Journal of Artificial Intelligence and Autonomous Intelligence

Received 15/08/2025; Accepted 26/10/2025; Published 02/11/2025

A Comparison of Multi-Object Tracking Methods in Image Sequences

Maria Cãzilã mariacazila4@gmail.com
PiNTeam Motor Vehicle Manufacturing,
Sibiu, Romania.

Remus Brad remus.brad@ulbsibiu.ro
Universitatea Lucian Blaga din Sibiu, Computer Science and Electrical Engineering Department,
550024 Sibiu, Romania.

Raluca Brad raluca.brad@ulbsibiu.ro
Universitatea Lucian Blaga din Sibiu, Computer Science and Electrical Engineering Department,
550024 Sibiu, Romania.

Corresponding Author: Remus Brad

Abstract

Multi-Object Tracking is one of the main tasks in computer vision. It deals with the real-time detection and tracking of several objects across video frames. This paper discusses and compares three MOT algorithms: SORT, DeepSORT, and JDE on pedestrian tracking in urban scenes. A brief discussion on some important theoretical aspects such as online vs offline tracking, the use of Kalman filters, data association methods, and the use of appearance features for identity continuity is presented. The three tracking algorithms are practically evaluated based on theoretical knowledge on the MOT16 and MOT17 benchmark datasets. The algorithms were tested under similar conditions using standard MOT metrics MOTA, MOTP, IDF1 and number of identity switches along with visual inspection. Results state that SORT is fast and simple but does not maintain consistent identities most of the time; DeepSORT does better by adding appearance features; JDE does even better by combining detection and feature embedding into one model at a cost of increased computational complexity. Implementation issues are also discussed, and future work will include testing newer models for better runtime efficiency and adaptability to real-world tracking scenarios.

Keywords: Multi-Object Tracking, SORT, Pedestrian Tracking, Appearance Features, Computer Vision.

1. INTRODUCTION

Multi-Object Tracking (MOT) is a task in computer vision that aims to find and follow multiple objects as they change over time in a sequence of frames. Given a stream of video or image sequences, the goal of MOT is to estimate the trajectories of all objects by associating their detections over time. MOT has several important real-world applications like traffic monitoring, intelligent

Citation: Maria Cãzilã, Remus Brad and Raluca Brad. A Comparison of Multi-Object Tracking Methods in Image Sequences. Journal of Artificial Intelligence and Autonomous Intelligence ;Review 2025;2(3).23.

video surveillance, autonomous driving, human-computer interaction, and sports analytics. The challenge with MOT compared to single-image object detection is that it needs to reason not just about space but also about time; this temporal dimension adds more complexity such as dealing with occlusions between objects, entry/exit from the field of view and sudden changes in motion or appearance [1].

At its most basic level, MOT can be thought of as a joint optimization problem where the system decides which object detection across consecutive frames belongs to the same physical object. The Tracking-by-Detection paradigm has been dominant in this area; here, an object detector is first run on every individual frame in the sequence followed by some data association algorithm linking these detections together into continuous trajectory [2].

Traditional approaches for MOT predominantly employed probabilistic models and hand-crafted features in the tasks of motion prediction and object association. However, with the introduction of deep-learning approaches, it has shifted the paradigm for tracking and dramatically increased performance. Convolutional Neural Networks are frequently used to extract appearance features, while more recent architectures based on transformers and graph neural networks provide sophisticated methods to encode spatial and temporal relations. These advances boost not only accuracy, but also generalization performance in complex and crowded scenes. Nevertheless, despite the advancement of deep learning, MOT is still a very difficult task that requires dealing with heterogeneous object appearances, complex motion, occlusion, varying numbers of targets, and incomplete detections. Assigning identities consistently over time is a key challenge, particularly when dealing with occlusion, and for visually similar targets, highlighting the need for intelligent spatial, temporal and appearance feature integration [3, 4].

MOT methods are typically organized along different axes, such as, for instance:

- Detection-based versus detection-free tracking; online MOT versus offline MOT; 2D MOT versus 3D MOT. Detection-based tracking often uses object detectors to initialize tracks automatically, while detection-free tracking usually requires manual initialization and has a fixed number of objects throughout the sequence [2].
- Online methods perform data association using only past and current frames, which is suitable for real-time applications. Offline methods take advantage of future information and generally achieve higher accuracy at the expense of increased latency [2, 3].
- 2D vs. 3D MOT: Traditional 2D MOT systems run on image plane coordinates, whereas 3D MOT employs measurements from depth sensors or LiDAR to perform tracking in world coordinates. This allows enhanced spatial reasoning for deployments in the real world [3].

The selected algorithms reflect the most significant developments in the area: SORT and DeepSORT provide solid baselines, with DeepSORT incorporating appearance features to enhance identity continuity, FairMOT and JDE merge detection with Re-ID into one network, ByteTrack enhances tracking by taking advantage of low-confidence detections, Tracktor++ makes use of a detector to simplify tracking and MOTR presents an end-to-end transformer-based approach.

Performance in MOT is typically evaluated using standard datasets such as MOTChallenge (MOT16, MOT17, MOT20) with metrics including Multiple Object Tracking Accuracy (MOTA), ID F1 score (IDF1), and the number of ID switches and many more [3, 4].

This paper reviews the comparative performance of different state-of-the-art MOT algorithms on image sequences. The goal is to evaluate their performance in a variety of settings and understand their relative strengths and weaknesses across quantitative metrics and qualitative observations. In subsequent chapters, we discuss first real-world applications of MOT, second an overview of typical pitfalls encountered when running tracking tasks. Next, we present a fundamental basis for MOT, which consists of detection models, appearance modeling, and performing data association. The procedure for evaluating datasets and performance metric is discussed in depth. We then present experimental results in comparison for several selected algorithms, and we offer some research implicit to the results we present.

2. APPLICATION OF MOT IN REAL WORLD SCENARIOS

Multiple Object Tracking (MOT), also referred to as Multi-Target Tracking, is the leading task of computer vision that explores video streams for the purpose of discovering and tracking multiple objects from different categories, including pedestrians, cars, animals, and objects, with no previous knowledge of their appearance or actual number. The objectives of MOT are to discover objects in video frames, associate these objects over time, and estimate their trajectories in the environment. Dramatic improvements in accuracy and practical applicability to real-world scenarios of MOT, based on advances in deep learning, have occurred in the past decade [2, 3].

MOT has a variety of applications, including but not limited to, intelligent traffic management and surveillance in smart cities, autonomous intelligent driving, analysis, and tracking of sporting events and performance, medical imaging and medical analysis, and in retail analytics. A unique set of challenges and requirements exists for suitable and tailored solutions for each of these applications which exemplify the flexibility of MOT to improve efficiency and safety in many environments. The subsequent sections will elaborate on each of these domains and consider how MOT aids efficiency and decision-making in each scenario.

2.1 Autonomous Driving

MOT is essential to road safety by detecting, identifying, and tracking pedestrians, vehicles, and objects, facilitating the avoidance of collisions, navigation, and planning of infrastructure. The accurate tracking of pedestrian and vehicle movements is paramount to road safety; this tracked information is the substrate of navigation systems and assists in developing functions such as collision avoidance and lane-keeping assistance. MOT can enhance compliance with traffic rules by providing camera-based systems with the ability to reference identified vehicles and detect traffic rule violations including speeding, drifting out of lane, and illegal parking. Vehicle re-identification techniques are already employed in toll enforcement and fleet monitoring [3].

2.2 Surveillance and Security

MOT is used in surveillance to track people and objects across camera networks while monitoring the environment in real-time. This can be critical for crowded or high-security public environments (e.g., airports, train stations, public events, etc.) that require situational awareness and surveillance. MOT combined with person re-identification can help surveillance systems achieve persistent tracking of people who cross the views of different cameras or are temporarily occluded [1].

2.3 Sports Analytics

MOT is also widely used in sports for real-time tracking of players, balls, and other equipment. This tracking enables analysts and coaches an opportunity to capture movement patterns, assess performance, and extract strategic information. It is also essential for goal and wicket evaluation, where it operates by following the trajectory of the ball in conjunction with the critical moments of player positioning [12, 21].

