EMVOS [28]	✓		0.546	0.497	0.522
HMMN [146]	✓		0.578	0.541	0.560
RDE [101]	✓		0.553	0.529	0.541
RMNet [168]	✓		0.471	0.417	0.444
STCN[26]	✓		0.600	0.602	0.601
TBD [29]	✓		0.528	0.475	0.502
FusedCDNet	✓		**0.717**	**0.696**	**0.707**

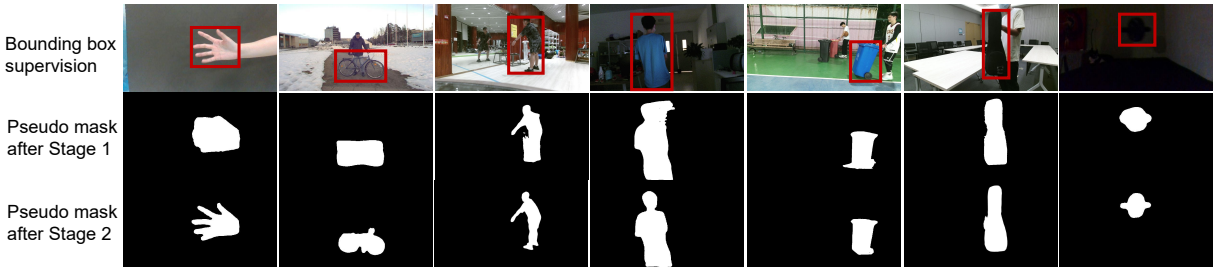


Figure 3.8: Pseudo masks generated after different training stages.

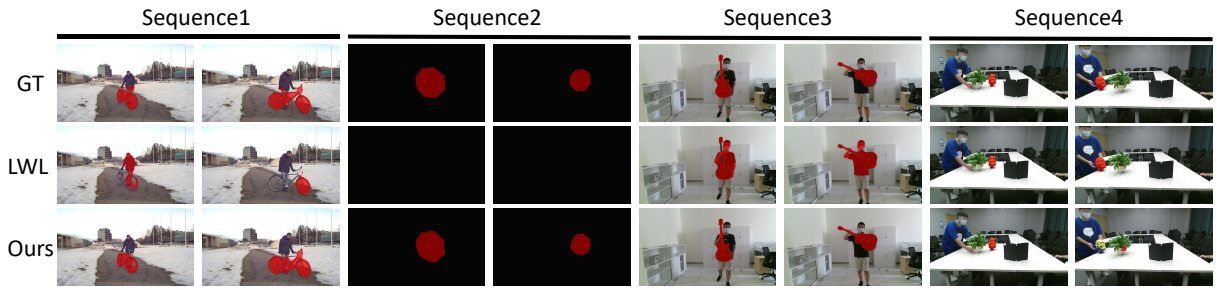


Figure 3.9: Qualitative results of our FusedCDNet compared with LWL [8]. Our approach provides accurate segmentations in very challenging scenarios, including appearance change (Seq. 1), dark scene (Seq. 2), and background distractors (Seq. 3). Seq. 4 shows an example failure case due to the full occlusion by the plant.

training stage 2 can be improved when references are considered (3rd row).

3.4.3 Comparison in VOS Domain

Compared models. As VOS in the RGBD domain remains unexplored, we compare the performance of our proposed baseline with the open-sourced RGB-based VOS methods. Models for comparison include SotA VOS models, *i.e.*, RGMP [129], STM [130], TBD [29], EMVOS [28], RDE [101], AOT [185], CFBI+ [186], HMMN [146], RMNet [168], and STCN [26], and SotA RGB segmentation trackers, *i.e.*, D3S [112], LWL [8], and SiamMask [158]. Note that all of them are fully-supervised models.

Quantitative results. We report $\mathcal{J}$, $\mathcal{F}$ and their average $\mathcal{J}\&\mathcal{F}$ between prediction mask and groundtruth annotations in Table 3.5. As shown, by only using bounding boxes for both training and inference, our method shows remarkable scores compared with the fully-supervised methods. Among previous approaches, LWL [8] obtains the highest overall score of 0.701 thanks to its dedicated bounding box initialization. While our approach outperforms LWL with

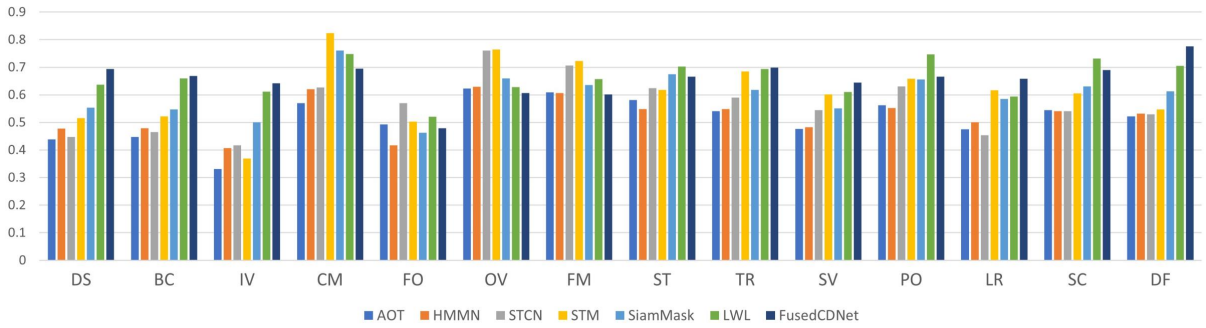


Figure 3.10: Per-attribute performance. We compare our FusedCDNet with the fully-supervised state-of-the-arts with bounding box input.

a relative improvement of 0.6%, achieving an overall score of 0.707. The results demonstrate that our FusedCDNet is effective under the weak supervision paradigm. Besides, we initialize the compared models with a bounding box and groundtruth mask in the first frame for comparison, respectively, to analyse the impact of initial settings. According to Table 3.5, most VOS models, *e.g.*, TBD [29], EMVOS [28], RDE [101], AOT [185], HMMN [146], and STCN [26], have obvious degradation when changing the supervision in the first frame to bounding box, which clearly show that the bounding box initialization is much more challenging. While segmentation trackers, *e.g.*, LWL [8], SiamMask [158], and D3S [112], are not severely influenced after downgrading the initial supervision, since they are usually pre-trained on large-scaled tracking datasets. Our FusedCDNet shows an impressive performance with only bounding box input as it can generate pseudo-masks for the target initialization.

Qualitative results. Qualitative examples are visualized in Fig. 3.9. Here we give the comparison of visualized results under complex scenes between the SotA LWL [8] and our proposed FusedCDNet. As shown, our method can provide satisfactory results even under severe appearance changes, dark scenarios, and background clutter, with only bounding box-level supervision. The state-of-the-art LWL fails in all three given situations. In detail, the challenge in the first sequence is the target rotation, which results in severe appearance change. For the second and third sequences, the challenges come from dark scenes and background clutter, which significantly reduce the discriminative ability of visible features used in LWL. In contrast, our FusedCDNet performs effectively since we focus on both color and depth channels

Figure 3.11: Qualitative results of our pseudo label generation network during inference. Our approach can effectively convert the bounding boxes to masks with effective RGBD fusion.

to utilize both visible and spatial information for segmentation. A failure case of our model is also shown in the 4^{th} sequence, in which our model fails to re-locate the object after occlusion.

Attribute-based analysis. Evaluation under each attribute is reported in Fig. 3.10, in which we compare our proposed method with the fully-supervised SotAs. We can find that our FusedCDNet obviously outperforms other methods under dark scenes (DS), background clutter (BC), illumination variation (IV), deformation, low resolution (LR), and scale variation (SV). We can conclude that FusedCDNet shows outperforming results on depth-related attributes, which validates the effectiveness of the proposed RGBD fusion. Also, it also verifies that depth information can assist segmentation in some color-failed scenarios to better distinguish the objects. Despite that, FusedCDNet obtains comparable performance on the remaining attributes. We also notice that STM [130] performs well on camera motion (CM), while LWL [8] outperforms on partial occlusion (PO) and shape complexity (SC). Overall, FusedCDNet shows impressive performance with only bounding box level weak supervision.

3.4.4 Extension on Tracking Domain

To the best of our knowledge, RGBD tracking is still limited to tracking by bounding box due to high-cost collection and time-consuming annotation of RGBD segmentation data. As an extension, we perform our weakly-supervised RGBD VOS with bounding box annotations to annotate large-scale tracking datasets, aiming to promote RGBD tracking to be fine-grained. In detail, we use our FusedCDNet to annotate large-scale RGBD tracking datasets CDTB [113] and DepthTrack [175], respectively. Some visualized results are given in Fig. 3.11. Compared to the original ground truth bounding boxes on CDTB and DepthTrack, our approach gives high-

quality and more accurate mask annotations, *e.g.*, a backpack in a dark room, a size-changed pedestrian, and a partially occluded mug. A failed case is also given in Fig. 3.11, due to the severe appearance change on the deformable soft belt. The pseudo-annotated tracking sequences can be helpful in improving RGBD tracking to pixel-level accurate tracking.

3.5 Summary

In this chapter, we propose a novel task, RGBD video object segmentation (VOS) under weak supervision. To our best knowledge, this is the first attempt to apply a weakly-supervised paradigm on VOS in the RGBD domain. To this end, we first construct a dataset involving 350 RGBD video sequences for both model training and evaluation. With providing bounding box and mask annotations, this dataset contains common challenges that appeared in RGBD VOS. In addition, we further propose *FusedCDNet* to perform multi-modal VOS with only the supervision of bounding boxes in both training and testing, which achieves weakly-supervised training to overcome the high- cost labeling, cross-modal fusion to handle complex scenes, and weakly-supervised prediction to increase ease of use. Finally, extensive experiments validate that our proposed weakly-supervised *FusedCDNet* can achieve comparative results on RGBD VOS with other fully-supervised models.

Chapter 4

RGBD Object Tracking in 3D Space

The previous chapter successfully achieved the generation of refined target masks by introducing depth information. However, in many tasks, especially robot perception tasks, outputting more refined 3D target bounding boxes is more valuable information. Therefore, in this chapter, a new algorithm will be introduced to achieve the output of 3D object bounding boxes by fusing depth information.

Declaration: The materials of this section have been organized as a paper, which was accepted and published on ECCV 2022 [182].

4.1 Preliminaries

4.1.1 Motivation

Object tracking is to distinguish an arbitrary object from a video, given only the object's location in the first frame. 3D object tracking, which can estimate not only the location but also the 3D size of objects, has a broader spectrum of practical applications involving augmented reality [150], autonomous driving [117], scene understanding [176] and robotic manipulation [144, 121, 182].

However, current state-of-the-art 3D trackers are mostly point cloud-based and highly rely on geometric information to estimate the shape of objects. In fact, LiDAR sensors are quite

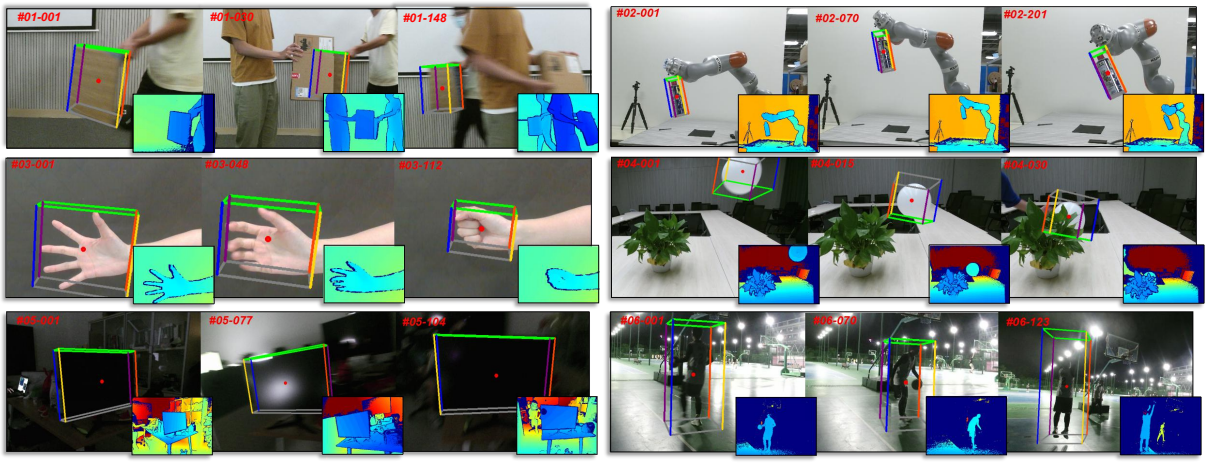


Figure 4.1: Examples of RGBD videos in our benchmark dataset. Each video is annotated with the object's per-frame 3D bounding box. Video sequences are captured towards 3D tracking challenges, *e.g.*, (1) similar objects and occlusion; (2) small-sized object; (3) deformation; (4) symmetric object and partial occlusion; (5) dark scene and camera motion; (6) outdoor scenario.

expensive, and the sparsity and disorder of the point cloud impose great challenges on identifying target objects from backgrounds. Whilst, compared with point clouds, the ignored color cues are more informative for computing appearance features which are widely used to distinguish the target object from backgrounds. In addition, similar to LiDAR, depth information captured by low-cost sensors such as Kinect can also provide geometric information to estimate the shape of targets for most natural tracking scenarios. Moreover, it is easy to get synchronised color channels from such cameras. Even for modeling target appearance, the depth information can be used to resolve tracking failures in cases of, *e.g.,* distractors or rotation [9, 80], due to its insensitivity to the variations in color, illumination, rotation angle, and scale. Therefore, an RGB+D fusion framework is a more reasonable and acceptable solution for 3D object tracking. On the one hand, appearance information in RGB channels and geometry information from the depth channel are two complementary data sources. On the other hand, the 3D coordinate of the object, with the spatial information given by depth information in 3D scenes, is more practical in real-world applications [182].

In addition, current state-of-the-art 3D tracking methods are mostly model-based: the trackers can track the target due to their discriminative ability to recognise targets' categories. For instance, P2B [137] trains the network on human and vehicle data to handle the challenges

dedicated to human and vehicle categories respectively. However, object tracking is in essence a class-agnostic task that should track anything regardless of the object category. Moreover, in autonomous driving applications, the target objects are mostly rigid and placed on the ground so that 3D BBox is set as 4DoF (Degree-of-Freedom) or 7DoF for convenience. As a result, the precise 3D description of arbitrary objects is still unavailable which is desirable for generic 3D object tracking.

To this end, in this chapter, we propose a novel task for 3D object tracking: given the real 3D BBox description of the target object in the first frame of RGBD videos, we aim to estimate the 3D BBox of it in the subsequent frames. To ensure the generic characteristics of object tracking, we collect a diverse RGBD video dataset for this task. The proposed *Track-it-in-3D* contains 300 video sequences with per-frame 3D annotations. The targets and scenarios are designed with a diverse range to avoid the semantic classification of specific targets. Specifically, the 3D BBox is freely rotating to fit the object's shape and orientation, which breaks the limitation of application scenarios. We provide some representative examples in Fig. 4.1. In addition, providing the input of RGB and depth data jointly provides new inspirations on how to leverage multi-modal information. Therefore, we propose a strong baseline, which for the first time realises tracking by 3D cross-correlation through dedicated RGBD fusion [182].

Our contributions are three-fold:

- We propose generic 3D object tracking in RGBD videos for the first time, which aims to realise class-agnostic 3D tracking in complex scenarios.

- We generate the benchmark *Track-it-in-3D*, which is, to the best of our knowledge, the first benchmark for generic 3D object tracking. With dense 3D BBox annotations and corresponding evaluation protocols provided, it contains 300 RGBD videos covering multiple tracking challenges.

- We introduce a strong baseline, *TrackIt3D*, for generic 3D object tracking, which handles 3D tracking difficulties by RGBD fusion and 3D cross-correlation. Extensive evaluations are given for in-depth analysis.

4.1.2 Problem Formulation

In current 3D tracking [137, 201], the 3D BBox is represented as $(x, y, z, w, l, h, \theta) \in R^7$, in which (x, y, z) represents the target center and (w, l, h) represents the target size. There is only one parameter θ indicating rotation because the roll and pitch deviations are usually aligned to the road in autonomous driving scenarios. Notice that any BBox is amodal (covering the entire object even if only part of it is visible). The current 3D tracking task is to compare the point clouds of the given template BBox (P_t) with that of the search area candidates (P_s) and get the prediction of BBox. Therefore, the tracking process is formulated as:

$$Track: \ (P_t, P_s) \ \rightarrow \ (x, y, z, \theta).$$

In most cases, because the target size is fixed, the final output only gives a prediction of the target center (x, y, z) and rotation angle θ [182].

Differing from the existing 3D tracking in point clouds, we explore a more flexible and generic 3D tracking mode. We formulate the new task as:

$$Track: \ B_t \ \rightarrow \ (x, y, z, w, h, l, \alpha, \beta, \gamma),$$

in which B_t is the template 3D BBox given in the first frame, (x, y, z) indicates the target position, (w, h, l) indicates the target scale, and (α, β, γ) indicates the target rotation angle. Specifically, this tracking problem predicts a rotated 3D BBox to best match the initial target.

4.1.3 Comparison with Related Tasks

As shown in Fig. 4.2, we compare our 3D object tracking in RGBD videos with related tasks [55, 148, 175, 154]. Compared to current *3D object tracking in point clouds* [55], we provide corresponding synchronised color information besides point clouds. Furthermore, instead of tracking with (x, y, z, θ), which only describes the location of the target center and one-dimensional rotation, we require a more flexible bounding box to better fit the object. Similarly, *3D object*

Table 4.1: Comparison with related datasets. I=Indoor, O=Outdoor. We are the first dataset that provides 3D annotations for dynamic objects to realise generic 3D single object tracking in natural scenes.

Dataset	Type	Task	Modality	Sequence	Frame	Label	Class	Scenario	Dynamic
DepthTrack[175]	Video	RGBD Tracking	RGB+D	200	294K	2D	46	I,O	✓
SUN-RGBD[148]	Image	3D Detection	RGB+D	-	10K	3D	63	I	×
Objectron[1]	Video	3D Detection	RGB	14,819	4M	3D	9	I,O	×
NOCS[154]	Image	Pose Tracking	RGB+D	-	300K	3D	6	I,O	×
KITTI[55]	Video	3D Tracking	PC	21	15K	3D	8	O	✓
NuScenes[11]	Video	3D Tracking	PC	1,000	40K	3D	23	O	✓
Track-it-in-3D	Video	3D Tracking	RGB+D	300	36K	3D	144	I,O	✓

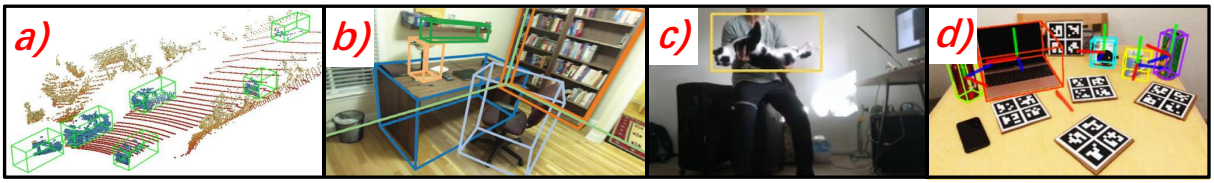


Figure 4.2: Samples from related tasks and corresponding datasets, which basically show the object/scenario/annotation styles. a) KITTI [55], b) SUN-RGBD [148], c) DepthTrack [175], d) NOCS [154].

detection [148] is to classify objects at the image level, which also places all objects on the plane and cannot give a precise description for generic objects *e.g.*, suspended or sloping objects. Compared to *RGBD tracking*, [175] which remains on tracking the object within 2D settings, our proposed task requires a more detailed description of the object in the spatial domain. In addition, *6D pose tracking* [154] focuses on describing the pose of specific objects, which is heavily model-based. Different from existing tasks, 3D single object tracking (SOT) in RGBD videos is more challenging, in which the objects, scenarios, and annotations are more diverse and flexible. A detailed comparison of the proposed *Track-it-in-3d* with representative datasets from related tasks is summarised in Table 4.1. Although the proposed dataset is not prominent in volume compared to existing datasets, it can represent characteristics of 3D tracking more effectively: 1) It achieves a high diversity for class-agnostic 3D tracking by covering indoor and outdoor scenarios, class-agnostic target objects, and freely rotated 3D target annotation. 2) It provides a more effective way to track objects in 3D scenes by providing synchronised RGB and depth information.

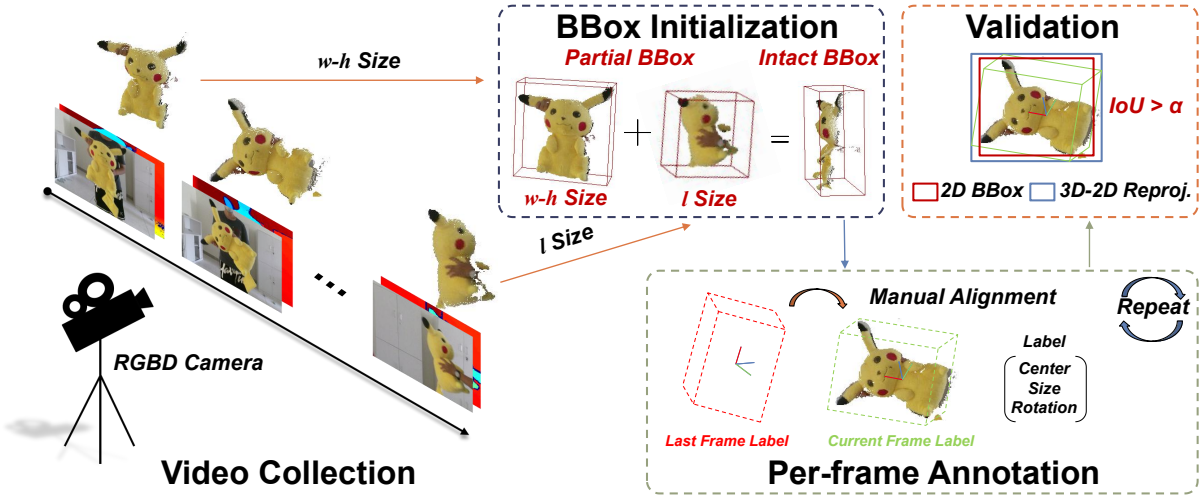


Figure 4.3: Steps of our data annotation strategy. *BBox Initialisation:* We complete the size of the initial BBox from multi-view partial BBoxes. *Per-frame Annotation:* Similar to the tracking pipeline, annotators align the last-frame BBox with the current-frame object and record the label. *Validation:* We re-project the 3D BBox to image and generate 2D BBox. By computing the IoU between the re-projected 2D BBox and with annotated 2D BBox, the accuracy of 3D annotation can be verified.

4.2 Track-it-in-3D: Dataset for 3D Tracking in RGBD Videos

4.2.1 Dataset Construction

Video collection. We collect the videos with *Microsoft Kinect V2* and *Intel RealSense SR300* for different depth ranges. We aim to provide a diverse set of groundtruthed synchronised color and depth sequences for generic 3D tracking, in which diversity is of priority. To this end, we carefully inspect each sequence among all candidate data for the availability and challenge of generic 3D tracking. Examples of some representative sequences are shown in Fig. 4.1. Finally, *Track-it-in-3D* comprises a total of 300 sequences with the data split as such: 250 sequences (32,343 frames) for training, and 50 sequences (6,224 frames) for testing. All the videos are captured at $30fps$. We do not provide a further partition to leave users with the freedom of the training/validation split. We provide the distribution of scenarios and objects in our test set in Fig. 4.4. We keep our test set compact but diverse for a fair and effective evaluation [182].

Attribute definition. Based on characteristics of the aforementioned problem, we annotate all the frames with 9 attributes to analyse how different kinds of challenges influence the tracking

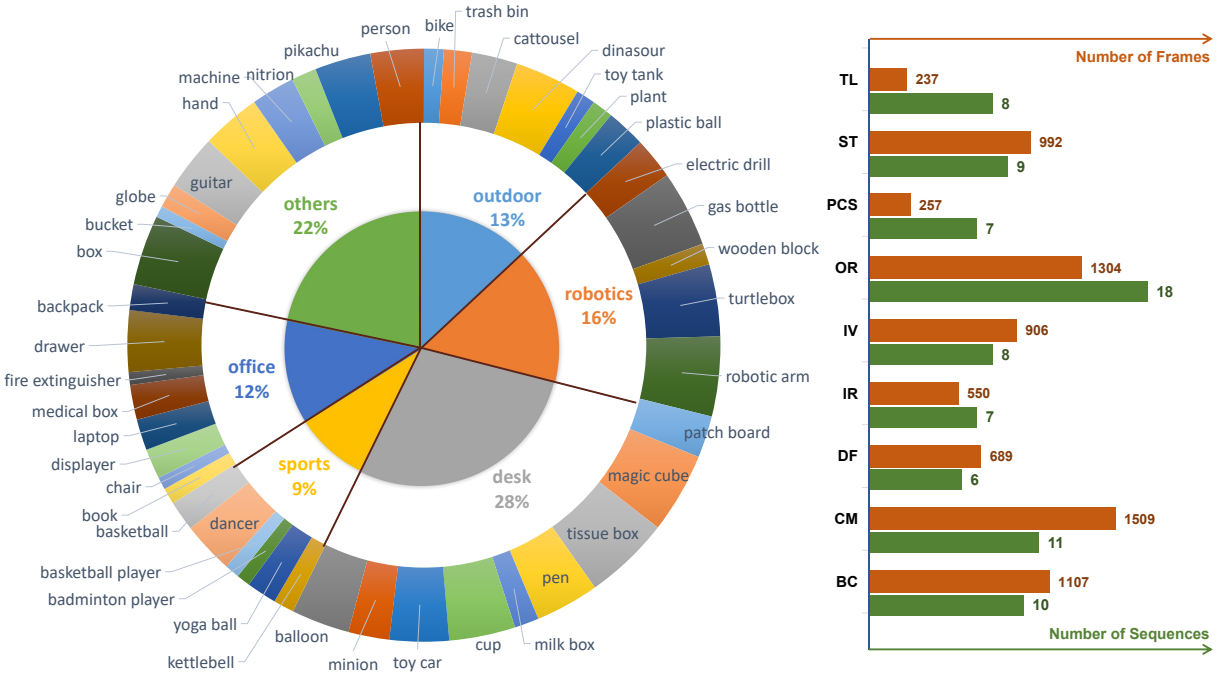


Figure 4.4: Distribution of the object, scenarios, and challenges in all test frames. Left: The inner pie-chart shows the distribution of the scenarios; The outside ring graph shows our target objects. Right: Brown histogram shows the attribute distribution on frame level; Green histogram shows the attribute distribution on sequence level.

performance: Background Clutter (BC), Camera Motion (CM), Deformation (DF), In-plane Rotation (IR), Illumination Variation (IV), Out-of-plane Rotation (OR), Similar Targets (ST), Target Loss (TL) and Point Cloud Sparsity (PCS). Among them, background clutter, similar targets, and illumination variation are closely related to depth favorable scenarios. In addition, point cloud sparsity, in-plane rotation, and out-of-plane rotation are specifically challenging to 3D scenes. Unlike existing attributes in 3D tracking datasets, we are the first 3D dataset to provide detailed visual attributes according to both objects and scenarios. The distribution of attributes is given in Fig. 4.4.

