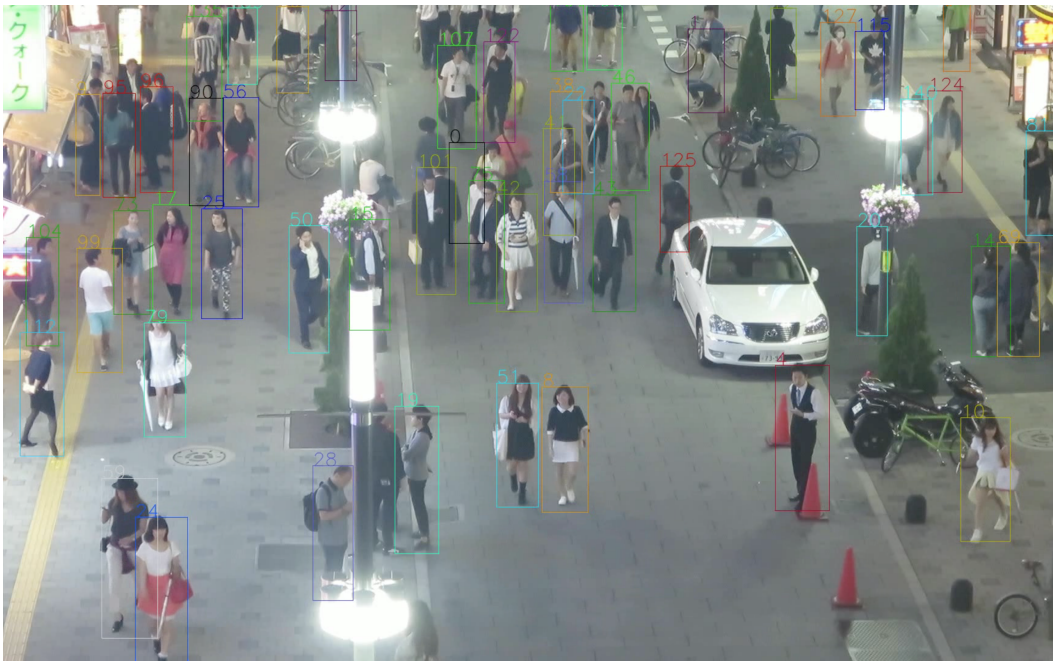
Deep Object Descriptions for MOT

- Multi-Object Tracking -



Master Thesis

Mathias Østergaard Poulsen

Aalborg University
Electronics and IT

AALBORG UNIVERSITY

STUDENT REPORT

Title:
Deep Object Descriptions for MOT

Theme:
Multi-Object Tracking

Project Period:
Spring 2022
Master Thesis

Participant:
Mathias Østergaard Poulsen

Supervisors:
Thomas B. Moeslund
Malte Pedersen

Copies: 1

Page Numbers: 86

Date of Completion:
June 2, 2022

Abstract:

This report investigates two deep approaches used to describe objects in multi-object tracking. The first approach masks out the background of detections and performs a deep feature extraction of a full frame to incorporate its spatial position. The approach achieves $\mu_{HOTA} = 46.0$ % on the MOT17 training sequences. The second approach uses SuperPoints to describe detections and SuperGlue to match the extracted SuperPoints. The method achieves $\mu_{HOTA} = 49.2$ % on the MOT17 training sequences. Combining the two descriptions into a single tracker proves beneficial, as it has access to the spatial incorporation from the first approach and the robust SuperPoint description from the second approach. The combined method achieves $\mu_{HOTA} = 51.8$ % on the MOT17 training sequences. Submitting to *MOTChallenge.net* it achieves $\mu_{HOTA} = 32.5$ % on the MOT17 test sequences. The report concludes that the first approach does not incorporate sufficient spatial information and that SuperPoint is a robust description that may benefit from a better spatial representation.

Contents

Preface v

1 Introduction 1

2 Multi-Object Tracking 3
- 2.1 Related Work . . . 3
- 2.2 The Tracking Problem . . . 5
- 2.3 Multi-Object Tracking Challenge . . . 10
- 2.4 Evaluation of Multi Object Tracking . . . 13
- 2.5 Contributions and Problem Statement . . . 16

3 Baseline Tracker 18
- 3.1 Tracking Approach . . . 19
- 3.2 Results . . . 22
- 3.3 Findings . . . 22

4 Proposed Method 24
- 4.1 Masking Out the Background . . . 24
- 4.2 Results . . . 27
- 4.3 Findings . . . 28

5 SuperPoint as a Tracker 30
- 5.1 SuperPoint . . . 30
- 5.2 SuperGlue . . . 32
- 5.3 Adapting SuperPoint to be a Tracker . . . 33
- 5.4 Results . . . 35
- 5.5 Findings . . . 35

6 Combining MOB with SuperPoint 36
- 6.1 Advanced Tracklet Handling . . . 36
- 6.2 Combined A - Joint Object Description . . . 37
- 6.3 Combined B - Unified Affinity Representation . . . 39

6.4 Results . . . 40
6.5 Findings . . . 41

7 Results **43**
7.1 Submission to MOTChallenge . . . 43
7.2 Ablation Study . . . 45
7.3 Generalizability to Other MOT Sequences . . . 46
7.4 Analysis of Failure Cases on the MOT17 Test Sequences . . . 46

8 Discussion **50**
8.1 Results on the Test Sequences . . . 50
8.2 Detection Quality . . . 51
8.3 Generalizability . . . 51
8.4 MOB and Spatial Incorporation . . . 51
8.5 Potentials with SuperPoint . . . 52
8.6 Runtime Perspectives . . . 52

9 Conclusion **53**
9.1 Future Research Areas . . . 54

Bibliography **55**

A MOT17 Examples **60**

B Affinity Investigations **63**
B.1 Affinity Results for the Baseline Tracker . . . 63
B.2 Affinity Results for the Proposed MOB Approach . . . 66
B.3 Affinity Results for the SuperPoint Framework . . . 69

C MOB - Additional Experiments **70**
C.1 Larger Input Shape . . . 70
C.2 Enlarging the Detection Size . . . 72
C.3 Training a Siamese Networks . . . 73

D Extended Results **76**
D.1 Baseline Results . . . 76
D.2 Results for the MOB Approach . . . 79
D.3 Results for SuperPoint Tracking . . . 82
D.4 Combined MOB and SuperPoint Results . . . 83
D.5 Ablation Results . . . 84
D.6 Results on the Test Sequences . . . 85
D.7 Results for Other MOT Sequences . . . 85

Preface

This report is submitted as the final project of the Robotic Master's Programme. The thesis investigates the use of deep learning to describe objects in multi-object tracking.

The software is implemented in Python 3.10 as an Emily project (ensures a stable Docker environment) and makes use of both the TensorFlow and Pytorch deep learning frameworks. The code can be found at GitHub.

Aalborg University, June 2, 2022

Mathias Østergaard Poulsen
<mapo17@student.aau.dk>

Chapter 1

Introduction

A wide array of modern applications and robotic system rely on tracking to provide an understanding of moving objects. A well known application is to analyze pedestrian movement in surveillance applications. This has received increasing attention with the Covid-19 pandemic and can be used to count the number of people in indoor areas and estimate their proximity. Self-driving cars heavily rely on tracking of people and other objects along the road to predict and prevent potential accidents. Other robotic systems such as modern traffic regulations can use tracking of vehicles to save fuel consumption and thus reduce the CO_2 emission [1, 2, 3, 4]. Lastly, tracking is also widely used in biology and medical research for tracking of e.g. cells and fish to avoid manual annotation in clinical experiments [5, 6, 7].

All of the aforementioned applications can benefit from multi-object tracking (MOT) to cover the association of corresponding objects across a video sequence. Depending on the application, this association is required to establish object trajectories, keep track of objects, predict motion of objects and avoid identity switches etc.

The work of MOT accelerated in 2015 with Taixé *et al.* [8] that introduced the MOTChallenge benchmark[1]. The challenge began with the MOT15 dataset and has since been extended with more challenging datasets such as the MOT16/17 and MOT20. These initial MOT challenges primarily focuses on pedestrian tracking in an urban environment. More novel additions include biological data e.g. the 3D-ZeF20 dataset which consist of 3D sequences of zebrafish [6] and the CTMC-v1 dataset which is sequences of cells [7]. The latest addition is the MOTSynth dataset that consists of synthetic data targeted towards the increased used of deep learning in tracking, since it often requires a vast amount of data [9].

An analysis of state-of-the-art MOT trackers in 2017, found that the best performing trackers have strong affinity models that is achieved with deep learning (AI) [10]. Ad-

[1]The MOTChallenge benchmark is hosted at `https://motchallenge.net`

ditionally, the use of deep learning in MOT has grown to be essential in detecting the objects to be tracked using modern detectors such as Faster-RCNN [11] and YOLO [12]. More advanced strategies use deep learning to explore the temporal dimension of video tracking through recurrent neural networks (RNN) to perform either motion prediction or association of objects [13].

To this end, deep learning is a core part of most modern tracking algorithms. The following chapter investigates state-of-the-art methods in tracking to understand the MOT problem, and identifies how deep learning is used to improve upon the various stages of a tracking algorithm.

Chapter 2

Multi-Object Tracking

This chapter takes its starting point in identifying state-of-the-art methods through a related work analysis. This leads to a formal definition of the MOT problem and an identification of 5 stages of a tracking framework. It gives an introduction to the MOTChallenge benchmark with examples of available datasets and establishes the evaluation metrics used to assess the performance of tracking algorithms.

2.1 Related Work

Deep learning in MOT can traditionally be divided into two categories; tracking-by-detection and joint detection and tracking. Additionally, siamese networks have proved efficient for single-object tracking, thus they have received increased attention in MOT.

2.1.1 Tracking-by-Detection

The usage of deep learning in tracking inherits from the introduction of fast and robust object detectors such as Faster R-CNN [11] and the various YOLO versions [12]. They transform the MOT problem into tracking-by-detection. This separates the tracking problem into two subproblems; detection of objects to be tracked and association of detected objects into tracklets [13]. An early tracking-by-detection approach from 2010 [14] utilized an SVM detector to detect pedestrians and associate them with tracklets across frames. Bewley *et al.* [15] introduced simple online and real-time tracking (SORT) in 2017, which is a tracking-by-detection method that is based on a Kalman Filter for motion prediction and performs data association between detections based on the intersection over union. The approach has later been expanded with DeepSORT, which extends the association process to include a deep appearance model [16]. StrongSORT included a larger CNN-backbone (ResNet-50) to better describe the deep appearances and added a separate re-identification module on top to re-identify lost objects and interpolate their position [17]. The latest addition to the SORT family is the observation-centric SORT method, which modifies the

motion model, e.g. the Kalman Filter, to be more robust to state noise and occlusions [18]. Other approaches for tracking-by-detection include advanced deep learning techniques to improve the association [19]. [20] uses a trained neural solver with a deep feature representation and [21, 22] use variations of RNNs to utilize the temporal information of both motion and appearance.

2.1.2 Joint Detection and Tracking

Bergmann *et al.* [10] proposed to modify an object detector into a tracker (*Tracktor*). The approach is also known in the literature as tracking-by-regression as it modifies the regression head of an object detector, in this case, Faster R-CNN, to regress the next object location of a tracklet. With the addition of a siamese network to perform re-identification, this approach achieved state-of-the-art on several of the MOTChallenges in 2019. The approach was later improved by [23] with more sophisticated track management and occlusion handling.

Another approach that also modifies existing object detectors is CenterTrack [24]. CenterTrack modifies the CenterNet [25] object detector, to predict offsets for object centers between the previous and the current frame. It aims to simplify the tracking problem to tracking points inspired by early tracking algorithms and uses greedy association to associate spatially close object centers. A key issue with CenterTrack and joint detection and tracking methods generally are that they weigh the detection task over the re-identification task. Hence, they are rarely able to perform long-range re-identifications. To accommodate this, [26] proposed modifications to CenterTrack, which includes a separate network branch dedicated to extracting features that enhance its ability to perform long-range re-identifications.

2.1.3 Siamese Networks in Tracking

Deep appearance models are key in representing objects and thus performing association and re-identification [10]. Other approaches utilize siamese networks to address the deep appearance since they predict an affinity measure directly, thus making them efficient for matching detections [27, 28].

Inspired by the joint tracking and detection methods, [29] replaced the regression head of a Faster-RCNN with a siamese head to re-detect a single object in consecutive frames. [30] uses a slightly different siamese network structure suited for tracking a single object, by predicting the most similar location of the object within a small search region of the next frame. While early work on siamese networks focused on single-object tracking [29, 30, 31], Shuai *et al.* [32] presented SiamMOT designed for MOT. SiamMOT extends a Faster R-CNN with a siamese tracking head to simultaneously detect and track objects based on their similarity. SiamMOT outperforms both Tracktor and CenterTrack that is previously considered state-of-the-art in MOT.

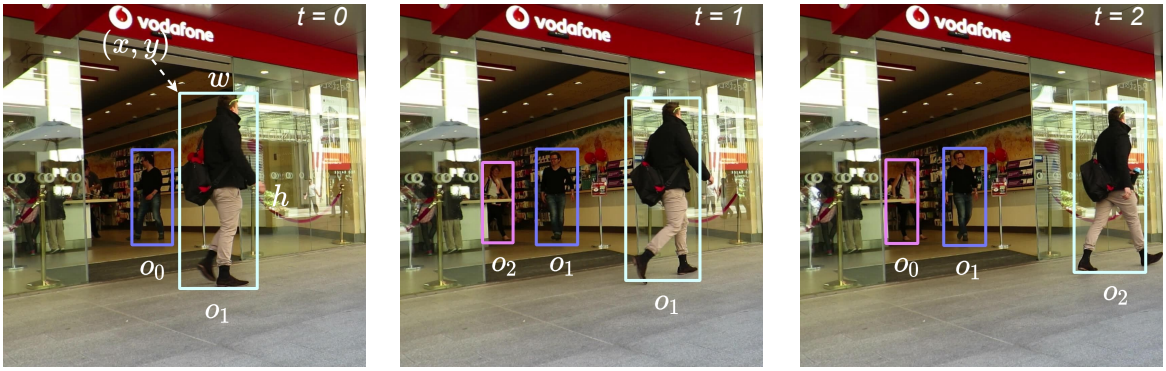


Figure 2.1: Visualization of the tracking problem. The set of objects from the three different frames $O_0 = \{o_0^0,\ o_1^0\}$, $O_1 = \{o_0^1,\ o_1^1,\ o_2^1\}$ and $O_2 = \{o_0^2,\ o_1^2,\ o_2^2\}$ can be associated to constitute the three tracklets $T_0 = \{o_0^0,\ o_0^1,\ o_0^2\}$, $T_1 = \{o_1^0,\ o_1^1,\ o_1^2\}$ and $T_2 = \{o_2^1,\ o_2^2\}$. The example is from the MOT17 dataset [34].

2.2 The Tracking Problem

To understand the extent and the aspects of MOT, the problem needs to be formally defined. This report adopts the definition of MOT from Bergmann *et al.* [33].

We can define an object, o, in a frame at time, t, as the bounding box given by

$$o_n^t = (x,\ y,\ w,\ h) \tag{2.1}$$

where n is the object number, $(x,\ y)$ is the top left corner of the bounding box and $(w,\ h)$ is the width and height of the bounding box, respectively.

We describe the collection of objects in a frame at time, t, by the set

$$O_t = \{o_n^t | n \in N\}, \tag{2.2}$$

where N is the total number of objects. Notice, that all objects might not be visible in the frame due to occlusions.

Lastly, we can describe the tracklet of an object, n, as a collection bounding boxes across the frames of a video sequence by

$$T_n = \{o_n^t, o_n^{t+1}, \cdots\} \tag{2.3}$$

Notice, that the objects might not be present in all frames, as they can move in and out of the captured scene.

An example of a simplified tracking problem is given in Figure 2.1.

2.2.1 Tracking-by-detection

When solving the MOT problem in a real-case scenario, the objects positions, o_n, and the total number of objects, N is not know in advantage. Taking starting point in tracking-by-detection, we can define a set of detections for a frame at time, t, as

$$D_t = \{d_m^t | m \in M_t\}, \tag{2.4}$$

where M_t is the total number of detections in frame t. d is defined by its four bounding box coordinates, similar to the definition of an object, o, from Equation 2.1.

After obtaining a set of detections, they need to be associated with a set of known tracklets. This set of known tracklets is defined as

$$\mathrm{T} = \{T_k | k \in K\}, \tag{2.5}$$

where K describes the total number of known tracklets in a video. Similarly to the previous definition, we can define a tracklet, T_k, by a set of associated detections given by:

$$T_k = \{d_m^t,\ d_m^{t+1},\ \cdots\} \tag{2.6}$$

In summary, tracking-by-detection is the problem of assigning M detections to K tracklets across time t.

2.2.2 Tracking Stages

To this end, tracking-by-detection can be divided into five stages as given below. These include the four stages identified by Ciaparrone *et al.* [13] in their survey on deep leaning usage in MOT with the addition of a stage for track handling. Examples of common methods found in the literature are given for each stage. Note, that depending on the tracker, some of the stages may be intertwined. The following introduction of tracking stages disregards the notation of time.

- **Object Detection Stage**
  The object detector is responsible for detecting the objects that need to be tracked. The output of the object detection stage is the aforementioned set, D, containing a number of M detections.

  The stage is visualized in Figure 2.2.
  Common methods for object detection include Faster-RCNN and YOLO [11, 12, 34].

- **Object Description Stage**
  When a set of objects are detected they need to be described. E.g. an object needs a descriptor or some features that distinguishes it from other objects. One can think of a function, that transforms a set of detections D into a set of feature representations as

  $$f_{\text{feature}}(D) = \mathbf{V}_M = \{\mathbf{v}_m | m \in M\} \tag{2.7}$$

  where $\mathbf{v}_m$ is the feature vector for the m'th detection and M is the number of detections.

