UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

Sports broadcasting and multiple object tracking with deep learning methods

Andreu Girbau Xalabarder

UPC UNIVERSITAT POLITÈCNICA DE CATALUNYA

AUTOMATICTV

Universitat Politècnica de Catalunya
Teoria del Senyal i Comunicacions

This thesis is submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy (PhD)

Sports broadcasting and multiple object tracking with deep learning methods

by Andreu Girbau Xalabarder

Advisor: Prof. Ferran Marqués Acosta
Co-advisor: Prof. Xavier Giró Nieto
Co-advisor: Dr. Ignasi Rius Ferrer

Barcelona, January 2021

Abstract

Since less than a decade ago, deep learning techniques started to dominate many different fields, revolutionizing the possibilities of artificial intelligence. Seeing their potential, industrial sectors started to invest on applying such technologies as key components of the company strategy. This thesis has been developed in an industrial context, in AutomaticTV. The main focus along this period has been the transfer of knowledge and know-how between the academia and industry, development of tools to exploit this knowledge, the exploration of new techniques for future challenges, and, from an academic research perspective, contributions to the multiple object tracking problem.

The first part of the thesis is devoted to the introduction of deep learning technologies to AutomaticTV, a company dedicated to automatic sports analysis and broadcasting, and the development of tools and tasks that surround the application.

The second part of this thesis introduces the contributions to the multiple object tracking challenge. We present *TrajE*, a trajectory estimator based on mixture density networks and beam search, used to boost the performance of existing multiple object trackers, and introduce an occlusion reconstruction step using the estimated trajectory information. By adding TrajE to an existing multiple object tracker, we boost its performance by 6.3, 1.8 points in MOTA and IDF1 scores respectively, becoming the new state of the art in the MOTChallenge dataset.

Acknowledgments

Ten years ago I joined the UPC without knowing much about anything in particular. Ten years later, with the presentation of this thesis, I still do not know much about anything, but all I have are words of gratitude for the people I met during these years.

First of all, I want to thank my advisor of my bachelor thesis, master thesis, and, finally, PhD thesis, Ferran Marqués, for introducing me to this exciting research world. To my co-advisors Xavi Giró and Ignasi Rius, for betting on me to carry on this industrial PhD.

To all the people along the way, university colleagues, a special shoutout to ”els competitius” and co., to the friends made across the globe during this period, to my friends in Girona ”leones team” (I didn't chose the name), to the university personnel, and to my co-workers in AutomaticTV, which made this thesis a pleasure.

Of course, this thesis could not have been fulfilled without the support of my family. Both my parents, Lluís and Eulàlia, which have been my referent for all these years, and to my siblings, Maria, Joan, Mercè, Joaquim, Anna, Pere, Xavier, Núria, and Tomàs, along with my grandparents and cousins. Finally, a special thanks to Arisa, my next big adventure.

Contents

Introduction

1

In 2011, the term *fourth industrial revolution* was coined by the German government as a strategy to promote the computerization of manufacturing. Since then, it has included new emerging technologies, such as Internet Of Things, Cloud Computing, or Artificial Intelligence.

In this "4.0 revolution" context, the project AutomaticTV was born in 2010, from the R&D development team in Mediapro, in Barcelona. Some of the company goals can be summarized in generating automatic productions of sports events, and providing tools for the teams to be able to further analyse their matches and training sessions. To do so, the system uses computer vision techniques to gather useful information of the scene.

Deep learning techniques started their own revolution in 2012, where the computer vision community foresaw the uprising when the results on the ImageNet challenge were published. AlexNet [1], competing in the image classification track, used a convolutional neural network to solve the task, outperforming by a huge margin the challenge winners from previous years. While research on deep learning [2, 3] has been cultivated since 1943, with the formal definition of an Artificial Neural Network, it wasn't until recently that the three dimensions that deep learning techniques need to work, **algorithms**, **datasets**, and **computation**, began to be accessible to the general public, making many research fields to be dominated by these methods. In Figure 1.1, we show the evolution throughout the years for challenges related to computer vision.

With the rise of deep learning algorithms, several challenges that were thought unsolvable started to fall down like domino pieces, and the industry began to be interested. In this context, this thesis was born from the collaboration between AutomaticTV, and the UPC (Universitat Politècnica de Catalunya).

1.1 Objectives

This thesis has been conducted under the Generalitat de Catalunya's Industrial PhD program in AutomaticTV and UPC. The main goal of the program is to transfer knowledge and know-how between academia and industry, while developing the research profile of the PhD candidate. For this, the main contributions of this thesis in AutomaticTV have been threefold: (i) knowledge assimilation from the academia, (ii) development of tools to exploit this knowledge, (iii) the exploration of new techniques to further develop the product. Some results of this industrial thesis are currently being deployed in the AutomaticTV systems, used by clients around the globe.

On the other side, from an academic research perspective, this thesis has focused on the Multiple Object Tracking (MOT) challenge. The main goal has been on exploiting

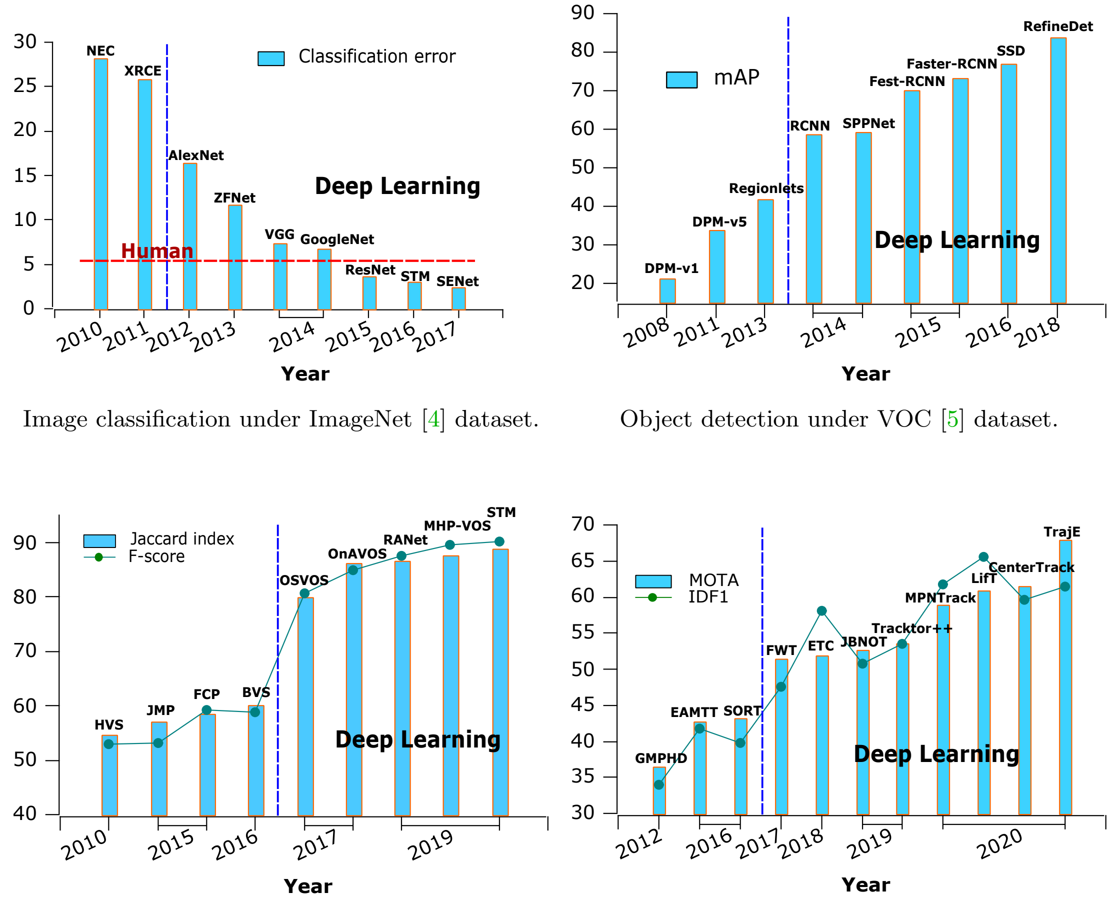


Image classification under ImageNet [4] dataset.

Object detection under VOC [5] dataset.

Object segmentation under DAVIS [6] dataset.

Multiple Object tracking under MOT17 [7] dataset.

Figure 1.1: Evolution of four different computer vision related fields throughout the years. Deep learning techniques became a turning point when AlexNet [1] results were presented in the ImageNet [4] challenge in 2012.

temporal cues, in form of trajectories, providing information that can be very helpful to further boost the performance of the tracker. For this, we developed TrajE, a trajectory estimator using mixture density networks and beam search to generate trajectory hypotheses. We also proposed to use an occlusion reconstruction based on the estimated trajectories. By adding TrajE to an existing multiple object tracker, CenterTrack [8], its performance is boosted by 6.3 points in the MOTA score, becoming the current state of the art in the MOTChallenge [7] multiple object tracking dataset.

1.2 Other contributions

While doing this industrial PhD we contributed to two peer-reviewed publications that are not within the main focus of the thesis.

Tracked Instance Search. *Andreu Girbau, Ryota Hinami, Shinichi Satoh.* This work consisted on using tracking as a generic addition to the task of instance search. In instance search the goal is to, given an example of an object, find several instance of it in a given dataset. In the TRECVID dataset [9], the challenge consists on retrieving a specific person in a specific location in the BBC TV show *Eastenders*, using all the video information needed.

Typically, an instance search system takes a query given as an image (or images) optionally region specified (rectangle or segment), and returns a ranked list of possible instances of that query from a dataset. A problem is that, usually, only a single image is provided, and the results become solely dependent to the specific characteristics of that image (pose, illumination, viewpoint...). In other words, the system becomes very dependent towards the matching between the given visual example and the dataset. Exploiting the video information, we tracked the given example back and forth in time, and provided several instances of the example query to the instance search engine. To combine the several queries, once the instances rankings were retrieved, a voting and re-ranking method were also proposed.

This work was published in the 2018 ICASSP (International Conference on Acoustics, Speech, and Signal Processing) conference.

RVOS: End-to-End Recurrent Network for Video Object Segmentation. *Carles Ventura, Míriam Bellver, Andreu Girbau, Amaia Salvador, Ferran Marqués, Xavier Giró-i-Nieto.* This work consisted on building an end-to-end deep learning model for the task of instance segmentation. This project uses as the baseline architecture RSIS [10], a model for semantic instance segmentation, that produces sequences of binary masks for the different objects within the image using a recurrent neural network in the spatial domain. RVOS extends RSIS by adding recurrence to the temporal dimension, making the instance segmentation per object available for video sequences, and resulting on an end-to-end solution with no pre or post-processing for multiples objects at the same time, making RVOS a very fast multiple video object segmentator.

RVOS is compared against DAVIS [11], and YOUTUBE-VOS [12] video instance segmentation datasets, and was presented in the 2019 CVPR (Conference on Computer Vision and Pattern Recognition) main conference.

1.3 Thesis outline

This thesis is divided in two parts. **The first part** focuses on the technology transfer in the industrial context, its challenges, and the work done to make AutomaticTV system improve. **The second part** consists on the contributions to the multiple object tracking challenge.

Part I

Technology transfer in the Mediapro context: AutomaticTV

Introduction

2

This first part of the thesis is devoted to the introduction of deep learning technologies to an industrial context. The main goal of the Industrial PhD program is the flow of information and knowledge between the academia and the industry in both directions. This knowledge may not come directly from a research perspective, but also from other sources, such as coding practices or team and resources management. The industrial development has been carried out in AutomaticTV, which focuses on automatic sports broadcasting.

Starting with a little bit of history, broadcasting of sports is dated to start in the 1890s. The plays between the Montreal and Winnipeg ice hockey teams was being broadcasted via telegraph, to update the Winnipeg fans of the Stanley Cup challenge series. In the 1920s, the first radio broadcast of an ice hockey game took place. Latter, in 1936 with the Berlin Olympic games, the first sports broadcasting through television was made. Since then, the broadcasting technology has improved considerably.

Nowadays, a production of a live match involves the use of OB (Outside Broadcasting) vans, broadcasting cameras, camera operators, producers, TV engineers, a director, and other staff that are required to carry on the event. The cost of a minimum infrastructure for this broadcast might not justify, in monetary terms, the production of minority sports or leagues. An open debate on whether a sport is minoritary due to the lack of coverage in comparison to others is being held, but still, the broadcasting model seems not to be scalable.

In this context, AutomaticTV is born from Mediapro's R+D department. The objective of AutomaticTV is to make sport productions and analysis scalable. The target market is divided into low budget segments, which are primarily low-cost internet streaming channels, minority sports, schools and college sports, or the lower leagues of major sports, and higher budget segments, which are major leagues, clubs, and production companies. AutomaticTV proposed solution is to provide a setting that allows to make an analysis and/or an automatic production of the event, reducing by a huge amount the cost of a typical broadcasting operation.

The following sections introduce the AutomaticTV solution, challenges, and an introduction to deep learning object detection as the main challenge to solve in the AutomaticTV context.

2.1 AutomaticTV

AutomaticTV bases its solution to a sports production depending on the user needs. Typically, two or three cameras cover all the game zone, allowing the user to increase the viewpoints with more cameras. From these physical cameras we extract the information to generate, what we call, virtual cameras, who emulate a camera operator. With these virtual operators the system focuses on the elements of the scene that are interesting for the user, both at broadcast and/or analysis levels.

The product allows remote operation via internet, and its output can be easily integrated to other solutions that are present in the world of production, including graphics, repetitions, or direct sound from commentators.

The system works in a fully automated manner, although there is the possibility to override it and operate manually. As an example, the manual mode is typically used in training cities, where an analyst follows the different training groups in a training session, and records several videos in parallel with the virtual cameras for a posterior analysis. In broadcast use cases, the system uses machine learning and computer vision techniques for the automatic production, which takes the decisions on the camera movement and framing. Every sport has its own logic, and the aim of the system is to find the patterns that take the production to a good solution, in both terms of capturing the plays and production naturalness. It is a very interesting challenge, as there are many non-controllable elements, e.g. a second ball in the field or players trying to fool their opponents, that can make the system get confused.

2.1.1 Setting

The AutomaticTV (ATV) solution consists on one to three fixed cameras, called master cameras, covering the whole scene. In addition, several side cameras can be used to have different viewpoints. From the master cameras, a stitching of the gaming zone is made (shown in Figure 2.1), and, using this stitching and the side cameras, a virtual camera captures the scene to generate a video output. Typically, a huge percentage of the broadcast consists on the virtual camera moving through the stitching of the master cameras, making the side cameras an alternative viewpoint for specific actions (such as a serve in volleyball), replays, or VAR (Video Assistant Referee).

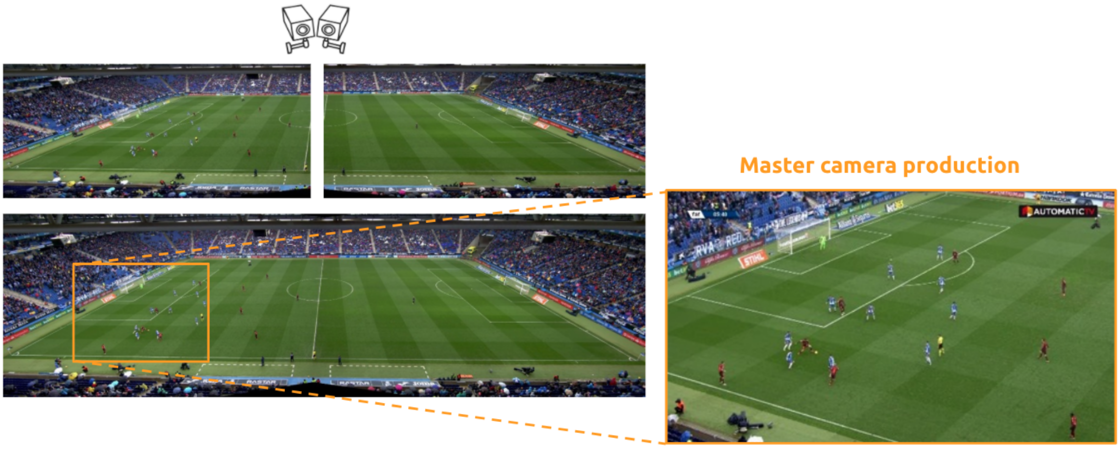


Figure 2.1: AutomaticTV setting. In this example, two physical cameras cover the whole gaming zone, their stitching is computed, and the video output is generated from a virtual camera.

The system includes an on-site server, processing the input stream of all the physical cameras, generating the production, and sending the outputting broadcast video through internet to the broadcaster or to the desired streaming platform.

2.1.2 Challenges

The common use case in an automatic production is to have a single virtual camera (emulating a camera operator) focusing on the region of interest of the scene. The positioning and zooming of the camera will depend on the production strategy.

Understanding visual contents in a scene is crucial to have a good production strategy. Whether these contents are objects of interest in the scene, actions, or directly inferring how a camera operator would move in certain situations, the ATV production system could use this information to propose a camera movement, and the different kind of shot, suitable for the action occurring in the scene.

In recent years deep learning has arised as the state of the art for many different methods. We believe it is not only due to their outstanding performance, but also for being much "standardizable", in the sense of being able to be easily customizable to other areas and challenges.

For the ATV broadcasting strategy, an existing challenge is concerning the object detection. If all the objects in the scene were correctly detected, the strategy could set its base to these detections. For this, the main challenge regarding the ATV automatic production strategy addressed in this thesis was to enhance the product by applying state of the art deep learning techniques, including the building of a dataset, designing the models architecture, and the training and maintenance of such models.

2.2 Object detection

There are many techniques that allow us to understand a scene. Object detection, object segmentation, saliency prediction, pose estimation, or scene captioning are some of the methods that can provide useful tools for further analysis. For the ATV case, knowing the position and semantic class of every object can give good insights on what is happening during a sports match, and seems to be the first and probably the most important milestone.

Object detection is defined as the task of assigning a location and label to the objects in the scene, answering the question of "what objects are where?". This location is usually modelled as a bounding box, defined depending on the consumer's taste. Some researchers use its centroid, width, and height, others the top-left corner, width, and height, and some others use the top-left and the bottom-right corner. While this 4-point object modeling has been the main trend throughout the years, some researchers argue that other definitions for the bounding boxes are more suit to solve the object detection problem, e.g. predicting these 4 points + rotation of the bounding box [13]. For simplicity, and because typically the elements in a sports field are usually placed vertically in the scene, we stick with the 4-point bounding boxes. In this part of the thesis we use centroid (cx, cy), width (w), and height (h), and define a bounding box as:

$$s = \{cx, cy, w, h\} \tag{2.1}$$

We detect the objects of interest in a scene, depending on the ATV production needs. For an object detector to be in the ATV production pipeline, the constraints it should fulfill are: high speed, high accuracy, and low memory consumption.

Here, we review the two main families of object detectors, two-stage and one-stage, and the state of the art in object detection.

2.2.1 Object detection review

To detect the objects in a scene we could use a sliding window at different sizes and scales over an image, and classify every proposal to know whether it is an object of interest or not. If the classifier was fast enough, this implementation would be feasible, and could work as our object detector. These are the approaches of the so-called ”classical methods”, SIFT [14] (1999), Viola-Jones [15] (2003), HOG [16] (2005), or DPM [17] (2009), which use hand-crafted features to extract and classify elements of interest. In 2012, with the appearance of AlexNet [1] using deep learning to do image classification, there was a huge shift in the paradigm, as the classifier could rely on features learned by a Convolutional Neural Network (CNN), rather than relying on hand-crafted features.

These deep learning models are very powerful in the sense that they can infer and combine robust and high level features that we are not able to express, even if internally our brain does the job. The problem with directly training a deep neural network classifier and use a sliding window approach is that deep learning methods are very demanding both memory and computation-wise, making this setting not feasible for a real-time object detection working along with other many processes.

In this context, two families of object detectors, *two-stage* and *one-stage*, appeared. The two-stage approach assumes this lack of computation resources and, instead of performing a classification of every position and scale over an image, classifies a subset of positions and scales, known as *object proposals*. The one-stage approach parallelizes the feature extraction across all the locations, making feasible the object detection at high speed.

Two-stage object detectors. Two-stage object detectors rely on two steps (i) a region proposal stage, where several object proposals from regions likely to have an object inside are generated, and (ii) a detection stage, where these proposals are regressed to better fit the object, and classified as an object of interest or background.

The first work using deep learning to detect objects was OverFeat [18] in 2013, which used a multiscale and sliding window approach integrated with a convolutional neural network to extract features for classification.

A year after, in 2014, RCNN [19] (Regions with CNN) was published. Its main idea consists on extracting a set of regions of interest (ROIs) as object proposals using image segmentation techniques such as Selective Search [20] or MCG (Multiscale Combinatorial Grouping) [21], rescale each proposal, fit each one into a CNN model to extract features, and use these deep features to train a Support Vector Machine (SVM) [22] classifier to predict the presence of an object and its class.

In the early 2015, SPPNet (Spatial Pyramid Pooling Networks) [23] was proposed. Previous CNN models required a fixed-size input (e.g. 224x224 for AlexNet), meaning that changing the resolution of the input would lead to wrong results due to the size of the

CNN activations not matching the feature vector size. SPPNet introduced the SPPLayer, allowing the CNN to generate a fixed-length representation regardless of the input size of the image or ROI without rescaling it.

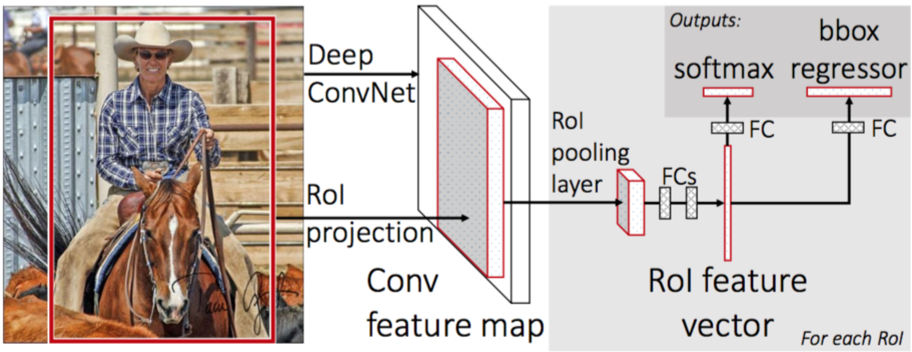


Figure 2.2: The architecture of Fast R-CNN. It computed the features of the whole image in a single pass instead of resizing all the object proposals. Figure from [24].

In 2015 the Fast and Faster-RCNN [24, 25] were proposed. The first one improved RCNN by doing a single forward pass of all the image instead of region by region (i.e. using shared features for every object proposal), crop an object proposal within the features convolutional map, and apply a ROI-pooling layer (following the idea of SPPNet) to have a fixed-length feature vector. Also, in Fast-RCNN the SVM classifier was changed by a set of multilayer perceptrons (fully connected layers) to further boost the performance. In Figure 2.2 the Fast-RCNN method is depicted.

Faster-RCNN was proposed shortly after Fast-RCNN, and it tackled the problem of generating region proposals, introducing the RPN (ROI Pooling Network), shown in Figure 2.3. Instead of relying on an external object proposal method, it developed an internal smaller network that handed object proposals for the inputted convolutional features. The RPN introduces the concept of *anchor boxes*, which are a set of bounding boxes with a size and scale, that are positioned over the feature map (in a sliding window approach), and produce a fixed number of object proposals.

In 2017 the Feature Pyramid Networks (FPN) [26], introduced an architecture that exploited features at different scales, combining them in a forward and backward pass of the network, in order to have high-level semantics at all scales, and, finally, in 2019 TridentNet [27] presented an architecture that uses the field of view of the filters to detect multi-size, multi-scale objects. This is done by using dilated convolutions [28], which are depicted in Figure 2.4, along with FPN's pyramid architecture.

One-stage object detectors. One-stage object detectors skip the region proposal stage, and run directly over a dense sampling of possible locations, as opposed to do the regression and classification over the object proposals in two stage object detectors. These methods became feasible due to hardware evolution and parallelization of feature extraction across all locations.

The first one-stage object detector using deep learning is YOLO (You Only Look Once) [29], presented in 2016. Shown in Figure 2.5, YOLO divides the image into a dense grid at the feature map level, each grid cell predicts B bounding boxes, a probability of that

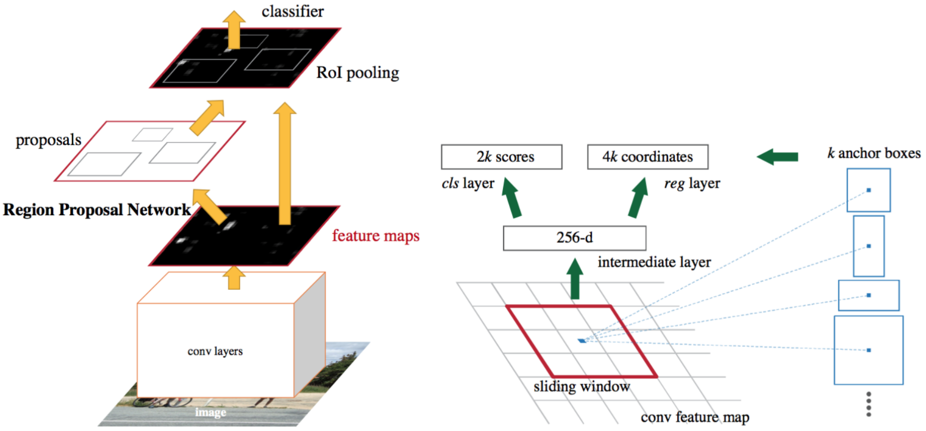
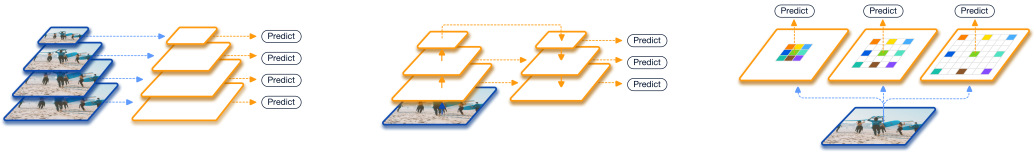


Figure 2.3: Region Proposal Network (RPN) of the Faster-RCNN. K anchor boxes, given a size and aspect ratio, are chosen, and the object proposals will be derived from them. Figure from [25].



