

*Deep Learning Project Report*

**Person Tracking System**

By

Sandhi Jain (209309030)

**Under the supervision**

**of:**

Rohini R. Rao

Abhilash K. Pai

**Department of Data Science and Computer Applications**

**Faculty of Engineering**

**Manipal Institute of Technology**

**Manipal**

October 2022

Table of Contents:

1. Abstract……………………………………………………………3
2. Introduction ……………………………………………………….5
3. Project Objectives……………………………………………….11
4. Literature Review ……………………………………………….11
5. Tracktor++….…..…..…………………………………………….16
6. Results and Discussion…………………………………………18
7. Bibliography……………………………...................................23

**Abstract:**

Multiple people tracking is a key problem for many applications such as surveillance, animation or car navigation, and a key input for tasks such as activity recognition. In crowded environments occlusions and false detections are common, and although there have been substantial advances in recent years, tracking is still a challenging task. Tracking is typically divided into two steps: detection, i.e., locating the pedestrians in the image, and data association, i.e., linking detections across frames to form complete trajectories. For the data association task, approaches typically aim at developing new, more complex formulations, which in turn put the focus on the optimization techniques required to solve them. However, they still utilize very basic information such as distance between detections. In this thesis, I focus on the data association task and argue that there is contextual information that has not been fully exploited yet in the tracking community, mainly social context and spatial context coming from different views.

This involves the following steps:

* Detection: - First, all the objects are detected in the frame.
* Association: - Once we have detection for the frame, a matching is performed for similar detections with respect to the previous frame.
* Object Detection: - detects if there are any objects in an image. (eg. x and y coordinates & width and height)
* Object Recognition: - recognizes what kind of the detected objects are (eg. class labels)
* Object Tracking: - it is the process of locating moving objects over time in videos.

It involves the following steps:

1. Taking an initial set of object detections,
2. Creating unique ID for each of the detections,
3. Tracking the objects over time,
4. Maintaining the ID assignment.

**Dataset:**

We aim to use a combination of the datasets:

[MOT17](https://motchallenge.net/data/MOT17/) - MOT17 Challenge. All MOT16 sequences are used with a new, more accurate ground truth. Each sequences is provided with 3 sets of detections: DPM, Faster-RCNN, and SDP.

[MOT20Det](https://motchallenge.net/data/MOT20Det/) - Pedestrian Detection Challenge. This benchmark contains 8 challenging video sequences (4 train, 4 test) in unconstrained environments. Tracking and evaluation are done in image coordinates. All sequences have been annotated with high accuracy, strictly following a well-defined protocol.

[MOT17Det](https://motchallenge.net/data/MOT17Det/) - Pedestrian Detection Challenge. All MOT16 sequences are used with a new, more accurate ground truth.

[MOT15](https://motchallenge.net/data/MOT15/) is a dataset for multiple object tracking. It contains 11 different indoor and outdoor scenes of public places with pedestrians as the objects of interest, where camera motion, camera angle and imaging condition vary greatly. The dataset provides detections generated by the ACF-based detector.

[MOT20](https://motchallenge.net/data/MOT20/) - Pedestrian Detection Challenge. This benchmark contains 8 challenging video sequences (4 train, 4 test) in unconstrained environments. Tracking and evaluation are done in image coordinates. All sequences have been annotated with high accuracy, strictly following a well-defined protocol.

## **Introduction:**

## What is Object Tracking?

Object tracking is a deep learning process where the algorithm tracks the movement of an object. In other words, it is the task of estimating or predicting the positions and other relevant information of moving objects in a video.

Object tracking usually involves the process of object detection. Here’s a quick overview of the steps:

* Object detection, where the algorithm classifies and detects the object by creating a bounding box around it.
* Assigning unique identification for each object (ID).
* Tracking the detected object as it moves through frames while storing the relevant information.

## Object Tracking vs. Object detection

Object tracking refers to the ability to estimate or predict the position of a target object in each consecutive frame in a video once the initial position of the target object is defined.

On the other hand, object detection is the process of detecting a target object in an image or a single frame of the video. Object detection will only work if the target image is visible on the given input. If the target object is hidden by any interference it will not be able to detect it. Object tracking is trained to track the trajectory of the object despite the occlusions.

### **Theoretical Background**

### **Types of Object Tracking**

There are two types of object tracking: image tracking and video tracking.

#### **Image tracking**

Image tracking is the task of automatically recognizing and tracking the images.

It is mostly applied in the field of augmented reality (AR). For instance, when given a two-dimensional image as an input through a camera, the algorithm detects two-dimensional planar images, which can be then used to superimpose a 3D graphical object.