  The stage is visualized in Figure 2.2.
  Common methods include: Deep appearance models from various CNN architectures, e.g. ResNet-50, and keypoint descriptors, e.g. SIFT. The spatial information of an object can also be used to describe it. [13, 15, 16, 17, 35].

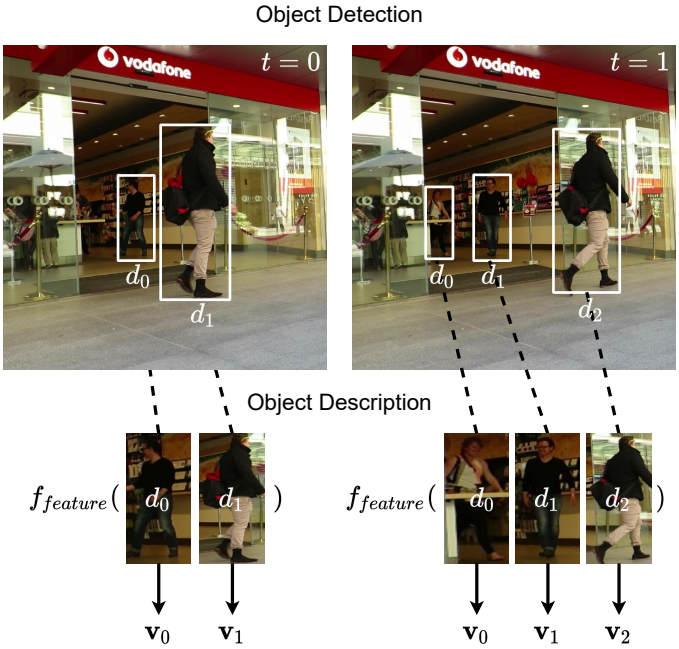


Figure 2.2: The detection and description stage of the tracking-by-detection problem. Two detections are made for frame $t = 0$ and three detections are made for frame $t = 1$. This results in an equal amount of feature vectors for the two frames. The example is from the MOT17 dataset [34].

- **Affinity Stage**
  The goal of the affinity stage is to compute a measure of the similarity between the feature vectors of all the known tracklets and the feature vectors of the detections. The set of feature vectors for the known tracklets is given by

  $$\mathbf{V}_K = \{\mathbf{v}_k | k \in K\}, \tag{2.8}$$

  where K is total the number of known tracklets and $\mathbf{v}_k$ is the feature vector describing tracklet T_k.

  One can think of a function, that calculates the similarity between $\mathbf{V}_k$ and $\mathbf{V}_m$. We can define such a function as

  $$f_{\text{similarity}}(\mathbf{V}_K, \mathbf{V}_M) = S \in \mathbb{R}^{K \times M}, \tag{2.9}$$

  where S is a similarity matrix containing all the similarity measures between the feature vectors. The stage is visualized in Figure 2.3.

  Common methods include: Cosine similarity, Euclidean distance, Mahalanobis distance and siamese networks [16, 32].

- **Object Association Stage**
  When the affinity measures between the different objects are calculated, they need

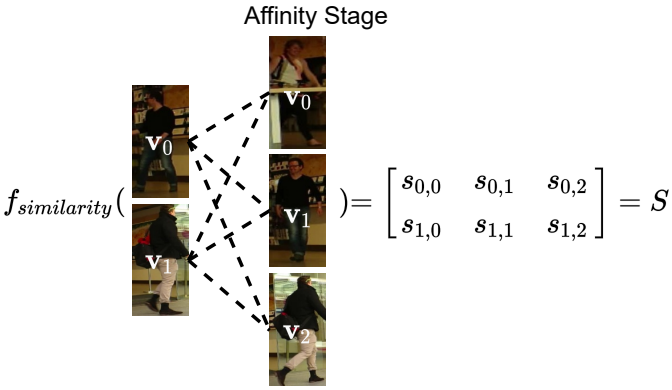


Figure 2.3: The affinity stage. The two feature vectors for $t = 0$ represents two tracklets and the three feature vectors for $t = 1$ represents the three current detections. These are used to calculate their similarity in form of the matrix $S \in \mathbb{R}^{K \times M}$. The example is from the MOT17 dataset [34].

to be associated to construct tracklets. Apart from a similarity measure, associations can also be based on their movement patterns.
Common methods to perform association include: Hungarian algorithm, greedy association, neural solvers, cascade matching [16, 17, 20, 36].

- **Tracklet Handling**
  Tracklet handling includes instantiation of new tracklets and deletion of unused tracklets. Additionally, this may also include motion prediction of the tracked objects.
  Common methods include: Thresholding of the affinity measure $s_{k,m}$ and thresholding on the number of missed associations [15]. Kalman Filter for motion prediction [17]. More advanced tracklet and occlusions handling also exist, although they often become very use-case specific to solve edge cases [23].

Figure 2.4 visualizes a collection of the independent tracking stages.

2.2.3 Online vs. Offline Tracking

Tracking algorithms can be divided into online and offline (batch) methods. Batch methods are allowed to use future information when constructing tracklets, whereas online methods only rely on current and previous information. Thus, offline methods often have better tracking performance, as they can associate the tracks given more information.

On the other hand, online methods are suited for applications that require real-time tracking, e.g. autonomous driving. However, online tracking methods are not necessarily working in real-time. Offline tracking methods are not suited for real-time applications [13].

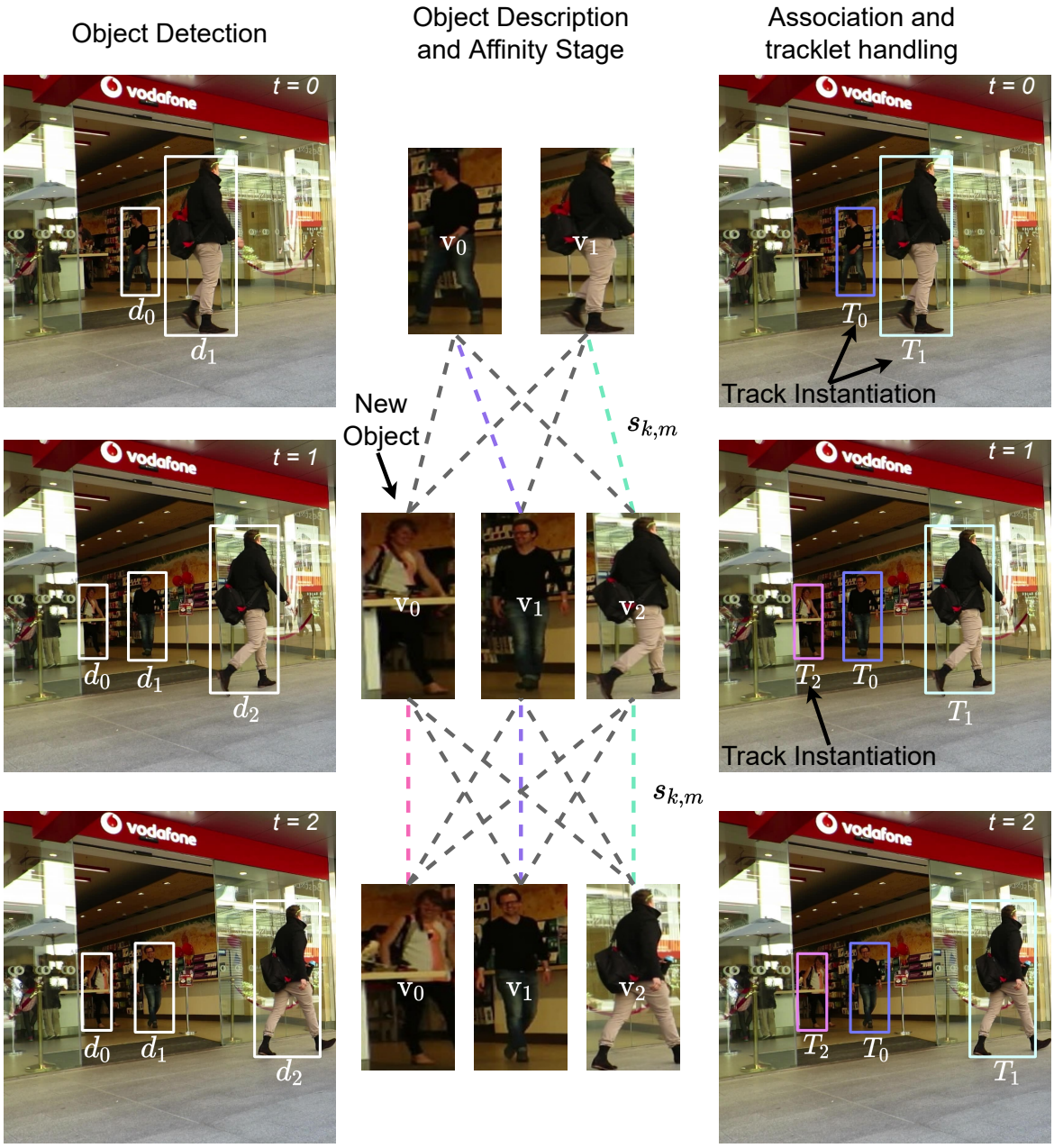


Figure 2.4: The tracking-by-detection problem with the identified tracking stages. The first column shows the object detections for 3 different frames. An affinity measure $s_{k,n}$ is calculated based on the feature vectors for the tracklets, $\mathbf{v}_k$, and the feature vectors for the current detections, $\mathbf{v}_m$, (indicated by the dotted lines). The detections are associated to create the three tracklets, T = $\{T_0, T_1, T_2\}$, given by $T_0 = \{d_0^0, d_1^1, d_1^2\}$, $T_1 = \{d_1^0, d_2^1, d_2^2\}$, $T_2 = \{d_0^1, d_0^2\}$. The example is from the MOT17 dataset [34].

The investigations made in this report will not focus on real-time methods, since tracking with deep learning is computationally heavy and requires optimization. Both online and offline methods will be considered.

2.3 Multi-Object Tracking Challenge

As stated in the introduction, Taixé *et al.* [8] introduced the *MOTChallenge* in 2015 as a benchmark to evaluate tracking algorithms. The MOTChallenge is inspired by other computer vision benchmarks such as ImageNet for image classification [37], Pascal VOC for object detection [38] and builds upon the PETS (Performance Evaluation of Tracking and Surveillance) dataset [39] and the KITTI dataset for autonomous driving [40].

All the MOT datasets consist of videos of pedestrians in an urban environment. The first challenge, MOT15, contained sequences from the PETS and KITTI datasets. However, the sequences were rather simple with a sparse amount of people. With the expansion of MOT16 and MOT17, the scenes became denser in the terms of people and objects in the scene. MOT16 and MOT17 are essentially the same sequences, with MOT17 including deep learning detections. Additionally, the annotation quality improved compared to MOT15. MOT20 includes far denser sequences that result in a high amount of occlusions which imposes higher requirements on the tracker. The latest addition, MOTSynth, is a large synthetic dataset aimed at training for the increasing amount of deep learning methods since they require a large amount of available data. Figure 2.5 includes examples of sequences for the different MOT datasets [8, 34, 41, 9].

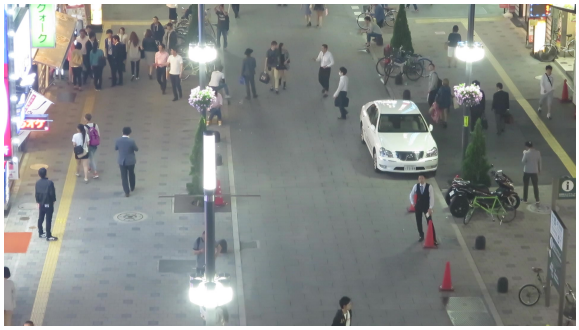
Since its introduction, the challenge has also been expanded with other types of datasets e.g. biology, in terms of cell tracking [7] and tracking of zebrafish [6]. Apart from the MOTChallenge benchmark, several datasets for autonomous driving and traffic applications also exist, including the aforementioned KITTI dataset, Argoverse and nuScenes [40, 42, 43]. Other interesting datasets not currently in the MOTChallenge include the ants dataset with indoor and outdoor sequences [44].

2.3.1 Dataset Structure

The independent datasets of the MOTChallenge are split into a train set and a test set. The train set includes a set of ground-truth annotations, which makes it possible to develop and evaluate tracking algorithms. The videos are given as sequences of *jpg* images, with the name being the frame number.

Both folders include sets of public detections e.g. FRCNN detections for MOT17. This limits the task to the association of detections (tracking-by-detection). It is allowed for participants in the MOTChallenge to use their own detections, however, it must be specified [8]. Figure 2.6 visualizes the folder structure of a MOT dataset.

(a) MOT15-ADL-RUNDLE-6 [8]



(b) MOT16/MOT17-04 [34]

(c) MOT20-01 [41]

(d) Zebrafish-04 Front-View [6]

(e) Ants Indoor Seq0001 [44]

(f) Ants Outdoor Seq0006 [44]

(g) MOTSynth-512 [9]

Figure 2.5: Examples of the MOT datasets (2.5a-2.5c), the Zebrafish Dataset 2.5d, the Ants Datasets (2.5e-2.5f) and MOTSynth 2.5g.

Data Structure

The data structure of the ground-truth annotation files (`gt.txt`) and the detection files (`det.txt`) are similar. They consist of nine comma-separated values, with the six first values being identical for both file types. Each row is an object to be considered. The row structure is given in Equation 2.10 and 2.11.

The first column represents the frame number. The second column is the ID (-1 for detection files). Column three to six hold the bounding box coordinates given as x, y, width and height (x, y, w, h). The seventh column is the confidence score c for detection files. For annotation files, the seventh value is a flag f that indicates, whether the object is considered in the evaluation or not. The eighth value indicates the object class. The ninth and last value indicates the visibility ratio of the object (1 is fully visible). Once again this value is -1 for detection files.

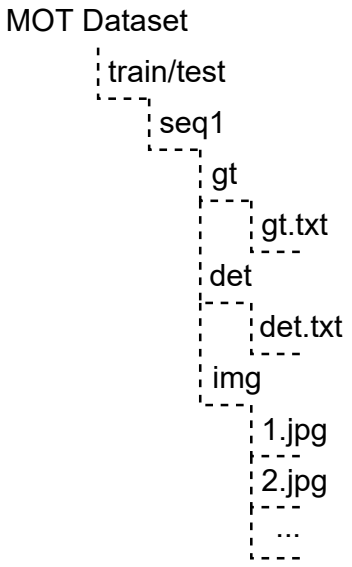


Figure 2.6: Folder structure of the MOT datasets. `gt.txt` is not present in the test split.

$$\texttt{gt.txt} = frame,\ ID,\ x,\ y,\ w,\ h,\ f,\ class,\ visibility \tag{2.10}$$

$$\texttt{det.txt} = frame,\ -1,\ x,\ y,\ w,\ h,\ c,\ class,\ \ -1 \tag{2.11}$$

Splitting of the Training Data

The training set of the dataset must be divided in two, to create a training split and a validation split. The training split is used to train the tracking algorithm, e.g. training of deep learning models or setting threshold parameters, while the validation split is used to validate the algorithms. This separation is important to see how well the tracking algorithm generalizes to the data and avoid overfitting to the training set.

The tracker design of this report will focus on the MOT17-dataset since it includes object detections made from deep learning. The MOT20-dataset also includes deep detections, however, the scenes are very dense and thus increasingly difficult for trackers. The split is chosen arbitrarily as below, but to represent the dataset characteristics, e.g. in terms of scenes with moving cameras as in the sequences 05, 10, 11 and 13.

Training split: MOT17-02, MOT17-04, MOT17-05, MOT17-13
Validation split: MOT17-09, MOT17-10, MOT17-11

Examples of all 14 sequences from the MOT17 dataset are given in Appendix A. For each of the 7 sequences in the training set, a similar sequence is present in the test set.

The MOTChallenge is hosted at `https://motchallenge.net`, which also serves as the evaluation server for the test sequences. When evaluating the tracking algorithms in the MOTChallengs, it is critical to measure the performance with evaluation metrics. The following section discusses the key evaluation metrics of the MOT.

2.4 Evaluation of Multi Object Tracking

Metrics are key in evaluating the performance of independent trackers, especially in a benchmark such as the MOTChallenge. The metrics serve as an objective way to evaluate the best performing tracker. Although non-engineers only need one metric to assess the performance and the relative speed to select a tracking algorithm, several metrics are important to engineers in the development, design and debugging of tracking algorithms [45].

However, the problem of selecting one performance metric is not immediate, since it is difficult to claim a ground truth. As a result of this, different metrics emphasise either the detection accuracy or the association accuracy when evaluating MOT. Therefore, this section introduces common error types in tracking as well as the three metrics CLEAR MOT, IDF1 and HOTA. The introduction will focus on their advantages and disadvantages [46].

2.4.1 Error Types in MOT

Common error types in MOT can occur in different stages of the tracker and includes *detection errors*, *localization errors* and *association errors*.

Detection Errors

Detection errors relate to the performance of the object detection stage. They can occur as a false-positive (FP), which is the detection of an incorrect object class or a non-annotated object. False-negative (FN) is when the detector fails to detect an object or incorrectly classifies an object. An object is classified as a true positive (TP) when the object is classified correctly and the intersection-over-union (IoU) of a detection with the annotation exerts a certain IoU threshold, λ_{IoU}.

Localization Errors

Localization errors are also related to detections since it defines a tracker's ability to track the spatial location of an object. The amount of localization errors made by a tracker inflicts its precision.