(a) Image Pyramid (b) Feature Pyramid Network (c) Trident Network

Figure 2.4: (a) Image pyramid methods perform feature extraction and object detection for each scale of several input scales. (b) The feature pyramid methods utilize the features from different layers of CNNs for different scales. (c) Trident Network generates scale-aware feature maps by trident blocks with different receptive fields. Figure from [27].

bounding box to contain an object, and a probability score per class. A big problem of the YOLO(v1) model is the detection over small objects. The subsequent versions of YOLO(v2-v3-v4) [30, 31, 32] improve the main concept of YOLO using many state of the art techniques.

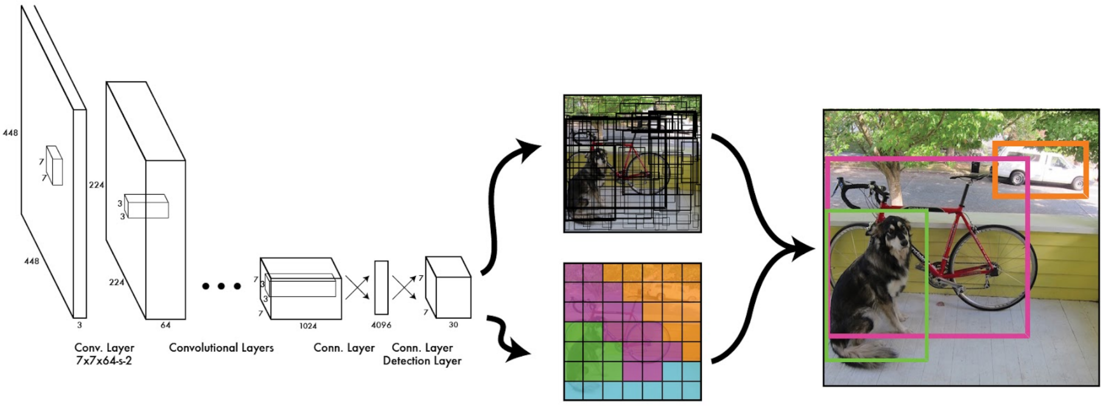


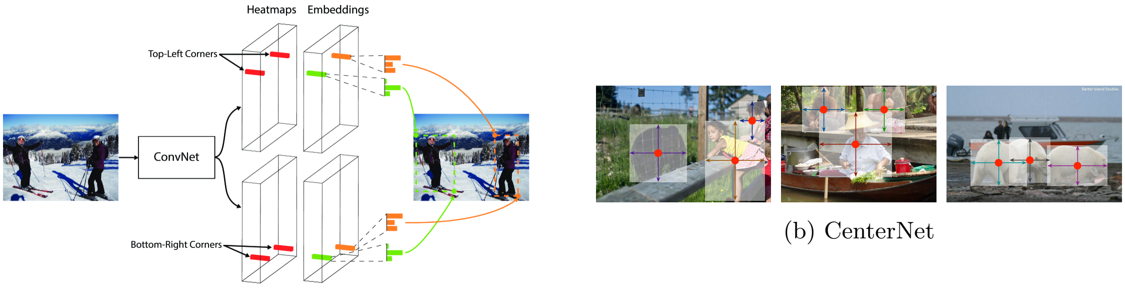
Figure 2.5: YOLOv1 model. Both bounding box and class estimation are calculated in the last layer of the network. Figure from [29].

A couple of months later, SSD (Single Shot multibox Detector) [33] was proposed. The main contribution of SSD is to perform the detection over multiple feature map resolutions, significantly improving the detection accuracy of a one-stage detector, especially for small objects. As an example of this, for the class "bottle" in the VOC-12 [34] challenge, SSD had an mAP of 52.6, while YOLOv1 had a score of 22.7.

Following the YOLO series, YOLOv2, presented in 2017, applied many state of the art techniques, along with a new architecture, to improve the initial YOLOv1. From these techniques we want to highlight three: the model was modified to be fully convolutional, anchors were introduced in the detection stage, and the training was done in a multi-scale manner, modifying the input resolution as a data augmentation technique, doable thanks to the newly fully-convolutional nature of the detector. Following the previous example on the bottle class in VOC-12, YOLOv2 mAP increased to 51.8.

Also in 2017, RetinaNet [35] presented a new loss term to balance the weight of the negative samples on the loss function that they called *focal loss*, and a new neural architecture based on the Feature Pyramid Network (FPN) [26]. The aim of the focal loss is to balance the weights in the loss function of foreground and background classes by adding an exponential decreasing factor for well classified samples. The main claim is that this imbalance was the central cause of the one-stage object detectors performing worse than the two stage object detectors, as the loss was dominated by the grid cells containing background.

YOLOv3 was presented in the following year (2018). As in YOLOv2, they improved the detection by developing a new architecture using several state of the art techniques, such as residual blocks presented in the 2016 in ResNet [36], skip connections, upsampling, detection at multiple scales, and allowing a grid cell to have multiple classes instead of a single one. Finally, YOLOv4, presented in 2020, follows the development of new architectures, maintenance of the code, and new features (usually from the state of the art on several fields) of the YOLO community.



(b) CenterNet

(a) CornerNet

Figure 2.6: Left: CornerNet. Its aim is to estimate the top-left and bottom-right corners of an object associated with a class. Right: CenterNet. It detects the center of the objects in the image, estimating the width and height from it. Figures from [37] and [38] respectively.

Another trend that has recently attracted the attention of the object detection community is to eliminate the need of anchor boxes, and infer the bounding box from estimating key points. CornerNet [37], presented in 2018, changes the paradigm from modeling objects as bounding boxes to detecting objects as a pair of points (top-left and bottom-right). Following this idea, ExtremeNet [39], presented in 2019, detects the top-left, bottom-right, and center points of all objects. Finally, CenterNet [38], also presented in 2019, models the objects as single points (centroid of the object), and derives the information needed from there (the width and height in the case of object detection), and can be easily extended to other fields such as pose estimation.

Other recent works (2020) considering new architectures for improving the model efficiency and accuracy are EfficientDet [40], which proposes a weighted bi-directional feature pyramid network and a scaling method to rescale the model architecture efficiently, and DETR [41], which introduces transformers for the task of object detection.

2.2.2 Object detection in the ATV context

From the ATV perspective, the object detector should be, as all the other processes in the ATV production pipeline, as fast and lightweight as possible. While two-stage object detectors are still giving a better performance than one-stage ones, several works in progress [35, 42] are specifically working on the training step of the one-stage object detectors to close that gap, while keeping the low run-time of the architectures.

Part II

Trajectory estimation for multiple object tracking

Introduction

3

This part of the thesis is devoted to the research conducted on the multiple object tracking challenge. In this introductory chapter, the multiple object tracking and the trajectory estimation frameworks are presented, as well as some core concepts regarding each of these tasks, and the state of the art.

In Chapter 4 we present *TrajE*, a trajectory estimation method based on mixture density networks that aims to boost existing multiple object trackers by predicting the trajectories of the objects in the video sequence, its definition, integration with two state of the art multiple object trackers, and experiments.

3.1 Multiple object tracking

The Multiple Object Tracking (MOT) task aims to estimate tracks of multiple objects across a sequence of frames. These objects must be detected with an accurate bounding box, and their identity over time must be maintained.

Maintaining the identity of the objects detected in a sequence can be very helpful for scene understanding and its further analysis. For instance, in the case of a sports match (e.g. football or basketball), maintaining the identity of every player could be used for detecting offensive or defensive actions and collecting player statistics.

Tracking multiple objects in a video sequence remains an open problem. Some of the challenges come from the unknown number of present objects (as they may vary along the sequence), ambiguities in assigning new detections to the existing tracks (e.g. due to occlusions), or high similarity between objects.

3.1.1 MOT concepts

To solve the multiple object tracking challenge, recent methods benefit from the advances in object detection, and mainly adopt the *tracking-by-detection* paradigm [43]. This consists on (i) obtaining the detections per frame, and (ii) linking the detections to build the objects track along the video sequence. For this reason, many MOT algorithms formulate the task as an assignment problem. Other techniques that are being explored are: regressing the bounding boxes from one frame to another [44, 8], applying correlation filters to the features extracted from a CNN [45], or treating the different objects in a scene as agents from a reinforcement learning perspective [46].

Throughout this thesis we use the following definitions:

Object ID. The identifier assigned to a given object. It consists on a label (usually a number) that specifically identifies the object from the others.

Track. List of positions assigned to an Object ID until the current time instant. For a whole video sequence, tracks gather the position information of all tracked objects.

Trajectory. Concrete instantiation of the past, current, and future positions that an object can take.

Motion. Displacement of a given object between two time instants.

To introduce the main concepts of MOT, we put focus on the *tracking-by-detection* approach. The following steps show the main pipeline.

Detection. The first step is to detect the objects in a scene. These object detections can be already provided by the dataset [7, 47], or be detected by a custom object detector [25, 33, 30, 38]. Formally, although some works characterise detections with fewer parameters [38], we define each detection as

$$s = \{x, y, w, h\} \tag{3.1}$$

where (x, y) correspond to the centroid of a bounding box, and (w, h) correspond to the width and height of the bounding box.

Assignment. Every detection is assigned to a track. If the detection is not assigned to an existing track, a new track is defined with the detection as its first element.

To assign a detection to a track, a distance between the track and the new detection is computed using several measures, that range from the difference between the last object position in the track and the current detection, to the computation of a feature embedding associated to every detection in the scene that compares to a track embedding, usually computed over the last few detections associated to the track (visual example in Figure 3.1). A technique widely used to generate this feature embedding are siamese networks [48], which use a triplet loss regarding positive and negative samples of an anchor object to cluster the ones corresponding to the same object (positive samples), while keeping the negatives samples (from other objects) at a distance.

Track management. Every track has a state associated to it depending on its interaction with the environment. When a track is initialized, i.e. a track is born due to a detection not assigned to any existing track, the track state is set to *active*, and it stays in this state while subsequent detections are associated to the track. If the track lacks a detection in a posterior time step, the track state is set to *lost*. A track can be jumping from one state to the other, depending on whether it has a detection associated to it or not. Usually, a patience indicator is used to terminate the tracks that have spent too much time in the lost state.

Datasets. The datasets for the multiple object tracking challenges include video sequences containing different point of view scenes (from a top-down perspective to almost horizontal). During the sequence, the camera can be static or in movement. The objects in a scene are labeled with an ID, and their positions are given as bounding boxes aligned to each frame time-stamp, thus being modeled as (id, t, x, y, w, h). Popular datasets in the field are KITTI-tracking [49], UA-DETRAC [47], MOTChallenge [7], and MOTS

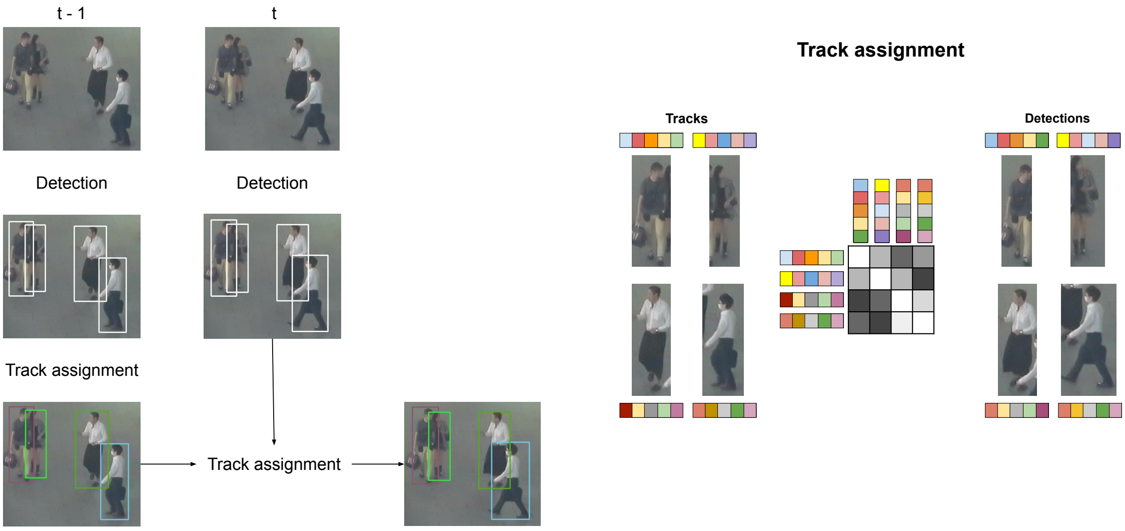
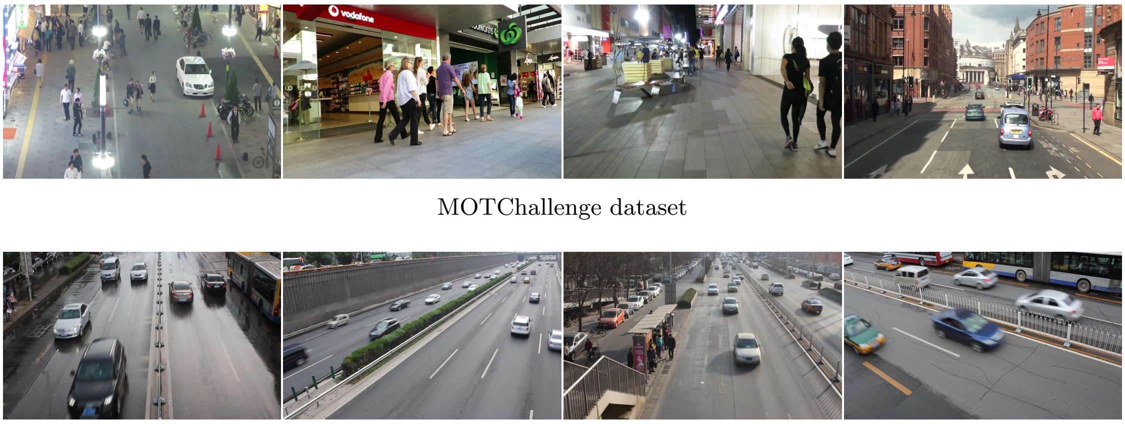


Figure 3.1: Track assignment using feature embeddings. The embeddings from the detections in time t are compared to the tracks coming from $(t - \tau, ..., t - 1)$, and assigned to a track if the embeddings are similar.

(which extends the challenge to segmentation) [50]. In Figure 3.2, we show two examples on MOT datasets (MOTChallenge and UA-DETRAC).


MOTChallenge dataset

UA-DETRAC dataset.

Figure 3.2: Samples on two multiple object tracking datasets. MOTChallenge focuses on pedestrians, while UA-DETRAC focuses on tracking cars in traffic.

3.1.2 State of the art

In general, in tracking we can differentiate between offline [51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63] and online [64, 44, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 45, 8] methods. Offline methods try to globally optimize the tracks tacking into account a whole video sequence (or a part of it), whereas online methods base their decision on previous observations and current data, and do not take into account future information.

Even though the two perspectives usually tackle the multiple object tracking problem from different angles, both paradigms have some common challenges to solve.

To have better object representation in a video sequence, [59, 76] combine high-level detections and lower-level features (in both cases superpixels) to track the objects, [58] uses head and body detections for tracking pedestrians, [67, 74, 55] use CNN features on different network depths, while [54] aggregates multiple features based on temporal attentions, [78] builds a graph similarity model between objects, and [65, 67] build deep affinity networks to model the object at different time instants.

To refine tracks [52] uses behavioral patterns to achieve global consistency, [51] exploits different degrees of dependencies among tracks, [56, 66, 68, 63] focus on filtering noisy detection candidates, [57] cleaves and reconnects the tracks by means of a siamese recurrent network, [62] uses an iterative clustering method to generate tracklets, [60, 61] find a global solution by optimizing a graph, [69, 77] use recurrent neural networks for data association, [73] encodes awareness within and between target models, and [75] uses relation networks in spatio-temporal domain to combine object cues.

To simplify the tracking pipeline, [70, 71] associate detections in $t-1$ to t by means of a Kalman filter and the Hungarian algorithm, [44, 8] connect the detections in $t-1$ to the detections in t by regression, [64] directly assumes good detections in all frames and uses the IoU (Intersection over Union) metric to associate the tracks, while [45] computes the correlation between features from a deep network at multiple depths.

In this thesis, we focused on exploiting the trajectory information for multiple object tracking, and built TrajE, a trajectory estimation model that can be added to those aforementioned trackers that use motion information, showing that it can be useful to boost their tracking performance. In addition, the estimated trajectory during an occlusion can be included to the tracks once they are recovered.

It is not trivial to categorize TrajE as online or offline. If the estimated trajectory is not used to fill the gaps between occlusions, we consider it as a fully online method. Nevertheless, as the trajectory information can be used to fill these gaps, we talk about an online method with a delayed response on the information that is being computed online.

3.2 Trajectory estimation

The trajectory estimation task aims to infer future information of dynamic objects. Several tasks, such as self-driving vehicles, risks prevention, and robotics, use such predictions to make decisions as key factors for their success.

In the context of multiple object tracking, several of its challenges may be alleviated using trajectory information, as knowing the posterior locations of an object may disambiguate and solve situations such as occlusions, re-identification, and identity switching.

The challenge of making accurate predictions of a dynamic object (known as agent) arises from the complexity and the variety of movements that are found in nature, and the way we are able to capture such information. Human motion, for instance, would rely on information not only of the person's past positions, but also on external information (other people crossing paths or objects in the scene), social interaction, or the person's free will to change the walking pattern for no apparent reason.

In this section we introduce trajectory estimation concepts, focusing on human motion, and the state of the art.

3.2.1 Trajectory estimation concepts

When analysing the human motion prediction challenge, [79] decomposes the different methods based on the modeling approach and the type of contextual cues used (if available). The motion modeling approach consists on how the prediction approaches represent human motion, and the contextual cues are all the relevant stimuli (internal and external) that can influence the motion behavior.

Based on the model, the different methods are classified in (i) Physics-based methods, where motion is predicted by dynamics equations based on physical models, (ii) Pattern-based methods, where the dynamics are learned from data, and (iii) Planning-based methods, where there is a reasoning on the agent actions. These models can use (or not) available contextual cues, classified in [79] as (i) Target agent cues, which use the target agent information, (ii) Dynamic environmental cues, where the target agent is aware of other agents, and (iii) Static environmental cues, where the target agent is aware of the environment information (e.g. static obstacles, such as trees or buildings).

In Figure 3.3, the typical elements of a trajectory estimator are depicted. First, there is a stimuli coming from the accessible information (positioning of an agent over time, environmental information...), this information is fed into a trajectory estimation model, and the agent's trajectory prediction is outputted.

Lately, the trend in the field is to use pattern-based methods as the trajectory estimation. These methods follow the *Sense - Learn - Predict* paradigm, and learn motion behaviors by fitting different function approximators (i.e. neural networks, hidden Markov models, Gaussian processes) to data. These Pattern-based methods are classified in [79] as *sequential methods*, which learn models conditioned on the past states, and *non-sequential methods*, which directly model the distribution over full trajectories.

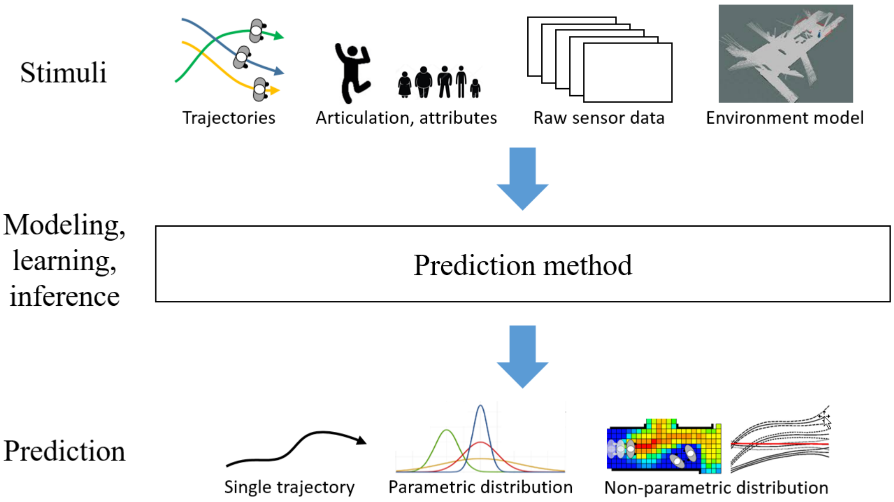


Figure 3.3: Typical trajectory estimation pipeline. A stimuli from an agent is captured depending on the data available, processed by a model, and the trajectory estimation of the agent is predicted. Figure from [79].

First, the trajectory estimator captures the available data (stimuli in Figure 3.3), thus

senses. These stimuli depend on the contextual cues available. They can be partial trajectories, the agents motion, environment information (such as scene geometry), and other extra stimuli that can directly or indirectly influence the movement of the agent. In the trajectory estimation setting, it is common to assume perfect tracking over the sequence for the different agents.

The next step is to model the incoming data to predict trajectories (prediction method in Figure 3.3). In the pattern-based approach, this model is **learned** from the training data.

Finally, the **prediction** of the agents trajectory is inferred by feeding the stimuli to the model. The trajectory estimation can be given in a parametric, non-parametric or structured form of predictions, such as probability distributions over grids, singular or multiple trajectory samples, or motion patterns using graphical models.

It is common in the trajectory estimation methods to use contextual cues, depending on the data available. Target agent cues include agent motion (position and sometimes velocity), articulated pose (such as head orientation), and semantic attributes (such as age and gender or personality). Dynamic environmental cues can be ignored in the so-called unaware methods (where the trajectory estimation does not take into consideration other agents), can take into account other agents individually, or as a group. Finally, regarding the static environmental cues, trajectory estimators can consider obstacles in the scene (static elements or no-go zones), can have the map information (taking into account the topology of the terrain), or can be unaware of this information, assuming an open-space environment.

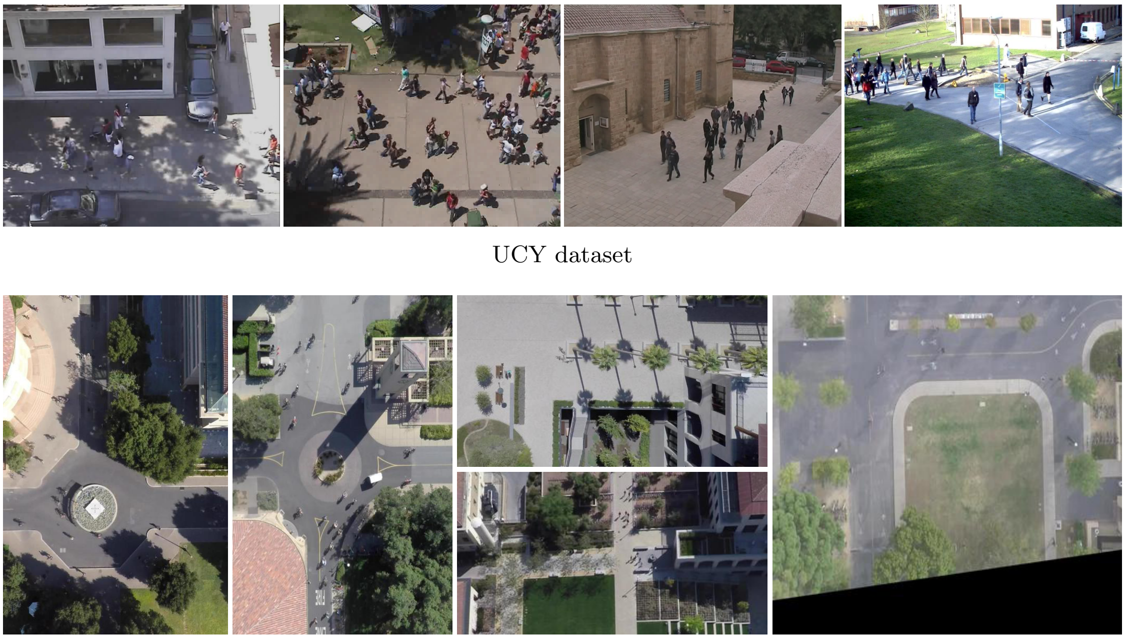
Datasets. The datasets for the trajectory estimation challenges usually include video-cameras with static top-down view of the scene, or ground-based lasers and/or depth sensors, mounted on a static or moving platform. The agents in a scene are labeled with an ID, and their positions are given as points together with the frame time-stamp, thus being modeled as (id, t, x, y). Furthermore, social grouping information, motion, interaction between agents, presence of obstacles in the environment, and other contextual cues can be provided.

Currently, the trajectory estimation community is working towards a common benchmark to compare among methods [80]. Some trajectory estimation datasets include ETH [81], UCY [82], NGSIM [83], KITTI [84], and Stanford drone [85]. In Figure 3.4, two dataset samples (UCY and Stanford drone dataset) are depicted.