Once the 3D graphic is superimposed, the user can move the camera without actually losing track of the 2D planar surface and graphic on top of it.

Companies like Apple and Ikea use such technologies to give the customers a virtual experience of how their products will look in their personal settings.

#### **Video tracking**

Video tracking is the task of tracking a moving object in a video.

The idea of video tracking is to associate or establish a relationship between target objects as it appears in each video frame. In other words, video tracking is analyzing the video frames sequentially and stitching the past location of the object with the present location by predicting and creating a bounding box around it. Video tracking is widely used in traffic monitoring, self-driving cars, and security because it can process real-time footage.

4 stages of the Object Tracking process

**Target initialization**

The first step involves defining the object of interest or targets.

It incorporates the process of drawing a bounding box around it in the initial frame of the video. The tracker must then estimate or predict the object’s position in the remaining frames while simultaneously drawing the bounding box simultaneously.

### **Appearance modeling**

Appearance modeling deals with modeling the visual appearance of the object. When the targeted object passes through various scenes like the lighting condition, angle, speed, etc., they may change the appearance of the object, and it may lead to misinformation and the algorithm losing track of the object.

Appearance modeling has to be conducted so that modeling algorithms can capture various changes and distortions introduced when the target object moves.

Appearance modeling consists of two components:

1. Visual representation: it focuses on constructing robust features and representation that can describe the object
2. Statistical modeling: it uses statistical learning techniques to build mathematical models for object identification effectively.

### **Motion estimation**

Motion estimation usually infers the predictive capability of the model to predict the object’s future position accurately.

### **Target positioning**

Motion estimation approximates the possible region where the object could most likely be present. Once the location of the object is approximated, we can then use a visual model to lock down the exact location of the target.

## Levels of Object Tracking

Object tracking can be defined by two levels:

1. Single Object Tracking(SOT)
2. Multiple Object Tracking(MOT): it aims to track objects of multiple classes as we see in self-driving cars.

### **Single Object Tracking**

Single Object Tracking (SOT) aims to track an object of a single class instead of multiple objects. It is also sometimes referred to as *Visual Object Tracking.*

In SOT, the bounding box of the target object is defined in the first frame. The goal of the algorithm is then to locate the same object in the rest of the frames.

SOT belongs to the category of detection-free tracking because one has to manually provide the first bounding box to the tracker. This means that

Single Object Trackers should be able to track whatever object they are given, even an object on which no available classification model was trained.

### **Multiple Object Tracking**

Multiple Object Tracking (MOT) refers to the approach where the tracking algorithm tracks every single object of interest in the video.

Initially, the tracking algorithm determines the number of objects in each frame, following that it keeps track of each object’s identity from one frame to the next frame until they leave the frame.

## *Object tracking challenges (and solutions)*

There are several challenges that one might face while working on object tracking algorithms.

Firstly, it is easy to track an object on a straight road or in a simple environment. In a real-world scenario, the target object will go through deformation, occlusion, background noise, etc.

*Occlusion*

The occlusion of objects in videos is one of the most common challenges. It refers to an interference phenomenon where the object is affected by the background or foreground in which the tracking algorithm loses track of the object. In other words, the algorithm gets confused as multiple objects come

closer. This leads to the issue where the initially identified object is again (mistakenly) tracked as a new object.

One can implement occlusion sensitivity to prevent it. Occlusion sensitivity allows the user to identify which particular feature of the object is confusing the network. Once identified, similar images can be used to correct the biases and help the network to extract features that differentiate the objects.

### *Background clutter*

In any machine learning or deep learning task, the background of the images fed into the algorithm creates a lot of issues. It is the same with object tracking models.

In theory, the more densely populated the background, the more difficult it is to extract features, detect or even track the object of interest.

A densely populated background introduces redundant information or noise that makes the network less receptive to features that are important; they also make the network slow to learn and optimize.

To prevent background clutter, one can use a well-curated dataset that has a sparse background.

### *Training and Tracking Speed*

The modern deep learning algorithms have become much more complex, which means they can extract features and make meaningful correlations; this, in turn, also means that they consume more energy and time.

The tracking algorithm is not a single task algorithm, as we see in image classification and object detection. It is a multitask algorithm that performs object detection, localization, classification and also keeps track of the objects. This type of algorithm is mathematically complex, and it takes a lot of time to train.

We must also keep in mind that the tracking algorithms must perform quickly during the inference time to yield accurate results. Enhancing tracking speed is especially imperative for real-time object tracking models.

