An Approximate Dynamic Programming Framework for Occlusion-Robust Multi-Object Tracking

by

Pratyusha Musunuru

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2024 by the
Graduate Supervisory Committee:

Dimitri Bertsekas, Co-Chair
Subbarao Kambhampati, Co-Chair
Andréa Richa

ARIZONA STATE UNIVERSITY

May 2024

ABSTRACT

In this work, the problem of multi-object tracking (MOT) is studied, particularly the challenges that arise from object occlusions. A solution based on a principled approximate dynamic programming approach called ADPTrack is presented. ADPTrack relies on existing MOT solutions and directly improves them. When matching tracks to objects at a particular frame, the proposed approach simulates executions of these existing solutions into future frames to obtain approximate track extensions, from which a comparison of past and future appearance feature information is leveraged to improve overall robustness to occlusion-based error. The proposed solution when applied to the renowned MOT17 dataset empirically demonstrates a 0.7% improvement in the association accuracy (IDF1 metric) over a state-of-the-art baseline that it builds upon while obtaining minor improvements with respect to all other metrics. Moreover, it is shown that this improvement is even more pronounced in scenarios where the camera maintains a fixed position. This implies that the proposed method is effective in addressing MOT issues pertaining to object occlusions.

DEDICATION

To my loving parents and my dear partner, Varshit.

ACKNOWLEDGMENTS

I would like to take this opportunity to express my sincere gratitude to everyone who helped me go through the two years of my graduate studies.

First and foremost, I would like to thank my advisor, Prof. Dimitri P. Bertsekas who has been a pillar of support to me. He has always been patient with me throughout, whether it was during the problem-statement search or the implementation phases. His willingness to invest time and effort into my academic growth has been instrumental in motivating me to improve my research aptitude. His ideas, guidance, and feedback always helped open a new perspective during discussions. Talking to Prof. Dimitri has always been informal whether it is about problem-solving or about life. He has always been understanding and available when I wanted to communicate with him. He provided me with an intellectually stimulating and comfortable environment to discuss concepts and issues I was facing, work, and enjoy as well. I would not have had a great master's research experience without his mentorship.

Dr. Yuchao Li and Dr. Jamison W. Weber were my first points of contact to get any clarifications, whether regarding better problem-solving approaches, correlating implementations to theoretical foundations, or writing a thesis. They played a crucial role in helping me shape into a better research scholar. Their insightful feedback and willingness to engage in discussions greatly contributed to the clarity and precision of my work. I'm grateful for this pivotal mentorship that Yuchao and Jamison have provided, without which this thesis would not be possible. Moreover, I would like to specifically thank Yuchao for his constructive criticism that helped me grow throughout my research journey and Jamison for his invaluable guidance on structuring and writing a research paper, which has helped me significantly improve my outlook in this area.

Putting my academic situation aside, my landlady Marie Nichols has been like

a second family away from home. Her genuine concern, willingness to help, and emotional support have made a huge difference in my journey.

I would like to thank my friends Lalith, Anoop, Sandeep, Suraj, Mohammad, Anirudh, Srikar, and Anurag who were always available for support, encouragement, trips, deadlines, and many other things to help create a sense of belonging that is essential for navigating this academic journey successfully. I would like to specifically thank Lalith and Sandeep for teaching me how to drive and helping me while moving, Anoop for his detailed explanations, Suraj for helping me with the thesis formalities, Mohammad for teaching me to cook yummy dishes, and Srikar and Anurag for coming to my rescue in times of need.

I am grateful to my parents (Anup, Vani), sister (Bhavana), grandparents, and my cats for supporting me greatly during this journey. I would like to thank my extended family for helping me explore various aspects of life during this journey. I would like to specifically thank my uncle (Prasad) and aunt (Madhavi) for being incredibly supportive of me. This thesis marks success for everyone who is a major part of my life.

Lastly, I am extremely grateful to my partner Varshit for his unwavering support and understanding during this academic journey. His presence and attentive care made this journey significantly smoother and enjoyable.

TABLE OF CONTENTS

LIST OF TABLES

Table Page

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Chapter 1

INTRODUCTION

The work presented in this thesis is the result of the joint efforts of Pratyusha Musunuru, Jamison W. Weber, Yuchao Li, and Dimitri P. Bertsekas. Pratyusha Musunuru was the lead researcher for this project.

In this work, we consider the problem of *multi-object tracking* (MOT). Informally, MOT is a special case of the multidimensional assignment problem applied to data association [Emami et al. 2020]. It involves the assignment of identifiers to objects in motion over a sequence of image frames. However, the task of ensuring a consistent assignment remains a formidable challenge. In particular, for scenarios where objects become occluded, where shadows temporally affect the illumination of objects, where the objects themselves alter in appearance, or in the presence of sensor error. To overcome the occlusion challenge, we introduce a method based on approximate dynamic programming (also known as reinforcement learning) techniques that improves over an arbitrary online tracking algorithm. Online tracking algorithms match frame-to-frame and do not use any subsequent frames' information while matching the current frames. We move the online tracking algorithm to a near-online (i.e., considering a limited amount of future information) setting, where we use a near-online simulation of subsequent frames and appearance feature comparison of past and future frames. We argue that our more principled approach dramatically reduces error caused by occlusions—namely, identifier swaps, i.e., erroneous assignment of identifiers while maintaining the overall accuracy of existing state-of-the-art solutions.

Many existing state-of-the-art methods for online tracking employ an *object detection module* and a *tracking module* [Yifu Zhang et al. 2022; Aharon, Orfaig, and Bobrovsky 2022; Yang et al. 2023; Q. Wang et al. 2021; Zhao et al. 2022; Du et al. 2023; Qin et al. 2023; Q. Chu et al. 2017; Hyun et al. 2023; Cetintas, Brasó, and Leal-Taixé 2023]. The object-detection module involves the computing of bounding boxes for objects in the frame. The tracking module involves a motion model (e.g. a constant velocity Kalman filter [Bewley et al. 2016]) for state estimation, an appearance model (e.g. a re-identification[1] neural network [He et al. 2020]) for calculating appearance similarities, and a matching algorithm to match the bounding boxes over different frames. Note that at any given frame there may not exist a bijection of tracks to objects, as some objects may disappear or new objects may appear suddenly. In recent publications, there has been a considerable improvement in the methodology used for object detection [Ge et al. 2021; Wang, Bochkovskiy, and Liao 2022; S. Ren et al. 2016]. However, there has been a lack of focus on effectively using subsequent frames to match the current frames. As such, this work presents a methodology for applying existing online state-of-the-art multi-object tracking methods that comprise motion and appearance models as a base heuristic for a reinforcement learning technique called *approximation in value space* [Bertsekas 2023] with the aim to better associate occluded objects through information obtained from online simulations.

1.1 Our Contributions

In this work, we offer a key insight for improved multi-object tracking: Namely that, MOT has an intuitive dynamic programming structure (discussed in Chap-

[1]Appearance features of an object calculated from the image patch of its bounding box

ter 2. Although the problem is difficult to solve exactly, it brings to bear existing approximate dynamic programming techniques for near-optimal solutions of MOT. In particular, approximation in value space is a technique used to approximate the cost-to-go terms of the Bellman equations, and these approximations frequently involve online simulation to compute approximately optimal costs for smaller subproblems. Approximation in value space has substantial theoretical support, which suggests that it may be a promising candidate method for MOT. We demonstrate empirically that an approximation approach using simulation renders existing methods more robust to object occlusions, and therefore may supplement any arbitrary state-of-the-art method. Specifically, we present an MOT tracking algorithm for near-online motion model and appearance feature-based tracking called *ADPTrack*, where ADP refers to approximate dynamic programming. In our experiments, we equip ADPTrack with one of the current state-of-the-art online MOT algorithms at the time of writing—BoT-SORT [Aharon, Orfaig, and Bobrovsky 2022], as a base heuristic. We tested our solution on the MOT17 [Milan et al. 2016] dataset and observed an overall relative improvement of 0.7% in the IDF1 score metric over BoT-SORT with a minor improvement in accuracy (MOTA, HOTA metrics). This implies ADPTrack is useful for reducing false positives, false negatives, ID switches, and fragmentation with respect to the ground truth, and may be applied to any existing method that uses motion models and appearance features. Figure 1 provides a frame-by-frame comparison of BoT-SORT and ADPTrack with a BoT-SORT heuristic for a fixed video example and illustrates occlusion scenarios where ADPTrack improves over existing solutions.

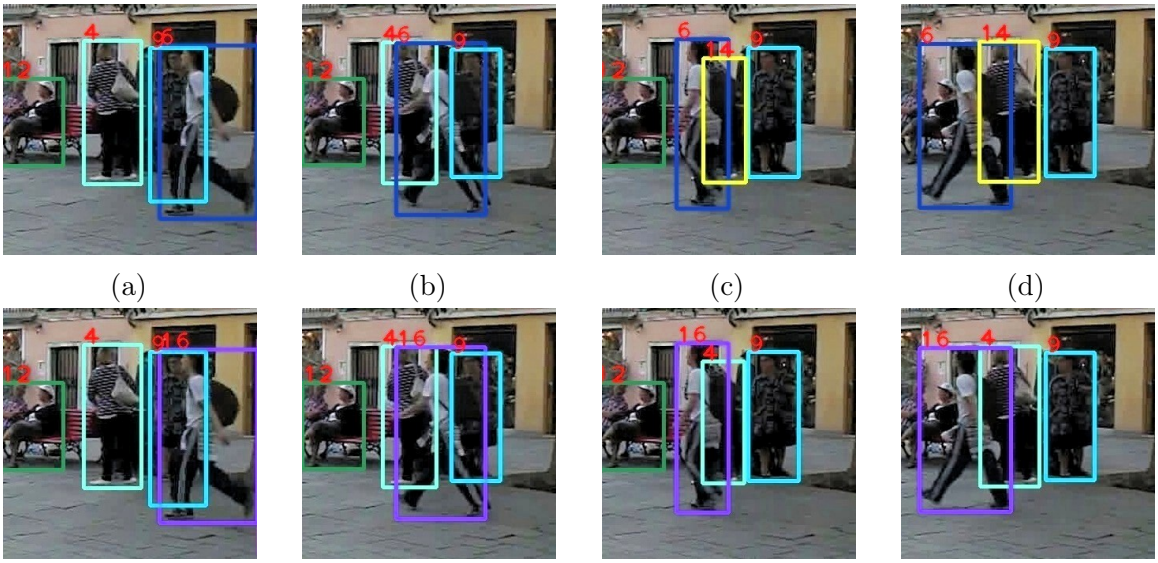

Figure 1. Frame-by-frame comparison of BoT-SORT-Reid (a-d) versus ADPTrack with BoT-SORT-Reid heuristic (e-h) for a fixed video example. With BoT-SORT-Reid alone, person 6 occludes person 4, which causes 4 to be erroneously assigned another person's identifier (14) located between person (4) and person (9). With ADPTrack based on BoT-SORT, the same person (here identified with identifier 16) occludes person 4 without an identifier switch.

1.2 Related Work

In this section, we provide an overview of the state-of-the-art MOT algorithms and focus on literature that most closely relates to our solution. In particular, we contrast our approach with existing near-online and reinforcement learning-based solutions, as our work relates most closely with this intersection.

The algorithms proposed for multi-object tracking can be classified into one of four areas: online, near-online, batch, and offline tracking. Online tracking matches objects present in adjacent frames by using accumulated information from assigned previous frames. Specifically, at any time $k+1$, the online tracking methods perform matching only between the tracks generated to time k and objects at frame $k+1$. Note that

this does not use information from subsequent frames, and as such, the algorithms proposed in this area are considerably faster. Near-online tracking is similar to online tracking, where the tracker (i.e., a tracking algorithm) matches the tracks generated until time k to objects at a frame $k+1$, where only two frames are matched and the algorithm proceeds sequentially temporally. However, the tracker uses information from the past as well as a limited amount of subsequent frames' information. On the other hand, batch tracking matches a set of frames simultaneously using information available from every frame. Offline tracking can be viewed as a special case of batch tracking where the batch consists of the entire frame sequence.

1.2.1 Online Tracking

In this section, we provide an overview of the existing approaches to online tracking. Online trackers typically consist of motion and appearance models, and a matching algorithm. The state at time $k+1$ would consist of existing tracks at time k and new detections at time k. A motion model performs a state estimation of each track for the next time step. This information is compared to the new detections at time t using a localization scoring metric to obtain the distance between the new and estimated objects. An appearance model is used to compute similarity scores between the tracks and the new detections based on appearance. The information from one or both of these models is used to produce a pairwise similarity score between each existing track and new detection. The pairwise similarity scores are computed between the existing tracks and the new detections, and all this data is represented as a cost matrix. An optimal minimum cost bipartite matching algorithm (e.g. Hungarian algorithm [Bewley et al. 2016]) or a greedy algorithm is generally applied to the cost

matrix. It is used to extend the existing tracks by one further frame. Any unmatched tracks may be maintained for some frames (typically given as a parameter) before being discarded, whereas new objects initiate new tracks. We first discuss online tracking methods that involve a motion model. Then, we explore tracking algorithms that use an appearance model to match objects based on appearance features. Lastly, we discuss approaches that use a set of previously matched objects from a track to match a new object.

Many state-of-the-art papers have shown the importance of a motion model, especially in the case of a fast-moving camera [Aharon, Orfaig, and Bobrovsky 2022; Du et al. 2023; Bergmann, Meinhardt, and Leal-Taixe 2019]. A motion model is used to estimate the position and size of an object in the next state using the velocity of the object. An object's velocity (which we refer to from here onward as *motion cues*), position, and size are used to calculate an association cost with a new object using distance metrics. In [Bewley et al. 2016; Yifu Zhang et al. 2022; Aharon, Orfaig, and Bobrovsky 2022], a Kalman filter with a linear-constant velocity model is used for state estimation, and *intersection over union overlap* (IOU overlap) between the predicted bounding box and the detections is used in the estimation of the cost-matrix that is used for an optimal association using Hungarian algorithm or greedy assignment. In [Wojke, Bewley, and Paulus 2017], the motion affinity is estimated by calculating a Mahalanobis distance between the estimated state and the new detections. In [Cao et al. 2023], the authors re-update the Kalman filter parameters and predictions after a lost track is matched again by constructing a virtual trajectory between the previously found detection and the newly matched detection. In [Du et al. 2023], the authors use the Noise-Scale-Adaptive (NSA) Kalman filter [Du et al. 2022] where the detection confidence score weights the update step. In [Yang et al. 2023], instead of

the Kalman filter, the authors estimate the next state using a linear combination of limited previous frames, to account for a change in the object's direction or motion. In [Qin et al. 2023], the authors have an interaction module to calculate interactions between existing tracks and update the next state of every track accordingly using a graph convolution module. In [Bochinski, Eiselein, and Sikora 2017; Bochinski, Senst, and Sikora 2018], the authors do not use an explicit motion model to estimate the next state. They make an assumption that detections in consecutive frames would have a high overlap. In [Bochinski, Senst, and Sikora 2018], the authors use a visual tracker to track the lost tracks for a few frames forward and backward in time, to check and merge the newly generated tracks with their original tracks. In [Bergmann, Meinhardt, and Leal-Taixe 2019], the authors use bounding box regression between the detections to track objects. Similar to these methods, our method also relies on a base heuristic with a motion model to specifically help match objects that are mostly occluded during occlusion.

