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

Pratyusha Musunuru
Arizona State University
pmusunu1@asu.edu

Yuchao Li
Arizona State University
yuchaoli@asu.edu

Jamison Weber
Arizona State University
jweber@asu.edu

Dimitri Bertsekas
Arizona State University
dimitrib@mit.edu

Abstract

In this work, we consider data association problems involving multi-object tracking (MOT). In particular, we address the challenges arising from object occlusions. We propose a framework called approximate dynamic programming track (ADPTrack), which applies dynamic programming principles to improve an existing method called the base heuristic. Given a set of tracks and the next target frame, the base heuristic extends the tracks by matching them to the objects of this target frame directly. In contrast, ADPTrack first processes a few subsequent frames and applies the base heuristic starting from the next target frame to obtain tentative tracks. It then leverages the tentative tracks to match the objects of the target frame. This tends to reduce the occlusion-based errors and leads to an improvement over the base heuristic. When tested on the MOT17 video dataset, the proposed method demonstrates a 0.7% improvement in the association accuracy (IDF1 metric) over a state-of-the-art method that is used as the base heuristic. It also obtains improvements with respect to all the other standard metrics. Empirically, we found that the improvements are particularly pronounced in scenarios where the video data is obtained by fixed-position cameras.

1 Introduction

In this work, we consider the problem of *multi-object tracking* (MOT), which can be viewed as a special case of the multidimensional assignment (MDA) problem [17]. It involves the assignment of track identifiers to objects in motion over a sequence of image frames. In general, ensuring consistent identifiers over long sequences of frames is a formidable challenge, especially when the assignments are computed in real-time. To strike a good balance of performance and computational expedience, the so-called *online tracking* methods have been researched in the literature [7, 64, 1, 58]. Given a set of tracks and a new image frame, referred to as a *target frame*, the online tracking methods extend the given tracks by assigning objects in the target frame to the tracks. During this process, they rely exclusively on the information of objects' movement and/or appearance in the tracks and the target frame. This makes them prone to errors in cases where objects become partially/fully occluded over one or more contiguous frames.

To overcome the occlusion challenge, we propose a method based on approximate dynamic programming (also known as reinforcement learning) techniques, which we call *approximate dynamic programming track* (ADPTrack for short). It relies on an arbitrary given online tracking method and improves upon it. In particular, given a set of tracks and a target frame, ADPTrack collects a few additional subsequent frames beyond the target frame and applies the online tracking method to

construct tentative tracks starting from the target frame. It then relies on the information in the given tracks and the tentative tracks for assigning objects in the target frame to the given tracks. Since future information beyond the target frame is needed, our scheme can be viewed as a *near-online* method. Despite the additional computation compared with the online methods, the proposed method offers a systematic approach to address mismatches due to occlusion.

Some existing near-online methods [13, 20] share similarities with our proposed framework. In particular, given a set of tracks, a target frame, and a few subsequent frames beyond it, they construct tentative tracks of objects in the target frame and the subsequent frames. These tentative tracks either directly extend the given tracks to the objects in the target frame, or provide information that is useful for computing weights on arcs connecting these two. Despite the similarity, these schemes have been designed based on heuristic grounds and/or require specific neural networks. In contrast, ADPTrack is a general framework that is built on the connection between MOT and dynamic programming (DP). The process of constructing tentative tracks is viewed as performing *near-online simulation*. These tracks are used to approximate the *optimal value function* in the context of DP. As a result, ADPTrack is flexible and can leverage arbitrary online tracking methods.

To evaluate the performance of the ADPTrack framework, we apply as the base heuristic the BoT-SORT method [1], one of the state-of-the-art open-source online MOT algorithms at the time of writing. We addressed the problems in the MOT17 dataset [37] and obtained an overall relative improvement of 0.7% in the IDF1 score 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 is flexible to leverage any given online tracking method. Fig. 1 provides a frame-by-frame comparison of BoT-SORT and ADPTrack with BoT-SORT as base heuristic for a fixed video example and illustrates characteristic occlusion scenarios where ADPTrack outperforms BoT-SORT.

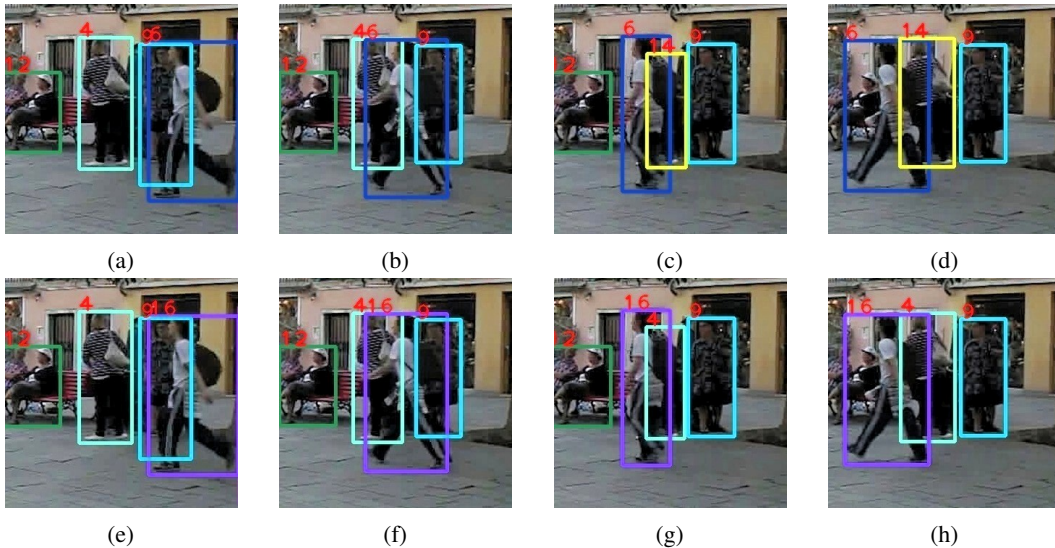


Figure 1: Frame-by-frame comparison of BoT-SORT (a-d) versus ADPTrack with BoT-SORT as the base heuristic (e-h) in an example. Using BoT-SORT directly, person (4) in (a) is erroneously assigned to another person’s identifier (14) in (c)-(d) after occluded by the person (6) in (b). When applying ADPTrack with BoT-SORT as the base heuristic, the same person (4) is assigned with the same identifier in (a) and (c)-(d) despite being occluded by the person (16) [the person (6) in (b)].

2 Related Work

Multi-Object Tracking: We provide a brief discussion of online and near-online tracking methods that are closely related to ADPTrack. Online tracking methods perform frame-by-frame association based on the information contained in the given tracks and the target frame. They often involve motion [1, 40, 10, 16] and/or appearance models [1, 56, 53, 66, 32, 15, 11, 2, 62]. A motion model estimates the position and size of an object in the next frame using the object’s velocity. Typically, the

motion model is a Kalman filter with the assumption that the object moves at a constant velocity [7, 64, 1, 56] or at a linear combination of recent frames’ velocities [58]. An appearance model extracts the visual information of an object to compute similarity scores for performing the assignment. One obtains similarity scores between objects either via deep neural networks directly [30, 52, 61, 50, 60] or using a distance metric between deep appearance features [1, 56, 12, 54, 65]. Several studies [19, 43, 57, 48, 21, 27, 67, 49, 14, 23, 36, 34, 9, 44, 39] directly leverage aggregated appearance information of given tracks for assignment, indicating the importance of past information.

Near-online tracking methods perform frame-by-frame association similar to online tracking, whilst leveraging a limited number of subsequent frames [13, 25, 41, 18, 63]. This effectively introduces a delay for the benefit of more accurate assignments. In [20], the authors generate tentative tracks by visual comparison and generate assignment weights using three different neural networks. In [59], the authors compartmentalize the problem locally around a time window and perform assignments for the target frame using flow-based optimization techniques. In contrast, ADPTrack builds upon existing online tracking methods and therefore does not rely on additional offline training.

Approximation in Value Space: Approximation in value space [4] is a broad class of reinforcement learning methods, which have been essential in some high-profile success stories, such as AlphaGo [46], AlphaZero [47, 45], and TD-Gammon [51]. The key idea in approximation in value space is to obtain approximations of the optimal value function in the context of DP. One representative method is called the *rollout algorithm*, which improves upon a given base heuristic via real-time simulation and has been applied in a variety of applications, e.g., [6, 5, 26, 8, 55, 31]. While our approach is not mathematically equivalent to the rollout algorithm, it uses several rollout ideas, including the simulation of a base heuristic to perform approximation in value space.

3 Preliminaries and Mathematical Model

In this section, we first present an overview of the MDA problem as a generalization of MOT, its DP formulation, and its exact solution method via DP. Then we provide a mathematical description of a typical online tracking method using the DP formulation introduced earlier.