To enable faster inference time, the model needs to be carefully designed or chosen. A convolutional neural network, which is the primary architecture for computer vision tasks, can be used in object tracking. These networks are capable enough to perform with great accuracy if designed carefully.

Some of the common algorithms used for tracking objects are Fast R-CNN and Faster R-CNN and their variants. These networks have been proven very efficient in the task of object tracking.

### *Varying spatial scales*

### One of the issues with object tracking is that the target objects can vary in shape and size; with such a variety of information, the learning algorithm can get confused, leading to generalization error.

Below is the list of techniques that can help to tackle the issue of varying spatial scales:

#### Anchor Boxes

These are the predefined measurement of the target object. The boxes are meant to acquire the scale and aspect ratios of target objects. These boxes are fed into the network during training, and it allows the network to learn and understand the position and size of the object.

Anchor boxes also allow the network to detect multiple objects if they are overlapping by separately evaluating features and yielding accurate results.

#### Feature extraction

Features extraction is an important process in all deep learning techniques.

It enables neural networks to understand the data fed into them. The convolutional neural network (CNN) used for object tracking and other computer vision tasks can efficiently extract spatial information.

These networks, however, must be able to extract multi-scale spatial information. At times, when the object is smaller in the given input image, the network may lose too much signal during downsampling or as the signal propagates through the network. As a result, the network will not be able to detect and track smaller objects during the inference. To tackle such a situation, one should use or develop a network that can preserve the information extracted from each layer so that the network can detect objects within multiple CNN layers, including earlier layers where higher resolution remains.

## **Project Objectives:**

* To perform multi-person tracking and assigning Re-IDs to the person present in the image
* To re-implement the Tracktor++ paper accepted by ICCV 2021.
* Learning how to deal with tracking issues in the image dataset

## **Literature Review:**

## *Deep Learning-based approaches to Object Tracking*

## Object tracking has been around for almost 20 years now and a lot of methods and ideas were introduced to improve the accuracy and efficiency of the tracking models.

Some of the methods involved traditional or classical machine learning approaches like k-Nearest Neighbor or Support Vector Machine. These approaches are good in predicting the target object, but they require important and discriminatory features extracted by professionals.

On the other hand, deep learning methods extract these important features and representations by themselves.

***MDNet***

Multi-Domain Net is a type of object tracking algorithm which leverages large-scale data for training. Its objective is to learn vast variations and spatial relationships. MDNet is trained to learn the shared representation of

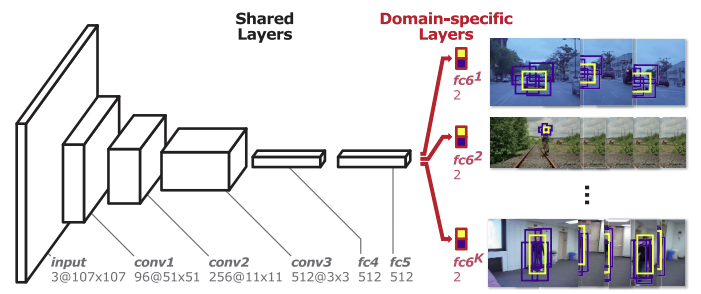
targets from multiple annotated videos, meaning it takes multiple annotated videos belonging to different domains.

MDNet consists of pretraining and online visual tracking:

Pretraining: In pretraining, the network is required to learn multi-domain representation. To achieve this, the algorithm is trained on multiple annotated videos to learn representation and spatial features.

Online visual tracking: Once pre-training is done, the domain-specific layers are removed and the network is only left with shared layers, which consist of learned representations. During the inference, a binary classification layer is added, which is trained or fine-tuned online.

This technique saves time as well as it has proven to be an effective online-based tracking algorithm.