Association Errors

The last major error type is association errors. Association errors occur in the construction of tracklets, e.g. wrongly assigning IDs to correctly detected objects. This can occur either as mismatched detections or ID switches. A tracker can only make an association error if objects are detected correctly.

The problem of defining one single evaluation metric for tracking algorithms is the problem of weighing the three error types. The following describes the most common evaluation metrics and their weighing of the error types [46, 47].

2.4.2 CLEAR MOT

Bernadin and Stiefelhagen [48] introduced the CLEAR MOT metrics in 2008. They were used as the standard metrics in the establishment of the MOTChallenge in 2015. The metrics consist of two performance metrics; Multi-Object Tracking Precision (MOTP) and Multi-Object Tracking Accuracy (MOTA).

MOTP addresses localization errors and is defined in Equation 2.12.

$$MOTP = \frac{\sum_{i,t} e_{i,t}}{\sum_t c_t}, \tag{2.12}$$

$e_{i,t}$ is the of precision error for match i at time t. c_t is the total number of matches for frame t. A match refers to a match between a ground-truth annotation and a prediction made by the tracker.

MOTA is defined in Equation 2.13 and addresses detection and association errors.

$$MOTA = 1 - \frac{\sum_t (\mathrm{FN}_t + \mathrm{FP}_t + \mathrm{mme}_t)}{\sum_t g_t}, \tag{2.13}$$

where FN_t is the number of false-negatives (missed detections). FP_t is the number of false-positive detections. mme_t is the number of mismatched detections (ID switches). g_t is the number of ground truth detections. t is the frame number [13] [48].

The twofold metric thereby has MOTP to assess the precision of tracklets and MOTA to assess the association of detections into tracklets. An issue with the MOTA metric is that it weighs mismatched detections the same as false-positives and missed detections, although the mismatched detections can be more critical in some applications [47].

2.4.3 Identification F1

Unlike the CLEAR MOT metrics, which handle its matches at a detection level, the Identification F1 (IDF1) uses matches at an association level. Specifically, IDF1 defines Identity True Positives (IDTP) as the number of matches on the overlapping parts of a tracklet. Identity False Positives (IDFP) is the number of incorrect associations and Identity False

Negatives (IDFN) is the number of missed associations over both overlapping and non-overlapping part of tracklets. Based on these, the F1 metric for identification is given in Equation 2.14.

$$IDF1 = \frac{|IDTP|}{|IDTP| + 0.5|IDFN| + 0.5|IDFP|}. \tag{2.14}$$

One of the issues with the IDF1 metric is that it does not consider localization errors and it weighs association errors the highest [46, 49].

2.4.4 Higher Order Tracking Accuracy

The latest introduced evaluation metric and the common standard for evaluating new trackers is the Higher Order Tracking Accuracy (HOTA). HOTA is proposed to accommodate some of the limitations outlined with the CLEAR MOT and IDF1 metrics and thus is a complex evaluation metric taking into account several different aspects.

HOTA is defined as a double Jaccard index as given in Equation 2.15 and is calculated for a specific λ_{IoU}.

$$HOTA_{\lambda_{\text{IoU}}} = \sqrt{\frac{\sum_{c \in TP} \mathcal{A}(c)}{|TP| + |FN| + |FP|}}, \tag{2.15}$$

where TP is the number of true positive detections, FN is the number of false negative detections and FP is the number of false positive detections for a given λ_{IoU}. $\mathcal{A}(c)$ is given in Equation 2.16 and defines whether the TP detections are assigned correctly.

$$\mathcal{A}(c) = \frac{|TPA(c)|}{|TPA(c)| + |FNA(c)| + |FPA(c)|} \tag{2.16}$$

$TPA(c)$ is the number of True Positive Associations, $FNA(c)$ is the number of False Negative Associations and $FPA(c)$ is the number of False Positive Associations. All the three are calculated for the TP detections only. Thus, the tracker is only evaluated on the associations it makes for correctly detected objects.

To summarize, $TPA(c)$ is the number of TP detections for which IDs have been assigned correctly and $FNA(c)$ and $FPA(c)$ are the number of TP detections that have been wrongly assigned or did not correspond to an object [46].

However, this metric in itself does not take into account the localization errors. HOTA incorporates this, by calculating the HOTA metric for a set of 19 different IoU thresholds, $\lambda_{\text{IoU}} \in \{0.05, 0.10, \ldots 0.95\}$ and average the score. The final HOTA metric is defined in Equation 2.17.

$$HOTA = \frac{1}{19} \sum_{\lambda_{\text{IoU}} \in \{0.05, 0.10, \ldots 0.95\}} HOTA_{\lambda_{\text{IoU}}} \tag{2.17}$$

To this end, HOTA is the current metric that best weighs all three error types, thus it will be used to evaluate the performance in the MOTChallenge [46].

2.5 Contributions and Problem Statement

This chapter formally defined the tracking-by-detection problem and identified the five tracking stages: Object detection, object description, affinity calculation, object association and tracklet handling. The MOTChallenge is introduced as the common benchmark to evaluate MOT algorithms. The MOT17 dataset has been divided into a training and a validation set that will be used to develop tracking frameworks. The evaluation of a tracking algorithm will primarily use the HOTA metric to evaluate the performance, as it weighs the error types in tracking equally.

The analysis of related work showed how tracking algorithms utilize deep learning in various ways to create robust MOT. As an extension to the tracking-by-detection, joint tracking and detection frameworks aim to modify the head of detector networks, to predict the following position of an object. Siamese networks are often used as re-identification modules or for single-object tracking. To do so, the majority of these methods require specific training of the tracker, to be able to track and differentiate between objects in video sequences. As a consequence, it makes the trackers unable to generalize to other types of data, since they are often trained for specific types of sequences and only validated on these sequences.

2.5.1 Problem Statement

Based on the introduction and the analysis of MOT the following problem statement is created:

How can deep learning be used to create a robust description of detected objects, that can be used to distinguish and associate objects in MOT?

Contributions

While answering the above problem statement the report will make the following three contributions. Firstly, an investigation of a novel idea for deep learning in tracking is made. The idea takes its base in the tracking-by-detection framework, where it aims to distinguish objects by masking out the background of detections. The deep feature representation aims to incorporate both the spatial and textural information of a detection. Secondly, the capabilities of the deep keypoint extractor SuperPoint and its deep matching framework SuperGlue to describe and distinguish detections in MOT are investigated. Thirdly, the viability of using the masked out background approach as a global description of objects and SuperPoint as a local description to aid in the re-identification of lost objects is explored.

2.5.2 Report Structure

The structure of the report is given as follows:

- Chapter 3 establishes a baseline tracking method based on the tracking-by-detection framework and a deep appearance model. It investigates the influence of various affinity measures and association approaches.
- Chapter 4 explains the proposed idea to incorporate the spatial and textural information in a deep object description and evaluates its performance.
- Chapter 5 investigates the viability of using the SuperPoint keypoint descriptor for object description and its matching framework SuperGlue to match detections.
- Chapter 6 combines the findings from the proposed idea and the SuperPoint trackers to construct two combined tracking frameworks.
- Chapter 7 evaluates the combined tracking strategies both on the MOT17 test sequences and other MOT datasets.
- Chapter 8 discusses the obtained results and identifies the strengths and weaknesses of the investigated methods.
- Chapter 9 concludes with the main findings and proposes future work areas.

Chapter 3

Baseline Tracker

Inspired by classical tracking-by-detection frameworks such as DeepSORT, a baseline tracker is created to evaluate the performance of developed tracking frameworks. In short, the baseline tracker revolves around deep feature extraction and association of the detected bounding boxes. It compares two different types of feature extractions MobileNetV3 and EfficientNetV2.

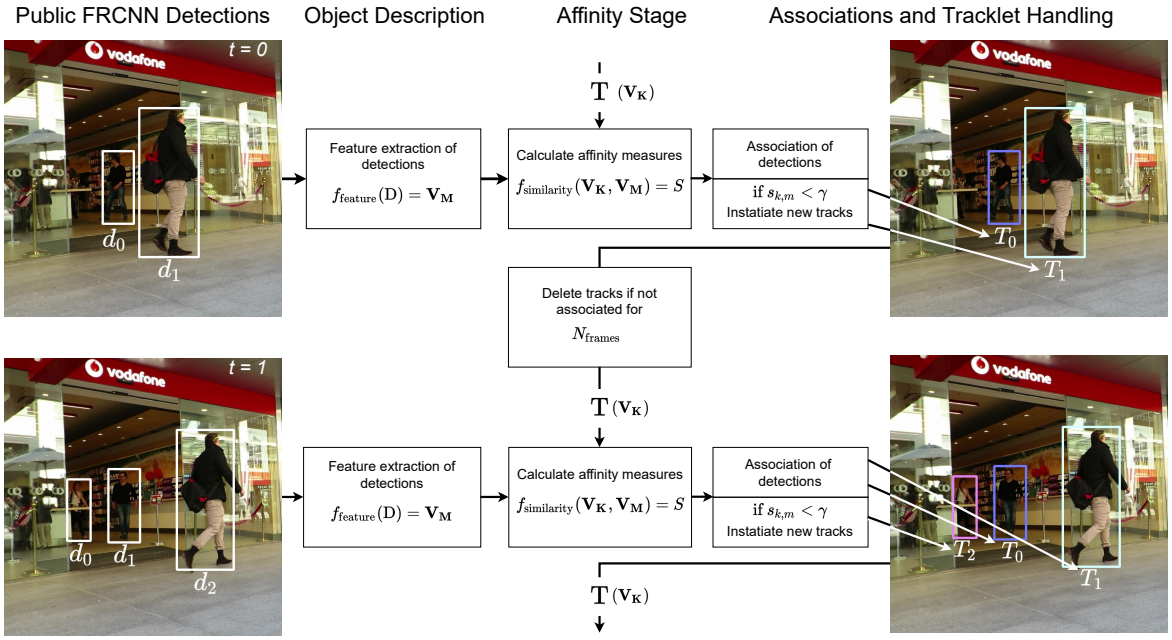


Figure 3.1: The structure of the baseline tracker. Public FRCNN detections are used as detections to extract a set of features, $\mathbf{V}_M$, for each frame. An affinity measure is calculated between the obtained feature representations and the current tracklets. The Hungarian and greedy algorithms perform associations. New tracklets are instantiated if the similarity measure is less than a given threshold γ and tracklets are deleted if they are not associated for a number of frames, N_{frames}. The example is from the MOT17 dataset [34].

3.1 Tracking Approach

The tracking approach follows the identified tracking stages from Section 2.2.2. Figure 3.1 visualizes the structure of the baseline tracker. The specifics of the independent stages are described below.

The tracker assumes that detections only move slightly in-between frames and hence it does not contain any motion prediction nor re-identification of any sort. It is distinct from DeepSORT since it does not consider an object's spatial location when making the association, thus only evaluating the deep feature description of object detections [16].

3.1.1 Object Detection

For object detections, the public FRCNN detections of pedestrians provided in the MOT17 dataset are used. To ensure a high detection quality, only detections with $p_{object} > 0.5$ are used.

3.1.2 Feature Extraction - Object Description

Feature extraction on the detected bounding boxes has been performed using the MobileNetV3-Large network and the EfficientNetV2-Small. Table 3.1 compares the main parameters.

MobileNetV3-Large

The MobileNetV3-Large network is pretrained on the ImageNet dataset and has an input shape of (224x224) [50]. The deep network performs the following feature extraction of a detection, d:

$$f_{\text{MobNet}}(d) = \mathbf{v} = (v_1,\ v_2 \cdots v_{960}), \tag{3.1}$$

A feature vector $\mathbf{v}$ is extracted with a total of 960 features. The TensorFlow implementation of MobileNetv3 is used [51].

EfficientNetV2-Small

The EfficientNetV2-Small is also pretrained on the ImageNet classification dataset and has an input shape of (384x384) [52]. The network performs the following feature extraction of a detection, d:

$$f_{\text{EffV2S}}(d) = \mathbf{v} = (v_1,\ v_2 \cdots v_{1280}), \tag{3.2}$$

thus it extracts a total of 1280 deep features from the detection. The pretrained version by Google from TensorFlow Hub is used [53].

Network	Parameters	Input Shape	Top1 acc*
MobileNetv3-L	5.4 M	224x224	75.2 %
EfficientNetv2-S	21.5 M	384x384	83.9 %

Table 3.1: The table compares the parameters of the two used feature extractors for the baseline tracker. * is the top1 % accuracy on the ImageNet1k dataset [50] [52].

3.1.3 Affinity Measures

An affinity measure is calculated based on two approaches: Cosine similarity as given in 3.3 and euclidean distance as given in 3.4.

$$c(\mathbf{v}_k, \mathbf{v}_m) = \frac{\mathbf{v}_k \cdot \mathbf{v}_m}{\|\mathbf{v}_k\| \, \|\mathbf{v}_m\|} \tag{3.3}$$

$$e(\mathbf{v}_k, \mathbf{v}_m) = \|\mathbf{v}_k - \mathbf{v}_m\|, \tag{3.4}$$

where c and e represents the cosine similarity and euclidean distance, respectively. $\mathbf{v}_k \cdot \mathbf{v}_m$ is the dot product between the two vectors.

$\mathbf{v}_m$ is the feature vector of the m'th detection and $\mathbf{v}_k$ is the feature vector of the k'th tracklet.

3.1.4 Association

The baseline tracker is evaluated on two different ways of performing associations, greedy association and Hungarian association.

Greedy Association

The greedy association solves the association problem by making a locally optimal choice for associating detections to tracklets. Recall, that the similarity functions establishes an affinity matrix $S \in \mathbb{R}^{K \times M}$, with the independent similarity measures as given in Equation 3.5.

$$S = \begin{bmatrix} s_{0,0} & \dots & s_{1,M} \\ \vdots & \ddots & \vdots \\ s_{K,1} & \dots & s_{K,M} \end{bmatrix} \tag{3.5}$$

The greedy association then follows Algorithm 1. Thereby making a local optimal choice of assigning the m'th detection to the k'th tracklet.

Hungarian Association

The Hungarian association solves the association problem different from the greedy algorithm, by making the assignment choice that achieves the lowest global cost. The algorithm

Algorithm 1 Greedy Association

while `size` $(S) > 0$ **do**
  $k, m =$`argmin`(S) ▷ Extract the indexes of the most similar value
  $T_k \leftarrow d_m$ ▷ Assign detection m to tracklet k
  $S =$ `shrink`(S, k, m) ▷ Remove row k and column n from S
end while

Notice, that the algorithm is written for the euclidean distance, since the smallest similarity value represents the most similar detection. The algorithm is valid for cosine similarity by using `argmax(S)` instead of `argmin(S)`

works on a cost matrix, which is chosen to be the affinity matrix S. Thus, applying the Hungarian algorithm on S assigns detections m to tracklets k and achieves the lowest global cost.

The implementation uses the python library *munkres* [54], which is an implementation of the Kuhn-Munkres variant of the Hungarian algorithm [55].

3.1.5 Tracklet Handling

Tracklet handling is done by deleting tracklets that have not been associated for a threshold of $N_{\text{frames}} = 10$.

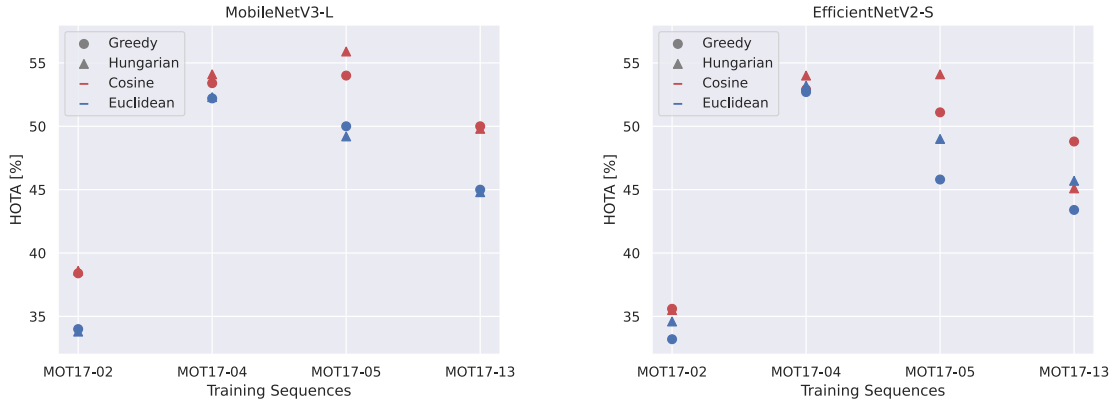
New tracklets are instantiated based on a similarity threshold, γ, on the affinity measures. For cosine similarity new tracklets are instantiated when $c < \gamma_c$ and for the euclidean distance when $e > \gamma_e$.

The similarity thresholds are chosen based on experiments with the affinity measures on the ground-truth annotations of the training split, as described in Appendix B. N_{frames} is chosen by trial and error to achieve as high a score HOTA as possible.

3.1.6 Variations

In summary these variations in feature extraction, affinity measure and association leads to a total of eight different baseline trackers to be evaluated. A summary of the variations are given in the listing below.

- **Feature Extraction**
  MobileNetV3-L v EfficientNetV2-S
- **Affinity Measure**
  Cosine similarity v Euclidean Distance
- **Association**
  Greedy Association v Hungarian Association



(a) The results for euclidean distance use $\gamma_e = 16$. The results for cosine similarity use $\gamma_c = 0.85$.

(b) The results for euclidean distance use $\gamma_e = 14$ for hungarian association and $\gamma_e = 15$ for the greedy association. The results for cosine similarity use $\gamma_c = 0.85$.

Figure 3.2: The baseline HOTA scores for the two different feature extractors.