3.1 Multi-Object Tracking as Multi-Dimensional Assignment

An instance of the N -dimensional assignment problem is represented by an $(N + 1)$ -partite graph arranged in layers $\mathcal{N}_0, \mathcal{N}_1, \dots, \mathcal{N}_N$, 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} , $k = 0, 1, \dots, N - 1$. We refer to a subset of $N + 1$ nodes i_0, i_1, \dots, i_N , where $i_k \in \mathcal{N}_k$ for all k , and their corresponding arcs $(i_0, i_1), (i_1, i_2), \dots, (i_{N-1}, i_N)$ as a *grouping*. Each grouping has an associated value. A feasible solution to an instance of the N -dimensional assignment problem is a set of m node-disjoint groupings. A solution is optimal if the sum of all grouping values is maximized. When $N = 1$, the problem is called *bipartite matching* and can be solved exactly in polynomial time. We focus on the case of the MDA problem where $N > 1$, which is NP-hard [38], and hence only approximate solutions are known.

When modeling MOT as an MDA, 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 (complete) track. A given track with a length smaller than $N + 1$ is viewed as a partial grouping. The value of a grouping should be large if the object associated with each frame is consistent according to some ground truth. In particular, the optimal solution assigns each object to the groupings as given by the ground truth.¹

3.2 Dynamic Programming for Multidimensional Assignment Problems

Let us now formulate the MDA problem as a deterministic DP problem. The exact solution to this DP problem is intractable. On the other hand, the DP formulation brings to bear approximation in

¹Note 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 situations. For the convenience of presentation, we make the simplifying assumption that there are m objects in each frame throughout our discussion. On the other hand, the implementation of our solution handles the cases where the numbers of objects vary over frames.

value space, a major approach for approximate DP and reinforcement learning, which forms the basis of our proposed algorithm in Section 4.

Generally, deterministic DP is used to solve a problem of sequential decision-making over N stages, by breaking it down into a sequence of simpler single-stage problems. It aims to find a sequence of decisions or actions u_0, \dots, u_{N-1} by generating a corresponding sequence of optimal value functions J_1^*, \dots, J_{N-1}^* . The algorithm uses a known value function J_N^* to compute the next value function J_{N-1}^* , by solving a single-stage decision problem whose optimization variable is u_{N-1} . It then uses J_{N-1}^* to compute J_{N-2}^* , and proceeds similarly to compute all the remaining value functions J_{N-3}^*, \dots, J_1^* .

More specifically, a deterministic DP problem involves a discrete-time dynamic system of the form $x_{k+1} = f_k(x_k, u_k)$, $k = 0, \dots, N-1$, where k is a time index, x_k is called the *state* of the system at time k , which belongs to a set X_k , and u_k is the *control*, to be selected at time k from a given set U_k . The function f_k describes the mechanism by which the state is updated from k to $k+1$ under the influence of a control. Given an initial state x_0 , in our DP problem we aim to maximize the value of a given terminal function G , i.e., to select the sequence of controls (u_0, \dots, u_{N-1}) such that the value of $G(x_N)$ is maximized.

To model the MDA problem as a DP problem, we define the set U_k as the collection of all bipartite matchings between \mathcal{N}_k and \mathcal{N}_{k+1} , i.e., each of its element u_k is a set of arcs $\{(i_n, j_n) \mid n = 1, \dots, m\}$ that forms a legitimate assignment between nodes in \mathcal{N}_k and \mathcal{N}_{k+1} . The state evolution is given by $f_0(x_0, u_0) = u_0$, and $f_k(x_k, u_k) = (x_k, u_k)$ for $k = 1, \dots, N-1$, and x_0 is an artificial state. In particular, the state $x_k = (u_0, \dots, u_{k-1})$ is the set of m partial groupings with objects the first $k+1$ layers. The value that we aim to maximize is $G(x_N) = G(u_0, \dots, u_{N-1})$. Here x_N represents a feasible set of m groupings (as defined in Section 3.1), and $G(x_N)$ represents the sum of the values of these m groupings.

The exact DP algorithm involves computing a sequence of optimal value functions J_k^* , $k = 1, \dots, N$. It first sets

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

and then computes backwards

$$J_k^*(x_k) = \max_{u_k \in U_k} J_{k+1}^*(x_k, u_k), \quad \text{for all } x_k, k = 1, \dots, N-1. \quad (1)$$

Having completed this calculation, it then computes the optimal controls $(u_0^*, \dots, u_{N-1}^*)$ via

$$u_k^* \in \arg \max_{u_k \in U_k} J_{k+1}^*(x_k^*, u_k), \quad k = 0, \dots, N-1, \quad (2)$$

where x_0^* is the artificial state, and $x_{k+1}^* = f_k(x_k^*, u_k^*)$. Although theoretically appealing, this exact method cannot be used to solve MOT due to reasons discussed later.

3.3 Online Tracking Methods

Let us now provide a mathematical description of a typical online tracking method using the DP terminology introduced earlier, which we will use as a base heuristic. Given the k th frame, we denote by I_k^j the information associated with the j th object in this frame that is useful for matching. In particular, we assume that $I_k^j = (v_k^j, p_k^j)$, where v_k^j is the visual information associated with the j th object in the k th frame, and p_k^j is the position and size information associated with the same object. They are used in the appearance models and motion models discussed in Section 2. Suppose that m tracks have been formed for objects contained in the 0th to k th frames via $x_k = (u_0, \dots, u_{k-1})$. We denote by $T^i(x_k)$ the information associated with the i th track, so that

$$T^i(x_k) = (v_0^i, v_1^i, \dots, v_k^i, p_0^i, p_1^i, \dots, p_k^i).$$

Note that here the indices of the objects have been assigned according to the indices of the tracks that they belong to, which depends on $x_k = (u_0, \dots, u_{k-1})$. Given a track $T^i(x_k)$ and the information vector associated with the j th object in the $(k+1)$ th frame, the base heuristic computes a weight $w_{k+1}^{ij}(x_k)$ by some function H , i.e.,

$$w_{k+1}^{ij}(x_k) = H(T^i(x_k), I_{k+1}^j). \quad (3)$$

Once the weights $w_{k+1}^{ij}(x_k)$ are computed for all track-object pairs (i, j) , a standard bipartite matching is performed to extend the tracks from $T^i(x_k)$ to $T^i(x_{k+1})$. From the perspective of MOT, the weights $w_{k+1}^{ij}(x_k)$ measure the ‘‘similarity’’ between the first $k + 1$ objects in grouping i specified by x_k , and the object j in the $(k + 1)$ th frame. This can be viewed as an approximation to the grouping values and leads to errors in the assignment. As we will show shortly, our proposed framework improves upon the online tracking methods by constructing better grouping value approximations.

4 Approximate Dynamic Programming Track

The exact and approximate methods discussed in Section 3 have their respective drawbacks when used for MOT. The DP method faces two major challenges. The first applies specifically to MOT, namely that evaluating the quality of a matching sequence is difficult and thus the function G is not known. Secondly, even if G were known, computing J_{k+1}^* is still intractable as the size of X_{k+1} grows exponentially with k . The online tracking methods use approximate grouping values $w_{k+1}^{ij}(x_k)$ to approximate G [cf.(3)]. They allow real-time computation but also introduce assignment errors due to the simplifications introduced by the weights. This is particularly true when the objects are occluded in the target frame. In contrast, the proposed ADPTrack framework leverages the DP formulation and improves the quality of approximation used in online tracking methods by considering the information contained in the subsequent frames, thus addressing the challenges posed by occlusion. In what follows, we describe the ADPTrack as a form of approximation in value space, provide the procedure through which the approximation to the optimal value function is computed, and provide details on the implementation with BoT-SORT as the base heuristic.

4.1 Approximation in Value Space for Multidimensional Assignment

The approximation in value space method introduces various approximations to the components in the maximization from Equation (2). One such approximation replaces value functions J_{k+1}^* , $k = 0, \dots, N - 1$, in (2) with value function approximations \tilde{J}_{k+1} that can be computed efficiently. To facilitate the corresponding maximization calculation, we consider functions \tilde{J}_{k+1} of the form

$$\tilde{J}_{k+1}(x_k, u_k) = \sum_{(i,j) \in u_k} c_{k+1}^{ij}(x_k) \quad (4)$$

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 weight for arc (i, j) where $i \in \mathcal{N}_k$ and $j \in \mathcal{N}_{k+1}$. As a result, given the current state $x_k = (\tilde{u}_0, \dots, \tilde{u}_{k-1})$, the control is selected via the maximization

$$\tilde{u}_k \in \arg \max_{u_k \in U_k} \tilde{J}_{k+1}(x_k, u_k), \quad (5)$$

which is equivalent to solving a bipartite matching problem, with weights on arcs (i, j) specified by $c_{k+1}^{ij}(x_k)$. Compared with the weight $w_{k+1}^{ij}(x_k)$ used in online methods, the weight $c_{k+1}^{ij}(x_k)$ represents the ‘‘similarity’’ between the first $k + 1$ objects (contained in 0th to k th frames) in grouping i assigned by x_k , and the tentative track starting from the object j in the $(k + 1)$ th frame. Here the tentative track involves $\ell + 1$ objects starting from the $(k + 1)$ th frame, with ℓ being a *truncated horizon* in DP terminology. Thus we refer to $c_{k+1}^{ij}(x_k)$ as *similarity scores* between tracks. An illustration of the key components involved in computing \tilde{J}_{k+1} is shown in Fig. 2a. Intuitively, these scores contain more information than $w_{k+1}^{ij}(x_k)$, which tends to improve the assignment quality, as we discuss next.