3.2.2 State of the art

Taking into account the previous classification, based on the model, TrajE follows the pattern-based sequential approach. It is fully data-driven, meaning that we do not rely on any prior information to model the trajectories to estimate, letting the model reason about the underlying distribution of the data conditioned on the previous observations.

To model temporal dependencies for a sequence, [86, 87, 88, 89, 90, 91, 92, 93] combine social information to predict trajectories, [94] uses convolutional neural networks with overlapping windows, [95, 96, 46] consider a pedestrian in a scene as an agent that makes decisions to predict the next position, and [97, 98] define a spatiotemporal graph to predict multiple futures.



UCY dataset

Stanford drone dataset.

Figure 3.4: Samples on two trajectory prediction datasets, both focusing on pedestrians.

Specifically, [96, 46] use reinforcement learning to predict the objects position in the next time step, [87, 86, 97, 98] use graphical models to model pedestrian interactions, [93] propose a two-step architecture for overcoming mixture density networks limitations, and [89] proposes a GAN for trajectory prediction, using an LSTM to model each trajectory and shares the information as in [88], taking into account obstacles in the scene and other objects in the modelled trajectories.

To predict trajectories, [88, 91, 93, 97, 98], as TrajE, use mixture density networks. Regarding contextual cues, all these methods are aware of the dynamic obstacles in the form of other agents, and some take into account the static obstacles in the scene [88, 91], are aware of the map geometry and topology [93] (considering roads and sidewalks in the scene), or are able to use many diverse contextual information [98], given its modular nature. All these works use complex models, combining the different agent cues in a social-aware layer in [88], using generative-adversarial networks in [91], taking into account the whole image as input to have contextual information in [93], or introducing variational autoencoders with recurrency in a graph formulation in [97, 98].

In contrast, we define TrajE as a much lighter model: it consists on a single layer recurrent neural network, it is unaware of the other agents (dynamic cues) or environmental elements (static cues) in the scene, and the inputs of the network are directly the offsets between the centroids of the object previous positions, i.e. without any other visual or previously ordered information.

We added TrajE to two state of the art multiple object trackers, CenterTrack [8] and Tracktor [44], evaluating them against two well known datasets in multiple object tracking: MOTChallenge 2017 [7], focused on pedestrians in the wild, and UA-DETRAC [47], focused on cars in traffic, modeling the trajectories of all the objects in the sequences.

4 Mixture Density Networks for Trajectory Estimation

To tackle the multiple object tracking problem there are many active fronts in the community, which are directly associated to parts of the multiple object tracking pipeline, such as object modelling, data association, or track refinement. Lately, with the emerge of the *tracking-by-detection* and *tracking-by-regression* paradigms, a strong prior of the object position has become determinant. This prior allows the system to associate tracks with detections, removing them from the combinatorial association problem between objects from different time instants, alleviating the system from possible ambiguities.

To infer this positional prior, it is a common practice to use motion estimators over consecutive frames (e.g. the Kalman filter or optical flow). Though being effective, we observed that this motion is usually considered as an add-on for compensating some weakness that the tracker might have. Instead, we believe that modeling the trajectories of the objects in a scene can provide many useful insights, and can become a key factor of a tracker's performance.

We built *TrajE*, standing for TRAJectory Estimator, a trajectory estimator based on mixture density networks, that learns to estimate the underlying distribution of an object trajectory. By sampling from such a distribution, multiple hypotheses for likely positions of the object can be forecasted.

We combine TrajE with beam search, a widely used technique in machine translation algorithms, in order to keep several hypotheses alive per object. This is not only useful for track association, but also for re-identification, as an object trajectory is being estimated even when the object is occluded, allowing to infer the likely position of an object after an occlusion. We use the estimated trajectory to reconstruct the object track after an occlusion, allowing the tracker to reduce the amount of false negatives corresponding to the occluded parts where the object detector was not able to properly detect the object.

We designed TrajE as a lightweight model, consisting in a single layer recurrent neural network, capable of estimating the trajectories of the objects in a video sequence. We added TrajE to two state of the art multiple object trackers, *CenterTrack* [8] and *Tracktor* [44], improving their performance by a considerable margin.

The rest of the chapter depicts mixture density networks and beam bearch in the key of TrajE, the algorithm itself, and the experiments conducted when applying TrajE on top of the two aforementioned trackers, tested in two datasets, MOTChallenge 2017 [7] and UA-DETRAC [47].

4.1 Mixture Density Networks

Minimization of error functions such as sum-of-squares or cross-entropy leads a neural network model to approximate the conditional averages of the target data, conditioned on the input vector [99]. For problems involving the prediction of continuous variables, the conditional averages provide a very limited information on the underlying structure of data.

For this reason, predicting the outputs corresponding to input vectors from the conditional probability distribution of the target data might lead to better results, as in the context of anomaly detection [100], or handwriting generation [101]. To estimate such a distribution we make use of mixture density networks (MDNs), introduced in [99]. These networks combine a conventional neural network with a mixture density model (e.g. a mixture of Gaussians), so that the neural network estimates the parameters of the distribution.

Following a set of detections from $t - \tau$ to t, TrajE estimates the parameters of a distribution, in this case a bivariate Gaussian distribution, that models the evolution of the trajectory for the next time step $t + 1$. From such a distribution, we sample a set of possible states corresponding to a set of hypotheses for the position of the object in the next time step.

TrajE takes an input a vector, x_t, that consists on a real-valued pair $(x_{1,t}, x_{2,t})$, representing a trajectory point of an object being tracked. As we are dealing with multiple object tracking, we take the centroids of the bounding boxes as the trajectory points. The network outputs, $\hat{y}_t$, model the parameters of a distribution $Pr(x_{t+1}|y_t)$. Some of these outputs model the mixture weights π_t^k while the others estimate the parameters of each mixture component. In this case we use a mixture of Gaussians, so the remaining outputs are in charge of estimating the mean μ_t^k, variance σ_t^k, and correlation coefficient ρ_t^k for the M Gaussians that generate the probability distribution. Note that the mean and standard deviation are two dimensional vectors, whereas the weight component and correlation coefficient are scalar.

$$x_t \in \mathbb{R} \times \mathbb{R} \tag{4.1}$$

$$y_t = \{\pi_t^k, \mu_t^k, \sigma_t^k, \rho_t^k\}_{k=1}^{M} \tag{4.2}$$

$$\pi_t^k \in \mathbb{R};\ \mu_t^k \in \mathbb{R}^2;\ \sigma_t^k \in \mathbb{R}^2;\ \rho_t^k \in \mathbb{R} \tag{4.3}$$

It is important to remark that the mixture components must satisfy several constraints in order to correctly form a valid probability density distribution. To achieve this, several operations are made to the direct outputs of the network $\hat{y}_t = \{\hat{\pi}_t^k, \hat{\mu}_t^k, \hat{\sigma}_t^k, \hat{\rho}_t^k\}_{k=1}^{M}$. First, weights π_t^k must satisfy

$$\sum_{k=1}^{M} \pi_t^k = 1 \tag{4.4}$$

which can be achieved by applying the softmax normalization.

$$\pi_t^k = \frac{\exp(\hat{\pi}_t^k)}{\sum_{k'=1}^{M} \exp(\hat{\pi}_t^{k'})} \tag{4.5}$$

For the other components, to keep their values within a meaningful range, μ_t^k remains the same, the elements of σ_t^k must be positive, and ρ_t^k must be a value between $(-1, 1)$, therefore we apply:

$$\mu_t^k = \hat{\mu}_t^k \tag{4.6}$$

$$\sigma_t^k = \exp(\hat{\sigma}_t^k) \tag{4.7}$$

$$\rho_t^k = \tanh(\hat{\rho}_t^k) \tag{4.8}$$

Formally, the probability density $Pr(x_{t+1}|y_t)$ of the next object position x_{t+1} given the mixture components y_t is defined as follows:

$$Pr(x_{t+1}|y_t) = \sum_{k=1}^{M} \pi_t^k \mathcal{N}(x_{t+1}|\mu_t^k, \sigma_t^k, \rho_t^k) \tag{4.9}$$

where

$$\mathcal{N}(x|\mu, \sigma, \rho) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left[\frac{-Z}{2(1-\rho^2)}\right] \tag{4.10}$$

with

$$Z = \frac{(x_1 - \mu_1)^2}{\sigma_1^2} + \frac{(x_2 - \mu_2)^2}{\sigma_2^2} - \frac{2\rho(x_1 - \mu_1)(x_2 - \mu_2)}{\sigma_1\sigma_2} \tag{4.11}$$

Mixture density networks are trained by maximising the log probability density of the targets under the induced distributions. Equivalently, it is common to minimize the negative log likelihood as loss function.

$$\mathcal{L}(\mathbf{x}) = \sum_{t=1}^{T} -\log\left(\sum_{k} \pi_t^k \mathcal{N}(x_{t+1}|\mu_t^k, \sigma_t^k, \rho_t^k)\right) \tag{4.12}$$

where T corresponds to the length of a sequence (in this case an object trajectory).

Biased sampling. A problem with directly sampling from the generated distribution is that the generated trajectories might lead to a huge space of possible trajectories per object. As the process is iterative (the probability of every output depends on all previous outputs) and the number of samples limited, the generated trajectories may not lead to an optimal solution, as seen in the experiments section 4.4, Figure 4.12 and Figure 4.13, for bias $b = 0$. A way to palliate this effect is by biasing the sampler towards more likely predictions at each time step. To do so, we modify the distribution as in [101]:

$$\pi_t^k = \frac{\exp(\hat{\pi}_t^k(1+b))}{\sum_{k'=1}^{M} \exp(\hat{\pi}_t^{k'}(1+b))} \tag{4.13}$$

$$\sigma_t^k = \exp\left(\hat{\sigma}_t^k - b\right) \tag{4.14}$$

where $b \geq 0$, being Equation 4.5 and Equation 4.7 a specific case for $b = 0$.

4.2 Beam Search

Beam search is a well known technique used in the Natural Language Processing (NLP) community, widely used in tasks like machine translation.

When translating a sentence to another language, there are many possible combinations of words that could be outputted, but some solutions are better than others. These solutions may come not in the first word, but after a few generated words. Ideally, keeping all the possible combinations would be optimal in terms of translation, but unpractical in terms of computation. To deal with it, a subset of options is used and explored in order to find the best translation to the given text.

In the context of multiple object tracking, we hypothesize that a beam search technique would be useful to take into account multiple trajectory hypotheses, as sampling a single point (even the most likely one) does not guarantee the best solution.

In beam search, B stands for beam width. In machine translation the top-B predicted words are considered for making the translation at every estimation step. This is, when translating a sentence, the best B possibilities (given their likelihood) are taken as the first word hypotheses. The next word to be generated comes from this top-B words, ending up in B^2 combinations of words. After this, again, the top-B words are chosen (i.e. pruning the rest), and the next word is generated from these. It is important to note that all the hypotheses are conditioned on the previously generated hypotheses.

The translation is terminated when an <EOS> (End Of Sequence) token is estimated as the next word. Finally, the sentence with the highest likelihood, given the T predictions that form the translated sentence, is chosen as the best translation for the input sentence. This is done by keeping the top-B trajectories that maximize the conditioned probability of the output (translated) sentence y conditioned to the input sequence x, and choosing the one that maximizes the combined likelihood.

$$\operatorname*{argmax}_{y} \prod_{t=1}^{T_y} P(y^{<t>} | x, y^{<1>}, ..., y^{<t-1>}) \tag{4.15}$$

Equivalently, to have a numerically stable algorithm, we can maximize the log probability of the output.

$$\underset{y}{\text{argmax}} \sum_{t=1}^{T_y} logP(y^{<t>}|x, y^{<1>}, ..., y^{<t-1>}) \tag{4.16}$$

In our multiple object tracking case, B encodes both the amount of bounding box centroid positions that are going to be sampled from the trajectory distribution, coming from the MDN, and the beam width.

We start the tracking step by doing B hypotheses for the position in the next time step $t+1$ of an object associated to a track. These hypotheses are propagated and, if there is no new detection associated to the track, again B hypotheses are made, ending up with B^2 hypotheses. To avoid an exponential growth, we prune to keep B trajectory hypotheses, as it is usual in machine translation algorithms. Note that, for $B = 1$, beam search becomes the greedy search algorithm.

In the multiple object tracking scenario, this cycle is over when a) the trajectory being estimated is assigned to a detection, or b) the track patience counter is over, and the track is terminated.

4.3 Trajectory estimation with mixture density networks

To estimate the trajectories of the objects in a video sequence we combine a mixture density network, that aims to model the distribution of the trajectory position in time $t+1$, with a beam search technique to explore several track hypotheses. We believe that, for this problem, using recurrent neural networks is suitable, as the output distribution is conditioned not only on the current input, but on the history of previous inputs.

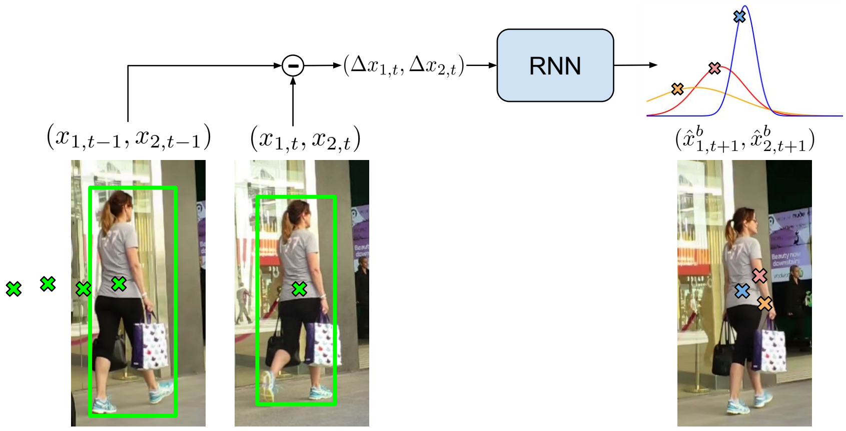

$(\Delta x_{1,t}, \Delta x_{2,t})$
RNN
$(x_{1,t-1}, x_{2,t-1})$
$(x_{1,t}, x_{2,t})$
$(\hat{x}^b_{1,t+1}, \hat{x}^b_{2,t+1})$

Figure 4.1: Main concept of TrajE. The offset between the centroids of two consecutive detections corresponding to a track is computed, and fed to a single layer recurrent neural network, whose outputs correspond to the parameters of a mixture of Gaussians. From these, a probability density function is generated, and the centroids corresponding to the next possible position of the track are sampled from such a distribution.

As trajectory points we use the centroid of the detected objects bounding boxes. For each bounding box centroid $x_t^j = (x_{1,t}^j, x_{2,t}^j)$ assigned to a track, an instance of TrajE will estimate a mixture of Gaussians, modeling the distribution from where the centroid x^j_{t+1} is going to be sampled. Concretely, we feed the difference between the centroids x^j_{t-1} and x^j_t, $\Delta(x^j_{t-1}, x^j_t)$, to the network, and sample the B offsets in the next time step $\Delta(x^{j,b}_t, x^{j,b}_{t+1})$ from the distribution. Predicting the trajectory distribution using the offset instead of the position values can be seen as predicting the motion distribution itself. Also, as in [101], using offsets was essential to train the network. For simplicity, we refer to centroids instead of centroids offset for the rest of the chapter.

Model architecture. As the number of objects in a scene can grow very large, we designed the backbone network of TrajE as a lightweight architecture consisting in a single layer Gated Recurrent Unit (GRU) [102] with a hidden state of size 64 connected with 4 heads of fully connected layers. The two heads that predict the mixture weights π and correlation coefficient ρ have dimension M, while the two heads that predict the mean μ and variance σ of the coordinates have dimension $2M$. The number of Gaussians used (M) can vary, depending on what is being modeled. In TrajE, we set the number of Gaussians in the mixture model to $M = 5$.

4.3.1 Integration of TrajE to object trackers

We have integrated TrajE into two existing multiple object trackers, *CenterTrack* [8] and *Tracktor* [44]. We substituted the motion modules of both trackers for the trajectory estimator, allowing not only to keep estimating the motion, but also the trajectory.

Baselines. CenterTrack and Tracktor are defined as trackers-by-regression, meaning that, by "regressing" the detections in $t-1$ to detections in t, they are able to assign the newly detected object in t to the same track where the original object in $t-1$ belonged.

The main idea in CenterTrack is to regress consecutive frame centroids to associate the tracked objects from $t-1$ to the detections in t. They extend the CenterNet [103] object detector to a network that is able to associate detections over time by means of feeding the model with two consecutive images (I_{t-1}, I_t) and the heatmap of the detections in the previous frame ($\mathcal{H}_{t-1}$), biasing the detector towards previous detections. To further boost their performance, the model computes, as the motion model, the offset between detections in two consecutive time steps, as some sort of sparse optical flow between object centroids. By modeling the objects as points and estimating their motion as offsets, the tracks and detections are associated by means of L1 distance. In Figure 4.2, the regression of detections in CenterTrack from $t-1$ to t is depicted.

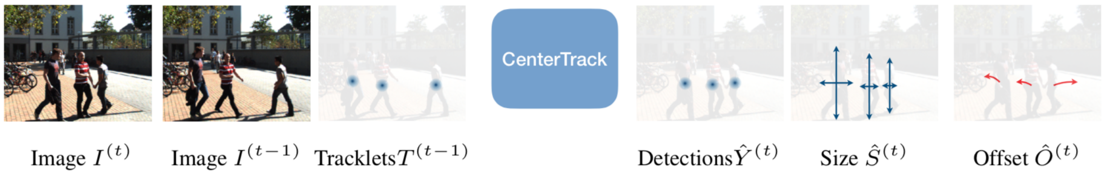


Figure 4.2: CenterTrack regression step. From two consecutive images and a heatmap corresponding to the previous detections, the centroids, widths and heights of the detections in the next time step are computed. Also, the offset between each detection from $t-1$ to t is outputted, and used in a posterior assignment step. Figure from [8].

Tracktor [44] also forms tracks by propagating tracked objects from $t-1$ to t, regressing them using the ROI-pooling layer present in Faster-RCNN [25], as if they were object proposals computed by the network. To improve their results, they make use of re-identification using siamese networks and a motion model based on the Constant Velocity (CV) assumption or Camera Motion Compensation (CMC), depending on whether there is large camera movement in the video sequence or not. In Figure 4.3, the regression step used by Tracktor is depicted.

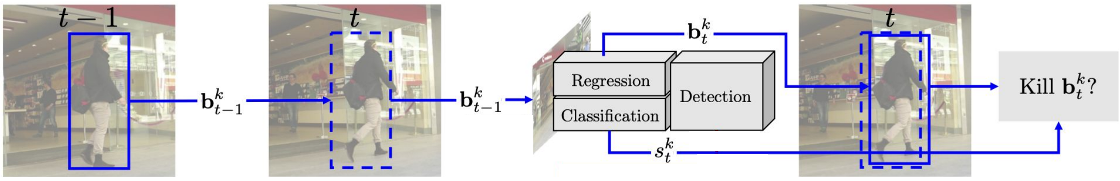


Figure 4.3: Tracktor regression step. The detections from $t-1$ are regressed by means of the ROI-pooling layer of the Faster-RCNN [25] object detector. Figure from [44].

Both methods heavily rely on the positioning of the objects to associate them with the existing tracks projected from $t-1$, leading to the intuition that a better projection of the objects between time instants should straightforwardly boost the trackers performance. Also, with a reliable estimation of the trajectory, the re-identification and occlusion reconstruction problems would be also reduced, as both trackers association mechanism is highly biased towards the detections on the previous frame, making it difficult to associate lost objects further in time (from $t-\tau$) to newly detected objects in t.

4.3.2 Trajectory estimation applied to tracking

An *active* or *lost* state is associated to an object being tracked, depending on whether it is detected and associated to an existing track, or lost due to occlusions, false negatives in the detection phase, or leaving the scene.

When a track is initialized, a patience value is assigned to it. For every frame the track is not associated to any detection (lost state), the patience value of the track is decreased by 1, until it becomes 0, where that object tracking will be considered as terminated. If the object is associated to a track (active state) the patience for that specific track is reset. As TrajE helps the tracker handle occlusions, we set this patience value to 100, allowing objects to reappear in the short-mid term. To add TrajE to a tracker, we define the following common track handle policies.

Active tracks. Given an active track in $t-1$ and its B position hypotheses in t, the active track is associated with a detection in t. The beam corresponding to the trajectory that best fits the detection is chosen, its hidden state copied to the other $B-1$ instances of the trajectory estimator, and its patience reset. The input to the trajectory estimator will be the offset between the new detection and the previous one, and there will be again B possible object mappings in $t+1$.

Lost tracks. If a track in t is not associated to any detection, the track will be considered as lost. From this moment until a detection is associated with this track (using re-identification) in $t+\tau$, the track will be in the lost state. The B sampled hypotheses in $t-1$ are kept, and fed to B new instances of the trajectory estimation network in t. The resulting B^2 hypotheses are pruned to B to avoid exponential growth.

Forwarding the information of lost tracks is important to re-identify the track and recover from occlusions. Also, the object track after an occlusion can be reconstructed if the new detection is coherent with the estimated trajectory.

Recovered tracks. If a lost track since $t-\tau$ is associated to a detection in t, the best beam given the likelihood of the estimated trajectory is associated to that track, the beam search exploration is reset, and a decision is taken whether the trajectory that was being estimated in the lost state (e.g. due to an occlusion) is added to the track or not.

This decision is made regarding the spatio-temporal coherence of the new detection and the estimated trajectory. To compare both, we consider a bounding box with the centroid of the estimated trajectory and the width and height of the last detection in $t-\tau$. If the detection associated to the track in t and the estimated trajectory of the lost object in $t-\tau$ have an Intersection over Union (IoU) above a certain threshold (we use 0.5), we consider that TrajE generated a good trajectory estimation, and keep the trajectory as part of the path that the track has followed. Otherwise, we discard the estimated trajectory during the occlusion, and associate only the new detection to the track. In Figure 4.4 we graphically depict the beam search, track recovery, and occlusion reconstruction for an object.

Terminated tracks. If the track remains a maximum number of time instants (maximum patience) in the lost state, the track will be terminated and removed from the possible re-identification with new detections in the sequence.

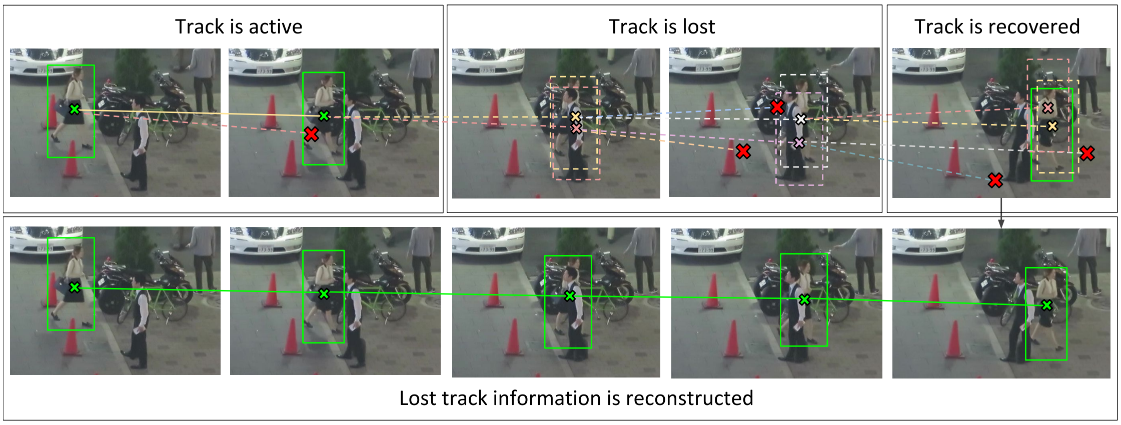


Figure 4.4: Visualization of the beam search and occlusion handling. First, B (for illustration purposes in this case $B=2$) hypotheses of where the object could be in the next time step are sampled from the generated trajectory distribution. If a detection is associated to a track, the beam search corresponding to that track is reset. If no detection is associated to an active track, several hypotheses are sampled from the trajectory distribution and pruned (red crosses) in the next time step to keep B hypotheses. If the track is recovered (a detection is assigned to the track again), and the estimated trajectory is coherent with it, the computed trajectory during the lost state of the track is added to the object track. The occlusion-filling bounding boxes are generated using the centroids of the estimated trajectory, and the width and height of the new detection.

Training. To train TrajE we used the MOT17 [7] challenge data. To generate the training data, we sampled random trajectories from objects in different sequences. These trajectories are split in batches of 100 points in order to have different beginnings and ends of a trajectory. Also, we apply noise to the input sequences to make our model more robust to noisy detections. We end up with 20000 sequences for training and 2000 as validation set. We trained the model for 100 epochs with a learning rate of 0.001, multiplying it by a learning decay of 0.1 at iterations 15, 40, 80.

4.3.3 Integration strategies

We have analyzed three ways to integrate TrajE to an object tracker based on the exploration strategy after estimating the trajectory probability density function. The first one is a Greedy Beam Search (GBS). It takes the local best sample (using maximum likelihood) at every time step given the estimated distribution as the motion of the tracked object. The second one is a Pure Beam Search (PBS) that uses all B hypotheses to forward the track in B possible ways. If a detection is associated to the track, the best beam given the historic is chosen in order to keep a single detection per track at every time step. Note that, for $B = 1$, both strategies become the same.

The last strategy is the Best Mean (BM), which takes the most likely mean of one of the Gaussians forming the probability distribution of the trajectory. Note that the difference between GBS and BM strategies is that GBS samples from the distribution, while BM assumes that the best possible position in $t + 1$ is the mean of the Gaussian with higher likelihood of the mixture of Gaussians. BM ignores the beam search, as all the samples are the same (the most likely mean).