3.2 Results

The results for the variations of the baseline tracker are given for the four training sequences: MOT17-02, MOT17-04, MOT17-05 and MOT17-13. The HOTA scores for the MobileNetv3-L and the EfficientNetV2-S feature extractors are given in Figure 3.2a and Figure 3.2b, respectively. Detailed results for the three different metrics HOTA, MOTA and IDF1 are given in Appendix D.1.1.

Figure 3.3 shows the results of the best performing tracker on the validation sequences. Example videos for this baseline tracker are given at: Baseline.

3.3 Findings

In establishing the baseline tracker a few conclusions can be made. Overall, the baseline results of both feature extractors perform well on the majority of the training sequences except MOT17-02. All the other training sequences perform well with scores around and above 50 % HOTA, taking into account that the baseline tracker is a simple tracker without motion prediction, re-identification module etc.

The results obtained using cosine similarity (red) are better than those obtained with euclidean distance (blue) for the majority of the sequences.

The difference between greedy (circles) and Hungarian (triangles) association is small, with a slight advantage over the Hungarian method. The two feature extractors perform similarly with a slight advantage to the MobileNetv3-Large on the MOT17-02 and MOT17-05 sequences. Hence, the best performing baseline tracker is the MobileNetV3-Large fea-

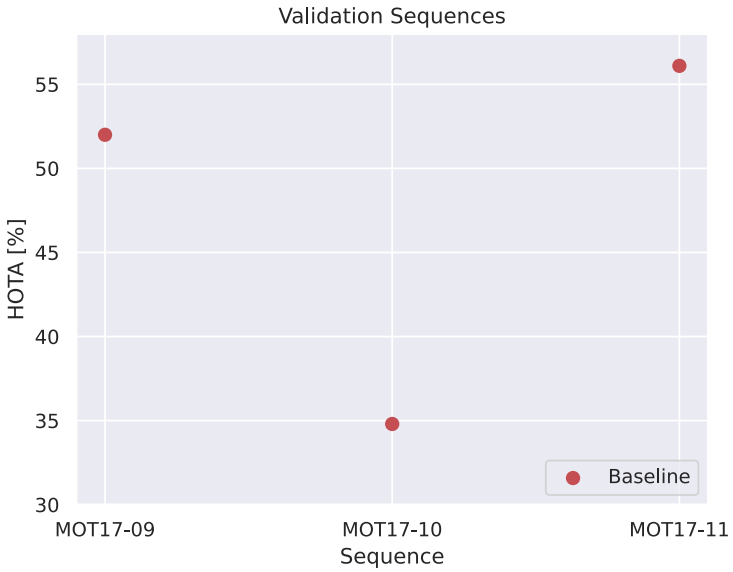


Figure 3.3: The HOTA scores for the best performing baseline tracker on the validation sequences. MobileNetV3-Large feature extraction with cosine similarity and Hungarian association. $\gamma_c = 0.85$ and $N_{\text{frames}} = 10$.

ture extractor with Hungarian association and cosine similarity as an affinity measure. It achieves an average HOTA score of 49.6 % on the training sequences. The tracker achieves an average HOTA score of 47.6 % on the validation sequences, thus showing its ability to generalize.

The following chapter shows the performance of the proposed idea for a deep feature description evaluated against the baseline tracker.

Chapter 4

Proposed Method

The following chapter contains an introduction to a proposed base method for tracking with deep learning. At first, the approach is described taking starting point in the aforementioned tracking stages.

To validate the approach and discover advantages and limitations, experiments are conducted to compare various feature extractors and variations of the proposed method.

4.1 Masking Out the Background

The idea for a base method consists of a modified way of describing the detections. In short, the approach aims to focus a deep feature extractor by masking out the background of object detections, thus maintaining the spatial position of the detection in the frame. The approach is visualized in Figure 4.1 and is referred to as the masked out background (MOB) approach.

- **Object Detection** For each frame detect a set of objects, D. The public FRCNN detections from the MOT17 sequences are used with $p_{object} > 0.5$.
- **Object Description** For each detected object, mask out the background such that it results in a distinct image, denoted β_m (See Figure 4.1). This set of images with a detection and a masked out blackground is denoted as:

$$\mathrm{B} = \{\beta_m | m \in M\}, \tag{4.1}$$

with M being the number of detections. Similarly, to the object description stage we can apply a deep neural network on this set of of images to extract a set of deep feature vectors:

$$f_{\text{feature}}(\mathrm{B}) = \mathbf{V}_M = \{\mathbf{v}_m | \in M\} \tag{4.2}$$

Conclusively, each object detection has a deep feature representation.

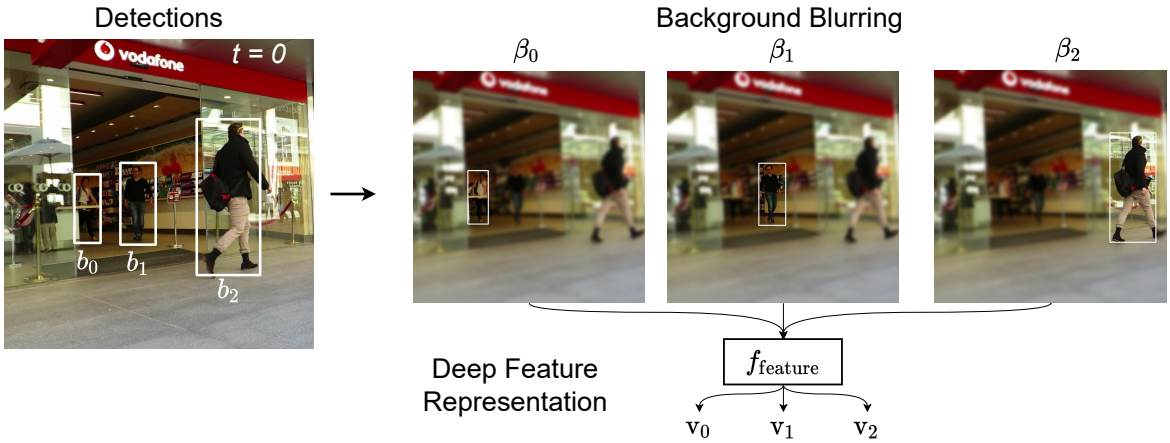


Figure 4.1: For each of the three detected bounding boxes, mask out the background (blurring in this case) and perform deep feature extraction to obtain 3 feature vectors. For the three frames with background blurring, the white marking box showing the detection is not included in the actual feature extraction. [34].

The general intuition behind the MOB approach is to incorporate all the bounding box information in the deep feature extraction. First of all, this covers the textural information of the bounding box, e.g. what is in the bounding box. This is somewhat in line with the previous usages of deep learning in MOT (DeepSORT [16]). Second of all, this also covers the spatial information of the detected bounding box, e.g. where and what is the size of the bounding box. It aims to allow the deep feature representation to implicitly contain all of the information in the detected box and thereby use the blurring to focus the feature extraction. To the best of the author's knowledge, this method has not been investigated for MOT previously.

The experiments will compare two variations of MOB; using a blurred background and using a black background.

4.1.1 Feature Extraction (Object Description)

Feature extraction is performed with three different deep feature extractors, the MobileNetV3-L, the EfficientNetV2-S and the EfficientNetV2-M. The variations of MobileNetv3-Large and EfficientNetV2-S are identical to the ones introduced for the baseline tracker in Section 3.1.2. Table 4.1 compares their parameters.

EfficientNetV2-M

The EfficientNetV2-M is used in addition to the smaller variation, since it is pretrained for an even larger input shape (480x480) which may be beneficial for the proposed method. The network performs the following feature extraction on a detection with a masked out background, b:

$$f_{\text{EffV2M}}(b_m) = \mathbf{v}_m = (v_1,\ v_2 \cdots v_{1280}) \tag{4.3}$$

It extracts a total of 1280 features from an image with the background masked out. The version pretrained on ImageNet by Google from TensorFlow Hub is used [53].

Network	Parameters	Input Shape	Top1 acc*
MobileNetv3-L	5.4 M	224x224	75.2 %
EfficientNetv2-S	21.5 M	384x384	83.9 %
EfficientNetv2-M	54.1 M	480x480	85.2 %

Table 4.1: The table compares the parameters of the two used feature extractors for the baseline tracker. * is the top1 % accuracy on the ImageNet1k dataset [50] [52].

4.1.2 Affinity Measures

The cosine similarity is used as an affinity measure for the proposed method, since it achieved the best results on the baseline tracker. The cosine similarity is given below (same as Equation 3.3).

$$c(\mathbf{v}_k, \mathbf{v}_m) = \frac{\mathbf{v}_k \cdot \mathbf{v}_m}{\|\mathbf{v}_k\| \, \|\mathbf{v}_m\|} \tag{4.4}$$

4.1.3 Association

The proposed method is performed both with greedy and Hungarian association, as introduced in Section 3.1.4. This is done, to compare whether the MOB approach benefits from a local or a global association approach.

4.1.4 Variations

In summary, the different variations in feature extraction and association gives the following combinations to be evaluated for the proposed method:

- **Feature Extraction (Object Description)**
  Blurred Background v Black Background
  MobileNetV3-L (224x224) v EfficientNetV2-S (384x384) v EfficientNetV2-M (480x480)
- **Association**
  Greedy Association v Hungarian Association

4.1.5 Tracklet Handling

All of the variations use $N_{frames} = 5$ to delete unassigned tracklets. The similarity thresholds are chosen based on the affinity experiments from Appendix B.2 to select the parameters for tracklet handling that achieves the best results on the training sequences. Hence, they differ slightly for the variations.

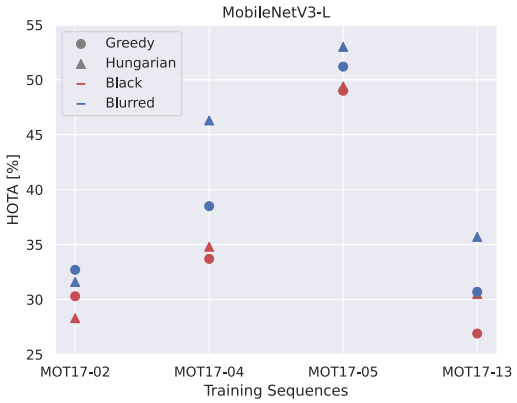


Figure 4.2: Results for MobileNetV3-L (224x224). Thresholds are $N_{\text{frames}} = 5$ and $\gamma_c = 0.85$

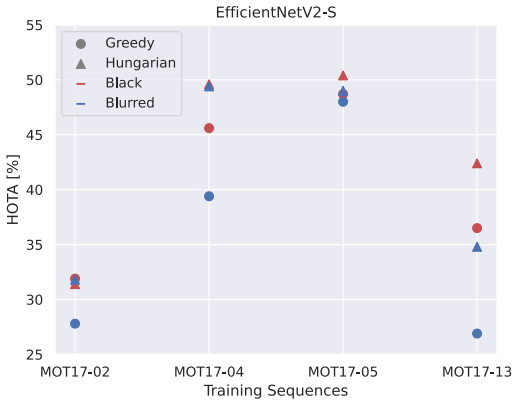


Figure 4.3: Results for EfficientNetv2-S (384x384). $N_{\text{frames}} = 5$. The black version use $\gamma_c = 0.85$ and the blurred version use $\gamma_c = 0.88$.

4.2 Results

The results for the three different feature extractors on the training sequences are given in Figure 4.2, 4.3 and 4.4. Results on the validation sequences are given in 4.5.

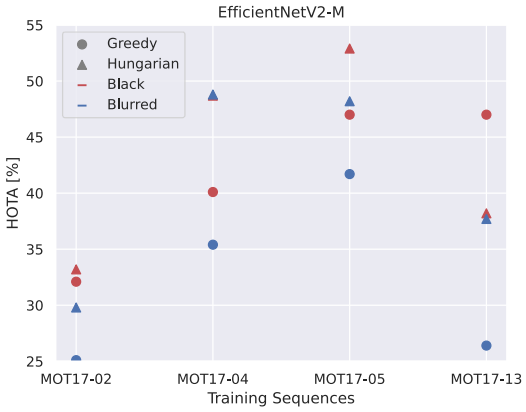


Figure 4.4: Results for EfficientNetv2-M (480x480). $N_{\text{frames}} = 5$. The black version use $\gamma_c = 0.85$ and the blurred version use $\gamma_c = 0.88$.

Example videos of the two best performing trackers from Figure 4.5 are given at: EfficientNetV2-M EfficientNetV2-S

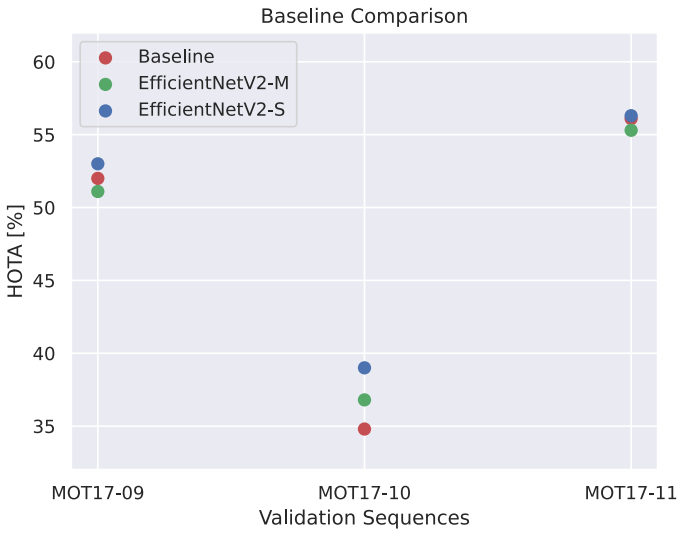


Figure 4.5: The EfficientNetV2-S and M with black background and hungarian association compared to the baseline results achieved previously.

4.3 Findings

As a general trend across all the variations, the Hungarian association outperforms the greedy association. The only exception to this trend is the MOT17-02 sequence using MobileNetV3-L and the MOT17-13 sequence for EfficientNetV2-M. Thus, the following only compares the Hungarian association.

The feature extractors perform rather similar on the MOT17-02 sequence. Both the EfficientNet feature extractors perform the best on the MOT17-04 and MOT17-13 sequence, whereas the MobileNetV3-L performs the best on the MOT17-05 sequence. For the MobileNetV3-L feature extractor the blurred background performs the best ($\mu_{HOTA} = 38.7\%$), whereas the black background performs the best for both of the EfficientNet networks with $\mu_{HOTA} = 43.4\%$ for the small version and $\mu_{HOTA} = 43.5\%$ for the medium version).

However, the EfficientNetV2-S generalizes the best to the validation sequences with $\mu_{HOTA} = 49.5\%$ compared to $\mu_{HOTA} = 47.7\%$ for the EfficientNetV2-M. Both trackers perform better on the validation sequences than the baseline which achieves $\mu_{HOTA} = 47.6\%$. Thus, the results suggest that the MOB approach can perform comparatively and generalize slightly better to the validation sequences, although it only uses a simple tracking framework that does not require training. Due to the simple tracking framework, it struggles with occlusions and re-identification of lost tracklets. In the following chapters, the efficientNetV2-S feature extractor with a black background and Hungarian association is referred to as the MOB tracker.

4.3.1 Problems With Small Detections

The MOB tracker achieves low performance on the MOT17-02, 10 and 13 sequences since they include scenes with smaller background detections. Examples of these sequences are given in Figure 4.6. However, it is a well-known problem that the trackability decreases with the object size [33].

(a) MOT17-02

(b) MOT17-13

Figure 4.6: Two examples of the worst performing sequences for the MOB tracker with small detections [34].

Appendix C describes additional experiments that have been attempted without essential success to improve the performance of the MOB tracker e.g. on the sequences with smaller detections.

Chapter 5

SuperPoint as a Tracker

This chapter investigates the use of SuperPoint and SuperGlue to describe and match objects in MOT. At first, SuperPoint and SuperGlue are described along with relevant parameters. Thereafter, steps to adapt the SuperPoint framework to a tracker are described along with results on the MOT17 sequences.

5.1 SuperPoint

DeTone *et al.* [56] from Magic Leap introduced SuperPoint in 2018 as a CNN based keypoint extractor trained to extract both points of interest and corresponding descriptors. The network structure of SuperPoint is visualized in Figure 5.1. The network utilizes a shared encoder and has two separate decoders to extract keypoints and descriptors.

5.1.1 Encoder

The input to the encoder is a gray-scale image $I \in \mathbb{R}^{H \times W}$. The encoder is inspired by the VGG-nets and consists of 8 convolutional layers with the ReLU activation function to introduce non-linearity. Three max-pooling operations reduce the output shape of the decoder with $2^3 = 8$, thus the output shape of the encoder is $H_e = \frac{H}{8}$ and $W_e = \frac{W}{8}$ with a channel depth of F. In other words, the decoder outputs a tensor $E \in \mathbb{R}^{H_e \times W_e \times F}$, which is used as an input to the two decoders.

5.1.2 Keypoint Decoder

The keypoint decoder starts with a convolutional head that converts the encoded tensor E to have a fixed channel depth of 65, $P \in \mathbb{R}^{H_e \times W_e \times 65}$. The first 64 channels correspond to an 8×8 pixel encoding of the image and the last channel represents a bin for non-interest points. A channel-wise soft-max operation is applied to P and the last channel is removed. The corresponding $\mathbb{R}^{H_e \times W_e \times 64}$ tensor is reshaped to $\mathbb{R}^{H \times W}$. Hence, the output

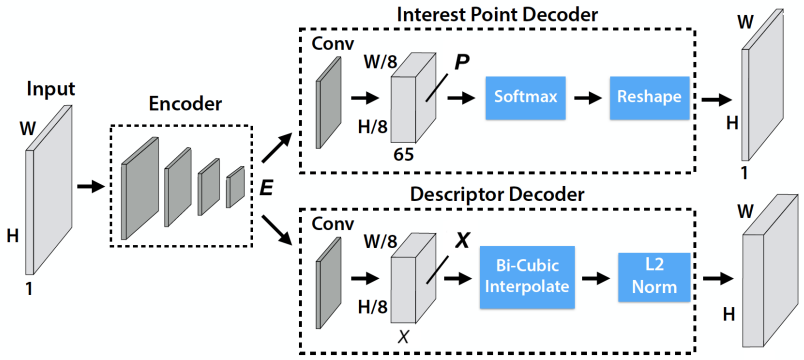


Figure 5.1: The architecture of the SuperPoint network. A shared VGG-inspired encoder encodes the input image to a tensor. Through two separate decoders two grids of interest points and their corresponding descriptors are created. Adopted from [56].

of the interest point decoder is a pixel map with keypoint point-ness scores for each pixel in the input image I.