2.4 Robotics and Human-Computer Interaction (HCI)

MOT is important in robotics and HCI by following object and human trajectories to facilitate navigation and interaction in dynamic settings. Through its use, robots can learn human gestures, prevent obstacles, and adapt to their environment, providing functionality and better user experience. Using thermal or IR camera systems or deep learning models increases accuracy and reliability in difficult conditions and situations [3].

2.5 Medical Imaging and Biomechanics

MOT has become an important aspect in a range of real-world applications. It is especially used in areas that require the monitoring of dynamic, yet complicated systems. In biomedical imaging, MOT allows us to accurately track multiple particles, be it molecules, drug carriers, or cancerous cells, allowing scientists to begin to study disease progression, drug delivery mechanisms, and cellular interactions at an advanced level. High accuracy and robustness are a necessity for these applications, as there is considerable noise in data or high-density environments. The growth of deep learning has been beneficial in this regard, as it provides improved tracking performance through increased accuracy, scaling, and handling occlusion difficulties [3, 13].

The various applications demonstrate the significance of MOT across a wide array of real-world problems. The advancements in technology, especially deep learning, represent step changes in addressing major real-world issues such as occlusion, large data volumes, and efficiency. This increased advancement is helping to broaden the applicability of MOT systems as well as making them more efficient.

3. CHALLENGES IN MULTI OBJECT TRACKING

Despite the progress made in MOT, through the application of deep learning for detection and associate, there are still numerous challenges that limit tracking accuracy. These challenges stem from various environmental, data quality, and algorithmic factors, and all are actively researched topics.

3.1 Occlusion and Object Overlap

Occlusion is a huge challenge in MOT. Objects often traverse behind each other or beyond visibility range causing missed detection and track breakages. This frequently leads to identity swaps or disjointed tracks particularly in dense scenarios such as pedestrian walkways or vehicle intersections. Occlusion missed detections will lead to broken tracks and overall tracking performance decrease, therefore occlusion handling has become a significant focus in MOT [3, 4].

3.2 Appearance Similarity Among Objects

In situations where there are multiple similarly looking objects (such as people in uniform, or identical vehicles), data association may be untrustworthy. Low resolution images, and absence of unique information (such as license plates or face) make this challenging even if the objects are at some distance. Deep appearance embeddings help, but are in general insufficient to separate objects that have the same visual appearance. Therefore, the trackers may mistake two identities and switch them frequently very often [3, 4].

3.3 Unexpected Motion and Unpredictable Trajectories

Simple motion models, such as constant velocity and Kalman filters, assume smooth and continuous object dynamics. In reality, objects can stop, turn quickly, and accelerate unpredictably, which breaks these assumptions and leads to poor predictions and broken tracks. To handle nonlinear motion, more flexibility is needed, learned motion models [3].

3.4 Variable Object Counts

The number of visible objects in a scene can change quickly as new ones come in or existing ones go out of the frame. So, MOT algorithms must be able to accurately start new tracks when objects show up and safely end them when objects are no longer seen. Errors made in dealing with these changes can produce false positives, whether because an object has been kept without it existing, or because an object is duplicate, or can produce false negatives because an object tracked has not been tracked at all. Either way, it has an impact on the consistency and completeness of the end trajectories. [4].

3.5 Scale Variation and Illumination Changes

Tracking performance can be impacted by changes in both object scale and lighting conditions. Objects can appear small due to distance or camera zoom and can introduce issues due to visual detail missing for detection and re-identification purposes. Similarly, whether shadows or reflections or in cases of low-light details can change appearances across frames and impact detection or trigger of identity switches. These conditions directly affect appearance-based tracking models, which rely on appearance properties being unchanged. To better the robustness of MOT approaches, the use of scale-invariant representations and the ability to be resilient to lighting changes using motion and/or temporal cues would need to be better integrated into the designs [1, 3, 4].

3.6 Camera Motion and Scene Dynamics

When it comes to moving cameras, as in the case of UAV/Drone footage, autonomous driving (e.g. self-driving cars, etc.), or handheld video, MOT systems face additional challenges of separating object motion from ego-motion. Changes to the background due to motion, as well as changes to the camera or observer-egomotion (being the user as individual moving camera) creates distortion to the spatial consistency across frames [5].

These differences affect appearance, and motion readings, increasing the chances of tracking errors, particularly for small or fast-moving objects. Established trackers, which were developed for static scenes, are typically cumbersome to use for motion-based scenes. As a result, newer methods have integrated motion compensation or camera estimation techniques to ensure tracker consistency in dynamic scenes.

3.7 Spatial Information Loss

Most visual tracking systems, especially when using standard CCTV video footage, function in 2D video, employing a monocular camera. When a complicated 3D scene is recorded, a monocular camera compresses the scene into a two-dimensional representation, will also suffer a loss of depth-related information. Converting a scene from the real world to a numerical digital format always involves some level of abstraction and may introduce some imprecision creating the actual scene surface-recorded data. This loss of spatial context complicates an algorithm's ability to detect and track an object(s) consistently, especially when occlusion occurs or due to complex scene geometry in the physical world [3].

4. AN OVERVIEW OF MOT ALGORITHMS

As mentioned in the previous chapter, Multi-Object Tracking is the process used for identifying and following the trajectories of multiple objects within a video sequence by detecting them in each frame and maintaining their identities across time. Most MOT algorithms adopt a multi-step processing architecture, where each stage handles a distinct aspect of the tracking process, described in FIGURE 1.

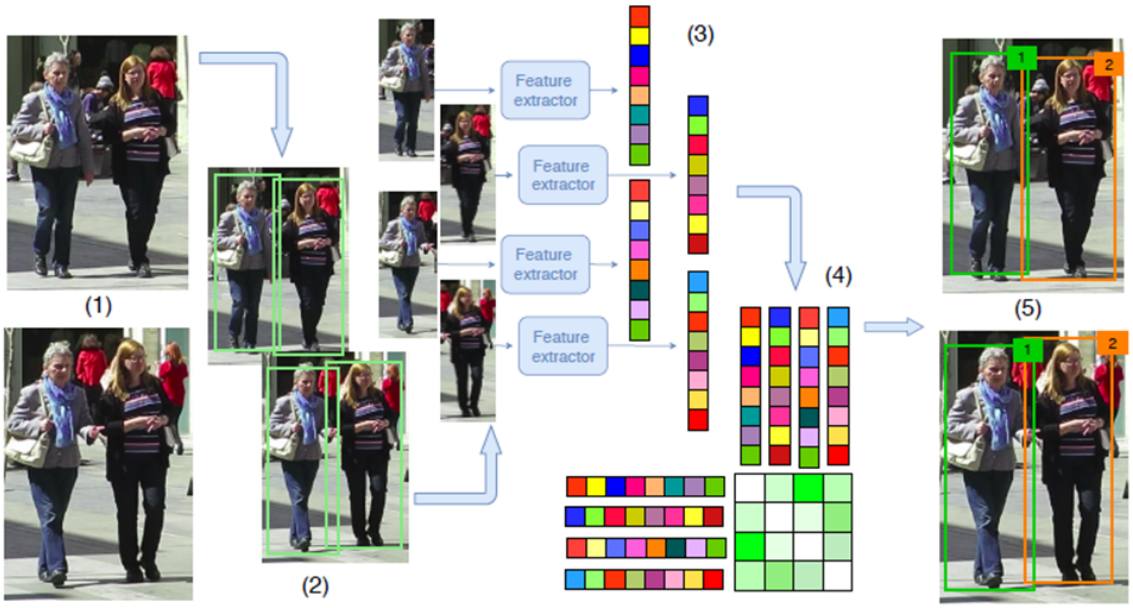

Figure 1: MOT algorithm workflow [1]

1. Detection Stage: Each video frame is processed by an object detector to locate all instances of the target class (e.g., people, vehicles) using bounding boxes. These detections are the starting point for tracking. (e.g. common trackers are YOLO, CenterNet, and Faster R-CNN) [1]
2. Future extraction and motion prediction: To track objects consistently, the system extracts feature like appearance (e.g., deep embeddings), motion (e.g., velocity), and object interactions. A motion model, such as a Kalman filter, may also predict where each object is likely to appear in the next frame [1]
3. Similarity Computation Stage: This step calculates how likely each new detection matches existing tracks by comparing their features and predicted positions. The results form an affinity matrix used to measure similarity between objects across frames
4. Association Stage: The system utilizes the affinity scores as a basis for linking detections to ongoing tracks, and assigns them consistent IDs. The Hungarian algorithm or learned matching networks are among the techniques used to create these associations, even in difficult or populous locations

Moreover, MOT algorithms are divided into online and offline (batch) methods, according to their video frame processing capability.