In CenterTrack, we swapped its offset estimation directly with GBS, PBS or BM. Note that CenterTrack's offset estimation matches detections by estimating the objects offset from t to $t - 1$, while TrajE estimates the trajectory of the objects from $t - 1$ to t. For the GBS we swapped the motion module output (offset estimation in this case) with the best sample given by TrajE at every time step. In this case we assume that taking the best local decision per frame will lead to the best possible result. In a similar way, for the BM strategy we take the most likely position of the object by taking the mean of the Gaussian with higher likelihood in the distribution generated by TrajE.

For the PBS, we take the B position hypotheses per object in t, compare them to the detections of the CenterTrack in t, and solve an assignation problem to end up with their closest detection (if any). For all the different strategy cases, the assignment of IDs to the detections follows the same strategy as in CenterTrack, which links the projected object from $t - 1$ to t to its closest detection, in terms of L1 distance, in t.

Tracktor relies on the regression from the Faster-RCNN's *ROI pooling* layer to map the objects from $t - 1$ to t. For the GBS and BM strategies we follow the same strategy as in CenterTrack's case, i.e. swapping the motion estimation by the next most likely sample or mean (for GBS and BM respectively), projecting the object to its next probable position in t.

For the PBS, B hypotheses are projected to the next time step (using the width and height of the projected bounding box), and B regressions per track are made. If several detections are associated to the track, the most probable trajectory associated to a detection (given maximum likelihood) is chosen.

4.4 Experiments

Experiments and ablation studies are performed in the MOTChallenge dataset [7] (pedestrians). To see whether adding TrajE to a tracker generalizes, we also evaluate it against the UA-DETRAC dataset [47] (cars), using the same trajectory estimation model trained on pedestrians from the MOTChallenge training set.

The MOTChallenge dataset contains 14 challenging video sequences recording pedestrians in the wild (7 for training, 7 for testing) with both static and moving camera, recorded in the wild (unconstrained environments), and different locations. The UA-DETRAC dataset consists on 100 videos (60 for training, 40 for testing) with static camera (stable or unstable), recording the road traffic.

Training sequences for MOTChallenge 2017									
Name	FPS	Resolution	Length	Tracks	Boxes	Density	Camera	Viewpoint	Conditions
MOT17-02	30	1920x1080	600 (00:20)	49	17,833	29.7	static	medium	cloudy
MOT17-04	30	1920x1080	1,050 (00:35)	80	47,557	45.3	static	high	night
MOT17-05	14	640x480	837 (01:00)	124	6,818	8.1	moving	medium	sunny
MOT17-09	30	1920x1080	525 (00:18)	25	5,257	10.0	static	low	indoor
MOT17-10	30	1920x1080	654 (00:22)	54	12,318	18.8	moving	medium	night
MOT17-11	30	1920x1080	900 (00:30)	67	9,174	10.2	moving	medium	indoor
MOT17-13	25	1920x1080	750 (00:30)	68	11,450	15.3	moving	high	sunny
Total training			5,316 (03:35)	512	110,407	20.8			

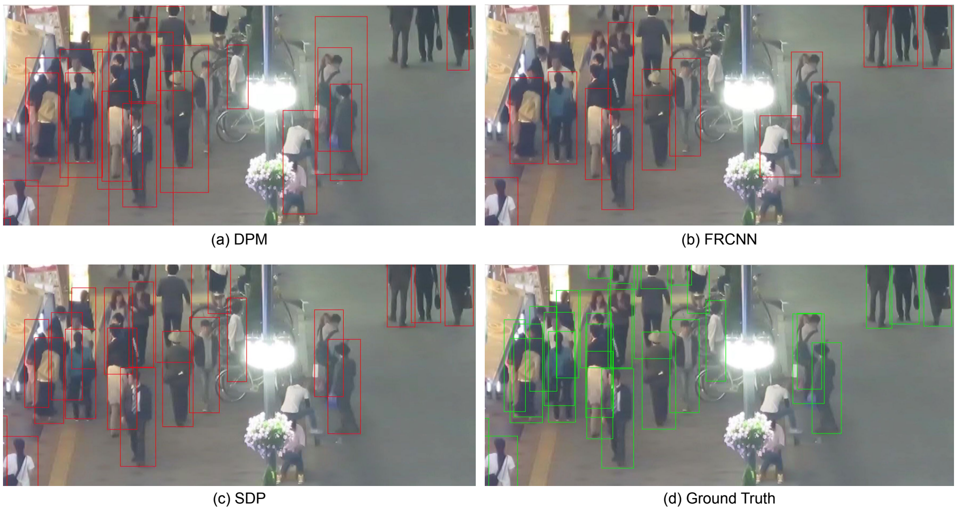
Table 4.1: Overview of the sequences included in the MOTChallenge 2017 training set.

For the MOTChallenge, we use the object detection models provided by the two trackers (and re-identification in case of Tracktor), for the UA-DETRAC dataset we re-trained those models for car detection and re-identification.

Comparing trackers is a difficult task. Both MOTChallenge and UA-DETRAC datasets provide the detections of the objects of interest for the different sequences. These detections are defined as *public detections*, and the idea behind this is that, using the same detections, the trackers would be comparable in terms of tracking itself. In reality, these detections are usually a source of errors (e.g. missing detections, false positives, repeated detections over the same object, the detection size of of an object is not consistent over frames...). In Figure 4.5, samples on the three detectors provided by the MOTChallenge dataset are depicted.

For this, many methods started to refine these detections to do tracking [104], including them in the tracks once modified. Following this refinement idea, several trackers started using their own object detections, defined as *private detections*, using public detections in some way to meet the challenge requisites. Some works do tracking by filtering the public detections based on the private ones, initializing a track only if a public detection is near the private detection, or use the public detections only to start the tracking, relying afterwards on their private detections. Up to now this is an unsolved problem, and the multiple object tracking community is still working on how to do a more fair evaluation between trackers.

In our experiments, we use both trackers in their public detection mode, as MOTChallenge requires the public detections to be used to compare with other state of the art methods. Both trackers extend their tracking algorithm with private detections from the public detection setting. CenterTrack initializes the new tracks if the private detection is



(a) DPM

(b) FRCNN

(c) SDP

(d) Ground Truth

Figure 4.5: Detection samples on the three different detectors that MOTChallenge provides as public detections. The MOTChallenge testing set uses the average result from the tracking method, over the public detections from the three detectors, in order to compare to the other state of the art techniques.

closer than a threshold to a public detection, while Tracktor initializes a track by regressing a public detection, as the first detection associated to a track, using the ROI-pooling layer present in Faster-RCNN. Both use non maximum suppression to filter detections.

4.4.1 Metrics

The purpose of the evaluation metrics is to be able to compare the similarity between the results coming from a method with the ground truth corresponding to the solution. In the case of multiple object tracking, there are many ways of scoring this similarity, and the properties of the metric determine how different errors contribute to a final score.

In the MOTChallenge benchmark, the MOTA [105] and IDF1 [106] scores are the predominant, being MOTA the main metric to rank the trackers. Recently, these metrics have been extensively reviewed in [107], where a new metric called HOTA is proposed.

MOTA. In MOTA (Multi-object Tracking Accuracy) the matching between ground truth and predictions is computed at a detection level. This is, every predicted detection (*prDet*) in the sequence is associated to a ground truth detection (*gtDet*) without repetition. The matching detections, given a threshold and an Intersection Over Union (IOU) value over the bounding boxes, become True Positives (TP). The remaining *prDet* that are not matched with a *gtDet* become False Positives (FP). The *gtDet* that are not matched with a *prDet* become False Negatives (FN).

The association in MOTA is measured using the concept of the Identity Switch (IDSW). An IDSW occurs when a tracker swaps identities between different objects, or when an object is wrongly re-identified (the object ID doesn't correspond to the initially assigned ID).

The MOTA scoring function takes into account three types of tracking errors: the detection errors of FNs, FPs, and the association error IDSW. The total score is calculated by summing these errors, dividing by the number of ground truth detections *gtDets*, and subtracting from one. Therefore, the closer to one, the better.

$$\text{MOTA} = 1 - \frac{|\text{FN}| + |\text{FP}| + |\text{IDSW}|}{|gtDet|} \tag{4.17}$$

IDF1. In IDF1 (IDentification F1 score), the matching between ground truth and predictions is computed on a track level. The mapping between the predicted tracks (*prTrack*) and the ground truth tracks (*gtTrack*) is done by matching overlapping parts of the tracks, given an overlapping threshold. Identity True Positives (IDTP) are defined as the matches on the overlapping part between *prTrack* and *gtTrack*, Identity False Positives (IDFP) are the parts of the *prTrack* that are not matched with any *gtTrack*, and Identity False Negatives (IDFN) are the parts of the *gtTrack* that are not matched with any *prTrack*.

The IDF1 scoring function is derived from the Precision, Recall, and F-score concepts.

$$\text{IDRecall} = \frac{|\text{IDTP}|}{|\text{IDTP}| + |\text{IDFN}|} \tag{4.18}$$

$$\text{IDPrecision} = \frac{|\text{IDTP}|}{|\text{IDTP}| + |\text{IDFP}|} \tag{4.19}$$

$$\text{IDF1} = 2 \cdot \frac{\text{IDPrecision} \cdot \text{IDRecall}}{\text{IDPrecision} + \text{IDRecall}} = \frac{|\text{IDTP}|}{|\text{IDTP}| + \frac{1}{2}(|\text{IDFN}| + |\text{IDFP}|)} \tag{4.20}$$

The IDF1 metric considers trajectories, and computes the best matching sets between *prTrack* and *gtTrack*, allowing only a single best set of matching tracks to be evaluated. This means that any track that is not included in this matching set will be counted as an IDFP, decreasing the score even if it contributes to correct detections and associations.

HOTA. The recently presented HOTA (Higher Order Tracking Accuracy) metric builds upon MOTA, while addressing many of its deficits. The matching between predictions and ground truth occurs at a detection level. As in MOTA, a True Positive (TP) consists on a match between predicted detections (*prDet*) and ground truth detections (*gtDet*). The False Positives (FP) are the remaining *prDet* that are not matched with a *gtDet*, and the False Negatives (FN) are the *gtDet* that are not matched to any *prDet*.

The association measurement in HOTA is made upon the newly introduced concepts of True Positive Associations (TPAs), False Negative Associations (FNAs), and False Positive Associations (FPAs). For a given TP(c) (i.e. a match between *prDet* and *gtDet*), the set of TPAs is the set of k TPs, following c, that have the same ID as the initial TP(c) for the predicted ID (*prID*) and the ground truth ID (*gtID*) sets.

$$\text{TPA}(c) = \{k\}, \quad k \in \{\text{TP} | prID(k) = prID(c) \land gtID(\text{k}) = gtID(\text{c})\} \tag{4.21}$$

For a given TP(c), the set of FPAs is the set of k *prDets* with the same *prID* as c, but that were assigned either a different *gtID* at some point over the set k, or no *gtID* if they did not actually correspond to an object (FP in detection stage).

$$\begin{aligned}&\text{FPA}(c) = \{k\},\\ &\quad k \in \{\text{TP}|\mathit{prID}(k) = \mathit{prID}(c) \wedge \mathit{gtID}(\text{k}) \neq \mathit{gtID}(\text{c})\} \cup \{\text{FP}|\mathit{prID}(k) = \mathit{prID}(c)\}\end{aligned} \tag{4.22}$$

For a given TP(c), the set of FNAs is the set of k *gtDets* with the same *gtID* as c, but that were assigned either a different *prID* at some point over the set k, or no *prID* if they were missed (FN in detection stage).

$$\begin{aligned}&\text{FNA}(c) = \{k\},\\ &\quad k \in \{\text{TP}|\mathit{prID}(k) \neq \mathit{prID}(c) \wedge \mathit{gtID}(\text{k}) = \mathit{gtID}(\text{c})\} \cup \{\text{FN}|\mathit{gtID}(k) = \mathit{gtID}(c)\}\end{aligned} \tag{4.23}$$

The HOTA scoring function combines the previous values (TP, FP, FN, TPA, FPA, FNA) to form its metric for a particular localisation threshold α (IOU between *prDet* and *gtDet*). The main claim in HOTA is that the metric balances the tracking score, both in detection and association terms. Also, HOTA can be separated in order to do an extensive study on the two main factors of the tracking, the detection and the association part. These two main factors are defined as DetA (Detection Accuracy score) and AssA (Association Accuracy score).

$$\text{DetA}_\alpha = \frac{|\text{TP}|}{|\text{TP}| + |\text{FN}| + |\text{FP}|} \tag{4.24}$$

$$\text{AssA}_\alpha = \frac{1}{|\text{TP}|} \sum_{c \in \{\text{TP}\}} \mathcal{A}(c) \tag{4.25}$$

$$\mathcal{A}(c) = \frac{|\text{TPA}(c)|}{|\text{TPA}(c)| + |\text{FNA}(c)| + |\text{FPA}(c)|} \tag{4.26}$$

$$\text{HOTA}_\alpha = \sqrt{\frac{\sum_{c \in \{\text{TP}\}} \mathcal{A}(c)}{|\text{TP}| + |\text{FN}| + |\text{FP}|}} = \sqrt{\text{DetA}_\alpha \cdot \text{AssA}_\alpha} \tag{4.27}$$

We conduct experiments using the three metrics: MOTA and IDF1 for being the metrics currently used in the multiple object tracking community, and HOTA as it gives many useful insights, and we expect it to become the referent metric for the multiple object tracking task. In the case of HOTA, we evaluate for $\alpha = 0.5$.

4.4.2 Experiments

Ablation study. In Table 4.2 and Table 4.3 we present an ablation study on whether the trackers, CenterTrack and Tracktor, benefit from estimating the trajectory of an object.

	MOTA ↑	IDF1 ↑	HOTA ↑	DetA ↑	AssA ↑	IDSW ↓
CenterTrack	67.4	62.7	63.0	69.3	57.2	1356
Centertrack with motion enhancements						
+OFF	67.7	63.8	63.8	69.2	58.8	1077
Trajectory estimation baselines						
+Kalman	68.7	64.4	64.0	70.9	57.7	1642
+BM	68.4	65.6	64.9	69.9	60.2	833
+GBS	68.9 ±0.28	65.6 ±0.47	65.1 ±0.51	70.3 ±0.27	60.2 ±1.06	857
+PBS	69.0 ±0.15	66.0 ±0.08	65.3 ±0.19	70.4 ±0.13	60.5 ±0.42	789
Trajectory estimation with occlusion reconstruction						
+Kalman+occ	68.8	64.6	64.1	71.2	57.8	1382
+BM+occ	69.1	65.9	65.4	70.6	60.5	732
+GBS+occ	69.5 ±0.22	66.0 ±0.47	65.5 ±0.54	70.9 ±0.24	60.6 ±1.13	751
+PBS+occ	69.6 ±0.13	66.3 ±0.08	65.7 ±0.18	71.0 ±0.14	60.9 ±0.43	706

Table 4.2: Ablation study on CenterTrack + TrajE with the three different exploration strategies, GBS, PBS, and BM, and CenterTrack + Kalman filter for trajectory estimation. occ stands for occlusion reconstruction after recovering a lost track, and OFF for offset prediction between two objects in consecutive frames. We evaluate on the Faster R-CNN public detections of MOT17, with $bias = 1$, $B = 5$.

The starting point are two baselines: the trackers without any motion estimation, and the trackers with their own defined motion estimators (see the baselines in the beginning of Section 4.3.1 for details). We then swap the motion estimation of the baselines for TrajE, and study the impact of the three exploration strategies, best mean (BM), Greedy Beam Search (GBS), and Pure Beam Search (PBS). To see whether the benefits of including trajectory information into object trackers generalize, we extended the Kalman filter implemented in SORT [70] to do trajectory estimation by keeping generating trajectories for a track which state is set to lost, while including the new detections assigned to the track whenever its state is set to active, as a baseline for trajectory inclusion. Furthermore, we study the impact of doing occlusion reconstruction (assuming an occlusion to be the interval between the lost state and the recovery of a track), and using camera motion compensation in the case of Tracktor.

Motion estimation. The first thing to notice is that any motion information that can be provided is helpful to the trackers. In Table 4.2, by computing the motion of the object with an offset head for CenterTrack, and Table 4.3, by assuming a constant velocity (CV) of an object for Tracktor, both trackers improve their performance considerably in all metrics. For both trackers, our trajectory estimator TrajE and the Kalman filter extended to trajectories improve the motion estimator baselines. This is due to the fact that both trackers motion estimators only take into account the previous frame (or two frames in case of Tracktor), while the trajectory estimators adapt to all the previous seen trajectory.

Occlusion reconstruction. When doing occlusion reconstruction, i.e. adding the esti-

	MOTA ↑	IDF1 ↑	HOTA ↑	DetA ↑	AssA ↑	IDSW ↓
Tracktor v2	60.6	62.0	60.7	61.6	59.7	913
-ReID	59.7	59.8	59.2	61.6	56.8	1928
Tracktor with motion enhancements						
+CV-ReID	60.8	62.4	60.9	61.9	59.9	1026
+CV	61.2	63.9	62	61.9	62.1	557
+CV+CMC	61.7	64.9	62.8	62.1	63.5	269
Trajectory estimation baselines						
+Kalman	61.4	65.2	63.1	62.0	64.2	462
+BM	61.6	66.4	63.7	62.1	65.3	399
+GBS	61.4 ±0.11	66.6 ±0.54	63.6 ±0.31	62 ±0.08	65.4 ±0.62	407
+PBS	61.5 ±0.04	66.7 ±0.45	63.7 ±0.29	62.1 ±0.06	65.5 ±0.55	369
Trajectory estimation with camera motion compensation						
+Kalman+CMC	61.8	66.2	64.0	62.2	65.8	240
+BM+CMC	61.8	67.4	64.4	62.2	66.6	237
+GBS+CMC	61.8 ±0.05	67.3 ±0.48	64.4 ±0.28	62.2 ±0.07	66.6 ±0.55	231
+PBS+CMC	61.7 ±0.09	67.1 ±0.44	64.2 ±0.25	62.1 ±0.04	66.3 ±0.53	219
Trajectory estimation with occlusion reconstruction						
+Kalman+occ	61.2	66.9	64.4	62.8	66.2	506
+BM+occ	62.2	66.8	64.2	62.8	65.7	399
+GBS+occ	61.9 ±0.11	66.9 ±0.54	64.1 ±0.31	62.5 ±0.07	65.7 ±0.63	409
+PBS+occ	62 ±0.09	67 ±0.43	64.2 ±0.29	62.6 ±0.05	65.8 ±0.59	370
Trajectory estimation with camera motion compensation and occlusion reconstruction						
+Kalman+occ+CMC	62.3	68.3	65.7	63.4	68.1	303
+BM+occ+CMC	62.4	67.7	64.9	62.9	66.9	246
+GBS+occ+CMC	62.4 ±0.13	67.7 ±0.51	64.9 ±0.31	62.9 ±0.09	67 ±0.59	236
+PBS+occ+CMC	62.3 ±0.13	67.5 ±0.49	64.7 ±0.28	62.8 ±0.08	66.6 ±0.6	223

Table 4.3: Ablation study on Tracktor + TrajE with the three different exploration strategies, GBS, PBS, and BM, and Tracktor + Kalman filter for trajectory estimation. occ stands for occlusion reconstruction after recovering a lost track, CV for Constant Velocity assumption, and CMC for Camera Motion Compensation as used in Tracktor. We evaluate on the Faster R-CNN public detections of MOT17 with $bias = 1$, and $B = 5$.

mated trajectories while an occlusion occurs, both trackers improve in all metrics, as a result of reducing much more the false negatives rather than introducing false positives during the tracking. For a more fine-grained discussion, we refer to the comparison of Table 4.6 vs Table 4.7 for CenterTrack with or without occlusion reconstruction, and Table 4.8 vs Table 4.10 for Tracktor. For all sequences, both Precision (DetPr, AssPr) and Recall (DetRe, AssRe) for detection (DetA) and association (AssA) metrics are consistently improving the Recall by reducing the amount of false negatives when including the occluded trajectory to a track, with detriment to the Precision by introducing new false positives due to erroneous occlusion reconstruction, usually by misconnecting a track with another. In Figure 4.6 and Figure 4.7, we depict two visual examples considering the pros and cons of using occlusion reconstruction.

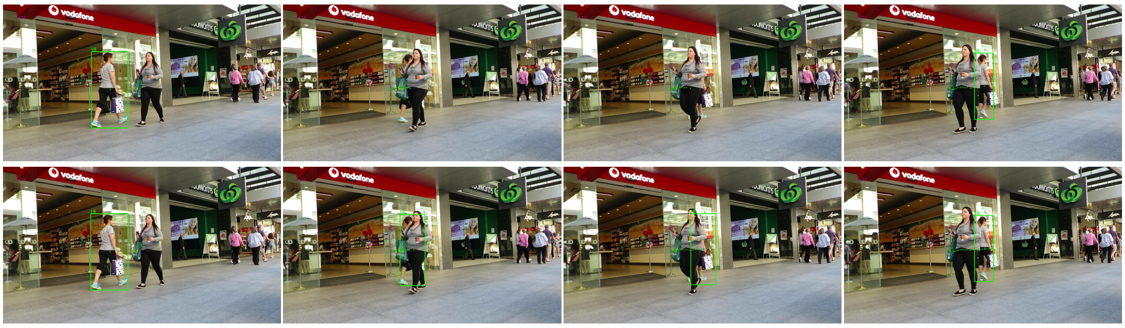


Figure 4.6: Occlusion reconstruction (occ) pros. For a visual comparison, bottom sequence uses occlusion reconstruction, while top sequence does not. By adding the occluded trajectory to the track, the amount of false negatives with respect to the ground truth is reduced. Considering that there are many occluded elements in a sequence, this can be key to make the tracker performance improve.

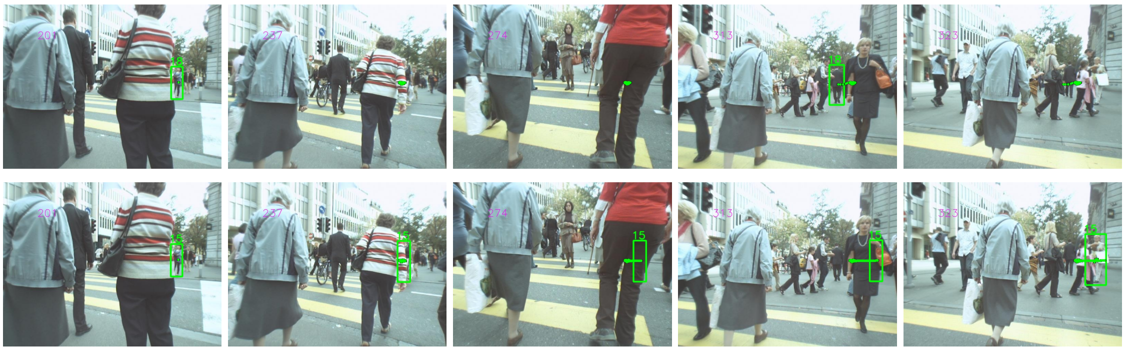


Figure 4.7: Occlusion reconstruction (occ) cons. For a visual comparison, bottom sequence uses occlusion reconstruction, while top sequence does not. By misconnecting two tracks, the occlusion reconstruction becomes a set of false positives.

Camera Motion Compensation. In Table 4.3, we observe a huge boost coming from the camera motion compensation (CMC), used in Tracktor. Logically, this boost comes from moving camera videos, where the camera motion estimation helps to better position the objects in the scene by translating them following the camera movement from $t-1$ to t. We can see this effect by comparing Figure 4.16 vs Figure 4.17, with the sequences with camera movement (MOT17-05, MOT17-10, MOT17-11, MOT17-13). In Figure 4.8, we show a visual example of how applying CMC can lead to better results in the sequence with camera movement MOT17-10.

It is interesting to observe in Figure 4.18 vs Figure 4.19 that, in some scenarios (MOT-05, MOT-09), when combining occlusion reconstruction (occ) and CMC, there is a decrease in performance. This is related to the type of CMC being used in Tracktor (Enhanced Correlation Coefficient maximization [108]), which maximises the correlation between a template image (in this case, the image in time $t-1$), and an incoming image in time t. During an occlusion, the object trajectory keeps being estimated, and the CMC applied to the estimated trajectory at every time instant. If the occluding object is big enough with respect to the image size, it can dominate the CMC at the occluded object position, displacing the estimated trajectory away from the object being occluded. In Figure 4.9 we depict a visual example on how CMC can mislead the trajectory estimation.

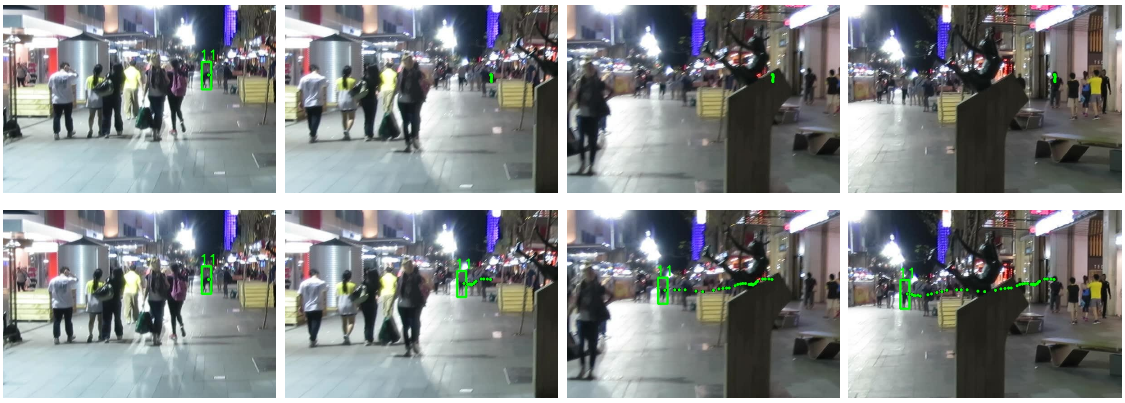


Figure 4.8: Visual example on how the camera motion compensation (CMC) helps in the matching of objects in consecutive images. Top sequence considers the trajectory estimation taking into account the CMC, bottom sequence does only take into account trajectory estimation.