5.1.3 Descriptor Decoder

The convolutional head of the descriptor decoder computes a tensor of shape $X \in \mathbb{R}^{H_e \times W_e \times X}$, where X is the dimensionality of the descriptor, which in this case is 256. X then represents a smaller and not fully dense grid of descriptors, obtained from the encoded tensor E. To obtain a dense descriptor grid with descriptors for each point in the image, Bi-Cubic interpolation is used. Finally, an L2-normalization step ensures a fixed length of all descriptors.

Both decoders use a non-learned way of up-sampling to a dense point and descriptor grid with reshaping and Bi-Cubic interpolation, respectively. This is done to keep the network fast and efficient.

To this end, one can think of SuperPoint as a function that transforms a detection, d_m, into a set of keypoints and associated descriptors, similarly to the deep feature extractors used previously, see Equation 5.1.

$$f_{\text{SuperPoint}}(d_m) = \{\mathbf{p}_m, \mathbf{x}_m\} \tag{5.1}$$

here $\mathbf{p}_m \in \mathbb{R}^{N_{\text{keypoints}} \times 2}$ is a vector containing the keypoints and $\mathbf{x}_m \in \mathbb{R}^{N_{\text{keypoints}} \times 256}$ is a vector containing descriptors.

5.1.4 Adapting Parameters

For SuperPoint the default model weights obtained from the GitHub page are used [57].

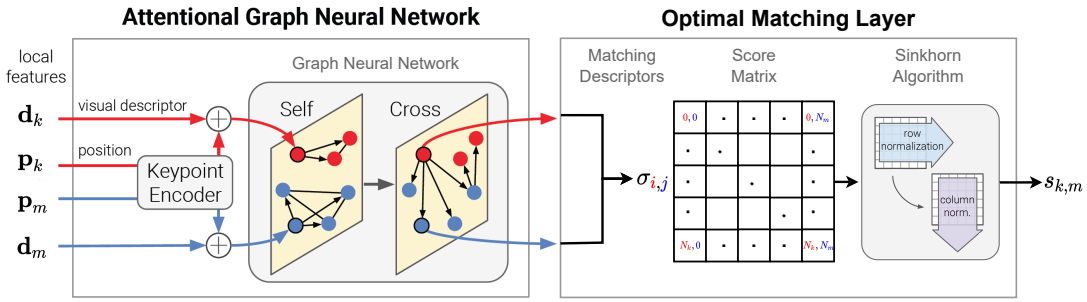


Figure 5.2: The SuperGlue matching framework. The input is the keypoint and descriptors for the tracklets $\{\mathbf{p}_k, \mathbf{x}_k\}$ and detections $\{\mathbf{p}_m, \mathbf{x}_m\}$ and the output is a sum of the matching scores $s_{k,m}$ for the optimal match of keypoints found with the Sinkhorn Algorithm. N_k is the number of keypoints for the tracklet and N_m is the number of keypoints for the detection. Adopted from [58].

SuperPoint is designed to extract and describe keypoints of full-size images, thus modifications of the default parameters are needed to fit it for tracking. The default input shape for SuperPoint is a (640x480) image, which is changed to (240x480) to fit with the approximate shape of the detected pedestrians.

Experiments have been conducted with choosing a larger input shape. The larger the input shape, the more keypoints are extracted from the images which is desirable. On the other hand, SuperPoint loses its ability to properly describe the additionally extracted keypoints, if detections are scaled up too much from the original size.

When downscaling the input shape from the default shape, it is also necessary to reduce the non-maximum suppression (NMS) ratio. The NMS ratio is reduced from 4 to 2 pixels, which allows the extracted keypoints to reside closer to each other.

The threshold for the maximum amount of keypoints and the keypoint point-ness score are left at their default values of 1024 and 0.005, respectively [56].

5.2 SuperGlue

In addition to the SuperPoint keypoint extractor, Sarlin *et al.* proposed SuperGlue as a deep matching framework to match keypoints. The framework is made for matching keypoints with associated descriptors, meaning that it can be used both for SuperPoints and more traditional keypoint descriptors e.g. SIFT and FAST.

The SuperGlue matching framework can be divided into two parts as seen in Figure 5.2. A Graph Neural Network (GNN) is used to predict a scoring matrix $\sigma \in \mathbb{R}^{N_k \times N_m}$ between keypoints. N_k is the number of keypoints for a tracklet and N_k is the number of keypoints for a detection. The GNN considers both intra- and inter-frame keypoints.

Thereafter the Sinkhorn algorithm is used to solve the optimal matching problem between keypoints. This Sinkhorn algorithm is a version of the Hungarian algorithm, that solves the optimal matching problem between keypoints by iterative normalization of $e^{\sigma_{i,j}}$

along rows and columns.

SuperGlue then outputs a set of matched keypoints with an associated matching score, e.g. a probability $[0, 1]$, that the matched keypoint is the same. A matching threshold is used to filter out non-probable keypoints. Thus, summing the matching score for all matched keypoints gives an estimate of the total number of matched keypoints, $s_{k,m}$, between tracklets and detections [58].

One can think of the SuperGlue as a function that calculates the best possible matching scores between two sets of keypoints and descriptors as given in Equation 5.2.

$$f_{\text{SuperGlue}}(\{\mathbf{p}_k, \mathbf{x}_k\}, \{\mathbf{p}_m, \mathbf{x}_m\}) = s_{k,m} \tag{5.2}$$

where the $s_{k,m}$ is an estimate of the number of matched keypoints between a detection m and a tracklet k. This score can then be used as an affinity measure to associate detections to tracklets. The score can in theory achieve a value of up to 1024 if the maximum number of SuperPoints are extracted and perfectly matched. Affinity experiments from Appendix B.3 investigate the number of matches between corresponding and non-corresponding detections.

5.2.1 Adapting Parameters

For SuperGlue the default 'outdoor' model obtained from the GitHub Page is used [59].

The rest of the parameters for SuperGlue are left at their default values. The keypoint threshold for allowing a match is set at matching_score > 0.2, which filters out poor matches. The default number of 20 Sinkhorn iterations is used in the optimization step [58].

5.3 Adapting SuperPoint to be a Tracker

5.3.1 Object Detection

As the previous trackers, the SuperPoint tracker uses the public FRCNN detections with $p_{object} > 0.5$.

5.3.2 Object Description and Affinity Measure

When using SuperPoint and SuperGlue for a tracker, it can replace the object description stage and the affinity stage. SuperPoint is used to extract and describe the detections. SuperGlue performs matching between potential pairs and outputs a score as an affinity measure, that can be used to associate the objects. When a detection is associated with a tracklet, T_k, its keypoints and descriptors are updated, hence associations are always performed based on the latest SuperPoint description. A visualization of the tracking framework is given in Figure 5.3.

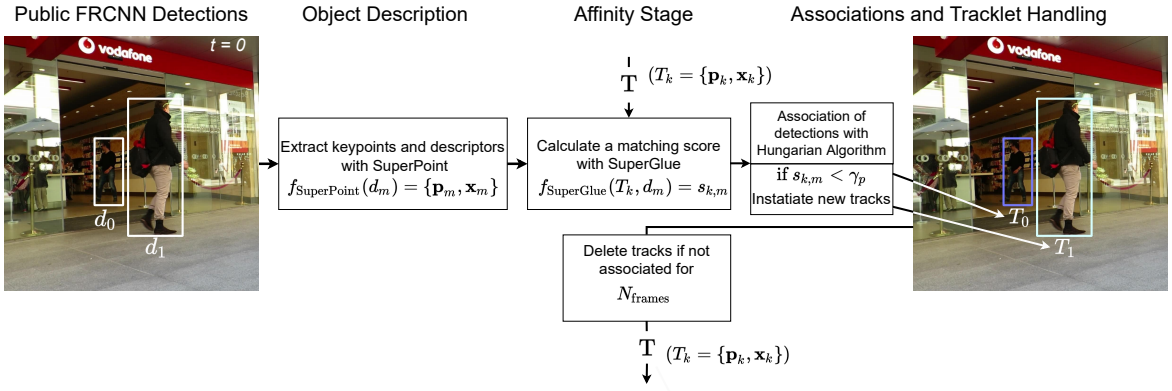


Figure 5.3: The tracking framework that utilize SuperPoint and SuperGlue in the object description stage and affinity stage, respectively. New tracks are instantiated if the matching score is less than a threshold γ_p and tracks are deleted if not associated N_{frames}. The example is from the MOT17 dataset [34].

One issue with the SuperGlue algorithm is that all potential pairs are required to pass through SuperGlue's GNN to predict the costs. This makes the processing time grow quadratically with the number of potential pairs. To reduce the amount of processing, a search region for SuperGlue is implemented to only calculate a matching score if the pair is within the region. The region is given as: $e_{k,m} < 0.1$, where the e is the euclidean distance calculated between the normalized center points of d_m and T_k. This is used to ignore matching scores outside of the search region as given in Equation 5.3.

$$s_{k,m} = \begin{cases} s_{k,m}, & \text{if } e_{k,m} < 0.1 \\ 0, & \text{otherwise} \end{cases} \tag{5.3}$$

5.3.3 Association

Similar to the previously introduced trackers, the Hungarian algorithm is used to associate the tracklets.

5.3.4 Tracklet Handling

Tracklet handling is done similar to the previous frameworks. A similarity threshold is used to instantiate new tracks if $\gamma_p > s_{k,m}$ with $\gamma_p = 1$. To this end, we require only 1 matched keypoint to perform an association. Tracklets are deleted if not associated for $N_{\text{frames}} = 10$.

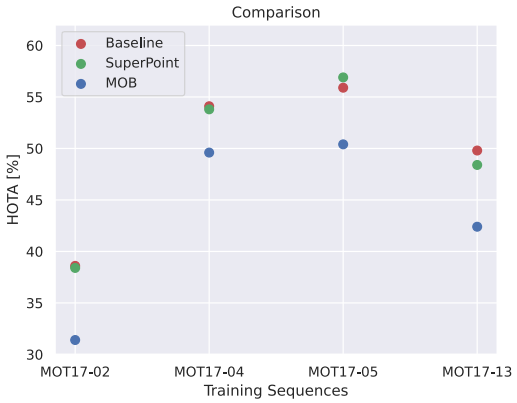


Figure 5.4: Results for the SuperPoint tracker on the training sequences compared to the MOB and baseline trackers.

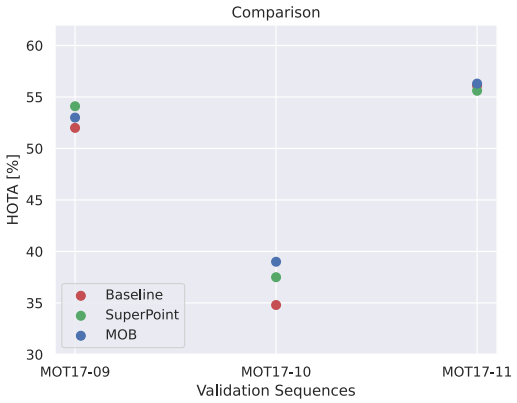


Figure 5.5: Results for the SuperPoint tracker on the validation sequences compared to the MOB and baseline trackers.

5.4 Results

The results for the SuperPoint tracker is given in Figure 5.4 for the training sequences and in Figure 5.5 for the validation sequences. Videos for the SuperPoint tracker are given at: SuperPoint

5.5 Findings

The performance of the SuperPoint tracker performs similar to the baseline tracker and is better than the proposed MOB approach on the training sequences. However, the SuperPoint tracker generalizes better to the validation sequences as it achieves $\mu_{HOTA} = 49.1\%$ compared to $\mu_{HOTA} = 47.6\%$ for the baseline and comparatively to the MOB tracker $\mu_{HOTA} = 49.5\%$. Thus, the results show that SuperPoints and SuperGlue matching can be used as a description and affinity measure that performs better on the validation sequences than a baseline deep learning description.

However, as with the MOB tracker, the framework struggles with occlusion handling and tends to lose tracks after even short term occlusions, when experimenting with a larger N_{frames}. The following chapter investigates how this can be accounted for by combining the various track description methods, to construct a more advanced tracking framework.

Chapter 6

Combining MOB with SuperPoint

The following chapter investigates combining the MOB tracker from Chapter 4 with the SuperPoint framework that proved its tracking viability in the previous Chapter 5.

At first, a description of changes to the track handling strategy is made to boost the tracking performance and enhance the generalizability of the trackers. Thereafter, two tracking strategies are proposed to take advantage of the spatial incorporation from the MOB approach and the robustness of the SuperPoint tracker.

The first strategy keeps the two modules separated, e.g. using the MOB module to associate detections into tracklets and using the SuperPoint module to re-identify lost objects after occlusions. The second strategy merges the two modules to produce a unified cost for associating detections and re-identify lost objects.

6.1 Advanced Tracklet Handling

To enhance the generalizability of the combined trackers the following modifications are created.

6.1.1 Adapting the Similarity Threshold

For previous tracking frameworks, a similarity threshold is used to instantiate new tracklets based on the affinity measure. Recall, that this threshold is γ_c for the MOB tracker. An issue with this is that it can be difficult to generalize this approach since the affinity measure depends on an object's movement speed in a frame and its size.

Thus, an adaptive threshold is proposed to reduce this problem. The adaptive threshold works as a percentage value, of a moving average filter applied on the previous similarity measures. We can write up the threshold γ_c at time t as in Equation 6.1:

$$\gamma_c[t] = \frac{1}{N_{mov}} \sum_{i=0}^{N_{mov}-1} c[t-i] \cdot \alpha \tag{6.1}$$

where N_{mov} is the length of the moving average filter and α is the percentage value [60].

To this end, we require the current affinity measure to be higher than a percentage value of the last N_{mov} averaged similarities to make an association, otherwise a new tracklet is instantiated.

Although this introduces two new variables (α and N_{mov}), these are more intuitive to select and help increase the tracking performance.

6.1.2 Removing the N_{frames} Threshold

Additionally, removing the rigid N_{frames} threshold for deletion of tracklets is desirable to fully allow for objects to be re-identified. As inspired by other tracking algorithms, we want to keep long and more certain tracklets, while deleting short and less certain tracklets. Thus, tracklets are deleted based on the adaptive threshold given in Equation 6.2:

$$N_{\text{frames}} = \min(60, \frac{N_{\text{associations}}}{2}), \tag{6.2}$$

where 60 is the maximum number of frames to keep a tracklet and is chosen arbitrarily to correspond to 2 seconds. $N_{\text{associations}}$ is the number of associations for the tracklet.

6.1.3 Interpolating Re-Identified Tracklets

Lastly, as inspired by StrongSORT, interpolation between detections is introduced when a tracklet is re-identified in the sequence [17]. Linear interpolation is used between the location of where the object is lost and the location where it is re-detected. The linear interpolation is applied on both the position (x, y) and the size (w, h) of the detection.

6.2 Combined A - Joint Object Description

The first strategy (Combined A) is the one, that is most similar to other tracking approaches since it has a separate re-identification module. The first part of the tracker uses the proposed base method as introduced in Chapter 4, to perform a strong association of detections without considering occlusion cases. The second part is a re-identification module with SuperPoint used to find objects that have been lost in the sequence. The tracking structure is shown in Figure 6.1.

6.2.1 Object Description

Due to the use of the two separate tracking modules, the object description of the tracklets is twofold. This means that a tracklet T_k is described both by its latest feature vector, $\mathbf{v}_k$ and by its latest keypoints and descriptors from SuperPoint, $\{\mathbf{p}_k, \mathbf{x}_k\}$.

The MOB module is used to perform the primary association of detections into tracklets. Thus, a tracklet is handled by the MOB module if the time since the last association,

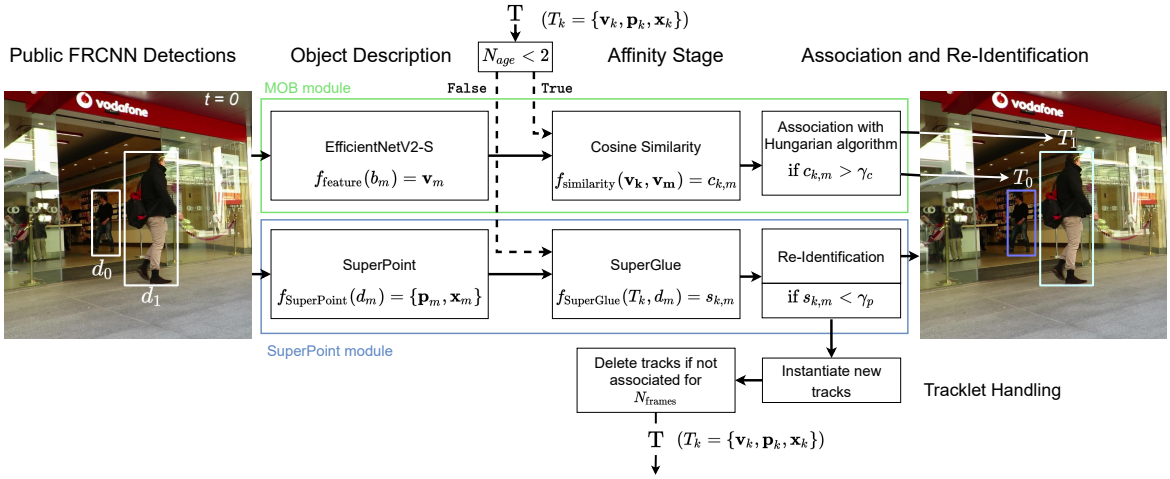


Figure 6.1: Combined A. Notice the green box that represents the utilization of the MOB module and the blue box that represents the usage of SuperPoint as a re-identification module. The example is from the MOT17 dataset [34].