4.2 Value Function Approximation via the Base Heuristic

The computation of similarity scores relies on tentative tracks, which are obtained by solving an MOT problem starting from the $(k + 1)$ th frame and ending at $(k + \ell + 1)$ th frame. We refer to the process of solving this problem as the near-online simulation; see Fig. 2b. For objects contained in the $(k + 1)$ th frame to $(k + \ell + 1)$ th frame, we treat the $(k + 1)$ th frame as if it is the first frame, and set $T^j(\bar{x}_0) = I_{k+1}^j$, where \bar{x}_0^{k+1} is an artificial initial state starting at $(k + 1)$ th frame, j is the index

associated with objects in the $(k+1)$ th frame. We apply the online tracking method to solve the MOT problem starting from the $(k+1)$ th frame and ending with $(k+\ell+1)$ th frame. The information associated with the tentative track that starts from the j th object in $(k+1)$ th frame is denoted by $T^j(\bar{x}_\ell^{k+1})$, where $\bar{x}_\ell^{k+1} = (\bar{u}_{k+1}, \dots, \bar{u}_{k+\ell+1})$. The information contained in $T^j(\bar{x}_\ell^{k+1})$ includes the information I_{k+1}^j , and will be used to compute the weights associated with the object j , as we discuss next. This is the key to overcome the challenges associated with occlusion. Intuitively, if some object is partially occluded in the target frame, matching based on its visual information in the target frame may lead to error. In contrast, ADPTrack corrects this error by incorporating information in the subsequent frames, where the object may no longer be occluded. Moreover, ADPTrack provides a flexible platform for constructing tentative tracks by using arbitrary existing online tracking methods and through online simulation. This is an essential idea in many approximate DP algorithms and sets apart ADPTrack from existing near-online methods that often require additional off-line training.

Suppose that m tracks have been formed for objects contained in 0th to k th frames according to the matching x_k . The information associated with each track i is denoted by $T^i(x_k)$. For the tentative tracks formed according to \bar{x}_ℓ^{k+1} via the base heuristic starting from the j th object in $(k+1)$ th frame, the information associated with it is $T^j(\bar{x}_\ell^{k+1})$. The weight term $c_{k+1}^{ij}(x_k)$ is given by

$$c_{k+1}^{ij} = (1 - \alpha)w_{k+1}^{ij}(x_k) + \alpha z_{k+1}^{ij}(x_k), \quad (6)$$

where α is a tuning parameter, and $w_{k+1}^{ij}(x_k)$ is given by the base heuristic via (3), as a similarity measure between objects in i th track and j th object in the target frame. The term $z_{k+1}^{ij}(x_k)$ reflects the similarity between objects with the information $T^i(x_k)$ and $T^j(\bar{x}_\ell^{k+1})$, and therefore is a similarity measure between given and tentative tracks; see Fig. 2c. In particular, it is defined as

$$z_{k+1}^{ij}(x_k) = F(T^i(x_k), T^j(\bar{x}_\ell^{k+1})), \quad (7)$$

where F is a user-defined function, which may depend on the base heuristic. It is used to describe the ‘similarity’ of objects contained in the i th given track and the tentative track that starts from object j in $(k+1)$ th frame. Intuitively, this helps to overcome the difficulty due to occlusion, as discussed earlier, and a particular implementation is given in Section 4.3. Note that \bar{x}_ℓ^{k+1} is independent of x_k . Once all track-object pairs are computed, we compute the cost function approximation \tilde{J}_{k+1} according to (4), and perform the maximization calculation accordingly. A pseudocode description of ADPTrack for matching one frame is given as Algorithm 1.

Algorithm 1 ADPTrack with a Base Heuristic

- Input:** Tracks $T^i(x_k), i = 1, \dots, m$; target frame \mathcal{N}_{k+1} , subsequent frames $\mathcal{N}_{k+2}, \dots, \mathcal{N}_{k+\ell+1}$.
Output: Tracks $T^i(x_{k+1}), i = 1, 2, \dots, m$.
- 1: Set $w_{k+1}^{ij}(x_k) \leftarrow H(T^i(x_k), I_{k+1}^j)$ according to (3) for all $i = 1, \dots, m$ and $j \in \mathcal{N}_{k+1}$.
 - 2: Compute $T^j(\bar{x}_\ell^{k+1})$ via simulation the base heuristic on $\mathcal{N}_{k+1}, \dots, \mathcal{N}_{k+\ell+1}$, for all $j \in \mathcal{N}_{k+1}$.
 - 3: Set $z_{k+1}^{ij}(x_k) \leftarrow F(T^i(x_k), T^j(\bar{x}_\ell^{k+1}))$, for all $i = 1, \dots, m$ and $j \in \mathcal{N}_{k+1}$.
 - 4: Set $c_{k+1}^{ij} \leftarrow \alpha w_{k+1}^{ij}(x_k) + (1 - \alpha)z_{k+1}^{ij}(x_k)$, for all (i, j) .
 - 5: Compute maximum weight bipartite matching \tilde{u}_k with weights given by $c_{k+1}^{ij}(x_k)$; cf. (5).
 - 6: Set $x_{k+1} \leftarrow (x_k, \tilde{u}_k)$, compute $T^i(x_{k+1})$ for $i = 1, \dots, m$.
-

4.3 ADPTrack with BoT-SORT as Base Heuristic

As an example implementation, we describe ADPTrack with BoT-SORT [1] as a base heuristic. The testing results of this implementation are reported in Section 5. BoT-SORT is an online tracking method that uses a Kalman filter [29] as a motion model and a reidentification neural network [24] as an appearance model. We apply BoT-SORT directly to generate the tentative tracks in the near-online simulation. Once we obtain the tentative tracks, we compute $z_{k+1}^{ij}(x_k)$ according to (7). For our numerical studies, F performs an object-to-object visual comparison using the appearance model used in BoT-SORT and computes an average of all the pairwise visual similarity scores to obtain an overall score. In particular, cosine-similarity (CS) is used in BoT-SORT to extract a visual similarity