Data annotation. For annotation, we manually annotate each target object in the video sequences with per-frame rotated 3D BBox on our modified version of SUSTechPoints tool [99]. We follow this principle for data annotation: given an initial target description (3D BBox) in a video, if the target appears in the subsequent frames, we will edit the 3D BBox to tightly cover the whole target; otherwise, we will maintain the BBox state from the adjacent frame, and annotate the current frame with a "target loss" label. To guarantee annotation accuracy, we

Table 4.2: Description of attributes in our dataset.

Attribute	ID	Description
Background Clutter	BC	Background has similar colors with the object.
Camera Motion	CM	There are abrupt motions of the camera.
Deformation	DF	The target object is deformable.
In-plane Rotation	IR	Target rotates in-plane.
Illumination Variation	IV	The illumination is too low or high or varies.
Out-of-plane Rotation	OR	Target rotates out-of-plane.
Point Cloud Sparsity	PCS	Point clouds are too few to distinguish the object.
Similar Targets	ST	There are objects similar to the target in the scene.
Target Loss	TL	The target is fully occluded or out-of-view.

adopt a three-stage annotation strategy: 1) *BBox initialisation:* We firstly go through the whole sequences to best describe the target size (w, h, l) and give an initial 3D BBox. For example, we may not get precise length l_p in the first frame, but we can get precise width w_p and height h_p with an estimated length l_e of the target. Then we will go through the whole video to find the frame best showing the precise length l_p of the target, duplicate the 3D box to the first frame, and finally fine-tune the 3D box to get a precise length l_p for the target. 2) *Per-frame annotation:* an annotator edits the initial BBox in the subsequent frames to make the BBox best fit the target; the annotator can change the BBox's location and angle, and size if necessary (for cases like deformable objects) in this stage; 3) *Validation:* the authors finally check the annotation frame by frame to verify the annotation accuracy. The annotation workflow is shown in Fig. 6.4, which ensures high-quality annotation BBoxes in 3D scenes. Under such a strategy, we can obtain the intact target BBox of the target in the specific frame, while it is tightest to fit the object with containing the real target size information in 3D space.

Analysis on 3D Annotation Accuracy. To ensure high-quality annotation, we test the similarity between the projected 2D BBoxes from our 3D annotation and the manually annotated 2D BBoxes. In the projection process, we project the 3D BBox to a 2D plane to give 2D-level annotations, *i.e.*, axis-aligned BBox. The projection from 3D to 2D BBox is implemented by finding the minimum point set and forming a bounding rectangle given 8 corner points. Following the principle that the 2D BBox will tightly fit the 3D BBox according to the associations, we generate the 2D BBoxes from the projection of the 3D BBox. We manually annotate 10% of

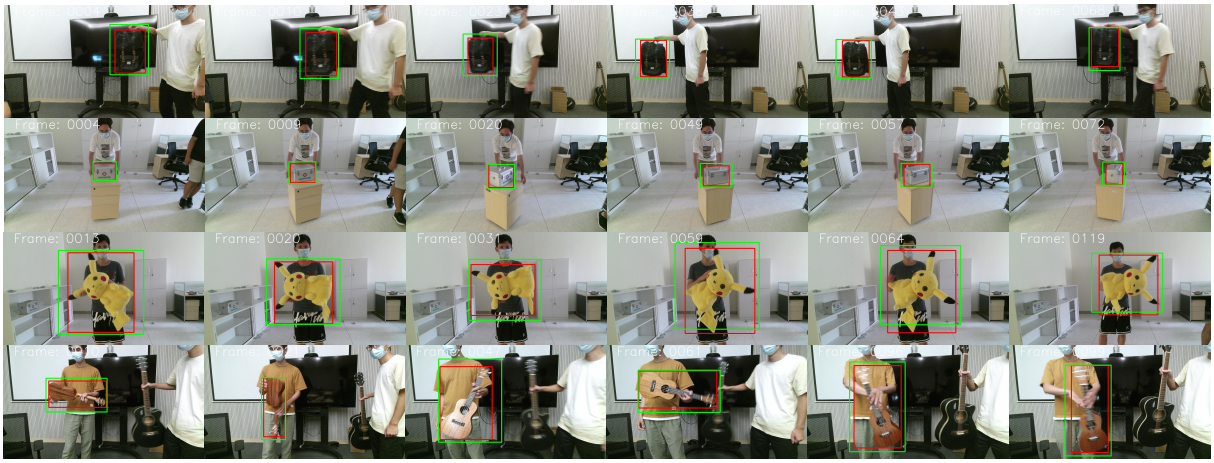


Figure 4.5: Qualitative examples of projected 2D BBoxes from 3D annotation (Green) and manually annotated 2D BBoxes (Red).

the data randomly with 2D BBoxes, which are used to validate 3D annotation accuracy. Then the projected BBoxes will be compared with the manually annotated ones. The visualisation of comparison is shown in Fig. 4.5. We calculate the average projection error and average IoU between them, which is 12.2 pixels and 66%, indicating that our annotated 3D BBoxes are reliable. Besides, the projection enables multiple evaluations and mixed-use on the proposed dataset.

4.2.2 Evaluation Protocols

To judge the quality of 3D tracking, measures are designed to reflect the 3D BBox tracking performance. Therefore, we follow the One Pass Evaluation (OPE) and the standard evaluation protocols to calculate the object center bias and 3D IoU accuracy [182]. In the following, we present our evaluation protocols.

Precision plot. One widely used evaluation metric for object tracking is the center bias, which is used to measure the Euclidean distance between the centers of predicted BBox and groundtruth BBox. We present the precision plots of the trackers averaged over all sequences with the threshold from $0m$ to $0.5m$. We obtain the area-under-curve (AUC) of a tracker's precision plot as its "Precision".

Success plot. For a long time, 3D tracking BBoxes have been set as 7DoF descriptions,

restraining the axis-aligned 3D box. As we propose the rotated 3D BBox description in the 3D tracking scenes, 3D Intersection-over-Union (IoU) is essential to measure the tracking accuracy. According to [40], we provide the IoU measure for general 3D-oriented boxes based on the Sutherland-Hodgman Polygon clipping algorithm. We first clip each face as the convex polygon between the predicted box and the groundtruth box. For two 3D boxes, we first transform both boxes using the inverse transformation, after which one box will be axis-aligned and centered around the origin. Then, each face is clipped as the convex polygon between the predicted box and the ground truth box according to the polygon clipping algorithm. Finally, the IoU is computed from the volume of the intersection and the volume of the union of two boxes by swapping the two boxes, as used in [1]. AUC in the success plot of IoU between groundtruth and predicted BBox is defined as "Success". For more details, we refer readers to [40, 1].

4.3 TrackIt3D: 3D Object Tracking with RGBD Inputs

Sole RGB-based and point cloud-based trackers already exist, and they perform well in specific cases respectively. Here, we propose a generic 3D tracker, namely *TrackIt3D*, which fuses the RGB and depth information in a seamless way. In this section, we first describe the overall network architecture, including the main components, and then illustrate our implementation details [182].

4.3.1 Network Architecture

The input of our network is two frames from an RGBD video, defined as a target template frame and a search area frame respectively. The goal is simplified to localise the template target in the search area per frame. Our network consists of three main modules as shown in Fig. 3.4. We first design a Siamese RGBD Fusion Network to fuse the surface information (RGB Info.) and the spatial information (XYZ Info.) together. Next, the 3D Cross-Correlation Network is proposed to merge the template information into the search area. Finally, the fused feature is fed into the VoteNet module [134] to yield 3D BBox and confidence scores via the proposed BBox

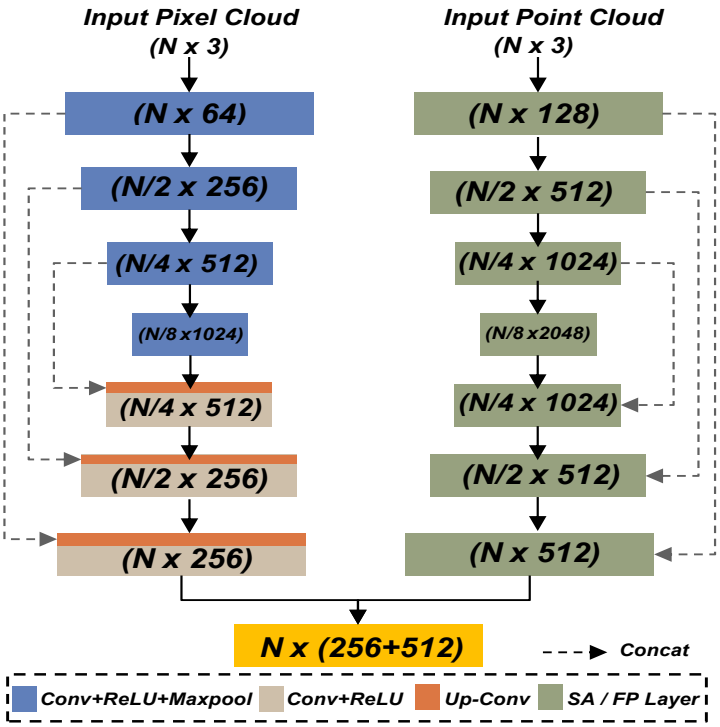


Figure 4.6: The network architecture of our backbone, consisting of Sparse 3D CNN and Pointnet++.

Loss and IoU Loss.

Backbones. The network architecture is shown in Fig. 4.6. The model of Sparse 3D CNN is a 3D U-Net structure, in which the convolution and up-convolution are implemented as sparse convolution [30]. As shown in the left part of Fig. 4.6, the parameters of each convolution layer are the number of input pixels and the size of output features respectively. In the right part of Fig. 4.6, Pointnet++ encoder-decoder takes as input a point cloud to generate dense features. We use four Set Abstraction (SA) layers [136] to downsample and encode the point clouds. We also leverage three Feature Propagation (FP) layers to interpolate and decode features. The parameters of each layer are also the number of input points and the size of output features respectively. Finally, the pixel-wise features and point-wise features are concatenated together and fed into the next module.

RPN module. The Fig. 4.7 illustrates the architecture of the VoteNet-based RPN module. A point-wise MLP (256, 256, 3+256) is applied to the fused feature for object center voting. In addition, another MLP (256, 256, 1) is used to predict a classification score (target or background)

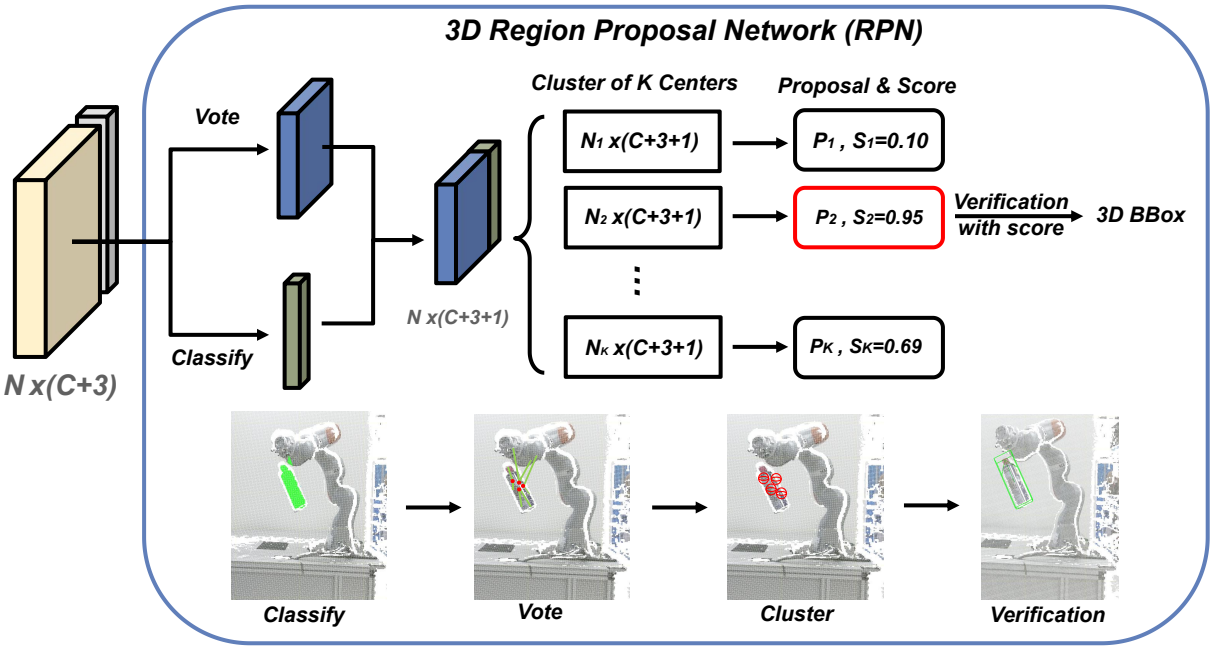


Figure 4.7: Region proposal network (RPN) module in our architecture.

for each point. After that, the predicted vote centers and scores are concatenated together and then clustered into k groups through the furthest point sampling and the ball query. Finally, a mini-PointNet is used to produce the proposals and proposal-wise scores of each group. The proposals and scores are supervised with our annotated 3D BBox and the 3D IoU respectively.

Siamese RGBD fusion network. The key idea of our fusion network is to enable surface information and spatial information to complement each other. To better exploit the spatial information of the depth map, we convert the depth image to a point cloud. Given an RGBD template t and a search region s, our network first associates each point with its corresponding image pixel based on the projection on the image plane using intrinsic parameters of the camera. The obtained pairs P of template and search area are then downsampled to $P^t \in \mathbb{R}^{N_1 \times 6}$ and $P^s \in \mathbb{R}^{N_2 \times 6}$ separately. Every pair P is represented as (x, y, z, R, G, B), in which (x, y, z) indicates the target spatial information and (R, G, B) indicates the surface information. We adopt an encoder-decoder structure with skip connections constructed by sparse 3D CNN [30], to extract the pixel-wise feature map $f^t_{rgb} \in \mathbb{R}^{N_1 \times 256}$ and $f^s_{rgb} \in \mathbb{R}^{N_2 \times 256}$ from the sparse surface pixels. We also implement a variant of the PointNet++ [136] architecture, by adding a decoder with skip connections to generate dense point-wise feature maps $f^t_{xyz} \in \mathbb{R}^{N_1 \times 512}$

and $f^s_{xyz} \in \mathbb{R}^{N_2 \times 512}$. The output feature maps of sparse 3D CNN and Pointnet++ are then concatenated and fed to an MLP network to generate the fused feature maps $f^t \in \mathbb{R}^{N_1 \times 512}$ and $f^s \in \mathbb{R}^{N_2 \times 512}$.

3D cross-correlation network. Learning to track arbitrary objects can be addressed by similarity matching [6]. Following this, our 3D cross-correlation network learns to conduct a reliable similarity between the template features and the search area features. Different from unordered point sets [137], our points are in order because of pixel and point alignment, so that we can do similarity matching directly over 3D feature maps. As shown in Fig. 3.4, after obtaining the fused feature maps of the template and search area, we can compute the similarity map $Sim \in \mathbb{R}^{N_1 \times N_2}$ between f^t and f^s using the cosine distance. The column i in Sim means the similarity score of each feature in f^t to the i^{th} feature in f^s. We then find the top score of i column, which represents the most similar template feature to the i^{th} search feature. After getting all top score indices, we search the template feature by the index in f^t and then concatenate it with the corresponding feature in f^s, yielding a feature map of size $N_2 \times (512 + 512)$. Then we feed it into an MLP network to obtain the final feature map $f \in \mathbb{R}^{N_1 \times 512}$. The point-wise feature map f and the corresponding 3D position of each point are fed to the VoteNet module to obtain the final 3D BBox.

Loss function. We train our network with the following loss function:

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{reg}} + \lambda_2 \mathcal{L}_{\text{bbox}} + \lambda_3 \mathcal{L}_{\text{IoU}}. \tag{4.1}$$

Following [134], a shared voting module is used to predict the coordinate offset between points and the target center. The predicted 3D offset is supervised by Vote loss $\mathcal{L}_{\text{reg}}$, which enforces the network to produce potential centers of the object. BBox loss $\mathcal{L}_{\text{bbox}}$ is designed to pull the K proposal BBoxes closer to the groundtruth BBox. Our 3D groundtruth BBox is defined by $\bar{B} = \left[\bar{x}, \bar{y}, \bar{z}, \bar{w}, \bar{h}, \bar{l}, \bar{q}\right]$, in which quaternion q represents the rotation. The BBox loss

is computed via Huber (smooth-L1) loss:

$$\mathcal{L}_{\text{bbox}} = \frac{1}{K}\sum_{i}^{K}\left\|B_i - \bar{B}_i\right\|_1 . \tag{4.2}$$

IoU loss $\mathcal{L}_{\text{IoU}}$ aims to ensure that the confidence score S_k approximates the IoU between proposals and groundtruth BBox. Following [40], we compute the IoU between the two 3D BBoxes based on the Sutherland-Hodgman Polygon clipping algorithm. The loss function is written as follows:

$$\mathcal{L}_{\text{IoU}} = \frac{1}{K}\sum_{i=1}^{K}\left\|IoU_k - S_k\right\|_1 . \tag{4.3}$$

4.3.2 Implementation Details

Architecture. For our network, we downsample the points and pixels for the template and search area to $N_1 = 512$ and $N_2 = 1024$. The cluster parameter in the VoteNet module is $K = 64$. The coefficients for the loss terms are $\lambda_1 = 1$, $\lambda_2 = 0.5$ and $\lambda_3 = 0.5$.

Training phase. We train our model using the training set which consists of RGBD videos and 3D object bounding box annotations. 1) *Template and Search Area:* we randomly sample RGBD image pairs from all the videos with a maximum gap of 10 frames. In each pair, the first image will serve as the template and the second will be the search area. The template is generated by cropping pixels and points inside the first given 3D BBox and we enlarge the second BBox by 4 times in each direction and collect pixels and points inside to generate the search area. 2) *3D Deformation:* To handle the shape variation of the target, we generate the augmented data for each pair by enlarging, shrinking, or changing some part of the point cloud following [22]. 3) The learning rate is 0.001, the batch size is 50, and Adam [84] is adopted as an optimiser and trained for a total of 120 epochs. The learning rate decreased by 5 times after 50 epochs [182].

Inference phase. During the inference, we also use the proposed dataset in Sec. **??**. Different from the training phase, we track a target across all RGBD frames in a video. The given 3D

BBox will be used to crop the template area, and the search area of the current frame is generated by enlarging (by 4 times in each direction) the predicted 3D BBox in the last frame and collecting the pixels and points in it.

4.4 Experiments

4.4.1 Benchmark Settings

As our proposed *TrackIt3D* is the first tracker designed for generic 3D tracking, we evaluate some representative 3D trackers based on point clouds for comparison. The compared trackers are SC3D[56], P2B[137], and BAT[201]. For model-based 3D trackers, we evaluate their default pre-trained models and the models finetuned on our proposed training set (if the model is trainable). Experiments are run on a single NVIDIA Tesla V100S GPU 32GB.

4.4.2 Benchmark Results

Overall results. Table 4.3 gives the comparison results of 3D trackers. Our method achieves the highest score compared to the existing ones, in terms of both Success (31.1%) and Precision (35.0%). With a dedicated combination of color and depth modalities, TrackIt3D is capable of distinguishing the object in the RGB domain and makes good predictions of 3D BBox in the point cloud domain. It is worth noting that the SC3D, which performs worse on KITTI compared with P2B and BAT, shows better performance on our test set even without finetuning on the proposed training set. The reason is that SC3D aims to compare the similarity between the template and 3D target proposals, while P2B and BAT utilise VoteNet to vote an object center, which tends to learn the center location based on strong category-related priors. We use their car-based model for testing. Therefore, when facing the class-agnostic tracking sequences, the sole VoteNet is not enough for center prediction. The P2B and BAT show remarkable improvements after finetuning on our training set. However, they still suffer low scores because the threshold of the center error is around 0.5m in our proposed dataset, while it is 2m in KITTI

Table 4.3: Quantitative comparison between our method and state-of-the-art methods. Our method outperforms the compared models by a large margin on our *Track-it-in-3d* test set. Speed is also listed and "_ft" means the method is finetuned on our training dataset. **Bold** denotes the best performance.

Tracker	SC3D [56]	P2B [137]	P2B_ft [137]	BAT [201]	BAT_ft [201]	TrackIt3D
Success	9.2%	4.2%	9.4%	2.5%	2.5%	**31.1%**
Precision	6.8%	1.1%	8.4%	0.8%	4.7%	**35.0%**
Speed(FPS)	0.51	23.78	21.25	**28.17**	25.08	6.95

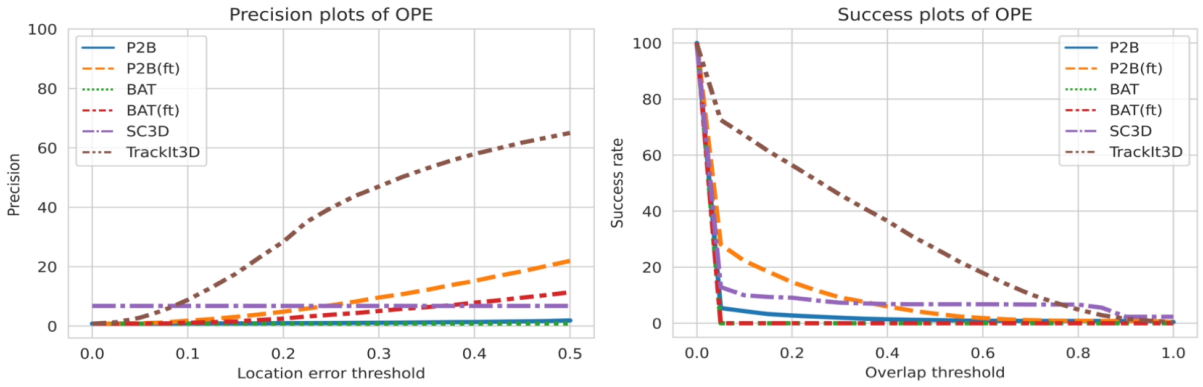


Figure 4.8: The Success and Precision plots of the compared trackers and the proposed *TrackIt3D*.

[55]. In addition, they can only regress an axis-aligned BBox while we get a 9DoF BBox which contributes to a higher IoU score. We show the precision and success plots in Fig. 4.8.

Fig. 4.10 shows several representative samples of results comparing our *TrackIt3D* with finetuned P2B. As shown, unlike P2B which only gives an axis-aligned estimation of the target object, our *TrackIt3D* can also distinguish the target orientation and track the target rotation. Specifically, row a) shows a scene with similar objects, in which P2B fails in total while our method can accurately track the target object. Besides, our method is more robust to challenging cases like object rotation and deformation, as shown in rows b) and c), due to its strong discriminative ability based on RGBD fusion. Moreover, row d) gives an outdoor scenario under low illumination, where it is difficult to locate the object, but our method shows a good estimation. The last row gives a failed case in which the target is severely occluded by a plant, both *TrackIt3D* and P2B fail due to their lack of a re-detection mechanism.

Attribute-based results. Per-attribute results are reported in Fig. 4.9. Although the overall

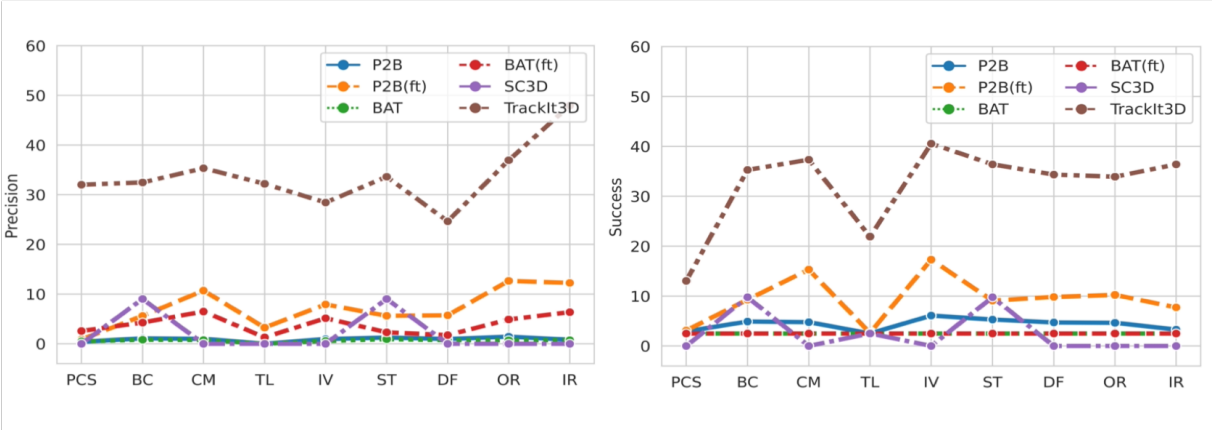


Figure 4.9: Optimal Precision (left) and Success (right) scores over the visual attributes.

performance is low, we can obtain an informative analysis from the per-attribute result. Our method obviously outperforms the compared models in all attributes, especially in in-plane rotation and illumination variation. Clearly, the superior performance of our RGBD fusion over point cloud is evident. However, *TrackIt3D*'s success score degrades severely on the point cloud sparsity and target loss, indicating that it still needs improvement on long-term discriminative ability and target localisation under little spatial information. Despite that, it is worth noting that the finetuned P2B performs well under in-camera motion and illumination variation, while SC3D beats the other trackers on background clutter and similar targets [182].

Fig. 4.11 shows the additional results of the comparison to finetuned P2B [137]. These results demonstrate that our method is more robust to several difficulties. The first sequence shows our TrackIt3D's robustness to background clutter. The 2^{nd} and 3^{rd} sequences show that our method can handle the problem of our-of-plane rotation, thanks to the freely rotated 3D BBox which can better describe the objects compared to the rotation-unable P2B. In the last sequence, our method can track the turtlebot under fast motion and camera motion, demonstrating the effectiveness of our proposed model compared to P2B_ft.