$N_{age} < 2$ (Following Figure 6.1). Conversely, if a tracklet has not been assigned a detection for two consecutive frames, it enters the re-identification module. When an object is re-identified in the scene, N_{age} resets, and associations are again made with the MOB module.

Similar to the SuperPoint tracker as described in Section 5.3, the SuperPoint re-identification module uses a search region for the tracklets. Since the tracklets are now kept in memory for a longer time due to Equation 6.2, the euclidean distance for the search region is expanded to $e_{k,m} < 0.3$.

6.2.2 Affinity Measure and Association

As previously introduced, the MOB module uses cosine similarity between the feature vectors as an affinity measure to perform association with the Hungarian Algorithm.

The SuperPoint module uses the SuperGlue matching score with Hungarian association if re-identification of more than one tracklet is required.

6.2.3 Tracklet Handling

Tracklet handling for the MOB module is performed according to the aforementioned description (Equation 6.1 and 6.2) with the parameters $N_{mov} = 10$ and $\alpha = 0.9$. These are established through trial and error and knowledge from the Affinity experiments from Appendix B.2, however, slight variations do not affect the tracker's performance as much as with the previous similarity thresholds.

As for the SuperPoint module we want to be certain when making a re-identification

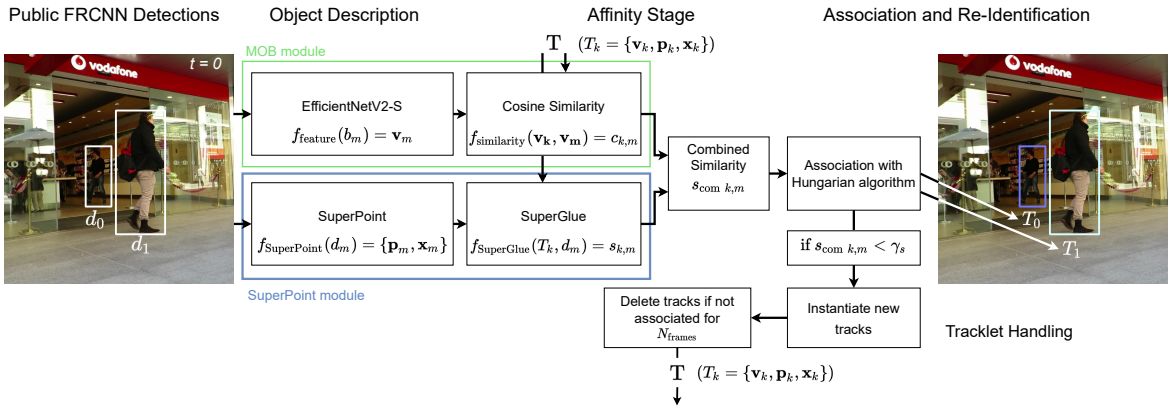


Figure 6.2: Combined B. The MOB module from the green box and the SuperPoint module from the blue box is combined to unified similarity measure used to associate and re-identify the tracklets. The example is from the MOT17 dataset [34].

to avoid identity switches. Thus, we only choose to re-detect an object if we can find at least 10 matching keypoints, $\gamma_p = 10$. This is chosen based on the affinity experiments from Appendix B.3.

6.3 Combined B - Unified Affinity Representation

The second strategy (Combined B) is different from the first one, as it aims to fuse the MOB module and the SuperPoint module into a unified representation of the objects. Hence, this representation is in charge of making both associations of detections in consecutive frames and re-identify lost objects. The tracking framework of Combined B is given in Figure 6.2.

6.3.1 Object Description

Similar to Combined A, an object is described by its latest feature vector from EfficientNet $\mathbf{v}_k$ and by its latest keypoints and descriptors from SuperPoint given as $\{\mathbf{p}_k, \mathbf{x}_k\}$. Opposite to Combined A, both feature representations are used simultaneously to provide a joint description of the objects. A search region of $e_{k,m} < 0.3$ is used as in Combined A.

6.3.2 Affinity Stage - Establishing a Unified Similarity Measure

The cosine similarity between feature vectors, $c_{k,m}$, as defined in Equation 3.3 and provides a measure of the affinity between $[0, 1]$.

For the SuperPoint descriptions, the SuperGlue matching framework is used to calculate a matching score, $s_{k,m}$. Recall, that this score, in theory, can achieve a maximum value of 1024, if the maximum number of keypoints are extracted and they match perfectly. This

is never the case in a real case scenario, since the number of extracted keypoints varies, which also makes the matching score vary.

To accommodate this, it is chosen to normalize $s_{k,m}$ as given in Equation 6.3.

$$s'_{k,m} = \frac{s_{k,m}}{N_{\text{keypoints, } k}} \tag{6.3}$$

where $N_{\text{keypoints, } k}$ is the number of extracted keypoints for tracklet T_k. The normalized score, $s'_{k,m}$, then represents the percent of extracted keypoints that are matched. Hence, it also achieves a value between $[0, 1]$ and thereby weighs the MOB and SuperPoint modules equally.

Summing the two affinity measures from the MOB module and the SuperPoint module we can achieve a combined similarity measure.

$$s_{\text{com } k,m} = s'_{k,m} + c_{k,m} \tag{6.4}$$

Where $s_{\text{com } k,m}$ is the combined similarity measure that achieves values in the range $[0, 2]$.

6.3.3 Association

The matrix of combined similarity measures is used as an input to the Hungarian algorithm that performs the association of detections to tracklets.

6.3.4 Tracklet Handling

For tracklet handling the proposed moving average filter solution from Equation 6.1 is used on s_{com} instead of c.

$$\gamma_s[t] = \frac{1}{N_{mov}} \sum_{i=0}^{N_{mov}-1} s_{\text{com}}[t-i] \cdot \alpha \tag{6.5}$$

The percentage value we require to make an association is set to $\alpha = 0.7$. This value is lower than for Combined A since we also want the combined description to be able to re-identify lost objects. Additionally, we lower the length of the moving average filter to $N_{mov} = 5$, to accommodate for the large variations in matched keypoints as found in the affinity experiments in Appendix B.3.

6.4 Results

The results for the combined trackers with MOB and SuperPoint (A and B) are given in Figure 6.3 and 6.4. Extended results for the training and validation sequences are given in Appendix D.4.

Videos for the two trackers are given at: Combined A Combined B

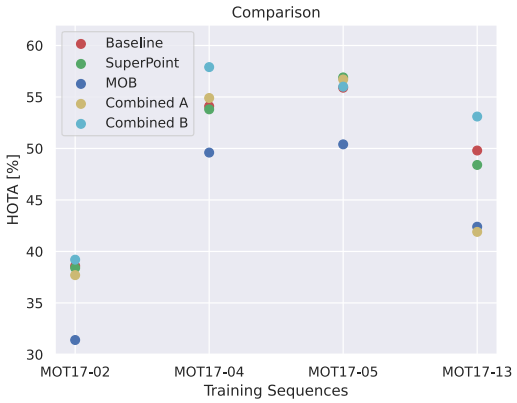


Figure 6.3: Results on the training sequences for the two combined trackers A and B compared to the baseline, MOB and SuperPoint trackers.

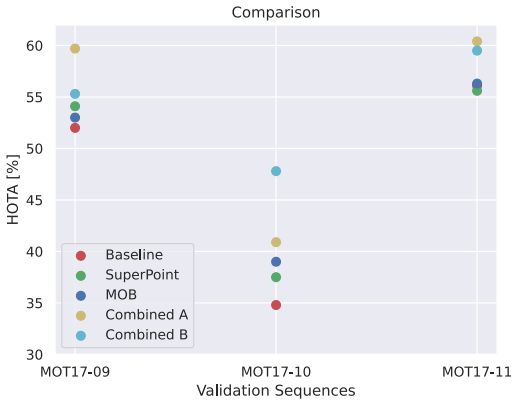


Figure 6.4: Results on the validation sequences for the two combined trackers A and B compared to the baseline, MOB and SuperPoint trackers.

6.5 Findings

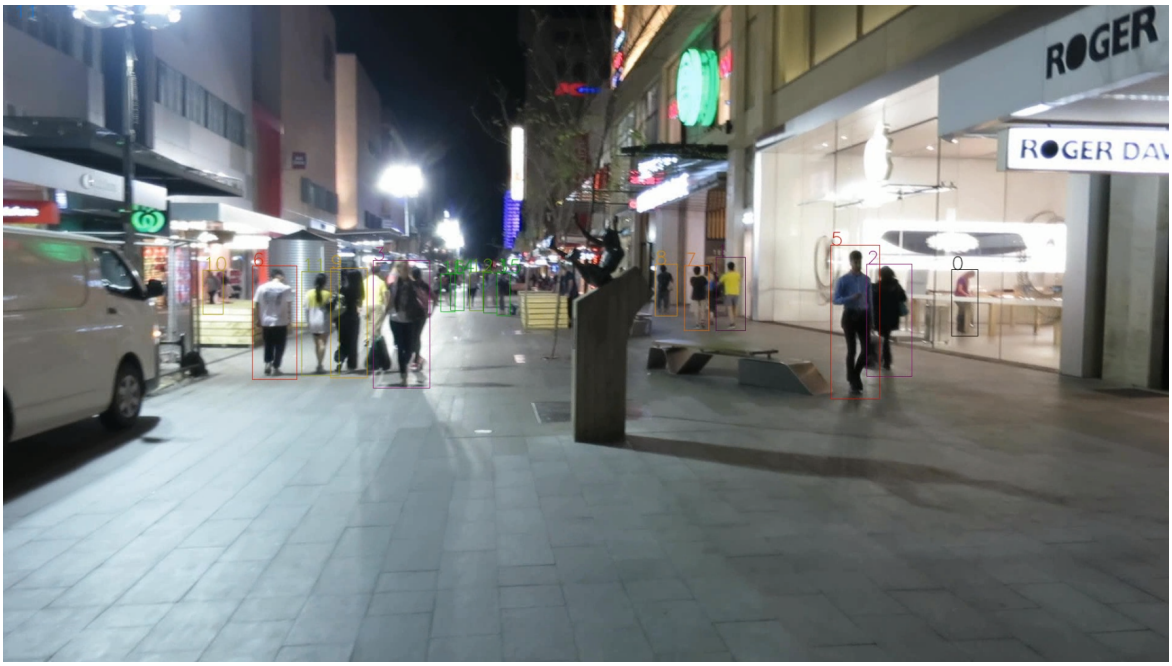
Both strategies improve upon the MOB tracker on the training sequences and perform better or comparatively to the baseline and the SuperPoint tracker. The exception, MOT17-13, is captured from a bus and contains several small detections in the background. Therefore, Combined A is not expected to perform well on this sequence, since the MOB tracker has issues with making strong associations of small detections.

Combined B performs the best on the training sequences with an average of $\mu_{HOTA} = 51.6\%$ compared to $\mu_{HOTA} = 47.8\%$ for Combined A. Combined A still outperforms the MOB tracker $\mu_{HOTA} = 43.4\%$, thus including SuperPoint as a re-identification module improves upon the MOB tracker.

On the validation sequences, both strategies outperform all of the three previous trackers with $\mu_{HOTA} = 53.7\%$ for Combined A and $\mu_{HOTA} = 54.2\%$ for Combined B. They perform similarly on the MOT17-11 sequence. Combined A performs the best on MOT17-09, whereas Combined B performs the best on the MOT17-10 sequence. Examples of these two sequences are given in Figure 6.5. The reason Combined A performs well on MO17-09 is that the detections have a decent size, which allows the MOB module to perform strong associations and the SuperPoint module to re-identify the lost objects. Conversely, the MOT17-10 sequence has smaller background detections which the MOB tracker fails to describe and associate as well as the SuperPoint module.

Overall, the combined similarity representation from Combined B performs the best on both the training and validation sequences, thus it is considered the best of the proposed deep description of objects for MOT.

(a) MOT17-09



(b) MOT17-10

Figure 6.5: Examples of the MOT17-09 and 10 sequences. MOT17-09 is a static camera and MOT17-10 is a moving camera [34].

Chapter 7

Results

The following chapter presents the results that is obtained with the developed trackers. The best tracking representation from the previous chapter, Combined B, is evaluated on the MOT17 test set and an analysis of common failure cases on the test sequences is conducted to understand its strengths and weaknesses. An ablation study is conducted to evaluate how the detection quality impacts the performance of the trackers. Lastly, the results on the Zebrafish dataset and the two Ants datasets are given to further evaluate the performance of the Combined B tracker on other types of data.

7.1 Submission to MOTChallenge

To fully evaluate the performance of the Combined B tracker proposed in Chapter 6 it is evaluated on the MOT17 test sequences. Although the sequences are publicly available, the ground-truth annotations are not, thus one can only obtain scores by submitting results to *MOTChallenge.net*. It is encouraged not to tweak parameters by visually inspecting the test sequences before submitting. The public FRCNN detections with $p_{object} > 0.5$ are used as detections.

7.1.1 Results on the MOT17 Test Sequences

The results for the Combined B tracker on the individual sequences are given in Figure 7.1. The average scores are given in Table 7.1. Extended results for the sequences are given in Appendix D.6. The videos of the test sequences can be found at: Combined B.

7.1.2 Computation Time

The following section gives the computation times of the combined B tracker on the MOT17 test sequences. A total run-time for the current non-optimized tracking framework is stated along with runtimes for the independent modules.

Tracker	Combined B
μ_{HOTA}	32.5 %
μ_{MOTA}	32.4 %
μ_{IDF1}	38.7 %

Table 7.1: The average HOTA, MOTA and IDF1 scores for Combined B tracker on the MOT17 test sequences.

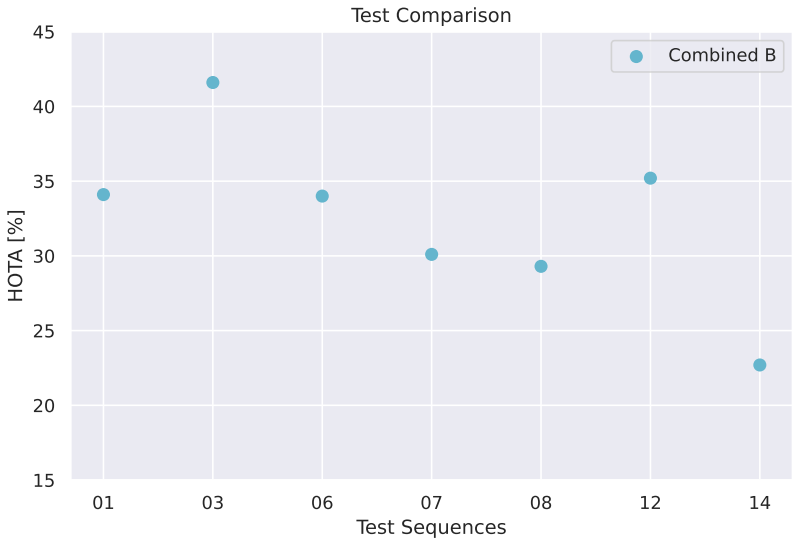


Figure 7.1: The results on the independent MOT17 test sequences for the Combined B tracker.

Hardware

The tracker has been run on a PC with the following hardware. CPU: Intel Core i5-8400 6 Cores, 2.8 GHz. GPU: GeForce GTX 1050Ti. RAM: 8 GB.

The deep learning operations are GPU optimized, where EfficientNetV2-S uses TensorFlow and SuperPoint/SuperGlue uses Pytorch.

Runtime

The total runtime for the Combined B tracker is given in Table 7.2. Detailed runtime information for the deep learning methods is given in Table 7.3.

	Combined B
Total Runtime	0.54 Hz

Table 7.2: Total runtime in Hz for the Combined B tracker evaluated on the test sequences. The total runtime includes loading and assignment operations.

	MOB	SuperPoint	SuperGlue
Runtime pr. Object/Pair	26.0 Hz	79.0 Hz	24.3 Hz

Table 7.3: Runtimes for the deep learning operations (GPU). For MOB (EfficientNetV2-S) and SuperPoint the runtime is pr. object and for SuperGlue it is pr. pair.

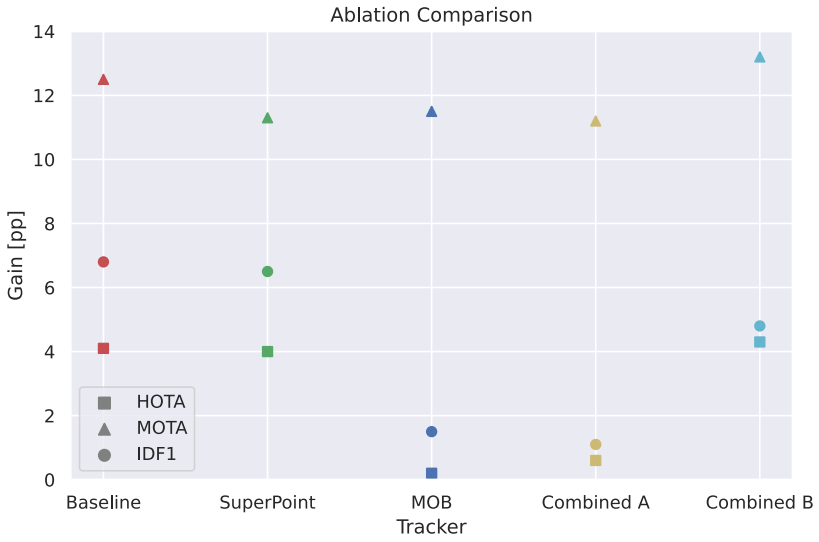


Figure 7.2: Gain in HOTA, MOTA and IDF1 for the five trackers.