Several state-of-the-art (SOTA) works adopt various methods to learn the appearance features of the objects [Aharon, Orfaig, and Bobrovsky 2022; Q. Wang et al. 2021; Zhao et al. 2022; Liang et al. 2022; Q. Chu et al. 2017; Cetintas, Brasó, and Leal-Taixé 2023; Bae and Yoon 2018; Yu et al. 2016]. In [Wojke, Bewley, and Paulus 2017], the authors use a cosine distance between the appearance features generated by a deep neural network as trained by the authors on a re-identification dataset and fuse both the motion affinity and the appearance scores. In [L. Chen et al. 2018], the authors use a scoring function based on image classification to obtain candidate tracks and then perform hierarchical association by using appearance features-based similarity scoring on the first set of tracks, and only IOU overlap for the second set of tracks. In [Z. Wang et al. 2020; Yifu Zhang et al. 2021], the authors train a neural

network to jointly learn and infer detections as well as embeddings for appearance features. In [Z. Wang et al. 2020], the authors propose smoothing the representative appearance features of the track with the new detection features using an exponential moving average after a match has been identified between the existing track and the corresponding new detection. In [Leal-Taixé, Ferrer, and Schindler 2016; B. Wang et al. 2016; Yoon et al. 2020; S. Sun et al. 2021; Yin et al. 2020], deep neural networks are used to obtain similarity costs between image patches of bounding boxes of the current frame and next frame detections, which are used to perform associations, without the usage of any motion model for state estimation. In [Aharon, Orfaig, and Bobrovsky 2022], the authors train the SBS50 model from FastReid [He et al. 2020] for appearance features and fuse appearance feature similarities with IOU overlap scores to get association costs and perform hierarchical matching in three phases as adopted from [Yifu Zhang et al. 2022]. We are using the BoT-SORT algorithm as a baseline for our work.

Solutions that focus on track histories demonstrate the importance of considering past frame features while matching objects in the current frame. These proposed solutions were given in [Fang et al. 2018; Sadeghian, Alahi, and Savarese 2017; Xu et al. 2019; Son et al. 2017; Feng, Li, and Ouyang 2022; Hung et al. 2020; Zhu et al. 2019; P. Sun et al. 2021; P. Chu et al. 2021; Guo et al. 2021; Meinhardt et al. 2022; Ma et al. 2022; Cai et al. 2022; Shan et al. 2020; Pang et al. 2021], where the previous frame features are passed in conjunction as input to a deep neural network such as a CNN, RNN, or a transformer to obtain pairwise similarity scores of the existing tracks and the new detections. While this is not how we use the past frame features in our method, these papers demonstrate the importance of the past frame features during association.

1.2.2 Near-online Tracking

Near-online tracking consists of methods that consider a limited number of future and possibly past frames. As the near-online tracking methods consider a limited number of future frames, say ℓ, while matching the current frames k and $k+1$, there is a certain lag (of at least ℓ frames) associated with this type of algorithms. This lag places the camera input at $k+\ell+2$th (or further) frame while processing the matching between frames k and $k+1$. There may also be a certain lag associated with the computation of the matching between the frames k and $k+1$. Overall, these algorithms may not be as fast as online tracking algorithms, but this compromise in real-time tracking is seen as a trade-off for improved tracking accuracy. In [Choi 2015], the authors introduce the concept of near-online multi-object tracking. They construct a future trajectory for every detection by using a flow descriptor, i.e. a heuristic used to assess the likelihood that two temporally distant objects represent the same object. These future trajectories are used to obtain the association between the current frames using a graphical model (conditional random field). In [Feng et al. 2021], the future tracks for a limited number of frames are generated by greedy associations using only appearance features. Then three neural networks are used to generate similarity scores between tracks at a time k and new detections at time $k+1$. These neural networks compare tracks at time k to detections at $k+1$, as well as the track histories of tracks at k to detections at $k+1$, as well as track histories to track hypotheses. Once the costs are generated, a greedy association is performed.

In [Henschel, Zou, and Rosenhahn 2019], the authors present a novel way of handling appearance features using body and joint detections, and the association optimization is modeled as a minimum-cost graph labeling problem. A similar graph

labeling approach that ranks trajectories was used in [Yang Zhang et al. 2020]. Moreover, in [Yang, Wu, and Jia 2017], the authors compartmentalize the problem locally around a time window using a network construction technique, which is then solved as if it were the full problem using multicommodity flow-based optimization techniques. In [Rangesh et al. 2021], the authors perform a near-online data association directly using graph neural networks. In [Fagot-Bouquet et al. 2016], the authors use a statistical physics approach for approximately optimal data association over a sliding window of frames.

In contrast, our approach focuses on improvement over an online base heuristic. We simulate the base heuristic equipped with both motion and appearance models over a limited set of subsequent frames to generate tracklets. We may then leverage any appearance feature association method given by the base heuristic, as well as information from future tracks as generated by the base heuristic. As such, our approach differs from the methods listed here in that it focuses on improving existing solutions via online simulation.

1.2.3 Batch and Offline Tracking

One of the fundamental and well-known batch methods is multi-hypothesis tracking [Reid 1979], which maintains a collection of tree topologies representing track hypotheses. There are several extensions of this method [Kim et al. 2015; Sheng, Chen, et al. 2019; J. Chen et al. 2017; Kim, Li, and Rehg 2018] that introduce pruning and propose hypothesis-scoring techniques. In [Tang et al. 2016], the multi-object tracking problem is formulated as a minimum-cost multicut problem. There are also batch methods that use neural networks for end-to-end data association for a fixed

number of frames [Schulter et al. 2017; Chu and Ling 2019; Zhou et al. 2022; Brasó and Leal-Taixé 2020]. Other offline solutions include [Dehghan, Assari, and Shah 2015; Tang et al. 2017; Sheng, Zhang, et al. 2019; Henschel et al. 2018].

1.2.4 Reinforcement Learning-based Multi-object Tracking

In [Choudhuri, Chowdhary, and Schwing 2021], the authors formulate the multi-object tracking and segmentation problem as a dynamic programming problem. It is a batch method where they consider a set of frames to be matched at once. At any time k, they consider a limited number of best matches between the detections of time k and time $k+1$ using the Hungarian-Murty algorithm [Murty 1968]. Here the cost of a matching is the sum of the total associations made in that matching, which is driven by IOU overlap and appearance similarities. They formulate a cost between the temporally adjacent matches to construct the cost of a total trajectory. The optimal cost is then obtained by solving it through dynamic programming. In [Xiang, Alahi, and Savarese 2015], the track's state is formulated as a policy that is decided by a Markov decision process. It learns the association cost while learning the policy. In [Jiang et al. 2019; Rosello and Kochenderfer 2018], the multi-object tracking problem is formulated as a multi-agent reinforcement learning problem, where data association between two frames is equivalent to joint policies taken by all the agents. Deep reinforcement learning models such as Trust Region Policy Optimization (TRPO) and Q-learning are trained on the ground-truth data to learn the joint actions. In [L. Ren et al. 2018], a prediction and a decision network are trained to predict the detections in the next frame using the current tracks and decide a policy for the tracks and detections in a multi-agent environment according to the valid actions

present, respectively. In contrast to these approaches, we introduce approximations to the Bellman equations based on the results of online simulation of heuristics.

1.3 Road Map

In Chapter 2, we present preliminary materials, including a formal description of MOT as a special case of the multidimensional assignment problem, as well as an overview of approximation in value space. In Chapter 3, we present the ADPTrack algorithm and its technical specifications. In Chapter 4, we present an experimental study that applies ADPTrack to the MOT17 dataset. Lastly, in Chapter 5, we discuss the implications of our findings as well as avenues for future research.

Chapter 2

PRELIMINARIES AND MATHEMATICAL MODEL

In this chapter, we present a formal overview of the multi-dimensional assignment problem. We describe how data association is a special case of the multi-dimensional assignment problem. Next, we introduce a sequential decision-making framework based on dynamic programming for solving the multi-dimensional assignment problem, and then lastly describe techniques for introducing approximations.

2.1 Multi-object Tracking as Multi-dimensional Assignment

Multi-object tracking can be formulated as a multi-dimensional assignment problem [Emami et al. 2020]. We present a visualization of a 6-dimensional instance in Figure 2. An instance of the N-dimensional assignment problem is represented by an $(N+1)$-partite graph arranged in layers $\mathcal{N}_0, \mathcal{N}_1, \ldots, \mathcal{N}_N$, each of which contains exactly m nodes. The arcs of the graph take the form (i, j), where i is a node in layer $\mathcal{N}_k$ and j is a node in layer $\mathcal{N}_{k+1}$, for $k = 0, 1, \ldots, N-1$. We refer to a subset of $N+1$ nodes where each of these nodes is from a distinct layer as a *grouping*. A grouping has an associated cost. A feasible solution to an instance of the N-dimensional assignment problem is a set of m node-disjoint groupings. A feasible solution is optimal whenever the sum of all grouping costs is minimum. The assignment problem is a special case where $N = 1$, and there exist many optimal polynomial time algorithms that solve it. On the other hand, the multi-dimensional assignment problem is NP-hard even

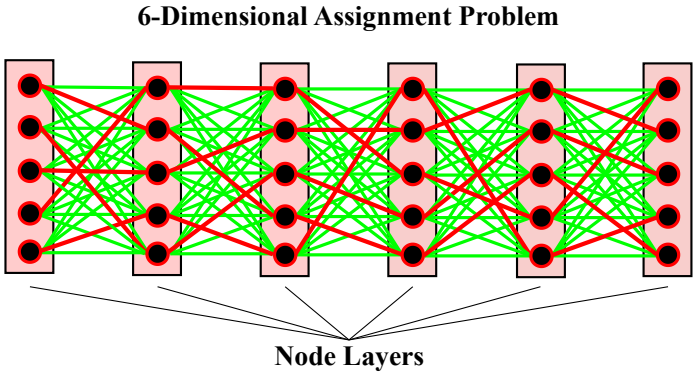

Figure 2. Depiction of a 6-dimensional instance of the multi-dimensional assignment problem where $m = 5$. Green arcs represent the underlying arcs of the instance; red arcs define groupings. The solution is m node-disjoint paths of length 6 and where each path contains exactly one node from each layer (i.e. grouping).

when $N = 2$ [Nguyen, Le Thi, and Pham Dinh 2014], and hence approximations are imperative.

With respect to multi-object tracking, each of the $N+1$ layers represents a frame of m objects. A grouping represents an association of a single object over multiple frames and can be viewed as a track. The cost of a grouping should be small if the object associated over each frame is consistent according to some ground truth. An optimal association will map each distinct object (according to the ground truth) bijectively to the groupings set. Note that for real-world data association problems, the number of objects in a frame need not be fixed, as objects may appear and disappear. Our theoretical model can be extended to account for these, but to retain the simplicity of our model, we make the simplifying assumption that there are m objects in each frame. On the other hand, the implementation of our solution (see Chapter 4) handles such cases.

2.2 Approximation in Value Space and Data Association

We consider a dynamic programming approach since such frameworks admit straightforward approximations. We first describe a dynamic programming formulation for the multi-dimensional assignment problem, and then describe a technique called *approximation in value space*, which forms the foundation of our proposed algorithm in Chapter 3. For a dynamic programming framework, we require four objects: a state space Ω; a state-dependent control space $U(x)$; a system equation $x' = f(x, u)$ where $x, x' \in \Omega$ and $u \in U(x)$; and a cost function $G(\cdot)$. We frame the problem as a sequential decision problem where at each layer $\mathcal{N}_k$, we choose a perfect matching of cardinality m to the nodes of layer $\mathcal{N}_{k+1}$. The control space $U_k(x_k)$ at state x_k is the set of all perfect matchings between layer $\mathcal{N}_k$ and $\mathcal{N}_{k+1}$. The state space is the set of all partial sequences of perfect matchings between consecutive layers up to layer k, for $k = 0, 1, \ldots, N-1$. As such, a state $x_k \in \Omega$ consists of a particular set of perfect matchings between layers $\mathcal{N}_\ell$ and $\mathcal{N}_{\ell+1}$, for all $\ell = 0, 1, \ldots, k-1$, and can be expressed as a sequence of controls $x_k = (u_0, \ldots, u_{k-1})$. The system equation $x_{k+1} = f(x_k, u_k) = (x_k, u_k)$ extends the sequence of perfect matchings by one, and the cost function $G(x_N) = G(u_0, \ldots, u_{N-1})$ applies to a complete grouping set (i.e. m groups of $N+1$ nodes). The exact dynamic programming algorithm to determine the cost $J_k^*(x_k)$ of an optimal sequence of perfect matchings beginning at layer $\mathcal{N}_k$ is given by the Bellman equations and takes the form:

$$J_N^*(x_N) = G(x_N) = G(u_0, \ldots, u_{N-1}),$$

and

$$J_k^*(x_k) = \min_{u_k \in U_k(x_k)} J_{k+1}^*(x_k, u_k), \qquad \text{for all } x_k, k = 0, \ldots, N-1 \tag{2.1}$$

Three major issues arise when attempting to compute $J_k^*(\cdot)$. The first applies specifically to the problems of data association and multi-object tracking, namely that evaluating the quality of a matching sequence is difficult and cost function $G(\cdot)$ is not known. Secondly, the size of the control space per state corresponds to the number of perfect matchings on bipartite graph, of which there may be exponentially many. Lastly, even if $G(\cdot)$ were known and the control space polynomial in size, computing $J_{k+1}^*(\cdot)$ is still intractable. To address these challenges, we resort to a broad technique called *approximation in value space*, which introduces various approximations to the components of the Bellman equations. One such approximation replaces $J_{k+1}^*(\cdot)$ with an approximation given by $\tilde{J}_{k+1}(\cdot)$ that can be computed efficiently. To address the remaining two challenges, $\tilde{J}_{k+1}(\cdot)$ is granted a special structure suitable for the use of fast 2-dimensional assignment algorithms. Namely,

$$\tilde{J}_{k+1}(x_k, u_k) = \sum_{(i,j)\in u_k} c_{k+1}^{ij}(x_k) \tag{2.2}$$

where each $(i, j) \in u_k$ is an arc from the valid perfect matching specified by control u_k, and $c_{k+1}^{ij}(x_k)$ represents the cost for arc (i, j) where $i \in \mathcal{N}_k$ and $j \in \mathcal{N}_{k+1}$. In the case of multi-object tracking, these costs are generated from heuristics, sensor data, and assignments in previous layers. To handle the large control spaces, costs for all possible pair-wise arcs between layers $\mathcal{N}_k$ and $\mathcal{N}_{k+1}$ are generated and an algorithm for weighted matching online is applied to extract a control. The cost of an approximately optimal sequence of matchings beginning at layer $\mathcal{N}_k$ is then given by

$$\min_{u_k \in U_k(x_k)} \tilde{J}_{k+1}(x_k, u_k), \qquad \text{for all } x_k, \tag{2.3}$$

which can be computed efficiently as the problem is reduced to a sequence of N 2-dimensional weighted bipartite matching problems.