4.4.3 Ablation Study

Effectiveness of RGBD fusion. To validate the effectiveness of the proposed RGBD fusion on 3D tracking, we apply it on P2B and BAT instead of their original heads and obtain corresponding

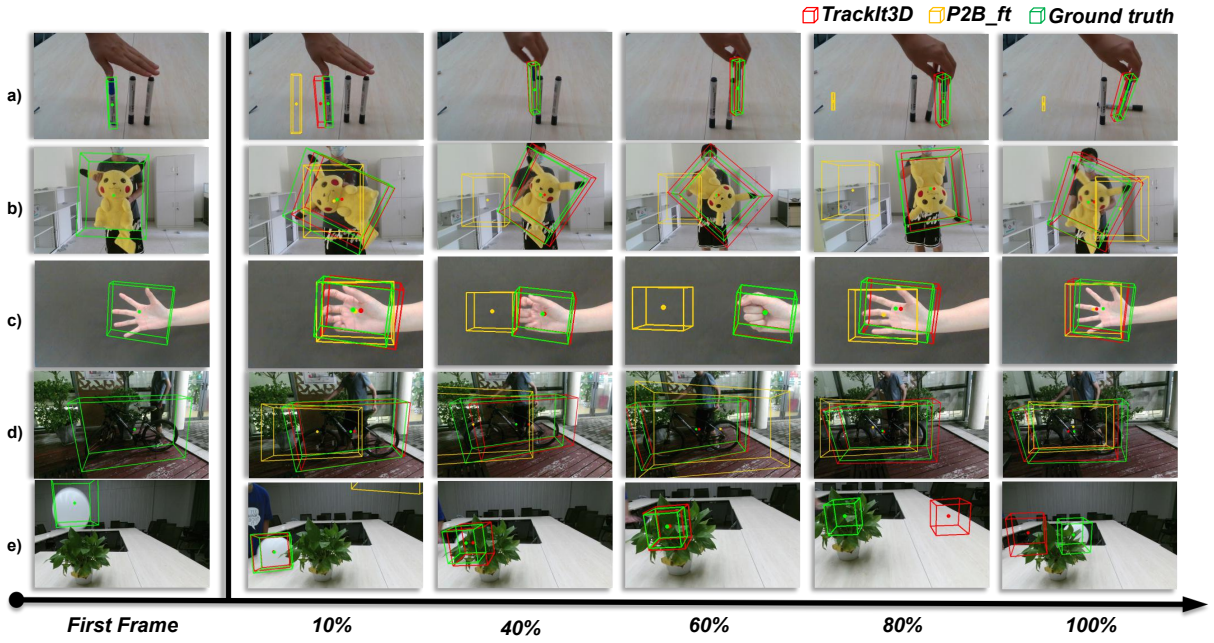


Figure 4.10: Qualitative results of our baseline *TrackIt3D* compared with the fine-tuned *P2B*. We can observe our baseline's advantage over P2B in many challenge scenarios, *e.g.*, a) similar objects, b) rotation, c) deformation, and d) dark scene. The last row is a failed case when the object is fully occluded.

Table 4.4: Performance of the RGBD variant of original 3D point cloud tracker, and P2B++ and BAT++ have been finetuned on our training dataset.

Tracker	P2B++ [137]	BAT++ [201]	TrackIt3D
Success	24.5%	18.1%	31.1%
Precision	28.2%	26.0%	35.0%

variants P2B++ and BAT++. Table 4.4 shows the comparison between the variants with the RGBD fusion head and our *TrackIt3D*. Specifically, there are striking improvements (at least 15.1% and 19.8%) in terms of Success and Precision compared with the finetuned P2B and BAT, which proves that the RGBD fusion boosts the performance of point cloud voting models. Also, the performance of BAT++ is lower than the P2B++ due to its strong object prior with a fixed size.

Different ways for 3D cross-correlation. Besides our default settings in Sec. 4.3.1, we consider other possible ways for 3D cross-correlation, *e.g.*, 2D correlation [105], which is commonly used in 2D tracking, instead of 3D correlation. The left section in Fig. 4.12 shows how we implement 2D correlation. Surprisingly, the results in Table. 4.5 show that the 2D

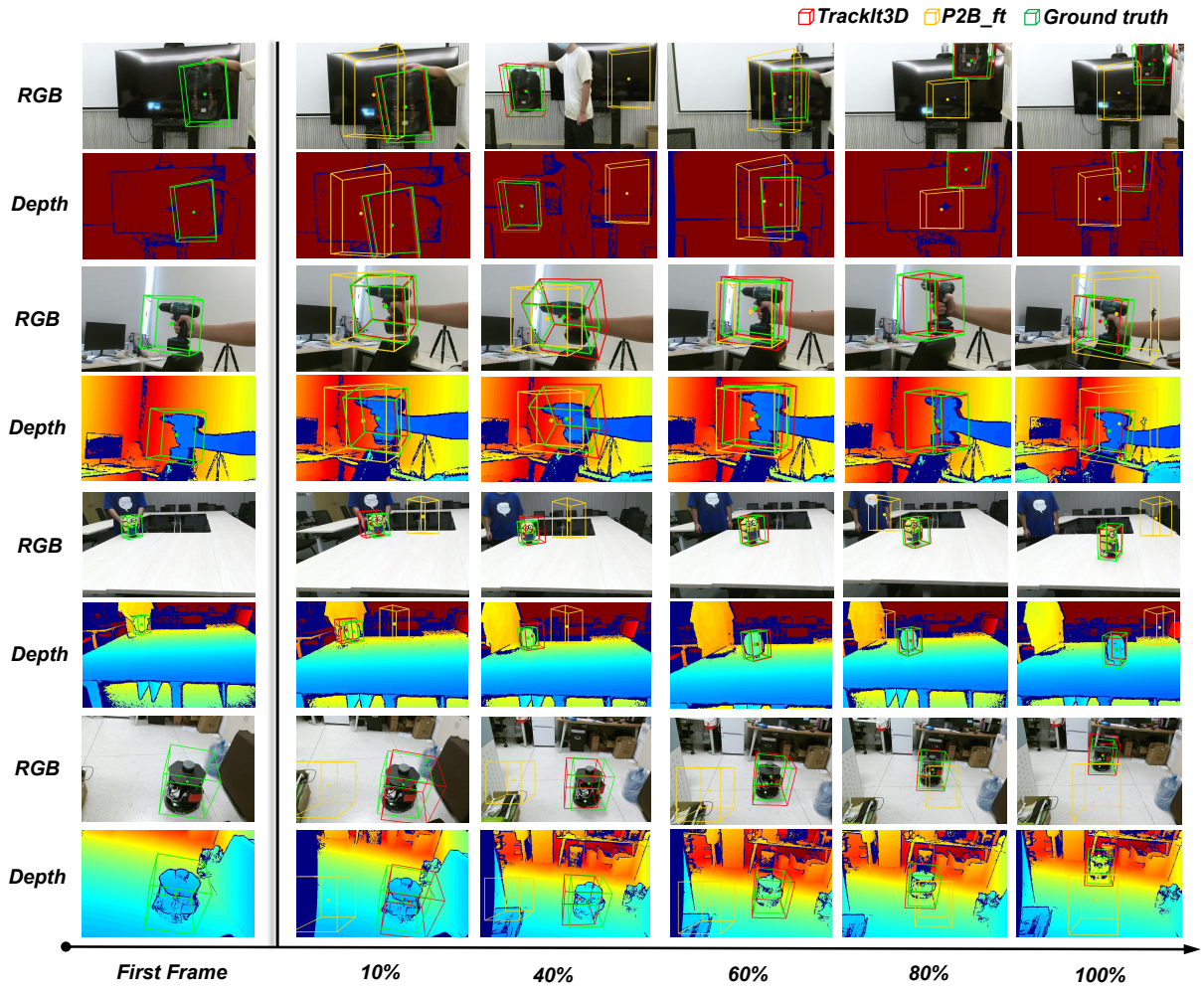


Figure 4.11: More qualitative results of our baseline *TrackIt3D* compared with the fine-tuned *P2B*.

correlation setting outweighs our 3D correlation on Precision, although it gives a lower Success. This may reveal that the 2D-based method is more robust in estimating an accurate target center, while it is weaker on 3D BBox prediction as it omits the spatial correlation in 3D space. We also try to remove the similarity map and template feature, as shown in the right part of Fig. 4.12. The performance degrades without using the two parts. Specifically, once removing the template feature, Success and Precision degrade with 7% and 5%, which proves that the tracker loses the discriminative ability without the reference feature.

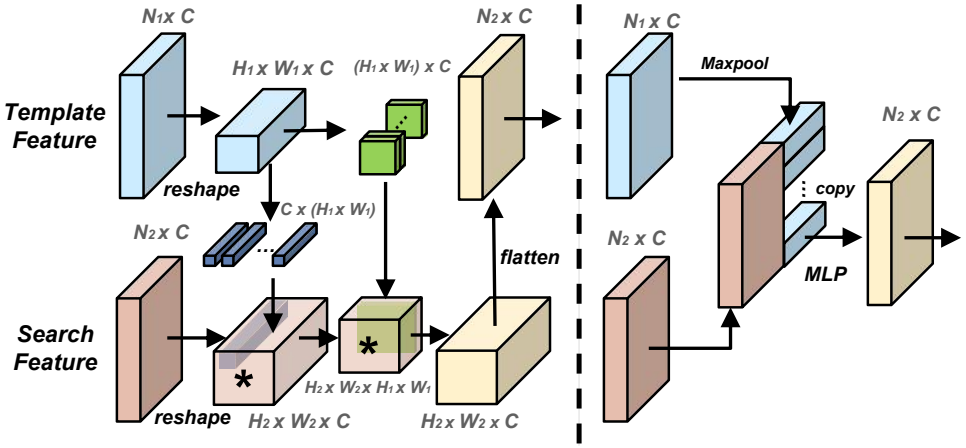


Figure 4.12: Different ways for 3D cross-correlation. The left part follows the 2D tracking pipeline. The right part is without calculating the similarity map. $*$ means convolution operation.

Ways for 3D xcorr.	Success	Precision
our default setting	**31.1%**	35.0%
w/ 2D xcorr. setting	28.3%	**38.4%**
w/o similarity map	30.9%	33.1%
w/o template feature	7.0%	5.0%

Table 4.5: Different ways for 3D cross-correlation (xcorr.). Methods for similarity learning between search features and template following 2D tracking method are illustrated in Fig. 4.12.

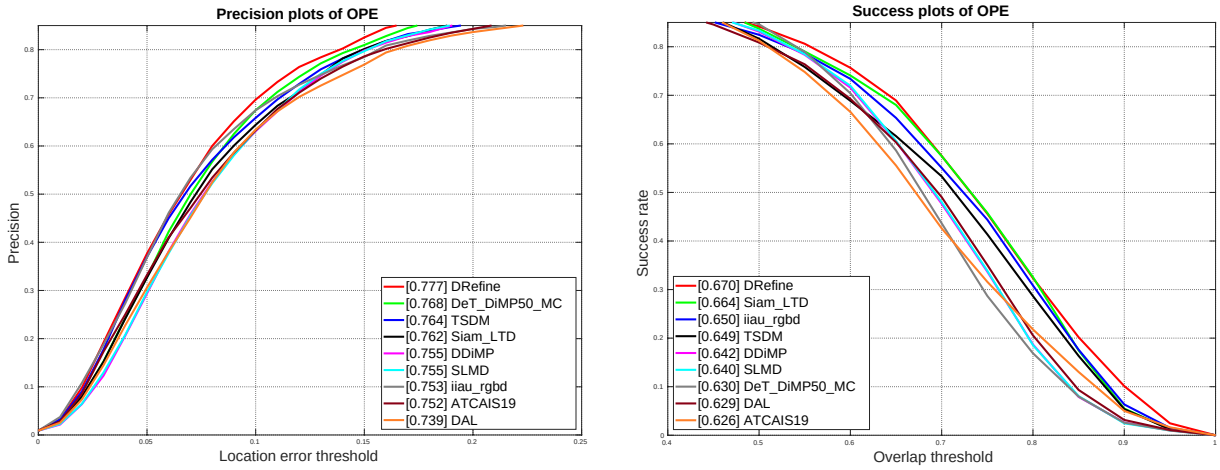


Figure 4.13: Precision and success plots of evaluated trackers on our dataset with 2D settings. Compared trackers: DeT [175], TSDM [200], DAL [138], DRefine [86], iiau_rgbd [86], SLMD [86], DDiMP [85], Siam_LTD [85], ATCAIS [87].

4.4.4 Extension on RGBD Tracking

With the projection from 3D BBox to 2D BBox, we can evaluate existing RGBD trackers on our proposed dataset. Fig. 4.13 shows the precision and success plots of state-of-the-art trackers. Here we use the projected 2D BBoxes as the groundtruth. The precision plots compute the Euclidean distance between the target center location and the labeled groundtruth position of the current frame. Different trackers are ranked with this metric on a threshold (20 pixels). Since the precision metric is sensitive to the target size and image resolution, we use the normalised precision [127]. With the normalised precision metric, we rank precision scores using the Area-Under-Curve (AUC) between 0 to 0.5. The success plots calculate the Intersection-over-Union (IoU) between predicted BBox and groundtruth BBox. The tracking methods are ranked using the AUC between 0 to 1. Here we test RGBD state-of-the-art trackers on this dataset, including high-performance trackers from VOT-RGBD challenge, *i.e.*, DRefine [86], iiau_rgbd [86], SLMD [86], DDiMP [85], Siam_LTD [85], and ATCAIS [87], and advanced RGBD trackers, *i.e.*, DeT [175], TSDM [200], and DAL [138]. As shown, state-of-the-art trackers can achieve 60% to 70% on precision and success, indicating that our dataset is less challenging on 2D tasks if we only "track the object on the plane". Note that existing state-of-the-arts only obtain precision and success scores of lower than 30% on our proposed 3D task. Therefore, it is more difficult for a tracker to predict target description in 3D scenes rather than 2D BBoxes [182].

4.5 Summary

In this chapter, we investigate a novel topic to track generic objects with 3D rotated BBox in RGBD videos. We first construct a novel benchmark *Track-it-in-3D* with 300 RGBD videos for training and testing, which covers diverse objects and challenging scenarios in 3D scenes. Also, this benchmark enables generic 3D tracking in complex scenarios with novel target annotation and performance evaluation. Furthermore, we propose an end-to-end method *TrackIt3D* for tracking class-agnostic 3D objects. With effective RGBD fusion and 3D cross-correlation, our baseline shows superior performance on this challenging task. We hope this work will facilitate

further research on generic 3D tracking.

Chapter 5

Training-efficient RGBD Tracking

The previous two chapters solved the problem of finer-grained output and achieved a more accurate description of the target state, making the multi-modal tracking algorithm with a wider range of application scenarios. However, the efficiency of model construction and prediction also seriously restricts the scope of application. Therefore, this chapter will first introduce a very efficient modal fusion scheme to achieve the construction efficiency of multi-modal trackers.

Declaration: The materials of this section have been organized as a paper, which was accepted and published on ACM MultiMedia 2022 [179].

5.1 Preliminaries

5.1.1 Motivation

Unlike the success of color-based object tracking, we note that there exists a significant development gap in multi-modal tracking, especially RGBD tracking. High-performing data-driven models are lacking for multi-modal tracking, due to the deficiency of large-scale training datasets. For example, a comparison among RGB tracking datasets and multi-modal tracking ones is shown in Table 5.1, clearly showing that the multi-modal tracking datasets are orders of magnitude smaller than the RGB counterparts. Also, the resolution of multi-modal data is relatively smaller than the RGB ones. In addition, multi-modal datasets also suffer from

Table 5.1: Dataset comparison between RGB tracking datasets and multi-modal ones. "M" denotes million. "Resolution" indicates the maximum resolution.

Dataset	Year	Modality	Videos	Frames	Resolution
LaSOT[42]	2019	RGB	1400	3.52M	1280×720
GOT-10k[70]	2019	RGB	10,000	1.5M	1920×1080
TrackingNet[127]	2018	RGB	31,000	14M	1280×720
DepthTrack[175]	2021	RGB+D	200	294,600	640×480
VisEvent[160]	2021	RGB+E	820	371,127	346×260
LasHeR[97]	2021	RGB+T	1224	734,800	960×576

low-quality problems like desynchronization and misalignment between different modalities or sensors [179].

To handle the data-hungry bottleneck, compared to manually collecting the multi-modal data for deep model designing, a more efficient way is to best exploit the state-of-the-art pre-trained RGB tracking models. Current state-of-the-arts in multi-modal tracking already have many attempts on it mostly with the "pre-trained RGB baselines + multi-modal data fine-tuned" paradigm. In RGB-D tracking, trackers use pre-trained RGB trackers as strong baselines and show promising performance [175]. In the VisEvent benchmark, Wang et al. [160] proposed a series of baselines by embedding event information into RGB trackers and re-trained on its training set. However, these models can only learn a small amount of new data with modality gaps, resulting in incomplete learning of the cross-modal fusion mechanism. Further, inevitably, such a modality gap will confuse models that were originally well-learned on large-scale RGB datasets. Even though we can have large-scale cross-modal datasets in the future, current research clearly illustrates that over-training on a new dataset also potentially causes a knowledge-forgetting problem. Therefore, a question is raised: how can we effectively utilize both the large-scale RGB knowledge and the complementary information from non-color modalities?

To achieve high-performing multi-modal tracking more efficiently, we explore a different route in this paper. Firstly, we observe that there are visible and auxiliary modalities in multi-modal tracking. Tracking performance highly depends on the models' discriminative ability in color views, which can be gained from large-scale data training, while auxiliary modalities

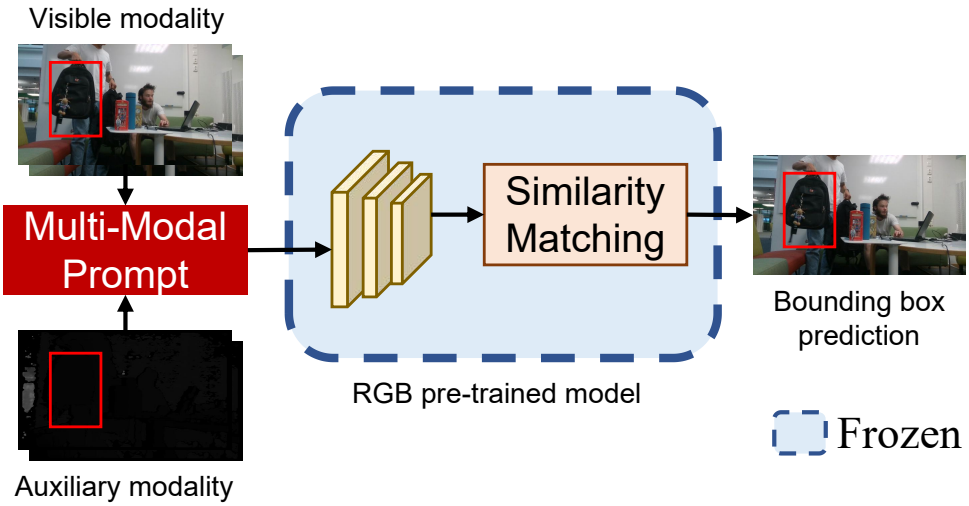


Figure 5.1: How our multi-modal prompt works. Given a frozen, pre-trained RGB tracker, we expect it can perform well on multi-modal tracking tasks with only a modality-agnostic prompt on the test videos.

are effective in specific scenarios. Moreover, due to the data sources/sensors, the auxiliary modality is less informative. Inappropriately importing auxiliary modalities may receive more kicks than halfpence. Drawing inspiration from the recent advances in prompting in Natural Language Processing (NLP), we therefore ask, can we still use auxiliary modalities but without the inappropriate fine-tuning process? To this end, we propose a new simple but efficient prompt method for multi-modal object tracking tasks, namely multi-modal Prompt Tracker (ProTrack). Instead of altering or fine-tuning the pre-trained model itself, we directly modify multi-modal inputs to adapt the state-of-the-art RGB trackers as shown in Fig. 5.1. The comparison between our ProTrack and the existing three types of fusion strategies for multi-modal tracking methods is shown in Fig. 5.2. Regardless of the stage at which classical methods fuse multiple data, there are two main differences compared to our methods. On one hand, these models have to be fine-tuned on new datasets for new tasks, while our model can be merely trained once on RGB videos. On the other hand, since these models are fine-tuned for only one type of data, they cannot be used in different multi-modal scenarios, while our one model can handle multiple tasks, simultaneously [179].

By choosing appropriate prompts, we can manipulate model behavior so that the pre-trained model itself can be used to predict the desired output without any additional modality-

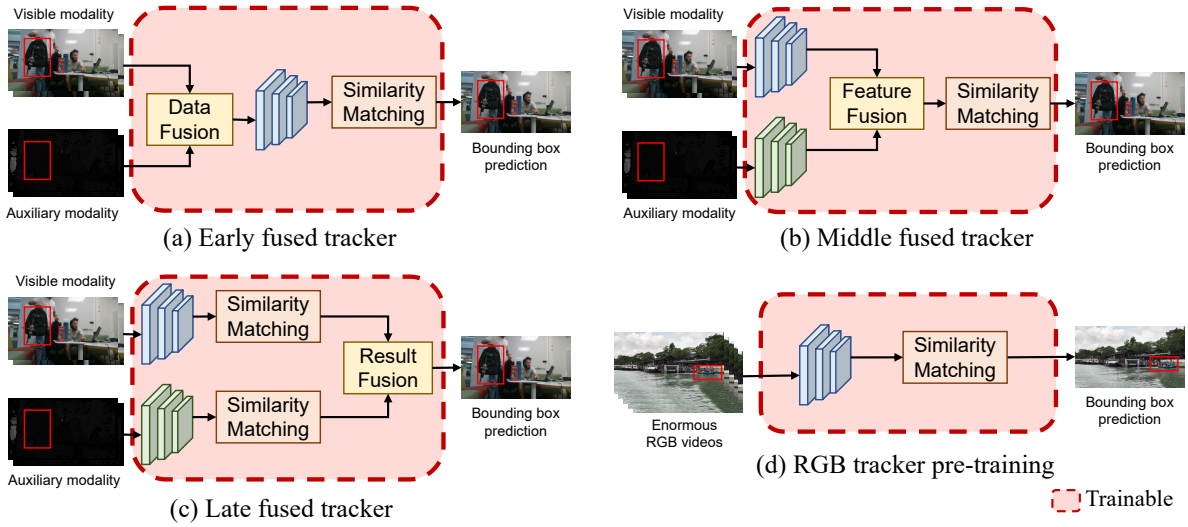


Figure 5.2: Comparison between our proposed method (d) and the existing ones (a, b, and c) during the training process.

specific training. Unlike efforts on vision-and-language tasks with textual prompts, we introduce the tracking applications of this promising paradigm with multi-modal visual prompts. By transferring the multi-modal data to colored ones, the invisible auxiliary modality information can be leveraged by RGB trackers as color-based referential markers. We adopt the prompt engineering in 5 multi-modal object tracking benchmarks. Enormous experiments verify its effectiveness. Our contributions are three-fold:

- We present a novel prompt learning paradigm for multi-modal tracking, in which both the large-scale RGB knowledge and the complementary information are effectively utilized. To the best of our knowledge, this is the first attempt at prompt learning in multi-modal tracking areas.

- We present a principled solution to cross-modal prompt configurations for various kinds of multi-modal tracking but without the inappropriate fine-tuning process.

- We unify different multi-modal object tracking tasks into a prompting framework and conduct comprehensive experiments on different scenarios that demonstrate the effectiveness of ProTrack.

5.1.2 Prompt Background

Prompt learning. "Pre-train, prompt, and predict" paradigm is replacing the "pre-train, fine-tune" one in NLP area. In the prompt paradigm, downstream tasks are reformulated to look more like the solved ones with the help of textual prompts [107]. Instead of adapting pre-trained language models to downstream tasks via fine-tuning, prompting learning can reach high performance even in the few-shot or zero-shot settings. Extending from NLP, Pre-trained vision-language models also show promising capabilities in many tasks. CLIP [139] is trained from scratch with 400 million image and text pairs, which are collected from the internet, and evaluated on 30 different computer vision datasets. Despite CLIP being fairly new, multiple works in various research fields have emerged. Zhou *et al.* [204] proposed context optimization (CoOp) which learns continuous soft prompts that perform well for downstream tasks. CLIP-Adapter [51] is then proposed to conduct fine-tuning with feature adapters on both visual and language branches.

Prompting for vision tasks. There have been preliminary attempts at prompts with images in vision tasks. CPT [187] converts visual grounding into a fill-in-the-blank problem by creating visual prompts with color-based co-referential markers in both images and text. Very recently, Jia *et al.* [75] proposed VPT which applied prompting learning to vision backbones on 24 classification tasks. Bahng *et al.* [5] proposed visual prompting which learns a task-specific image perturbation to adapt the pre-trained models to downstream tasks. With only changing a few pixels, the visual prompting shows surprising effectiveness. To the best of our knowledge, there are no prompt paradigms designed for semantic-agnostic multi-modal vision tasks [179].

5.1.3 Problem Formulation

Prompt formulation. In the classical paradigm of NLP, in general, models are first pre-trained on large-scale datasets and then fine-tuned on downstream data to adapt to new tasks. The data and supervision are often different between the two stages, thus leading to forgetting or under-fitting problems. Instead of updating the model, a new prompt paradigm is proposed,

Table 5.2: Terminology and notation of our proposed multi-modal prompting methods.

Name	Notation	Description
Input	$X = \{V, A\}$	Multi-modal frames.
Prompt function	$f(X)$	A function that converts the input into a specific form.
Prompt	$X' = f(X)$	Prompted RGB frames generated by the prompt function.
Model	$tracker\{X'\}$	A pre-trained model for object tracking with input X'.
Output	B	Predicted target bounding box.

using text templates to modify the data to fit the input of the pre-trained model. The reason to fix these models is that they have learned at scale, seeing more concepts that cannot be provided by downstream datasets. The key component of prompting is how to design the templates (textual prompts), which can reduce the distribution gap between the pre-trained data and downstream data.

Intuition. Similarly, the key to our multi-modal prompts is to reformulate multi-modal tracking into a single-modal tracking problem. To this end, our ProTrack establishes fine-grained connections between single-modal videos and multi-modal videos. Table 5.2 shows how we define the prompting method in multi-modal tracking. The multi-modal input $X = \{V, A\}$ is converted to $X' \in \mathcal{R}^{3\times H\times W}$ after the prompt function $f(X)$, where H and W are the height and width of the input image, respectively. Here we denote visible modality as V and auxiliary modality as A.

In ProTrack, the pre-trained model is from the RGB tracking area while the downstream tasks are multi-modal object tracking tasks. Specifically, the ProTrack framework consists of two components: 1) **a multi-modal prompt** that transfers the multi-modal video sequences into visible single-modal ones; 2) **a pre-trained model** $tracker\{X\}$ that has a strong discriminative ability in the visible tracking area. We then illuminate how we generate multi-modal prompts and reformulate the tracking problems.

5.2 ProTrack: Efficient Multimodal Tracking by Prompt Learning

5.2.1 Multi-Modal Prompt Design

Given that the prompt specifies the task, choosing a proper prompt has a large effect on not only the accuracy but also which task the model performs in the first place.

Pre-trained tracker. In this work, given a multi-modal video in multi-modal tracking, we usually have the Visible Modality (RGB, grayscale) and Auxiliary Modality (*e.g.*, depth, event, or thermal). The original multi-modal tracking process can be formulated as:

$$tracker : \{X_t, X_1, B_1\} \rightarrow B_t \tag{5.1}$$

where B_t denotes the predicted bounding box and B_1 is the bounding box supervision in the first frame. Thus, as shown in Fig. 5.2 (a,b,c), the model $tracker$ will be trained using multi-modal data $X = \{V, A\}$.

Instead, as shown in Fig. 5.2 (d), our model $tracker$ is trained merely using existing large-scale RGB data to associate the objects in different frames. Thus, the input size of the tracker is fixed to $3 \times H \times W$. Therefore, no fusion module is required, neither in the training stage nor in the testing stage. We consider a variant of the spatial-temporal transformer [173] as the baseline tracker. For more details, please refer to Sec. 5.3.2. The intuitive consideration is that RGB videos are more readily available than collecting multi-model datasets, so we have already had more RGB videos. The immediate benefit of using an RGB tracker is that the tracker has seen many challenges in learning from big data. If we can transfer other challenges to these already-seen challenges by introducing new modality data, the association performance will be improved further [179].