7.2 Ablation Study

The purpose of the ablation study is to evaluate how the object detection stage affects the tracker. To simulate, that the tracker has access to perfect detections, the ground-truth annotations are used.

The ground-truth annotations of objects in the MOT sequences are present even when objects are fully occluded. Thus, it is chosen to only encounter annotations if more than 0.5 of their bounding box area is visible.

7.2.1 Ablation Results

The ablation study has been run on both the training and validation sequences. Figure 7.2 shows the average performance gain in percentage points (pp) for the various trackers. Both the training and validation sequences are used. Videos for the ablation study are given at: Ablation Study

7.3 Generalizability to Other MOT Sequences

To establish the generalizability of the Combined B tracker to other types of data, it is evaluated on different sequences. The results are given for the training sequences of the Zebrafish dataset [6] and the Ants Indoor/Outdoor datasets [44]. For the Zebrafish dataset, no public detections are available, thus the ground-truth annotations are used. Similar to the ablation study, an annotation is only encountered if more than 0.5 or its bounding box area is visible. For the Ants datasets, the public detections with $p_{object} > 0.5$ are used.

The averaged results on the datasets are given in Table 7.4. Extended results are given in Appendix D.7. Videos for the sequences are given at: Combined B

Dataset	Zebrafish	Ants Indoor	Ants Outdoor
μ_{HOTA}	75.2 %	98.2 %	74.2 %

Table 7.4: Average HOTA scores for the training sequences of the various datasets.

7.4 Analysis of Failure Cases on the MOT17 Test Sequences

An analysis of the three most common failure cases is conducted, as the combined tracking framework did not perform as expected on the MOT17 test sequences.

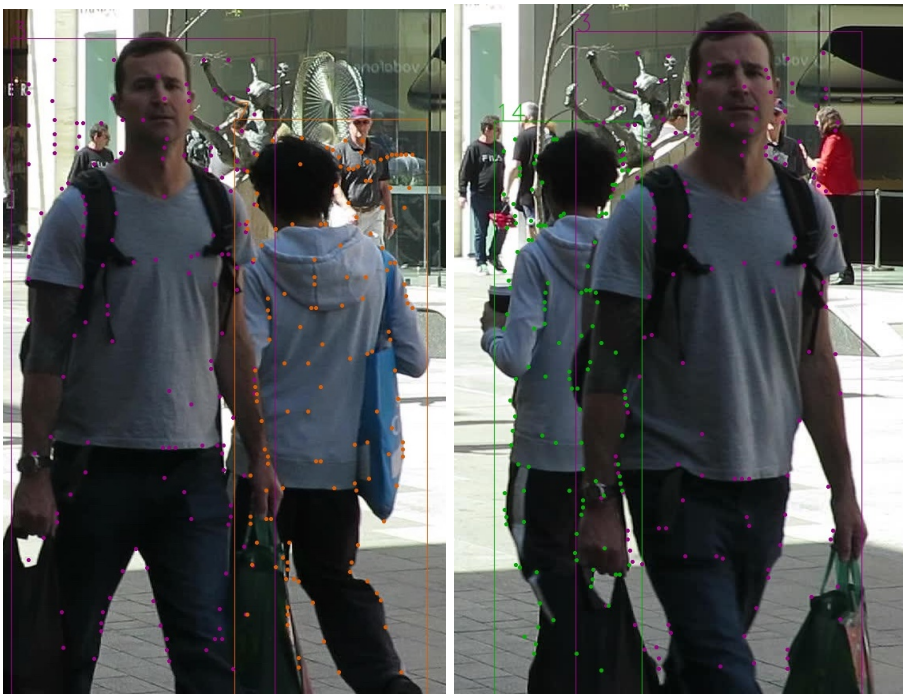
7.4.1 Occlusions - What SuperPoints Should You Take?

When an object enters an occlusion case, the last detection of the object is often partly occluded resulting in a mirrored case scenario. Consider the occlusion example in Figure 7.3. As the person with ID 7 moves into the occlusion case, his bounding box consists of his right side and the occluder's right side. Conversely, when the person with ID 7 leaves the occlusion case the bounding box consists of the person's left side and the occluder's left side. E.g. the bag on his right shoulder is intuitively a good feature to describe him and contains some SuperPoints. However, this bag is fully occluded when the person leaves the occlusion case. This makes the tracker unable to re-identify the object.

Thus, the problem of re-identifying objects in tracking is not only a matter of the deep learning approaches being able to extract discriminate features. It is also a matter of selecting the features, in this case, SuperPoints best allows to perform re-identification.

7.4.2 Detections

As already established in Section 4.3 the detection size has a great impact on the trackability of objects. However, the detection quality of the test sequences also affects the performance if an object is not continuously detected. In Figure 7.4 an example of this



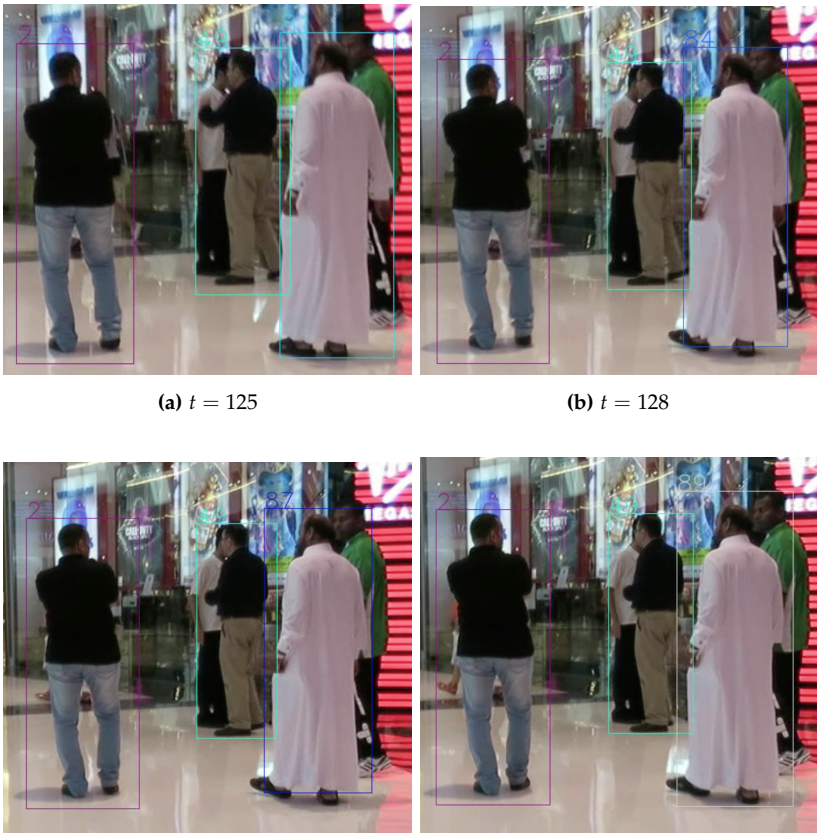
(a) $t = 52$ **(b)** $t = 65$

Figure 7.3: Example of the occlusion problem from the MOT17-08 sequence [34]. The Combined B tracker fails to re-identify object 7.

problem for the MOT17-12 sequence is given. The video example can be seen at MOT17-12 Example at $t = 128$. Here the person in white clothes obtains 4 different ids across 7 frames since the object detector does not continuously detect him. Additionally, in the frames leading up to the case, the object detector has consistently failed to detect the person. In result of this, the tracker is unaware that this object is a "visually clear" object that should have been kept in memory.

This problem can be handled in two ways. Tracklets could either have been kept alive for longer, allowing the tracker to re-identify the person. However, the better solution would be for the object detector to detect the object more consistently. This can either be done by using a different set of detections or by lowering the detection thresholds from $p_{object} > 0.5$ to give the tracker access to less probable detections.

In hindsight, the detector detects the object somewhat consistently, but at times with an object probability as low as $p_{object} < 0.2$.



(a) $t = 125$ **(b)** $t = 128$

(c) $t = 130$ **(d)** $t = 132$

Figure 7.4: Example of the tracklet instantiation problem, if an object is not detected somewhat continuously from the MOT17-12 sequence [34]. Notice, that the person in white clothes obtains 3 different ids across 4 seconds.

7.4.3 Failure Cases of the MOB Approach

Figure 7.5 visualizes an example of an association that moves severely in terms of spatial position in the frame. The association of ID 25 moves 20 % of the image width in spatial position.

The incorporation of the spatial position is handled implicitly by the deep feature representation of the MOB approach. Ultimately, it is the affinity measure between the deep feature representations that decide how similar the objects are both in terms of the spatial and textural information. For the Combined B framework, the object description stage consists partly of the cosine similarity from the MOB approach. This suggests, that the MOB approach does not incorporate sufficient spatial information of objects, thus allowing for associations to move severely in a frame.

(a) $t = 158$

(b) $t = 174$

Figure 7.5: Example the MOT17-01 sequence where the association of object 25 moves 20 % of the image width [34].

Chapter 8

Discussion

The following chapter discusses the results from the previous chapter and identifies the strengths and weaknesses of the combined tracking framework. At first, the results from the MOT17 test sequences are examined. Thereafter, the implications of the detection quality are discussed. The capabilities of the MOB and SuperPoint representations are discussed along with the perspectives of the combined tracking framework.

8.1 Results on the Test Sequences

The Combined B achieves an average score of $\mu_{HOTA} = 32.5\%$ on the MOT17 test sequences. This result is much lower than those obtained on the MOT17 training validation sequences ($\mu_{HOTA} = 51.6\%$ and $\mu_{HOTA} = 54.2\%$, respectively). This suggests that the difficulty of the selected validation sequences does not match the difficulty of the test sequences, as the tracker can generalize to the validation sequences.

Assessing the performance on the individual sequences visually, the tracker performs close to expected, but with low scores, on the 01, 03, 06, 07, 08 and 12 sequences. Thus, the test sequences have an increased difficulty. The performance of the 14'th sequence is considered a failure. This sequence is from a moving camera mounted on a bus, thus it consists of small and fast-moving detections. Additionally, the sequence is one of the most challenging sequences of the MOT17 dataset [61]. Leaving out this sequence the remaining test sequences achieve an average score of $\mu_{HOTA} = 34.1\%$ for Combined B.

The detection issues explained in Section 7.4.2 are especially a problem for the 07 and 12 sequences, as the detector does not consistently detect some clear objects. However, the detection quality is generally a problem for the tracker, as they have a low MOTA score of $\mu = 32.4\%$ and a low recall rate of 43.0 % (See Appendix D.6). Additionally, the tracker suffers from the associations being able to move severely in a frame, as explained in Section 7.4.3.

8.2 Detection Quality

Figure 7.2 shows the gain improvements of the HOTA, MOTA and IDF1 metrics for the five trackers in the ablation study. The MOB and Combined A tracker achieve a lower HOTA gain (0.2 pp and 0.6 pp, respectively) and IDF1 gain (1.5 pp and 1 pp, respectively) compared to the other three trackers. This suggests, that the MOB approach does not benefit from improved detections as the MOB approach also acts as the base for the Combined A tracker.

Specifically, the low IDF1 gain suggests that there is a limit to how well the developed MOB approach can perform associations, although it has access to ground truth annotations. The baseline, SuperPoint and Combined B tracker all performs better associations when having access to better detections. Combined B does not achieve as high an IDF1 gain as the SuperPoint tracker, most likely since its affinity measure consists partly of the MOB description.

Using ground-truth annotations as detections makes a substantial MOTA gain of around 12 pp for all trackers, with Combined B achieving the highest gain of 13.2 These results are expected, as the MOTA metric weights the detections the highest.

8.3 Generalizability

The results for the Combined B tracker on the Zebrafish, Ants Indoor and Outdoor datasets (Table 7.4) show the tracker's ability to generalize to other types of data. The Ants Indoor dataset is tracked almost perfectly, however, the sequences do not have any occlusions. The Zebrafish and Ants Outdoor datasets achieve an average HOTA score of around 75 %, thus showing that the combined description can describe other objects than pedestrians since neither MOB nor the SuperPoint module is finetuned on any of the data.

8.4 MOB and Spatial Incorporation

The MOB approach showed to some extent that it can incorporate the spatial information of the detections, as it can associate corresponding objects across frames. Since it is an implicit incorporation of spatial information it allows for associations to move severely in frames since the affinity measure describes both spatial and textural similarity (Section 7.4.3). Experiments from Appendix C have tried to enhance its ability to extract valuable information from detections through either larger input shapes, training of the feature extractor or enlarging the detection sizes without essential success. Thus, incorporating both the spatial and textural information into a deep feature representation with the MOB approach is not necessarily a good representation as the objects need to have a decent size in the frame. Additionally, the approach is not suited for the re-identification of objects.

8.5 Potentials with SuperPoint

The SuperPoint framework aids the Combined B tracker in describing the detections with robust keypoints extracted through deep learning. SuperPoint is aimed at performing re-identification of lost objects, however, the framework generally has a strong object description including smaller object sizes.

Taking into account the aforementioned issues with the MOB approach not fully utilizing the spatial information , the combined tracker might not take full advantage of SuperPoint's capabilities. Additionally, the current method uses the pretrained versions of SuperPoint with gray-scale detections as input. Thus, potential improvements can be made by training it for color inputs as pedestrians (and other objects) often can be distinguished by their color representations.

8.6 Runtime Perspectives

The runtime for the combined B tracker is slow being around 2 seconds per frame on the MOT17 test sequences. The high complexity comes from the use of SuperGlue. SuperGlue is the slowest of the used deep learning methods. Moreover, SuperGlue needs to run for all potential pairs to calculate a matching score. A search region is used to lower the computation time. As re-identification of objects is required it needs a larger search region to consider a multitude of objects.

Due to the slow runtime, the tracker is not suited for applications that require real-time performance, e.g. tracking of pedestrians in autonomous driving or other traffic applications. Instead, it is more suited for biology or medical research, where correct associations are more essential than the runtime. Although this would require solving some of the failure cases described in Section 7.4, the tracker showed its ability to generalize to other types of data.

Chapter 9

Conclusion

The analysis defined the MOT problem as tracking-by-detection, where the task is to associate detections and utilize state-of-the-art object detectors in tracking. The MOTChallenge is introduced as the common benchmark to evaluate tracking frameworks with the MOT17 dataset chosen to develop a tracker. The three tracking metrics MOTA, IDF1 and HOTA are introduced, with HOTA selected as the main metric as it weighs detections and associations equally. Based on the analysis and the discovered related work, the following problem analysis is established:

How can deep learning be used to create a robust description of detected objects, that can be used to distinguish and associate objects in MOT?

A baseline deep tracking framework is developed consisting of the MobileNetV3-L to obtain a deep representation of objects, the cosine similarity to calculate an affinity measure and the Hungarian algorithm to perform associations. The baseline tracker achieves $\mu_{HOTA} = 49.6\%$ on the MOT17 training split and $\mu_{HOTA} = 47.6\%$ on the validation split.

The MOB approach is proposed to mask out the background of detections and implicitly incorporate the spatial information of detections in a deep representation. The MOB tracker uses the EfficientNetV2-S with a black MOB as an object description, cosine similarity as an affinity measure and the Hungarian algorithm to perform associations. The MOB tracker achieves $\mu_{HOTA} = 43.4\%$ and $\mu_{HOTA} = 49.5\%$ on the training and validation sequences, respectively. The incorporation of spatial position in a deep description of objects is able to generalize slightly better to the validation sequences than the baseline. The ablation study shows that the MOB approach is unable to perform better associations when using ground-truth annotations.

The SuperPoint framework is used to describe detections with keypoints descriptors and the SuperGlue framework matches the detections to produce an affinity measure. Using the Hungarian algorithm to perform associations, the method achieves $\mu_{HOTA} = 49.5\%$ on the training sequences and $\mu_{HOTA} = 49.1\%$ on the validation sequence. Thus,

SuperPoint can obtain a robust deep description of the objects, as it performs better than the deep baseline.

The MOB and SuperPoint approaches are merged into two combined trackers to utilize the deep description of the two approaches. The first tracker (Combined A) uses the MOB approach as the base module for tracking and the SuperPoint framework as a re-identification module. The second tracker (Combined B) uses the MOB and SuperPoint modules to create a unified affinity measure used to perform associations. Combined B has a slight edge over the first approach as it achieves $\mu_{HOTA} = 51.6\%$ and $\mu_{HOTA} = 54.2\%$ on the training and validation sequences respectively, compared to $\mu_{HOTA} = 47.8\%$ and $\mu_{HOTA} = 53.7\%$ for the combined A tracker.

Results submitted to the MOTChallenge show that the Combined B tracker does not perform as well on the MOT17 test sequences with $\mu_{HOTA} = 32.5\%$. Removing the failed MOT17-14 sequence, a score of $\mu_{HOTA} = 34.1\%$ is reported. The best performing test sequence is the MOT17-03 with a HOTA score of 41.6 %. The public FRCNN detections are a problem for the tracker. This is reflected in the low MOTA score ($\mu_{MOTA} = 32.4\%$) and the low recall rate ($\mu_{recall} = 43.0\%$).