- Online tracking: operates in real time, relying solely on the current and past frames for updating the object trajectories. Nevertheless, since they lack future information, they usually find it hard to rectify the earlier errors and consequently might be less accurate in the overall results. Nevertheless, these methods are a necessity for time-critical applications such as autonomous driving and video surveillance. (e.g., SORT, DeepSORT, and FairMOT that link fresh discoveries to existing tracks on a frame-by-frame basis) [1, 4].

- Offline (batch) tracking: the entire video sequence is processed simultaneously, utilizing both past and future frames. These algorithms can execute global optimization, and the accuracy of their object identity refinement is comparatively higher. Hence, they are employed in areas that do not demand instant results, like video evidence analysis or offline scene comprehension. The reason batch trackers produce more dependable and uniform trajectories is that they are able to rectify the early mistakes through future context.

4.1 Tracking Paradigms

Different methods have been developed to solve the task of tracking multiple objects in videos. These methods, known as tracking paradigms, define how detection and tracking steps are organized and connected. In this chapter, to begin with, we present and examine the three primary MOT paradigms which are Detection-Based Tracking (DBT), Detection-Free Tracking (DFT), and Joint Detection and Tracking (JDT). The handling of object detection and identity matching varies with each approach. The comprehension of these paradigms plays a vital role in the selection of the suitable method for tracking troubles and real-world applications.

4.1.1 Detection based tracking (Dbt)

Detection-Based Tracking (DBT), also referred to as Tracking-by-Detection (TBD), is one of the most widely used techniques in Multi-Object Tracking. In FIGURE 2.2, we can see that the objects are detected in every frame with the help of a pre-trained object detector, and subsequently, these detections are connected through time to create object paths.

The procedure is carried out in two major steps:

- Object detector: It is trained in specific object types such as people, cars, or faces. It will detect and locate objects in every frame with the assistance of bounding boxes or segmentation masks.
- Tracker: It employs the position and appearance features of these detections to link them through the frames and create continuous tracks [5, 7].

The distinction between detection and tracking contributes to the simplification of the overall task, making it more manageable to develop and improve each part separately. DBT methods can be executed in online or offline modes. This is based on the usage of past information only or of the entire video sequence by the tracking algorithm. One of the main limitations of DBT is that it is very much dependent on the reliability of the object detector: if the objects are missed or wrongly detected, the tracker might either lose the object or assign a wrong identity [2].

4.1.2 Detection free tracking (DFT)

Detection-Free Tracking (DFT) is a term used to describe the methods where objects are tracked without the help of an object detector. Instead, the tracker is initialized manually or semi-automatically

in the first frame, and the object's position is updated across frames using motion information such as optical flow or Kalman filtering. Since DFT does not detect new objects entering the scene, it is better suited for controlled environments with limited targets. However, it often struggles in crowded scenes or under occlusions [2].

4.1.3 Joint detection and tracking(JDT)

Recent MOT research has shifted toward models that combine detection and tracking in a single, unified system. Joint Detection and Tracking (JDT) or Joint Detection and Embedding (JDE) approaches use deep learning to perform both tasks simultaneously often through a single neural network trained end-to-end [10]. This means the system does not need a separate, pre-trained object detector. The JDE model learns to find objects in the frame and track them across time as part of the same training process [4].

Common examples include FairMOT and JDE, which use a shared feature backbone to produce bounding boxes and identity embeddings in a single stage at inference time [6]. This strategy also has some downsides when objects frequently appear and disappear from the frame or when some objects are not present in all frames. Since detection and tracking are learned jointly, failure in one can propagate to the other, making it more challenging to keep consistent object identities through occlusions or missed detections.

The three MOT paradigms each have their own strengths and weaknesses. Since each of these approaches has its own advantages and is better suited for specific tasks, there is no one-size-fits-all solution. A comparison between all the paradigms presented earlier is shown in TABLE 1.

Table 1: Summary of tracking paradigms

ccc

	DBT	**DFT**	**JDE**
Detection	Pre-trained detector	No detector (manual initialization)	Detection and tracking in one network
Object classes	Predefined	Tracks initial object only	Learning to track without class labels
New Object	Can detect new objects	Cannot detect new entries	Can track and detect new instances
Example	SORT, DeepSORT	Optical Flow, Kalman Filter	JDE, FairMOT, MOTR

4.2 Referenced Tracking Algorithms

In this section a set of important tracking algorithms are introduced, and the scope is to show how Multi-Object Tracking (MOT) has developed over time, from traditional motion-based methods to newer deep learning models. The algorithms are grouped based on how they handle detection, how they are built, and whether they use appearance features.

4.2.1 Classical trackers

SORT (Simple Online and Realtime Tracking), proposed by Bewley [7], is famous for its simplicity and fast performance. Predictions are matched to fresh detections in each frame with the Hungarian algorithm, and a Kalman filter is used for motion prediction. The ability to operate on hundreds of frames per second in real time is also one of its greatest strengths and is well suited to applications where speed is the critical requirement. As SORT is based on object positions rather than appearance, it can be applied with any object detector in a straightforward manner. But it does have one downside: when objects overlap or appear similar, they cannot recognize or find them solely by their appearance.

DeepSORT extends SORT with additional identities for better tracking by using appearances, as in Wojke [5]. This involves instance-wise feature extraction (Re-ID embeddings) from each detected object based on a deep neural network. These features allow the tracker to distinguish between things, even if they are close, occluded, or go off screen and come back. This reduces the number of identity switches, as a result making DeepSORT more robust when it comes to crowded or cluttered scenes. Although more powerful compared to SORT, it is not computationally expensive and can work well for online systems. This trade-off between speed and accuracy has made DeepSORT attractive in several state-of-the-art tracking applications.

4.2.2 Deep learning trackers

FairMOT is a representative real-time MOT framework with the joint detection and embedding (JDE) paradigm, which conducts object detection and appearance embedding within the same neural network simultaneously. In contrast to the standard tracking-by-detection methods, where detection and re-identification (ReID) are conducted in two separate steps, FairMOT incorporates these two tasks in one single pipeline with CenterNet as its backbone. Developed by Zhang [6], presents a two-branch architecture, which consists of an object detection branch and an appearance feature branch. These branches are jointly learned by sharing their convolutional layers.

ByteTrack, proposed by Zhang [8], enhances MOT by employing a simple but effective principle: keep low-confidence detections during data association. Standard trackers ignore such detections, while ByteTrack associates high-confidence detections with existing tracks and utilizes low-confidence ones for the recovery of missing tracks. This strategy decreases missing detections and identity fragmentation, particularly in scenes with dense populations. It is one of the methods that does not use appearance features or a Re-ID module, but still achieves competitive performance on benchmarks, such as MOTA.

Tracktor++, proposed by Bergmann [9], is a tracking-by-detection approach that reuses the regression head of a Faster R-CNN detector to predict object positions across frames, effectively turning the detector into a tracker. Unlike traditional methods that rely on explicit motion models or data association, Tracktor++ regresses previous bounding boxes to new positions and then applies non-maximum suppression to maintain track consistency. To improve tracking under occlusion, it integrates a Re-Identification (ReID) module and a simple Kalman filter for handling track reactivation.

JDE (Joint Detection and Embedding), presented by Wang et al. [10], integrates object detection and Re-ID embedding into a single, real-time deep learning model. Unlike traditional pipelines that use separate models for detection and identity features, JDE trains both jointly use a shared backbone (e.g. YOLOv3). This design allows the tracker to run faster while maintaining competitive accuracy in both detection and identity tracking.