Chapter 3

APPROXIMATE DYNAMIC PROGRAMMING TRACK

In this chapter, we describe our proposed algorithm called *Approximate Dynamic Programming Track* (ADPTrack). Our proposed solution aims to compute 2.3, and our methodology generates the costs from Equation 2.2 using online simulation of a base heuristic over future frames as well as past information through appearance feature-based methodology prescribed by the base heuristic. To further improve online efficiency, we impose a *truncated horizon*, meaning online simulations are limited to ℓ frames into the future. We present an illustration of approximation in value space as applied to the multi-dimensional assignment problem in Figure 3.

ADPTrack can be categorized as a near-online tracking method where we first simulate the base heuristic for a limited number of subsequent frames from $k+1$ to

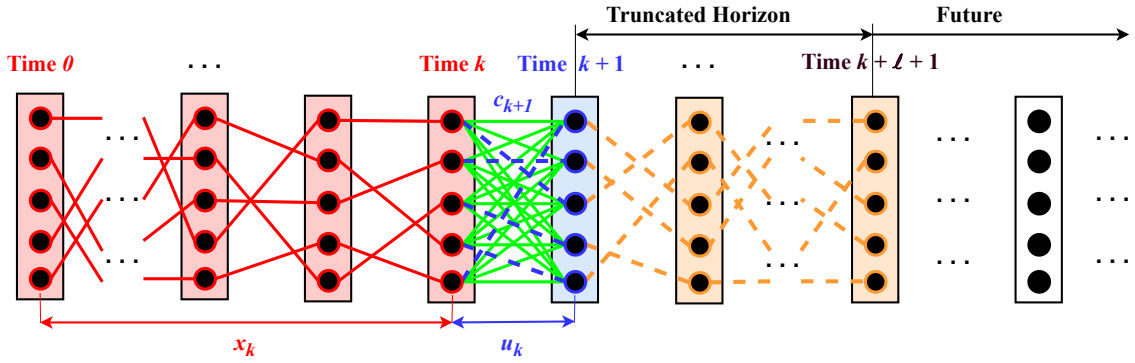

Figure 3. Visualization of approximation in value space applied to the multi-dimensional assignment problem. The red layers represent those included in partial groupings up to time k. The cost function approximation $\tilde{J}_{k+1}(\cdot)$ is dependent on a cost matrix c_{k+1}, which is computed using information obtained from a truncated horizon of length ℓ (blue and orange layers). We select a minimum weight perfect bipartite matching u_k at time k applied to c_{k+1}.

$k+\ell+1$. Next, we augment the cost matrix between existing tracks at k and detections at $k+1$ using the tracklets generated via simulation. Similar to the general near-online methods, we perform the association between frames k and $k+1$ by considering a temporal window from frame k to frame $k+\ell+1$ by following a sliding-window concept, where the window shifts one time-step forward after each association. ADPTrack comprises two components: Near-online simulation, and cost-matrix augmentation & matching.

3.1 Near-online Simulation

Given a base heuristic that is an online tracking algorithm based on a motion model (and may also exploit an appearance feature model), we simulate ℓ frames starting from frame $k+1$ and generate *tracklets* (i.e. tracks of limited length) until frame $k+\ell+1$. In this context, simulation implies using the base heuristic to sequentially match consecutive frames from time $k+1$ to $k+\ell+1$. An illustration is presented in Fig 4. Conceptually, we assume that objects that become occluded have similar appearance features before and after the occlusion, and also similar motion cues. This motivates the method of near-online simulation to produce future tracklets, which can then be compared with the past when determining associations. For this simulation, we choose base heuristics grounded in motion models (or at least that incorporate localization information by using position and size properties between detections of adjacent frames), as the appearance features of an object are not informative when the object is mostly occluded, and hence a motion model can provide better estimations in such scenarios. Note that the appearance features become valuable again when the object becomes partially visible again. How the appearance features of the generated

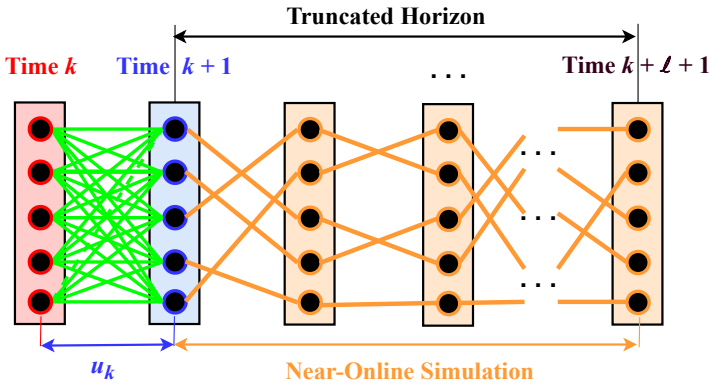

Figure 4. Visualization of near-online simulation in ADPTrack. A tracking simulation is performed for a truncated horizon (ℓ frames) using a base heuristic with a motion model. The base heuristic is initialized at time $k+1$ for the simulation.

tracklets are used for matching the current frames k and $k+1$ is explained in the next section.

At the beginning of the near-online simulation, a motion model adopted by the base heuristic is initialized at frame $k+1$ for each tracklet starting at frame $k+1$, and considers no information on the past frames until k. The motion model is then updated as a result of associations of detections in frames $k+1$ to $k+\ell+1$. For object j, a tracklet generated by near-online simulation starting at time $k+1$ is represented by t^j_{k+1}. During the execution of the simulation, we follow the same motion model policies, appearance features-based score calculations (if present in the base heuristic), track-management policies, and any other additional components involved that are prescribed by the base heuristic.

3.2 Cost-matrix Augmentation & Matching

In this section, we discuss how the tracklets generated by near-online simulation are used for matching a frame k to $k+1$. We describe how to generate a cost matrix c_{k+1} (corresponding to Equation 2.2), which is expressed as a convex combination of two matrices c'_{k+1} and c''_{k+1}. At a high level, c'_{k+1} is a matrix whose entries are based on comparisons (both appearance feature and motion-based) between the tracks until frame k and new bounding boxes are generated at frame $k+1$. On the other hand, c''_{k+1} contains costs that represent relationships between past and future information (i.e. frames beyond $k+1$) with respect to appearance features **only**. As such, c_{k+1} is a cost matrix that temporally aggregates appearance features in both the past and the future, and also accounts for motion cues in the present. More weight applied to c'_{k+1} implies that the information extracted from the current frame $k+1$ is more relevant to the costs, whereas more weight to c''_{k+1} implies that future information is more relevant to costs. Figure 5 illustrates the computation of a cost value for a single arc corresponding to an existing track at frame k and a new detection at frame $k+1$.

We generate the cost matrix c'_{k+1} using the base heuristic, which is described by four major components. There is a set of motion models M, an object localization algorithm Y, an appearance model $A(\cdot)$, and a localization scoring function $L(\cdot)$. Inductively, at time k, there exist a set of m tracks. We let $v^{(i,k')}$ denote a vector that represents an aggregation of appearance feature information until frame k' for track i. Moreover, each track i is associated with a motion model M_i. For each object j at time $k+1$, the base heuristic will use Y to generate a bounding box B^j_{k+1}. Then, the motion model for each track i will estimate the next state of i via a new bounding box B^i_{k+1} on frame $k+1$. The base heuristic then (if applicable) generates

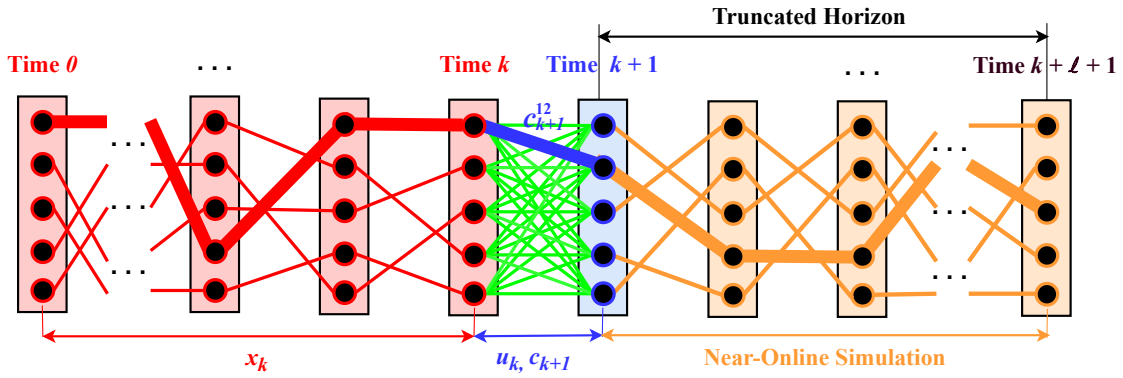

Figure 5. Visualization of cost matrix approximation in ADPTrack. Consider the arc between the first track at frame k and the second detection at frame $k+1$. To calculate the cost value c^{12}_{k+1}, unlike the base heuristic (online tracker) that compares the track at frame k to the detection at frame $k+1$, we compare the track at frame k to the tracklet starting at frame $k+1$ generated by the near-online simulation. The comparison is performed by pairwise appearance features similarity computations (Equation 3.2).

an appearance score $A(v^{(i,k)}, \bar{B}^j_{k+1})$, where $\bar{B}^j_{k+1}$ denotes a representation of the image defined by bounding box B^j_{k+1}. We then compute a localization score $L(B^i_{k+1}, B^j_{k+1})$, which yields a spatial similarity measure. The matrix c'_{k+1} is then defined as

$$c'^{(ij)}_k = f(A(v^{(i,k)}, \bar{B}^j_{k+1}), L(B^i_{k+1}, B^j_{k+1})), \tag{3.1}$$

where $f(\cdot)$ is a function defined by the base heuristic, and $c'^{(ij)}_{k+1}$ refers to the element of the i^{th} row and j^{th} column of c'_{k+1}.

We now explain how to generate matrix c''_{k+1}, which represents a cost matrix that aggregates appearance score information over time. At a high level, we generate c''_{k+1} by comparing each existing track at k with the generated future tracklets, specifically those with their first detection as the detections from time $k+1$. We propose exploiting the previous frame features of the existing tracks by obtaining the appearance similarity between the past and the future frame features, which is intuitively and experimentally (confirmed in Appendix B) a better measure to match an object that has re-appeared.

For an object j at frame $k+1$, we use the base heuristic to generate a tracklet t_{k+1}^{j} that extends to time $k+\ell+1$. Note that these new tracklets are obtained from the simulation of the base heuristic using a reinitialized motion model at frame $k+1$. At a high level, we will perform a pairwise comparison of tracklets generated by the base heuristic with existing tracks. Suppose an object j is mostly occluded over some time window. Then in this time window, the appearance features of j are less relevant since j is not visible. Let $v'^{(i)}$ be a k-dimensional vector, where each element $v_{k'}^{\prime(i)}$ is a real value indicating how informative the detection at frame k' is to matching future detections for track i. We generate $v'^{(i)}$ by appending appearance score $A(v^{(i,k)}, \bar{B}_{k+1}^{j})$ to $v'^{(i)}$ after each frame association to the j^{th} object. We refer to $v'^{(i)}$ as the *appearance quality vector* for track i. For each track/tracklet pair i, t_{k+1}^{j}, we begin at frame k and search backward through i until we find a time k' where $v_{k'}^{\prime(i)}$ is below some threshold parameter τ. This represents a time in the past when the corresponding object is visible. We then consider a limited window W_i of s frames in track i beginning at time $k'-s+1$ and ending at k'. We then compare each of the detections in t_{k+1}^{j} with each of the detections in W_i. We compare a detection n of t_{k+1}^{j} and a detection n' of W_i by independently calculating the appearance feature similarity score using appearance model A on n and n'. Lastly, we obtain $c_{k+1}^{\prime\prime(ij)}$ by taking an average across all pair-wise comparisons, i.e.

$$c_{k+1}^{\prime\prime(ij)} = \frac{1}{s \cdot \ell} \sum_{n' \in W_i} \sum_{n \in t_{k+1}^{j}} A(n, n') \tag{3.2}$$

In the final phase of ADPTrack, we compute the cost matrix c_{k+1} as

$$c_{k+1} = \alpha c_{k+1}'' + (1-\alpha) c_{k+1}', \tag{3.3}$$

where $0 \leq \alpha \leq 1$ is a user-selected weight parameter. For performing association between frames $k+1$ and k over the augmented cost matrix c_{k+1}, we apply the

weighted bipartite matching scheme as prescribed by the base policy. Lastly, if an object j is matched to a track i, we then append $A(v^{(i,k)}, \bar{B}^j_{k+1})$ to $v'^{(i)}$ and continue to associate frames $k+1$ and $k+2$. A pseudocode description of ADPTrack is presented in Algorithm 1.