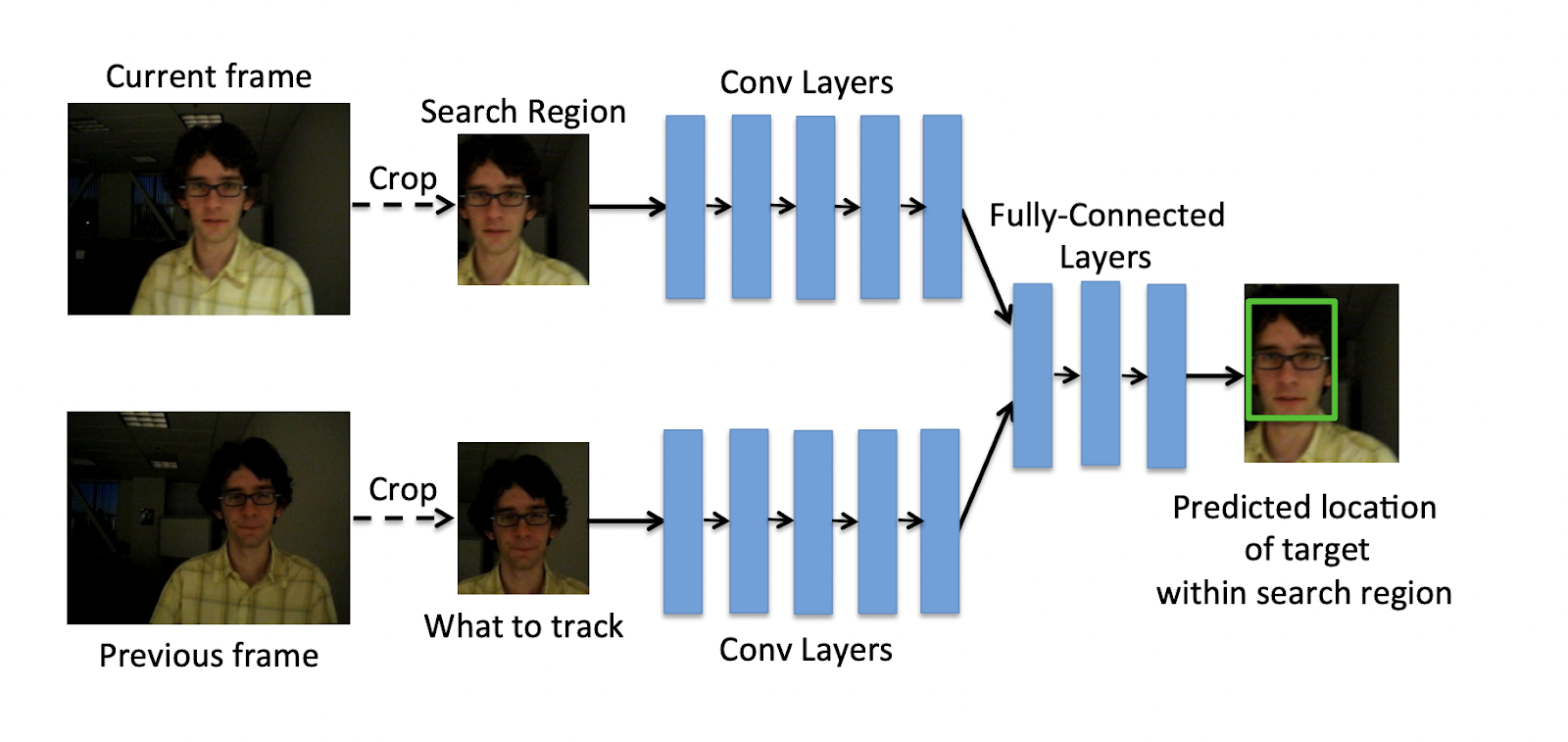
### ***GOTURN***

Deep Regression Networks are offline training-based models. This algorithm learns a generic relationship between object motion and appearance and can be used to track objects that do not appear in the training set.

Online tracker algorithms are slow and do not perform well in real-time; this is because they cannot take advantage of a large number of videos to improve their performance. Offline tracker algorithms, on the other hand, can be trained to handle rotations, changes in viewpoint, lighting changes, and other complex challenges.

Generic Object Tracking Using Regression Networks or GOTURN uses a regression-based approach to tracking objects. Essentially, they regress directly to locate target objects with just a single feed-forward pass through the network.

The network takes two inputs: a search region from the current frame and a target from the previous frame. The network then compares these images to find the target object in the current image.



### ***ROLO—Recurrent YOLO***

ROLO is a combination of recurrent neural networks and YOLO. Generally, LSTM is preferred in combination with CNN.

ROLO combines two types of neural networks: one is CNN which is used to extract spatial information while the other is an LSTM network which is used for finding the trajectory of the target object.

At each time step, spatial information is extracted and sent to the LSTM, which then returns the location of the tracked object.

The video sequence is fed into the YOLO architecture which is primarily made of CNN, here features are extracted as well as bounding boxes are detected.

The visual features and bounding boxes are then concatenated and fed to the LSTM

The LSTM then predicts the trajectory of the objects.

### ***DeepSORT***

DeepSORT is one of the most popular object tracking algorithms. It is an extension to Simple Online Real-time Tracker or SORT, which is an online-based tracking algorithm.

SORT is an algorithm that uses the Kalman filter for estimating the location of the object given the previous location of the same. The Kalman filter is very effective against the occlusions.