MOTR (Multiple Object Tracking with Transformers), which was recently proposed in Zeng et al. [11], is a contemporary tracker that enables simultaneous detection and tracking within a single model because of the changes caused by transformers. MOTR introduces the idea of track queries, which enables the model to recall an object's identity across multiple frames because of attention. MOTR is simply an extension of DETR, a transformer-style object detection algorithm, to work with video sequences rather than individual images. Therefore, MOTR is able to tract objects through time, without the typical associated step, like a Kalman filter or a Hungarian algorithm.

Table 2: Summary of most common algorithms

Tracker	Year	Detection	Re-ID	Joint detection +Tracking	Architecture
SORT [7]	2016	Public	No	No	Kalman Filter + Hungarian Algorithm
DeepSORT [5]	2017	Public	Yes	No	CNN Re-ID + Kalman Filter
FairMOT [6]	2020	Private	Yes	Yes	CenterNet +Re-ID
ByteTrack [8]	2021	Both	No	No	IoU based two stage matching with low confidence boxes
Tracktor ++ [9]	2019	Public	Optional	No	Fast R-CNN regression-based tracking
JDE [10]	2020	Private	Yes	Yes	Shared YOLOv3 +Re-ID
MOTR [11]	2022	Private	Yes	Yes	Transformers + track queries (DETR)

5. DATASETS FOR MOT ASSESSMENT

The MOTChallenge is the overall benchmark to test and compare multi-object tracking algorithms. It provides standard benchmarks or datasets, annotation formats, and evaluation format to enable fair and consistent comparisons among tracking algorithms. The official benchmark consists of, and are described in TABLE 3 to ease description, several datasets: MOT16, MOT17 [30], MOT20 [29] which all contain real-life video sequences with annotated pedestrian tracks in challenging environments such as crowded streets, moving cameras, and occlusions.

Table 3: MOTChallenge Dataset description [31]

Dataset	Total Videos (train, test)	Detectors	Total Frames	Total Tracks	Bounding Boxes	Density of People
MOT-16	14	DPM	11235	1276	292733	moderate
MOT-17	14	DPM, FRCNN, SDP	17757	2355	564228	moderate to high
MOT-20	8	Faster R-CNN	13410	3833	2102385	extremely high

There are ground truth values available for the training set of MOT, and that is why they would be useful for comparing predicted results to correct object positions and identities so that we can precisely measure metrics for evaluation.

In FIGURE 2 we can visualize frames from the MOT16 training set, we visualize the ground truth bounding boxes. In FIGURE 2. left, the left side shows frame 90 without any boxes, while the right side displays the same frame with ground truth bounding boxes overlaid. FIGURE 2 right presents frame 131 from the same sequence. As observed, the man in the red jacket is consistently tracked across these frames, demonstrating correct identity preservation by the annotations.

Figure 2: MOT16 frame 90 and MOT16 frame 131 toghether with the MOT detection.

Public detections are provided so that different algorithms can be evaluated fairly using the same detection inputs, focusing only on tracking performance. The benchmark also supports private detections, allowing researchers to use their own detectors if desired, with separate leaderboards for both.

MOTChallenge includes baseline algorithms, code, and training methodologies as starting points for researchers. Participants can submit their results to the official website, where progress is tracked using leaderboards ranked primarily by MOTA (Multi-Object Tracking Accuracy), along with other metrics like **IDF1**, **MT** (Mostly Tracked), and **ML** (Mostly Lost) [3].

Each dataset is divided into training videos and testing videos. The training videos include the following key folders and files:

Img1: Contains all video frames as individual .jpg images, named using 6-digit frame numbers starting from 000001.jpg.

Gt/gt.txt: This file provides the ground truth annotations used for training and evaluation. Each line includes information such as <frame>, <id>, <bb_left>, <bb_top>, <bb_width>, <bb_height>, <conf>, <x>, <y>, <z>.

Det/det.txt: Public detection file provided for tracking-by-detection algorithms.

Seqinfo.ini: Configuration file that contains data about the sequence. [31]

The structure of the output file and gt.txt should be the same size. For the 2D tracking challenge, the last three values (*x, y, z*) can be set to -1 because they are ignored. For the 3D tracking challenge, the bounding box fields are ignored.

In ground truth files, the conf value acts as a flag: zero means the object is ignored, and any other value includes it in the evaluation. In contrast, in detection result files the conf value represents the confidence score of the detection and is used to indicate the reliability of the detected object.

6. EVALUATION METRICS

To evaluate how well a MOT system works, we use standard evaluation metrics. A good MOT system should detect all objects in each frame and keep the same ID for each object throughout the video. The metrics check if the tracker can follow each object accurately, even with occlusions, missing frames, or when objects enter or leave the scene.

CLEAR MOT metrics evaluate the accuracy of a multi-object tracking (MOT) system by comparing the predicted object positions to the actual ones. For each frame t, we assume there is a set of **ground truth objects** $(o_1, o_2, \ldots, o_n)$ and a set of tracker predictions, called **hypotheses** $(h_1, h_2, \ldots, h_m)$. The evaluation process begins by finding the best matches between each object o_i and prediction h_j, then measuring errors for incorrect matches, missed detections, and false positives [30].

To match an object o_i with a hypothesis h_j, a distance d_{ij}, is calculated and must be smaller than a predefined threshold T. The type of distance used depends on how the object is represented, for example, 2D or 3D Euclidean distance for points and Intersection over Union (IoU) for bounding boxes. For every frame in the video, the following steps are used to perform the mapping between ground truth objects and tracker predictions:

1. If object o_i and hypothesis h_j were matched in the previous frame, and their distance d_{ij} is less than the threshold T in the current frame, the same match is kept avoiding breaking the track [30].
2. Match all unmatched o_i to h_j, allowing only one-to-one pairs only when $d_{ij} <$ T. If a new match conflicts with a previous one, update it and count it as a mismatch. Where mm_t is the number of mismatches in frame *t*.
3. Now all matching pairs for the current time frame are identified. The number of matches at time *t* is c_t. For each match, calculate the distance d_t^i between object o_i and its hypothesis h_j in frame t.

4. Any unmatched h_j are treated as false positives, and any unmatched o_i are treated as misses. Let fp_t and m_t be the number of false positives and misses at time t, respectively. Also, the total number of objects present at time t is g_t.

Multiple Object Tracking Precision measures how accurately a tracking system predicts the positions of objects across video frames. It is calculated by averaging the position errors between the predicted and actual object locations in all frames.

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c^t} \tag{1}$$

where
d_t^i = distance between the predicted i object and its matched ground truth object at t
c^t = represents the number of correct matches between predicted and ground truth detections in frame t

A higher MOTP value indicates larger errors and therefore poorer tracking precision. This metric assesses only the accuracy of the localization of objects and does not assess the tracker's ability to maintain identities, follow paths of motion, or cluster objects.

Multiple Object Tracking Accuracy is another measure used to assess tracking performance, but does include missed detections, false positives, and identity switches. MOTA reflects how accurately and consistently the tracker detects and maintains the identities of a multitude of objects.

$$\mathbf{M}OTA = \mathbf{1} - \frac{\sum_{\mathbf{t}} \left(\mathbf{m_t} + \mathbf{f}p_{\mathbf{t}} + \mathbf{m}me_{\mathbf{t}}\right)}{\sum_{\mathbf{t}} \mathbf{g_t}} \tag{2}$$

where:
m_t = number of misses at time t
fp_t = number of false positives at time t
mme_t = number of mismatches at time t (identify switches)

A high score of MOTA means the model is linking detections across frames to create object tracks with fewer errors. The total errors found is divided by the total number of ground truth objects across all frames. This way, a fair assessment is made for tracking, even for datasets with large numbers of objects.