Algorithm 1 ADPTrack with Base Heuristic $H = (M, Y, A, L, f)$

Parameters: $\ell, s \in \mathbb{Z}^+$; $\tau \in \mathbb{R}^+$; $\alpha \in [0, 1]$.
Input: A set of $N+1$ frames; a set of m tracks of length k; a set of (k)-dimensional appearance quality vectors $\{v'^{(i)} \mid i \in \{1, \ldots, m\}\}$.
Output: A set of m tracks of length $k+1$; a set of $k+1$-dimensional appearance quality vectors $\{v'^{(i)} \mid i \in \{1, \ldots, m\}\}$.

1: Initialize empty matrix c'_{k+1}.
2: **for** each track i **do**
3: **for** each object j on frame $k+1$ **do**
4: Generate bounding box B^j_{k+1} with Y.
5: Generate bounding box B^i_{k+1} with M_i.
6: Set $c'^{(ij)}_{k+1} \leftarrow f(A(v^{(i,k)}, \bar{B}^j_{k+1}), L(B^i_k, B^j_{k+1}))$.
7: Perform data association over frames $k+1$ to $k+\ell+1$ using H.
8: Use output from Line 7 to obtain tracklet t^j_{k+1} for each object j at frame $k+1$.
9: Initialize empty matrix c''_{k+1}.
10: **for** each track i **do**
11: Set $k' \leftarrow \infty$.
12: **for** $k'' = k, k-1, \ldots, 1$ **do**
13: **if** $v'^{(i)}_{k''} < \tau$ **then**
14: Set $k' \leftarrow k''$
15: **break**
16: **if** $k' = \infty$ **then**
17: Set $k' \leftarrow k$
18: Set W_i as set of frames $\max\{1, k'-s+1\}$ to k'.
19: **for** each object j on frame $k+1$ **do**
20: Set $c''^{(ij)}_{k+1} \leftarrow \frac{1}{s \cdot \ell} \sum_{n' \in W_i} \sum_{n \in t^j_{k+1}} A(n, n')$.
21: Set $c_{k+1} \leftarrow \alpha c''_{k+1} + (1-\alpha) c'_{k+1}$.
22: Compute minimum weight bipartite matching u_k on c_{k+1}.
23: Append u_k to track set.
24: **for** each $(i, j) \in u_k$ **do**
25: Append $A(v^{(i,k)}, \bar{B}^j_{k+1})$ to end of $v'^{(i)}$.

Chapter 4

EXPERIMENTAL STUDY

This chapter has a detailed description of the experiment performed and the related implementation details. Our implementation of ADPTrack uses BoT-SORT-Reid [Aharon, Orfaig, and Bobrovsky 2022] as a base heuristic, and hence our primary baseline algorithm is BoT-SORT-Reid.

4.1 Dataset

We used the MOT17 [Milan et al. 2016] dataset under the private-detection protocol for evaluating our algorithms. We perform our experiments on the second half of the training dataset of the MOT17 dataset referred to as the validation dataset. This is because the first half of the training dataset has been used to train the object detector [Yifu Zhang et al. 2022] and re-identification network [Aharon, Orfaig, and Bobrovsky 2022], which are used in the base-heuristic we employed (BoT-SORT). We perform our benchmark evaluation on the testing set of the MOT17 dataset. The videos in the MOT17 dataset consist of many instances where people are temporarily occluded and reappear afterward, and therefore we consider it a good test for our proposed algorithm.

4.2 Metrics

We adopt *clear metrics* [Bernardin and Stiefelhagen 2008] to evaluate our proposed algorithm with respect to the baseline and other SOTA methods. More specifically, we describe our proposed algorithm results with respect to the following metrics: IDF1 [Ristani et al. 2016], *multi-object tracking accuracy* (MOTA) [Bernardin and Stiefelhagen 2008], *higher-order tracking accuracy* (HOTA) [Luiten et al. 2020], *identity switches* (IDSW), *fragmentations* (Frag), *false positives* (FP), and *false negatives* (FN). The IDF1 score mainly measures association accuracy, and MOTA mainly focuses on detection accuracy, whereas HOTA measures the middle ground of both the previous metrics. IDSW measures the number of incorrect ID switches and Frag measures the number of times a track is missing detections in its trajectory. We use Trackeval [Jonathon Luiten 2020] to generate the scores according to clear metrics. As the focus of our proposed algorithm is handling temporarily occluded objects, we expect a greater improvement in IDF1 and IDSW metrics, while FP, FN, Frag, MOTA, and HOTA may also experience an improvement.

4.3 Implementation Details

We use a Yolox model [Ge et al. 2021] trained by [Yifu Zhang et al. 2022] for object detection and FastReid's SBS-50 model [He et al. 2020] fine-tuned by [Aharon, Orfaig, and Bobrovsky 2022] as an appearance model, as used by the base heuristic. Our algorithm is not specific to any of these modules, which depend on the tracker used as the base heuristic. We adopt the matching thresholds, new track thresholds, minimum thresholds for a track, track buffers, duration to keep lost tracks active, etc.,

according to the base heuristic. We also perform a linear interpolation as proposed in [Yifu Zhang et al. 2022] to track the undetected objects in their occlusion durations. We ran all our experiments on a V100 GPU.

4.4 Base Heuristic

BoT-SORT is an online tracking method that matches the objects frame-by-frame. At time $k+1$, the tracker matches tracks obtained until time k to objects detected at the time $k+1$. This correspondence is performed by minimum cost bipartite matching, where each arc cost represents the cost of matching the corresponding existing track to the corresponding object. Once the matching is performed, the existing tracks are extended by the matched objects. If they are not matched, the tracks continue to exist for a specified period until they are deleted. If an object is not matched at a time $k+1$, it is recognized as a new track.

As mentioned previously, similar to [Yifu Zhang et al. 2022], the tracks are matched to detections in three phases using the Hungarian algorithm. The first phase matches the existing tracks with the high-confidence detections, and the second phase matches the remaining unmatched tracks with the low-confidence detections, whereas the third phase matches the unconfirmed detections from the previous frame with the remaining high-confidence detections. Each track has its own Kalman filter [Kalman 1960] and a smoothed appearance feature. Every time a track is matched to a detection, the Kalman filter and the appearance features are updated by *Kalman gain* and *exponential moving average* [Hunter 1986], respectively.

The cost matrix generation methodology varies over the phases. Firstly, the authors use a Kalman filter to estimate the position and size of the object at time $k+1$. These

predicted box coordinates and sizes are checked against the detected objects to get an area of overlap using IOU overlap. Secondly, BoT-SORT uses a re-identification neural network features to obtain appearance features for the object and compares them to the track's smoothed appearance features according to cosine distance. The arc cost is taken as a minimum of both these scores in phases one and three. In the second phase, the authors use only IOU-based overlap for the matching. The authors also incorporate a *camera compensation module* for handling videos recorded by a moving camera.

4.4.1 ADPTrack with BoT-SORT as Base Heuristic

In this section we explain how we integrated BoT-SORT with ADPTrack. In the Bot-SORT-Reid tracker, according to the proposed algorithm in section 3.2, the motion model M is a Kalman filter; the object localization algorithm Y is Yolox; the appearance model $A(\cdot)$ is the SBS50 model from Fast-Reid (that produces embeddings from an image patch, which are compared with other embeddings through cosine distance); the localization scoring function $L(\cdot)$ is the IOU-overlap algorithm; and function f takes the minimum of the IOU overlap and the appearance features' cosine distance scores. We integrate the Kalman filter, camera motion compensation, cost matrix calculation that fuses appearance similarity score and IOU overlap score, and track management policies similarly to how they are integrated in the BoT-SORT-Reid tracker.

Before comparing the previous frame features of a track with the generated tracklet detections, we check if the tracklet detections are candidates for the track. A tracklet detection becomes a candidate if the following hold: (i) The IOU overlap score of

the detection with the track's estimated next position is sufficiently large, and (ii) the appearance similarity of the object to the original track's smoothed appearance features is sufficiently high. Similarly, we also check if a track is suitable for matching its previous frames to the future frames. We keep track of the appearance similarity scores between the smoothed feature vectors and the detections every time a detection is matched to the track. This allows us to check if the match made is due to high appearance similarity or high IOU overlap, which will indicate whether a track is suitable for association based on previous to future frame comparisons.

We observed that our algorithm may experience difficulty in the presence of objects that are continuously occluded, as their appearance features do not capture a consistent feature of a single person, rather they may be a noisy mix of several intercepting bounding boxes. Therefore we propose a crowd-detection heuristic where we check if the bounding boxes have been occluded for an extended period with respect to their overall life and decide whether to use the proposed algorithm for performing associations for those tracks. In cases such as these, the Kalman filter alone may itself be a better estimator of the association scores.

4.5 Results

In this section, we present a benchmark evaluation that compares BoT-SORT-Reid and ADPTrack with the base heuristic as BoT-SORT-Reid on our chosen dataset. We present the video-wise (i.e., the scores obtained for specific videos) improvement scores of ADPTrack over the BoT-SORT-Reid algorithm for the videos in the MOT17 dataset (both validation and test) in Table 1. We illustrate the IDF1 and IDSW scores of both the trackers for all the videos of the MOT17 dataset in Figures 6 and

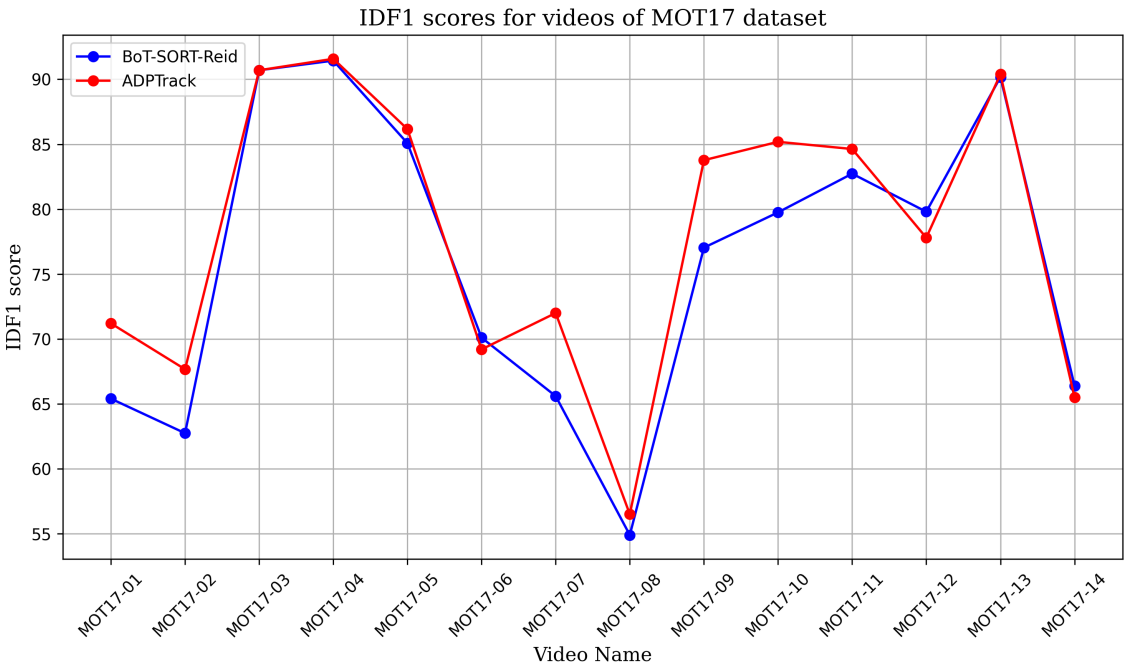

Figure 6. Video-wise IDF1 (↑) scores of ADPTrack and BoT-SORT-Reid trackers when applied to videos of the MOT17 dataset.

7, respectively. The video-wise scores of all other metrics of ADPTrack and the base heuristic over the MOT17 dataset are present in Appendix A. We present the overall scores for both the trackers in Tables 2 and 3, where Table 2 shows the overall scores of both the trackers over the validation dataset, and Table 3 shows the overall scores achieved by both of the trackers over the benchmark dataset. Figures 8 and 9 illustrate the percentage improvement of ADPTrack over the base heuristic BoT-SORT-Reid across all the metrics over the validation and test dataset, respectively. We performed ablation studies over different components of ADPTrack and parameter variation studies for weights, number of future frames, etc. We present these in Appendix B.

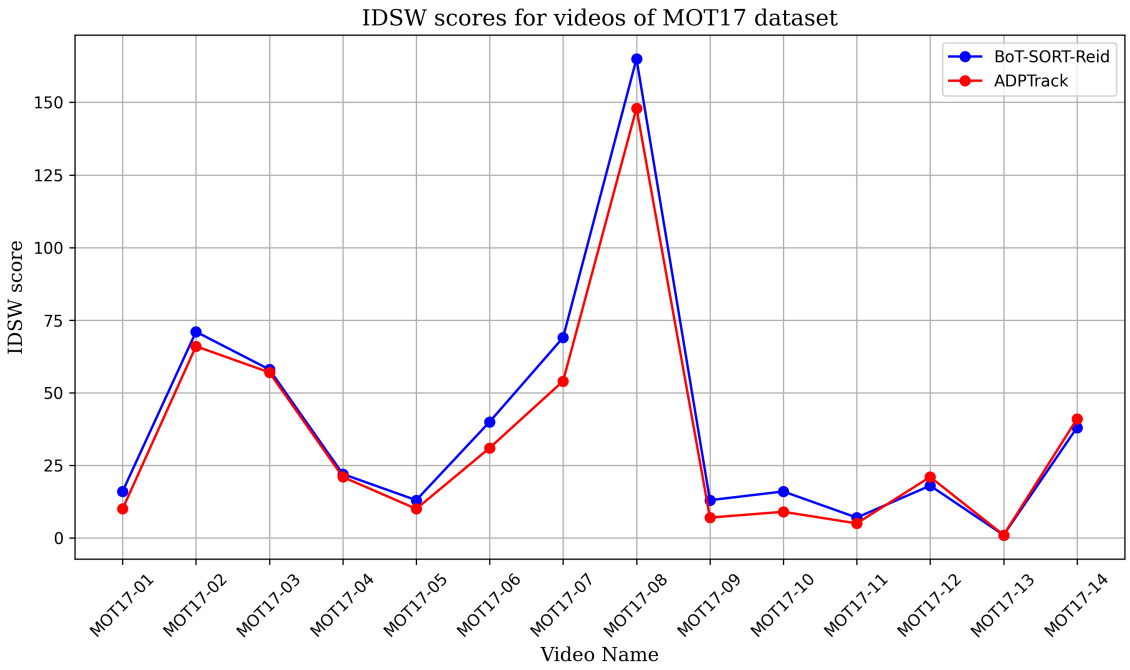

Figure 7. Video-wise IDSW (↓) scores of ADPTrack and BoT-SORT-Reid trackers when applied to videos of the MOT17 dataset.