Prompt. Obviously, if we want to use the pre-trained models without any updates, we need to transfer the multi-modal data $X = (V, A)$ into the solution $3 \times H \times W$. With only modifying

the trackers' input, our multi-modal prompt function can be formulated as:

$$f(V_t, A_t) = \lambda * Color(A_t) + (1 - \lambda) * Color(V_t), \tag{5.2}$$

where λ is a parameter. $Color(*)$ denotes the dyeing function of different modalities. If the data V has one channel, the result $Color(V)$ will have three channels. But if the data V is the RGB image, the operation $Color(V)$ does nothing. Equipped with multi-modal prompts, it is then straightforward to apply pre-trained RGB trackers on prompted videos, without any extra training:

$$tracker : \{f(V_t, A_t), f(V_1, A_1), B_1\} \rightarrow B_t. \tag{5.3}$$

It is worth noting that the pre-trained RGB tracker has not seen any multi-modal data.

5.2.2 Why Multi-Modal Prompt Works?

Compared with a special design to read and understand different channels, our solution is more straightforward. By dyeing salient colors or highlighting the auxiliary modality on brightness, the prompting in our ProTrack can reach state-of-the-art performance. In summary, the following three aspects clearly explain why the prompting strategy is very effective for such types of tasks.

Color image is more informative. Conventional cameras can capture informative images and videos. Multi-modal information is proposed for vision tasks at first due to their sensitivity to specific activities. At the same time, the information provided by the sensors is more focused and less informative. In existing works, researchers expect the visible modality and auxiliary modality from two sensors can cooperate with each other to achieve more reliable object tracking. However, inappropriate fusion or combination between different modalities may damage the information volume, which finally has side effects on tracking performance. The key to exploiting multi-modal information is preserving the visible information at most and embedding the useful parts of the auxiliary information at the same time. Thus, the prompt paradigm then provides us with a new perspective to adapt the tasks to the pre-trained models, in which our design will not damage the discriminative ability of the original frameworks. In other

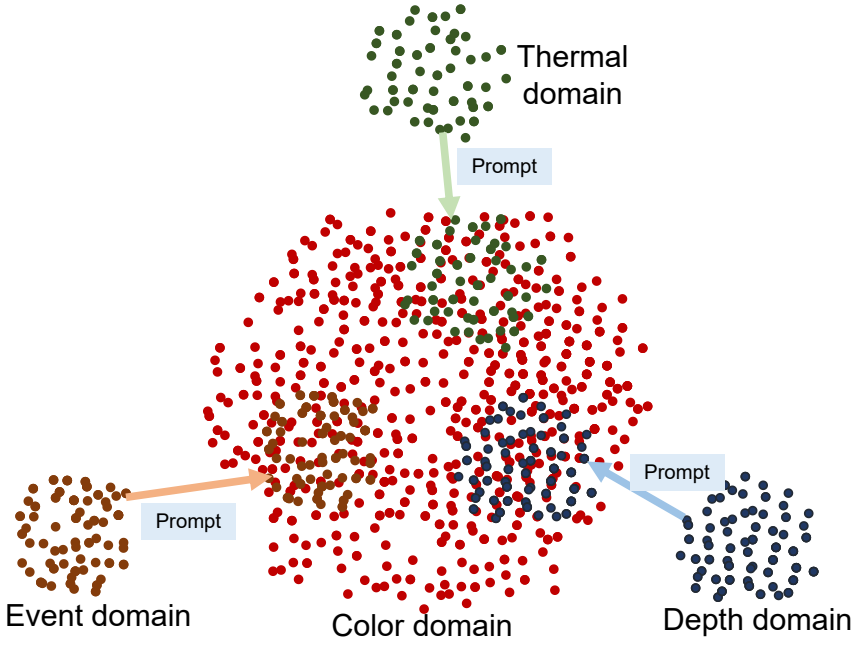


Figure 5.3: Multi-modal prompts.

words, to make multi-modal prompt work, we have to guarantee the parameter λ is relatively very small [179].

Prompting reduces the gap between distributions. Obviously, the data distribution of auxiliary modality is very different from that of the counterpart RGB videos. Thus, naturally, feeding this data directly into a model that learns only from RGB videos fails to achieve good performance. As shown in Fig. 5.3, the purpose of prompting is to transfer the data field of auxiliary modality to the main field of RGB images. The newly generated samples capture the inherent characteristics of both modalities, and since they lie within the distribution region learned by the model, the model can make better predictions. The essential benefit is that the distribution gap between the different modalities has narrowed considerably.

Large-scale data plays a vital role. The original purpose of learning new fused data is to enhance the ability to meet new challenges. However, due to limited data volumes, not only this goal is not achieved, but the fusion module is not fully learned. Furthermore, fine-tuning somewhat destroys the associating ability learned from RGB videos. That's why the main problem we have is that the current fusion models cannot work well or even match the performance of using only RGB trackers. In contrast, our prompting strategy can successfully

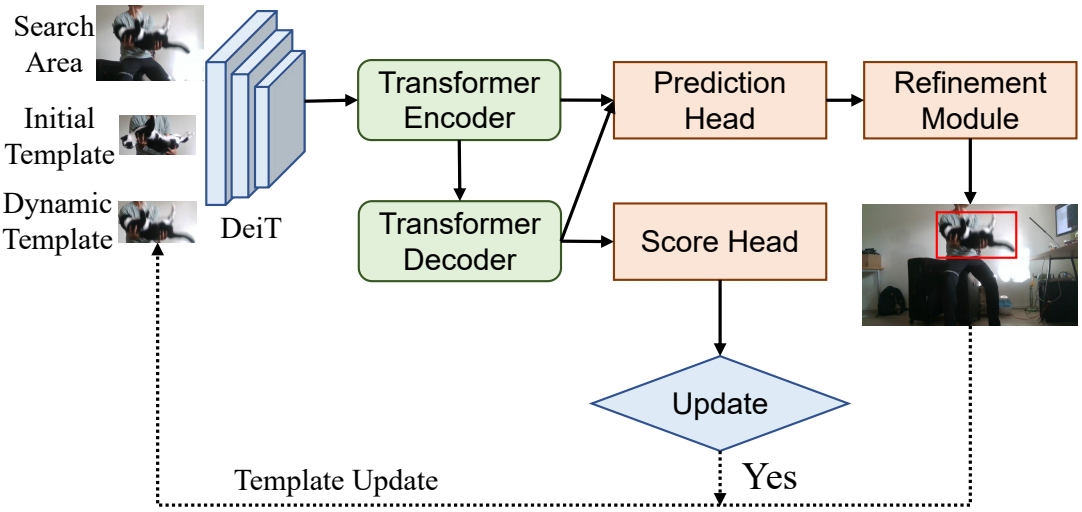


Figure 5.4: Architecture of the pre-trained model STARK.

avoid such a paradox. On the one hand, auxiliary data is used, and thus new challenges can be solved. On the other hand, trackers pre-trained on large-scale data remain unchanged, thus avoiding the knowledge-forgetting problem. Thus, we believe our prompting is a very good compromise under the condition that currently we have no large-scale multi-modal data. Nevertheless, in the future, we also believe that collecting more multi-modal data can better address this issue.

5.3 Experiments

We evaluate the proposed ProTrack for a wide range of downstream multi-modal tracking tasks with pre-trained RGB trackers. We first describe our experimental settings in Sec. 5.3.1, including the downstream tasks and parameter settings. Then a brief introduction to the pre-trained model is given in Sec. 5.3.2. Then we demonstrate the effectiveness of our method on the downstream multi-modal tracking tasks in Sec. 5.3.3. We also systematically study how different design choices would affect performance in Sec. 5.3.4, which leads to an improved understanding of our approach.

5.3.1 Experimental Settings

To verify the effectiveness of our proposed ProTrack, we select the following downstream tracking tasks:

- **RGB-Depth object tracking.** We compare trackers on the popular CDTB [113] and DepthTrack [175].
- **RGB-Thermal object tracking.** We compare trackers on the large-scale LasHeR [97] and RGBT234 [98].
- **RGB-Event object tracking.** We compare trackers on the largest VisEvent [160].

In experiments, we empirically set $\lambda = 0.05$. For color choices, we use JET colormaps for depth maps and thermal images by default, while for event data, we simply use the event images transformed from event flows. All experiments are run on a single 32GB Tesla V100 GPU.

5.3.2 Pre-trained Model Selection

Since we aim to fully employ the discriminative ability of RGB pre-trained models, the choice of pre-trained models is of vital importance. Here we choose a variant of spatial-temporal transformer [173], which leads the leaderboard of RGB object tracking benchmarks. In this paper, we specifically use "STARK" denoting the variant model we selected.

Here we briefly introduce the pre-trained model we selected. STARK is based on spatial-temporal transformer architecture. As shown in Fig. 5.4, STARK contains the following main components: backbone, encoder, decoder, prediction head, and score head. For feature extraction, STARK utilizes DeiT [151] to strengthen the deep features. The encoder-decoder is based on DETR [19]. With inputting both the initial template and dynamic template, the encoder extracts the spatial-temporal features by modeling the correlation in both spatial and temporal dimensions. The decoder takes a single target query to predict a bounding box. The prediction head first takes the search region features from the encoder's output and then computes the similarity between the search region features and the output embedding from the decoder. The

Table 5.3: Overall performance on the CDTB dataset [113].

Method	DS-KCF[14]	CA3DMS[109]	CSR_RGBD++[78]	OTR[80]	DAL[138]
Pr	0.036	0.271	0.187	0.336	0.662
Re	0.039	0.284	0.201	0.364	0.565
F-score	0.038	0.259	0.194	0.312	0.592
Method	TSDM[200]	DeT[175]	STARK[173]	**ProTrack**	
Pr	0.578	0.674	0.740	**0.747**	
Re	0.541	0.642	0.765	**0.767**	
F-score	0.559	0.657	0.752	**0.757**	

Table 5.4: Overall performance on DepthTrack test set [175].

Method	DS-KCF-shape[60]	CA3DMS[109]	CSR_RGBD++[78]	DAL[138]
Pr	0.023	0.212	0.113	0.478
Re	0.023	0.216	0.115	0.390
F-score	0.023	0.214	0.114	0.421
Method	TSDM[200]	DeT[175]	STARK[173]	**ProTrack**
Pr	0.393	0.560	0.558	**0.583**
Re	0.376	0.506	0.543	**0.573**
F-score	0.384	0.532	0.550	**0.578**

similarity is used to enhance the search area features and then estimate the probability distribution of the box corners. With a refinement module based on AlphaRefine [172], we finally get the bounding box prediction. The output is also used as a dynamic template to enhance the target template in most cases. Since there are mostly long-term settings in multi-modal tracking tasks, which means the targets may disappear and reappear during the tracking process, STARK utilizes a scoring head to determine whether the dynamic template should be updated [179].

Specifically, STARK is trained in two stages. In the first stage, the whole network, except for the score head, is trained end-to-end. In the second stage, only the score head is optimized with BCE loss. The training data consists of the training sets from the aforementioned LaSOT [42], GOT-10K [70], COCO2017 [106], and TrackingNet [127]. Top performance on multiple tracking benchmarks demonstrates the discriminative ability of this model. For more details, please refer to [173].

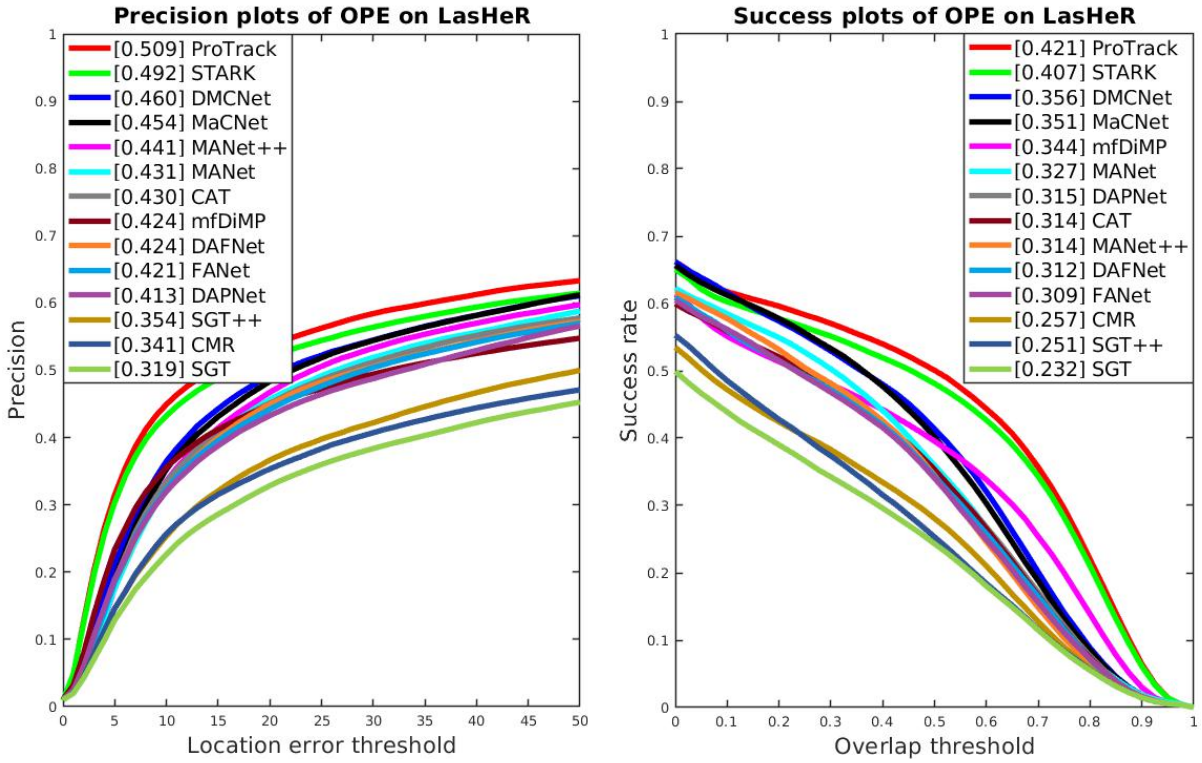


Figure 5.5: Overall performance on the LasHeR test set [97].

5.3.3 Main Results

CDTB dataset [113] consists of 80 long-term RGB-D video sequences for evaluation, covering a wide range of challenges in RGB-D tracking. In the CDTB dataset, target objects disappear and reappear frequently. Tracking Precision (Pr) and Recall (Re) are computed under a series of confidence thresholds. F-score is obtained by $F = \frac{2Re \times Pr}{Re+Pr}$. As shown in Table 5.3, we compare our ProTrack with existing state-of-the-art RGB-D trackers on it. As reported, with the same ResNet-50 backbones, our ProTrack outperforms DeT [175] by 10%. Besides, ProTrack reaches a new state-of-the-art F-score of 75.7%, which also surpasses the STARK without prompting.

DepthTrack dataset [175] is a large-scale long-term RGB-D tracking benchmark. We evaluate RGBD trackers on the DepthTrack test set, which contains 50 long RGB-D video sequences. DepthTrack dataset uses the same protocols as the CDTB dataset. Table 5.4 presents that our ProTrack surpasses all previous state-of-the-art trackers, obtaining a new state-of-the-art F-score of 57.8%. Also, without multi-modal prompts, the F-score of STARK is 0.550, while our ProTrack has an improvement of 2.8% thanks to the informative prompts.

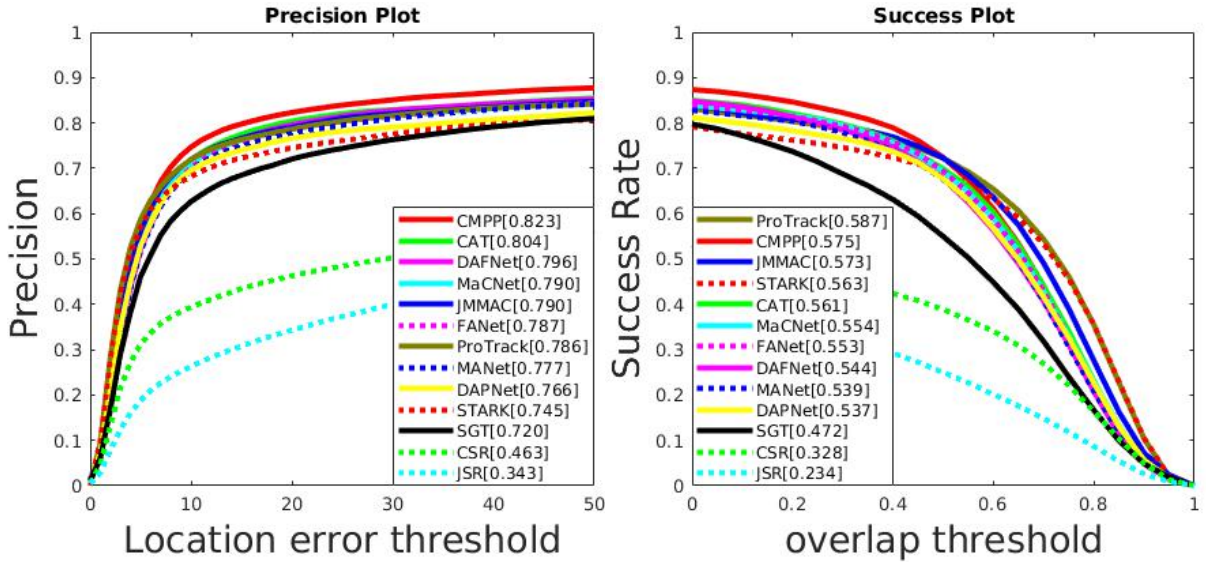


Figure 5.6: Overall performance on the RGBT234 dataset [98].

LasHeR dataset [97] is a large-scale high-diversity benchmark for short-term RGB-T tracking. LasHeR equips the standard tracking performance metrics including Precision Plot and Success Plot. We evaluate trackers on the testing subset which contains 245 video sequences. Here we compare our ProTrack with existing state-of-the-art RGB-T trackers. Note that the compared models are trained on the LasHeR training set. The results are reported in Fig. 5.5. As shown, our ProTrack shows the top performance of 50.9/41.9 (precision/success), outperforming the well-performing DMCNet [111] by 4.9%/6.3%.

RGBT234 dataset [98] is a large-scale RGB-T tracking dataset for performance evaluation, which contains 234 videos and 116.6k image pairs. Here we compare ProTrack with state-of-the-art RGB-T trackers. Comparison results are shown in Fig. 5.6. Without pre-training or adaptive learning on RGB-T data, our ProTrack can obtain a competitive precision rate of 78.6%. Moreover, ProTrack reaches the top success rate of 58.7%, which beats the well-designed RGB-T trackers. Compared to the STARK which is used for comparison, our ProTrack has improvements of 4.1% and 2.4% on precision and success rate, respectively. Till now, we have observed that our ProTrack shows better on the success plots compared to the precision ones. Note that the precision plot requires manually setting thresholds for location error, which might be too tolerant for trackers to obtain higher precision. While the success plot reflects the real overlap ratio,

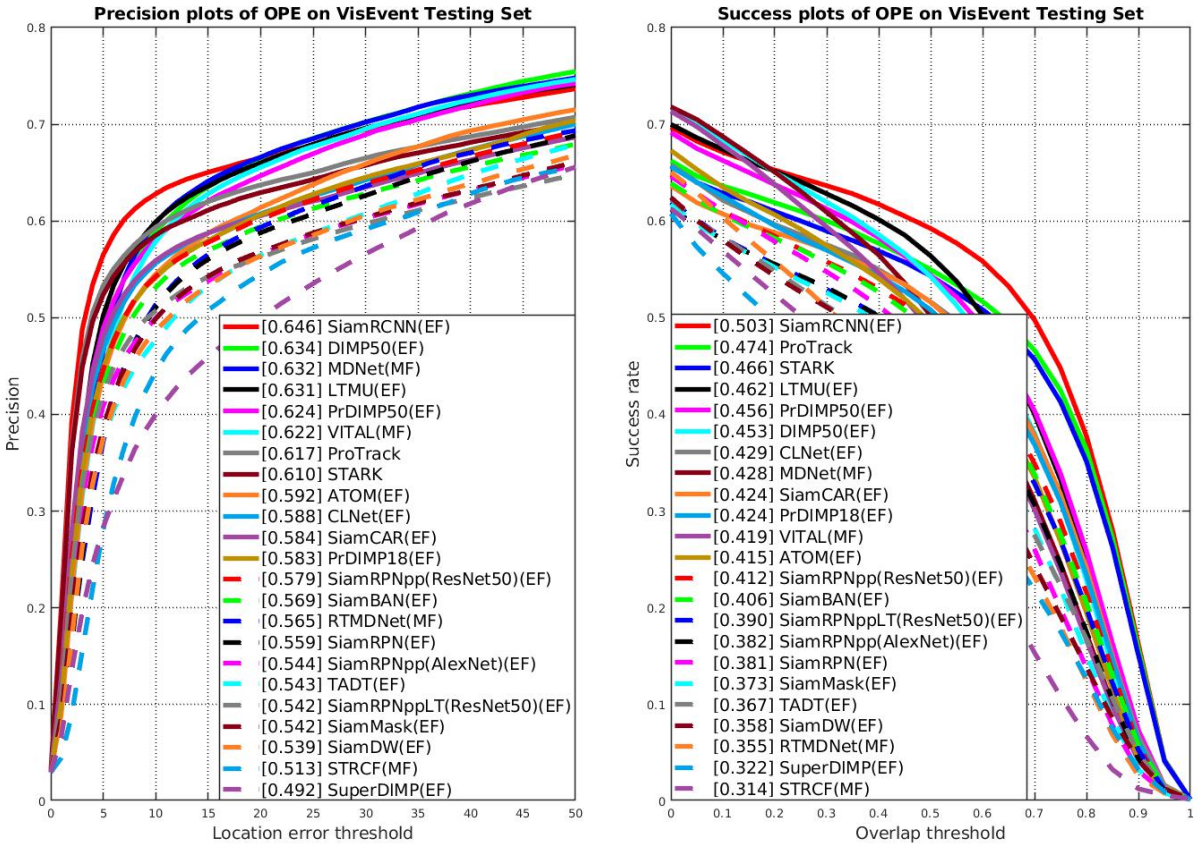


Figure 5.7: Overall performance on VisEvent test set [160].

which can be fairer than the precision one.

VisEvent dataset [160] is a large-scale Visible-Event benchmark. We evaluate trackers on the VisEvent test subset (320 videos). Two standard metrics are adopted for the evaluation of tracking performance, including Precision Plot and Success Plot. We involve the multiple baseline methods [160] to compare, which keep state-of-the-art on VisEvent. Note that the compared trackers are dedicated to visible-event object tracking and fine-tuned on the VisEvent training subset. Our ProTrack still performs on par with current SOTAs. Except for without domain-specific training, another reason for our sub-optimal performance is that the event modality after prompting can only provide positive/negative $(+1/-1)$ on some pixels, which is much less informative compared to the depth or thermal maps.

5.3.4 Ablation Study

We ablate different prompt design choices on the large-scale benchmarks for ablation study. For DepthTrack [175], we use F-score for comparison. For LasHeR [97] and VisEvent [160], we use precision rate and success rate.

Effect of different modalities. To investigate the contributions of different modalities, we evaluate the tracking performance with single-modal inputs. Results are reported in Table 5.5. "Default" denotes the input data we used with multi-modal prompts. Auxiliary information is transferred to colormaps for evaluation. As shown, the tracker performs better in visible modality compared to auxiliary modalities, and the performance is highly dependent on visible information. While they are not comparable to the ones with our default multi-modal prompts, which effectively work on multi-modal data.

Effect of color choices. As we choose colormaps for representation, color choices are the key components in tracking the performance of ProTrack. Specifically, we compare different color choices as shown in Table 5.6. By default, the colormaps follow the JET style for covering more colors in the RGB domain. Since event cameras only capture the changing pixels, there are only red or blue pixels by default in RGB-Event experiments. "RED" is the single-color colormap covering (0,0,0) to (255,0,0). "GRAY" indicates that we only use grayscale maps. As reported, the default settings can achieve better performance compared to the single-colored ones. It indicates that more colors bring more information and are more similar to the natural RGB image color distribution.

Effect of parameter λ. In practice, the hyperparameter λ, which indicates the trade-off between different modalities, is crucial in ProTrack. To investigate the effect of λ, we evaluate ProTrack with different λ, as shown in Table. 5.7. When $\lambda = 0$, there are no prompts and the input remains the visible modality only. As reported, the performance of ProTrack increases first and then decreases with λ improves on DepthTrack and VisEvent. This can be explained by the that, a tiny λ can preserve strong discriminative ability from visual appearances but will undermine the visibility of the auxiliary modality, and vice versa. In addition, we observe that the performance on LasHeR keeps going higher with λ going larger, indicating that the thermal

Table 5.5: Ablation study on different modalities.

	DepthTrack[175]	LasHeR[97]		VisEvent[160]	
Modality	F-score	Pre	Suc	Pre	Suc
Default	0.578	0.509	0.419	0.617	0.474
Visible	0.550	0.492	0.405	0.610	0.466
Auxiliary	0.297	0.349	0.289	0.411	0.277

Table 5.6: Ablation study on color choices.

	DepthTrack[175]	LasHeR[97]		VisEvent[160]	
Color	F-score	Pre	Suc	Pre	Suc
Default	0.578	0.509	0.419	0.617	0.474
RED	0.564	0.486	0.401	0.596	0.443
GRAY	0.561	0.501	0.410	0.546	0.396

Table 5.7: Ablation study on parameter λ. **Bold** denotes the highest score.

	DepthTrack[175]	LasHeR[97]		VisEvent[160]	
λ	F-score	Pre	Suc	Pre	Suc
0	0.550	0.492	0.405	0.610	0.466
0.01	0.557	0.495	0.407	0.612	0.469
0.05(Default)	**0.578**	0.509	0.419	**0.617**	**0.474**
0.1	0.537	0.527	0.436	0.611	0.468
0.2	0.499	**0.531**	**0.439**	0.606	0.454

Table 5.8: ProTrack with more pre-trained models. Performance on DepthTrack [175] is shown according to F-score. "_P" denotes tracking performance after prompting.

ATOM[36]	ATOM_P	DiMP[7]	DiMP_P	PrDiMP[35]	PrDiMP_P
0.313	0.349	0.377	0.431	0.392	0.402
TransT[24]	TransT_P	KeepTrack[124]	KeepTrack_P		
0.489	0.504	0.509	0.542		

maps are more informative or more color-like, and thus they can provide more complementary compared to depth and event information. Thus, choosing a proper λ is essential in our ProTrack.

5.4 Analysis and Discussion

In this section, we conduct a deep analysis of ProTrack for a better understanding of its working mechanism from various perspectives.