Identity F1 Score is a measure used in multi-object tracking to quantify the accuracy of identity assignments or measure the extent to which the tracker maintained correct object identity over time. A higher IDF1 score reflects better tracking performance in maintaining correct object identities through frames.

$$\text{ID precision} \qquad \mathbf{I}DP = \frac{\mathbf{I}D\mathbf{T}P}{\mathbf{I}DTP + \mathbf{I}DFP} \tag{3}$$

$$\text{ID recall} \qquad \mathbf{I}DR = \frac{\mathbf{I}DTP}{\mathbf{I}DTP + \mathbf{I}DFN} \tag{4}$$

$$\mathbf{I}DF\mathbf{1} = \frac{\mathbf{2} \times \mathbf{I}DTP}{\mathbf{2} \times \mathbf{I}DTP + \mathbf{I}DFP + \mathbf{I}DFN} \tag{5}$$

IDTP = Identity True Positives. Correctly matched detections with the correct identity.

IDFP = Identity False Positives. Predicted detections that were assigned the wrong identity.

IDFN = Identity False Negative. Ground truth detections that were either missed or assigned to the wrong identity.

To compute ID scores, all video frames are analyzed together to establish matches between predicted and ground truth trajectories. This matching process is modeled as a bipartite graph problem, aiming to maximize the number of frames where predictions align with the correct ground truth objects. Precision, recall, and the F1 score are then calculated based on a one-to-one correspondence between predicted and actual trajectories, without considering the specific causes of tracking errors [3].

Higher Order Tracking Accuracy is a state-of-the-art evaluation metric that computes the average score of all matched detections and does not count unmatched detections. It presents a fair evaluation by measuring detection and association accuracy, which is a better indicator for measuring tracking performance [4].

$$\mathbf{H}OT\mathbf{A} = \sqrt{\mathbf{D}etA \cdot \mathbf{A}ssA} \tag{6}$$

DetA - detection accuracy, measures how well the tracker detects objects (recall vs. false alarms)

AssA - association accuracy, measures how well the tracker maintains object identities over time.

A key component of HOTA is the function *A(c)*, which quantifies the similarity between predicted and ground-truth trajectories for identity *c*. To evaluate association accuracy HOTA introduces [4]:

- **True Positive Association *TPA(c)***: the predicted and ground-truth IDs both correctly match *c*.
- **False Negative Association *FNA(c)***: the ground-truth ID is *c*, but the prediction is incorrect or missing.
- **False Positive Association *FPA(c)***: the predicted ID is *c*, but it either mismatches the ground-truth or corresponds to no real target.

$$\mathbf{A}(\mathbf{c}) = \frac{|\mathbf{T}PA(\mathbf{c})|}{|\mathbf{T}PA(\mathbf{c})| + |\mathbf{F}NA(\mathbf{c})| + |\mathbf{F}PA(\mathbf{c})|} \tag{7}$$

By adding these association types, HOTA captures errors in a way that closely aligns with human intuition of what ”correct tracking” should look like.

7. RESULTS ANALYSIS AND COMPARATIVE STUDY

In this chapter, we have provided a detailed analysis and a comparative survey of the three popular MOT algorithms (e.g. SORT, DeepSORT and JDE) executed on applied to challenging real-world video sequences from the MOT16/train and MOT17/train datasets.

The goal of this work is to analyze the strengths and weaknesses of different trackers across various situations under a common set of performance metrics and controlled scenarios.

While there is no need to introduce these datasets and their annotations further, it is worth mentioning that both MOT16 and MOT17 are part of the MOTChallenge benchmark, which is known for providing challenging, complex urban scenes with heavy occlusions and unpredictable camera motion in congested scenes. All of these are one of the main challenges faced by most real-time tracking systems.

The MOT16 and MOT17 datasets are both focusing on pedestrian tracking in real-world scenarios. While they share the same set of video sequences and annotations, the key difference lies in the detection sources provided. MOT16 has a single set of detections generated using the DPM (Deformable Parts Model) detector, while MOT17 extends this by including three different sets of detections per sequence, using DPM, Faster R-CNN and SDP (Scale-Dependent Pooling) detectors. This allows for a more complete evaluation of how tracking algorithms perform under different detection qualities. MOT17 is therefore a superset of MOT16, used for comparing the robustness and adaptability of tracking methods.

7.1 Algorithms Used in Comparative Study

SORT (Simple Online and Realtime Tracking) is a simpler algorithm that does not require any training because it operates entirely based on classical methods. SORT takes as input a series of bounding box detections per frame, such as those provided in the det.txt files of the MOT16 and MOT17 datasets.

These files contain the output of a pre-trained object detector (e.g. DPM, Faster R-CNN or SDP) and include information about bounding box position, size, and confidence score for each detected object.

SORT uses a Kalman filter to predict the next position of each tracked object based on its previous state and then matches current detections to predicted positions using the Hungarian algorithm based on Intersection over Union (IoU) cost. This approach allows SORT to associate objects frame by frame without learning appearance features. [7].

While its simplicity enables high speed, especially when processing video in real time, it tends to struggle with occlusions and identity switches due to the absence of appearance-based re-identification.

FairMOT is a tracking algorithm that performs object detection and identity embedding in a single network. In our setup, we used the pre-trained ReID model mars-small128.pb that was also used in the paper for getting the results, which provides compact appearance features to help FairMOT distinguish between people across frames. This model was trained on the MARS dataset, a large person re-identification dataset. [6].

Since MARS do not share any data with MOT17, using this model does not affect the fairness of the evaluation. The training was done only on MARS, so there is no shared data into the MOT17 sequences. This allowed us to benefit from better identity tracking without breaking MOTChallenge rules or retraining MOT17 data.

The tracking of DeepSORT was performed in two stages. We first applied a person detector on the MOT16 or MOT17 data and extracted appearance features using the MARS re-ID model. The

results were saved as .npy, which contain the bounding boxes and scores for detections on each frame. In the second step, these .npy files were used to perform tracking. We set min_confidence to 0.3 to filter out weak detections, and nn_budget to 100 to influence how many appearance features are retained for each tracked person.

For JDE, we used the official implementation with a model pre-trained on both object detection and person re-identification tasks using the same training data described in the original paper. The model uses a ResNet backbone and is trained end-to-end on public pedestrian tracking datasets, combining detection and embedding in a single network. Instead of the two-stage pipeline required by DeepSORT, which involves tasks of detection and ReID, Since the pretraining data for JDE does not overlap with the MOT17 evaluation set, this ensures a fair evaluation.

In the upcoming sections, we present a quantitative and qualitative comparison of these trackers focusing on standard multitude-object tracking (MOT) metrics including MOTA, IDF1, identity switches, and tracking precision to justify much more than performance scores but also the practical tradeoffs between speed, accuracy, and robustness over the different methods.

7.2 Result Analysis on MOT16 and MOT17 Dataset

To evaluate and compare the tracking performance of JDE, SORT, and DeepSORT we used a comprehensive set of standard MOT metrics including MOTA (Multiple Object Tracking Accuracy), MOTP (Precision), IDF1 (Identity F1 score), HOTA (Higher Order Tracking Accuracy), MT (Mostly Tracked), ML (Mostly Lost), IDs (Identity Switches), FP (False Positives) and FN (False Negatives). These metrics together evaluate how accurately and consistently each algorithm detects tracks maintains object identities over time.

All three tracking methods are based on online tracking paradigms meaning they operate sequentially frame-by-frame using only past and current information without any access to future frames. This online design is particularly important for real-time or streaming applications where low-latency inference Immediate response are required.

The experiments were done using Python 3.8 and PyTorch as the primary deep learning framework. CUDA and GPU acceleration were utilized where supported. For the JDE algorithm tracking tests were performed using RunPod.io with a high-performance NVIDIA RTX 4000 Ada GPU. The environment was configured with CUDA 11.x and appropriate PyTorch GPU builds to ensure compatibility and efficiency.

DeepSORT and SORT were run on local systems with moderate GPU resources since they require significantly less computational power than JDE. This setup reflects a realistic environment and highlights computational demands of end-to-end embedding-based trackers like JDE compared to simpler modular trackers like SORT.