Over the validation dataset in Table 2, we see an overall improvement of 2.1% in the IDF1 metric, 0.4% in the MOTA metric, 1.1% in the HOTA metric, and a considerable reduction in the FP, FN, IDSW, and Frag metrics. In the benchmark evaluation (on the test dataset) in Table 3, we see an overall improvement of 0.7% in the IDF1 metric, 0.2% in the MOTA metric, 0.4% in the HOTA metrics, and a considerable reduction in FP, FN, IDSW, and Frag metrics. Our algorithm also does better than (or is comparable to) several other SOTA trackers and the comparison results are mentioned in Appendix A (Table 6). Figure 7 shows the consistent considerable improvement in IDSW metric across all of the videos except for two (MOT17-12 and 14) in which the base heuristic performs only slightly better. Figures 8 and 9 demonstrate the huge overall improvement obtained in the IDSW metric on the validation and the test dataset, respectively. With respect to specific video examples in Table 1 and

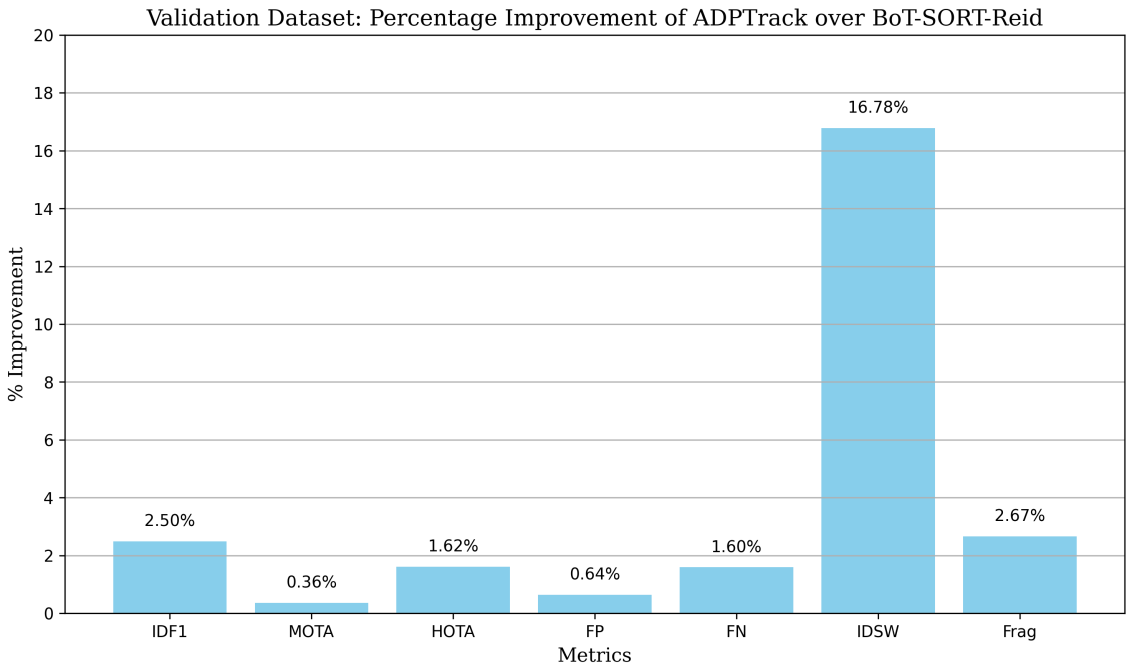

Figure 8. Percentage improvement of ADPTrack over BoT-SORT tracker across several MOT metrics on the validation dataset.

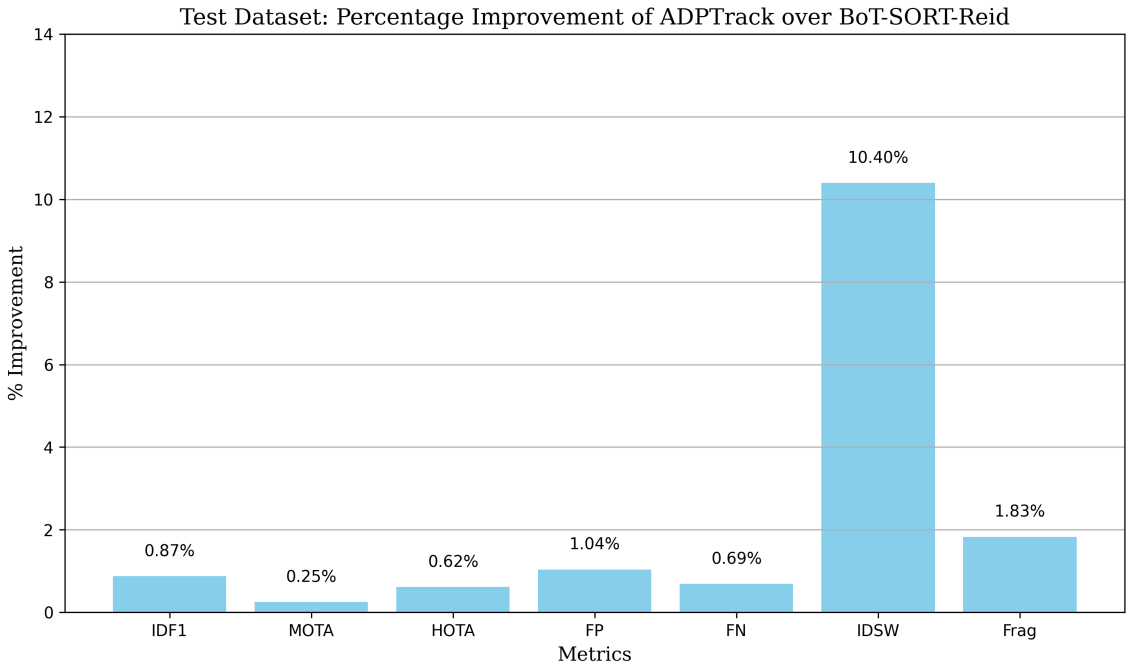

Figure 9. Percentage improvement of ADPTrack over BoT-SORT tracker across several MOT metrics on the test dataset.

Table 1. Video-wise improvement scores of ADPTrack over BoT-SORT when applied to the videos of the MOT17 dataset.

Video Name	$IDF1 \uparrow$	$MOTA \uparrow$	$HOTA \uparrow$	$FP \downarrow$	$FN \downarrow$	$IDSW \downarrow$	$Frag \downarrow$
MOT17-01	**5.8**	**0.399**	**2.9**	**-7**	**-9**	**-6**	**-1**
MOT17-02	**4.923**	**0.677**	**3.191**	**-13**	**-49**	**-5**	**-1**
MOT17-03	0	**0.100**	0	**-41**	**-92**	**-1**	1
MOT17-04	**0.141**	-0.012	-0.156	22	**-18**	**-1**	2
MOT17-05	**1.087**	**1.012**	**1.146**	**-21**	**-10**	**-3**	1
MOT17-06	-0.89	**0.300**	-0.299	43	**-63**	**-9**	4
MOT17-07	**6.400**	**0.80**	**3.8**	**-88**	**-36**	**-15**	**-16**
MOT17-08	**1.6**	0	**0.799**	**-10**	22	**-17**	**-7**
MOT17-09	**6.736**	**1.633**	**4.804**	**-4**	**-37**	**-6**	**-4**
MOT17-10	**5.433**	**-0.253**	**3.538**	17	5	**-7**	**-2**
MOT17-11	**1.899**	**-0.132**	**0.965**	0	8	**-2**	**-1**
MOT17-12	-2	-0.5	-0.70	29	8	3	4
MOT17-13	**0.220**	**1.077**	**0.201**	**-21**	**-13**	0	**-1**
MOT17-14	-0.900	**0.10**	-0.5	**-4**	**-27**	3	4

Table 2. Overall-scores' comparison of the proposed algorithm with the base heuristic over the validation dataset.

Algorithm	$IDF1 \uparrow$	$MOTA \uparrow$	$HOTA \uparrow$	$FP \downarrow$	$FN \downarrow$	$IDSW \downarrow$	$Frag \downarrow$
BoT-SORT	83.277	80.718	70.607	9312	21432	429	675
ADPTrack	**85.355**	**81.011**	**71.749**	**9252**	**21090**	**357**	**657**

Table 3. Overall-scores' comparison of the proposed algorithm with the base heuristic over the test dataset.

Algorithm	$IDF1 \uparrow$	$MOTA \uparrow$	$HOTA \uparrow$	$FP \downarrow$	$FN \downarrow$	$IDSW \downarrow$	$Frag \downarrow$
BoT-SORT	80.2	80.5	65.0	22521	86037	1212	1803
ADPTrack	**80.9**	**80.7**	**65.4**	**22287**	**85446**	**1086**	**1770**

Figure 6, we would like to emphasize the significant improvement of the IDF1 scores in MOT17-01 (5.8%), MOT17-02 (4.923%), MOT17-05 (1.08%), MOT17-07 (6.4%), MOT17-08 (1.6%), MOT17-09 (6.736%), MOT17-10 (5.433%), and MOT17-11 (1.9%) videos, which contain many instances of temporary occlusions. In MOT17-03 and MOT17-04 videos, we maintain the accuracy attained by the base heuristic. In MOT17-06, 12, and 14, we see a slight reduction in the IDF1 metric (also reflected in

the slight increase in IDSW for MOT17-12 and 14). This can be attributed to the fast-moving cameras, and as such, the appearance features of the future frames can be quite different from the appearance features of the previous frames, thus preventing the association of the same object even if it is matched according to the output of the Kalman filter. We would like to address this issue in future offerings. Secondly, we are considering only the future tracklets that have their first detections starting at frame $k+1$ while matching frames $k+1$ and k. However, looking at the other tracklets starting later than $k+1$ could be useful in penalizing the incorrect associations in the previous tracks. We would also like to address this in the future.

Chapter 5

CONCLUSION

We have presented an approximate dynamic programming framework for rendering existing MOT solutions more robust to the challenges that arise from object occlusions. As the performance of our solution on the validation and the test datasets is eminently promising, we conclude that the techniques associated with approximate DP, specifically approximation in value space and online simulation, can be effectively adapted for occlusion-prone multi-object tracking instances. Moreover, the representation of the base heuristic in our model is very general, suggesting virtually any online tracker with motion and appearance models can be adapted to improve performance against occlusion. Lastly, our method is timeless (unless the occlusion problem is completely solved) in the sense that one may compose the latest SOTA tracker to generate even better results.

With respect to limitations, since we perform a near-online simulation over l frames repeatedly, our solution is necessarily more computationally expensive than the base heuristic. However, there may be methods to save on this extra computation by reusing calculations from previous frames, which we hope to explore in future work. Moreover, the cost-matrix augmentation component of our method consists of computationally expensive appearance feature similarity scores, however, there is an opportunity for parallelization here since these calculations are independent. There may be other optimizations, such as storing the appearance cost values instead of recomputing in some cases, as a particular future frame may be present in multiple look-ahead calculations. Lastly, an interesting avenue for future research is the idea of

repeated compositions. ADPTrack with a base heuristic can itself be viewed as a base heuristic for ADPTrack. Hence, it may be interesting to see if additional performance gain can be obtained from a nested composition.

REFERENCES

Aharon, Nir, Roy Orfaig, and Ben-Zion Bobrovsky. 2022. *BoT-SORT: Robust Associations Multi-Pedestrian Tracking.* arXiv: 2206.14651 [cs.CV].

Bae, Seung-Hwan, and Kuk-Jin Yoon. 2018. "Confidence-Based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-Object Tracking." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40 (3): 595–610. https://doi.org/10.1109/TPAMI.2017.2691769.

Bergmann, Philipp, Tim Meinhardt, and Laura Leal-Taixe. 2019. "Tracking Without Bells and Whistles." In *2019 IEEE/CVF International Conference on Computer Vision (ICCV).* IEEE, October. https://doi.org/10.1109/iccv.2019.00103.

Bernardin, Keni, and Rainer Stiefelhagen. 2008. "Evaluating multiple object tracking performance: The CLEAR MOT metrics." *EURASIP Journal on Image and Video Processing* 2008 (January). https://doi.org/10.1155/2008/246309.

Bertsekas, Dimitri. 2023. *A course in reinforcement learning.* Athena Scientific.

Bewley, Alex, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. 2016. "Simple online and realtime tracking." In *2016 IEEE International Conference on Image Processing (ICIP),* 3464–3468. https://doi.org/10.1109/ICIP.2016.7533003.

Bochinski, Erik, Volker Eiselein, and Thomas Sikora. 2017. "High-Speed tracking-by-detection without using image information." In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS),* 1–6. https://doi.org/10.1109/AVSS.2017.8078516.

Bochinski, Erik, Tobias Senst, and Thomas Sikora. 2018. "Extending IOU Based Multi-Object Tracking by Visual Information." In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS),* 1–6. https://doi.org/10.1109/AVSS.2018.8639144.

Brasó, Guillem, and Laura Leal-Taixé. 2020. *Learning a Neural Solver for Multiple Object Tracking.* arXiv: 1912.07515 [cs.CV].

Cai, Jiarui, Mingze Xu, Wei Li, Yuanjun Xiong, Wei Xia, Zhuowen Tu, and Stefano Soatto. 2022. *MeMOT: Multi-Object Tracking with Memory.* arXiv: 2203.16761 [cs.CV].

Cao, Jinkun, Jiangmiao Pang, Xinshuo Weng, Rawal Khirodkar, and Kris Kitani. 2023. *Observation-Centric SORT: Rethinking SORT for Robust Multi-Object Tracking.* arXiv: 2203.14360 [cs.CV].