SORT comprises of three components:

Detection: Detecting the object of interest in the initial stage i.

Estimation: Predicting the future location i+1 of the object from the initial stage using the Kalman filter. It is worth noting that the Kalman filter just approximates the object’s new location, which needs to be optimized.

Association: As the Kalman filter estimates the future location of the object i+1, it needs to be optimized using the correct position. This is usually done by detecting the position of the object in that position i+1. The problem is solved optimally using the Hungarian algorithm.

With the basics of SORT out of the way, we can incorporate deep learning techniques to enhance the SORT algorithm. Deep neural networks allow

SORT to estimate the object’s location with much higher accuracy because these networks can now describe the features of the target image.

Essentially, the CNN classifier is trained on a task-specific dataset until it achieves good accuracy. Once it is achieved, the classifier is stripped, and we are left with only the features extracted from that dataset. This extracted feature is then incorporated with the SORT algorithm to track objects.

### ***SiamMask***

SiamMask aims to improve the offline training procedure of the fully-convolutional Siamese network. Siamese networks take in two inputs: a cropped image and a larger search image to obtain a dense spatial feature representation.

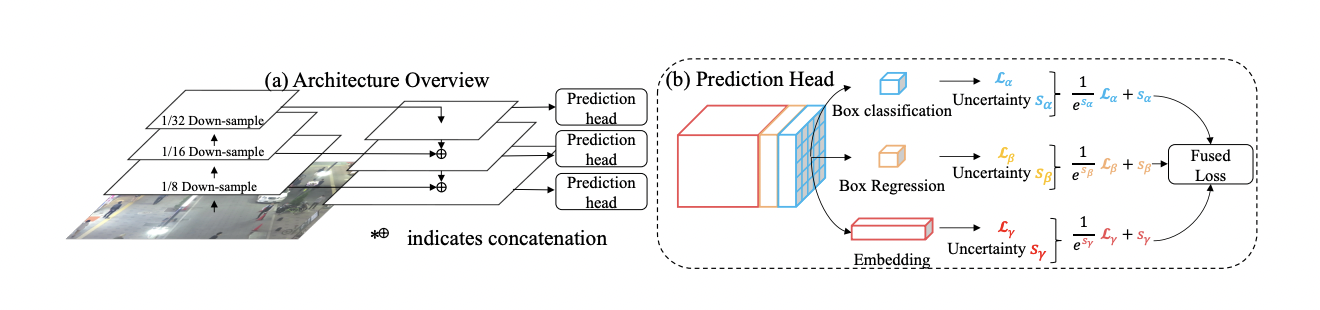
The Siamese network yields one output. It measures the similarity of two input images and determines whether or not the same objects exist in the two images. This framework is very efficient for object tracking by augmenting their loss with a binary segmentation task.

Once trained, SiamMask solely relies on a single bounding box initialization and operates online yielding object segmentation masks.

### ***JDE (Joint Detection and Embedding)***

​​Joint Detection and Embedding (JDE) is a single-shot detector designed to solve a multi-task learning problem. JDE learns target detection and appearance embedding in a shared model.

JDE uses Darknet-53 as the backbone to obtain feature representation at each layer. These feature representations are then fused using up-sampling and residual connections. The prediction heads are then attached on top of the fused feature representation, which yields a dense prediction map.

*JDE architecture*

To perform object tracking, JDE yields bounding boxes classes and appearance embedding from the prediction head. These appearance embeddings are compared to embeddings of previously detected objects using an affinity matrix.

Finally, the Hungarian algorithm and the Kalman filter are used for smoothing out the trajectories of the target object and as well as estimating the locations of the same.

### ***Tracktor***

Tracktor is an online tracking algorithm. It uses an object detection method to perform tracking by training a neural network only on the task of detection.

It essentially predicts the position of an object in the next frame by calculating the bounding box regression. It doesn't perform any training or optimization on tracking data.

The object detector for Tracktor is usually Faster R-CNN with 101-layer ResNet and FPN. It extracts features from the current frame by using the regression branch of Faster R-CNN.