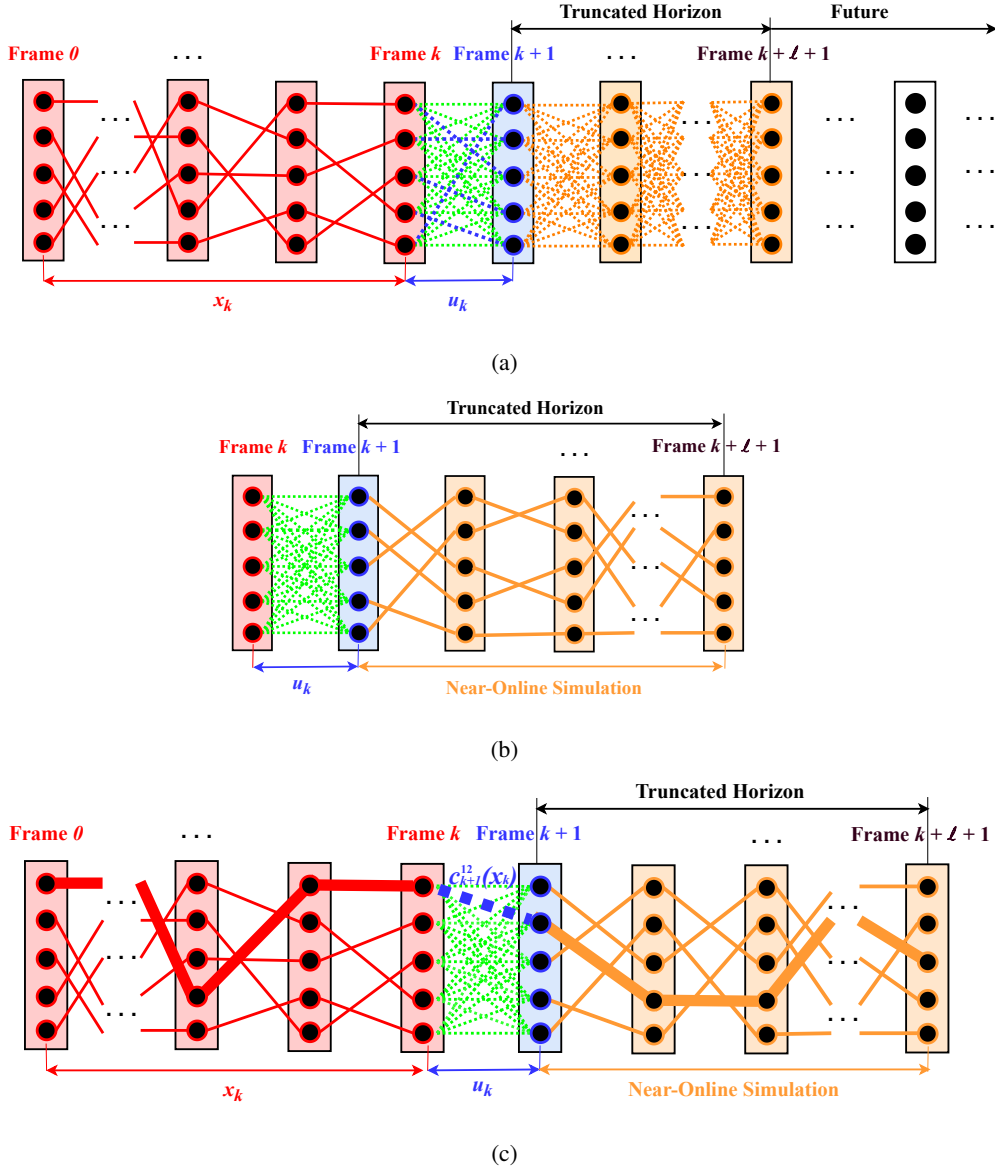


Figure 2: (a) Overview of key components involved in ADPTrack for MDA (MOT). Red layers and respective arcs represent objects formed in partial groupings (given tracks) up to frame k . Blue layer $k + 1$ represents the target frame. Green arcs between k and $k + 1$ layers represent all pairwise arcs between \mathcal{N}_k and \mathcal{N}_{k+1} . Dashed blue matching indicates a selected control u_k . Orange layers and arcs represent subsequent frames. (b) Visualization of near-online simulation in ADPTrack. The base heuristic is applied to simulate the solution of the MOT problem starting at frame $k + 1$ and ending at frame $k + \ell + 1$. The obtained tentative tracks are shown by the orange solid lines. (c) Illustration of computation of similarity scores $c_{k+1}^{ij}(x_k)$. For example, the weight $c_{k+1}^{12}(x_k)$ is assigned to the bold blue dashed arc, which is dependent on the given track (bold red) and the tentative track (bold orange) that the arc connects.

score between objects, where CS takes as inputs a visual information pair (v_a^i, \bar{v}_b^j) and returns a high value if they appear ‘alike’ and low otherwise. Roughly speaking, we define the function F as

$$F(T^i(x_k), T^j(\bar{x}_\ell^{k+1})) = \frac{1}{s \cdot \ell} \sum_{v_a^i \in \bar{V}^i(x_k)} \sum_{\bar{v}_b^j \in V^j(\bar{x}_\ell^{k+1})} \text{CS}(v_a^i, \bar{v}_b^j), \quad (8)$$

where $\bar{V}^i(x_k) = (v_{q-s+1}^i, v_{q-s+2}^i, \dots, v_q^i)$ represents the high quality visual information of i th given track where $q \leq k$, s is a tuning parameter, and $V^j(\bar{x}_\ell^{k+1}) = (\bar{v}_{k+1}^j, \bar{v}_{k+2}^j, \dots, \bar{v}_{k+\ell+1}^j)$ represents the entire visual information associated with the tentative track starting from the object j in \mathcal{N}_{k+1} . Further details on F , the calculation of q , and the choice of s are provided in the appendix.

5 Experimental Study

In this section, we describe the results of our benchmark evaluation that compares ADPTrack using BoT-SORT as a base heuristic, with BoT-SORT itself. We apply visual modules from BoT-SORT without modification for our experiments. In particular, we use a Yolox model [22] trained by [64] for object detection and FastReid’s SBS-50 model [24] fine-tuned by [1] as an appearance model, as has been done for BoT-SORT. We ran all our experiments on a V100 20-core GPU.

Dataset and Metrics: We used the MOT17 [37] 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, which we refer to as the validation dataset. This is because the first half of the training dataset has been used by the base heuristic to train the re-identification network [1] (i.e. the appearance model). Note that our solution does not require any additional training. 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.

We adopt *clear metrics* [3] (among others) to evaluate ADPTrack with respect to the baseline and other state-of-the-art methods. More specifically, we describe our results with respect to the following metrics: IDF1 [42], *multi-object tracking accuracy* (MOTA) [3], *higher-order tracking accuracy* (HOTA) [33], *ID-switches* (IDSW), *fragmentations* (Frag), *false positives* (FP), and *false negatives* (FN). The IDF1 score mainly assesses association accuracy, and MOTA mainly assesses detection accuracy. 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 [28] to generate the scores according to clear metrics. As the focus of ADPTrack is handling temporarily occluded objects, we hypothesize a greater improvement in IDF1 and IDSW metrics.

Results: Table 1 shows the overall scores of both tracking methods over the validation dataset, and Table 2 shows the overall scores achieved by both of tracking methods over the benchmark dataset. We present the video-wise improvement scores (i.e., the scores obtained for specific videos) of ADPTrack as compared to the BoT-SORT algorithm for videos in the MOT17 dataset in Table 3. In all tables, the arrow \uparrow (\downarrow) in the first row indicates an increase (resp. decrease) in the corresponding score is more desirable. We provide the video-wise scores of all other metrics of ADPTrack relative to BoT-SORT on the MOT17 dataset (including graphical illustrations) in the Appendix. Moreover, we performed ablation studies over various components of ADPTrack. We also performed parameter variation studies for the tuning parameter, the number of subsequent frames, etc. We present these results in the Appendix.

Table 1: 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

Over the validation dataset in Table 1, 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 2, 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 outperforms (or is comparable to) several other state-of-the-art tracking methods (see Appendix).

Table 2: 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

Table 3: Video-wise improvement of ADPTrack over BoT-SORT for the videos in 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

With respect to specific video examples in Table 3, 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 the 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 increase in error can be attributed to fast-moving cameras. Therefore, the appearance information of the subsequent frames can vary significantly from those that came previously. ADPTrack with BoT-SORT as the base heuristic achieves an FPS (frames processed per second) rate of 0.8 compared to BoT-SORT which has 4.5. This is because ADPTrack involves near-online simulations and pairwise comparisons of objects from given and tentative tracks.

6 Conclusions

We have presented an approximate DP framework for rendering existing MOT solution methods more effective and more robust to object occlusions. The performance of our method on the validation and the test datasets is eminently promising and points to the potential of approximate DP ideas for addressing complex data association problems. Moreover, the representation of the base heuristic in our model is very general, suggesting that virtually any online tracking method with motion and appearance models can be adapted to improve performance against occlusion.

With respect to limitations, our solution is more computationally expensive than the base heuristic and introduces delay. Consequently, there may be practical disadvantages associated with moving from the online to the near-online setting. These limitations highlight a tradeoff between introducing delays/increasing time complexity and improving the quality of assignments. On the other hand, there is an opportunity for parallelization in computing the similarity scores since the computations necessary for each arc are independent. Moreover, there are other potential optimizations that we have not implemented, such as storing the appearance similarity scores instead of recomputing them in some cases.

Regarding future work, we hypothesize that generating tentative tracks from frames $k + 2$ or beyond may be useful for reducing error, particularly when the object is occluded in frame $k + 1$. This is because an occluded object may negatively influence similarity scores. Lastly, an interesting avenue for future research is the idea of using ADPTrack with a specified base heuristic as a base heuristic itself.