All the results after applying the algorithms described above on MOT16 dataset can be summarized in Table 4. After reviewing the standard evaluation metrics of MOTChallenge, we can note that:

1. JDE is evidently better than both SORT and DeepSORT in all metrics.

2. SORT has very weak tracking ability with low MOTA (24.91%) and extremely high FN (68,904), which means it often misses objects. It tracks only 7.93% mostly (MT) targets and loses 52.41% (ML).
3. DeepSORT is much better than SORT; its MOTA is 2.4× higher, and it has fewer identity switches (689 vs. 2534).
4. JDE excels in MT (63,2%), IDF1 (68,27), and HOTA (57,18), suggesting that it maintains identity better while also detecting and following objects more consistently.
5. However, both DeepSORT and JDE incur similar FP rates (~7k), while SORT has double that.
6. Deep learning methods (DeepSORT and JDE) clearly outperform SORT in all core metrics.

Table 4: Experimental results of algorithms using MOT16 dataset

Model	**MOTA ↑**	**MOTP ↑**	**IDF1 ↑**	**HOTA ↑**	**MT ↑**	**ML ↓**	**IDs ↓**	**FP ↓**	**FN ↓**
SORT	24,908	77,883	29,719	26,402	7,93	52,41	2534	12913	68904
DeepSORT	60,39	81,606	64,367	53,047	34,81	15,86	689	6737	36369
JDE	72,67	81,703	68,273	57,184	63,20	5,35	803	7122	19899

TABLE 5 presents the output results on MOT17, from which the following observations can be made regarding the performance of each algorithm:

1. JDE again outperforms both SORT and DeepSORT across all metrics, confirming its robustness and consistency.
2. SORT shows moderate improvement performance: MOTA increases from 24,91% to 44,78%, and IDF1 from 29,72% to 45,84%. But still suffers from high identity switches and tracking instability.
3. DeepSORT continues to show stable performance: MOTA improves slightly to 47,68%, and IDF1 climbs to 53,87%.

Table 5: Experimental results of algorithms using MOT17 dataset

Model	**MOTA ↑**	**MOTP ↑**	**IDF1 ↑**	**HOTA ↑**	**MT ↑**	**ML ↓**	**IDs ↓**	**FP ↓**	**FN ↓**
SORT	44,777	84,656	45,835	41,133	16,60	39,74	4885	15477	167863
DeepSORT	47,684	84,137	53,871	45,698	19,78	37,36	1570	7478	167467
JDE	74,242	81,624	69,05	57,746	63,73	6,227	980	6137	21277

After analyzing the results from both MOT16/train and MOT17/train, all algorithms perform better on MOT17, due to improved detection (using better detectors) quality. SORT shows some improvement in MOTA and IDF1 but still lags due to high identity switches and tracking losses. DeepSORT maintains stable results with slight gains and fewer errors than SORT. JDE always gives better results than this one with the highest scores for all important metrics on both datasets. These results can be taken as proof that deep learning-based methods, especially end-to-end models like JDE, provide reliable and superior performances over different tracking conditions.

Though SORT may be fast and easy, it will not work well for complex tasks because of its failure to do any appearance-based associations. Therefore, current MOT research continues to push forward deep learning-based approaches using transformer architectures with temporal attention plus end-to-end optimization toward improving tracking accuracy as well as robustness in real-world applications.

Table 6: Results of DeepSORT using 3 types of detectors on MOT17-04 dataset

Sequence	**MOTA ↑**	**MOTP ↑**	**IDF1 ↑**	**MT ↑**	**ML ↓**	**IDs ↓**	**FP ↓**	**FN ↓**
MOT17-04-DPM	33,333	80,051	40,401	7,22	49,39	112	1082	30510
MOT17-04-FRCNN	54,027	89,423	62,428	18,07	50,62	93	41	21781
MOT17-04-SDP	75,717	86,711	72,614	50,62	33,73	91	157	11279

The results shown in TABLE 6 prove how much the choice of detectors affect the overall performance of DeepSORT tracking algorithm. Out of three tested detectors: DPM, FRCNN and SDP; DeepSORT with SDP gives best results with top MOTA (75.72%) and IDF1 (72.61%) along with least false negatives (11,279) and identity switches (91). This proves that good detections lead to precise and stable tracking. In contrast, DPM, the oldest and least accurate detector, results in the poorest performance, with MOTA dropping to 33,33% and IDF1 to 40,40%, as well as increased identity switches and tracking loss.

Another way of evaluating the performance of object tracking can be HOTA metrics that use a graphic representation where x axis is represented by the alpha (α) value and y axis represents the score of each metric.

The α parameter controls the trade-off between the detection and the association accuracy, ranging from 0 to 1 (or from no association to only association). When α grows, the consideration is more on consuming consistent object identities over frames.

The HOTA-based evaluation shows that JDE achieves the highest overall performance with a HOTA score of 0,57, thanks to strong detection accuracy (DetA = 0,62), recall (DetRe = 0,68), and high precision (DetPr = 0,78) and localization (LocA = 0,84). The association accuracy is moderate (AssA = 0.53), but robust detection places it at the top. DeepSORT comes next with a HOTA of 0.53, showing balanced performance in both detection and association by having slightly better AssA (0.54) and AssRe (0.60) compared to JDE while keeping DetPr and LocA similar; meanwhile SORT has a HOTA of 0,26, due to low scores in both detection and association, particularly DetA (0,28) and AssA (0,26), confirming its limitations in identity tracking caused by its lack of appearance modeling.

The comparison between the MOT16 and MOT17 datasets shows that JDE is better than DeepSORT and SORT in both HOTA and CLEAR MOT. JDE has the best HOTA scores,0,57 on MOT16 and 0,58 on MOT17. This means that it has a better ability to keep a balance between how well things are detected and how well they are associated with identities. Strong detection performance (e.g., DetA = 0.63, DetRe = 0.69) and good association (e.g., AssA = 0.54, AssRe = 0.60) make it the most robust tracker in different conditions.

HOTA is important and, in addition, to traditional metrics such as MOTA and IDF1 to evaluate the quality of detection, association, and localization together. It provides a more comprehensive