**Tracking without bells and whistles**

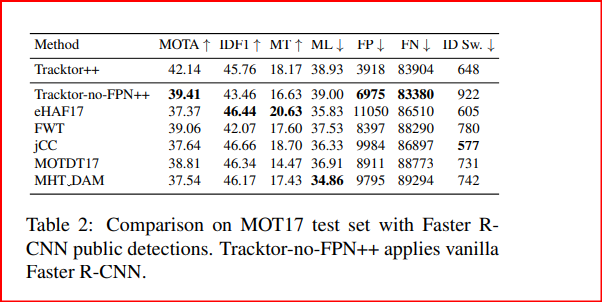
The problem of tracking multiple objects in a video sequence poses several challenging tasks. For tracking-by-detection, these include object re-identification, motion prediction and dealing with occlusions. We present a tracker (without bells and whistles) that accomplishes tracking without specifically targeting any of these tasks, in particular, we perform no training or optimization on tracking data. To this end, we exploit the bounding box regression of an object detector to predict the position of an object in the next frame, thereby converting a detector into a Tracktor. We demonstrate the potential of Tracktor and provide a new state-of-the-art on three multi-object tracking benchmarks by extending it with a straightforward re-identification and camera motion compensation. We then perform an analysis on the performance and failure cases of several state-of-the-art tracking methods in comparison to our Tracktor. Surprisingly, none of the dedicated tracking methods are considerably better in dealing with complex tracking scenarios, namely, small and occluded objects or missing detections. However, our approach tackles most of the easy tracking scenarios. Therefore, we motivate our approach as a new tracking paradigm and point out promising future research directions. Overall, Tracktor yields superior tracking performance than any current tracking method and our analysis exposes remaining and unsolved tracking challenges to inspire future research directions.

**Object detector**. As mentioned before, our approach requires no dedicated training or optimization on tracking ground truth data and performs tracking only with an object detection method. To this end, we train the Faster R-CNN (FRCNN) [ multiobject detector with Feature Pyramid Networks (FPN) on the MOT17Det dataset. In addition, we follow the improvements suggested by [?]. These include a replacement of the Region of Interest (RoI) pooling by the crop and resize pooling suggested by Huang et al. [?] and training with a batch size of N = 1 instead of N = 2 while increasing the number of extracted regions from R = 128 to R = 256. These changes and the addition of FPN ought to improve the detection results for comparatively small objects. We achieve the best results with a ResNet-101 as the underlying feature extractor. In Table 1, we compare the performance on the official MOT17Det detection benchmark for the three object detection methods mentioned in this work. The results demonstrate the incremental gain in detection performance of DPM , FRCNN and SDP (ascending order). Our FRCNN implementation without FPN is on par with the official MOT17Det entry and represents the detector applied in the Tracktor-no-FPN variant of our ablation study.

**Results and Conclusion:**

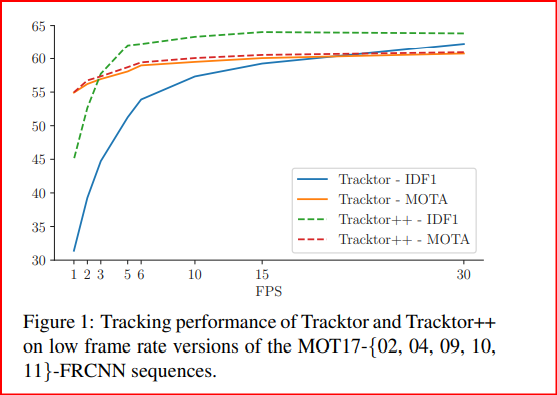
**1. Evaluation metrics** In order to measure the performance of a tracker, we mentioned the Multiple Object Tracking Accuracy (MOTA) and ID F1 Score (IDF1). However, previous Tables such as 3 included additional informative metrics. The false positives (FP) and negatives (FN) account for the total number of either bounding boxes not covering any ground truth bounding box or ground truth bounding boxes not covered by any bounding box, respectively. To measure the track identity preserving capabilities, we report the total number of identity switches (ID Sw.), i.e., a bounding box covering a ground truth bounding box from a different track than in the previous frame. The mostly tracked (MT) and mostly lost (ML) metrics provide track wise information on how many ground truth tracks are covered by bounding boxes for either at least 80% or at most 20%, respectively. MOTA and IDF1 are meaningful combinations of the aforementioned basic metrics. All metrics were computed using the official evaluation code provided by the MOTChallenge benchmark.

**2. Evaluation on public detections** By reclassifying and regressing the given public detections with a private object detector, Tracktor reduces the equalizing effect of public detections to the initialization of new tracks. In addition to our remarks in Section 3 regarding the publicness of our method, we emphasize the potential of Tracktor in comparison with other state-of-the-art trackers even without the advantage of the reclassification and regression. To this end, we show Table 2, which evaluates all trackers on the MOT17 test set only with Faster R-CNN public detections. Tracktor-no-FPN++ (without Feature Pyramid Networks) uses a vanilla Faster R-CNN for reclassification and regression, effectively, not altering the public detections. However, the results support the overall conclusions from Table 2 of our main work.