Cetintas, Orcun, Guillem Brasó, and Laura Leal-Taixé. 2023. *Unifying Short and Long-Term Tracking with Graph Hierarchies.* arXiv: 2212.03038 [cs.CV].

Chen, Jiahui, Hao Sheng, Yang Zhang, and Zhang Xiong. 2017. "Enhancing Detection Model for Multiple Hypothesis Tracking." In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW),* 2143–2152. https://doi.org/10.1109/CVPRW.2017.266.

Chen, Long, Haizhou Ai, Zijie Zhuang, and Chong Shang. 2018. "Real-Time Multiple People Tracking with Deeply Learned Candidate Selection and Person Re-Identification." In *2018 IEEE International Conference on Multimedia and Expo (ICME).* IEEE, July. https://doi.org/10.1109/icme.2018.8486597.

Choi, Wongun. 2015. *Near-Online Multi-target Tracking with Aggregated Local Flow Descriptor.* arXiv: 1504.02340 [cs.CV].

Choudhuri, Anwesa, Girish Chowdhary, and Alexander G. Schwing. 2021. "Assignment-Space-based Multi-Object Tracking and Segmentation." In *2021 IEEE/CVF International Conference on Computer Vision (ICCV),* 13578–13587. https://doi.org/10.1109/ICCV48922.2021.01334.

Chu, Peng, and Haibin Ling. 2019. *FAMNet: Joint Learning of Feature, Affinity and Multi-dimensional Assignment for Online Multiple Object Tracking.* arXiv: 1904.04989 [cs.CV].

Chu, Peng, Jiang Wang, Quanzeng You, Haibin Ling, and Zicheng Liu. 2021. *TransMOT: Spatial-Temporal Graph Transformer for Multiple Object Tracking.* arXiv: 2104.00194 [cs.CV].

Chu, Qi, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, and Nenghai Yu. 2017. *Online Multi-Object Tracking Using CNN-based Single Object Tracker with Spatial-Temporal Attention Mechanism.* arXiv: 1708.02843 [cs.CV].

Dehghan, Afshin, Shayan Modiri Assari, and Mubarak Shah. 2015. "GMMCP tracker: Globally optimal Generalized Maximum Multi Clique problem for multiple object tracking." In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 4091–4099. https://doi.org/10.1109/CVPR.2015.7299036.

Du, Yunhao, Junfeng Wan, Yanyun Zhao, Binyu Zhang, Zhihang Tong, and Junhao Dong. 2022. *GIAOTracker: A comprehensive framework for MCMOT with global information and optimizing strategies in VisDrone 2021.* arXiv: 2202.11983 [cs.CV].

Du, Yunhao, Zhicheng Zhao, Yang Song, Yanyun Zhao, Fei Su, Tao Gong, and Hongying Meng. 2023. *StrongSORT: Make DeepSORT Great Again.* arXiv: 2202.13514 [cs.CV].

Emami, Patrick, Panos M. Pardalos, Lily Elefteriadou, and Sanjay Ranka. 2020. "Machine Learning Methods for Data Association in Multi-Object Tracking." *ACM Computing Surveys* 53, no. 4 (August): 1–34. https://doi.org/10.1145/3394659.

Fagot-Bouquet, Loïc, Romaric Audigier, Yoann Dhome, and Frédéric Lerasle. 2016. "Improving Multi-frame Data Association with Sparse Representations for Robust Near-online Multi-object Tracking." In *Computer Vision – ECCV 2016,* edited by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, 774–790. Cham: Springer International Publishing.

Fang, Kuan, Yu Xiang, Xiaocheng Li, and Silvio Savarese. 2018. *Recurrent Autoregressive Networks for Online Multi-Object Tracking.* arXiv: 1711.02741 [cs.CV].

Feng, Weijiang, Long Lan, Yong Luo, Yue Yu, Xiang Zhang, and Zhigang Luo. 2021. "Near-Online Multi-Pedestrian Tracking via Combining Multiple Consistent Appearance Cues." *IEEE Transactions on Circuits and Systems for Video Technology* 31 (4): 1540–1554. https://doi.org/10.1109/TCSVT.2020.3005662.

Feng, Weitao, Baopu Li, and Wanli Ouyang. 2022. "Multi-Object Tracking with Multiple Cues and Switcher-Aware Classification." In *2022 International Conference on Digital Image Computing: Techniques and Applications (DICTA),* 1–10. https://doi.org/10.1109/DICTA56598.2022.10034575.

Ge, Zheng, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. 2021. *YOLOX: Exceeding YOLO Series in 2021.* arXiv: 2107.08430 [cs.CV].

Guo, Song, Jingya Wang, Xinchao Wang, and Dacheng Tao. 2021. *Online Multiple Object Tracking with Cross-Task Synergy.* arXiv: 2104.00380 [cs.CV].

He, Lingxiao, Xingyu Liao, Wu Liu, Xinchen Liu, Peng Cheng, and Tao Mei. 2020. *FastReID: A Pytorch Toolbox for General Instance Re-identification.* arXiv: 2006.02631 [cs.CV].

Henschel, Roberto, Laura Leal-Taixé, Daniel Cremers, and Bodo Rosenhahn. 2018. *Fusion of Head and Full-Body Detectors for Multi-Object Tracking.* arXiv: 1705.08314 [cs.CV].

Henschel, Roberto, Yunzhe Zou, and Bodo Rosenhahn. 2019. "Multiple People Tracking Using Body and Joint Detections." In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW),* 770–779. https://doi.org/10.1109/CVPRW.2019.00105.

Hung, Wei-Chih, Henrik Kretzschmar, Tsung-Yi Lin, Yuning Chai, Ruichi Yu, Ming-Hsuan Yang, and Dragomir Anguelov. 2020. *SoDA: Multi-Object Tracking with Soft Data Association.* arXiv: 2008.07725 [cs.CV].

Hunter, J. Stuart. 1986. "The Exponentially Weighted Moving Average." *Journal of Quality Technology* 18 (4): 203–210. https://doi.org/10.1080/00224065.1986.11979014. eprint: https://doi.org/10.1080/00224065.1986.11979014.

Hyun, Jeongseok, Myunggu Kang, Dongyoon Wee, and Dit-Yan Yeung. 2023. *Detection Recovery in Online Multi-Object Tracking with Sparse Graph Tracker.* arXiv: 2205.00968 [cs.CV].

Jiang, Mingxin, Tao Hai, Zhigeng Pan, Haiyan Wang, Yinjie Jia, and Chao Deng. 2019. "Multi-Agent Deep Reinforcement Learning for Multi-Object Tracker." *IEEE Access* 7:32400–32407. https://doi.org/10.1109/ACCESS.2019.2901300.

Jonathon Luiten, Arne Hoffhues. 2020. *TrackEval.* https://github.com/JonathonLuiten/TrackEval.

Kalman, Rudolph Emil. 1960. "A new approach to linear filtering and prediction problems."

Kim, Chanho, Fuxin Li, Arridhana Ciptadi, and James M. Rehg. 2015. "Multiple Hypothesis Tracking Revisited." In *2015 IEEE International Conference on Computer Vision (ICCV),* 4696–4704. https://doi.org/10.1109/ICCV.2015.533.

Kim, Chanho, Fuxin Li, and James M. Rehg. 2018. "Multi-object Tracking with Neural Gating Using Bilinear LSTM." In *Proceedings of the European Conference on Computer Vision (ECCV).* September.

Leal-Taixé, Laura, Cristian Canton Ferrer, and Konrad Schindler. 2016. *Learning by tracking: Siamese CNN for robust target association.* arXiv: 1604.07866 [cs.LG].

Liang, Chao, Zhipeng Zhang, Xue Zhou, Bing Li, Shuyuan Zhu, and Weiming Hu. 2022. *Rethinking the competition between detection and ReID in Multi-Object Tracking.* arXiv: 2010.12138 [cs.CV].

Luiten, Jonathon, Aljosa Osep, Patrick Dendorfer, Philip Torr, Andreas Geiger, Laura Leal-Taixé, and Bastian Leibe. 2020. "HOTA: A Higher Order Metric for Evaluating Multi-object Tracking." *International Journal of Computer Vision* 129, no. 2 (October): 548–578. https://doi.org/10.1007/s11263-020-01375-2.

Ma, Fan, Mike Zheng Shou, Linchao Zhu, Haoqi Fan, Yilei Xu, Yi Yang, and Zhicheng Yan. 2022. *Unified Transformer Tracker for Object Tracking.* arXiv: 2203.15175 [cs.CV].

Meinhardt, Tim, Alexander Kirillov, Laura Leal-Taixé, and Christoph Feichtenhofer. 2022. *TrackFormer: Multi-Object Tracking with Transformers.* arXiv: 2101.02702 [cs.CV].

Milan, Anton, Laura Leal-Taixé, Ian Reid, Stefan Roth, and Konrad Schindler. 2016. *MOT16: A Benchmark for Multi-Object Tracking.* arXiv: 1603.00831 [cs.CV].

Murty, Katta G. 1968. "Letter to the Editor - An Algorithm for Ranking all the Assignments in Order of Increasing Cost." *Oper. Res.* 16:682–687. https://api.semanticscholar.org/CorpusID:207237751.

Nguyen, Duc Manh, Hoai An Le Thi, and Tao Pham Dinh. 2014. "Solving the Multidimensional Assignment Problem by a Cross-Entropy method." *J. Comb. Optim.* (Berlin, Heidelberg) 27, no. 4 (May): 808–823. https://doi.org/10.1007/s10878-012-9554-z.

Pang, Jiangmiao, Linlu Qiu, Xia Li, Haofeng Chen, Qi Li, Trevor Darrell, and Fisher Yu. 2021. *Quasi-Dense Similarity Learning for Multiple Object Tracking.* arXiv: 2006.06664 [cs.CV].

Qin, Zheng, Sanping Zhou, Le Wang, Jinghai Duan, Gang Hua, and Wei Tang. 2023. *MotionTrack: Learning Robust Short-term and Long-term Motions for Multi-Object Tracking.* arXiv: 2303.10404 [cs.CV].

Rangesh, Akshay, Pranav Maheshwari, Mez Gebre, Siddhesh Mhatre, Vahid Ramezani, and Mohan M. Trivedi. 2021. *TrackMPNN: A Message Passing Graph Neural Architecture for Multi-Object Tracking.* arXiv: 2101.04206 [cs.CV].

Reid, D. 1979. "An algorithm for tracking multiple targets." *IEEE Transactions on Automatic Control* 24 (6): 843–854. https://doi.org/10.1109/TAC.1979.1102177.

Ren, Liangliang, Jiwen Lu, Zifeng Wang, Qi Tian, and Jie Zhou. 2018. "Collaborative Deep Reinforcement Learning for Multi-object Tracking." In *Computer Vision – ECCV 2018,* edited by Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, 605–621. Cham: Springer International Publishing.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun. 2016. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.* arXiv: 1506.01497 [cs.CV].

Ristani, Ergys, Francesco Solera, Roger S. Zou, Rita Cucchiara, and Carlo Tomasi. 2016. *Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking.* arXiv: 1609.01775 [cs.CV].

Rosello, Pol, and Mykel J. Kochenderfer. 2018. "Multi-Agent Reinforcement Learning for Multi-Object Tracking." In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems,* 1397–1404. AAMAS '18. Stockholm, Sweden: International Foundation for Autonomous Agents / Multiagent Systems.

Sadeghian, Amir, Alexandre Alahi, and Silvio Savarese. 2017. *Tracking The Untrackable: Learning To Track Multiple Cues with Long-Term Dependencies.* arXiv: 1701.01909 [cs.CV].

Schulter, Samuel, Paul Vernaza, Wongun Choi, and Manmohan Chandraker. 2017. *Deep Network Flow for Multi-Object Tracking.* arXiv: 1706.08482 [cs.CV].

Shan, Chaobing, Chunbo Wei, Bing Deng, Jianqiang Huang, Xian-Sheng Hua, Xiaoliang Cheng, and Kewei Liang. 2020. *Tracklets Predicting Based Adaptive Graph Tracking.* arXiv: 2010.09015 [cs.CV].

Sheng, Hao, Jiahui Chen, Yang Zhang, Wei Ke, Zhang Xiong, and Jingyi Yu. 2019. "Iterative Multiple Hypothesis Tracking With Tracklet-Level Association." *IEEE Transactions on Circuits and Systems for Video Technology* 29 (12): 3660–3672. https://doi.org/10.1109/TCSVT.2018.2881123.

Sheng, Hao, Yang Zhang, Jiahui Chen, Zhang Xiong, and Jun Zhang. 2019. "Heterogeneous Association Graph Fusion for Target Association in Multiple Object Tracking." *IEEE Transactions on Circuits and Systems for Video Technology* 29 (11): 3269–3280. https://doi.org/10.1109/TCSVT.2018.2882192.

Son, Jeany, Mooyeol Baek, Minsu Cho, and Bohyung Han. 2017. "Multi-object Tracking with Quadruplet Convolutional Neural Networks." In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 3786–3795. https://doi.org/10.1109/CVPR.2017.403.

Sun, Peize, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. 2021. *TransTrack: Multiple Object Tracking with Transformer.* arXiv: 2012.15460 [cs.CV].

Sun, ShiJie, Naveed Akhtar, HuanSheng Song, Ajmal Mian, and Mubarak Shah. 2021. "Deep Affinity Network for Multiple Object Tracking." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43 (1): 104–119. https://doi.org/10.1109/TPAMI.2019.2929520.

Tang, Siyu, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. 2016. *Multi-Person Tracking by Multicut and Deep Matching.* arXiv: 1608.05404 [cs.CV].

Tang, Siyu, Mykhaylo Andriluka, Bjoern Andres, and Bernt Schiele. 2017. "Multiple People Tracking by Lifted Multicut and Person Re-identification." In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* 3701–3710. https://doi.org/10.1109/CVPR.2017.394.

Wang, Bing, Li Wang, Bing Shuai, Zhen Zuo, Ting Liu, Kap Luk Chan, and Gang Wang. 2016. *Joint Learning of Siamese CNNs and Temporally Constrained Metrics for Tracklet Association.* arXiv: 1605.04502 [cs.CV].

Wang, Chien-Yao, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. 2022. *YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.* arXiv: 2207.02696 [cs.CV].