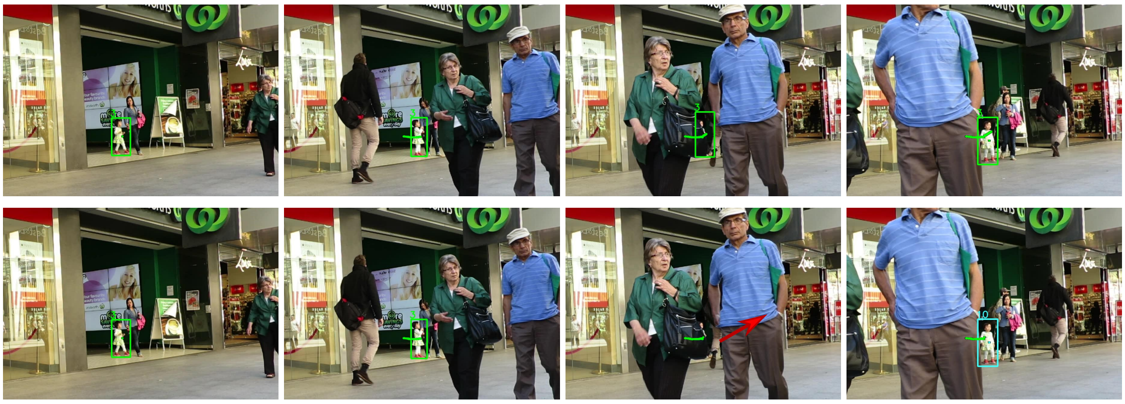


Figure 4.9: Visual example on how the camera motion compensation (CMC) can interfere with occlusion reconstruction (occ). Top sequence considers trajectory estimation + occ, bottom sequence considers trajectory estimation + occ + CMC. Red arrow in the bottom sequence symbolizes the CMC in the depicted place for that specific instant.

Trajectory estimation. Focusing on the trajectory estimation, we see in Table 4.2 and Table 4.3 that using the trajectory estimation information in a multiple object tracker can be a key factor for improving its performance. Both trajectory estimators, TrajE with its exploration configurations and Kalman filter extended to trajectory estimation, increase at least 2 points in HOTA score for both CenterTrack and Tracktor, around 2.5

points in IDF1 score, and around 2 and around 0.6 points of MOTA score for CenterTrack and Tracktor respectively.

Both trackers benefit from the exploration strategies in TrajE (BM, GBS, PBS), with an increment in performance by using beam search in the GBS and PBS strategies. As GBS and PBS strategies have a random sampling step that affects the trajectory hypothesis generation, we did five runs per ablation, ending up with a mean and a variance per strategy.

With respect to the Kalman filter extended to trajectories, TrajE better adapts even if the sequence has camera movement (MOT17-10, MOT-13 in Figure 4.14 and Figure 4.15), while Kalman excels in static camera sequences (MOT-09 in Figure 4.18). We believe this is due to the fact that the Kalman filter model fits very well to simple human trajectories. For this, when the CMC is used (in the case of Tracktor) to compensate the camera movement, the Kalman filter increments considerably its capabilities, as it can focus on the pedestrian trajectory, rather than having to adapt to the camera motion component. Also, the Kalman filter model implemented in SORT [70] has its state modeled as a bounding box, rather than only its centroid as in TrajE, helping the Kalman filter to become more robust to noisy inputs.

CenterTrack vs Tracktor. It is interesting to compare CenterTrack and Tracktor using the two separable metrics from HOTA, corresponding to detection (DetA), and association (AssA). We see in Table 4.2 and Table 4.3 that CenterTrack has a much higher DetA score than Tracktor for all the blocks in the ablation study, but Tracktor has its counterpart on the AssA metric. The DetA score corresponds to the detected objects associated to the ground truth, this is, it measures how well the object detector associated to the tracker performs. Including TrajE into the trackers makes DetA and AssA improve, both for providing new detection priors, and for better connecting the detections among a track. In Figure 4.10 we show a visual example of CenterTrack detections vs Tracktor detections.

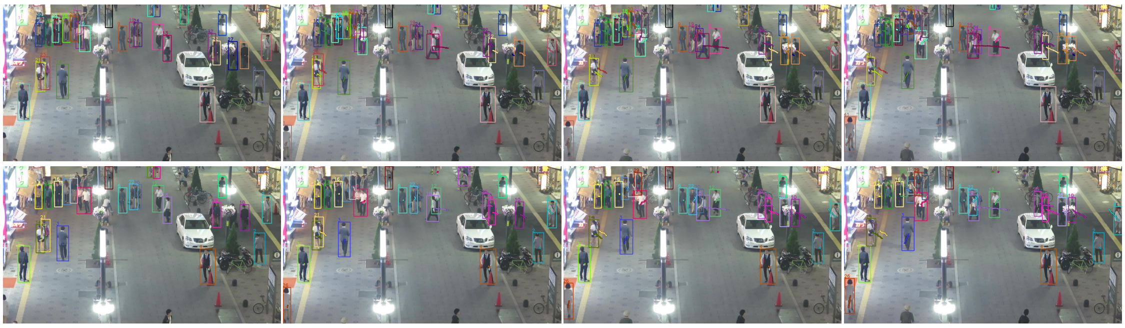


Figure 4.10: Visual example considering the DetA score between trackers. Top corresponds to the active tracks in CenterTrack, bottom corresponds to the active tracks in Tracktor. The difference lies in how good the object detector from CenterTrack is, with respect to Tracktor's one.

On the other hand, Tracktor dominates in the AssA metric, which corresponds to how long an object kept its identity over a period of time. In this case, the way regression of the bounding boxes is made in both trackers (CenterTrack looks for the closest centroid while Tracktor regresses from the previous bounding box), and the fact that Tracktor uses re-identification features (see how not using re-identification in Table 4.3 affects to the AssA score), makes Tracktor dominate in the association. As an example, we show in Figure 4.11 how Tracktor keeps the identity of a tracked object, how it performs without re-identification, and how CenterTrack mixes two tracks due to the association being made using the bounding box centroids.

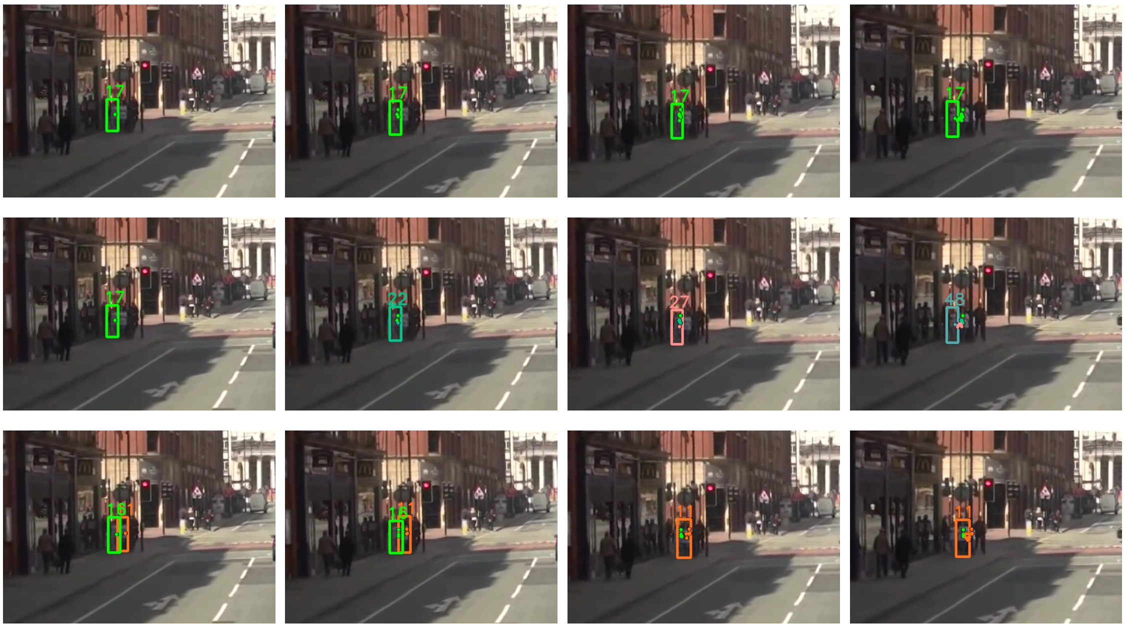


Figure 4.11: Visual example considering the AssA score between trackers. Top corresponds to the tracking of Tracktor + ReID, middle to Tracktor without ReID, and bottom corresponds to CenterTrack, where two tracks are swapped due to the association by means of the distance between centroids. Same color correspond to same track.

TrajE parameters. In Figure 4.12 and Figure 4.13 we study the impact of the bias and beam width (B) values for the different trajectory estimation strategies. We show the behavior of the two parameters over the different exploration strategies, Best Mean (BM), Greedy Beam Search (GBS), and Pure Beam Search (PBS), and put them in comparison with the Kalman filter extended to trajectory estimation, and the trackers baselines (using their own motion estimators). In the trajectory estimation strategies, the occlusion reconstruction is used. The columns of the figures represent the beam width for values $B = 1, 3, 5, 10$. The y axis represents the score of each metric, and the x axis represents the different bias values $b = 0, 0.1, 0.5, 1, 5, 10$. Due to the stochastic nature of TrajE, for the GBS and PBS exploration strategies we did five runs for every configuration pair (bias and beam width). We show the maximum and minimum values that TrajE can achieve per exploration strategy. Note that for $B = 1$, GBS and PBS strategies become the same.

It is difficult to jump to conclusions over the amount of beams used to do the beam search for trajectory hypothesis exploration. It is clear from applying low bias to the trajectory distribution ($bias = 0$, $bias = 0.1$, $bias = 0.5$) that using beam search with $B > 1$ helps to explore more possibilities, leading to better results, but it is not straightforward to see whether using a beam width $B > 3$ (for low biased distributions) helps to find better trajectories. We believe this requires further investigation on the decision over how to chose between trajectories coming from the beam search, and increasing amount of runs per experiment.

It is interesting to see how important is to bias the distribution towards more likely results. In the limit ($bias \rightarrow \infty$), we have the BM strategy, which takes the mean of the Gaussian with highest weight in the mixture model. We see that the exploration strategies that use beam search (GBS or PBS), can help the tracker to improve its performance by exploring beyond the most likely solution, but taking into account the average performance increase with respect to the BM exploration strategy (+0.1 for GBS and +0.3 for PBS in CenterTrack + TrajE for HOTA score), and the amount of resources used by the GBS and PBS strategies, coming at a cost of $O(n^2)$ with respect to the beam width, we believe that it might be not worth to use in real world applications.

Comparison with the state of the art. For the state of the art comparison, we use the metrics defined by the MOTChallenge, which are mainly MOTA and IDF1. In Table 4.4 we compare the two trackers (CenterTrack and Tracktor) adding TrajE in its PBS exploration setting with occlusion reconstruction, using $bias = 1$, $B = 5$, with respect to the state of the art of multiple object trackers in the MOTChallenge testing set. By using TrajE to predict trajectories, their performance in both MOTA and IDF1 scores is boosted by a considerable margin, and set a new state of the art results in the case of CenterTrack + TrajE, with an increase of 6.3 points in the MOTA score and 1.8 in IDF1, and improve the performance of Tracktor + TrajE by 0.3 in MOTA score and 2.9 in IDF1 without using camera motion compensation.

To see if the trajectories learned from pedestrians in TrajE generalize, we tested both trackers + TrajE trained with MOTChallenge data against the UA-DETRAC dataset. In Table 4.5 we compare both Tracktor and CenterTrack with or without TrajE in the PBS exploration configuration and occlusion reconstruction, with respect to other state of the art results on the UA-DETRAC dataset. In both cases, using TrajE for trajectory estimation boosts the performance of the tracker, with an increase of 1.1 and 2.2 points in MOTA, and 2.7 and 1.8 in IDF1 score for CenterTrack and Tracktor respectively.

It is important to state that UA-DETRAC dataset does not have fully annotated frames (i.e only some of the cars inside the scene are annotated), and some of the non-annotated cars are inside "ignore zones", where the detections should be disregarded. We modified the MOT evaluation tool to take into account such zones by ignoring the detections whose bounding box centroid is inside the "ignore zones".

4.5 Conclusions

In this part of the document we have presented TrajE, a lightweight trajectory estimator based on mixture density networks and beam search, and used it in two sate of the art multiple object trackers, CenterTrack [8] and Tracktor [44]. We have also proposed an occlusion reconstruction based on the estimated trajectory. We have tested our technique in the multiple object tracking datasets MOTChallenge [7], which focuses on tracking pedestrians, and UA-DETRAC [47], which focuses on tracking cars in traffic.

The experiments have showed that using the trajectory information can be a key factor for the multiple object tracking. To proof the validity of our method, we also extended the Kalman filter to estimate trajectories, and used it in the aforementioned trackers, making their performance significantly improve.

By adding TrajE to the trackers in the MOTChallenge testing set, there is an increase

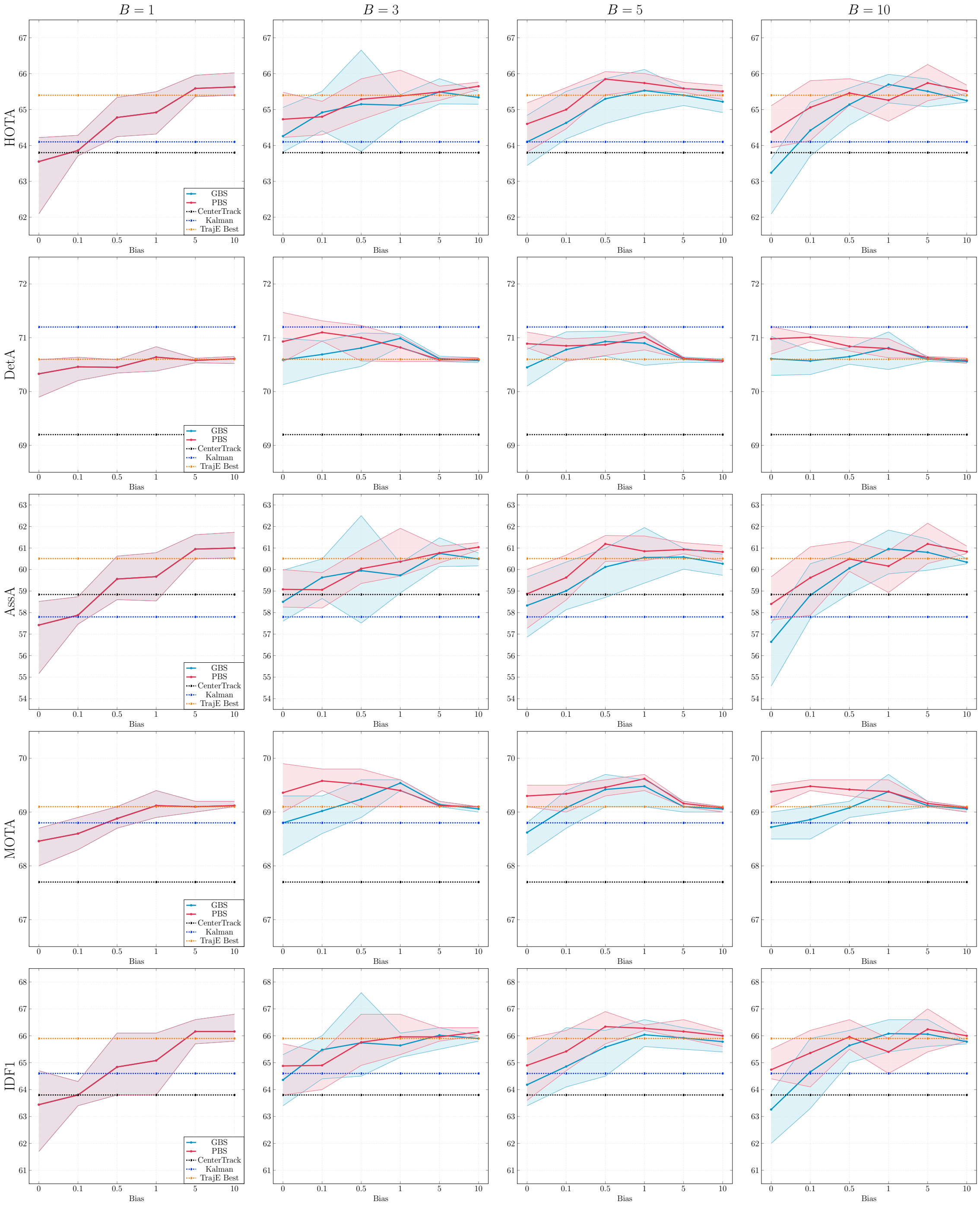


Figure 4.12: Experiments on biasing the trajectory estimation distribution (*bias*), and modifying the beam width (B) over different values ($bias = (0, 0.1, 0.5, 1, 5, 10)$, $B = (1, 3, 5, 10)$) for CenterTrack using TrajE with occlusion reconstruction for HOTA, DetA, AssA, MOTA, and IDF1 metrics. The y-axis correspond to the metric score, the x-axis to the *bias*, and the columns correspond to the different B values. Blue and red solid lines correspond to the average value, given five runs per ($bias$, B) pair, for GBS and PBS techniques respectively. The more transparent region limits correspond to the maximum and minimum metric scores for the five runs. In dashed lines, black stands for CenterTrack with offset estimation for motion, dark blue for Kalman with occlusion reconstruction, and orange for BM with occlusion reconstruction.

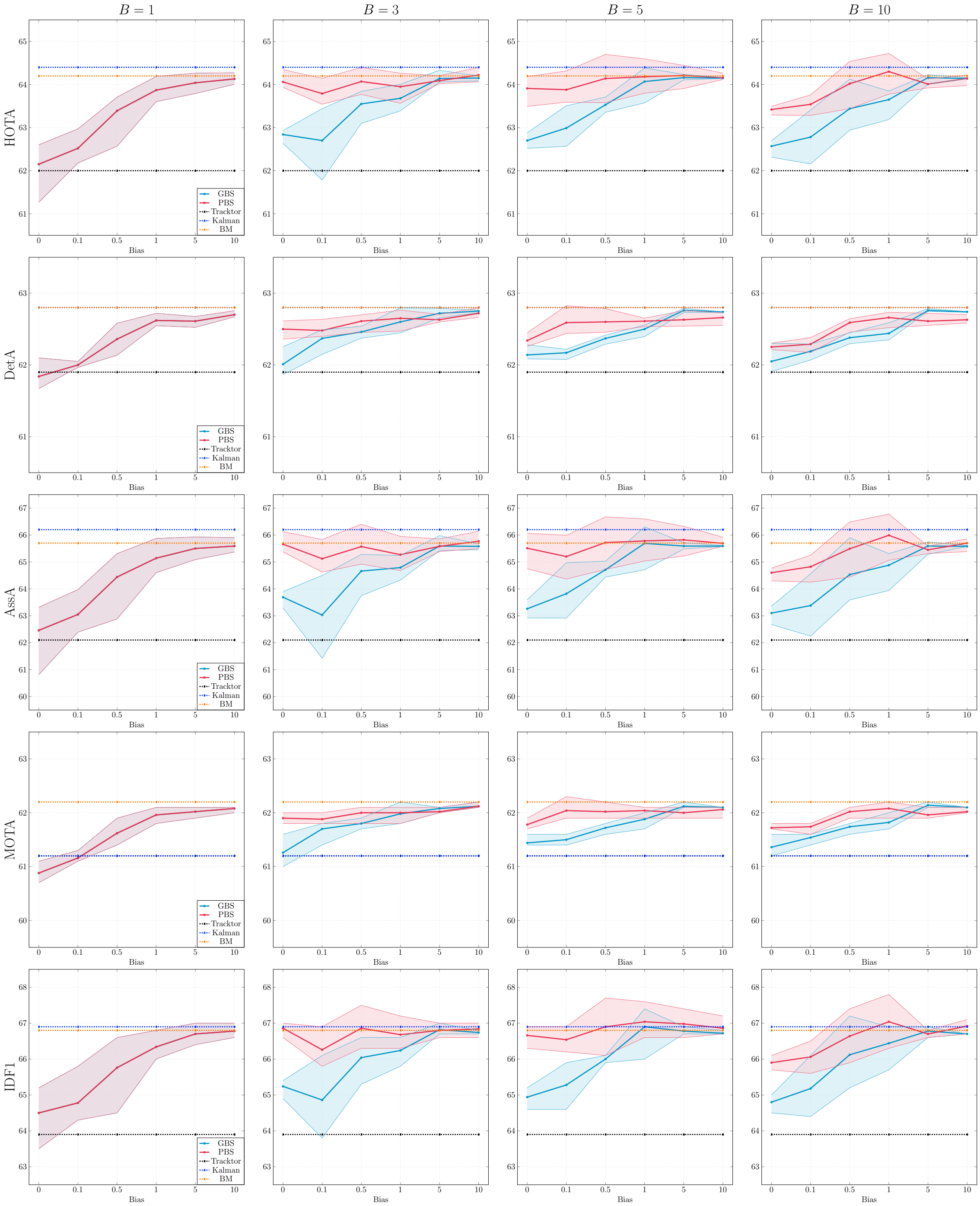


Figure 4.13: Experiments on biasing the trajectory estimation distribution (*bias*), and modifying the beam width (B) over different values ($bias = (0, 0.1, 0.5, 1, 5, 10)$, $B = (1, 3, 5, 10)$) for Tracktor using TrajE with occlusion reconstruction for HOTA, DetA, AssA, MOTA, and IDF1 metrics. The y-axis correspond to the metric score, the x-axis to the *bias*, and the columns correspond to the different B values. Blue and red solid lines correspond to the average value, given five runs per (*bias*, B) pair, for GBS and PBS techniques respectively. The more transparent region limits correspond to the maximum and minimum metric scores for the five runs. In dashed lines, black stands for Tracktor with constant velocity and camera motion estimation for motion, dark blue for Kalman with occlusion reconstruction, and orange for BM with occlusion reconstruction.

MOT17 [7]						
Method	MOTA ↑	IDF1 ↑	MT % ↑	ML % ↓	FP ↓	FN ↓
CenterTrack + TrajE	**67.8**	61.4	**36.0**	**24.5**	20982	**157468**
CenterTrack [8]	61.5	59.6	26.4	31.9	14076	200672
Lif_T [60]	60.5	**65.6**	27.0	33.6	14966	206619
MPNTrack [61]	58.8	61.7	28.8	33.5	17413	213594
Tracktor(v2) + TrajE	56.6	58.2	21.9	35.7	10119	231091
GSM [78]	56.4	57.8	22.2	34.5	14379	230174
Tracktor(v2)[44]	56.3	55.1	21.1	35.3	**8866**	235449
TT17 [62]	54.9	63.1	24.4	38.1	20236	233295
TPM [63]	54.2	52.6	22.8	37.5	13739	242730
JBNOT [109]	52.6	50.8	19.7	35.8	31572	232659
FAMNet [65]	52.0	48.7	19.1	33.4	14138	253616
ETC [110]	51.9	58.1	23.1	35.5	36164	232783
eHAF[59]	51.8	54.7	23.4	37.9	33212	236772
AFN [111]	51.5	46.9	20.6	35.5	22391	248420
NOTA [54]	51.3	54.7	17.1	35.4	20148	252531
FWT [58]	51.3	47.6	21.4	35.2	24101	247921
jCC [76]	51.2	54.5	20.9	37.0	25937	247822
STRN [75]	50.9	56.0	18.9	33.8	25295	249365

Table 4.4: Comparison integrating *TrajE* to CenterTrack and Tracktor against the state of the art methods on the MOTChallenge 17 testing set using public detections. In bold tracker + our method (TrajE). In red the best result, in blue the second best. In MOTChallenge, MOTA and IDF1 are the metrics used to evaluate the tracking performance, being HOTA not yet integrated to the MOTChallenge evaluation for the testing set. Table adapted from [7]

UA-DETRAC [47]						
Method	MOTA ↑	IDF1 ↑	MT % ↑	ML % ↓	FP ↓	FN ↓
CenterTrack + TrajE	**72.0**	**80.8**	**72.5**	7.4	44415	**130911**
CenterTrack [8]	69.9	78.1	71.5	8.5	46657	141356
Tracktor(v2) + TrajE	69.2	76.1	62.2	9.3	26236	181029
POI [112]	69.2	-	67.4	5.2	-	-
WD [113]	68.5	-	70.2	**3.1**	-	-
Tracktor(v2) [44]	67.7	74.3	60.0	9.9	**23422**	193526
DeepSORT [71]	65.4	-	65.9	4.7	-	-
RMOT [114]	62.6	-	42.1	6.5	-	-
IHTLS [115]	62.6	-	63.4	3.8	-	-

Table 4.5: Comparison integrating *TrajE* to CenterTrack and Tracktor with state-of-the art methods on the UA-DETRAC dataset using CompACT [116] public detections. In bold tracker + our method (TrajE). In red the best result, in blue the second best.

of 6.3, 0.3 points in the MOTA score, and 1.8, 2.9 in IDF1 respectively for CenterTrack and Tracktor, and, for the UA-DETRAC testing set, an increase of 0.5, 2.2 points in the MOTA score, and 0.2, 1.8 in IDF1 respectively for CenterTrack and Tracktor. In the case of CenterTrack + TrajE, the algorithm became the new state of the art for multiple object tracking in the MOTChallenge dataset.