**3. Tracktor thresholds** To demonstrate the robustness of our tracker with respect to the classification score and IoU thresholds, we refrained from any sequence or detection-specific fine-tuning. In particular, we performed our experiments on all benchmarks with σactive = 0.5, λactive = 0.6 and λnew = 0.3, which were chosen to be optimal for the MOT17 training dataset. In general, a higher λactive than λnew introduces stability into the tracker, as less active tracks are killed by the NMS and less new tracks are initialized. A comparatively higher λactive relaxes potential object-object occlusions and implies a certain confidence in the regression performance.

**4. Tracktor video frame rate robustness** A successful Tracktor bounding box regression depends on sufficiently high video frame rates or, in other words, small frameby-frame object displacements. A possible approach to address this issue is the extension with a powerful motion model. A rudimentary motion model, the camera motion compensation (CMC), is presented in Section 2.3 and evaluated in the ablation study in Table 1. However, MOT16 and MOT17 mostly consist of sequences with benevolent video frame rates and slow moving objects (pedestrians). We therefore complement our analysis of Tracktor in challenging tracking scenarios from Section 4.1 with an evaluation of its video frame rate robustness. To this end, we evaluate Tracktor and Tracktor++ on all MOT17 training sequences with originally 30 frames per second (FPS) and reduce their frame rates by removing frames from the data and ground truth. In Figure 1, both versions exhibit a fairly robust object tracking (MOTA) and identity preservation (IDF1) for rates as low as 5 FPS. As expected, the performance for very small rates suffers particularly with respect to identity preservation.



**Oracle trackers**

In their main work, they conclude a comparison of multiple oracle trackers that highlight the potential of future research directions.For each oracle, one or multipleaspects of our vanilla Tracktor are substituted with ground truth information, thereby simulating perfect behavior. For further understanding, we provide more details on the oracles for each of the distinct tracking aspects:

• **Oracle-Kill**: This oracle kills tracks only if they have an

IoU less than 0.5 with the corresponding ground truth bounding box. The matching between predicted and ground truth tracks is performed with the Hungarian [?] algorithm. In the

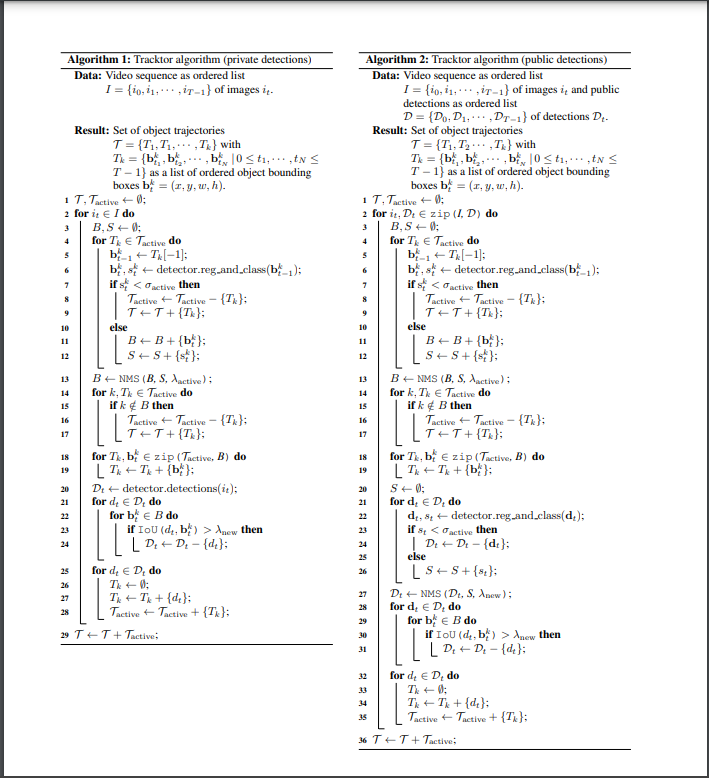
case of an object-object occlusion (IoU > 0.8), the ground truth matching is applied to decide which of the objects is occluded by the other and therefore should be killed.