Wang, Qiang, Yun Zheng, Pan Pan, and Yinghui Xu. 2021. *Multiple Object Tracking with Correlation Learning.* arXiv: 2104.03541 [cs.CV].

Wang, Zhongdao, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. 2020. *Towards Real-Time Multi-Object Tracking.* arXiv: 1909.12605 [cs.CV].

Wojke, Nicolai, Alex Bewley, and Dietrich Paulus. 2017. *Simple Online and Realtime Tracking with a Deep Association Metric.* arXiv: 1703.07402 [cs.CV].

Xiang, Yu, Alexandre Alahi, and Silvio Savarese. 2015. "Learning to Track: Online Multi-object Tracking by Decision Making." In *2015 IEEE International Conference on Computer Vision (ICCV),* 4705–4713. https://doi.org/10.1109/ICCV.2015.534.

Xu, Jiarui, Yue Cao, Zheng Zhang, and Han Hu. 2019. *Spatial-Temporal Relation Networks for Multi-Object Tracking.* arXiv: 1904.11489 [cs.CV].

Yang, Fan, Shigeyuki Odashima, Shoichi Masui, and Shan Jiang. 2023. "Hard to track objects with irregular motions and similar appearances? make it easier by buffering the matching space." In *Proceedings of the IEEE/CVF winter conference on applications of computer vision,* 4799–4808.

Yang, Min, Yuwei Wu, and Yunde Jia. 2017. "A Hybrid Data Association Framework for Robust Online Multi-Object Tracking." *IEEE Transactions on Image Processing* 26, no. 12 (December): 5667–5679. https://doi.org/10.1109/tip.2017.2745103.

Yin, Junbo, Wenguan Wang, Qinghao Meng, Ruigang Yang, and Jianbing Shen. 2020. *A Unified Object Motion and Affinity Model for Online Multi-Object Tracking.* arXiv: 2003.11291 [cs.CV].

Yoon, Young-Chul, Du Yong Kim, Young-min Song, Kwangjin Yoon, and Moongu Jeon. 2020. *Online Multiple Pedestrians Tracking using Deep Temporal Appearance Matching Association.* arXiv: 1907.00831 [cs.CV].

Yu, Fengwei, Wenbo Li, Quanquan Li, Yu Liu, Xiaohua Shi, and Junjie Yan. 2016. *POI: Multiple Object Tracking with High Performance Detection and Appearance Feature.* arXiv: 1610.06136 [cs.CV].

Zhang, Yang, Hao Sheng, Yubin Wu, Shuai Wang, Wei Ke, and Zhang Xiong. 2020. "Multiplex Labeling Graph for Near-Online Tracking in Crowded Scenes." *IEEE Internet of Things Journal* 7 (9): 7892–7902. https://doi.org/10.1109/JIOT.2020.2996609.

Zhang, Yifu, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. 2022. "Bytetrack: Multi-object tracking by associating every detection box." In *European conference on computer vision,* 1–21. Springer.

Zhang, Yifu, Chunyu Wang, Xinggang Wang, Wenjun Zeng, and Wenyu Liu. 2021. "FairMOT: On the Fairness of Detection and Re-identification in Multiple Object Tracking." *International Journal of Computer Vision* 129, no. 11 (September): 3069–3087. https://doi.org/10.1007/s11263-021-01513-4.

Zhao, Zelin, Ze Wu, Yueqing Zhuang, Boxun Li, and Jiaya Jia. 2022. *Tracking Objects as Pixel-wise Distributions.* arXiv: 2207.05518 [cs.CV].

Zhou, Xingyi, Tianwei Yin, Vladlen Koltun, and Philipp Krahenbuhl. 2022. *Global Tracking Transformers.* arXiv: 2203.13250 [cs.CV].

Zhu, Ji, Hua Yang, Nian Liu, Minyoung Kim, Wenjun Zhang, and Ming-Hsuan Yang. 2019. *Online Multi-Object Tracking with Dual Matching Attention Networks.* arXiv: 1902.00749 [cs.CV].

APPENDIX A

RESULTS

In this section, we present the video-wise scores of ADPTrack and the base heuristic over the validation and test datasets.

Table 4. Video-wise scores of BoT-SORT-Reid when applied to the MOT17 dataset.

Video Name	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
MOT17-01	65.4	63.4	54.4	461	1883	16	31
MOT17-02	62.743	60.152	52.711	1007	2859	71	92
MOT17-03	90.7	92.6	73.1	3942	3789	58	119
MOT17-04	91.44	91.008	79.886	894	1258	22	31
MOT17-05	85.08	82.306	67.458	221	360	13	14
MOT17-06	70.1	66.6	56.9	959	2933	40	73
MOT17-07	65.6	74.6	53.9	550	3674	69	106
MOT17-08	54.9	65.9	48.5	1059	5977	165	184
MOT17-09	77.036	86.662	65.876	26	345	13	11
MOT17-10	79.753	74.911	59.841	240	1230	16	57
MOT17-11	82.733	71.773	70.984	611	657	7	14
MOT17-12	79.8	72.2	64.2	291	2104	18	34
MOT17-13	90.171	82.858	71.038	105	435	1	6
MOT17-14	66.4	53.5	48.3	245	8319	38	54

Table 5. Video-wise scores of ADPTrack when applied to the MOT17 dataset.

Video Name	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
MOT17-01	71.2	63.8	57.3	454	1874	10	30
MOT17-02	67.666	60.83	55.902	994	2810	66	91
MOT17-03	90.7	92.7	73.1	3901	3697	57	120
MOT17-04	91.581	90.996	79.729	916	1240	21	33
MOT17-05	86.167	83.318	68.604	200	350	10	15
MOT17-06	69.2	66.9	56.6	1002	2870	31	77
MOT17-07	72.0	75.4	57.7	462	3638	54	90
MOT17-08	56.5	65.9	49.3	1049	5999	148	177
MOT17-09	83.772	88.295	70.681	22	308	7	7
MOT17-10	85.186	74.658	63.379	257	1235	9	55
MOT17-11	84.633	71.64	71.949	611	665	5	13
MOT17-12	77.8	71.7	63.5	320	2112	21	38
MOT17-13	90.392	83.935	71.24	84	422	1	5
MOT17-14	65.5	53.6	47.8	241	8292	41	58

Table 6. Overall scores of the proposed, baseline and other state-of-the-art trackers on the MOT17 test dataset.

Method	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
OCSORT	77.5	78.0	63.2	**15129**	107055	1950	2040
Deep-OCSORT	80.6	79.4	64.9	16572	98796	**1023**	2196
StrongSORT++	79.5	79.6	64.4	27876	86205	1194	1866
ByteTrack	77.3	80.3	63.1	25491	83721	2196	2277
MotionTrack	80.1	**81.1**	65.1	23802	81660	1140	**1605**
UTM	78.7	81.8	64.0	25077	**76298**	1431	1889
BoT-SORT-Reid	80.2	80.5	65.0	22521	86037	1212	1803
ADPTrack	**80.9**	80.7	**65.4**	22287	85446	1086	1770

APPENDIX B

ABLATION STUDIES

In this chapter, we perform several ablation studies for all the components we introduced into the proposed algorithm. In the first experiment, we use the base heuristic with only a motion model and do not use an appearance model. In the second experiment, we introduce the appearance model used by the base heuristic, however, we do not use a comparison to the previous frames as in the main experiment. In the third experiment, we consider the standalone base heuristic without our simulation approach. We compare the new frame detection to the previous frames and show how the comparison to the previous frames can be exploited. The last section is a parameter study of the various parameters proposed for our main tracker.

B.1 Experiment 1

The first experiment uses BoT-SORT without an appearance model as the base heuristic where the motion model is a Kalman filter. The near-online simulation is performed using the BoT-SORT tracker, as explained in Section 3.1, albeit, there is no appearance feature model involved here. We generate the cost c''_{k+1} between an existing track and a generated tracklet by following the next steps. We take the motion model of the existing track at time k and update it with the corresponding detection at time $k+1$ to predict the next state for frame $k+2$. We calculate the IOU overlap of the predicted state of the track at frame $k+2$ with the detection at time $k+2$ from the generated tracklet. We update the motion model of the track at frame $k+1$ with the detection at frame $k+2$ and predict the next state of the track at frame $k+3$. We then calculate the IOU overlap with the detection at time $k+3$ and continue so on until the end of the generated tracklet. We take an average of all these IOU overlaps to get the c''_{k+1} value for one detection at time $k+1$ and one track at k. Once the whole cost matrix is calculated by iterating over all the existing tracks at k and all future tracks corresponding to the detections at time $k+1$, we take a weighted average of c''_{k+1} with c'_{k+1} (generated by the base heuristic).

As our base heuristic for this experiment is the BoT-SORT tracker (without the appearance model), we present a second set of baseline results for comparison to our proposed algorithm in this section. Table 7 and 8 show the video-wise results of BoT-SORT and ADPTrack methods for all the videos of the validation dataset. The overall scores for both the trackers are mentioned in Table 9.

We see a significant improvement in the IDF1 metric for MOT17-02 (4.04%), MOT17-04 (0.78%), MOT17-05 (1.948%), and MOT17-11 (4.093%). In fact, out of all the experiments, we see the best score for the MOT17-04 video in this experiment, which is 92.236%. We see a reduction in MOT17-09, MOT17-10, and MOT17-13. This may be a result of the motion cues not being accurate in the presence of a moving camera, or the objects moving closely where there may be fallouts because of solely basing the association upon IOU-overlap (specifically future frames-based

IOU overlap). We believe that this kind of tracker is suitable for videos in which the motion cues can be more helpful, such as MOT17-04, where the appearance features may be noisy, however, the objects mostly can be accurately tracked based on motion cues, as it has an overhead fixed camera to observe the motion of the objects. Across all the videos, we see an overall improvement of 1.35% in the IDF1 metric, 0.5% in the HOTA metric, and better IDSW, FP, FN, and Frag scores at a slightly better MOTA metric. We think that this is quite promising, given that it is only using a motion model, which can be quite inexpensive.

Table 7. Video-wise scores of BoT-SORT tracker (without appearance model) when applied to the validation dataset.

Video Name	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
MOT17-02	60.09	59.615	51.484	1060	2867	63	84
MOT17-04	91.448	90.893	80.002	949	1236	17	30
MOT17-05	81.218	82.038	65.948	223	367	13	16
MOT17-09	83.407	86.419	70.049	23	357	11	10
MOT17-10	77.879	75.063	58.677	239	1222	16	55
MOT17-11	80.53	71.596	69.233	614	660	9	14
MOT17-13	90.689	83.08	71.354	90	441	3	6

Table 8. Video-wise scores of ADPtrack as per experiment 1 over the validation dataset.

Video Name	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
MOT17-02	64.132	59.919	53.049	1118	2784	58	86
MOT17-04	92.236	90.847	80.272	940	1256	17	31
MOT17-05	83.076	82.335	65.993	235	343	15	20
MOT17-09	82.522	86.732	69.962	27	344	11	10
MOT17-10	77.093	75.131	58.459	240	1216	17	56
MOT17-11	84.623	71.618	71.861	612	665	5	13
MOT17-13	90.369	83.112	71.236	89	441	3	6

Table 9. Comparison of overall scores' of BoT-SORT and ADPTrack trackers as per experiment 1 over the validation dataset.

Algorithm	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
BoT-SORT	82.55	80.553	70.346	9594	21450	396	645
Experiment 1	83.908	80.635	70.878	9783	21147	378	666

B.1.1 Parameter selection for experiment 1

The number of frames for near-online simulation and the weight parameter in cost-matrix augmentation are varied in two ablation studies and the experimentation results are presented in Tables 10 and 11, respectively. We observe that the performance of the tracker increases as we reach a certain length of future frames, and then decreases as we go further. This indicates that as we look further ahead into the future and keep updating the track's Kalman filter to compare it to the generated tracklet, it can change too much compared to the original track's Kalman filter at frame k and result in discrepancies.

Table 10. An ablation study over the number of future frames; ADPtrack with BOT-SORT tracker as a base-heuristic as per experiment 1 over the validation dataset; ℓ: number of future frames for near-online simulation.

ℓ	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
1	83.449	80.49	70.567	9864	21285	393	672
2	83.566	80.403	70.677	9879	21417	387	663
3	83.627	80.494	70.775	9906	21246	384	666
4	83.782	80.64	70.771	9786	21132	381	666
5	83.738	80.598	70.742	9786	21204	378	660
6	83.878	80.59	70.851	9783	21219	378	663
7	83.908	80.635	70.878	9783	21147	378	666
8	83.861	80.596	70.797	9738	21249	384	672
9	83.861	80.594	70.796	9741	21249	384	672
10	83.82	80.551	70.824	9768	21291	384	666
11	83.796	80.51	70.825	9765	21363	381	660
12	83.781	80.501	70.82	9741	21396	387	666
13	83.78	80.512	70.827	9741	21378	387	663
14	83.649	80.503	70.754	9750	21381	390	663
15	83.657	80.572	70.748	9759	21261	390	669

Table 11. An ablation study over the weight given to c''_{k+1}; ADPtrack with BOT-SORT tracker as a base-heuristic as per experiment 1 over the validation dataset; ℓ: number of future frames for near-online simulation.