Future work will be directly related to improve the trajectory model for better exploiting existing track information. This can be done by estimating the evolution of the bounding boxes of the tracked objects instead of using a single point (centroid of the bounding box), improving the backbone model by specifically focusing on previous trajectory information with self-attention mechanisms by using transformers, making TrajE aware of other dynamic agents in the scene, or using visual information, in the form of feature vectors from the tracked objects, to better match existing tracks.

Ethical concerns. Tools are neutral, it is the action that provide the moral implication to them. We believe that multiple object tracking and trajectory estimation can bring much good to the society (robotics, automatic broadcasting, person flux optimization, counter-terrorism...), but wrongly applied can become very scary (population control or offensive military purposes). As there seems to be no turning back on the development of these algorithms, we believe that the research communities should shift the interest of the usage of these methods, starting by changing the datasets used (the reference dataset for MOT, MOTChallenge, focuses on people surveillance), and building new ones to make these tools to better serve humanity.

4.6 Experiments annex. Analysis per sequence

In this section we include the figures and tables corresponding to each sequence and ablation study block for the MOTChallenge training set.

For formatting purposes, this page is intentionally left blank.

Sequence	HOTA	DetA	AssA	DetPr	DetRe	AssPr	AssRe	HOTA_TP	HOTA_FP	HOTA_FN
CenterTrack + OFF										
MOT17-02	39.4	44.7	34.7	97.8	45.2	94.0	35.3	8398	187	10183
MOT17-04	75.9	78.9	73.0	99.7	79.1	96.0	74.3	37626	122	9931
MOT17-05	49.5	59.8	40.9	91.3	63.5	85.7	44.1	4390	420	2527
MOT17-09	58.3	70.3	48.4	99.4	70.6	94.2	50.9	3759	22	1566
MOT17-10	54.2	69.9	42.1	95.1	72.5	88.8	44.8	9310	478	3529
MOT17-11	64.7	71.6	58.5	98.1	72.6	96.5	59.9	6853	136	2583
MOT17-13	61.4	71.7	52.6	93.7	75.3	88.3	55.1	8772	587	2870
OVERALL	63.8	69.2	58.8	97.6	70.4	93.5	60.5	79108	1952	33189
CenterTrack + Kalman										
MOT17-02	40.2	45.1	35.9	96.9	45.7	77.3	38.6	8495	273	10086
MOT17-04	78.9	82.5	75.4	99.6	82.7	86.2	81.2	39351	155	8206
MOT17-05	45.7	60.3	34.6	91.1	64.1	62.1	43.9	4433	432	2484
MOT17-09	58.2	70.3	48.1	99.4	70.6	78.4	52.9	3761	22	1564
MOT17-10	48.7	70.0	33.8	95.0	72.7	66.5	39.5	9336	496	3503
MOT17-11	61.2	72.1	52.0	98.0	73.1	80.0	59.9	6899	138	2537
MOT17-13	57.1	71.8	45.4	92.8	76.1	65.4	55.1	8856	685	2786
OVERALL	64.0	70.9	57.7	97.4	72.2	78.5	63.9	81131	2201	31166
CenterTrack + BM										
MOT17-02	40.1	45.1	35.7	97.0	45.7	74.3	39.1	8493	264	10088
MOT17-04	78.7	80.0	77.3	99.6	80.3	88.5	82.0	38190	156	9367
MOT17-05	46.2	60.7	35.1	91.1	64.5	52.8	52.2	4458	433	2459
MOT17-09	55.0	70.3	43.0	99.4	70.6	64.5	53.6	3759	22	1566
MOT17-10	54.7	70.1	42.8	94.9	72.8	71.8	49.4	9347	502	3492
MOT17-11	65.2	71.8	59.1	98.0	72.8	81.7	67.1	6873	137	2563
MOT17-13	59.5	72.0	49.3	93.0	76.1	68.6	60.7	8863	672	2779
OVERALL	64.9	69.9	60.2	97.3	71.2	79.2	67.0	79983	2186	32314
CenterTrack + GBS										
MOT17-02	42.4	45.1	39.9	97.0	45.7	78.1	43.4	8491	264	10090
MOT17-04	78.0	81.1	75.0	99.6	81.4	87.2	80.7	38707	159	8850
MOT17-05	46.3	60.4	35.5	91.1	64.2	56.6	51.0	4442	433	2475
MOT17-09	54.2	70.4	41.8	99.4	70.7	62.9	54.0	3763	22	1562
MOT17-10	55.1	70.0	43.4	95.0	72.7	73.9	49.9	9338	495	3501
MOT17-11	66.4	71.8	61.4	98.0	72.8	86.3	67.5	6874	137	2562
MOT17-13	61.1	72.0	51.8	93.1	76.1	72.7	60.8	8858	658	2784
OVERALL	65.0	70.3	60.2	97.4	71.7	80.2	67.0	80473	2169	31824
CenterTrack + PBS										
MOT17-02	43.6	45.0	42.3	97.0	45.7	79.6	45.3	8487	263	10094
MOT17-04	77.9	81.3	74.6	99.6	81.6	86.6	80.4	38809	161	8748
MOT17-05	48.4	60.5	38.7	91.1	64.3	58.8	52.9	4449	433	2468
MOT17-09	54.7	70.3	42.6	99.4	70.6	64.3	56.5	3759	22	1566
MOT17-10	55.2	70.0	43.6	95.0	72.7	73.5	49.9	9339	495	3500
MOT17-11	65.6	71.8	59.9	98.0	72.9	85.1	66.4	6878	137	2558
MOT17-13	62.1	72.0	53.5	93.0	76.1	73.6	62.4	8865	663	2777
OVERALL	65.3	70.4	60.5	97.4	71.8	80.2	67.4	80586	2174	31711

Table 4.6: Ablation study per sequence for CenterTrack. For GBS and PBS parameters are set to $bias = 1$, $B = 5$, and the results are the mean of five different runs.

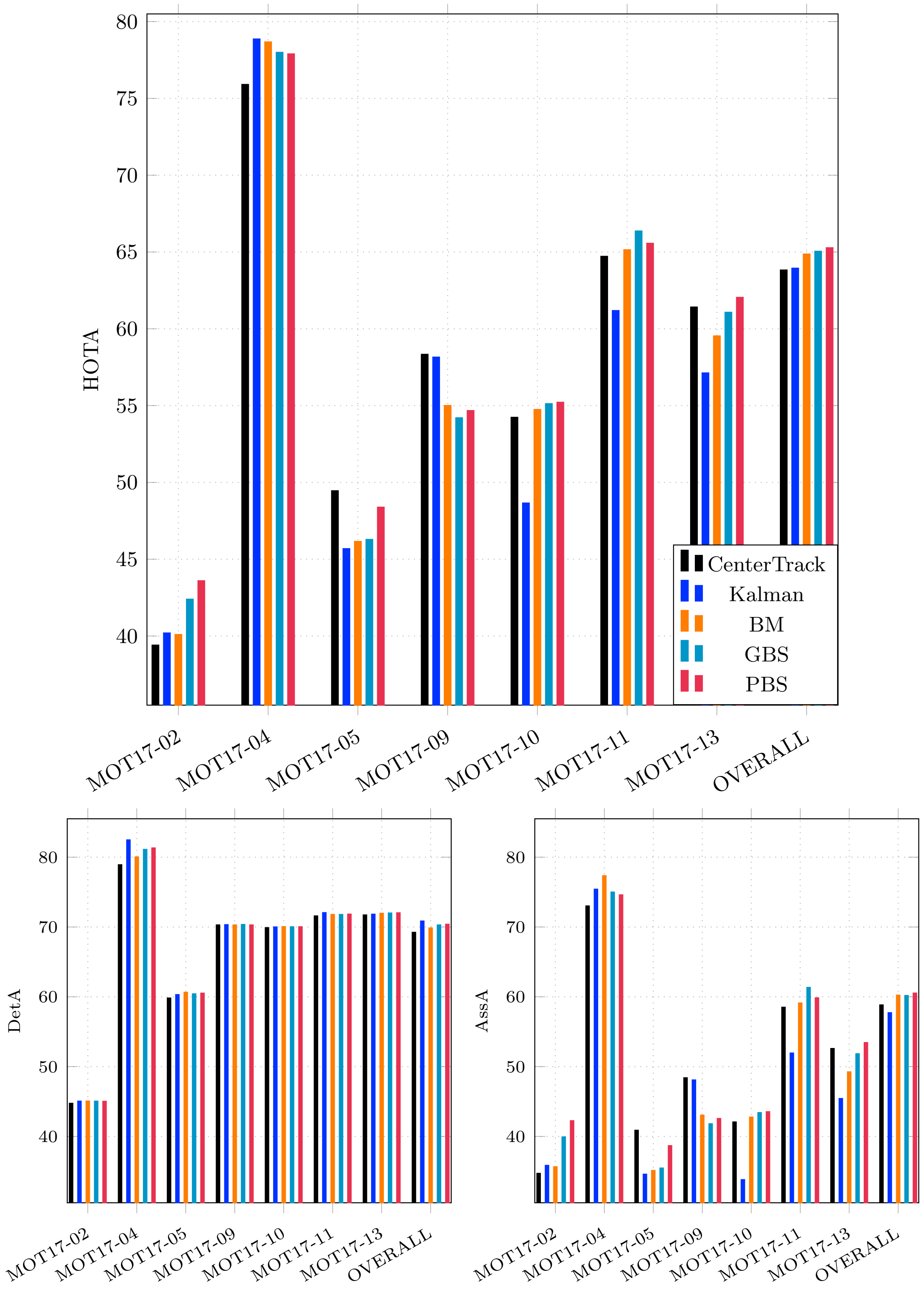


Figure 4.14: CenterTrack + TrajE per sequence.

Sequence	HOTA	DetA	AssA	DetPr	DetRe	AssPr	AssRe	HOTA_TP	HOTA_FP	HOTA_FN
CenterTrack + OFF										
MOT17-02	39.4	44.7	34.7	97.8	45.2	94.0	35.3	8398	187	10183
MOT17-04	75.9	78.9	73.0	99.7	79.1	96.0	74.3	37626	122	9931
MOT17-05	49.5	59.8	40.9	91.3	63.5	85.7	44.1	4390	420	2527
MOT17-09	58.3	70.3	48.4	99.4	70.6	94.2	50.9	3759	22	1566
MOT17-10	54.2	69.9	42.1	95.1	72.5	88.8	44.8	9310	478	3529
MOT17-11	64.7	71.6	58.5	98.1	72.6	96.5	59.9	6853	136	2583
MOT17-13	61.4	71.7	52.6	93.7	75.3	88.3	55.1	8772	587	2870
OVERALL	63.8	69.2	58.8	97.6	70.4	93.5	60.5	79108	1952	33189
CenterTrack + Kalman + occ										
MOT17-02	41.3	47.1	36.3	93.8	48.6	74.7	40.0	9032	595	9549
MOT17-04	79.4	83.5	75.6	99.4	83.9	85.9	81.6	39898	247	7659
MOT17-05	45.8	58.8	35.6	85.6	65.2	61.5	45.5	4510	757	2407
MOT17-09	58.4	71.5	47.7	98.2	72.5	76.3	53.0	3861	72	1464
MOT17-10	48.5	69.2	33.9	91.4	74.0	65.0	40.1	9504	891	3335
MOT17-11	61.2	72.3	51.9	96.7	74.1	79.1	60.1	6989	237	2447
MOT17-13	56.4	70.1	45.4	87.9	77.5	64.1	56.0	9028	1240	2614
OVERALL	64.1	71.2	57.8	95.3	73.8	77.5	64.4	82822	4039	29475
CenterTrack + BM + occ										
MOT17-02	41.3	46.4	36.7	96.3	47.3	73.7	40.4	8782	340	9799
MOT17-04	79.1	80.7	77.6	99.5	81.0	88.5	82.3	38510	182	9047
MOT17-05	46.6	60.8	35.7	89.8	65.3	52.8	53.3	4520	514	2397
MOT17-09	56.1	72.8	43.3	99.1	73.3	63.4	54.6	3901	35	1424
MOT17-10	55.6	70.9	43.6	94.2	74.1	71.4	50.5	9515	587	3324
MOT17-11	65.5	71.9	59.5	97.4	73.3	81.5	67.7	6920	183	2516
MOT17-13	59.7	72.2	49.3	91.9	77.1	68.1	61.2	8980	788	2662
OVERALL	65.4	70.6	60.5	96.9	72.2	78.8	67.5	81128	2629	31169
CenterTrack + GBS + occ										
MOT17-02	43.7	46.4	41.2	96.4	47.2	78.0	44.9	8771	331	9810
MOT17-04	78.4	81.7	75.3	99.5	82.0	87.1	81.0	39010	207	8547
MOT17-05	46.8	60.5	36.3	89.6	65.0	56.8	52.4	4498	522	2419
MOT17-09	55.1	71.7	42.4	99.3	72.0	62.7	55.0	3836	26	1489
MOT17-10	55.4	70.5	43.6	94.4	73.6	73.3	50.3	9447	564	3392
MOT17-11	66.9	72.2	61.9	97.9	73.3	86.3	68.2	6919	149	2517
MOT17-13	61.4	72.2	52.2	92.3	76.8	72.3	61.4	8946	750	2696
OVERALL	65.5	70.9	60.6	97.0	72.5	80.0	67.5	81427	2551	30870
CenterTrack + PBS + occ										
MOT17-02	44.8	46.2	43.6	96.3	47.0	79.6	46.7	8733	334	9848
MOT17-04	78.3	81.9	74.8	99.5	82.3	86.4	80.7	39124	194	8433
MOT17-05	48.9	60.6	39.4	89.7	65.1	59.0	54.1	4506	518	2411
MOT17-09	55.9	72.1	43.4	99.2	72.5	64.0	57.7	3861	33	1464
MOT17-10	55.4	70.5	43.6	94.4	73.6	73.0	50.2	9444	561	3395
MOT17-11	66.1	72.3	60.4	97.9	73.4	85.1	67.1	6926	147	2510
MOT17-13	62.2	72.2	53.6	92.1	77.0	73.1	63.0	8963	767	2679
OVERALL	65.7	71.0	60.9	97.0	72.6	80.0	67.9	81557	2555	30740

Table 4.7: Ablation study per sequence for CenterTrack + occlusion reconstruction. For GBS and PBS parameters are set to $bias = 1$, $B = 5$, and the results are the mean of five different runs.

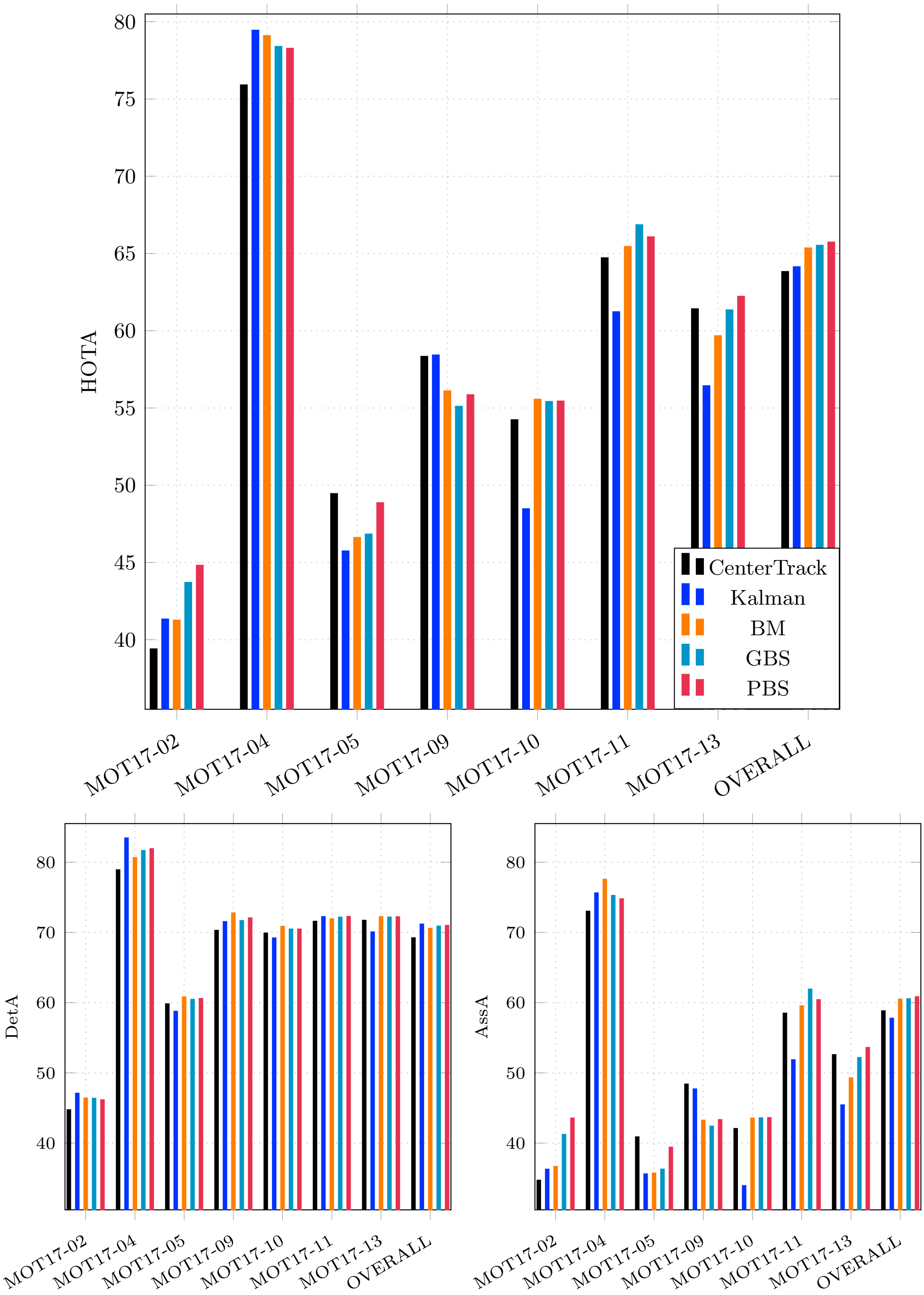


Figure 4.15: CenterTrack + TrajE + occ per sequence.

Sequence	HOTA	DetA	AssA	DetPr	DetRe	AssPr	AssRe	HOTA_TP	HOTA_FP	HOTA_FN
Tracktor + CV										
MOT17-02	41.5	41.1	41.9	99.7	41.1	95.6	42.7	7645	22	10936
MOT17-04	68.7	64.3	73.4	99.8	64.4	96.7	74.3	30631	70	16926
MOT17-05	60.0	56.8	63.3	98.0	57.5	93.7	65.7	3975	80	2942
MOT17-09	51.8	63.0	42.6	99.1	63.3	82.9	53.1	3373	30	1952
MOT17-10	57.2	69.9	46.8	98.0	70.9	87.8	50.6	9104	183	3735
MOT17-11	63.0	68.6	57.8	98.8	69.2	94.8	60.2	6527	82	2909
MOT17-13	69.6	73.1	66.3	99.0	73.7	90.1	71.7	8581	89	3061
OVERALL	62.0	61.9	62.1	99.2	62.2	93.6	64.6	69836	556	42461
Tracktor + Kalman										
MOT17-02	42.5	41.2	43.8	99.8	41.3	86.0	46.5	7665	19	10916
MOT17-04	70.0	64.6	76.0	99.8	64.7	96.1	77.3	30760	72	16797
MOT17-05	51.7	57.0	47.0	98.0	57.7	60.2	69.5	3988	80	2929
MOT17-09	58.9	64.0	54.2	99.1	64.4	79.2	64.4	3427	31	1898
MOT17-10	59.0	69.7	50.0	98.0	70.7	78.1	56.3	9071	181	3768
MOT17-11	68.4	68.5	68.4	98.7	69.1	92.4	71.5	6521	83	2915
MOT17-13	68.1	72.7	63.7	98.9	73.3	83.4	72.6	8538	96	3104
OVERALL	63.1	62.0	64.2	99.2	62.3	87.9	69.0	69970	562	42327
Tracktor + BM										
MOT17-02	43.2	41.2	45.2	99.8	41.2	89.2	47.5	7663	19	10918
MOT17-04	70.7	64.6	77.4	99.8	64.7	95.0	80.1	30760	72	16797
MOT17-05	59.0	56.9	61.3	98.0	57.5	83.9	69.2	3980	81	2937
MOT17-09	54.7	64.0	46.8	99.1	64.4	78.4	60.4	3427	31	1898
MOT17-10	59.5	70.1	50.5	98.1	71.0	85.4	54.0	9120	178	3719
MOT17-11	65.2	68.5	62.0	98.8	69.1	92.6	65.2	6521	81	2915
MOT17-13	70.1	73.5	66.9	98.9	74.1	87.2	74.2	8624	92	3018
OVERALL	63.7	62.1	65.3	99.2	62.4	90.5	69.5	70095	554	42202
Tracktor + GBS										
MOT17-02	43.6	41.2	46.2	99.8	41.2	89.6	48.3	7660	18	10921
MOT17-04	70.3	64.5	76.7	99.8	64.6	96.7	77.9	30709	72	16848
MOT17-05	59.0	56.8	61.3	98.0	57.5	84.8	69.0	3979	82	2938
MOT17-09	54.3	63.6	46.3	99.1	64.0	75.9	59.2	3408	30	1917
MOT17-10	59.6	70.0	50.9	98.0	71.0	84.1	55.3	9113	185	3726
MOT17-11	65.1	68.6	61.9	98.8	69.1	91.1	65.2	6522	78	2914
MOT17-13	70.4	72.8	68.1	98.9	73.4	89.2	74.3	8550	94	3092
OVERALL	63.6	62.0	65.2	99.2	62.3	91.2	68.7	69940	559	42357
Tracktor + PBS										
MOT17-02	43.6	41.2	46.2	99.8	41.2	89.9	48.2	7659	18	10922
MOT17-04	70.6	64.4	77.3	99.8	64.5	96.9	78.5	30686	72	16871
MOT17-05	59.3	56.8	61.9	98.0	57.5	85.5	69.2	3976	82	2941
MOT17-09	54.5	63.8	46.6	99.2	64.1	73.7	59.6	3414	29	1911
MOT17-10	59.0	70.0	49.8	98.0	71.0	85.5	54.0	9113	189	3726
MOT17-11	65.0	68.6	61.7	98.8	69.1	90.7	65.2	6524	78	2912
MOT17-13	70.4	72.8	68.1	99.0	73.4	89.6	74.1	8545	90	3097
OVERALL	63.6	62.0	65.4	99.2	62.3	91.4	68.8	69917	558	42380

Table 4.8: Ablation study per sequence for Tracktor and different techniques for trajectory estimation. For GBS and PBS parameters are set to $bias = 1$, $B = 5$, and the results are the mean of five different runs.

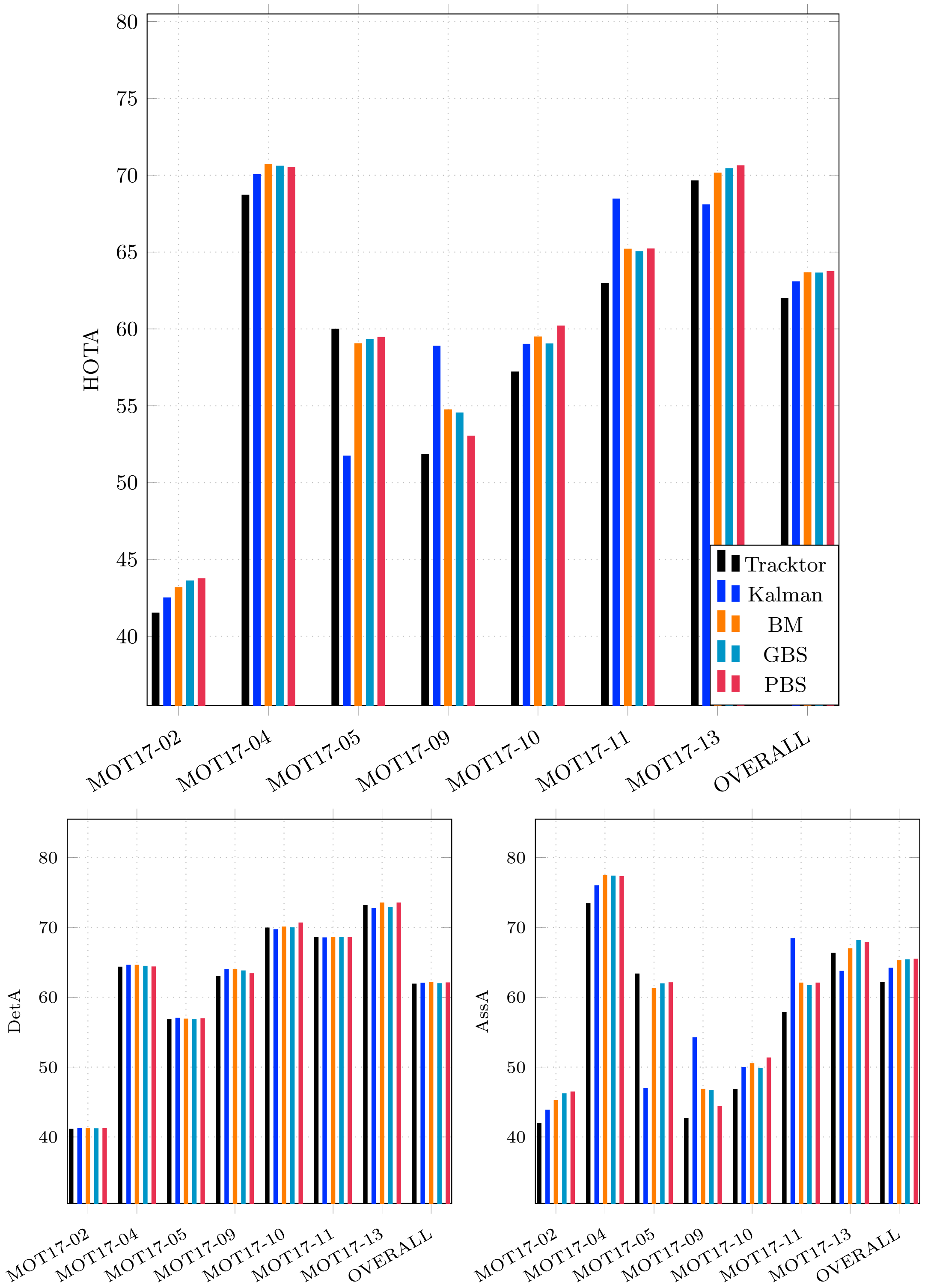


Figure 4.16: Tracktor + TrajE per sequence.