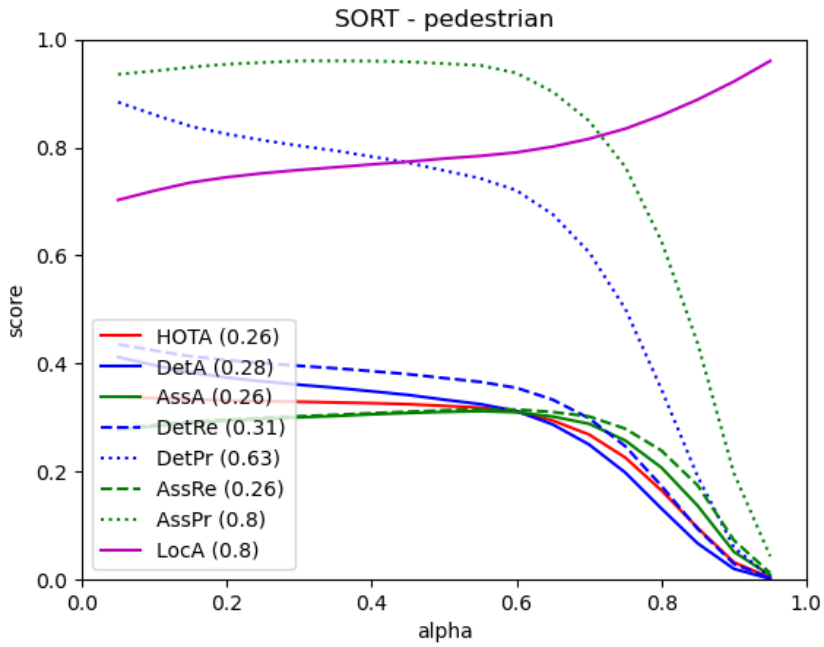

Figure 3: HOTA representation for MOT16 SORT

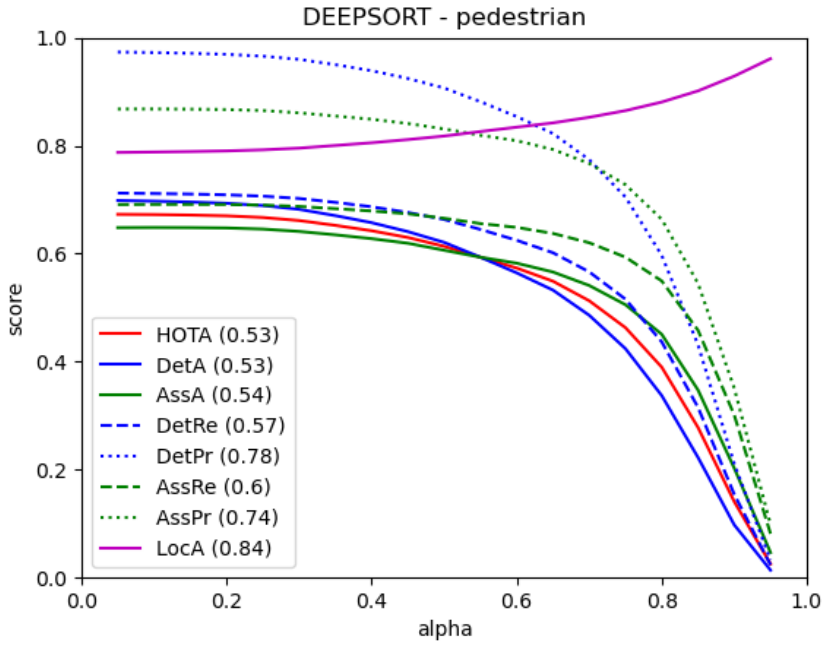

Figure 4: HOTA representation for MOT16 DeepSORT

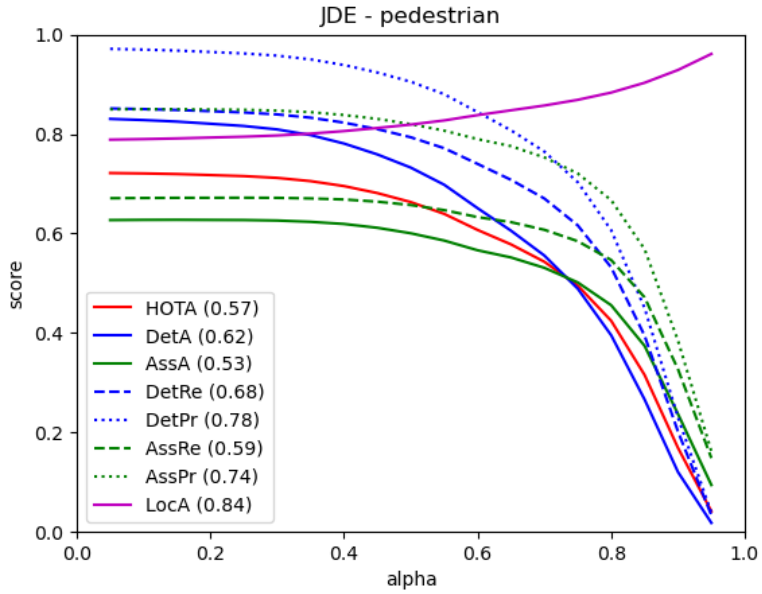

Figure 5: HOTA representation for MOT16 JDE

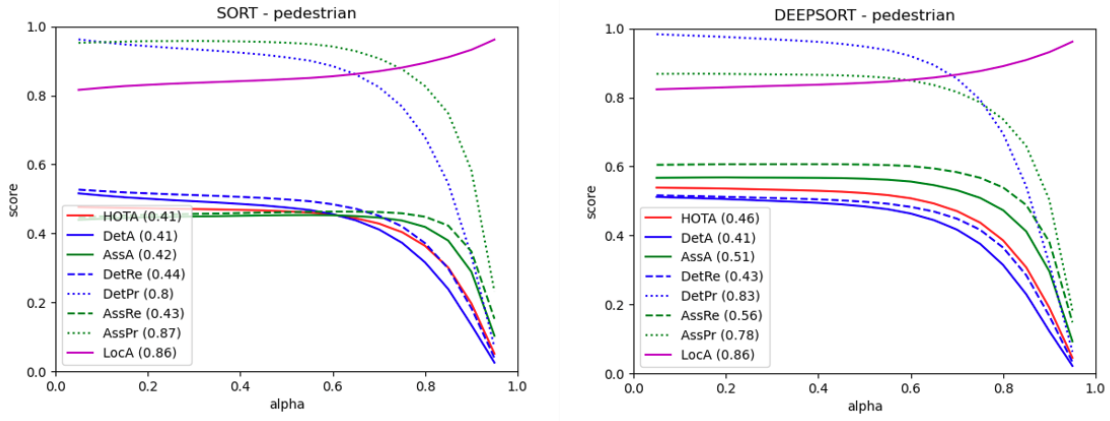

Figure 6: HOTA representation for MOT17 SORT

evaluation of tracking performance. This strongly suggests that practical approaches to multi-object tracking will be based on deep learning, mainly those which unify detection and embedding in a single deep network, such as in JDE. This study showed that deep, joint models provide the best tradeoff of accuracy, stability, and identity preservation in multi-object tracking tasks.

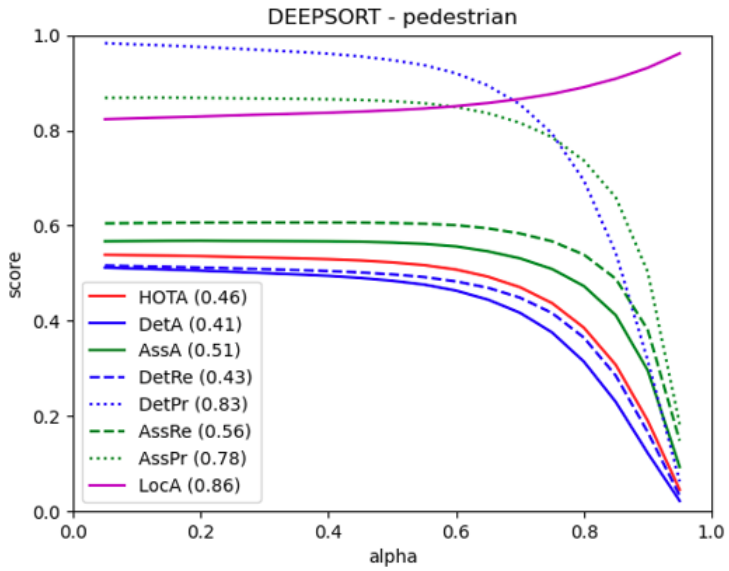

Figure 7: HOTA representation for MOT17 DeepSORT

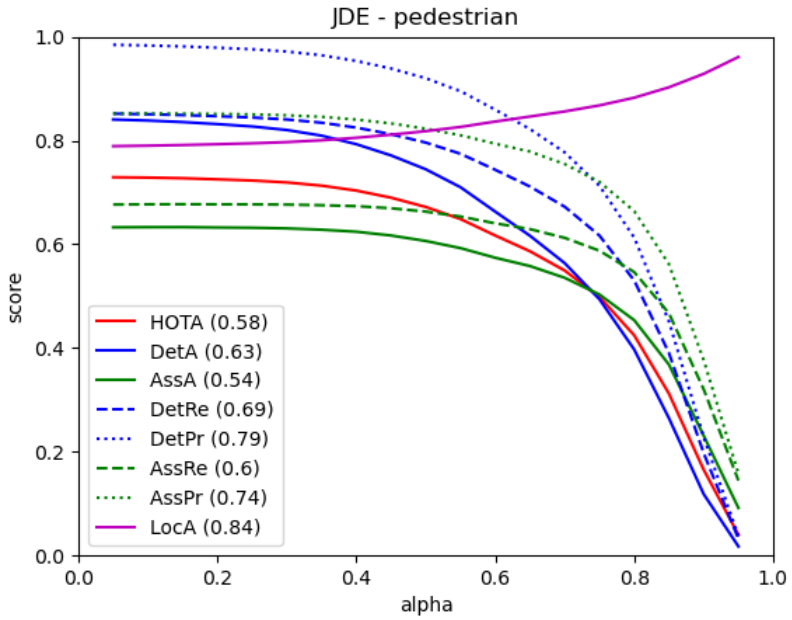

Figure 8: HOTA representation for MOT17 JDE

8. CONCLUSION

The main aim of this project was to assess and review the performance of different tracking algorithms using the image sequences from the well-known datasets MOT16 and MOT17. The three main tracking methods reviewed were SORT, DeepSORT, and JDE due to their real-time perfor-

mance, increasing use and demand in the research and commercial fields, and different methods of associating objects over time. The study ranged from understanding how each method works internally to running them on real sequences to also understanding their positive and negative aspects along with the trade-offs between accuracy and efficiency.