References

- [1] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. *BoT-SORT: Robust Associations Multi-Pedestrian Tracking*. 2022. arXiv: 2206.14651 [cs.CV].
- [2] Seung-Hwan Bae and Kuk-Jin Yoon. “Confidence-Based Data Association and Discriminative Deep Appearance Learning for Robust Online Multi-Object Tracking”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.3 (2018), pp. 595–610. DOI: 10.1109/TPAMI.2017.2691769.
- [3] Keni Bernardin and Rainer Stiefelwagen. “Evaluating Multiple Object Tracking Performance: The Clear Mot Metrics”. In: *EURASIP Journal on Image and Video Processing* 2008 (Jan. 2008). DOI: 10.1155/2008/246309.
- [4] Dimitri P. Bertsekas. *A Course in Reinforcement Learning*. Athena Scientific, 2023.
- [5] Dimitri P. Bertsekas and David A. Castanon. “Rollout Algorithms for Stochastic Scheduling Problems”. In: *Journal of Heuristics* 5 (1999), pp. 89–108.
- [6] Dimitri P. Bertsekas, John N. Tsitsiklis, and Cynara Wu. “Rollout Algorithms for Combinatorial Optimization”. In: *Journal of Heuristics* 3 (1997), pp. 245–262.
- [7] Alex Bewley et al. “Simple Online and Realtime Tracking”. In: *2016 Ieee International Conference on Image Processing (Icip)*. 2016, pp. 3464–3468. DOI: 10.1109/Icip.2016.7533003.
- [8] Siddhant Bhambri, Amrita Bhattacharjee, and Dimitri Bertsekas. “Reinforcement Learning Methods for Wordle: A POMDP/Adaptive Control Approach”. In: *arXiv preprint arXiv:2211.10298* (2022).
- [9] Jiarui Cai et al. *MeMOT: Multi-Object Tracking with Memory*. 2022. arXiv: 2203.16761 [cs.CV].
- [10] Jinkun Cao et al. *Observation-Centric SORT: Rethinking SORT for Robust Multi-Object Tracking*. 2023. arXiv: 2203.14360 [cs.CV].
- [11] Orcun Cetintas, Guillem Brasó, and Laura Leal-Taixé. *Unifying Short and Long-Term Tracking with Graph Hierarchies*. 2023. arXiv: 2212.03038 [cs.CV].
- [12] Long Chen et al. “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 2018. DOI: 10.1109/icme.2018.8486597. URL: <http://dx.doi.org/10.1109/ICME.2018.8486597>.
- [13] Wongun Choi. *Near-Online Multi-target Tracking with Aggregated Local Flow Descriptor*. 2015. arXiv: 1504.02340 [cs.CV].
- [14] Peng Chu et al. *TransMOT: Spatial-Temporal Graph Transformer for Multiple Object Tracking*. 2021. arXiv: 2104.00194 [cs.CV].
- [15] Qi Chu et al. *Online Multi-Object Tracking Using CNN-based Single Object Tracker with Spatial-Temporal Attention Mechanism*. 2017. arXiv: 1708.02843 [cs.CV].
- [16] Yunhao Du et al. *StrongSORT: Make DeepSORT Great Again*. 2023. arXiv: 2202.13514 [cs.CV].
- [17] Patrick Emami et al. “Machine Learning Methods for Data Association in Multi-object Tracking”. In: *Acm Computing Surveys* 53.4 (Aug. 2020), pp. 1–34. ISSN: 1557-7341. DOI: 10.1145/3394659. URL: [Http://Dx.Doi.Org/10.1145/3394659](http://dx.doi.org/10.1145/3394659).
- [18] Loïc Fagot-Bouquet et al. “Improving Multi-frame Data Association with Sparse Representations for Robust Near-online Multi-object Tracking”. In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 774–790. ISBN: 978-3-319-46484-8.
- [19] Kuan Fang et al. *Recurrent Autoregressive Networks for Online Multi-Object Tracking*. 2018. arXiv: 1711.02741 [cs.CV].
- [20] Weijiang Feng et al. “Near-Online Multi-Pedestrian Tracking via Combining Multiple Consistent Appearance Cues”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 31.4 (2021), pp. 1540–1554. DOI: 10.1109/TCSVT.2020.3005662.
- [21] Weitao Feng, Baopu Li, and Wanli Ouyang. “Multi-Object Tracking with Multiple Cues and Switcher-Aware Classification”. In: *2022 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. 2022, pp. 1–10. DOI: 10.1109/DICTA56598.2022.10034575.

- [22] Zheng Ge et al. *YOLOX: Exceeding YOLO Series in 2021*. 2021. arXiv: 2107.08430 [cs.CV].
- [23] Song Guo et al. *Online Multiple Object Tracking with Cross-Task Synergy*. 2021. arXiv: 2104.00380 [cs.CV].
- [24] Lingxiao He et al. *FastReID: A Pytorch Toolbox for General Instance Re-Identification*. 2020. arXiv: 2006.02631 [cs.CV].
- [25] Roberto Henschel, Yunzhe Zou, and Bodo Rosenhahn. “Multiple People Tracking Using Body and Joint Detections”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2019, pp. 770–779. DOI: 10.1109/CVPRW.2019.00105.
- [26] Jueming Hu et al. “Optimal Maintenance Scheduling under Uncertainties Using Linear Programming-enhanced Reinforcement Learning”. In: *Engineering Applications of Artificial Intelligence* 109 (2022), p. 104655.
- [27] Wei-Chih Hung et al. *SoDA: Multi-Object Tracking with Soft Data Association*. 2020. arXiv: 2008.07725 [cs.CV].
- [28] Arne Hoffhues Jonathon Luiten. *TrackEval*. <https://github.com/JonathonLuiten/TrackEval>. 2020.
- [29] Rudolph Emil Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: (1960).
- [30] Laura Leal-Taixé, Cristian Canton Ferrer, and Konrad Schindler. *Learning by tracking: Siamese CNN for robust target association*. 2016. arXiv: 1604.07866 [cs.LG].
- [31] Yuchao Li and Dimitri Bertsekas. “Most Likely Sequence Generation for N -grams, Transformers, Hmms, and Markov Chains, by Using Rollout Algorithms”. In: *Arxiv Preprint Arxiv:2403.15465* (2024).
- [32] Chao Liang et al. *Rethinking the Competition Between Detection and Reid in Multi-object Tracking*. 2022. arXiv: 2010.12138 [cs.CV].
- [33] Jonathon Luiten et al. “HOTA: A Higher Order Metric for Evaluating Multi-object Tracking”. In: *International Journal of Computer Vision* 129.2 (Oct. 2020), pp. 548–578. ISSN: 1573-1405. DOI: 10.1007/s11263-020-01375-2. URL: <http://dx.doi.org/10.1007/s11263-020-01375-2>.
- [34] Fan Ma et al. *Unified Transformer Tracker for Object Tracking*. 2022. arXiv: 2203.15175 [cs.CV].
- [35] Gerard Maggolino et al. *Deep OC-SORT: Multi-Pedestrian Tracking by Adaptive Re-Identification*. 2023. Arxiv: 2302.11813 (cs.CV).
- [36] Tim Meinhardt et al. *TrackFormer: Multi-Object Tracking with Transformers*. 2022. arXiv: 2101.02702 [cs.CV].
- [37] Anton Milan et al. *MOT16: A Benchmark for Multi-Object Tracking*. 2016. arXiv: 1603.00831 [cs.CV].
- [38] Duc Manh Nguyen, Hoai an Le Thi, and Tao Pham Dinh. “Solving the Multidimensional Assignment Problem by a Cross-entropy Method”. In: *J. Comb. Optim.* 27.4 (May 2014), pp. 808–823. ISSN: 1382-6905. DOI: 10.1007/S10878-012-9554-z. URL: <https://doi.org/10.1007/S10878-012-9554-z>.
- [39] Jiangmiao Pang et al. *Quasi-Dense Similarity Learning for Multiple Object Tracking*. 2021. arXiv: 2006.06664 [cs.CV].
- [40] Zheng Qin et al. *MotionTrack: Learning Robust Short-term and Long-term Motions for Multi-Object Tracking*. 2023. arXiv: 2303.10404 [cs.CV].
- [41] Akshay Rangesh et al. *TrackMPNN: A Message Passing Graph Neural Architecture for Multi-Object Tracking*. 2021. arXiv: 2101.04206 [cs.CV].
- [42] Ergys Ristani et al. *Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking*. 2016. arXiv: 1609.01775 [cs.CV].
- [43] Amir Sadeghian, Alexandre Alahi, and Silvio Savarese. *Tracking The Untrackable: Learning To Track Multiple Cues with Long-Term Dependencies*. 2017. arXiv: 1701.01909 [cs.CV].
- [44] Chaobing Shan et al. *Tracklets Predicting Based Adaptive Graph Tracking*. 2020. arXiv: 2010.09015 [cs.CV].
- [45] David Silver et al. “A General Reinforcement Learning Algorithm That Masters Chess, Shogi, and Go Through Self-play”. In: *Science* 362.6419 (2018), pp. 1140–1144.