Sequence	HOTA	DetA	AssA	DetPr	DetRe	AssPr	AssRe	HOTA_TP	HOTA_FP	HOTA_FN
Tracktor + CV + CMC										
MOT17-02	41.4	41.1	41.8	99.7	41.1	94.5	42.7	7641	22	10940
MOT17-04	68.7	64.3	73.4	99.8	64.4	96.7	74.3	30631	71	16926
MOT17-05	59.9	56.9	63.1	98.0	57.5	92.3	66.4	3979	80	2938
MOT17-09	51.8	63.0	42.6	99.1	63.3	82.9	53.1	3373	30	1952
MOT17-10	62.4	71.2	54.8	98.3	72.0	93.8	57.2	9248	156	3591
MOT17-11	62.7	68.5	57.4	98.8	69.1	93.5	60.1	6520	78	2916
MOT17-13	71.8	73.9	69.8	98.2	74.9	89.7	75.9	8716	160	2926
OVERALL	62.8	62.1	63.5	99.2	62.4	94.0	66.0	70108	597	42189
Tracktor + Kalman + CMC										
MOT17-02	42.5	41.2	43.8	99.8	41.3	86.0	46.5	7665	19	10916
MOT17-04	70.0	64.6	76.0	99.8	64.7	96.1	77.3	30760	72	16797
MOT17-05	53.4	56.8	50.2	98.0	57.5	64.1	70.6	3977	80	2940
MOT17-09	59.0	64.0	54.4	99.1	64.4	80.8	64.4	3427	31	1898
MOT17-10	63.2	71.2	56.1	98.3	72.1	81.5	62.6	9254	161	3585
MOT17-11	68.5	68.6	68.4	98.8	69.2	92.4	71.6	6529	79	2907
MOT17-13	71.0	73.3	68.8	98.3	74.2	84.5	78.0	8635	146	3007
OVERALL	64.0	62.2	65.7	99.2	62.6	88.7	70.5	70247	588	42050
Tracktor + BM + CMC										
MOT17-02	43.2	41.2	45.2	99.8	41.2	89.2	47.5	7663	19	10918
MOT17-04	70.7	64.6	77.4	99.8	64.7	95.0	80.1	30760	72	16797
MOT17-05	56.8	56.8	56.9	98.1	57.5	81.1	66.4	3974	79	2943
MOT17-09	54.7	64.0	46.8	99.1	64.4	78.4	60.4	3427	31	1898
MOT17-10	64.8	71.2	59.1	98.4	72.0	89.7	62.5	9250	153	3589
MOT17-11	66.3	68.6	64.1	98.8	69.2	93.9	66.6	6528	81	2908
MOT17-13	71.4	73.2	69.7	98.3	74.1	89.4	75.6	8628	146	3014
OVERALL	64.4	62.2	66.6	99.2	62.5	91.3	70.7	70230	581	42067
Tracktor + GBS + CMC										
MOT17-02	43.7	41.2	46.4	99.8	41.2	91.0	48.1	7664	18	10917
MOT17-04	70.5	64.3	77.3	99.8	64.4	97.1	78.3	30645	73	16912
MOT17-05	58.9	56.9	61.0	98.0	57.5	83.8	69.6	3980	82	2937
MOT17-09	52.5	63.4	43.5	99.1	63.8	73.5	58.0	3395	32	1930
MOT17-10	63.5	71.3	56.6	98.4	72.1	91.9	59.7	9258	150	3581
MOT17-11	65.3	68.6	62.1	98.8	69.2	90.0	66.3	6531	81	2905
MOT17-13	71.5	72.9	70.0	97.2	74.5	88.7	76.8	8678	254	2964
OVERALL	64.2	62.1	66.3	99.0	62.5	92.1	69.8	70151	691	42146
Tracktor + PBS + CMC										
MOT17-02	43.6	41.2	46.1	99.8	41.2	89.9	48.1	7659	18	10922
MOT17-04	70.5	64.5	77.1	99.8	64.6	96.7	78.3	30711	72	16846
MOT17-05	59.1	56.9	61.4	98.1	57.5	84.6	69.6	3978	78	2939
MOT17-09	54.4	63.6	46.5	99.1	64.0	76.5	59.3	3408	30	1917
MOT17-10	64.0	71.2	57.6	98.4	72.1	92.1	60.7	9254	154	3585
MOT17-11	65.2	68.6	62.0	98.8	69.1	91.6	65.4	6524	80	2912
MOT17-13	72.1	73.3	71.1	98.1	74.3	89.9	76.9	8654	172	2988
OVERALL	64.3	62.2	66.6	99.1	62.5	92.4	69.9	70188	603	42109

Table 4.9: Ablation study per sequence for Tracktor + camera motion compensation. For GBS and PBS parameters are set to $bias = 1$, $B = 5$, and the results are the mean of five different runs.

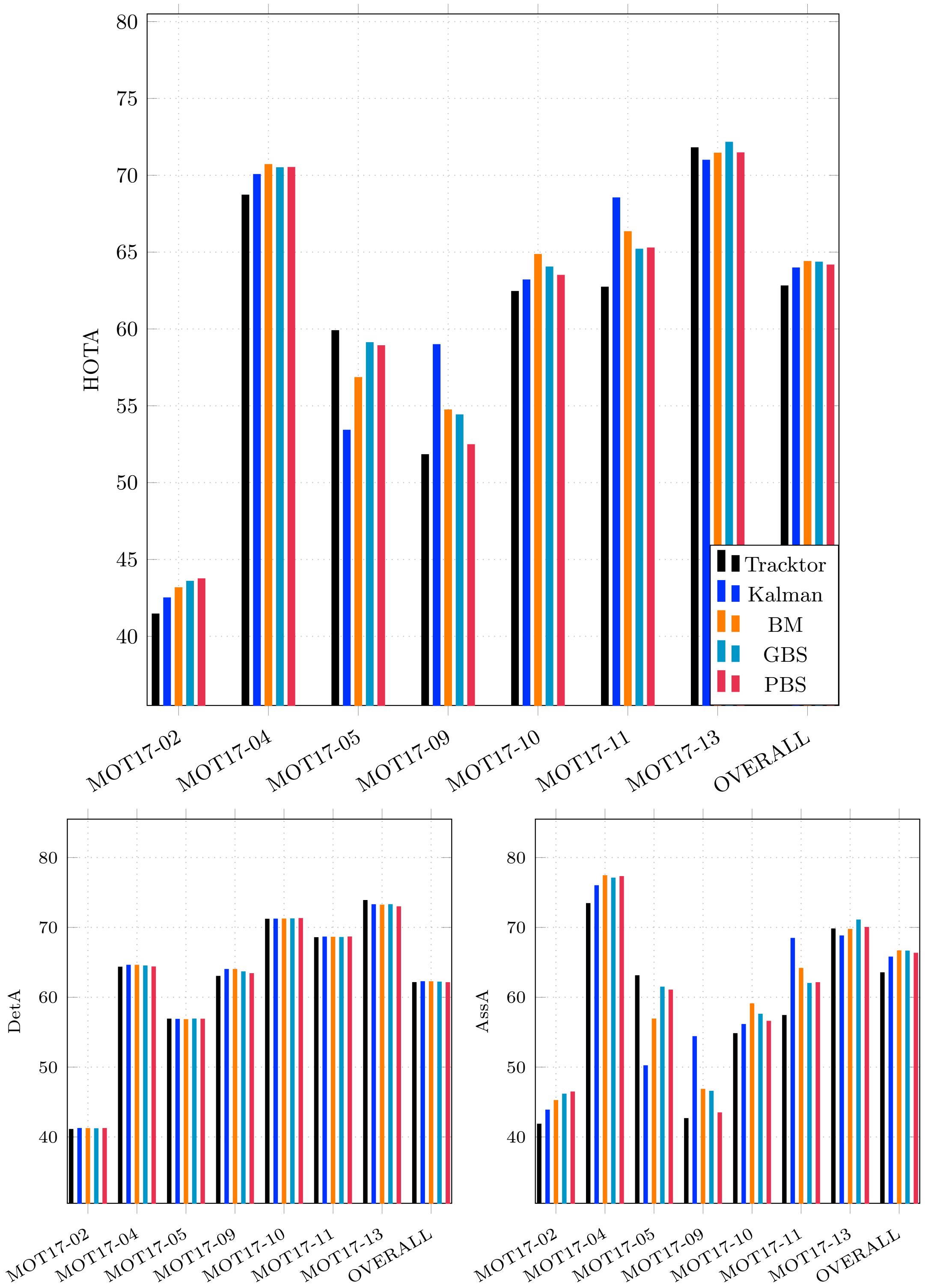


Figure 4.17: Tracktor + TrajE + CMC per sequence.

Sequence	HOTA	DetA	AssA	DetPr	DetRe	AssPr	AssRe	HOTA_TP	HOTA_FP	HOTA_FN
Tracktor + CV										
MOT17-02	41.5	41.1	41.9	99.7	41.1	95.6	42.7	7645	22	10936
MOT17-04	68.7	64.3	73.4	99.8	64.4	96.7	74.3	30631	70	16926
MOT17-05	60.0	56.8	63.3	98.0	57.5	93.7	65.7	3975	80	2942
MOT17-09	51.8	63.0	42.6	99.1	63.3	82.9	53.1	3373	30	1952
MOT17-10	57.2	69.9	46.8	98.0	70.9	87.8	50.6	9104	183	3735
MOT17-11	63.0	68.6	57.8	98.8	69.2	94.8	60.2	6527	82	2909
MOT17-13	69.6	73.1	66.3	99.0	73.7	90.1	71.7	8581	89	3061
OVERALL	62.0	61.9	62.1	99.2	62.2	93.6	64.6	69836	556	42461
Tracktor + Kalman + occ										
MOT17-02	50.4	46.6	54.5	97.0	47.2	86.6	58.6	8775	268	9806
MOT17-04	72.6	65.3	80.7	99.6	65.4	97.4	81.7	31125	117	16432
MOT17-05	56.0	56.1	56.0	87.2	61.1	76.7	68.0	4224	618	2693
MOT17-09	65.0	70.4	60.0	97.4	71.8	80.8	72.8	3823	103	1502
MOT17-10	49.2	66.1	36.6	88.9	72.1	71.9	42.4	9255	1161	3584
MOT17-11	66.2	67.9	64.5	97.0	69.4	90.7	68.8	6547	204	2889
MOT17-13	68.6	70.7	66.5	94.2	73.9	85.7	74.5	8605	529	3037
OVERALL	64.4	62.8	66.2	96.0	64.4	88.8	70.6	72354	3000	39943
Tracktor + BM + occ										
MOT17-02	45.3	43.3	47.3	99.5	43.4	89.2	49.8	8068	42	10513
MOT17-04	70.8	64.7	77.5	99.8	64.8	95.0	80.3	30817	75	16740
MOT17-05	60.0	57.9	62.2	95.8	59.4	83.1	70.5	4108	180	2809
MOT17-09	56.0	65.7	47.8	98.9	66.3	77.9	61.6	3528	41	1797
MOT17-10	60.2	70.7	51.3	97.9	71.8	85.5	54.9	9216	194	3623
MOT17-11	65.6	68.9	62.5	98.8	69.4	92.6	65.7	6553	81	2883
MOT17-13	70.3	73.6	67.1	98.4	74.5	86.9	74.5	8676	142	2966
OVERALL	64.2	62.8	65.7	98.9	63.2	90.4	70.0	70966	755	41331
Tracktor + GBS + occ										
MOT17-02	46.0	43.3	48.9	99.5	43.4	89.6	51.2	8071	42	10510
MOT17-04	70.4	64.6	76.8	99.8	64.7	96.7	78.0	30768	76	16789
MOT17-05	59.8	57.7	62.0	96.2	59.0	84.0	70.3	4082	163	2835
MOT17-09	54.9	64.9	46.4	98.9	65.4	75.2	59.8	3482	38	1843
MOT17-10	59.5	69.6	50.8	97.0	71.2	83.7	55.5	9139	283	3700
MOT17-11	65.6	68.9	62.4	98.5	69.6	91.1	65.9	6570	100	2866
MOT17-13	70.5	72.9	68.1	98.8	73.6	89.1	74.4	8570	106	3072
OVERALL	64.0	62.5	65.5	98.9	62.9	91.0	69.1	70682	807	41615
Tracktor + PBS + occ										
MOT17-02	46.0	43.3	48.8	99.5	43.4	89.9	51.1	8067	41	10514
MOT17-04	70.7	64.5	77.4	99.8	64.6	96.9	78.5	30727	75	16830
MOT17-05	60.4	58.2	62.8	97.1	59.2	84.9	70.6	4094	122	2823
MOT17-09	55.4	65.1	47.3	99.1	65.4	73.4	60.5	3485	31	1840
MOT17-10	59.0	69.8	49.8	97.4	71.2	85.2	54.1	9138	247	3701
MOT17-11	65.4	68.8	62.1	98.3	69.6	90.5	65.7	6566	112	2870
MOT17-13	70.5	72.9	68.2	98.7	73.6	89.4	74.1	8567	114	3075
OVERALL	64.1	62.5	65.7	99.0	62.9	91.2	69.2	70643	741	41654

Table 4.10: Ablation study per sequence for Tracktor + occlusion reconstruction. For GBS and PBS parameters are set to $bias = 1$, $B = 5$, and the results are the mean of five different runs.

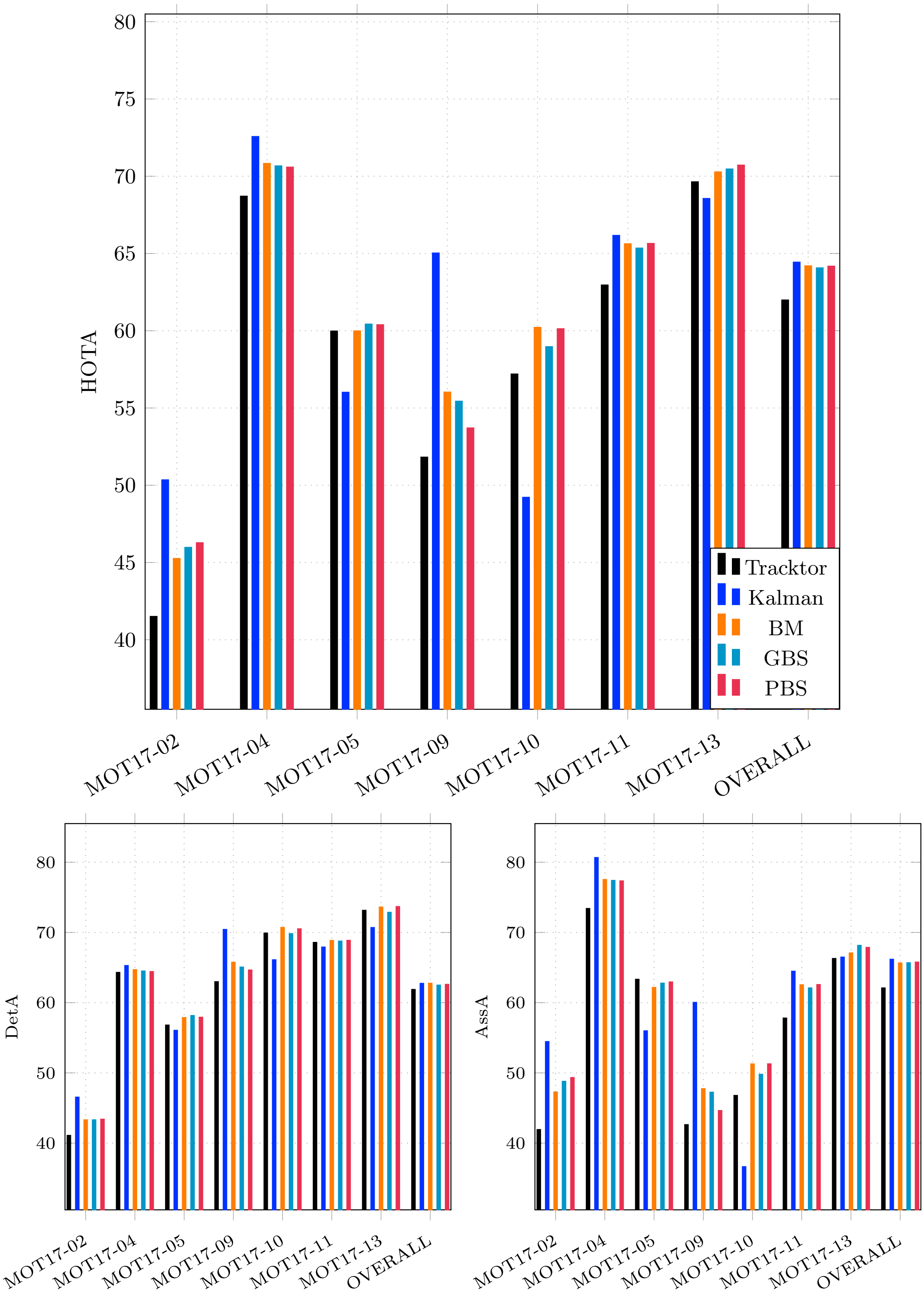


Figure 4.18: Tracktor + TrajE + occ per sequence.

Sequence	HOTA	DetA	AssA	DetPr	DetRe	AssPr	AssRe	HOTA_TP	HOTA_FP	HOTA_FN
Tracktor + CV + CMC										
MOT17-02	41.4	41.1	41.8	99.7	41.1	94.5	42.7	7641	22	10940
MOT17-04	68.7	64.3	73.4	99.8	64.4	96.7	74.3	30631	71	16926
MOT17-05	59.9	56.9	63.1	98.0	57.5	92.3	66.4	3979	80	2938
MOT17-09	51.8	63.0	42.6	99.1	63.3	82.9	53.1	3373	30	1952
MOT17-10	62.4	71.2	54.8	98.3	72.0	93.8	57.2	9248	156	3591
MOT17-11	62.7	68.5	57.4	98.8	69.1	93.5	60.1	6520	78	2916
MOT17-13	71.8	73.9	69.8	98.2	74.9	89.7	75.9	8716	160	2926
OVERALL	62.8	62.1	63.5	99.2	62.4	94.0	66.0	70108	597	42189
Tracktor + Kalman + occ + CMC										
MOT17-02	49.7	47.0	52.5	97.2	47.7	82.2	56.9	8858	256	9723
MOT17-04	72.9	65.4	81.2	99.7	65.5	97.8	82.1	31151	93	16406
MOT17-05	54.5	56.3	52.7	87.0	61.5	71.1	69.3	4253	638	2664
MOT17-09	60.8	69.3	53.3	97.8	70.3	78.9	66.9	3746	83	1579
MOT17-10	62.4	69.9	55.7	95.2	72.5	87.0	59.2	9310	473	3529
MOT17-11	65.6	68.3	63.0	96.9	69.8	91.3	66.8	6590	211	2846
MOT17-13	69.8	71.5	68.2	95.8	73.7	89.4	73.9	8585	372	3057
OVERALL	65.7	63.4	68.1	97.2	64.6	90.4	72.2	72493	2126	39804
Tracktor + BM + occ + CMC										
MOT17-02	45.7	43.6	47.9	99.5	43.7	89.2	50.4	8115	42	10466
MOT17-04	70.8	64.7	77.5	99.8	64.8	95.0	80.3	30817	75	16740
MOT17-05	57.3	57.4	57.2	95.1	59.2	79.5	67.9	4095	211	2822
MOT17-09	56.0	65.6	47.8	98.8	66.1	77.9	61.7	3520	41	1805
MOT17-10	65.2	71.7	59.2	98.2	72.7	89.4	62.7	9330	167	3509
MOT17-11	66.7	69.2	64.3	98.6	69.8	93.4	67.0	6591	94	2845
MOT17-13	71.4	73.3	69.5	97.7	74.6	88.7	75.6	8684	203	2958
OVERALL	64.9	62.9	66.9	98.8	63.4	90.9	71.2	71152	833	41145
Tracktor + GBS + occ + CMC										
MOT17-02	46.1	43.4	49.0	99.4	43.5	89.6	51.3	8077	46	10504
MOT17-04	70.3	64.6	76.6	99.8	64.7	96.5	77.9	30749	76	16808
MOT17-05	60.2	58.4	62.0	96.8	59.5	84.0	70.9	4118	137	2799
MOT17-09	54.3	64.8	45.5	98.9	65.2	73.0	59.7	3474	38	1851
MOT17-10	63.5	71.6	56.3	97.5	73.0	89.7	60.3	9370	244	3469
MOT17-11	66.4	69.3	63.6	98.6	70.0	91.9	67.0	6604	95	2832
MOT17-13	72.1	73.9	70.4	97.8	75.1	88.7	77.5	8746	195	2896
OVERALL	64.6	62.9	66.4	98.8	63.3	91.6	70.2	71137	830	41160
Tracktor + PBS + occ + CMC										
MOT17-02	46.0	43.3	48.8	99.5	43.4	89.9	51.0	8069	45	10512
MOT17-04	70.6	64.6	77.2	99.8	64.7	96.7	78.5	30766	75	16791
MOT17-05	58.8	57.7	59.9	96.5	58.9	83.4	68.3	4074	150	2843
MOT17-09	55.7	65.5	47.4	99.0	65.9	76.0	60.7	3511	35	1814
MOT17-10	64.0	71.4	57.5	98.1	72.4	90.7	61.0	9294	182	3545
MOT17-11	65.5	68.9	62.3	98.5	69.6	91.2	65.7	6569	101	2867
MOT17-13	72.1	73.3	70.9	97.8	74.6	91.0	75.7	8679	194	2963
OVERALL	64.7	62.8	66.8	98.9	63.2	92.1	70.1	70963	781	41334

Table 4.11: Ablation study per sequence for Tracktor + occlusion reconstruction + camera motion compensation. For GBS and PBS parameters are set to $bias = 1$, $B = 5$, and the results are the mean of five different runs.

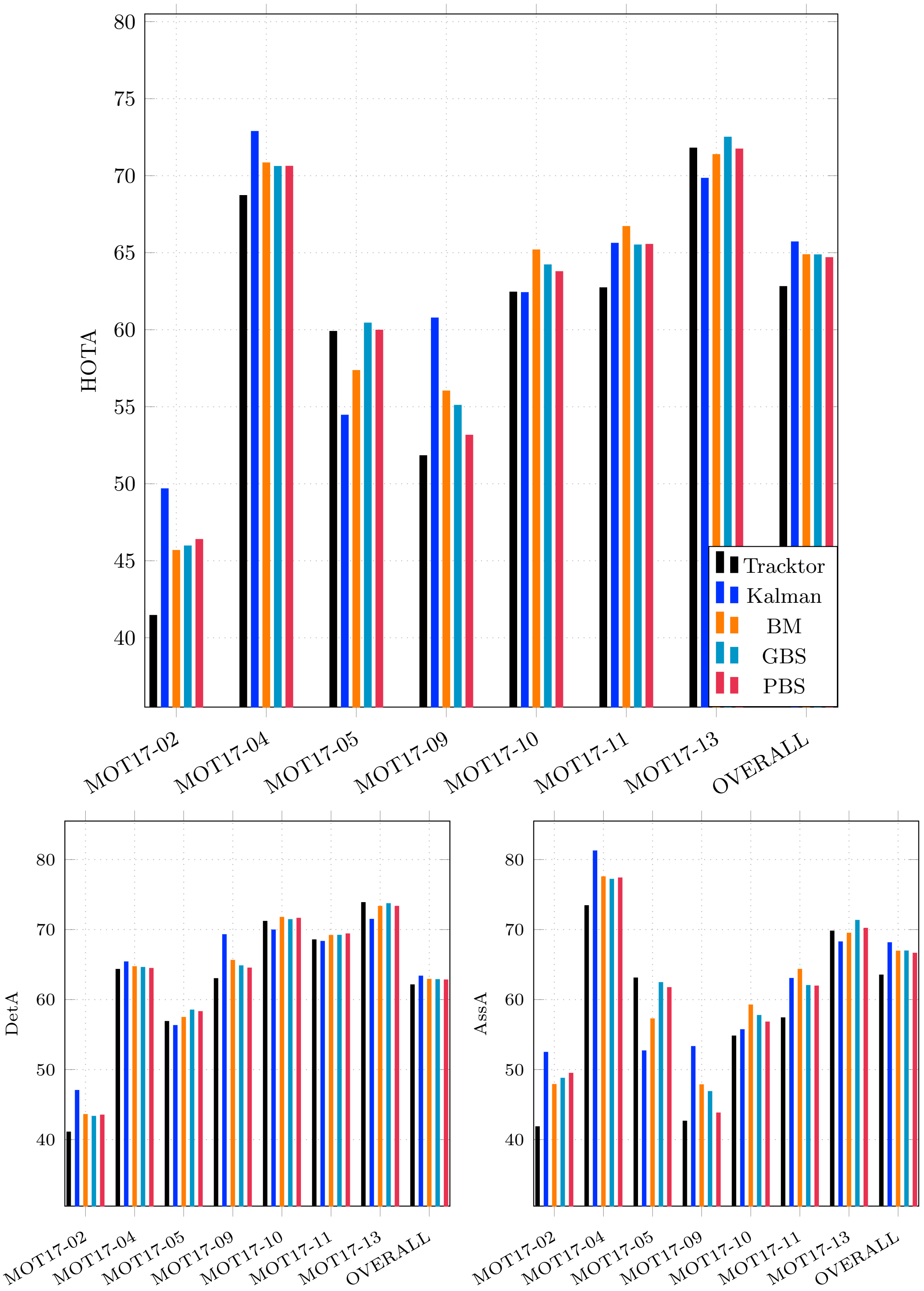


Figure 4.19: Tracktor + TrajE + occ + CMC per sequence

Conclusions

5

Less than ten years ago, the machine learning community foresaw the revolution that was going to come. AlexNet [1], applied to image classification, was the starting shot of a series of fields evolving at a vertiginous pace. With strong fundamentals built during the past eighty years, deep learning has become the leading technology to use in academia, and it is becoming the core of many industrial applications. This thesis is an example on how both worlds can benefit from each other, and, among other conclusions, in this final chapter we emphasize practices from where the academia could benefit from the industry and vice-versa.