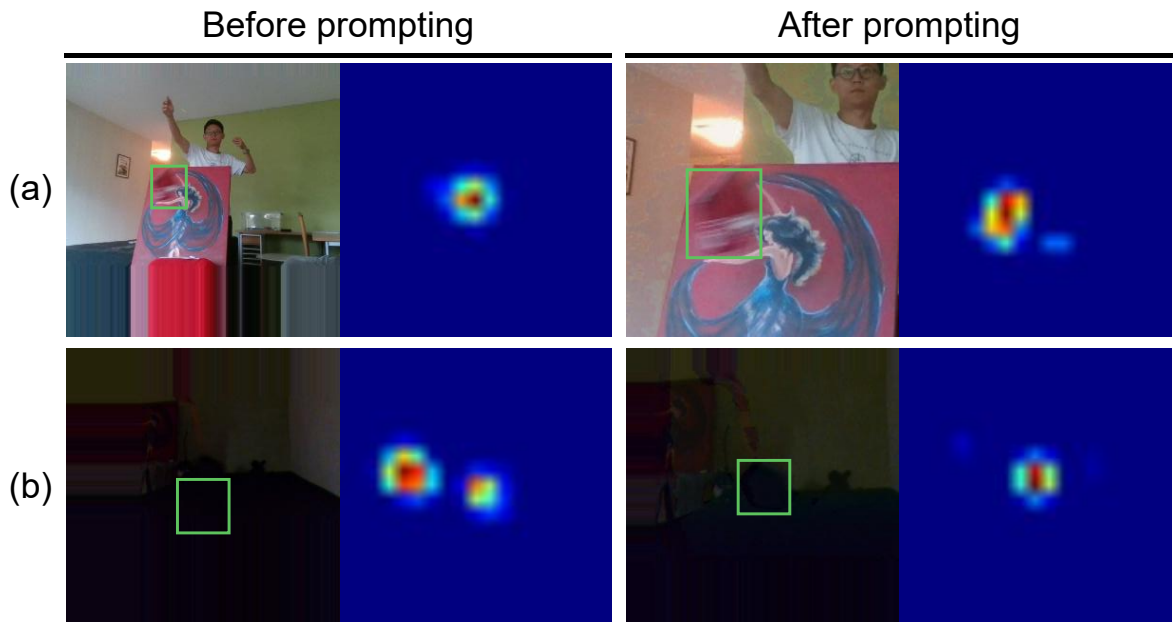


Figure 5.8: Visualized comparison of the score maps in search regions with/without prompting. The groundtruth bounding box is shown in green.

5.4.1 Visualization

To investigate how the multi-modal prompts work during the tracking process, we visualize the search regions and corresponding feature maps before and after prompting in two representative tracking frames, as shown in Fig. 5.8. As shown, we observe that our ProTrack has much cleaner and unambiguous target localization than the one without prompting and leads to an accurate bounding box estimation. While the one without prompts often produces multiple local maxima for distractors, our methods are able to almost fully suppress these. An important design enabling this is the prompts with auxiliary information. Although it is almost invisible to see the auxiliary information after prompting, the feature maps tend to distinguish the objects from background distractors due to the change in overall data distribution. The visualized examples verify the hypothesis shown in Fig. 5.3. Our multi-modal prompts can transfer the invisible auxiliary modality into the visible one by transferring the auxiliary data into the color domain [179].

5.4.2 ProTrack with More Pre-trained Models

As ProTrack aims to leverage the information learned from enormous RGB data, many alternatives can be used as pre-trained models in our ProTrack framework. Here we choose some RGB trackers to simply verify the effectiveness of multi-modal prompts. The representative trackers include ATOM [36], TransT [24], DiMP [7], PrDiMP [35], and KeepTrack [124], which are pre-trained on large-scale RGB datasets, *e.g.*, LaSOT [42], GOT-10K [70], COCO [106], and TrackingNet [127]. We report the results of trackers' F-scores on DepthTrack without and with prompting, respectively. Compared results are shown in Table 5.8. The RGB trackers show improved performance on multi-modal tracking after prompting. Specifically, DiMP and KeepTrack get improvements of 5.4% and 3.3% after prompting, respectively. Thus, ProTrack can continuously benefit from the strong discriminative ability of pre-trained models.

5.4.3 Beyond Dual-modal Tracking

As we claim that our ProTrack is modality-agnostic, its applications can be broader. For example, we can apply ProTrack to triple-modal tracking by modifying Eq. 5.2 to:

$$f(V, A_1, A_2) = \alpha * Color(A_1) + \beta * Color(A_2) + \gamma * Color(V), \tag{5.4}$$

where α, β, γ are parameters, satisfying $\alpha + \beta + \gamma = 1$. And, V, A_1, A_2 denote visible modality and two auxiliary modalities. With prompting, information from different modalities can be converged to the visible modality and get assistance from RGB pre-trained tracking models. Unfortunately, triple-modal tracking has not been explored yet. We hope our work will provide a straightforward solution for this promising direction [179].

5.4.4 Failed Cases

Fig. 5.9 shows failed cases of our tracker. In particular, it shows the adjacent frames in two sequences containing the groundtruth in search regions and the corresponding score maps of the pre-trained tracker with or without prompting [179]. We can see that the location of the

highest scores is not matching the groundtruth. Overall, our tracker typically fails due to two reasons: 1) The tracker's discriminative ability is limited to distinguishing the object in some challenging scenarios. 2) The input image is disturbed by the multi-modal prompts. For the former reason, complex sequences exist and challenge the trackers, as shown in Fig. 5.9(a). This can be solved by the improvements of pre-trained RGB trackers. For the latter, multi-modal prompts in ProTrack may wrongly guide the trackers as it changes the data distribution. For this occasion when the tracking challenges can be solved by RGB trackers, the tracking performance after prompting may be disturbed by auxiliary information, as shown in Fig. 5.9(b). Thus, it will be necessary to design more robust prompts in the future, although our current approach can solve a considerable part of the problems.

5.5 Summary

In this chapter, we propose multi-modal prompts for object tracking, a simple but effective approach to leverage large-scale RGB tracking models for a wide range of downstream multi-modal tracking tasks. Through applying prompting on multi-modal videos, we adapt the multi-modal tracking tasks to pre-trained RGB trackers. Thus, by solely modifying the trackers' input, we exploit most of the discriminative ability from pre-trained RGB trackers to handle the multi-modal tracking challenges. Promising results on 3 tasks and 5 benchmark datasets verify the effectiveness of our proposed ProTrack. We hope this work will spur further research on multi-modal tracking and provide inspiration to related areas.

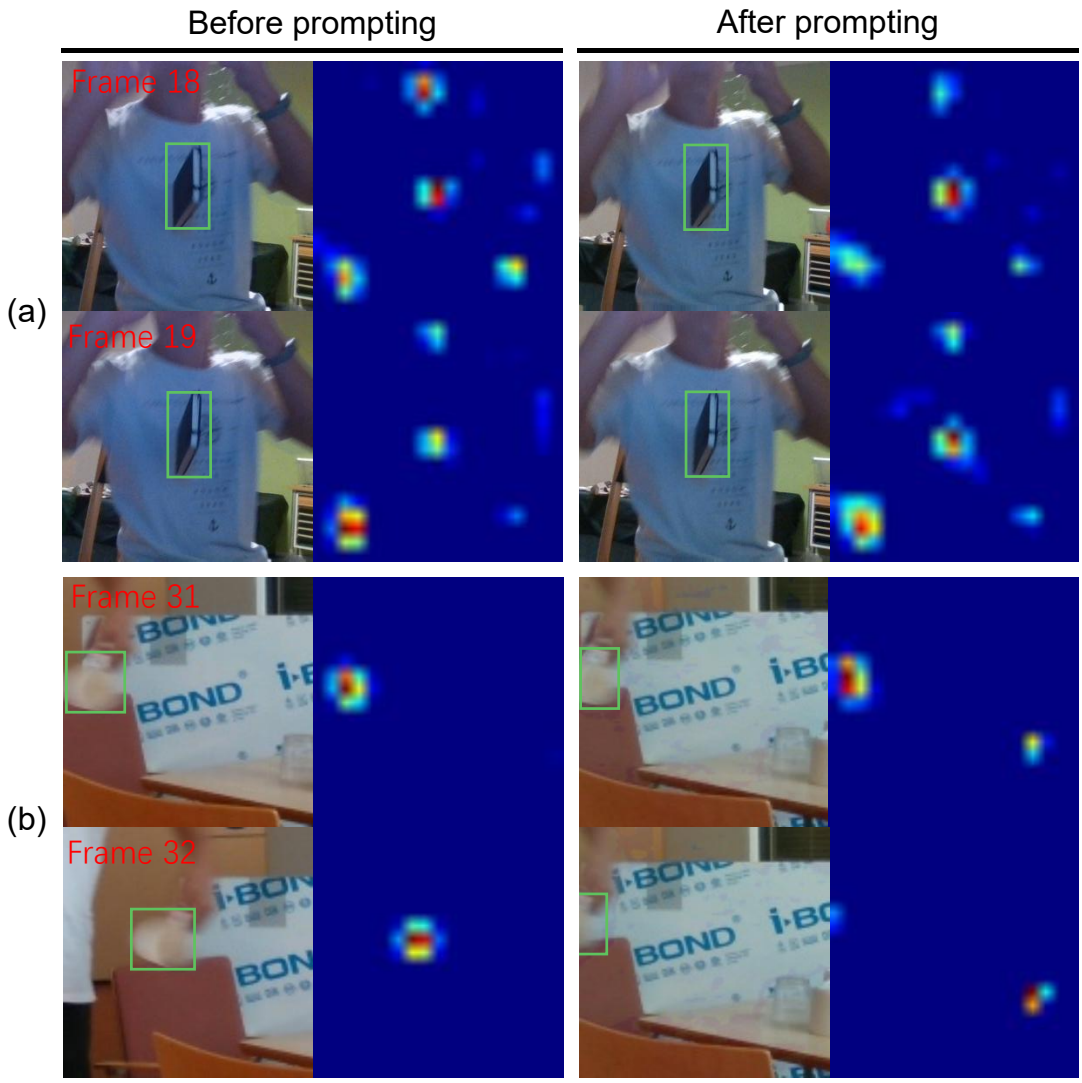


Figure 5.9: Visualization of failure cases of our tracker. The groundtruth bounding box is shown in green.

Chapter 6

Inference-efficient RGBD Tracking

The previous chapter proposed an efficient RGBD tracking algorithm, which not only allows the model to incorporate more modal information but also enables rapid model construction. In this chapter, the predictive efficiency of the model will be discussed in depth. A new lightweight system is proposed and applied to UAV platforms to achieve multi-modal target tracking tasks for UAVs.

Declaration: The materials of this section have been organized as a paper, which was accepted and published on CVPR 2023 [180].

6.1 Preliminaries

6.1.1 Motivation

Aerial robots have been widely used in complex missions. For example, Unmanned Aerial Vehicles (UAVs) equipped with cameras are able to perceive and understand unknown environments and have wide applications in agriculture and surveillance [47, 208]. Specifically, color-based visual tracking with drones has been rapidly developed, thanks to large-scale datasets [208, 126] and dedicated algorithms [45, 178, 71, 104, 46, 18, 16, 44, 17]. However, these UAVs merely equipped with color-based sensors generally fail to deal with the challenges in complex environments, such as background clutters and dark scenes, which break the visibility and illumination

limitations in the color-only domain. For example, current drones have difficulties tracking a person in dark scenes. While RGBD tracking is effective in tackling such kinds of tracking failures [180].

However, for a long time, depth sensors have only been incorporated with UAVs to enable aerial autonomy and collision avoidance [62]. Visual perception like RGBD tracking with drones is unexplored due to the multiple limitations. For example, commercial RGBD sensors are strictly limited by application scenarios and depth measurement range. On the other hand, we notice that current UAV tracking datasets record video sequences in the manner of aerial photography [39, 208]. The captured objects mainly focus on pedestrians and vehicles, and the captured scenes are in urban environments from a birds-eye view.

In this work, we explore RGBD aerial tracking from a more practical viewpoint. Different from existing UAV tracking works, we focus on the unexplored overhead space (2 - 5 meters above the ground), aiming to save the ground space greatly with drone-based visual perception. Instead of mainly focusing on people and vehicles, our research can include more generic objects of different categories, such as hands, cups, or balls. Thus, multimodal aerial platforms in this space are very important, as flying robots with short-range perception capabilities can potentially be used in a wider range of scenarios, such as human-robot interaction [180].

Notably, the new task brings challenges in drone-based visual perception, which can be concluded as follows:

Complex real-world circumstances. The real-world flight comes with complicated and changeable natural environments. On the one hand, the high mobility of drones brings intense pose changes, resulting in huge variations of target scale and considerable motion blurs. Except for the common challenges in visible situations, drone vision also suffers from other problems like low illumination, similar objects, and background clutter.

Limited onboard computational resources. In practical applications, flying platforms generally require higher efficiency on edge platforms with limited resources, while state-of-the-art trackers can only run on powerful GPUs. Especially for multimodal trackers, model efficiency is always the least valued in model design.

Real-time practical applications. Real time is a basic requirement in aerial tracking. Moving platforms require real-time responses and real-world applications also require trackers to function in real-time speed. However, most of the current state-of-the-art trackers even cannot achieve real-time speed on powerful GPUs, not to mention their real-world applications.

Therefore, to achieve UAV visual tracking with depth, we first build a novel RGBD aerial platform to collect videos. The platform is particularly designed to simulate the environments in real-world applications. The captured videos can comprehensively reflect those challenges to be tackled. Using this aerial platform, a large-scale dataset for **D**rone-based RGB**D** aerial tracking, named **D^{2}Cube**, is built. Some examples in our dataset are given in Fig. 6.6. In total, 1,000 sequences are provided with dense bounding box annotations. The settings of captured videos cover diverse scenarios in daily life.

Furthermore, we propose an efficient tracker named **EMT** to facilitate the development of on-board RGBD tracking. The proposed EMT can be treated as a strong baseline for on-board multimodal tracking to simultaneously tackle the above three issues. Thanks to efficient multimodal fusion and feature matching, our proposed tracker can successfully balance the tight computational budget and tracking accuracy. We perform extensive experiments in diverse scenarios and various platforms to validate the effectiveness of our EMT. Competitive tracking performance is observed in comparison with state-of-the-art RGB-only and multimodal trackers, in which EMT runs at a high frame rate of over 100 FPS. Practical application tests are given on *NVIDIA Jetson NX Xavier*, where our EMT can run at a frame rate of over 25 FPS. To conclude, our dataset covers complex aerial tracking scenarios and our method shows a promising balance of accuracy, resources, and speed [180].

The contributions are summarised below:

- **New Problem:** We propose a new task of RGBD air tracking for newly defined overhead space (2m - 5m). Unlike previous aerial tracking, this task is more relevant to human life and has wider applications.

- **New Benchmark:** We construct a large-scale high-diversity benchmark for RGBD aerial tracking. The advantage is that many more categories (34 classes) can be considered than

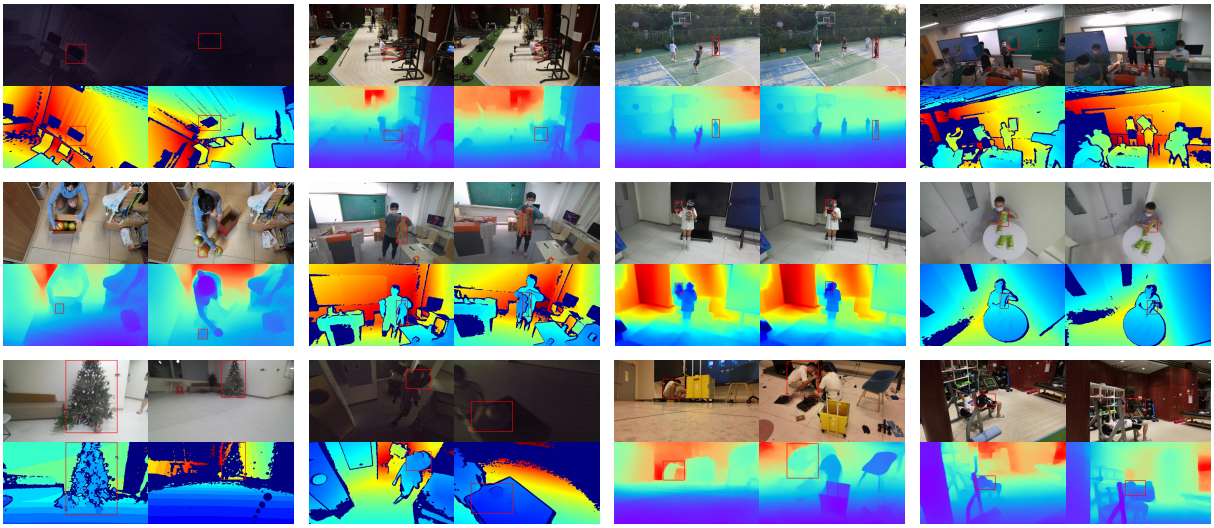


Figure 6.1: Annotated example video sequences in the proposed dataset. As shown, our D^{2}Cube contains multiple challenges.

Table 6.1: Comparison of related datasets for aerial tracking and RGBD tracking. T=Thermal, D=Depth, L=Language, A=Audio.

Scope	Dataset	Modality	Object Type	Scenario	Videos	Year
Aerial Tracking	UAV123 [126]	RGB	Generic	Outdoor	123	2016
	VisDrone-SOT [208]	RGB	Human; Vehicle	Outdoor	167	2018
	UAVDT-SOT [39]	RGB	Human; Vehicle	Outdoor	100	2018
	VT-UAV [194]	RGB; T	Generic	Outdoor	500	2021
	WebUAV-3M [192]	RGB; L; A	Generic	Outdoor	4,485	2022
RGBD Tracking	PTB [149]	RGB; D	Generic	Indoor	100	2016
	STC [167]	RGB; D	Generic	Indoor; Outdoor	36	2018
	CDTB [113]	RGB; D	Generic	Indoor; Outdoor	80	2019
	DepthTrack [175]	RGB; D	Generic	Indoor; Outdoor	200	2021
RGBD Aerial Tracking	**D^{2}Cube**	**RGB; D**	**Generic**	**Indoor; Outdoor**	**1,000**	**2022**

existing aerial tracking datasets. As far as we know, this is the first dataset that can test multimodal aerial tracking models.

- **New Baseline:** An efficient tracking baseline is proposed for RGBD aerial tracking, which is the first real-time tracker for efficient on-board multimodal tracking. It performs better than classical UAV trackers and maintains comparable efficiency.

6.1.2 Aerial Tracking

In general, aerial tracking, *i.e.*, UAV-based tracking, is to track target objects in consecutive frames with drone-based views. Various drone-based datasets are proposed for color-based object tracking, as shown in Table 6.6. We notice that existing UAV tracking datasets focus

on high-altitude aerial photography capturing vehicles mainly. For example, the well-known VisDrone [208] and UAVDT [39] both contribute to vehicle tracking in a birds-eye view. The limitations of them are obvious. On the one hand, they are captured at high altitudes, which has a gap with our daily life scenes. On the other hand, UAVs and cameras can only work under visible conditions. More complex scenarios will lead to flight and data failure. In contrast to the above datasets, our proposed D^2Cube contains multimodal information and more diverse scenarios, bringing new challenges to aerial tracking tasks.

At the arithmetic level, UAV-based tracking faces the challenges of both limited computational resources and strict real-time speed requirements, impeding the usage of state-of-the-art trackers. Thus, UAV tracking requires efficient tracking algorithms. Existing UAV trackers have shown their effectiveness in RGB-based tracking. LightTrack [174] achieves a lightweight tracking framework by using NAS. TCTrack [18] provides a holistic temporal encoding framework to handle temporal contexts in Siamese-based aerial tracking. HCAT [23] achieves high tracking speed thanks to the hierarchical cross-attention transformer. However, unlike color-based UAV tracking, there is much less attention paid to multimodal tracking efficiency and multimodal tracker's speed on edge devices [180].

However, RGBD tracking still suffers from bad speed and performance balance. To the best of our knowledge, this work is the first one contributing to RGBD tracking efficiency, in which our proposed method can run on edge platforms with real-time speed.

6.2 Dataset Construction

6.2.1 Data Collection

Flight platforms. We present our real-world data collection on a handcrafted flight platform, mounted with advanced RGBD cameras, *i.e.*, *Microsoft Azure Kinect DK*, *ZED 2i Stereo Camera*, and *Intel RealSense D455*, as shown in Fig. 6.6. The flight platform is to provide the aerial view and the RGBD cameras are to acquire high-quality synchronous color and depth flows. They are used for video collection under different scenarios and different viewpoints, which can increase

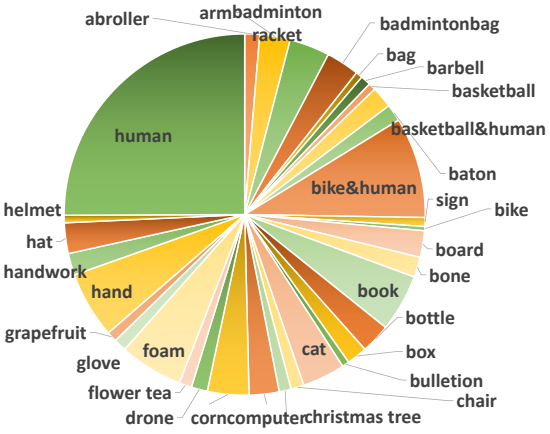


Figure 6.2: Object classification and distribution in the test set.

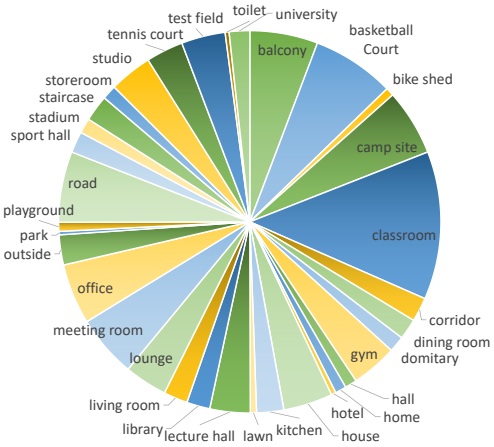


Figure 6.3: Data distribution of scenarios appeared in our test set.

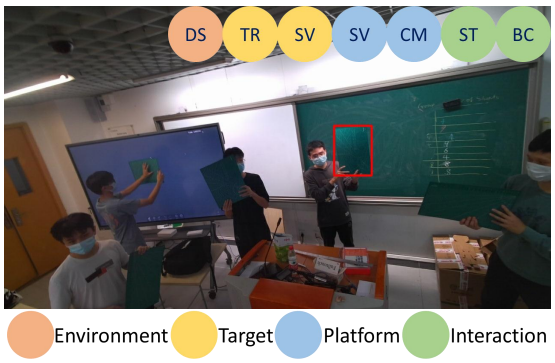


Figure 6.4: An annotated example with bounding box and attributes.

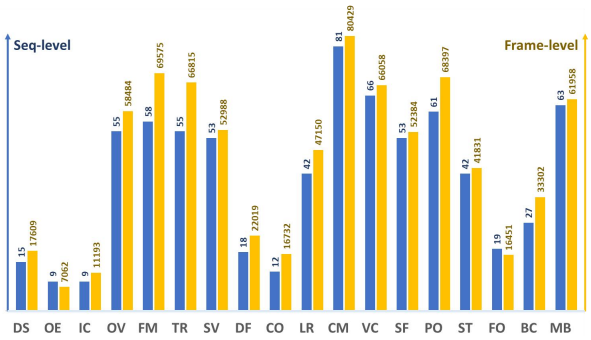


Figure 6.5: Attribute distribution in our test set.

the dataset diversity in the acquisition process. A *Nvidia Jetson NX Xavier* computer running Ubuntu 18.04 is mounted in our UAV for computational support. The overall weight (including LiPo battery and propellers) is about 2.5 kg, with dimensions of 450×450×250 mm. For ease of use, we also apply a compact commercial camera drone platform - *DJI Mavic Air 2* - to acquire high-quality RGB video streams, with which we then obtain depth maps by monocular depth estimation. This helps us to capture videos in some narrow spaces and guarantee flight safety. To maintain high-quality depth information in the whole dataset, we employ DenseDepth [2] to generate corresponding depth maps.

RGBD acquisition setups. The following three RGBD acquisition setups are used to increase the dataset diversity in terms of hardware: (i) *Microsoft Azure Kinect DK* is based

Table 6.2: Attributes and corresponding description.

Level	ABB.	Description
Environment	DS	*Dark Scene.* The light is too low to distinguish the target.
	OE	*Overexposure.* The illumination is too high to distinguish the target.
	IC	*Illumination Change.* There are illumination changes during one video sequence.
Target	OV	*Out of View.* Object partially or fully leaves the view.
	FM	*Fast Motion.* The average per-frame object motion is larger than 20 pixels.
	TR	*Target Rotation.* Target rotates in a plane or out of the plane.
	SV	*Scale Variation.* Ratio change of target size between minimum and maximum is more than 50%.
	DF	*Deformation.* The object is deformable.
	CO	*Composite Objects.* The target object is an ensemble of multiple objects (e.g. man with a basketball).
Platform	LR	*Low Resolution.* The ratio of the object area to the image size is lower than 5%.
	CM	*Camera Motion.* The camera moves/shakes.
	MB	*Motion Blur.* The target is blurred due to the motion of itself or the camera.
	VC	*Viewpoint Change.* The viewpoint is not fixed because the capturing angle changes.
	SF	*Sensor Failure.* At least one camera cannot provide useful information.
Interaction	PO	*Partial Occlusion.* The object is partially occluded.
	ST	*Similar Targets.* There are similar objects.
	FO	*Full Occlusion.* The object is fully occluded.
	BC	*Background Clutter.* There are distractors around the target object.

on the Time-of-Flight (ToF) method, measuring depth in a range of 0.5m to 5.46m. It is used for indoor scenarios. (ii) *Intel RealSense D455* uses structure light for depth perception, with an ideal depth measurement range of 0.6m to 6m, designed for both indoor and outdoor scenarios. (iii) *ZED 2i Stereo Camera* reproduces human vision based on stereo vision and neural networks, which provides depth perception from 0.2m to 20m for outdoor applications. All three devices provide synchronized RGB and depth camera streaming with configurable delay between cameras. RGBD cameras are connected to the drone by pan-tilt, thus the capturing viewpoints can be flexibly changed. All videos are captured under 30 fps, with resolution normalized to 1280×720 pixels.