- [46] David Silver et al. “Mastering the Game of Go with Deep Neural Networks and Tree Search”. In: *Nature* 529.7587 (2016), pp. 484–489.
- [47] David Silver et al. “Mastering the Game of Go Without Human Knowledge”. In: *Nature* 550.7676 (2017), pp. 354–359.
- [48] Jeany Son et al. “Multi-Object Tracking with Quadruplet Convolutional Neural Networks”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 3786–3795. DOI: 10.1109/CVPR.2017.403.
- [49] Peize Sun et al. *TransTrack: Multiple Object Tracking with Transformer*. 2021. arXiv: 2012.15460 [cs.CV].
- [50] ShiJie Sun et al. “Deep Affinity Network for Multiple Object Tracking”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.1 (2021), pp. 104–119. DOI: 10.1109/TPAMI.2019.2929520.
- [51] Gerald Tesauro and Gregory R. Galperin. “On-Line Policy Improvement Using Monte-Carlo Search”. In: *Proceedings of the 9th International Conference on Neural Information Processing Systems*. 1996, pp. 1068–1074.
- [52] Bing Wang et al. *Joint Learning of Siamese CNNs and Temporally Constrained Metrics for Tracklet Association*. 2016. arXiv: 1605.04502 [cs.CV].
- [53] Qiang Wang et al. *Multiple Object Tracking with Correlation Learning*. 2021. arXiv: 2104.03541 [cs.CV].
- [54] Zhongdao Wang et al. *Towards Real-Time Multi-Object Tracking*. 2020. arXiv: 1909.12605 [cs.CV].
- [55] Jamison W. Weber et al. “Distributed Online Rollout for Multivehicle Routing in Unmapped Environments”. In: *Arxiv Preprint Arxiv:2305.15596* (2023).
- [56] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. *Simple Online and Realtime Tracking with a Deep Association Metric*. 2017. arXiv: 1703.07402 [cs.CV].
- [57] Jiarui Xu et al. *Spatial-Temporal Relation Networks for Multi-Object Tracking*. 2019. arXiv: 1904.11489 [cs.CV].
- [58] Fan Yang et al. “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*. 2023, pp. 4799–4808.
- [59] Min Yang, Yuwei Wu, and Yunde Jia. “A Hybrid Data Association Framework for Robust Online Multi-Object Tracking”. In: *IEEE Transactions on Image Processing* 26.12 (Dec. 2017), pp. 5667–5679. ISSN: 1941-0042. DOI: 10.1109/tip.2017.2745103. URL: <http://dx.doi.org/10.1109/TIP.2017.2745103>.
- [60] Junbo Yin et al. *A Unified Object Motion and Affinity Model for Online Multi-Object Tracking*. 2020. arXiv: 2003.11291 [cs.CV].
- [61] Young-Chul Yoon et al. *Online Multiple Pedestrians Tracking using Deep Temporal Appearance Matching Association*. 2020. arXiv: 1907.00831 [cs.CV].
- [62] Fengwei Yu et al. *POI: Multiple Object Tracking with High Performance Detection and Appearance Feature*. 2016. arXiv: 1610.06136 [cs.CV].
- [63] Yang Zhang et al. “Multiplex Labeling Graph for Near-Online Tracking in Crowded Scenes”. In: *IEEE Internet of Things Journal* 7.9 (2020), pp. 7892–7902. DOI: 10.1109/JIOT.2020.2996609.
- [64] Yifu Zhang et al. “Bytetrack: Multi-object Tracking by Associating Every Detection Box”. In: *European conference on computer vision*. Springer. 2022, pp. 1–21.
- [65] Yifu Zhang et al. “FairMOT: On the Fairness of Detection and Re-identification in Multiple Object Tracking”. In: *International Journal of Computer Vision* 129.11 (Sept. 2021), pp. 3069–3087. ISSN: 1573-1405. DOI: 10.1007/s11263-021-01513-4. URL: <http://dx.doi.org/10.1007/s11263-021-01513-4>.
- [66] Zelin Zhao et al. *Tracking Objects as Pixel-wise Distributions*. 2022. arXiv: 2207.05518 [cs.CV].
- [67] Ji Zhu et al. *Online Multi-Object Tracking with Dual Matching Attention Networks*. 2019. arXiv: 1902.00749 [cs.CV].

A Appendix

A.1 Supplemental Results

We illustrate the IDF1 and IDSW scores of BoT-SORT and ADPTrack with BoT-SORT as the base heuristic for all the videos of the MOT17 dataset in Figs. 3 and 4, respectively. Figs. 5 and 6 illustrate the percentage improvement of ADPTrack over the base heuristic BoT-SORT across all the metrics over the validation and test dataset, respectively. We present the video-wise scores of base heuristic and ADPTrack over the validation and test datasets in Tables 4 and 5, respectively. In Table 6, we compare the overall scores of some state-of-the-art tracking methods with ADPTrack using BoT-SORT as the base heuristic. Upon acceptance, we will provide a link to the official scores on the MOT17 Challenge Website.

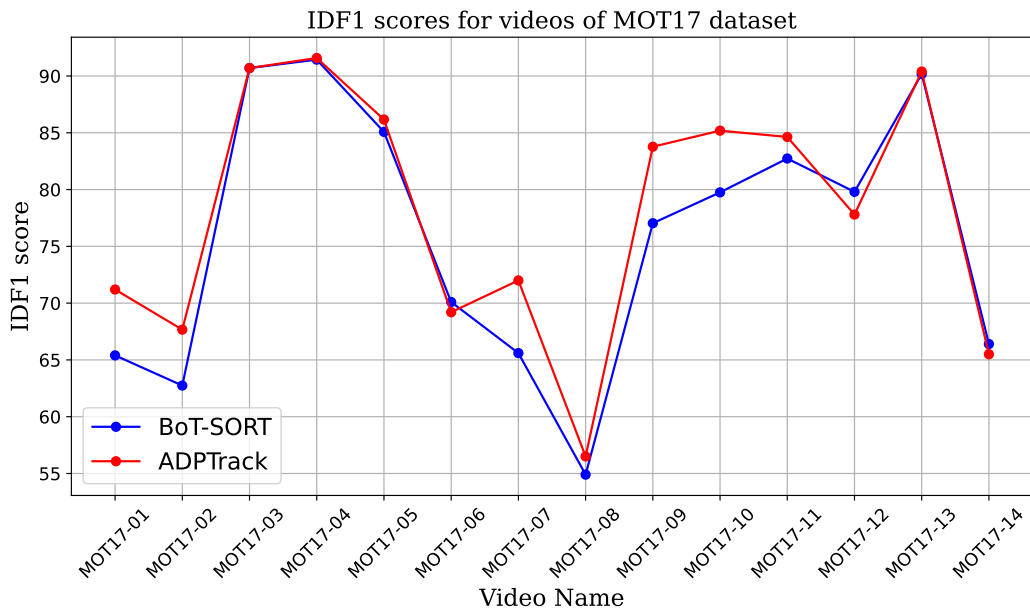


Figure 3: Video-wise IDF1(\uparrow) scores of BoT-SORT and ADPTrack with BoT-SORT as the base heuristic when applied to videos of the MOT17 dataset.

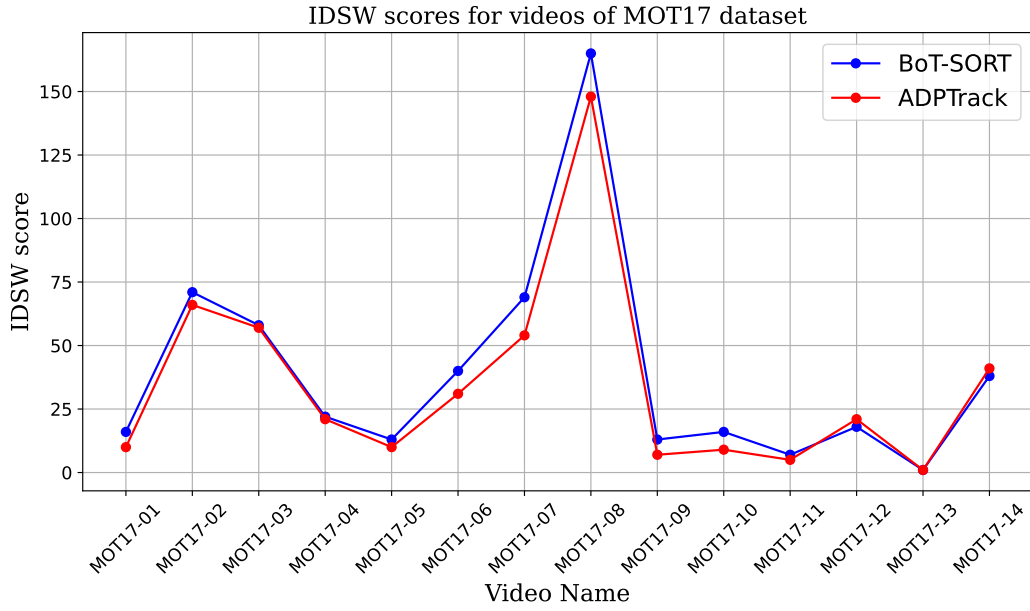


Figure 4: Video-wise IDSW(\downarrow) scores of BoT-SORT and ADPTrack with BoT-SORT as the base heuristic when applied to videos of the MOT17 dataset.

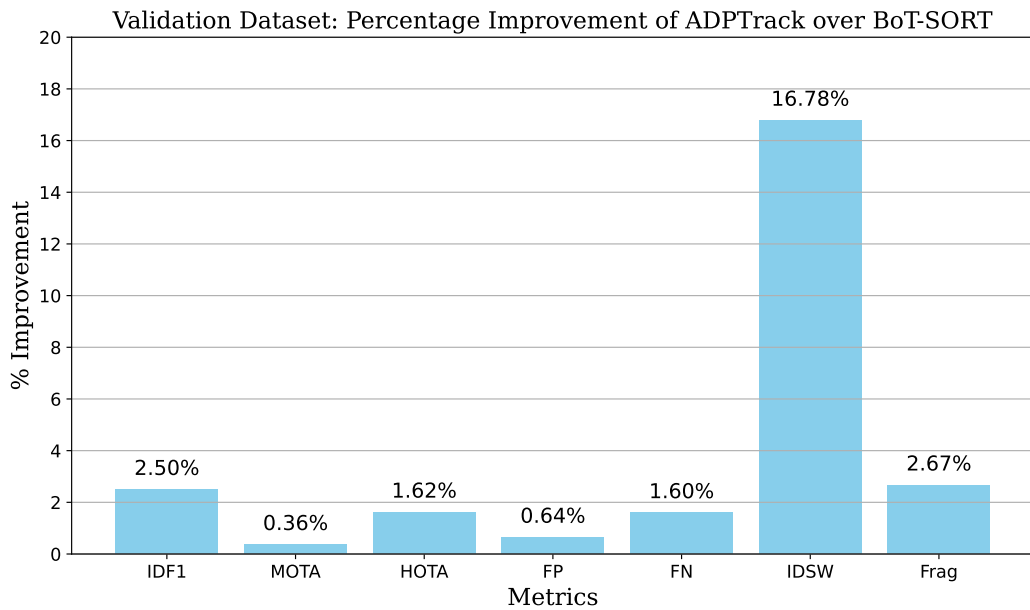


Figure 5: Percentage improvement of ADPTrack with BoT-SORT as the base heuristic over BoT-SORT itself across several MOT metrics on the validation dataset.