From the industrial point of view, deep learning has become a double edged sword. On the one hand, it allows to tackle challenges that were unsolvable just a few years ago due to its potential, and the improvements being done to standardize the working pipeline. On the other hand, many new competitors are appearing and, if the company does not keep up with the fast pace of the current developments, it may lead to loss of competitiveness and, in last instance, the disappearance of the firm. Also, a difficulty to incorporate deep learning techniques into the production pipeline lies in the feeling of lack of control, despite the final results, in comparison with other ”more classical” methods.

This thesis has been developed in the AutomaticTV context, where we focused on tackling the object detection challenge and its surrounding pipeline. To carry on this project, we relied on many knowledge developed by the academia, from deep learning architectures to the dataset generation, and kept integrating state of the art techniques to the final product. As a result, the generated object detection models are being currently used by many customers around the world, and became a key component for the automatic sports production.

On the other hand, for developing the second part of this thesis, devoted to the multiple object tracking from a pure academic perspective, we used many resources and learned practices from the industry. We introduced trajectory estimation as a key element for tracking. We built TrajE, a lightweight trajectory estimator based on mixture density networks, and integrated it to two state of the art multiple object trackers, CenterTrack [8] and Tracktor [44]. Together with a beam search technique to explore multiple trajectory hypotheses, and reconstructing occlusions based on the estimated trajectories, both trackers performance is boosted by a considerable margin. In the case of CenterTrack + TrajE, we set a new state of the art on both widely used MOTChallenge [7] and UA-DETRAC [47] datasets.

To keep on with this symbiosis, it is mandatory for industry to support academia, as it is the present and future of many yet unthought ideas, and for the academia to support industry, as it paves the road for the ideas to become reality.

During this thesis, we saw many practices that are not shared between these two worlds, and we believe they could help both grow stronger. For us, the academic world would benefit from the knowledge of industry on improving the standardization of the published solutions (especially code and datasets), methodologies for optimizing the time spent implementing an idea, and more teamwork developing the same idea, which we feel is usually discouraged for, what we call, "first author syndrome", where usually the first author gets all the credit. On the other side, we think industry would benefit from a more open community, as many secrets are not shared due to fear of giving away any competitive advantage, and more active engagement with the sate of the art, to be able to prepare for next advances.

To conclude, throughout this thesis we observed how artificial intelligence is currently shaping the future for next generations. We expect the trend to continue, due to the wide range of applications, and the amount of people working on making the whole pipeline, from annotating to deployment, accessible to anyone. Considering the extremely large, and uncontrollable, amount of possibilities using these emergent technologies, it will take a global effort to stick to an ethical code to make them focus on the wellness and freedom of society. Let us hope for the best.

Bibliography

[1] Krizhevsky, A, Sutskever, I, & Hinton, G. E. (2017) Imagenet classification with deep convolutional neural networks. *Communications of the ACM* 60, 84–90. 1, 2, 10, 63

[2] LeCun, Y, Bengio, Y, & Hinton, G. (2015) Deep learning. *nature* 521, 436–444. 1

[3] Goodfellow, I, Bengio, Y, Courville, A, & Bengio, Y. (2016) *Deep learning.* (MIT press Cambridge) Vol. 1. 1

[4] Deng, J, Dong, W, Socher, R, Li, L.-J, Li, K, & Fei-Fei, L. (2009) *Imagenet: A large-scale hierarchical image database.* (Ieee), pp. 248–255. 2

[5] Everingham, M, Van Gool, L, Williams, C. K. I, Winn, J, & Zisserman, A. (year?) The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results (http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html). 2

[6] Perazzi, F, Pont-Tuset, J, McWilliams, B, Van Gool, L, Gross, M, & Sorkine-Hornung, A. (2016) *A Benchmark Dataset and Evaluation Methodology for Video Object Segmentation.* 2

[7] Milan, A, Leal-Taixé, L, Reid, I, Roth, S, & Schindler, K. (2016) MOT16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs].* arXiv: 1603.00831. 2, 18, 23, 25, 32, 34, 44, 47, 63

[8] Zhou, X, Koltun, V, & Krähenbühl, P. (2020) Tracking objects as points. *arXiv preprint arXiv:2004.01177.* 2, 17, 19, 20, 23, 25, 30, 44, 47, 63

[9] Awad, G, Fiscus, J, Michel, M, Joy, D, Kraaij, W, Smeaton, A. F, Quénot, G, Eskevich, M, Aly, R, Jones, G. J. F, Ordelman, R, Huet, B, & Larson, M. (2016) *TRECVID 2016: Evaluating Video Search, Video Event Detection, Localization, and Hyperlinking.* 2

[10] Salvador, A, Bellver, M, Campos, V, Baradad, M, Marques, F, Torres, J, & Giro-i Nieto, X. (2017) Recurrent neural networks for semantic instance segmentation. *arXiv preprint arXiv:1712.00617.* 3

[11] Pont-Tuset, J, Perazzi, F, Caelles, S, Arbeláez, P, Sorkine-Hornung, A, & Van Gool, L. (2017) The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675.* 3

[12] Xu, N, Yang, L, Fan, Y, Yue, D, Liang, Y, Yang, J, & Huang, T. (2018) Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327.* 3

[13] Yang, X, Sun, H, Sun, X, Yan, M, Guo, Z, & Fu, K. (2018) Position detection and direction prediction for arbitrary-oriented ships via multitask rotation region convolutional neural network. *IEEE Access* 6, 50839–50849. 9

[14] Lowe, D. G. (1999) *Object recognition from local scale-invariant features.* (Ieee), Vol. 2, pp. 1150–1157. 10

[15] Jones, M & Viola, P. (2003) Fast multi-view face detection. *Mitsubishi Electric Research Lab TR-20003-96* 3, 2. 10

[16] Dalal, N & Triggs, B. (2005) *Histograms of oriented gradients for human detection.* (IEEE), Vol. 1, pp. 886–893. 10

[17] Felzenszwalb, P. F, Girshick, R. B, McAllester, D, & Ramanan, D. (2009) Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence* 32, 1627–1645. 10

[18] Sermanet, P, Eigen, D, Zhang, X, Mathieu, M, Fergus, R, & LeCun, Y. (2013) Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229.* 10

[19] Girshick, R, Donahue, J, Darrell, T, & Malik, J. (2014) *Rich feature hierarchies for accurate object detection and semantic segmentation.* pp. 580–587. 10

[20] Uijlings, J. R, Van De Sande, K. E, Gevers, T, & Smeulders, A. W. (2013) Selective search for object recognition. *International journal of computer vision* 104, 154–171. 10

[21] Arbeláez, P, Pont-Tuset, J, Barron, J. T, Marques, F, & Malik, J. (2014) *Multiscale combinatorial grouping.* pp. 328–335. 10

[22] Cortes, C & Vapnik, V. (1995) Support-vector networks. *Machine learning* 20, 273–297. 10

[23] He, K, Zhang, X, Ren, S, & Sun, J. (2015) Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence* 37, 1904–1916. 10

[24] Girshick, R. (2015) *Fast r-cnn.* pp. 1440–1448. 11

[25] Ren, S, He, K, Girshick, R, & Sun, J. (2015) *Faster r-cnn: Towards real-time object detection with region proposal networks.* pp. 91–99. 11, 12, 18, 31

[26] Lin, T.-Y, Dollár, P, Girshick, R, He, K, Hariharan, B, & Belongie, S. (2017) *Feature pyramid networks for object detection.* pp. 2117–2125. 11, 13

[27] Li, Y, Chen, Y, Wang, N, & Zhang, Z. (2019) *Scale-aware trident networks for object detection.* pp. 6054–6063. 11, 12

[28] Yu, F & Koltun, V. (2015) Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122.* 11

[29] Redmon, J, Divvala, S, Girshick, R, & Farhadi, A. (2016) *You only look once: Unified, real-time object detection.* pp. 779–788. 11, 12

[30] Redmon, J & Farhadi, A. (2017) *YOLO9000: better, faster, stronger.* pp. 7263–7271. 12, 18

[31] Redmon, J & Farhadi, A. (2018) Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767.* 12

[32] Bochkovskiy, A, Wang, C.-Y, & Liao, H.-Y. M. (2020) Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934.* 12

[33] Liu, W, Anguelov, D, Erhan, D, Szegedy, C, Reed, S, Fu, C.-Y, & Berg, A. C. (2016) *Ssd: Single shot multibox detector.* (Springer), pp. 21–37. 12, 18

[34] Everingham, M, Van Gool, L, Williams, C. K. I, Winn, J, & Zisserman, A. (year?) The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results (http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html). 13

[35] Lin, T.-Y, Goyal, P, Girshick, R, He, K, & Dollár, P. (2017) *Focal loss for dense object detection.* pp. 2980–2988. 13, 14

[36] He, K, Zhang, X, Ren, S, & Sun, J. (2016) *Deep Residual Learning for Image Recognition.* 13

[37] Law, H & Deng, J. (2018) *Cornernet: Detecting objects as paired keypoints.* pp. 734–750. 13

[38] Zhou, X, Wang, D, & Krähenbühl, P. (2019) Objects as points. *arXiv preprint arXiv:1904.07850.* 13, 18

[39] Zhou, X, Zhuo, J, & Krahenbuhl, P. (2019) *Bottom-up object detection by grouping extreme and center points.* pp. 850–859. 13

[40] Tan, M, Pang, R, & Le, Q. V. (2020) *Efficientdet: Scalable and efficient object detection.* pp. 10781–10790. 14

[41] Carion, N, Massa, F, Synnaeve, G, Usunier, N, Kirillov, A, & Zagoruyko, S. (2020) End-to-end object detection with transformers. *arXiv preprint arXiv:2005.12872.* 14

[42] Lu, X, Li, Q, Li, B, & Yan, J. (2020) Mimicdet: Bridging the gap between one-stage and two-stage object detection. *arXiv preprint arXiv:2009.11528.* 14

[43] Ciaparrone, G, Sánchez, F. L, Tabik, S, Troiano, L, Tagliaferri, R, & Herrera, F. (2020) Deep learning in video multi-object tracking: A survey. *Neurocomputing* 381, 61–88. 17

[44] Bergmann, P, Meinhardt, T, & Leal-Taixe, L. (2019) *Tracking without bells and whistles.* pp. 941–951. 17, 19, 20, 23, 25, 30, 31, 44, 47, 63

[45] Feichtenhofer, C, Pinz, A, & Zisserman, A. (2017) *Detect to track and track to detect.* pp. 3038–3046. 17, 19, 20

[46] Ren, L, Lu, J, Wang, Z, Tian, Q, & Zhou, J. (2018) *Collaborative deep reinforcement learning for multi-object tracking.* pp. 586–602. 17, 22, 23

[47] Wen, L, Du, D, Cai, Z, Lei, Z, Chang, M, Qi, H, Lim, J, Yang, M, & Lyu, S. (2020) UA-DETRAC: A new benchmark and protocol for multi-object detection and tracking. *Computer Vision and Image Understanding.* 18, 23, 25, 34, 44, 47, 63

[48] Hermans, A, Beyer, L, & Leibe, B. (2017) In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737.* 18

[49] Geiger, A, Lenz, P, & Urtasun, R. (2012) *Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite.* 18

[50] Voigtlaender, P, Krause, M, Ošep, A, Luiten, J, Sekar, B. B. G, Geiger, A, & Leibe, B. (2019) *MOTS: Multi-Object Tracking and Segmentation.* 19

[51] Wen, L, Du, D, Li, S, Bian, X, & Lyu, S. (2019) *Learning non-uniform hypergraph for multi-object tracking.* Vol. 33, pp. 8981–8988. 19, 20

[52] Maksai, A, Wang, X, Fleuret, F, & Fua, P. (2017) *Non-markovian globally consistent multi-object tracking.* pp. 2544–2554. 19, 20

[53] Schulter, S, Vernaza, P, Choi, W, & Chandraker, M. (2017) *Deep network flow for multi-object tracking.* pp. 6951–6960. 19

[54] Chen, L, Ai, H, Chen, R, & Zhuang, Z. (2019) Aggregate tracklet appearance features for multi-object tracking. *IEEE Signal Processing Letters* 26, 1613–1617. 19, 20, 47

[55] Ma, L, Tang, S, Black, M. J, & Van Gool, L. (2018) *Customized multi-person tracker.* (Springer), pp. 612–628. 19, 20

[56] Tang, S, Andriluka, M, Andres, B, & Schiele, B. (2017) *Multiple people tracking by lifted multicut and person re-identification.* pp. 3539–3548. 19, 20

[57] Ma, C, Yang, C, Yang, F, Zhuang, Y, Zhang, Z, Jia, H, & Xie, X. (2018) *Trajectory factory: Tracklet cleaving and re-connection by deep siamese bi-gru for multiple object tracking.* (IEEE), pp. 1–6. 19, 20

[58] Henschel, R, Leal-Taixé, L, Cremers, D, & Rosenhahn, B. (2017) Improvements to frank-wolfe optimization for multi-detector multi-object tracking. *arXiv preprint arXiv:1705.08314.* 19, 20, 47

[59] Sheng, H, Zhang, Y, Chen, J, Xiong, Z, & Zhang, J. (2018) Heterogeneous association graph fusion for target association in multiple object tracking. *IEEE Transactions on Circuits and Systems for Video Technology* 29, 3269–3280. 19, 20, 47

[60] Hornakova, A, Henschel, R, Rosenhahn, B, & Swoboda, P. (2020) Lifted disjoint paths with application in multiple object tracking. *arXiv preprint arXiv:2006.14550.* 19, 20, 47

[61] Brasó, G & Leal-Taixé, L. (2020) *Learning a neural solver for multiple object tracking.* pp. 6247–6257. 19, 20, 47

[62] Zhang, Y, Sheng, H, Wu, Y, Wang, S, Lyu, W, Ke, W, & Xiong, Z. (2020) Long-term tracking with deep tracklet association. *IEEE Transactions on Image Processing* 29, 6694–6706. 19, 20, 47

[63] Peng, J, Wang, T, Lin, W, Wang, J, See, J, Wen, S, & Ding, E. (2020) Tpm: Multiple object tracking with tracklet-plane matching. *Pattern Recognition* p. 107480. 19, 20, 47

[64] Bochinski, E, Eiselein, V, & Sikora, T. (2017) *High-speed tracking-by-detection without using image information.* (IEEE), pp. 1–6. 19, 20

[65] Chu, P & Ling, H. (2019) *Famnet: Joint learning of feature, affinity and multi-dimensional assignment for online multiple object tracking.* pp. 6172–6181. 19, 20, 47

[66] Zhu, J, Yang, H, Liu, N, Kim, M, Zhang, W, & Yang, M.-H. (2018) *Online multi-object tracking with dual matching attention networks.* pp. 366–382. 19, 20

[67] Sun, S, Akhtar, N, Song, H, Mian, A. S, & Shah, M. (2019) Deep affinity network for multiple object tracking. *IEEE transactions on pattern analysis and machine intelligence.* 19, 20

[68] Chen, L, Ai, H, Zhuang, Z, & Shang, C. (2018) *Real-time multiple people tracking with deeply learned candidate selection and person re-identification.* (IEEE), pp. 1–6. 19, 20

[69] Milan, A, Rezatofighi, S. H, Dick, A, Reid, I, & Schindler, K. (2017) *Online multi-target tracking using recurrent neural networks.* 19, 20

[70] Bewley, A, Ge, Z, Ott, L, Ramos, F, & Upcroft, B. (2016) *Simple online and realtime tracking.* (IEEE), pp. 3464–3468. 19, 20, 38, 42

[71] Wojke, N, Bewley, A, & Paulus, D. (2017) *Simple online and realtime tracking with a deep association metric.* (IEEE), pp. 3645–3649. 19, 20, 47

[72] Yang, B & Nevatia, R. (2012) *Multi-target tracking by online learning of non-linear motion patterns and robust appearance models.* (IEEE), pp. 1918–1925. 19

[73] Chu, P, Fan, H, Tan, C. C, & Ling, H. (2019) *Online multi-object tracking with instance-aware tracker and dynamic model refreshment.* (IEEE), pp. 161–170. 19, 20

[74] Chen, L, Ai, H, Shang, C, Zhuang, Z, & Bai, B. (2017) *Online multi-object tracking with convolutional neural networks.* (IEEE), pp. 645–649. 19, 20

[75] Xu, J, Cao, Y, Zhang, Z, & Hu, H. (2019) *Spatial-temporal relation networks for multi-object tracking.* pp. 3988–3998. 19, 20, 47

[76] Keuper, M, Tang, S, Andres, B, Brox, T, & Schiele, B. (2018) Motion segmentation & multiple object tracking by correlation co-clustering. *IEEE transactions on pattern analysis and machine intelligence* 42, 140–153. 19, 20, 47

[77] Sadeghian, A, Alahi, A, & Savarese, S. (2017) *Tracking the untrackable: Learning to track multiple cues with long-term dependencies.* pp. 300–311. 19, 20

[78] Liu, Q, Chu, Q, Liu, B, & Yu, N. (2020) *GSM: Graph Similarity Model for Multi-Object Tracking* ed. Bessiere, C. (International Joint Conferences on Artificial Intelligence Organization), pp. 530–536. Main track. 20, 47

[79] Rudenko, A, Palmieri, L, Herman, M, Kitani, K. M, Gavrila, D. M, & Arras, K. O. (2019) Human motion trajectory prediction: A survey. *arXiv preprint arXiv:1905.06113.* 21

[80] Sadeghian, A, Kosaraju, V, Gupta, A, Savarese, S, & Alahi, A. (2018) Trajnet: Towards a benchmark for human trajectory prediction. *arXiv preprint.* 22

[81] Pellegrini, S, Ess, A, Schindler, K, & Van Gool, L. (2009) *You'll never walk alone: Modeling social behavior for multi-target tracking.* (IEEE), pp. 261–268. 22

[82] Lerner, A, Chrysanthou, Y, & Lischinski, D. (2007) *Crowds by example.* (Wiley Online Library), Vol. 26, pp. 655–664. 22

[83] Colyar, J & Halkias, J. (2007) Us highway 101 dataset. *Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030.* 22

[84] Geiger, A, Lenz, P, Stiller, C, & Urtasun, R. (2013) Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research* 32, 1231–1237. 22

[85] Robicquet, A, Sadeghian, A, Alahi, A, & Savarese, S. (2016) *Learning social etiquette: Human trajectory understanding in crowded scenes.* (Springer), pp. 549–565. 22

[86] Leal-Taixé, L, Fenzi, M, Kuznetsova, A, Rosenhahn, B, & Savarese, S. (2014) *Learning an image-based motion context for multiple people tracking.* pp. 3542–3549. 22, 23

[87] Pellegrini, S, Ess, A, & Van Gool, L. (2010) *Improving data association by joint modeling of pedestrian trajectories and groupings.* (Springer), pp. 452–465. 22, 23

[88] Alahi, A, Goel, K, Ramanathan, V, Robicquet, A, Fei-Fei, L, & Savarese, S. (2016) *Social lstm: Human trajectory prediction in crowded spaces.* pp. 961–971. 22, 23

[89] Gupta, A, Johnson, J, Fei-Fei, L, Savarese, S, & Alahi, A. (2018) *Social gan: Socially acceptable trajectories with generative adversarial networks.* pp. 2255–2264. 22, 23

[90] Leal-Taixé, L, Pons-Moll, G, & Rosenhahn, B. (2011) *Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker.* (IEEE), pp. 120–127. 22

[91] Alahi, A, Ramanathan, V, Goel, K, Robicquet, A, Sadeghian, A. A, Fei-Fei, L, & Savarese, S. (2017) in *Group and Crowd Behavior for Computer Vision.* (Elsevier), pp. 183–207. 22, 23

[92] Xue, H, Huynh, D. Q, & Reynolds, M. (2018) *SS-LSTM: A hierarchical LSTM model for pedestrian trajectory prediction.* (IEEE), pp. 1186–1194. 22

[93] Makansi, O, Ilg, E, Cicek, O, & Brox, T. (2019) *Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction.* pp. 7144–7153. 22, 23

[94] Nikhil, N & Tran Morris, B. (2018) *Convolutional neural network for trajectory prediction.* pp. 0–0. 22

[95] Yamaguchi, K, Berg, A. C, Ortiz, L. E, & Berg, T. L. (2011) *Who are you with and where are you going?* (IEEE), pp. 1345–1352. 22

[96] Chen, B, Wang, D, Li, P, Wang, S, & Lu, H. (2018) *Real-time'Actor-Critic'Tracking.* pp. 318–334. 22, 23

[97] Ivanovic, B & Pavone, M. (2019) *The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs.* pp. 2375–2384. 22, 23

[98] Salzmann, T, Ivanovic, B, Chakravarty, P, & Pavone, M. (2020) Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. *arXiv preprint arXiv:2001.03093*. 22, 23

[99] Bishop, C. M. (1994) Mixture density networks. 26

[100] Tomé, A & Salgado, L. (2017) *Detection of anomalies in surveillance scenarios using mixture models.* (IEEE), pp. 1–4. 26

[101] Graves, A. (2013) Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*. 26, 28, 30

[102] Cho, K, Van Merriënboer, B, Gulcehre, C, Bahdanau, D, Bougares, F, Schwenk, H, & Bengio, Y. (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*. 30

[103] Duan, K, Bai, S, Xie, L, Qi, H, Huang, Q, & Tian, Q. (2019) *Centernet: Keypoint triplets for object detection.* pp. 6569–6578. 30

[104] Tang, S, Andres, B, Andriluka, M, & Schiele, B. (2016) *Multi-person tracking by multicut and deep matching.* (Springer), pp. 100–111. 34

[105] Bernardin, K & Stiefelhagen, R. (2008) Evaluating multiple object tracking performance: the clear mot metrics. *EURASIP Journal on Image and Video Processing* 2008, 1–10. 35

[106] Ristani, E, Solera, F, Zou, R, Cucchiara, R, & Tomasi, C. (2016) *Performance measures and a data set for multi-target, multi-camera tracking.* (Springer), pp. 17–35. 35

[107] Jonathon Luiten, Aljosa Osep, P. D. P. T. A. G. L. L.-T & Leibe, B. (2020) Hota: A higher order metric for evaluating multi-object tracking. *International Journal of Computer Vision*. 35

[108] Evangelidis, G. D & Psarakis, E. Z. (2008) Parametric image alignment using enhanced correlation coefficient maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30, 1858–1865. 41

[109] Henschel, R, Zou, Y, & Rosenhahn, B. (2019) *Multiple people tracking using body and joint detections.* pp. 0–0. 47

[110] Wang, G, Wang, Y, Zhang, H, Gu, R, & Hwang, J.-N. (2019) *Exploit the connectivity: Multi-object tracking with trackletnet.* pp. 482–490. 47

[111] Shen, H, Huang, L, Huang, C, & Xu, W. (2018) Tracklet association tracker: An end-to-end learning-based association approach for multi-object tracking. *arXiv preprint arXiv:1808.01562*. 47

[112] Yu, F, Li, W, Li, Q, Liu, Y, Shi, X, & Yan, J. (2016) *Poi: Multiple object tracking with high performance detection and appearance feature.* (Springer), pp. 36–42. 47

[113] Zeng, Y, Fu, X, Gao, L, Zhu, J, Li, H, & Li, Y. (2020) Robust multivehicle tracking with wasserstein association metric in surveillance videos. *IEEE Access* 8, 47863–47876. 47

[114] Yoon, J. H, Yang, M.-H, Lim, J, & Yoon, K.-J. (2015) *Bayesian multi-object tracking using motion context from multiple objects.* (IEEE), pp. 33–40. 47

[115] Dicle, C, Camps, O. I, & Sznaier, M. (2013) *The way they move: Tracking multiple targets with similar appearance.* pp. 2304–2311. 47

[116] Cai, Z, Saberian, M, & Vasconcelos, N. (2015) *Learning complexity-aware cascades for deep pedestrian detection.* pp. 3361–3369. 47