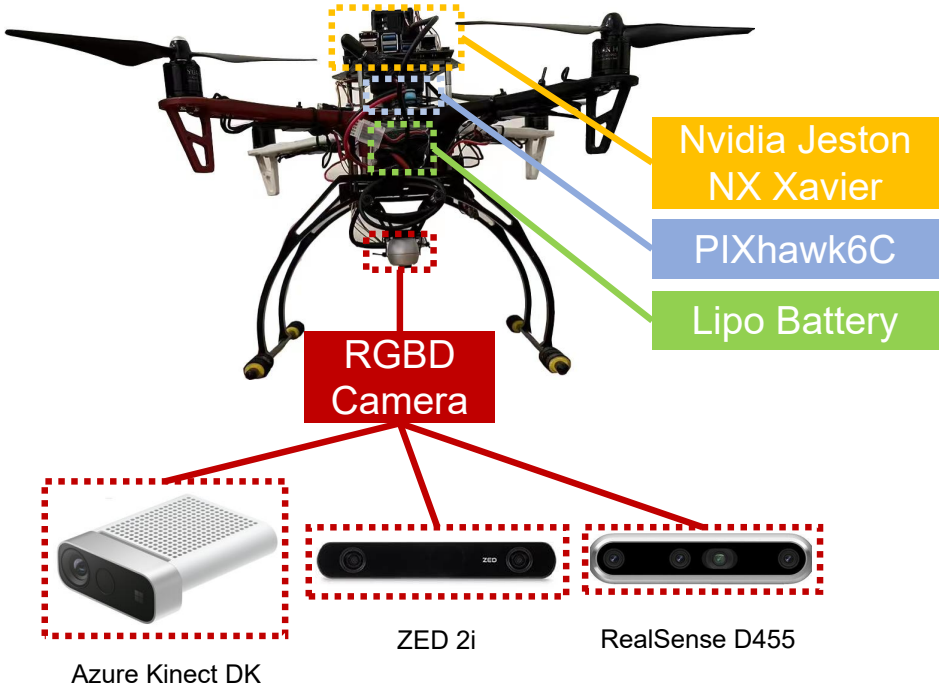


Figure 6.6: Overview of our data collection platform. Three alternatives are provided for capturing RGBD data. Note that RGBD cameras are connected to the drone by pan-tilt, thus the capturing viewpoints can be flexibly changed.

6.2.2 Dataset Statistics

Statistics. We provide 1,000 challenging video clips (1,030,097 frames) in total, including 900 sequences for training (929,370 frames), and 100 for testing (100,727 frames), with an average video length of 1030 frames (about 34 seconds). Regarding the training set, we do not provide a further partition and users can split the training and validation sets by themselves.

Objects. Unlike previous drone-based tracking datasets which only contain limited object categories, our D^{2}Cube covers generic objects. Fig. 6.2 shows the object classes in our test set, in which 34 classes are included. Specifically, we include some classes that rarely appeared in the semantic area, *e.g.*, part of an entire object (upper part of a body) or composite object (man holding a basketball). The whole D^{2}Cube includes more than 100 categories and covers diverse objects, it thus is representative of daily scenarios.

Scenarios. In this work, we mainly focus on daily life scenarios. We involve many applicable scenarios for RGBD aerial tracking, *e.g.*, sports, work, service, and entertainment, in which aerial robots and depth cameras can both work well. In detail, our recording scenarios cover daily life

scenes, including office, bedroom, meeting room, gym, stadium, kitchen, and so on. Both short-term and long-term tracking scenarios are included. We provide the distribution of captured scenarios in our test set in Fig. 6.3. As shown, 34 places are included in our test set, which covers diverse scenarios for generic aerial tracking evaluation. Specifically, our dataset includes indoor scenarios in human daily scenes, which provides potential on broad applications of aerial robots. Besides, multiple viewpoints and challenges guarantee that the proposed D^2Cube maintains a high diversity.

Annotations. As shown in Fig. 6.6, we provide tight axis-aligned bounding box annotations for the target objects at the frame level. A professional team annotates D^2Cube rigorously. The annotation process follows the following rules: (i) If the target appears in the frame, we annotate the visible part of the target by the tightest bounding box. (ii) if the target does not appear in the frame, we will mark this frame with a "target loss" tag.

Attributes. We define 18 tracking challenges in RGBD aerial tracking and classify the attributes in a hierarchical manner. All the attributes are defined in four levels: environment, target, platform, and interaction. At different levels, correspondingly there are different challenges. Details of each attribute are given in Table 6.2. With such a hierarchical classification of different challenges, we can justify what challenges RGBD trackers are indeed suffering from. We also give an annotated example in Fig. 6.4. The distribution of each attribute in our test set is given in Fig. 6.5.

6.3 EMT: Resource-efficient RGBD Tracking

To achieve RGBD tracking on UAV platforms, trackers' ability to run on edge platforms with limited resources is important. However, the vast majority of RGBD trackers focus on architectural design with heavy backbones and additional modules. Such complex frameworks cannot satisfy the real-time requirements of aerial tracking. In this section, we propose Efficient Multimodal Tracker (EMT) for RGBD aerial tracking, which discards the heavy backbones and additional modules [180].

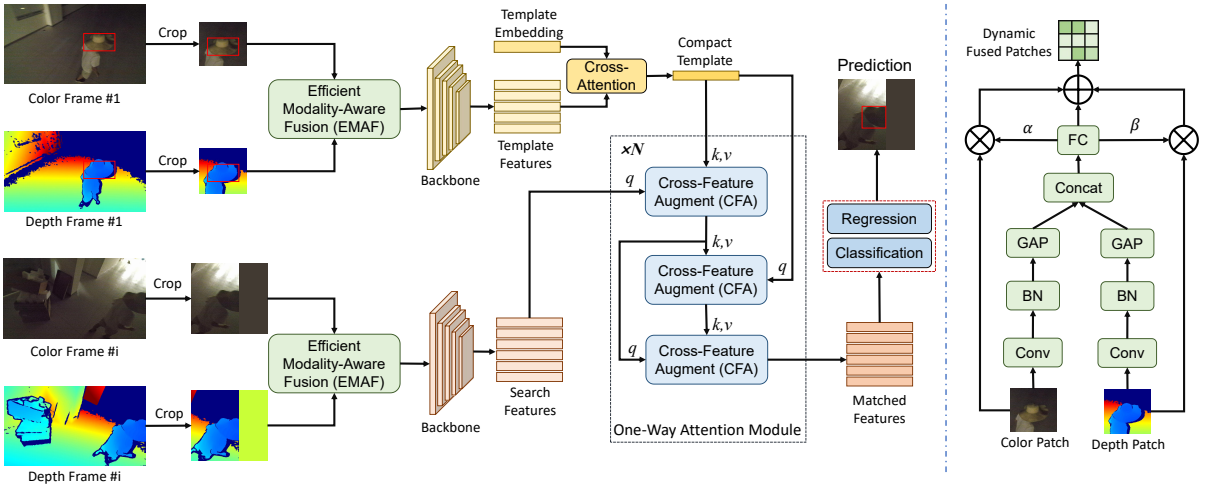


Figure 6.7: Overview of our proposed Efficient Multimodal Tracker (EMT). Left: Pipeline for EMT. Right: Architecture of Efficient Modality-Aware Fusion (EMAF) module.

6.3.1 Multimodal Fusion and Matching Architecture

The proposed EMT contains four main steps, *i.e.*, efficient modality-aware fusion, feature extraction, efficient attention-based feature matching, and target prediction. The overall architecture is shown in Fig. 6.7.

Efficient modality-aware fusion. Firstly, to speed up the fusion, we design a novel Efficient Modality-Aware Fusion (EMAF) module that can first fuse the raw data from multiple modalities at a very early stage. The key to speeding up is that once the high-dimensional features are well integrated, the amount of computation that follows is greatly reduced. In addition, in order to effectively adapt to the scene, we also employ a fusion strategy that can dynamically estimate the importance of features from two modalities. The details of the fusion module to obtain fused image patches $\mathbf{T_{fused}}$ and $\mathbf{X_{fused}}$ are referred to 6.3.2.

Feature extraction. Secondly, we assume that the fused template patch is $\mathbf{T_{fused}} \in \mathbb{R}_{\mathbf{3\times H_{t0}\times W_{t0}}}$ and the fused search patch is $\mathbf{X_{fused}} \in \mathbb{R}_{\mathbf{3\times H_{x0}\times W_{x0}}}$. Then, we treat them as the input of the parameter-sharing backbone network for feature extraction. A modified ResNet-18 Network [23] is used as a backbone network to obtain features maps for templates $f_t \in \mathbb{C}\times\mathbb{H}_t\times\mathbb{W}_t$ and search regions $f_x \in \mathbb{C}\times\mathbb{H}_x\times\mathbb{W}_x$. Here $(H_t, W_t) = (\frac{H_{t0}}{16}, \frac{W_{t0}}{16})$, $(H_x, W_x) = (\frac{H_{x0}}{16}, \frac{W_{x0}}{16})$ and $C = 256$.

Efficient attention-based feature matching. Thirdly, as we obtain the multimodal feature

maps for the template and the search region, the next step is to match the corresponding features. To further speed up this procedure, we first use a trainable embedding to reduce the dimension of template features and then design a one-way attention-based fusion module to efficiently fuse the template features and search area features. We give a detailed description of template-to-search matching in Sec. 6.3.3.

Target prediction. With the template-to-search map, we obtain target predictions by using a regression head and classification head. The regression head is to regress the overlap between groundtruth and bounding box candidates. The classification head is to classify the objects and background.

6.3.2 Efficient Modality-Aware Fusion

Our EMT takes the dual-modal image patches as input and performs a weighted fusion of modalities online. Unlike the modal independent backbone network design of traditional RGBD trackers, our proposed EMT reduces the model's size through a very early learnable fusion.

Raw patch preparation. Specifically, four image patches will be treated as the input, including the color and depth template patches, and the color and depth patches for search regions. On the one hand, the template image patches $\mathbf{T_{rgb}}$ and $\mathbf{T_{depth}}$ are obtained by expanding the target bounding box of the first frame twice in the video. To effectively enhance discrimination, these patches should include the local surrounding information. And, the perturbation is also added to the target to avoid learning location bias. On the other hand, the patches for search regions $\mathbf{X_{rgb}}$ and $\mathbf{X_{depth}}$ are obtained by expanding the target box in the previous frame by four times instead of the whole original image, which utilizes the temporal context in the video sequence and reduces the computational cost.

Dynamic cross-modal fusion. The aim of this step is to fuse the data and reduce the dimension at a very early stage. The intrinsic reason we can fuse these two types of data at such an early stage is that they are pixel-level aligned in image space. Moreover, it is required to dynamically judge the environment of the tracking target by extracting the global information of the image patches. To this end, we calculate the importance of the two modalities in the current

frame based on the context of the two modalities. Based on the importance ratio, the RGB image patch and depth image patch can be fused by the weights for early fusion.

Taking the search branch as an example, the RGB and depth image patches go through 3×3 convolution layer (Conv) and a batch norm layer (BN) to extract discriminative features. Then, a global pooling layer (GAP) is used to extract the global context of the two modalities. Next, the features are concatenated as input of the fully connected layers (FC) to output the importance (α and β) of the two modalities in the current frame. Finally, RGB image patches and depth image patches can be fused by importance weights. The process can be formulated as follows:

$$\begin{aligned} \mathbf{X_{rgb}} &= (\mathrm{GAP}(\mathrm{BN}(\mathrm{Conv}(\mathbf{X_{rgb}})), \\ \mathbf{X_{depth}} &= \mathrm{GAP}(\mathrm{BN}(\mathrm{Conv}(\mathbf{X_{depth}}))), \\ \alpha, \beta &= \mathrm{FC}(\mathrm{Cat}(\mathbf{X_{rgb}}, \mathbf{X_{depth}})), \\ \mathbf{X_{fused}} &= \alpha * \mathbf{X_{rgb}} + \beta * \mathbf{X_{depth}}, \end{aligned} \tag{6.1}$$

where $\mathbf{X_{fused}}$ has the same size of original images.

The immediate benefit of this early fusion is that we avoid extracting features from two modalities by using two separate backbones. Therefore, both the size of memory and the amount of computation have been greatly reduced.

6.3.3 Efficient Attention-based Feature Matching

For a given template feature f_t, we will use an attention-based module to find corresponding features in f_x for a search region. Naturally, similar to [24], both cross-attention between features and self-attention within features can be directly used. However, to speed computation up, we designed a more streamlined network to achieve feature matching.

Compact template representation. To make the template compact, we use a learnable embedding to reduce the dimensions of template vectors f_t. As shown in Fig. 6.7, through the cross-attention with dimension reduction embedding, we obtain a compact template representation f_t^c.

One-Way Attention (OWA) module. With such a compact template representation f_t^c, we then utilize an efficient matching from template to search area. Here, we merely deploy the cross-attention based Cross-Feature Augment (CFA) module [24] for fusion, in which multi-head cross-attention is used in a residual form. However, CFA is utilized in a one-way manner, due to the fact that we only need the template project on search areas to make a prediction in the search area [23]. OWA can be repeated several times. With such a one-way cross-attention module, we can achieve feature matching more efficiently.

6.3.4 Training and Inference

Loss Function. The losses are computed between the outputs of target prediction and groundtruth:

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{cls} + \lambda_2 \mathcal{L}_{bbox} + \lambda_3 \mathcal{L}_{GIou}. \tag{6.2}$$

Here, classification loss $\mathcal{L}_{cls}$ is to discriminate the object from the background. Regression loss consists of $\mathcal{L}_{bbox}$ - Mean Squared Error (MSE) between the predicted bounding boxes and the groundtruth bounding boxes, and $\mathcal{L}_{GIou}$ - generalized IoU loss [140]. We use $\lambda_1 = 0.8344$, $\lambda_2 = 5$, $\lambda_3 = 2$ in our experiments following [24].

Training phase. The training process follows the standard training recipe of current trackers [175, 138, 52]. We use ResNet-18 [63] pre-trained on the ImageNet as our backbone and fine-tune the whole tracking framework with the training set of our proposed D^2Cube and the training data recipe of [175]. We randomly sample two frames within an interval of 30 frames in sequences as template and search region. Subsequently, templates and search regions are obtained with some data augmentation such as jitter, and brightness change, and then resized to 128×128 and 256×256 pixels, respectively. The template embedding size is 16. The model is trained with AdamW [110] optimizer, and the learning rate for the backbone and EMAF module are $1e-5$ and $1e-4$, respectively. Weight decay is $1e-4$. The learning rate decays 10 times every 50 epochs. We sample 128,000 pairs in each epoch and the whole tracker is trained for 100 epochs on a single 32GB Tesla V100 GPU with a batch size of 128.

Tracking phase. During inference, the template and search image are resized to a fixed size. Dynamic cross-modal fusion module performs early fusion of the two modalities. After feature extraction and correlation, the prediction head outputs 256 bounding boxes and classification scores. A penalty window is employed to filter distractors.

6.4 Experiments

6.4.1 Experimental Settings

Hardware. All comparison experiments, except for the onboard tests, are executed on a single 32GB Tesla V100 GPU. A widely-used UAV onboard processor NVidia Jetson NX Xavier is used for onboard tests.

Evaluation protocols. We follow the evaluation principles in long-term RGBD tracking from VOT challenge [85]. One-Pass Evaluation (OPE) is used to test trackers' performance on our proposed D^{2}Cube. At frame t, θ_t is a prediction confidence score and τ_θ is a classification threshold. If the predicted θ_t is not below τ_θ, $A_t(\tau_\theta)$ is used to denote the corresponding prediction. Otherwise, we set $A_t(\tau_\theta) = \varnothing$. Thus, $\Omega(A_t(\tau_\theta), G_t)$ indicates the intersection-over-union (IoU) between the prediction result $A_t(\tau_\theta)$ and the groundtruth G_t. We here calculate the precision-recall over the whole test set as follows:

$$\begin{aligned} Pr(\tau_\theta) &= \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_p}\sum_{A_t(\tau_\theta)\neq\varnothing}\Omega(A_t(\tau_\theta), G_t), \\ Re(\tau_\theta) &= \frac{1}{M}\sum_{i=1}^{M}\frac{1}{N_g}\sum_{G_t\neq\varnothing}\Omega(A_t(\tau_\theta), G_t), \end{aligned} \tag{6.3}$$

where N_p denotes the number of frames in which the target is predicted visible in a video sequence, and N_g denotes the number of frames in which the target is indeed visible. $Pr(\tau_\theta)$ and $Re(\tau_\theta)$ denote the precision and recall metrics for M test videos. F-score is obtained by $F(\tau_\theta) = \frac{2Re(\tau_\theta)Pr(\tau_\theta)}{Re(\tau_\theta)+Pr(\tau_\theta)}$.

Table 6.3: Performance comparison of state-of-the-art RGB aerial trackers on D^2Cube dataset. The top 3 results are shown in red, green, and blue. Speed in FPS (frames per second).

Method	Pr	Re	F-score	Speed
LightTrack [174]	0.500	0.531	0.515	119.5
HiFT [16]	0.404	0.430	0.417	66.9
TCTrack [18]	0.416	0.448	0.432	78.1
SiamAPN [44]	0.413	0.441	0.427	140.2
SiamAPN++ [17]	0.411	0.436	0.423	114.9
DaSiamRPN [210]	0.392	0.415	0.403	200.6
HCAT [23]	0.544	0.578	0.561	148.2
UDT+ [156]	0.387	0.412	0.399	50.4
SiamRPN++ [96]	0.459	0.488	0.473	83.3
UDAT-CAR [189]	0.462	0.492	0.476	33.9
EMT	0.653	0.609	0.630	120.3

6.4.2 Comparison with Aerial Trackers

To show the superiority of multimodal tracking in the aerial tracking area, we compare the performance of our proposed EMT with existing state-of-the-art aerial trackers. Results are given in Table 6.3. According to both tracking accuracy and speed, EMT has performed favorably against other state-of-the-art deep aerial trackers. As shown, EMT outperform UDAT [189], TCTrack [18] and HCAT [23] on F-score by 15.4%, 19.8% and 6.9%, with maintaining high speed. The huge performance differences between EMT and sota aerial trackers show the effectiveness of depth information, especially when trackers work in complex scenarios [180].

6.4.3 Comparison with RGBD Trackers

We also compare our model with state-of-the-art RGBD trackers. Our EMT significantly beats most state-of-the-art RGBD trackers and is on par with ProTrack [179] on tracking accuracy. Specifically, EMT outperforms DeT [175] and DMT [52] by 3.3% and 5.4% on F-score. Besides, we compare the efficiency between EMT and state-of-the-art RGBD trackers. Here, we calculate the MACs, parameters, and tracking speed for a fair comparison of efficiency. EMT achieves comparable performance with ProTrack with 15× fewer params, 25× fewer MACs, and 20× higher speed. Therefore, our EMT can definitely achieve a balance of accuracy, resources, and speed.

Table 6.4: Performance comparison of state-of-the-art RGBD trackers on D^2Cube dataset. The top 3 results are shown in red, green, and blue. Speed in FPS (frames per second).

Method	DAL [138]	TSDM [200]	DeT [175]	DMT [52]	ProTrack [179]	EMT
Pr	0.529	0.521	0.608	0.584	0.669	0.653
Re	0.565	0.492	0.587	0.569	0.644	0.609
F-score	0.547	0.506	0.597	0.576	0.656	0.630
MACs	15.78G	74.08G	30.57G	40.44G	82.58G	3.43G
Params	19.60M	114.59M	34.63M	38.97M	159.61M	10.05M
Speed	21.3	18.2	26.8	25.5	5.4	120.3

6.4.4 On-board Tests

We deploy representative trackers on a commonly-used UAV onboard processor *NVIDIA Jetson NX Xavier* to simulate real-world UAV tracking circumstances. With onboard tests, trackers' real-time capabilities can be evaluated and verified. Fig. 6.8 shows several tests of our EMT in some challenging real-world tests. As shown, the tests cover multiple challenging scenarios, *e.g.*, dark scenes, fast motion, similar objects, and so on. While our EMT can perform successful tracking with an on-board speed of 25 fps. Center Location Error (CLE) refers to the center error between the predictions and groundtruth. We set the error to be within 40 pixels for successful tracking in real-world applications.

6.4.5 Visualized Results

To vividly show the performance of representative trackers on our proposed D^2Cube, we provide more visualized results in Fig. 6.12. The compared trackers include ProTrack [179], DeT [175], HCAT [23], UDAT [189] and the proposed EMT. 18 video sequences covering 18 attributes are shown for comparison. As shown, our EMT can perform well against most of the challenges. Specifically, our EMT can address difficulties like BC (background clutter) and CM (camera motion), in which tracking failures are presented by color-only trackers. Failed cases of EMT are given in OE (overexposure) and SF (sensor failure), which represent some extremely challenging tracking scenarios [180].

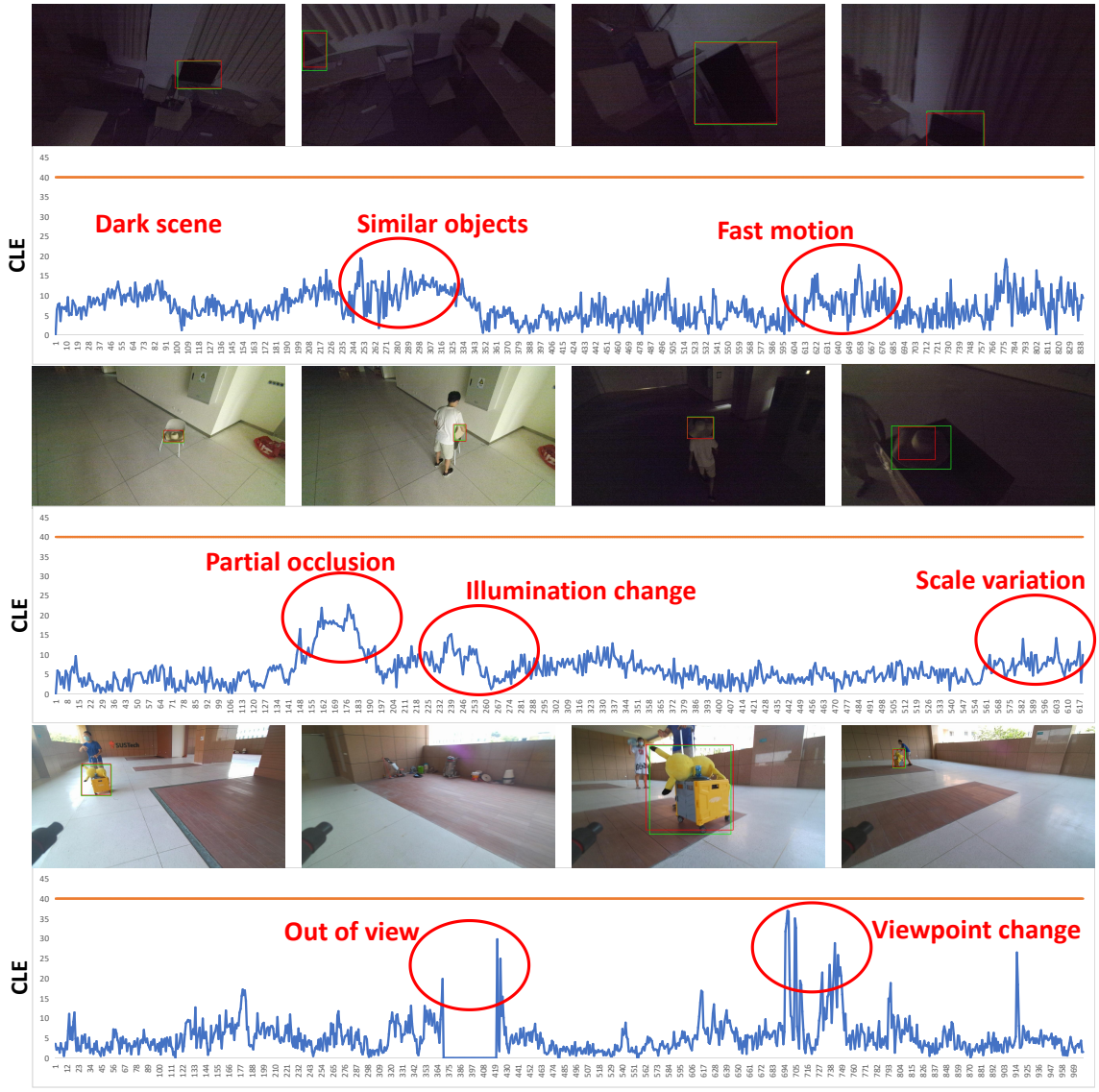


Figure 6.8: The proposed EMT is tested on the UAV platform with Nvidia NX Xavier. The tracking results and ground truth are marked with red and green boxes respectively.

6.4.6 Attribute-based Performance

We also investigate trackers' performance against different kinds of challenges according to our annotated attributes. As shown in Fig. 6.9, RGBD trackers outperform RGB-only trackers in all attributes, especially in the case of attributes like *dark scenes* and *illumination change*, with which the target appearance is not such informative. This verifies that the addition of depth information enhances the discriminative ability of trackers in complex environments. Among RGBD tackers, EMT achieves comparable performance with ProTrack, while the model size is

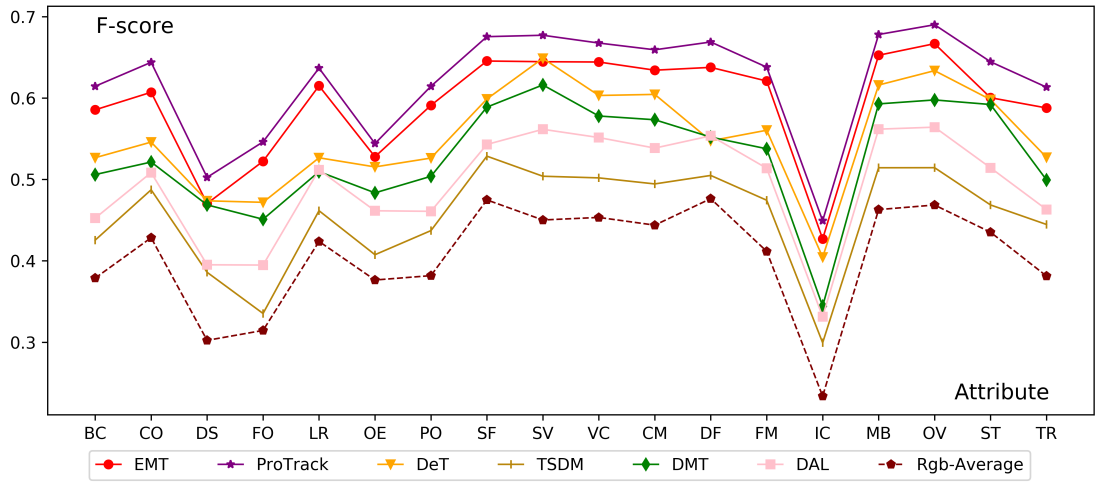


Figure 6.9: Attribute-based performance comparison on D^{2}Cube.

10× smaller and the speed is 20× faster. Besides, EMT far outperforms other sotas in terms of UAV-specific challenges, *e.g*, *low resolution*, *camera motion*, *fast motion*, *viewpoint change*, and *sensor failure*. It confirms that EMT can maintain high performance in complex UAV scenarios. We give some result visualization in Fig. 6.10 to show the qualitative comparison of representative RGB and RGBD trackers against difficulties.

For in-depth analysis, we provide attribute-based performance in terms of F-score on all compared trackers. The results are shown in Fig. 6.11. Obviously, ProTrack [179] shows outstanding performance on all attributes, while our proposed EMT ranks second on 17 of 18 attributes with a very compact model size. Besides, most tested trackers show consistent trends in some attributes, *e.g.*, low performance on illumination variation, and dark scenes, indicating that environment-level attributes are very challenging for state-of-the-art trackers. While trackers show much better performance on classical tracking challenges, *i.e.*, out-of-view, motion blur, scale variation, and so on. In terms of RGBD trackers, DeT [175] performs well on scale variation, except for the outstanding performance of ProTrack and the proposed EMT. On the other hand, RGB trackers perform generally lower than RGBD trackers. Notably, some popular efficient trackers, *e.g.*, HiFT [16], DaSiamRPN [210] and TCTrack [18], show severe performance degradation in terms of illumination change, overexposure, and background clutter, demonstrating that current color-only aerial trackers are very sensitive to the overall appearance change [180].