We used both numerical measures such as MOTA, IDF1, and identity switches to assess performance, and qualitative visual inspection to get helpful insights about each method's performance. SORT was the fastest and easiest algorithm and worked well because it did not require any training and made predictions about motion by filtering for motion and making associations using the Hungarian algorithm. However, because it does not model an appearance model, it was very common to see identity switches in crowded or occluded sequences.

DeepSORT was a marked enhancement over SORT. DeepSORT used ReID model-embedded appearance features, which allowed identity tracking to be preserved across frames for a longer duration. However, DeepSORT utilized more hardware resources and required another detection input. JDE emerged as an intriguing case from a joint detection and embedding perspective by combining appearance features and detection within one algorithm and framework. This allowed for the exclusion of the second detection files used for DeepSORT, which improved the speed for tracking and reduced the need for processing resources while still being the highest computational resource requirement among the reviewed procedures.

Collectively, this comparative study identified that no one method is perfect. For lightweight, real-time scenarios with minor occlusions, via market detection SORT was applicable. DeepSORT was fast and practical with solid robustness, and JDE showed promise as consideration for future unified frameworks, especially when the use of more computational resources was available. Overall, DeepSORT exhibited the best overall balance across the MOT16 and MOT17 sequences examined, as it met the challenge of maintaining consistent identities throughout complex scenes, while still being achievable in runtime.

Another important area is the consideration of optimization. Identifying opportunities to optimize runtime performance for more complicated models (e.g., JDE or DeepSORT), for example, could someday facilitate high accuracy tracking for real-time deployment by using methods or tools such as model pruning, quantization or hardware acceleration. In some cases, we may also generate domain-specific, purpose-built lightweight ReID models that may better optimize overall memory footprint, as well as tracking performance (i.e., traffic monitoring, or retail contexts).

Finally, placing the MOT algorithms in the context of a broader system (for example, for behavior analysis, anomaly detection or crowd monitoring) can be an extension of this work. Furthermore, it shifts the focus from just achieving tracking outputs to utilizing tracking outputs, which is becoming ever more prominent in relation to the use of tracking outputs in the application domain, such as in smart surveillance or autonomous navigation.

In general, this study presented three influential MOT algorithms and their advantages and drawbacks, both of which are essential to consider when seeking application in real-world situations. Although much progress has already been made into improving MOT research, the road to highly accurate, efficient tracking systems which can be accessed by everybody is still ongoing. Future work must therefore not only concentrate on improving accuracy as an objective, but also on looking for ways to utilize MOTs in practice and how we might maintain them over time.

References

[1] Ciaparrone G, Luque Sánchez FL, Tabik S, Troiano L, Tagliaferri R, et al. Deep Learning in Video Multi-Object Tracking: A Survey. Neurocomputing. 2020;381:61-88.

[2] Luo W, Xing J, Milan A, Zhang X, Liu W, et al. Multiple Object Tracking: A Literature Review. Artif Intell. 2021;293:103448.

[3] Agrawal H, Halder A, Chattopadhyay P. A Systematic Survey on Recent Deep Learning-Based Approaches to Multi-Object Tracking. Multimedia Tool Appl. 2024;83:36203-36259.

[4] Du C, Lin C, Jin R, Chai B, Yao Y, et al. Exploring the State-Of-The-Art in Multi-Object Tracking: A Comprehensive Survey Evaluation Challenges and Future Directions. Multimedia Tool Appl. 2024;83:73151-73189.

[5] Wojke N, Bewley A, Paulus D. Simple Online and Realtime Tracking With a Deep Association Metric. In: 2017 IEEE international conference on image processing. ICIP. IEEE. 2017:3645-3649.

[6] Zhang Y, Wang C, Wang X, Zeng W, Liu W. Fairmot: On the Fairness of Detection and Re-Identification in Multiple Object Tracking. Int J Comput Vis. 2021;129:3069-3087.

[7] Bewley A, Ge Z, Ott L, Ramos F, Upcroft B. Simple Online and Realtime Tracking. In: 2016 IEEE international conference on image processing. ICIP. IEEE. 2016:3464-3468.

[8] Zhang Y, Sun P, Jiang Y, Yu D, Weng F, et al. Bytetrack: Multi-Object Tracking by Associating Every Detection Box. In: European conference on computer vision. Cham: Springer Nature. 2022:1-21.

[9] Bergmann P, Meinhardt T, Leal-Taixe L. Tracking Without Bells and Whistles. In: Proceedings of the 2019 IEEE/CVF international conference on computer vision. New York: IEEE. 2019:941-951.

[10] Wang Z, Zheng L, Liu Y, Li Y, Wang S. Towards Real-Time Multi-Object Tracking. In: European Conference on Computer Vision. Cham: Springer International Publishing. 2020:107-122.

[11] Zeng F, Dong B, Zhang Y, Wang T, Zhang X, et al. Motr: End-To-End Multiple-Object Tracking With Transformer. In: 2022 European conference on computer vision. Cham: Springer Nature Switzerland. 2022:659-675.

[12] Manafifard M, Ebadi H, Abrishami Moghaddam HA. A Survey on Player Tracking in Soccer Videos. Comput Vis Image Underst. 2017;159:19-46.

[13] Smal I, Meijering E, Draegestein K, Galjart N, Grigoriev I, et al. Multiple Object Tracking in Molecular Bioimaging by Rao-Blackwellized Marginal Particle Filtering. Med Image Anal. 2008;12:764-777.

[14] https://encord.com/blog/yolo-object-detection-guide/.

[15] https://www.chooch.com/blog/what-is-object-detection/.

[16] Redmon J, Divvala S, Girshick R, Farhadi A. You Only Look Once: Unified Real-Time Object Detection. In: Proceedings of the 2016 IEEE conference on computer vision and pattern recognition. CVPR. New York: IEEE. 2016:779-788.

[17] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards Real-Time Object Detection With Region Proposal Networks. Adv Neural Inf Process Syst. 2015;28.

[18] Duan K, Bai S, Xie L, Qi H, Huang Q, et al. Centernet: Keypoint Triplets for Object Detection. In: Proceedings of the IEEE/CVF international conference on computer vision. 2019:6569-6578.

[19] Kalman RE. A New Approach to Linear Filtering and Prediction Problems. Journal of Basic Engineering. 1960;82:35-45.

[20] https://kalmanfilter.net/multiSummary.html.

[21] Cui Y, Zeng C, Zhao X, Yang Y, Wu G, et al. Sportsmot: A Large Multi-Object Tracking Dataset in Multiple Sports Scenes. In: Proceedings of the IEEE/CVF international conference on computer vision; 2023:9921-9931.

[22] https://en.wikipedia.org/wiki/Python_(programming_language.

[23] https://numpy.org/.

[24] https://pytorch.org/.

[25] https://pypi.org/project/opencv-python/.

[26] https://matplotlib.org/.

[27] https://github.com/cheind/py-motmetrics.

[28] Milan A, Leal-Taixé L, Reid I, Roth S, Schindler K. MOT16: A Benchmark for Multi-Object Tracking. 2016. arXiv preprint: https://arxiv.org/pdf/1603.00831.

[29] Dendorfer P, Rezatofighi H, Milan A, Shi J, Cremers D, et al. MOT20: A Benchmark for Multi Object Tracking in Crowded Scenes. 2020. arXiv preprint: https://arxiv.org/pdf/2003.09003

[30] Bernardin K, Stiefelhagen R. Evaluating Multiple Object Tracking Performance: The Clear Mot Metrics. EURASIP J Image Video Process. Semantic Scholar. 2008;2008:1-10.

[31] https://motchallenge.net/.

[32] https://www.lightly.ai/blog/yolo.