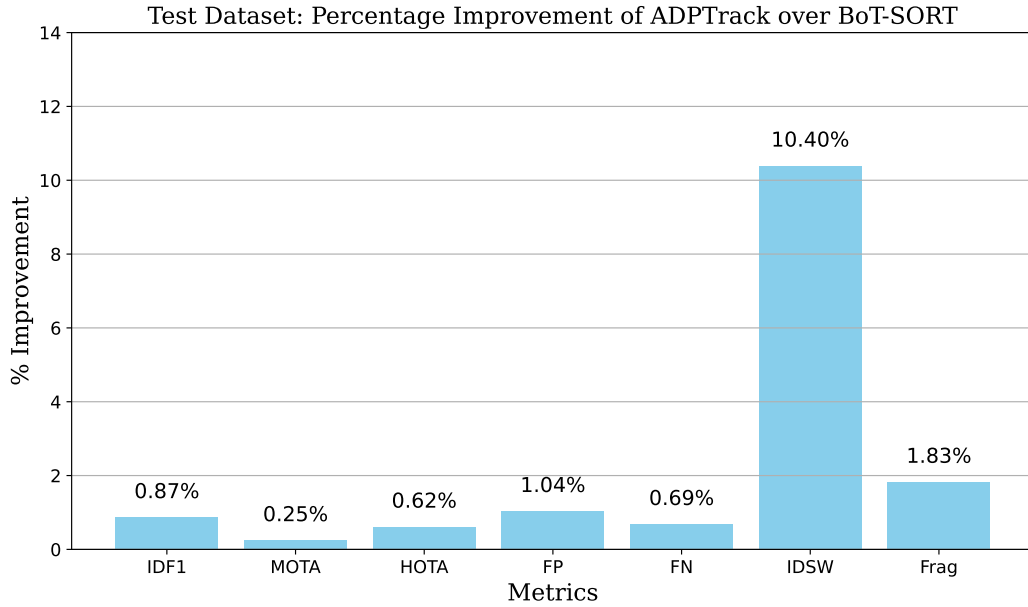


Figure 6: Percentage improvement of ADPTrack with BoT-SORT as the base heuristic over BoT-SORT itself across several MOT metrics on the test dataset.

Table 4: Video-wise scores of BoT-SORT 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 with BoT-SORT as a base heuristic when applied to the MOT17 dataset.

Video Name	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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 ADPTrack with BoT-SORT as a base heuristic, BoT-SORT itself, and other state-of-the-art tracking methods on the MOT17 test dataset.

Method	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
OCSORT [10]	77.5	78.0	63.2	15129	107055	1950	2040
Deep-OCSORT [35]	80.6	79.4	64.9	16572	98796	1023	2196
StrongSORT++ [16]	79.5	79.6	64.4	27876	86205	1194	1866
ByteTrack [64]	77.3	80.3	63.1	25491	83721	2196	2277
MotionTrack [40]	80.1	81.1	65.1	23802	81660	1140	1605
UTM [11]	78.7	81.8	64.0	25077	76298	1431	1889
BoT-SORT [1]	80.2	80.5	65.0	22521	86037	1212	1803
ADPTrack	80.9	80.7	65.4	22287	85446	1086	1770

A.2 Ablation Studies

This subsection describes several ablation studies performed for all the components we introduced in the ADPTrack framework. We list the experimental studies for several components and compare their results to the chosen setup for our algorithm presented in Section 4. In the first experiment, we use a simplified base heuristic with only the motion model and no appearance model. In this experiment, we test the importance of the appearance model in our framework. In the second experiment, we consider the standalone base heuristic and compare the visual information of the target frame objects with all the matched objects of the given tracks without our near-online simulation approach. This experiment tests the importance of using tentative tracks in our framework. In the third experiment, we use ADPTrack with a base heuristic that has both the motion and appearance models, but we use a modified similarity score [cf. (7)]. In this experiment, we test the importance of using visual information of previously matched objects directly instead of using overall averaged visual information for a given track i while comparing it with a tentative track j . In the fourth experiment, we present additional implementation details related to the main algorithm 1. Every experiment includes a parameter sweep.

A.2.1 Experiment 1

The first experiment tests as the base heuristic the BoT-SORT method without its appearance model. Since it relies solely on the motion model, we refer to it as the BoT-SORT-motion. In particular, we apply the concept of intersection-over-union overlap (IOU overlap for short) used by BoT-SORT to compute similarity scores. While performing assignments between the given tracks and the objects of the target frame, BoT-SORT uses a motion model to predict the position and size of the object of a given track in the next frame. It then computes an IOU overlap between the predicted position and

size of the object and the objects of the target frame based on the position and size of the objects. The IOU overlap scores are used to perform the assignment between the given tracks and the target frame.

In this ablation study, each of the tentative track’s objects is sequentially compared to the given track’s object in the subsequent frames using IOU overlap to obtain a sequence of IOU overlap scores. We calculate $z_{k+1}^{ij}(x_k)$ by taking an average of ℓ IOU overlap scores associated with the ℓ objects in tentative track j obtained previously. Note that we maintain m copies of a given track’s motion model to generate m different $z_{k+1}^{ij}(x_k)$ similarity scores for a single track i .

More specifically, we use a motion model, which predicts the position and size of the object based on historical information. Given the position and size information $(p_0^i, p_1^i, \dots, p_k^i)$ associated with the i th track, we predict the position and size information \tilde{p}_{k+1}^i ,

$$\tilde{p}_{b+1}^i = \text{MOTION}(p_0^i, p_1^i, \dots, p_b^i), \quad b = 1, \dots, k. \quad (9)$$

In our case, MOTION represents the calculation related to the Kalman filter. Therefore, the information \tilde{p}_{k+1}^i can be used to represent the size and position of the object at $k + 1$ the frame based on the information associated with track i . To compute the similarity score of given track i and tentative track starting from j without appearance information, we then compute a sequence of information $\tilde{p}_k^{ij}, \tilde{p}_{k+1}^{ij}, \dots, \tilde{p}_{k+\ell}^{ij}$ via

$$\tilde{p}_{b+1}^{ij} = \text{MOTION}(p_0^i, p_1^i, \dots, p_k^i, \tilde{p}_{k+1}^j, \dots, \tilde{p}_b^j), \quad b = k + 1, \dots, k + \ell + 1, \quad (10)$$

In addition, we define $\tilde{p}_{k+1}^{ij} = \tilde{p}_{k+1}^i$. Based on the obtained sequence $\tilde{p}_{k+1}^{ij}, \dots, \tilde{p}_{k+\ell}^{ij}, \tilde{p}_{k+\ell+1}^{ij}$, we compute the similarity score for this ablation study and refer to it as F_1 . It is given by

$$F_1(T^i(x_k), T^j(\bar{x}_\ell^{k+1})) = \frac{1}{\ell} \sum_{b=k+1}^{k+\ell+1} \text{IOU}(\tilde{p}_b^{ij}, \tilde{p}_b^j), \quad (11)$$

where $(\tilde{p}_{k+1}^j, \tilde{p}_{k+2}^j, \dots, \tilde{p}_{k+\ell+1}^j)$ represents all the position information associated with the tentative track starting from the object j . The function IOU takes two position and size information $\tilde{p}_b^{ij}, \tilde{p}_b^j$ and returns a big value if ‘the overlap’ of objects represented by $\tilde{p}_b^{ij}, \tilde{p}_b^j$ is large.

As the base heuristic for this experiment is BoT-SORT-motion, we apply BoT-SORT-motion as a baseline for comparison. Table 7 and 8 show the video-wise results of BoT-SORT-motion and ADPTrack with BoT-SORT-motion as the base heuristic for all the videos of the validation dataset. The overall scores for BoT-SORT-motion and ADPTrack with BoT-SORT-motion as the base heuristic 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%. However, we see a reduction in IDF1 accuracies for MOT17-09, MOT17-10, and MOT17-13 which may be because of a moving camera. We believe that this kind of tracker is suitable for videos in which the motion of the object can be more helpful, such as MOT17-04 with an overhead static camera. 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 believe 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-motion 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(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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-motion, and ADPTrack with BoT-SORT-motion as the base heuristic as per experiment 1 over the validation dataset.