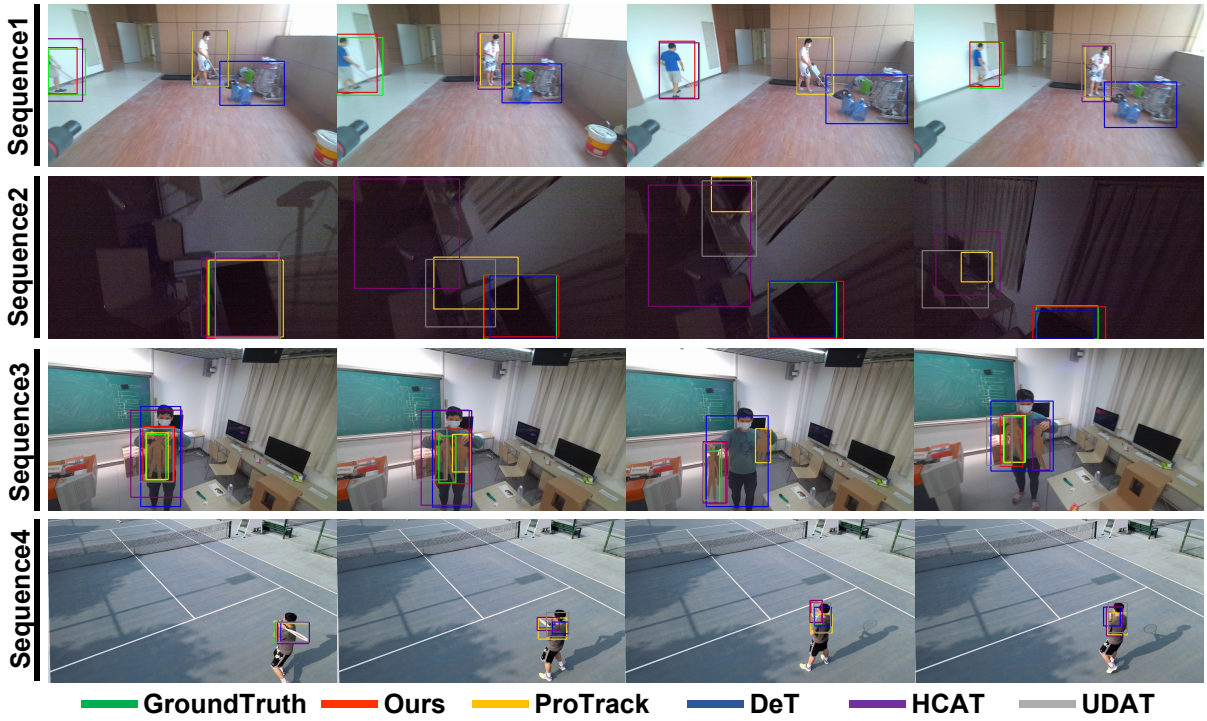


Figure 6.10: Qualitative results of representative RGB and RGBD trackers on D^2Cube dataset.

Table 6.5: Ablation study on different ways for cross-modal fusion.

Method	Pr	Re	F-score	Speed
Add(RGB,Depth)	0.574	0.415	0.472	147.5
Mean(RGB,Depth)	0.531	0.432	0.476	144.6
Max(RGB,Depth)	0.484	0.396	0.435	140.7
EMAF (Proposed)	0.653	0.609	0.630	120.3

6.4.7 Ablation Study

Different ways for cross-modal fusion. We investigate the impact of using different methods for cross-modal fusion. As shown in Table 6.5, common operations, *i.e.*, *add*, *mean* and *max*, show relatively lower performance with F-score degradation of over 10%, compared to the proposed EMAF. This demonstrates that our module can dynamically determine the importance of two modalities in terms of different environments and perform an effective fusion.

Different dimensions of template embedding. As we utilize a compact representation for the template, we also investigate the impact of different template dimensions. As reported in Table 6.6, the 16-dimension embedding gives similar performance to the 32-dimension one, while both of them are much higher than the 4-dimension one, confirming that the modality

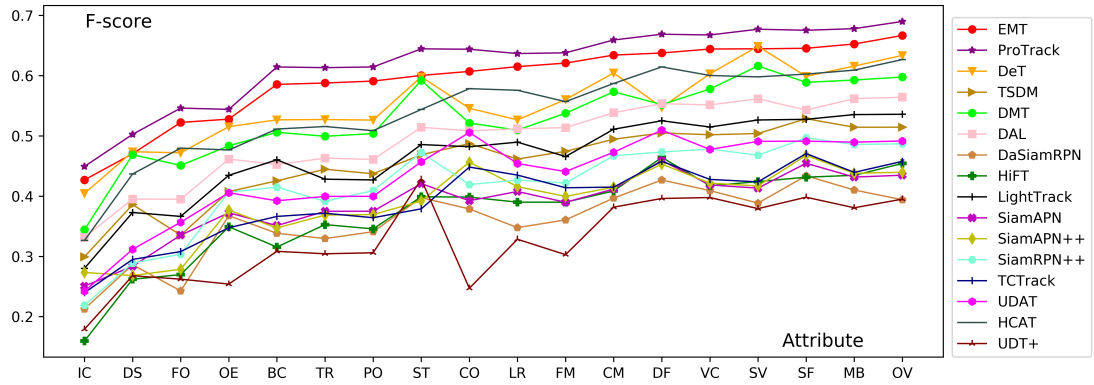


Figure 6.11: Attribute-based performance in terms of F-score. IC = Illumination Change, DS = Dark Scenes, FO = Full Occlusion, OE = Overexposure, BC = Background Clutter, TR = Target Rotation, PO = Partial Occlusion, ST = Similar Targets, CO = Composite Object, LR = Low Resolution, FM = Fast Motion, CM = Camera Motion, DF = Deformation, VC = Viewpoint Change, SV = Scale Variation, SF = Sensor Failure, MB = Motion Blur, OV = Out-of-view.

Table 6.6: Ablation study on the dimension of template features.

Dimension	Pr	Re	F-score	Speed
4	0.467	0.421	0.443	122.5
16 (Default)	0.653	0.609	0.630	120.3
32	0.653	0.604	0.628	114.6

Table 6.7: Ablation study on the number of OWA modules.

OWA modules	Pr	Re	F-score	Params	Speed
1	0.579	0.543	0.561	7.42M	135.5
2 (Default)	0.653	0.609	0.630	10.05M	120.3
4	0.569	0.506	0.536	15.31M	87.1

information is redundant [64, 4] and our compact representation is efficient and effective.

Different numbers of One-Way Attention (OWA) modules. In our experiments, we used the one-way attention modules twice. Table 6.7 gives the performance comparison with different numbers of OWA modules. As reported, two OWA modules perform best, exceeding the one-module approach by 19% with high speed. However, as the number of OWA modules increases to 4, the model performance decreases. It can be explained that too many OWA modules may force the model to aggregate the attention on the invalid information, *e.g.*, the failed value in depth images.

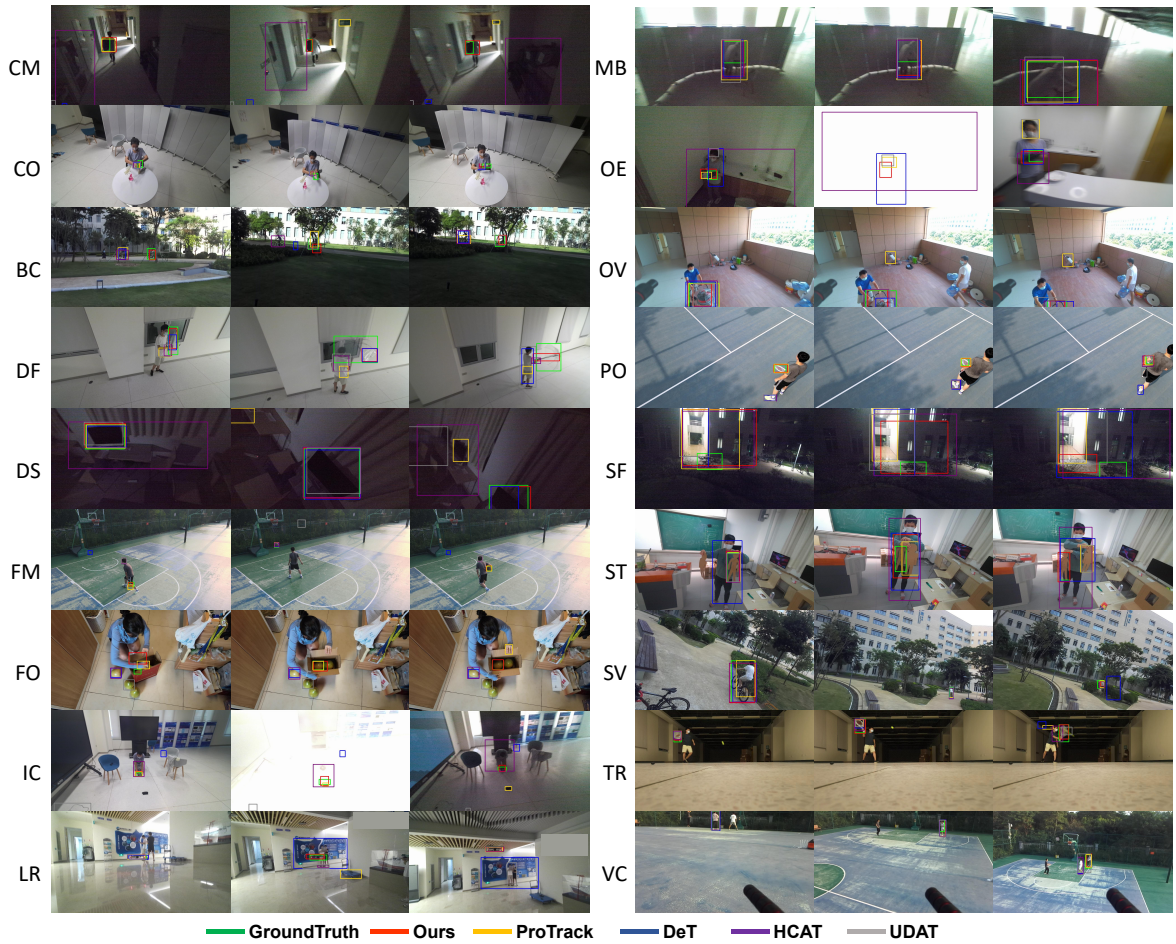


Figure 6.12: Visualized results for different challenges in D^{2}Cube. Zoom in for details.

6.5 Summary

In this chapter, to explore aerial perception in overhead space, we define a new RGBD aerial tracking task. Compared to the previous research scenario, this new task enables more complex drone-based perception. To validate models for this task, we collect a large-scale dataset covering more scenarios and categories than existing aerial tracking datasets. To facilitate research, a strong baseline has been proposed for the RGBD aerial tracking task, and experimental results on our new dataset clearly demonstrate the efficiency and effectiveness of the proposed model.

Chapter 7

Conclusions

In this chapter, the presented works and corresponding findings are summarised, and future works and potential directions are discussed.

7.1 Conclusions

To address the problems and achieve the goals mentioned in Chapter 1, this thesis has introduced four main works on RGBD tracking. Specifically, problems of RGBD tracking are carefully investigated. Based on the findings, corresponding solutions are unfolded from two perspectives, *i.e.*, flexibility, and efficiency. In total, these efforts will bring more flexible and efficient solutions to RGBD tracking.

The conclusions of each chapter are listed as follows:

- Chapter 3 proposed a new task, RGBD video object segmentation under weak supervision (VOS), which is defined as pixel-level tracking under bounding box supervision. To the best of our knowledge, this is the first attempt to apply a weakly supervised paradigm on VOS in the RGBD domain. To this end, we first construct a dataset containing 350 RGBD video sequences for model training and evaluation. This dataset covers common challenges arising in RGBD VOS by providing bounding box and mask annotations. In addition, we further propose *FusedCDNet* to perform multi-modal VOS with only the supervision of

bounding boxes in training and testing, thereby achieving weakly-supervised training to overcome high-cost labeling, cross-modal fusion to handle complex scenes and weakly-supervised predictions to improve ease of use. Finally, extensive experiments verify that our proposed weakly supervised *FusedCDNet* can achieve comparable results with other fully supervised models on RGBD VOS.

- Chapter 4 investigated a novel topic for tracking universal objects in RGBD videos using 3D rotating BBoxes. Firstly, a novel benchmark *Track-it-in-3D* is built using 300 RGBD videos for training and testing, covering various objects and challenging scenarios in 3D scenes. Additionally, this benchmark enables general 3D tracking in complex scenes through novel object annotation and performance evaluation. Furthermore, an end-to-end method, namely *TrackIt3D*, is proposed, to track class-independent 3D objects. With effective RGBD fusion and 3D cross-correlation, this baseline shows superior performance on this challenging task. We hope this work will stimulate further research into general 3D tracking.

- Chapter 5 proposed multimodal prompts for object tracking, a simple yet effective approach that utilizes large-scale RGB tracking models to perform various downstream multimodal tracking tasks. In this work, the multi-modal tracking task is adapted to a pre-trained RGB tracker by applying prompt learning on multi-modal videos. Therefore, by only modifying the input of the tracker, the discriminative power of the pre-trained RGB trackers can be fully exploited to address the multi-modal tracking challenge. Promising results on five benchmark datasets validate the effectiveness of our proposed ProTrack. We hope that this work can spur further research on multimodal tracking and provide inspiration for related fields.

- Chapter 6 explored the aerial perception of overhead space, and a new RGBD aerial tracking task is defined. Compared with previous research scenarios, this new task enables more complex UAV-based perception and has higher requirements on the compactness and real-time performance of the tracker. To validate the models for this task, we collected

a large-scale dataset of data taken in a defined overhead space covering more scenarios and categories than existing aerial tracking datasets. To facilitate research, a strong baseline is proposed for the RGBD aerial tracking task, and experimental results on our new dataset demonstrate the efficiency and effectiveness of the proposed model.

7.2 Future Works

The results achieved in the research of this thesis are only preliminary results in the field of multi-modal fusion tracking. In the future, there are still many challenges worthy of in-depth research, including more modalities input, more semantic and easier input, more fine-grained output, and more efficient training strategies.

More modality input. 1) In this thesis, in addition to RGB information, the main additional modality discussed is depth data. From the conclusion of this thesis, we can see that trackers incorporating depth data can indeed solve many practical problems. However, thermal infrared is a preferred choice in many scenarios where detection of living organisms is required. The data format of this modality is similar to depth data, but the physical properties of the data are completely different and contain a lot of unique information. Therefore, future research on multi-modal tracking algorithms that fuse thermal infrared data is a very valuable direction. 2) In this thesis, only two modal data are fused for target tracking. In fact, in practical applications, especially in complex scenarios, general agents are equipped with a variety of sensors with different attributes to overcome various challenges. Therefore, future research on tracking algorithms that can integrate more modal data at the same time is a very interesting topic. In addition, how to use the currently popular prompt technology to fuse multi-modal data for complex scene perception is also very valuable.

More semantic and easier input. 1) Existing target tracking algorithms assume that the first frame of the image is marked by detection algorithms or humans. Then, manually labeling the bounding box is a very unfriendly operation. This is why the application scenarios of tracking algorithms are very limited, mainly because the interactivity is too poor. In the future,

in multi-modal fusion tracking algorithms, using language to describe the state of the target to refer to the target will make the use of the algorithm more convenient. This strategy can not only track the target through markers in the initial frame but also more accurately track the target through language description of the target's motion changes. 2) In fact, in addition to using language to describe the target state to track the target and output the bounding box to determine the location of the target, directly outputting the mask of the target is more sophisticated and valuable research.

More fine-grained output. 1) Currently, the implemented fine-grained expressions include masks in 2D space and bounding boxes in 3D space. Compared with traditional algorithms that extract 2D bounding boxes in 2D space, these two methods have improved a lot in terms of precision and can achieve more complex applications. However, the true boundary between the object and the scene is a mask in 3D space, allowing the object and scene to be separated at the point level in space. Therefore, fusing multi-modal information to achieve mask-level tracking in 3D space is very valuable but very difficult research. 2) Outputting masks or bounding boxes to represent target locations is a way to use underlying visual information to express the target state. This method can only visually allow anyone to perceive the accuracy of target tracking, and it is difficult to form effective interactions with people. In the future, in addition to outputting the underlying status information, a sentence describing the target's motion status and position changes can be directly output to fully express the target's movement. Even, the model can further output speech and directly broadcast it, thereby increasing interactivity.

More efficient training strategies. 1) Generally speaking, when constructing a data set for machine learning, manual annotation is required at the same level as the information output when the model is predicted. This information includes categories, bounding boxes, masks, poses, and texts. The annotation difficulty of these annotation information is different. For example, the annotation difficulty of masks is much greater than that of bounding boxes. Likewise, annotating detailed textual descriptions is much more difficult than categories. Therefore, the value of weakly supervised learning lies in greatly reducing the difficulty of building a database and accelerating the efficiency of model construction. When building a data set for training,

we can collect annotations at a lower level of difficulty, and then use the algorithm to learn from them to predict the output at a higher level to provide a more refined description. 2) Weakly supervised learning can speed up model training to a certain extent, but it still requires very labor-intensive data annotation work. Currently, large models have made great progress in various fields. The fundamental reason is that the cost of constructing data has been greatly reduced through more convenient label-free self-supervised learning methods. Therefore, the reduction in data cost allows the model to be effectively trained on larger-scale data, and the model effect is greatly improved. Existing multi-modal fusion target tracking algorithms are supervised learning methods, which are difficult to scale up on a large scale. In the future, self-supervised methods can be used for model training to greatly increase the model size to further improve model performance.

References

[1] Adel Ahmadyan et al. “Objectron: A large scale dataset of object-centric videos in the wild with pose annotations”. In: *CVPR*. 2021, pp. 7822–7831.

[2] Ibraheem Alhashim and Peter Wonka. “High Quality Monocular Depth Estimation via Transfer Learning”. In: *arXiv e-prints* abs/1812.11941 (2018).

[3] Ning An, Xiao-Guang Zhao, and Zeng-Guang Hou. “Online RGB-D tracking via detection-learning-segmentation”. In: *ICPR*. IEEE. 2016, pp. 1231–1236.

[4] Roman Bachmann et al. “MultiMAE: Multi-modal Multi-task Masked Autoencoders”. In: *arXiv preprint arXiv:2204.01678* (2022).

[5] Hyojin Bahng et al. “Visual Prompting: Modifying Pixel Space to Adapt Pre-trained Models”. In: *arXiv preprint arXiv:2203.17274* (2022).

[6] Luca Bertinetto et al. “Fully-convolutional siamese networks for object tracking”. In: *ECCV*. Springer. 2016, pp. 850–865.

[7] Goutam Bhat et al. “Learning discriminative model prediction for tracking”. In: *ICCV*. 2019, pp. 6182–6191.

[8] Goutam Bhat et al. “Learning What to Learn for Video Object Segmentation”. In: *ECCV*. Vol. 12347. Lecture Notes in Computer Science. Springer, 2020, pp. 777–794.

[9] Adel Bibi, Tianzhu Zhang, and Bernard Ghanem. “3D Part-Based Sparse Tracker With Automatic Synchronization and Registration”. In: *CVPR*. June 2016.

[10] Sergi Caelles et al. “One-Shot Video Object Segmentation”. In: *CVPR*. IEEE Computer Society, 2017, pp. 5320–5329.

[11] Holger Caesar et al. “nuscenes: A multimodal dataset for autonomous driving”. In: *CVPR*. 2020, pp. 11621–11631.

[12] Jiarui Cai et al. “MeMOT: Multi-Object Tracking with Memory”. In: *CVPR*. IEEE, 2022, pp. 8080–8090.

[13] Massimo Camplani et al. “A Benchmarking Framework for Background Subtraction in RGBD Videos”. In: *ICIAP Workshops*. Vol. 10590. Lecture Notes in Computer Science. Springer, 2017, pp. 219–229.

[14] Massimo Camplani et al. “Real-time RGB-D Tracking with Depth Scaling Kernelised Correlation Filters and Occlusion Handling.” In: *BMVC*. Vol. 4. 2015, p. 5.

[15] Yuanzhouhan Cao, Chunhua Shen, and Heng Tao Shen. “Exploiting Depth From Single Monocular Images for Object Detection and Semantic Segmentation”. In: *IEEE Transactions on Image Processing* 26.2 (2017), pp. 836–846.

[16] Ziang Cao et al. “HiFT: Hierarchical feature transformer for aerial tracking”. In: *ICCV*. 2021, pp. 15457–15466.

[17] Ziang Cao et al. “SiamAPN++: Siamese attentional aggregation network for real-time uav tracking”. In: *IROS*. IEEE. 2021, pp. 3086–3092.

[18] Ziang Cao et al. “TCTrack: Temporal Contexts for Aerial Tracking”. In: *CVPR*. 2022, pp. 14798–14808.

[19] Nicolas Carion et al. “End-to-end object detection with transformers”. In: *ECCV*. Springer. 2020, pp. 213–229.

[20] Kai Chen et al. “Hybrid task cascade for instance segmentation”. In: *CVPR*. 2019, pp. 4974–4983.

[21] Lin-Zhuo Chen et al. “Spatial Information Guided Convolution for Real-Time RGBD Semantic Segmentation”. In: *IEEE Transactions on Image Processing* 30 (2021), pp. 2313–2324.

[22] Wei Chen et al. “FS-Net: Fast Shape-based Network for Category-Level 6D Object Pose Estimation with Decoupled Rotation Mechanism”. In: *CVPR*. 2021, pp. 1581–1590.

[23] Xin Chen et al. “Efficient Visual Tracking via Hierarchical Cross-Attention Transformer”. In: *arXiv preprint arXiv:2203.13537* (2022).

[24] Xin Chen et al. “Transformer tracking”. In: *CVPR*. 2021, pp. 8126–8135.

[25] Y. Chen et al. “3D object tracking via image sets and depth-based occlusion detection”. In: *Signal Processing* (2015).

[26] Ho Kei Cheng, Yu-Wing Tai, and Chi-Keung Tang. “Rethinking Space-Time Networks with Improved Memory Coverage for Efficient Video Object Segmentation”. In: *NeurIPS*. 2021, pp. 11781–11794.

[27] Jingchun Cheng et al. “SegFlow: Joint Learning for Video Object Segmentation and Optical Flow”. In: *ICCV*. IEEE Computer Society, 2017, pp. 686–695.

[28] Suhwan Cho et al. “Pixel-Level Equalized Matching for Video Object Segmentation”. In: *arXiv preprint arXiv:2209.03139* (2022).

[29] Suhwan Cho et al. “Tackling background distraction in video object segmentation”. In: *ECCV*. 2022, pp. 446–462.

[30] Christopher Choy, JunYoung Gwak, and Silvio Savarese. “4D spatio-temporal convnets: Minkowski convolutional neural networks”. In: *CVPR*. 2019, pp. 3075–3084.

[31] Andrew I. Comport, Éric Marchand, and François Chaumette. “Robust model-based tracking for robot vision”. In: *IROS*. IEEE, 2004, pp. 692–697.

[32] Yubo Cui et al. “3D Object Tracking with Transformer”. In: *arXiv preprint arXiv:2110.14921* (2021).

[33] Yutao Cui et al. “MixFormer: End-to-End Tracking with Iterative Mixed Attention”. In: *CVPR*. IEEE, 2022, pp. 13598–13608.

[34] Jifeng Dai, Kaiming He, and Jian Sun. “BoxSup: Exploiting Bounding Boxes to Supervise Convolutional Networks for Semantic Segmentation”. In: *ICCV*. IEEE Computer Society, 2015, pp. 1635–1643.

[35] Martin Danelljan, Luc Van Gool, and Radu Timofte. “Probabilistic regression for visual tracking”. In: *CVPR*. 2020, pp. 7183–7192.

[36] Martin Danelljan et al. “ATOM: Accurate Tracking by Overlap Maximization”. In: *CVPR*. 2019.

[37] Martin Danelljan et al. “ECO: Efficient Convolution Operators for Tracking”. In: *CVPR*. 2017, pp. 6638–6646.

[38] Ping Ding and Yan Song. “Robust object tracking using color and depth images with a depth based occlusion handling and recovery”. In: *International Conference on Fuzzy Systems and Knowledge Discovery*. 2016.

[39] Dawei Du et al. “The unmanned aerial vehicle benchmark: Object detection and tracking”. In: *ECCV*. 2018, pp. 370–386.

[40] Christer Ericson. *Real-time collision detection*. Crc Press, 2004.

[41] Mark Everingham et al. “The Pascal Visual Object Classes (VOC) Challenge”. In: *International Journal of Computer Vision* 88.2 (2010), pp. 303–338.

[42] Heng Fan et al. “Lasot: A high-quality benchmark for large-scale single object tracking”. In: *CVPR*. 2019, pp. 5374–5383.

[43] Xiaoxue Feng et al. “Student T-Based Maximum Correntropy Unscented Kalman Filter for UAV Target Tracking”. In: *Unmanned Systems* 11.4 (2023), pp. 287–300.

[44] Changhong Fu et al. “Onboard Real-Time Aerial Tracking With Efficient Siamese Anchor Proposal Network”. In: *IEEE Transactions on Geoscience and Remote Sensing* 60 (2021), pp. 1–13.

[45] Changhong Fu et al. “Real-time adaptive multi-classifier multi-resolution visual tracking framework for unmanned aerial vehicles”. In: *IFAC Workshops* 46.30 (2013), pp. 99–106.

[46] Changhong Fu et al. “Robust real-time vision-based aircraft tracking from unmanned aerial vehicles”. In: *ICRA*. IEEE. 2014, pp. 5441–5446.

[47] Changhong Fu et al. “Siamese Object Tracking for Unmanned Aerial Vehicle: A Review and Comprehensive Analysis”. In: *arXiv preprint arXiv:2205.04281* (2022).

[48] Huazhu Fu, Dong Xu, and Stephen Lin. “Object-Based Multiple Foreground Segmentation in RGBD Video”. In: *IEEE Transactions on Image Processing* 26.3 (2017), pp. 1418–1427.

[49] Jörg Gamerdinger et al. “Analyzing track management strategies for multi object tracking in cooperative autonomous driving scenarios”. In: *Autom.* 71.4 (2023), pp. 287–293.

[50] Mingqi Gao et al. “Deep learning for video object segmentation: a review”. In: *Artificial Intelligence Review* 56.1 (2023), pp. 457–531.

[51] Peng Gao et al. “Clip-adapter: Better vision-language models with feature adapters”. In: *arXiv preprint arXiv:2110.04544* (2021).

[52] Shang Gao et al. “Learning Dual-Fused Modality-Aware Representations for RGBD Tracking”. In: *arXiv preprint arXiv:2211.03055* (2022).

[53] Shanghua Gao et al. “Res2Net: A New Multi-Scale Backbone Architecture”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.2 (2021), pp. 652–662.