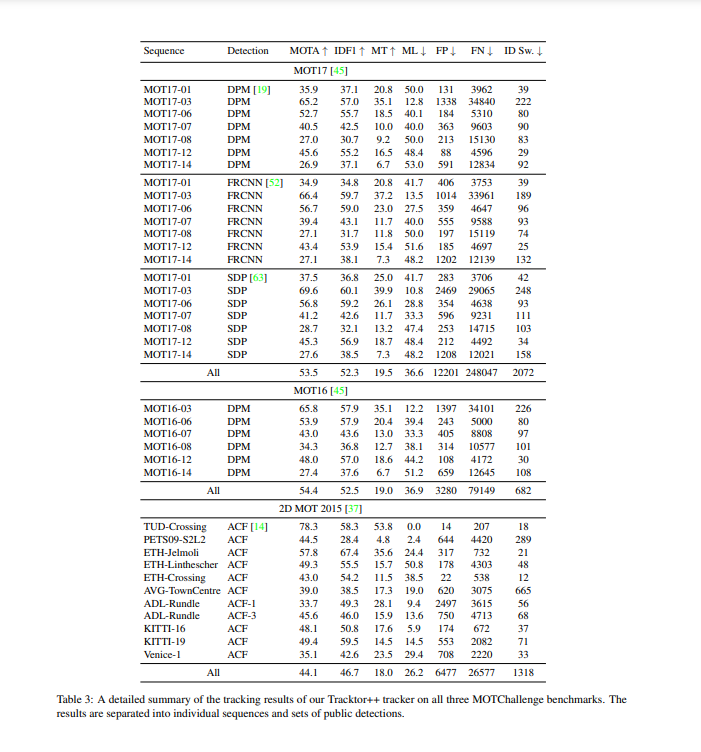
• **Oracle-REG**: We simulate a perfect regression by matching tracks with an IoU threshold of 0.5 to the ground truth at frame t − 1 . The regression oracle then sets track bounding boxes to the corresponding ground truth coordinates at frame t.

• **Oracle-MM**: A perfect motion model works analogous to Oracle-REG but we only move the previous bounding box center to the center of the ground truth bounding box at frame

t. However, the bounding box height and width are still determined by the regression.

• **Oracle-reID**: Again, we use the Hungarian algorithm to match the new set of detections to the ground truth data. Ground truth identity matches between inactive tracks and new detections yield a perfect re-identification.





# **Bibliography**

[1] Z. G. L. O. F. R. B. U. Alex Bewley, "SIMPLE ONLINE AND REALTIME TRACKING," ICIP , p. 5, 2016. A. . Bewley, Z. . Ge, L. . Ott, F. . Ramos and B. . Upcroft, "Simple online and realtime tracking," arXiv: Computer Vision and Pattern Recognition, vol. , no. , pp. 3464-3468, 2016.

[2] N. . Wojke, A. . Bewley and D. . Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," *arXiv: Computer Vision and Pattern Recognition,* vol. , no. , p. , 2017.

[3] P. Bergmann, T. Meinhardt and L. Leal-Taixé, "Tracking Without Bells and Whistles," 2019

IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 941-951, doi:

10.1109/ICCV.2019.00103.

[4] Wang, Z., Zheng, L., Liu, Y., Li, Y., Wang, S. (2020). Towards Real-Time Multi-Object Tracking. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12356. Springer, Cham. https://doi.org/10.1007/978-3-030-58621-8\_7

[5] Zhang, Y., Sun, P., Jiang, Y., Yu, D., Yuan, Z., Luo, P., ... & Wang, X. (2021). Bytetrack: Multi-object tracking by associating every detection box. *arXiv preprint arXiv:2110.06864*.

[6] Yun, S., & Kim, S. (2019, October). Recurrent YOLO and LSTM-based IR single pedestrian tracking. In *2019 19th International Conference on Control, Automation and Systems (ICCAS)* (pp. 94-96). IEEE.

[7] Held, D., Thrun, S., Savarese, S. (2016). Learning to Track at 100 FPS with Deep Regression Networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds) Computer Vision – ECCV 2016.Lecture Notes in Computer Science(), vol 9905. Springer, Cham. https://doi.org/10.1007/978-3-319-46448-0\_45

[8] Jung, I., Son, J., Baek, M., & Han, B. (2018). Real-time mdnet. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 83-98).

[9] S. Noor, M. Waqas, M. I. Saleem and H. N. Minhas, "Automatic Object Tracking and

Segmentation Using Unsupervised SiamMask," in IEEE Access, vol. 9, pp. 106550-106559, 2021, doi:10.1109/ACCESS.2021.3101054.

[10] L. Hongmei, H. Lin, Z. Ruiqiang, L. Lei, W. Diangang and L. Jiazhou, "Object Tracking in Video Sequence based on Kalman filter," 2020 International Conference on Computer Engineering and Intelligent Control (ICCEIC), 2020, pp. 106-110, doi: 10.1109/ICCEIC51584.2020.00029.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |  | |  |  | |  |
|  |  |