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

A.2.2 Parameter Study for Experiment 1

The number of subsequent frames ℓ for near-online simulation and the tuning parameter α of (6) are varied in two ablation studies and the experimentation results are presented in Tables 10 and 11, respectively. We observe that the performance of ADPTrack increases as we reach a certain number of subsequent frames, and then decreases as we go further.

Table 10: An ablation study over the number of subsequent frames; ADPTrack with BoT-SORT-motion as the base heuristic as per experiment 1 over the validation dataset; ℓ : number of subsequent frames for near-online simulation.

ℓ	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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 tuning parameter α ; ADPTrack with BoT-SORT-motion as a base-heuristic as per experiment 1 over the validation dataset; ℓ : number of subsequent frames for near-online simulation.

ℓ	α	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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

A.2.3 Experiment 2

In experiment 2, we neither perform the near-online simulation nor generate the tentative tracks. We consider BoT-SORT with both the motion and appearance models and exploit the visual information of previously matched objects of given tracks. For a given track i , we maintain an appearance quality vector where each value indicates the visual quality of a previously matched object. We generate the appearance quality vector by extending it every time an object is assigned to the given track i . Based on the appearance quality vector and a particular threshold referred to as the quality threshold β , we iterate backward starting at frame k and find the object at frame q with the appearance quality value greater than the quality threshold β . Then we select the visual information of objects from frame $q - s + 1$ to frame q assigned to track i , which is denoted by $\bar{V}^i(x_k)$, i.e., $\bar{V}^i(x_k) = (v_{q-s+1}^i, v_{q-s+2}^i, \dots, v_q^i)$. The high-quality visual information $\bar{V}^i(x_k)$ is then used to compute the similarity score.

To compute $z_{k+1}^{ij}(x_k)$, we compare the visual information v_a^i contained in $\bar{V}^i(x_k)$ with \bar{v}_{k+1}^j to generate a sequence of visual similarity scores between given track i and object j . We then calculate the average of those visual similarity scores. In particular, we use a modified similarity score F for this ablation study and refer to it as F_2 . It is given by

$$F_2(T^i(x_k), T^j(\bar{x}_\ell^{k+1})) = \frac{1}{s} \sum_{v_a^i \in \bar{V}^i(x_k)} \text{CS}(v_a^i, \bar{v}_{k+1}^j). \quad (12)$$

The number s of previously matched objects from consecutive frames of given tracks to be used for comparison is flexible and the ablation study for this parameter is shown in Table 12. We set the value of the quality threshold β as 0.15 based on empirical results. The video-wise results for BoT-SORT and the modified BoT-SORT as per experiment 2 are presented in Tables 4 and 13, respectively. The overall scores for BoT-SORT and experiment 2 are listed in Table 17.

Table 12: An ablation study over the number of previously matched objects ($\#s$) as per experiment 2.

$\#s$	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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

Table 13: Video-wise scores for modified BoT-SORT (experiment 2) over the validation dataset.

Video Name	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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

A.2.4 Experiment 3

In this experiment, we use BoT-SORT with both the motion and appearance models as the base heuristic. BoT-SORT maintains representative visual information associated with every given track. For a given track i , the representative visual information is updated whenever an object from a target frame is assigned to the given track i . It is updated by averaging the assigned object’s visual information with the given track’s representative visual information through an exponential moving average. While performing assignments between the given tracks and the objects of the target frame with respect to the appearance model, BoT-SORT compares the representative visual information of a given track i to the visual information of every object of the target frame.

In this ablation study, to calculate a similarity score between a given track i and a tentative track j , we compare the representative visual information of the given track i and the visual information of objects of tentative track j . As there are ℓ objects in tentative track j , this comparison generates a sequence of ℓ visual similarity scores. We also calculate IOU overlap scores as explained in experiment 1. Similar to BoT-SORT, we consider a minimum of the visual similarity score and IOU overlap score to obtain a similarity score for every object in the tentative track and generate a sequence of similarity scores corresponding to all the objects of the tentative track j . Note that we do not update the representative information of given track i with any of the objects of tentative track j , unlike experiment 1.

To compute $z_{k+1}^{ij}(x_k)$, we calculate the average of all the previously calculated similarity scores. We present a modified F for this ablation study and refer to it as F_3 . It is given by

$$F_3(T^i(x_k), T^j(\bar{x}_\ell^{k+1})) = \frac{1}{\ell} \sum_{b=k+1}^{k+\ell+1} \min \{ \text{IOU}(\tilde{p}_b^{ij}, \bar{p}_b^j), \text{CS}(\tilde{v}_k^i, \bar{v}_b^j) \}, \quad (13)$$

where $\tilde{p}_{k+1}^{ij}, \dots, \tilde{p}_{k+\ell}^{ij}, \tilde{p}_{k+\ell+1}^{ij}$ are computed via (9) and (10), as in experiment 1. The information \tilde{v}_k^i represents the representative visual information of a given track i , which can be viewed as a moving average of the sequence $v_0^i, v_1^i, \dots, v_k^i$. This is different from experiment 2 and experiment

4 where we compare individual visual information of previously matched objects of a given track i . Therefore, given a given track i and tentative track j , this experiment tests how important it is to compare the visual information of previously matched objects of given track i and objects of tentative track j directly.

The video-wise results for BoT-SORT and ADPTrack with BoT-SORT as the base heuristic are presented in Tables 4 and 14, 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 14: Video-wise scores of ADPTrack with BoT-SORT as a base-heuristic as per experiment 3 over the validation dataset.

Video Name	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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

A.2.5 Parameter Study for Experiment 3

The number of frames ℓ for near-online simulation and the tuning parameter α used in (6) are varied in two ablation studies and the experimentation results are presented in Tables 15 and 16, respectively. We observe that the performance of the ADPTrack increases as we increase both the number of frames and then plateaus as we go further.

Table 15: An ablation study over the number of subsequent frames ℓ ; ADPTrack with BoT-SORT as a base-heuristic as per experiment 3 over the validation dataset;

ℓ	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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

Table 16: An ablation study over the tuning parameter α . ADPTrack with BoT-SORT as a base heuristic as per experiment 3 over the validation dataset; ℓ : number of subsequent frames for near-online simulation.

ℓ	α	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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

A.2.6 Experiment 4

We refer to the combination of experiments 1,2 and 3 presented in Algorithm 1 as Experiment 4. We present the overall scores of all the experiments in Table 17.

We observe that objects that are either mostly occluded or distant from the given track may not need to be considered in the evaluation of F defined in (8). To address this, we introduce a heuristic that will discard certain objects from consideration for each tentative track j with respect to a given track i . We shortlist objects from the tentative track $T^j(\bar{x}_\ell^{k+1})$ and refer to them as candidates. These candidate objects will be used to calculate z_{k+1}^{ij} in (8). An object in a tentative track becomes a candidate for computations in (8) if the following holds: (i) The IOU overlap score of the object with the given track’s estimated next state is sufficiently large, and (ii) The visual similarity between the object and the given track’s representative visual information is sufficiently high. The number of candidate objects may be different for each tentative track $T^j(\bar{x}_\ell^{k+1})$ and given track i , and this value should replace ℓ in the denominator in (8) to produce a legitimate average value.

We observed that our algorithm may experience difficulty in the presence of objects that are continuously occluded, as their visual information does not capture consistent visual information of a single object, rather they may be a noisy mix of several intercepting objects. Therefore we propose a crowd-detection heuristic where we check if the objects 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 motion model alone may itself be a better estimator of the association scores.

This framework renders the computation of similarity scores between the given tracks and tentative tracks suitable for parallelization. We have adopted multi-processing where a particular process computes the similarity score between a given track and a tentative track. This parallelization can be further extended to object-to-object similarity score computations.

Table 17: Comparison of proposed algorithms 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
Experiment 2	84.436	80.891	71.219	9378	21144	372	660
Experiment 3	84.444	81.043	71.403	9183	21084	381	681
Experiment 4	85.355	81.011	71.749	9252	21090	357	657

A.2.7 Parameter Study for Experiment 4

In this section, we present ablation studies for various parameters present in Experiment 4 (Algorithm 1). Table 18 presents an ablation study performed over the number of subsequent frames considered. The ablation study over the tuning parameter (α) parameter is mentioned in Tables 19 and 20. For generalization and application to the test dataset, we select the tuning parameter as 0.25. We consider 15 and 5 for the number of subsequent frames (ℓ) and the number of previously matched objects (s) from given tracks, respectively.

Table 18: An ablation study over the number of subsequent frames ℓ as per experiment 4.

ℓ	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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: An ablation study over the tuning parameter (α) as per experiment 4. ℓ : number of subsequent frames for near-online simulation. The value of ℓ varies between 6 and 10.

ℓ	α	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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: Experiment 4: An ablation study over the tuning parameter (α) as per experiment 4. ℓ : number of subsequent frames for near-online simulation. The value of ℓ varies between 11 and 15.

ℓ	α	IDF1(\uparrow)	MOTA(\uparrow)	HOTA(\uparrow)	FP(\downarrow)	FN(\downarrow)	IDSW(\downarrow)	Frag(\downarrow)
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