[54] GM García et al. “Adaptive multi-cue 3D tracking of arbitrary objects.” In: *lecture notes in computer science* 7476 (2012), pp. 357–366.

[55] Andreas Geiger, Philip Lenz, and Raquel Urtasun. “Are we ready for autonomous driving? the kitti vision benchmark suite”. In: *CVPR*. IEEE. 2012, pp. 3354–3361.

[56] Silvio Giancola, Jesus Zarzar, and Bernard Ghanem. “Leveraging Shape Completion for 3D Siamese Tracking”. In: *CVPR*. June 2019.

[57] Brent A. Griffin and Jason J. Corso. “BubbleNets: Learning to Select the Guidance Frame in Video Object Segmentation by Deep Sorting Frames”. In: *CVPR*. Computer Vision Foundation / IEEE, 2019, pp. 8914–8923.

[58] Alexander Gutev and Carl James Debono. “Exploiting depth information to increase object tracking robustness”. In: *EUROCON*. IEEE, 2019, pp. 1–5.

[59] Arun Hampapur et al. “Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking”. In: *IEEE Signal Processing Magazine* 22.2 (2005), pp. 38–51.

[60] Sion Hannuna et al. “DS-KCF: a real-time tracker for RGB-D data”. In: *Journal of Real-Time Image Processing* 16.5 (2016), pp. 1–20.

[61] Soma Hazra et al. “UMTSS: a unifocal motion tracking surveillance system for multi-object tracking in videos”. In: *Multimedia Tools and Applications* 82.8 (2023), pp. 12401–12422.

[62] Botao He et al. “FAST-Dynamic-Vision: Detection and Tracking Dynamic Objects with Event and Depth Sensing”. In: *IROS*. IEEE, 2021, pp. 3071–3078.

[63] Kaiming He et al. “Deep residual learning for image recognition”. In: *CVPR*. 2016, pp. 770–778.

[64] Kaiming He et al. “Masked autoencoders are scalable vision learners”. In: *CVPR*. 2022, pp. 16000–16009.

[65] João F. Henriques et al. “High-Speed Tracking with Kernelized Correlation Filters”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37 (2015), pp. 583–596.

[66] Derek Hoiem, Santosh K Divvala, and James H Hays. “Pascal VOC 2008 Challenge”. In: *World Literature Today* (2009).

[67] Lingyi Hong et al. “Adaptive Selection of Reference Frames for Video Object Segmentation”. In: *IEEE Transactions on Image Processing* 31 (2022), pp. 1057–1071.

[68] Weiming Hu et al. “SiamMask: A Framework for Fast Online Object Tracking and Segmentation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45.3 (2023), pp. 3072–3089.

[69] Yuan-Ting Hu, Jia-Bin Huang, and Alexander G. Schwing. “VideoMatch: Matching Based Video Object Segmentation”. In: *ECCV*. Vol. 11212. Lecture Notes in Computer Science. Springer, 2018, pp. 56–73.

[70] Lianghua Huang, Xin Zhao, and Kaiqi Huang. “GOT-10k: A Large High-Diversity Benchmark for Generic Object Tracking in the Wild”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2019), pp. 1–1. ISSN: 1939-3539.

[71] Ziyuan Huang et al. “Learning aberrance repressed correlation filters for real-time UAV tracking”. In: *ICCV*. 2019, pp. 2891–2900.

[72] Shi Huizhang, Gao Changxin, and Sang Nong. “Using consistency of depth gradient to improve visual tracking in RGB-D sequences”. In: *Chinese Automation Congress*. 2016.

[73] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. “FusionSeg: Learning to Combine Motion and Appearance for Fully Automatic Segmentation of Generic Objects in Videos”. In: *CVPR*. IEEE Computer Society, 2017, pp. 2117–2126.

[74] Sajid Javed et al. “Visual Object Tracking with Discriminative Filters and Siamese Networks: A Survey and Outlook”. In: *CoRR* abs/2112.02838 (2021).

[75] Menglin Jia et al. “Visual Prompt Tuning”. In: *arXiv preprint arXiv:2203.12119* (2022).

[76] Mingxin Jiang et al. “Hierarchical multi-modal fusion FCN with attention model for RGB-D tracking”. In: *Information Fusion* 50 (2019), pp. 1–8.

[77] He Kaiming et al. “Deep residual learning for image recognition”. In: *CVPR*. 2016, pp. 770–778.

[78] Ugur Kart, Joni-Kristian Kämäräinen, and Jiri Matas. “How to Make an RGBD Tracker?” In: *ECCV Workshops*. Vol. 11129. Lecture Notes in Computer Science. Springer, 2018, pp. 148–161.

[79] Ugur Kart et al. “Depth Masked Discriminative Correlation Filter”. In: *ICPR*. 2018, pp. 2112–2117.

[80] Ugur Kart et al. “Object tracking by reconstruction with view-specific discriminative correlation filters”. In: *CVPR*. 2019, pp. 1339–1348.

[81] Hoel Kervadec et al. “Bounding boxes for weakly supervised segmentation: Global constraints get close to full supervision”. In: *MIDL*. Vol. 121. Proceedings of Machine Learning Research. PMLR, 2020, pp. 365–381.

[82] Anna Khoreva et al. “Simple Does It: Weakly Supervised Instance and Semantic Segmentation”. In: *CVPR*. IEEE Computer Society, 2017, pp. 1665–1674.

[83] Hamed Kiani Galoogahi, Terence Sim, and Simon Lucey. “Correlation filters with limited boundaries”. In: *CVPR*. 2015, pp. 4630–4638.

[84] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[85] Matej Kristan et al. “The eighth visual object tracking VOT2020 challenge results”. In: *ECCV*. Springer. 2020, pp. 547–601.

[86] Matej Kristan et al. “The Ninth Visual Object Tracking VOT2021 Challenge Results”. In: *ICCV*. 2021, pp. 2711–2738.

[87] Matej Kristan et al. “The seventh visual object tracking vot2019 challenge results”. In: *ICCV Workshops*. 2019, pp. 0–0.

[88] Matej Kristan et al. “The Tenth Visual Object Tracking VOT2022 Challenge Results”. In: *ECCV Workshops (8)*. Vol. 13808. Lecture Notes in Computer Science. Springer, 2022, pp. 431–460.

[89] Matej Kristan et al. “The Visual Object Tracking VOT2014 challenge results”. In: *ECCV Workshops* 8926 (2014), pp. 191–217.

[90] Matej Kristan et al. “The Visual Object Tracking VOT2015 challenge results”. In: *ICCV*. 2015.

[91] Matej Kristan et al. “The Visual Object Tracking VOT2017 Challenge Results”. In: *ICCV Workshops*. IEEE Computer Society, 2017, pp. 1949–1972.

[92] Yangliu Kuai et al. “Target-Aware Correlation Filter Tracking in RGBD Videos”. In: *IEEE Sensors Journal* 19.20 (2019), pp. 9522–9531.

[93] Victor S. Lempitsky et al. “Image segmentation with a bounding box prior”. In: *ICCV*. IEEE Computer Society, 2009, pp. 277–284.

[94] Jiaxu Leng and Ying Liu. “Real-time RGB-D Visual Tracking with Scale Estimation and Occlusion Handling”. In: *IEEE Access* (2018), pp. 1–1.

[95] Bo Li et al. “High Performance Visual Tracking with Siamese Region Proposal Network”. In: *CVPR*. 2018.

[96] Bo Li et al. “Siamrpn++: Evolution of siamese visual tracking with very deep networks”. In: *CVPR*. 2019, pp. 4282–4291.

[97] Chenglong Li et al. “LasHeR: A Large-Scale High-Diversity Benchmark for RGBT Tracking”. In: *IEEE Transactions on Image Processing* 31 (2022), pp. 392–404.

[98] Chenglong Li et al. “RGB-T object tracking: Benchmark and baseline”. In: *Pattern Recognition* 96 (2019), p. 106977.

[99] E Li et al. “SUSTech POINTS: A Portable 3D Point Cloud Interactive Annotation Platform System”. In: *2020 IEEE Intelligent Vehicles Symposium*. 2020, pp. 1108–1115.

[100] Guanqun Li et al. “Depth information aided constrained correlation filter for visual tracking”. In: *IOP Conference Series: Earth and Environmental Science*. Vol. 234. 1. IOP Publishing. 2019, p. 012005.

[101] Mingxing Li et al. “Recurrent Dynamic Embedding for Video Object Segmentation”. In: *CVPR*. 2022, pp. 1332–1341.

[102] Peiliang Li, Tong Qin, and Shaojie Shen. “Stereo Vision-Based Semantic 3D Object and Ego-Motion Tracking for Autonomous Driving”. In: *ECCV*. Vol. 11206. Lecture Notes in Computer Science. Springer, 2018, pp. 664–679.

[103] Yi Li et al. “Fully Convolutional Instance-Aware Semantic Segmentation”. In: *CVPR*. IEEE Computer Society, 2017, pp. 4438–4446.

[104] Yiming Li et al. “AutoTrack: Towards high-performance visual tracking for UAV with automatic spatio-temporal regularization”. In: *CVPR*. 2020, pp. 11923–11932.

[105] Bingyan Liao et al. “Pg-net: Pixel to global matching network for visual tracking”. In: *ECCV*. Springer. 2020, pp. 429–444.

[106] Tsung-Yi Lin et al. “Microsoft coco: Common objects in context”. In: *ECCV*. Springer. 2014, pp. 740–755.

[107] Pengfei Liu et al. “Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing”. In: *arXiv preprint arXiv:2107.13586* (2021).

[108] Weichun Liu, Xiaoan Tang, and Chenglin Zhao. “Robust RGBD Tracking via Weighted Convolution Operators”. In: *IEEE Sensors Journal* 20.8 (2020), pp. 4496–4503.

[109] Ye Liu et al. “Context-aware three-dimensional mean-shift with occlusion handling for robust object tracking in RGB-D videos”. In: *IEEE Transactions on Multimedia* 21.3 (2018), pp. 664–677.

[110] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).

[111] Andong Lu et al. “Duality-gated mutual condition network for RGBT tracking”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022).

[112] Alan Lukezic, Jiri Matas, and Matej Kristan. “D3S - A Discriminative Single Shot Segmentation Tracker”. In: *CVPR*. Computer Vision Foundation / IEEE, 2020, pp. 7131–7140.

[113] Alan Lukezic et al. “Cdtb: A color and depth visual object tracking dataset and benchmark”. In: *ICCV*. 2019, pp. 10013–10022.

[114] Alan Lukezic et al. “Discriminative correlation filter with channel and spatial reliability”. In: *CVPR*. 2017, pp. 6309–6318.

[115] Alan Lukezic et al. “Performance Evaluation Methodology for Long-Term Single-Object Tracking”. In: *IEEE Transactions on Cybernetics* 51.12 (2021), pp. 6305–6318.

[116] Chenxu Luo, Xiaodong Yang, and Alan L. Yuille. “Exploring Simple 3D Multi-Object Tracking for Autonomous Driving”. In: *ICCV*. IEEE, 2021, pp. 10468–10477.

[117] Wenjie Luo, Bin Yang, and Raquel Urtasun. “Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net”. In: *CVPR*. 2018, pp. 3569–3577.

[118] Chao Ma et al. “Adaptive Correlation Filters with Long-Term and Short-Term Memory for Object Tracking”. In: *International Journal of Computer Vision* 126.8 (2018), pp. 771–796.

[119] Cong Ma et al. “Deep Human-Interaction and Association by Graph-Based Learning for Multiple Object Tracking in the Wild”. In: *International Journal of Computer Vision* 129.6 (2021), pp. 1993–2010.

[120] Zi-ang Ma and Zhiyu Xiang. “Robust object tracking with RGBD-based sparse learning”. In: *Frontiers of Information Technology and Electronic Engineering* (2017).

[121] Eiji Machida et al. “Human motion tracking of mobile robot with Kinect 3D sensor”. In: *SICE*. IEEE. 2012, pp. 2207–2211.

[122] Madjid Maidi et al. “Vision-based tracking in large image database for real-time mobile augmented reality”. In: *MMSP*. IEEE, 2014, pp. 1–6.

[123] Kevis-Kokitsi Maninis et al. “Video Object Segmentation Without Temporal Information”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2018), pp. 1–1.

[124] Christoph Mayer et al. “Learning target candidate association to keep track of what not to track”. In: *ICCV*. 2021, pp. 13444–13454.

[125] Kourosh Meshgi et al. “An occlusion-aware particle filter tracker to handle complex and persistent occlusions”. In: *Computer Vision and Image Understanding* 150 (2016), pp. 81–94. ISSN: 1077-3142.

[126] Matthias Mueller, Neil Smith, and Bernard Ghanem. “A Benchmark and Simulator for UAV Tracking”. In: *ECCV*. Vol. 9905. Lecture Notes in Computer Science. Springer, 2016, pp. 445–461.

[127] Matthias Müller et al. “TrackingNet: A Large-Scale Dataset and Benchmark for Object Tracking in the Wild”. In: *ECCV*. 2018.

[128] Hyeonseob Nam and Bohyung Han. “Learning Multi-Domain Convolutional Neural Networks for Visual Tracking”. In: *IEEE* (2016).

[129] Seoung Wug Oh et al. “Fast Video Object Segmentation by Reference-Guided Mask Propagation”. In: *CVPR*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 7376–7385.

[130] Seoung Wug Oh et al. “Video Object Segmentation Using Space-Time Memory Networks”. In: *ICCV*. IEEE, 2019, pp. 9225–9234.

[131] Federico Perazzi et al. “A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation”. In: *CVPR*. IEEE Computer Society, 2016, pp. 724–732.

[132] Federico Perazzi et al. “Learning Video Object Segmentation from Static Images”. In: *CVPR*. IEEE Computer Society, 2017, pp. 3491–3500.

[133] Jordi Pont-Tuset et al. “Multiscale Combinatorial Grouping for Image Segmentation and Object Proposal Generation”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.1 (2017), pp. 128–140.

[134] Charles R Qi et al. “Deep hough voting for 3d object detection in point clouds”. In: *ICCV*. 2019, pp. 9277–9286.

[135] Charles R Qi et al. “Pointnet: Deep learning on point sets for 3d classification and segmentation”. In: *CVPR*. 2017, pp. 652–660.

[136] Charles R Qi et al. “Pointnet++: Deep hierarchical feature learning on point sets in a metric space”. In: *arXiv preprint arXiv:1706.02413* (2017).

[137] Haozhe Qi et al. “P2B: Point-to-Box Network for 3D Object Tracking in Point Clouds”. In: *CVPR*. June 2020.

[138] Yanlin Qian et al. “DAL: A Deep Depth-aware Long-term Tracker”. In: *ICPR*. 2020.

[139] Alec Radford et al. “Learning transferable visual models from natural language supervision”. In: *ICML*. PMLR. 2021, pp. 8748–8763.

[140] Hamid Rezatofighi et al. “Generalized intersection over union: A metric and a loss for bounding box regression”. In: *CVPR*. 2019, pp. 658–666.

[141] Florian Richter et al. “Robotic Tool Tracking Under Partially Visible Kinematic Chain: A Unified Approach”. In: *IEEE Transactions Robotics* 38.3 (2022), pp. 1653–1670.

[142] Konstantin Röhrl et al. “TrackAgent: 6D Object Tracking via Reinforcement Learning”. In: *CoRR* abs/2307.15671 (2023).

[143] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *MICCAI*. Vol. 9351. Lecture Notes in Computer Science. Springer, 2015, pp. 234–241.

[144] Germán Ros et al. “Vision-Based Offline-Online Perception Paradigm for Autonomous Driving”. In: *WACV*. IEEE Computer Society, 2015, pp. 231–238.

[145] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. “"GrabCut": interactive foreground extraction using iterated graph cuts”. In: *ACM Transactions on Graphics* 23.3 (2004), pp. 309–314.

[146] Hongje Seong et al. “Hierarchical memory matching network for video object segmentation”. In: *ICCV*. 2021, pp. 12889–12898.

[147] Zahra Soleimanitaleb and Mohammad Ali Keyvanrad. “Single Object Tracking: A Survey of Methods, Datasets, and Evaluation Metrics”. In: (2022).

[148] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. “Sun rgb-d: A rgb-d scene understanding benchmark suite”. In: *CVPR*. 2015, pp. 567–576.

[149] Shuran Song and Jianxiong Xiao. “Tracking revisited using RGBD camera: Unified benchmark and baselines”. In: *ICCV*. 2013, pp. 233–240.

[150] Catherine Taylor, Robin McNicholas, and Darren Cosker. “Towards An Egocentric Framework for Rigid and Articulated Object Tracking in Virtual Reality”. In: *2020 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. 2020, pp. 354–359.

[151] Hugo Touvron et al. “Training data-efficient image transformers & distillation through attention”. In: *arXiv preprint arXiv:2012.12877* (2020).

[152] Paul Voigtlaender and Bastian Leibe. “Online Adaptation of Convolutional Neural Networks for Video Object Segmentation”. In: *BMVC*. BMVA Press, 2017.

[153] Paul Voigtlaender et al. “FEELVOS: Fast End-to-End Embedding Learning for Video Object Segmentation”. In: *CVPR*. 2019.

[154] He Wang et al. “Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation”. In: *CVPR*. June 2019.

[155] Kangkan Wang, Guofeng Zhang, and Shihong Xia. “Templateless Non-Rigid Reconstruction and Motion Tracking With a Single RGB-D Camera”. In: *IEEE Transactions on Image Processing* 26.12 (2017), pp. 5966–5979.

[156] Ning Wang et al. “Unsupervised deep tracking”. In: *CVPR*. 2019, pp. 1308–1317.

[157] Qi Wang, Jianwu Fang, and Y. Yuan. “Multi-cue based tracking”. In: (2014).

[158] Qiang Wang et al. “Fast online object tracking and segmentation: A unifying approach”. In: *CVPR*. 2019, pp. 1328–1338.

[159] Wenguan Wang et al. “A survey on deep learning technique for video segmentation”. In: *arXiv preprint arXiv:2107.01153* (2021).

[160] Xiao Wang et al. “VisEvent: Reliable Object Tracking via Collaboration of Frame and Event Flows”. In: *arXiv:2108.05015* (2021).

[161] Yong Wang et al. “Robust fusion for RGB-D tracking using CNN features”. In: *Applied Soft Computing* 92 (2020), p. 106302.

[162] Zhoutao Wang et al. “MLVSNet: Multi-Level Voting Siamese Network for 3D Visual Tracking”. In: *ICCV*. 2021, pp. 3101–3110.

[163] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. “Object Tracking Benchmark”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), pp. 1834–1848.

[164] Feng Xiao, Qiuxia Wu, and Han Huang. “Single-scale siamese network based RGB-D object tracking with adaptive bounding boxes”. In: *Neurocomputing* 451 (2021), pp. 192–204.

[165] Huaxin Xiao et al. “MoNet: Deep Motion Exploitation for Video Object Segmentation”. In: *CVPR*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 1140–1148.

[166] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. “SUN3D: A Database of Big Spaces Reconstructed Using SfM and Object Labels”. In: *ICCV*. IEEE Computer Society, 2013, pp. 1625–1632.

[167] Jingjing Xiao et al. “Robust fusion of color and depth data for RGB-D target tracking using adaptive range-invariant depth models and spatio-temporal consistency constraints”. In: *IEEE Transactions on Cybernetics* 48.8 (2017), pp. 2485–2499.

[168] Haozhe Xie et al. “Efficient regional memory network for video object segmentation”. In: *CVPR*. 2021, pp. 1286–1295.

[169] Yujun Xie, Yao Lu, and Shuang Gu. “RGB-D Object Tracking with Occlusion Detection”. In: *2019 15th International Conference on Computational Intelligence and Security*. IEEE. 2019, pp. 11–15.

[170] Ning Xu et al. “YouTube-VOS: A Large-Scale Video Object Segmentation Benchmark”. In: *CoRR* abs/1809.03327 (2018).

[171] Zhengtian Xu et al. “Outdoor RGBD Instance Segmentation With Residual Regretting Learning”. In: *IEEE Transactions on Image Processing* 29 (2020), pp. 5301–5309.

[172] Bin Yan et al. “Alpha-Refine: Boosting Tracking Performance by Precise Bounding Box Estimation”. In: *arXiv preprint arXiv:2007.02024* (2020).

[173] Bin Yan et al. “Learning spatio-temporal transformer for visual tracking”. In: *ICCV*. 2021, pp. 10448–10457.

[174] Bin Yan et al. “LightTrack: Finding Lightweight Neural Networks for Object Tracking via One-Shot Architecture Search”. In: *CVPR*. June 2021.

[175] Song Yan et al. “DepthTrack: Unveiling the Power of RGBD Tracking”. In: *ICCV*. 2021, pp. 10725–10733.

[176] Xu Yan et al. “Pointasnl: Robust point clouds processing using nonlocal neural networks with adaptive sampling”. In: *CVPR*. 2020, pp. 5589–5598.

[177] Hanxuan Yang et al. “Recent advances and trends in visual tracking: A review”. In: *Neurocomputing* 74.18 (2011), pp. 3823–3831.

[178] Jinyu Yang et al. “A Saliency-Based Object Tracking Method for UAV Application”. In: *PRCV*. Vol. 11259. Lecture Notes in Computer Science. Springer, 2018, pp. 115–125.

[179] Jinyu Yang et al. “Prompting for Multi-Modal Tracking”. In: *ACMMM*. 2022, pp. 3492–3500.

[180] Jinyu Yang et al. “Resource-Efficient RGBD Aerial Tracking”. In: *CVPR*. IEEE, 2023, pp. 13374–13383.

[181] Jinyu Yang et al. “RGBD Object Tracking: An In-depth Review”. In: *arXiv preprint arXiv:2203.14134* (2022).

[182] Jinyu Yang et al. “Towards Generic 3D Tracking in RGBD Videos: Benchmark and Baseline”. In: *ECCV (22)*. Vol. 13682. Lecture Notes in Computer Science. Springer, 2022, pp. 112–128.

[183] Linjie Yang, Yuchen Fan, and Ning Xu. “Video instance segmentation”. In: *ICCV*. 2019, pp. 5188–5197.

[184] Xu Yang et al. “Object-Agnostic Transformers for Video Referring Segmentation”. In: *IEEE Transactions on Image Processing* 31 (2022), pp. 2839–2849.

[185] Zongxin Yang, Yunchao Wei, and Yi Yang. “Associating objects with transformers for video object segmentation”. In: *NeurIPS*. 2021, pp. 2491–2502.

[186] Zongxin Yang, Yunchao Wei, and Yi Yang. “Collaborative video object segmentation by multi-scale foreground-background integration”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

[187] Yuan Yao et al. “Cpt: Colorful prompt tuning for pre-trained vision-language models”. In: *arXiv preprint arXiv:2109.11797* (2021).

[188] Botao Ye et al. “Joint Feature Learning and Relation Modeling for Tracking: A One-Stream Framework”. In: *ECCV*. Vol. 13682. Lecture Notes in Computer Science. Springer, 2022, pp. 341–357.

[189] Junjie Ye et al. “Unsupervised domain adaptation for nighttime aerial tracking”. In: *CVPR*. 2022, pp. 8896–8905.

[190] Jae Shin Yoon et al. “Pixel-Level Matching for Video Object Segmentation Using Convolutional Neural Networks”. In: *ICCV*. IEEE Computer Society, 2017, pp. 2186–2195.

[191] Yayu Zhai et al. “Occlusion-Aware Correlation Particle Filter Target Tracking Based on RGBD Data”. In: *IEEE Access* 6 (2018), pp. 50752–50764.

[192] Chunhui Zhang et al. “WebUAV-3M: A Benchmark Unveiling the Power of Million-Scale Deep UAV Tracking”. In: *arXiv preprint arXiv:2201.07425* (2022).

[193] Han Zhang, Meng Cai, and Jianxun Li. “A Real-time RGB-D tracker based on KCF”. In: *CCDC*. IEEE. 2018, pp. 4856–4861.

[194] Pengyu Zhang et al. “Visible-Thermal UAV Tracking: A Large-Scale Benchmark and New Baseline”. In: *CVPR*. 2022, pp. 8876–8885.

[195] Wenli Zhang et al. “An Occlusion-Aware RGB-D Visual Object Tracking Method Based on Siamese Network”. In: *ICSP*. Vol. 1. IEEE. 2020, pp. 327–332.

[196] Yongchang Zhang et al. “3D Single-Object Tracking with Spatial-Temporal Data Association”. In: *IROS*. IEEE, 2022, pp. 264–269.

[197] Yucheng Zhang et al. “Recent advances of single-object tracking methods: A brief survey”. In: *Neurocomputing* 455 (2021), pp. 1–11.

[198] Zhipeng Zhang and Houwen Peng. “Deeper and wider siamese networks for real-time visual tracking”. In: *CVPR*. 2019, pp. 4591–4600.

[199] Bin Zhao et al. “Generating Masks from Boxes by Mining Spatio-Temporal Consistencies in Videos”. In: *ICCV*. IEEE, 2021, pp. 13536–13546.

[200] Pengyao Zhao et al. “TSDM: Tracking by SiamRPN++ with a Depth-refiner and a Mask-generator”. In: *ICPR*. IEEE. 2021, pp. 670–676.

[201] Chaoda Zheng et al. “Box-Aware Feature Enhancement for Single Object Tracking on Point Clouds”. In: *ICCV*. 2021, pp. 13199–13208.

[202] Wei-Long Zheng, Shan-Chun Shen, and Bao-Liang Lu. “Online depth image-based object tracking with sparse representation and object detection”. In: *Neural Processing Letters* 45.3 (2017), pp. 745–758.

[203] Bineng Zhong et al. “Online learning 3D context for robust visual tracking”. In: *Neurocomputing* 151 (2015), pp. 710–718.

[204] Kaiyang Zhou et al. “Learning to prompt for vision-language models”. In: *arXiv preprint arXiv:2109.01134* (2021).

[205] Tao Zhou et al. “Specificity-preserving RGB-D Saliency Detection”. In: *ICCV*. IEEE, 2021, pp. 4661–4671.

[206] Yanzhao Zhou et al. “Weakly Supervised Instance Segmentation Using Class Peak Response”. In: *CVPR*. Computer Vision Foundation / IEEE Computer Society, 2018, pp. 3791–3800.

[207] Jiawen Zhu et al. “Visual Prompt Multi-Modal Tracking”. In: *CoRR* abs/2303.10826 (2023).

[208] Pengfei Zhu et al. “Detection and Tracking Meet Drones Challenge”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021), pp. 1–1.

[209] Xuefeng Zhu et al. “RGBD1K: A Large-scale Dataset and Benchmark for RGB-D Object Tracking”. In: *CoRR* abs/2208.09787 (2022).

[210] Zheng Zhu et al. “Distractor-aware siamese networks for visual object tracking”. In: *ECCV*. 2018, pp. 101–117.

[211] Safa Ziadi and Mohamed Njah. “Pso-Dvsf2-MT: an Optimized Mobile robot motion Planning Approach for tracking Moving targets”. In: *International Journal of Robotics and Automation* 37.5 (2022).

[212] Hao Zou et al. “F-Siamese Tracker: A Frustum-based Double Siamese Network for 3D Single Object Tracking”. In: *IROS*. IEEE. 2020, pp. 8133–8139.