ℓ	Weight	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
6	0.05	83.214	80.549	70.671	9600	21447	399	648
6	0.1	83.694	80.685	70.879	9525	21330	372	648
6	0.15	83.878	80.59	70.851	9783	21219	378	663
6	0.2	83.361	80.215	70.511	10023	21513	450	702
7	0.05	83.214	80.549	70.671	9600	21447	399	648
7	0.1	83.718	80.698	70.893	9525	21312	369	645
7	0.15	83.908	80.635	70.878	9783	21147	378	666
7	0.2	83.599	80.282	70.648	9918	21528	432	699
8	0.05	83.215	80.551	70.67	9597	21447	399	648
8	0.1	83.718	80.698	70.893	9525	21312	369	645
8	0.15	83.861	80.596	70.797	9738	21249	384	672
8	0.2	83.513	80.154	70.562	10080	21546	459	702
9	0.05	83.215	80.551	70.671	9597	21447	399	648
9	0.1	83.726	80.679	70.928	9534	21330	372	648
9	0.15	83.861	80.594	70.796	9741	21249	384	672
9	0.2	83.094	80.109	70.531	10161	21552	444	705
10	0.05	83.215	80.551	70.671	9597	21447	399	648
10	0.1	83.74	80.649	70.933	9531	21378	375	648
10	0.15	83.82	80.551	70.824	9768	21291	384	666
10	0.2	82.433	80.174	70.114	9990	21615	447	714

B.2 Experiment 2

In the second experiment, we use the BoT-SORT-Reid tracker as the base heuristic, however, we do not use any comparison to the previously matched object of the track. Specifically, we compare the detection at frame $k+1$ of the generated tracklet to only the smoothed appearance feature of the track at frame k. We perform the near-online simulation using the BoT-SORT-Reid tracker and generate tracklets. The IOU overlap calculation is also the same as experiment 1. Coming to the appearance features part, each time we calculate the IOU overlap, we also calculate an appearance score by comparing the smoothed feature of the original track to the detection's appearance features, and fuse it with the corresponding IOU overlap. We do not update the appearance feature of the original track while comparing the generated track and the track at frame k, as we do with the motion model. We continue this process until the end of the generated tracklet and take an average of all scores calculated for the generated tracklet to get a single value of c''_{k+1} that indicates association cost between

the existing track until frame k and detection at time $k+1$. Similar to experiment 1, we calculate a weighted average of c''_{k+1} and c'_{k+1}.

The video-wise results for BoT-SORT-Reid and ADPtrack-BoT-SORT-Reid are presented in Tables 4 and 12, respectively. The overall results are mentioned in Table 17. We see a significant improvement in the IDF1 scores of the following videos: MOT17-09 (7.313%), MOT17-10 (4.76%), MOT17-02 (1.1%), MOT17-11(1.89%). We see a small improvement for some videos (MOT17-04 and MOT17-13) and a drop in the IDF1 metric for the other videos (MOT17-05). Overall, we see an improvement of 1.16% in the IDF1 metric, 0.4% in the MOTA metrics, and 0.8% in the HOTA metrics accompanied by a reduction in IDSW, FP, and FN metrics.

Table 12. Video-wise scores of ADPtrack with BOT-SORT-Reid tracker as a base-heuristic as per experiment 2 over the validation dataset (without leveraging previous frames).

Video Name	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
MOT17-02	63.844	61.326	54.267	961	2793	67	95
MOT17-04	91.887	91.033	79.969	908	1240	20	33
MOT17-05	80.743	82.246	65.415	214	364	18	21
MOT17-09	84.349	88.017	70.839	22	317	6	5
MOT17-10	84.515	74.911	63.025	259	1217	10	55
MOT17-11	84.623	71.618	71.861	612	665	5	13
MOT17-13	90.193	83.587	71.156	85	432	1	5

B.2.1 Parameter selection for experiment 2

The number of frames for near-online simulation and the weight parameter in cost-matrix augmentation are varied in two ablation studies and the experimentation results are presented in Tables 13 and 14. We observe that the performance of the tracker increases as we increase both the number of frames and then plateaus as we go further.

Table 13. An ablation study over the number of future frames; ADPtrack with BOT-SORT-Reid tracker as a base-heuristic as per experiment 2 over the validation dataset (without leveraging previous frames); ℓ: number of future frames for near-online simulation.

ℓ	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
1	84.004	81.047	71.074	9204	21066	372	663
2	83.925	80.982	71.027	9276	21087	384	675
3	83.925	80.982	71.027	9276	21087	384	675
4	83.925	80.982	71.027	9276	21087	384	675
5	83.929	80.995	71.03	9258	21084	384	675
6	83.945	81.035	71.08	9180	21093	387	681
7	84.309	81.024	71.303	9195	21096	387	684
8	84.309	81.024	71.303	9195	21096	387	684
9	84.309	81.024	71.303	9195	21096	387	684
10	84.309	81.024	71.303	9195	21096	387	684
11	84.444	81.043	71.403	9183	21084	381	681
12	84.444	81.043	71.403	9183	21084	381	681
13	84.444	81.043	71.403	9183	21084	381	681
14	84.444	81.043	71.403	9183	21084	381	681
15	84.413	81.021	71.385	9183	21120	381	684

B.3 Experiment 3

This ablation study is to verify if exploiting previous frames to match the current frames would result in any better association. We introduce a heuristic to compare selective previous frames of a track at frame k and the current frame feature $k+1$. The costs generated by this heuristic and the base heuristic are combined using a weighted average. As discussed in section 3.2, to compare a detection at time $k+1$ to an existing track at k, the heuristic compares some of the existing track's previously matched frames' detections with the features of the detection at time $k+1$ using a cosine distance (as per BoT-SORT-Reid) and computes an average of all the cosine distance comparison scores between the detection at frame $k+1$ and the detections of the existing track at time k. The number of previous frames that can be used for comparison is flexible and the experimental studies varying the number of previous frames are shown in Table 16. As mentioned in Algorithm 1, we store the appearance feature similarity scores when a new detection is matched to a track, and use them to check if that particular detection is a good resource to represent the track for comparing it to a new object. The video-wise scores for the best result are shown in Table 15. The overall results are mentioned in Table 17. The video-wise scores and the overall scores suggest that indeed using the previous frames is promising,

Table 14. An ablation study over the weight given to c''_{k+1}; ADPtrack with BOT-SORT-Reid tracker as a base-heuristic as per experiment 2 over the validation dataset (without leveraging previous frames); ℓ: number of future frames for near-online simulation.

ℓ	Weight	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
9	0.05	84.214	80.92	71.145	9285	21168	393	669
9	0.1	84.309	81.024	71.303	9195	21096	387	684
9	0.15	83.837	81.05	71.034	9231	21015	390	687
9	0.2	83.501	81.024	70.891	9120	21138	420	702
10	0.05	84.214	80.92	71.144	9285	21168	393	669
10	0.1	84.309	81.024	71.303	9195	21096	387	684
10	0.15	83.837	81.05	71.034	9231	21015	390	687
10	0.2	83.533	81.06	70.942	9072	21132	417	702
11	0.05	84.214	80.92	71.144	9285	21168	393	669
11	0.1	84.444	81.043	71.403	9183	21084	381	681
11	0.15	83.839	81.054	71.036	9228	21012	390	687
11	0.2	83.533	81.06	70.942	9072	21132	417	702
12	0.05	84.214	80.92	71.144	9285	21168	393	669
12	0.1	84.444	81.043	71.403	9183	21084	381	681
12	0.15	83.837	81.041	71.037	9219	21042	390	687
12	0.2	83.533	81.06	70.942	9072	21132	417	702
13	0.05	84.214	80.92	71.144	9285	21168	393	669
13	0.1	84.444	81.043	71.403	9183	21084	381	681
13	0.15	83.837	81.041	71.037	9219	21042	390	687
13	0.2	83.533	81.06	70.942	9072	21132	417	702

especially if they're compared to the future frames. The combination of experiments 1,2 and 3 is presented in Algorithm 1 and referred to as the main experiment in the next sections and Table 17. The results indeed indicate that the main experiment that compares the previous frames of the existing track with the future frames obtained by simulation produces the best outcome.

Table 15. Video-wise scores of standalone BoT-SORT-Reid equipped with pairwise previous frame similarity score comparisons as per experiment 3.

Video Name	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
MOT17-02	64.803	60.992	54.481	995	2793	66	90
MOT17-04	91.586	91.054	79.746	908	1235	20	32
MOT17-05	85.015	82.306	67.338	224	358	12	14
MOT17-09	81.787	87.669	68.861	26	321	8	6
MOT17-10	84.241	74.726	62.828	258	1228	11	55
MOT17-11	84.62	71.618	71.872	611	666	5	13
MOT17-13	90.528	83.587	71.289	85	432	1	5

Table 16. An ablation study over the number of previous frames to be considered; Standalone BoT-SORT-Reid equipped with pairwise previous frame similarity score comparisons as per experiment 3. $\#s$: Number of previous frames to be considered for comparison to a new detection.

$\#s$	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
1	84.59	80.956	71.321	9321	21099	369	645
2	84.567	80.95	71.312	9324	21102	372	651
3	84.43	80.87	71.213	9402	21144	381	660
4	84.43	80.87	71.213	9402	21144	381	660
5	84.43	80.87	71.213	9402	21144	381	660
6	84.43	80.87	71.213	9402	21144	381	660
7	84.43	80.87	71.213	9402	21144	381	660
8	84.43	80.87	71.213	9402	21144	381	660
9	84.436	80.891	71.219	9378	21144	372	660
10	84.436	80.891	71.219	9378	21144	372	660
11	84.436	80.891	71.219	9378	21144	372	660
12	84.436	80.891	71.219	9378	21144	372	660
13	84.436	80.891	71.219	9378	21144	372	660
14	84.436	80.891	71.219	9378	21144	372	660
15	84.436	80.891	71.219	9378	21144	372	660

B.4 Parameter study for the ADPtrack

In this section, we present ablation studies for various parameters present in the proposed algorithm. Table 18 presents an experimental study performed over the number of future frames considered. The ablation study over the weight parameter is mentioned in Tables 19 and 20. We select a weight of 0.25 for c''_{k+1}. We consider 15 and 5 for the number of future and previous frames, respectively.

Table 17. Comparison of proposed algorithms over the validation dataset.

Algorithm	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
BoT-SORT-Reid	83.277	80.718	70.607	9312	21432	429	675
Experiment 2	84.444	81.043	71.403	9183	21084	381	681
Experiment 3	84.436	80.891	71.219	9378	21144	372	660
Main experiment	85.355	81.011	71.749	9252	21090	357	657

Table 18. Main experiment: An ablation study over the number of future frames. ℓ: number of future frames for near-online simulation.

ℓ	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
1	84.768	80.842	71.286	9453	21135	384	672
2	84.38	80.976	71.125	9396	20976	384	663
3	84.599	80.987	71.237	9363	21012	363	657
4	84.641	80.974	71.31	9330	21054	375	672
5	84.451	80.941	71.285	9342	21096	375	675
6	84.904	80.946	71.434	9315	21123	366	660
7	85.289	80.941	71.631	9300	21150	363	657
8	85.147	80.995	71.721	9282	21090	354	654
9	85.345	80.98	71.743	9294	21102	354	645
10	85.148	81.037	71.721	9246	21057	354	660
11	85.148	81.037	71.722	9246	21057	354	660
12	85.341	81.045	71.746	9249	21042	354	651
13	85.341	81.048	71.746	9249	21036	354	657
14	85.355	81.011	71.749	9252	21090	357	657
15	85.334	81.008	71.747	9255	21093	357	660

Table 19. Main experiment: An ablation study over the weight given to c''_{k+1}. ℓ: number of future frames for near-online simulation. ℓ varies between 6 and 10.

ℓ	Weight	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
6	0.15	84.936	81.143	71.628	9141	21003	342	642
6	0.2	84.858	81.058	71.477	9141	21126	357	651
6	0.25	84.904	80.946	71.434	9315	21123	366	660
6	0.3	84.565	80.841	71.207	9396	21195	384	675
6	0.35	84.278	80.67	71.038	9474	21360	417	681
7	0.15	84.937	81.139	71.629	9144	21009	339	642
7	0.2	84.912	81.073	71.59	9159	21096	345	645
7	0.25	85.289	80.941	71.631	9300	21150	363	657
7	0.3	84.51	80.848	71.193	9387	21189	387	681
7	0.35	84.475	80.781	71.125	9351	21315	405	678
8	0.15	84.937	81.139	71.629	9144	21009	339	642
8	0.2	84.924	81.119	71.594	9150	21030	345	651
8	0.25	85.147	80.995	71.721	9282	21090	354	654
8	0.3	84.528	80.842	71.192	9360	21228	384	678
8	0.35	84.488	80.813	71.131	9324	21294	402	675
9	0.15	84.937	81.139	71.629	9144	21009	339	642
9	0.2	84.924	81.11	71.627	9177	21021	342	651
9	0.25	85.345	80.98	71.743	9294	21102	354	645
9	0.3	84.621	80.922	71.383	9288	21189	366	666
9	0.35	84.972	80.889	71.489	9207	21300	390	669
10	0.15	84.935	81.141	71.629	9147	21003	339	645
10	0.2	84.928	81.121	71.629	9153	21027	342	651
10	0.25	85.148	81.037	71.721	9246	21057	354	660
10	0.3	84.597	80.941	71.373	9252	21192	369	663
10	0.35	85.099	80.837	71.556	9249	21327	405	681

Table 20. Main experiment: An ablation study over the weight given to c''_{k+1}. ℓ: number of future frames for near-online simulation. ℓ varies between 11 and 15.

ℓ	Weight	IDF1(↑)	MOTA(↑)	HOTA(↑)	FP(↓)	FN(↓)	IDSW(↓)	Frag(↓)
11	0.15	84.959	81.102	71.591	9135	21072	345	645
11	0.2	84.922	81.119	71.593	9153	21027	345	651
11	0.25	85.148	81.037	71.722	9246	21057	354	660
11	0.3	84.595	80.941	71.372	9255	21186	372	663
11	0.35	85.044	80.859	71.501	9246	21297	402	675
12	0.15	84.943	81.119	71.631	9150	21036	339	642
12	0.2	84.922	81.119	71.592	9153	21027	345	651
12	0.25	85.341	81.045	71.746	9249	21042	354	651
12	0.3	84.849	80.915	71.502	9336	21159	360	666
12	0.35	85.022	80.816	71.486	9318	21291	405	678
13	0.15	84.943	81.119	71.631	9150	21036	339	642
13	0.2	84.922	81.119	71.592	9153	21027	345	651
13	0.25	85.341	81.048	71.746	9249	21036	354	657
13	0.3	84.813	80.82	71.473	9417	21228	363	666
13	0.35	84.671	80.811	71.339	9345	21279	399	678
14	0.15	84.937	81.117	71.594	9150	21036	342	642
14	0.2	84.922	81.119	71.592	9153	21027	345	651
14	0.25	85.355	81.011	71.749	9252	21090	357	657
14	0.3	85.367	80.842	71.706	9381	21225	366	660
14	0.35	85.059	80.811	71.524	9312	21306	405	678
15	0.15	84.938	81.119	71.595	9147	21036	342	642
15	0.2	84.918	81.11	71.59	9162	21033	345	654
15	0.25	85.334	81.008	71.747	9255	21093	357	660
15	0.3	85.367	80.842	71.705	9381	21225	366	660
15	0.35	84.636	80.815	71.332	9333	21279	405	672