The Combined B tracker shows generalizability to other data types as it achieves $\mu_{HOTA} = 75.2\%$, $\mu_{HOTA} = 98.2\%$ and $\mu_{HOTA} = 74.2\%$ on the Zebrafish, Ants Indoor and Outdoor datasets, respectively. Thus, the combination between MOB and SuperPoint can be used as a deep representation to describe objects for multi-object tracking, without the need for training. The MOB approach shows some issues with only incorporating the spatial information implicitly, as it fails to distinguish objects that are not spatially close. Although it has a high computation time, the SuperPoint framework can be used as a robust deep object description and may benefit from more direct representation of position.

9.1 Future Research Areas

Two future research areas are proposed. Better detections with a higher recall rate, e.g. not missing as many objects, may benefit the Combined B tracker and improve its performance on the MOT17 test sequences.

Training of the SuperPoint method to extract and describe keypoints based on color inputs may improve SuperPoint's description of objects. Using the SuperPoint framework in combination with a more direct usage of the spatial information may prove beneficial.

Bibliography

[1] Zhiqun He et al. "Multi-Camera Vehicle Tracking with Powerful Visual Features and Spatial-Temporal Cue". In: *CVPR Workshops*. 2019.

[2] Gaoang Wang et al. "Anomaly Candidate Identification and Starting Time Estimation of Vehicles from Traffic Videos". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019.

[3] Hung-Min Hsu et al. "Multi-Camera Tracking of Vehicles based on Deep Features Re-ID and Trajectory-Based Camera Link Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019.

[4] Renshu Gu, Gaoang Wang, and Jenq-neng Hwang. "Efficient Multi-person Hierarchical 3D Pose Estimation for Autonomous Driving". In: *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. 2019, pp. 163–168. DOI: 10.1109/MIPR.2019.00036.

[5] Vladimír Ulman et al. "An Objective Comparison of Cell Tracking Algorithms". In: *Nature methods* 14 (2017), pp. 1141 –1152.

[6] Malte Pedersen et al. *3D-ZeF: A 3D Zebrafish Tracking Benchmark Dataset*. 2020. arXiv: 2006.08466 [cs.CV].

[7] Samreen Anjum and Danna Gurari. "CTMC: Cell Tracking with Mitosis Detection Dataset Challenge". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2020, pp. 4228–4237. DOI: 10.1109/CVPRW50498.2020.00499.

[8] Laura Leal-Taixé et al. *MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking*. 2015. arXiv: 1504.01942 [cs.CV].

[9] Matteo Fabbri et al. "MOTSynth: How Can Synthetic Data Help Pedestrian Detection and Tracking?" In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 10849–10859.

[10] Laura Leal-Taixé et al. *Tracking the Trackers: An Analysis of the State of the Art in Multiple Object Tracking*. 2017. arXiv: 1704.02781 [cs.CV].

[11] Shaoqing Ren et al. “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

[12] Joseph Redmon et al. “You Only Look Once: Unified, Real-Time Object Detection”. In: 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[13] Gioele Ciaparrone et al. “Deep learning in video multi-object tracking: A survey”. In: *Neurocomputing* 381 (2020), 61–88. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2019.11.023. URL: http://dx.doi.org/10.1016/j.neucom.2019.11.023.

[14] Mykhaylo Andriluka, Stefan Roth, and Bernt Schiele. “Monocular 3D pose estimation and tracking by detection”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 2010, pp. 623–630. DOI: 10.1109/CVPR.2010.5540156.

[15] Alex Bewley et al. “Simple online and realtime tracking”. In: *2016 IEEE International Conference on Image Processing (ICIP)* (2016). DOI: 10.1109/icip.2016.7533003. URL: http://dx.doi.org/10.1109/ICIP.2016.7533003.

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

[17] Yunhao Du et al. *StrongSORT: Make DeepSORT Great Again*. 2022. DOI: 10.48550/ARXIV.2202.13514. URL: https://arxiv.org/abs/2202.13514.

[18] Jinkun Cao et al. *Observation-Centric SORT: Rethinking SORT for Robust Multi-Object Tracking*. 2022. DOI: 10.48550/ARXIV.2203.14360. URL: https://arxiv.org/abs/2203.14360.

[19] Jiangmiao Pang et al. “Quasi-Dense Similarity Learning for Multiple Object Tracking”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 164–173.

[20] Guillem Braso and Laura Leal-Taixe. “Learning a Neural Solver for Multiple Object Tracking”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[21] Chanho Kim et al. “Discriminative Appearance Modeling With Multi-Track Pooling for Real-Time Multi-Object Tracking”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 9553–9562.

[22] Kuan Fang, Yu Xiang, and Silvio Savarese. “Recurrent Autoregressive Networks for Online Multi-Object Tracking”. In: (2017).

[23] Daniel Stadler and Jurgen Beyerer. “Improving Multiple Pedestrian Tracking by Track Management and Occlusion Handling”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 10958–10967.

[24] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. "Tracking Objects as Points". In: 2020, pp. 474–490. ISBN: 978-3-030-58547-1. DOI: 10.1007/978-3-030-58548-8_28.

[25] Kaiwen Duan et al. "CenterNet: Keypoint Triplets for Object Detection". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6568–6577. DOI: 10.1109/ICCV.2019.00667.

[26] Yifu Zhang et al. "FairMOT: On the Fairness of Detection and Re-identification in Multiple Object Tracking". In: *International Journal of Computer Vision* 129.11 (2021), pp. 3069–3087. DOI: 10.1007/s11263-021-01513-4. URL: https://doi.org/10.1007\%2Fs11263-021-01513-4.

[27] Ning Wang et al. "Unsupervised Deep Representation Learning for Real-Time Tracking". In: *International Journal of Computer Vision* 129 (2021), pp. 400–418.

[28] Laura Leal-Taixé and Cristian Canton-Ferrer. "Learning by Tracking: Siamese CNN for Robust Target Association". In: 2016, pp. 418–425. DOI: 10.1109/CVPRW.2016.59.

[29] Paul Voigtlaender et al. "Siam R-CNN: Visual Tracking by Re-Detection". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[30] Luca Bertinetto et al. "Fully-Convolutional Siamese Networks for Object Tracking". In: *ECCV Workshops*. 2016.

[31] Siyuan Cheng et al. "Learning To Filter: Siamese Relation Network for Robust Tracking". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 4421–4431.

[32] Bing Shuai et al. "SiamMOT: Siamese Multi-Object Tracking". In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 12367–12377.

[33] Philipp Bergmann, Tim Meinhardt, and Laura Leal-Taixe. "Tracking Without Bells and Whistles". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[34] Anton Milan et al. *MOT16: A Benchmark for Multi-Object Tracking*. 2016. arXiv: 1603.00831 [cs.CV].

[35] G LoweDavid. "Distinctive Image Features from Scale-Invariant Keypoints". In: *International Journal of Computer Vision* (2004).

[36] Laura Leal-Taixé, Gerard Pons-Moll, and Bodo Rosenhahn. "Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker". In: 2011, pp. 120–127. DOI: 10.1109/ICCVW.2011.6130233.

[37] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[38] M. Everingham et al. "The Pascal Visual Object Classes (VOC) Challenge". In: *International Journal of Computer Vision* 88.2 (2010), pp. 303–338.

[39] James Ferryman and Anna-Louise Ellis. "Performance evaluation of crowd image analysis using the PETS2009 dataset". In: *Pattern Recognition Letters* 44 (2014). Pattern Recognition and Crowd Analysis, pp. 3–15. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2014.01.005. URL: https://www.sciencedirect.com/science/article/pii/S0167865514000191.

[40] A Geiger et al. "Vision meets robotics: The KITTI dataset". In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237. DOI: 10.1177/0278364913491297. eprint: https://doi.org/10.1177/0278364913491297. URL: https://doi.org/10.1177/0278364913491297.

[41] Patrick Dendorfer et al. *MOT20: A benchmark for multi object tracking in crowded scenes*. 2020. arXiv: 2003.09003 [cs.CV].

[42] Ming-Fang Chang et al. "Argoverse: 3D Tracking and Forecasting With Rich Maps". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019.

[43] Holger Caesar et al. "nuScenes: A multimodal dataset for autonomous driving". In: *CVPR*. 2020.

[44] xiaoyan Cao. *ANTS–ant detection and tracking*. 2021. DOI: 10.17632/9WS98G4NPW.3. URL: https://data.mendeley.com/datasets/9ws98g4npw/3.

[45] Vasant Manohar et al. "PETS vs . VACE Evaluation Programs : A Comparative Study". In: 2006.

[46] Jonathon Luiten et al. "HOTA: A Higher Order Metric for Evaluating Multi-object Tracking". In: *International Journal of Computer Vision* 129.2 (2020), 548–578. ISSN: 1573-1405. DOI: 10.1007/s11263-020-01375-2. URL: http://dx.doi.org/10.1007/s11263-020-01375-2.

[47] I Leichter and E Krupka. "Monotonicity and error type differentiability in performance measures for target detection and tracking in video". eng. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 2003–2009. ISBN: 9781467312264.

[48] K. Bernardin and R. Stiefelhagen. "Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics." In: *J Image Video Proc 2008*. 2008. DOI: https://doi.org/10.1155/2008/246309.

[49] Ergys Ristani et al. *Performance Measures and a Data Set for Multi-Target, Multi-Camera Tracking*. 2016. arXiv: 1609.01775 [cs.CV].

[50] Andrew Howard et al. "Searching for MobileNetV3". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019.

[51] TensorFlow. "MobileNetV3Large". In: DOI: https://www.tensorflow.org/api_docs/python/tf/keras/applications/MobileNetV3Large.

[52] Mingxing Tan and Quoc V. Le. "EfficientNetV2: Smaller Models and Faster Training". In: (2021). DOI: 10.48550/ARXIV.2104.00298. URL: https://arxiv.org/abs/2104.00298.

[53] Google. "EfficientNetV2". In: DOI: https://tfhub.dev/google/collections/efficientnet_v2/1.

[54] Brian Clapper. "munkres". In: DOI: https://pypi.org/project/munkres/.

[55] H. W. Kuhn. "The Hungarian method for the assignment problem". In: *NRL*. 1955. DOI: https://doi.org/10.1002/nav.3800020109.

[56] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. "SuperPoint: Self-Supervised Interest Point Detection and Description". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2018.

[57] MagicLeap. "SuperPoint Pretrained Network". In: DOI: https://github.com/magicleap/SuperPointPretrainedNetwork.

[58] Paul-Edouard Sarlin et al. *SuperGlue: Learning Feature Matching with Graph Neural Networks*. 2020. arXiv: 1911.11763 [cs.CV].

[59] MagicLeap. "SuperGlue Pretrained Networ". In: DOI: https://github.com/magicleap/SuperGluePretrainedNetwork.

[60] Pieter P. "Moving Average Filter". In: DOI: https://tttapa.github.io/Pages/Mathematics/Systems-and-Control-Theory/Digital-filters/Simple\%20Moving\%20Average/Simple-Moving-Average.html.

[61] MOTChallenge. "MOTChallenge Results". In: DOI: https://motchallenge.net/results/MOT17/.

[62] Xingping Dong and Jianbing Shen. "Triplet Loss in Siamese Network for Object Tracking". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.

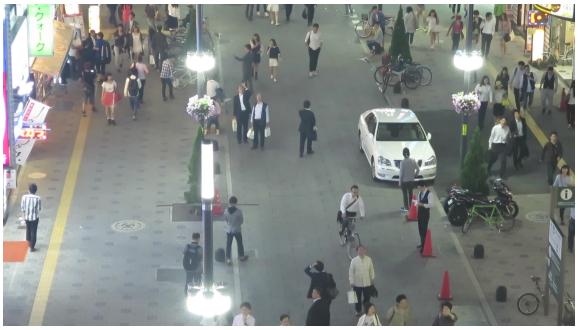
Appendix A

MOT17 Examples

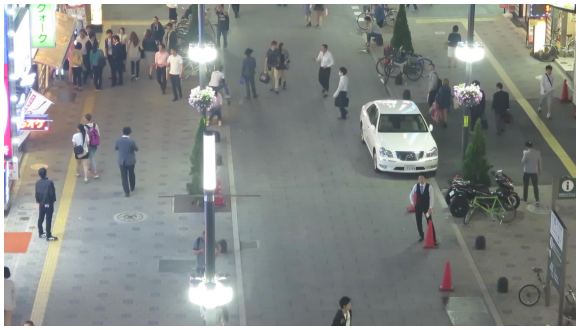
Examples of all the MOT17 sequences are given in Figure A.1, A.2 and A.3.

(a) MOT17-01 **(b)** MOT17-02

Figure A.1: All of the MOT17 sequences.

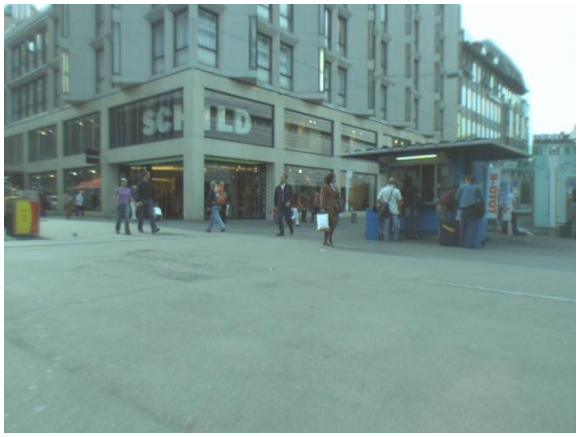


(a) MOT17-03



(b) MOT17-04

(c) MOT17-05



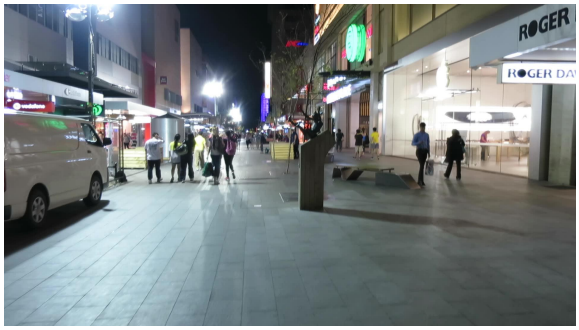
(d) MOT17-06

(e) MOT17-07

(f) MOT17-08

Figure A.2: All of the MOT17 sequences.

(a) MOT17-09



(b) MOT17-10

(c) MOT17-11

(d) MOT17-12

(e) MOT17-13

(f) MOT17-14

Figure A.3: All of the MOT17 sequences.

Appendix B

Affinity Investigations

Initial experiments have been conducted to investigate how the similarity between feature vectors is influenced in different scenarios. The experiments are conducted both for the baseline, MOB approach and SuperPoint approach as it also helps in establishing thresholds for instantiating new tracks.

In the following description *corresponding* refers to the ground-truth bounding box annotations from the MOTChallenge dataset with the same tracking ID.

1. **Aff**$_{\text{corresponding}}$**:** Affinity measure between the extracted feature vectors of the current frame and the corresponding feature vectors in the previous frame.

2. **Aff**$_{\text{most\_similar}}$**:** Maximum affinity measure between the extracted feature vectors of the ground-truth annotations in the current frame and the NOT corresponding ground-truth annotation in the previous frame. E.g. the most similar, but not corresponding, feature vector.

3. **Aff**$_{\text{mean}}$**:** Mean affinity measure between the extracted feature vectors of the ground-truth annotations in the current frame and all the NOT corresponding ground-truth annotations in the previous frame.

B.1 Affinity Results for the Baseline Tracker

The affinity results for the baseline used to set tracking thresholds. See Section B for details of the different scenarios.

B.1.1 Cosine Similarity

MobileNetV3-L (224x224)

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.96782 \pm 0.01825$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.85503 \pm 0.03963$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.70673 \pm 0.11433$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.97459 \pm 0.01324$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.8452 \pm 0.02724$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.7506 \pm 0.06153$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.95272 \pm 0.02817$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.81042 \pm 0.06468$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.71228 \pm 0.11855$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.95845 \pm 0.02345$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.83632 \pm 0.06068$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.70929 \pm 0.11067$

Table B.1: Cosine similarity results for the three different scenarios with the MobileNetV3-L (224x224) on the four different sequences.

EfficientNetV2-S (384x384)

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.97614 \pm 0.01629$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.86132 \pm 0.05666$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.68454 \pm 0.14472$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.97602 \pm 0.01221$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.83703 \pm 0.04162$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.70484 \pm 0.08536$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.96056 \pm 0.02693$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.79536 \pm 0.09536$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.65473 \pm 0.16287$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 0.97556 \pm 0.01793$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 0.87774 \pm 0.07638$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 0.75514 \pm 0.14173$

Table B.2: Cosine similarity results for the three different scenarios with the EfficientNetV2-S (384x384) on the four different sequences.

B.1.2 Euclidean Distance

MobileNetV3-L (224x224)

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 6.49057 \pm 2.27991$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 14.21191 \pm 4.10434$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 20.79316 \pm 6.36422$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 5.99418 \pm 1.64912$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 15.11233 \pm 2.79429$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 20.16888 \pm 3.85788$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 9.11627 \pm 3.1437$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 19.30954 \pm 4.4519$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 23.20209 \pm 5.2538$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 6.37941 \pm 2.34446$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 12.86448 \pm 4.52924$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 17.34368 \pm 6.04815$

Table B.3: Euclidean distance results for the three different scenarios with the MobileNetV3-L (224x224) on the four different sequences.

EfficientNetV2-S (384x384)

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 6.03598 \pm 2.45831$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 14.94742 \pm 4.9504$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 22.68552 \pm 7.14535$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 6.79134 \pm 1.85106$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 17.89403 \pm 3.28423$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 24.52612 \pm 4.55837$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 8.57994 \pm 3.11648$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 20.33122 \pm 6.00148$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 25.7661 \pm 6.88572$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 5.60442 \pm 2.32043$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 12.6333 \pm 5.20926$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 17.93078 \pm 6.99064$

Table B.4: Euclidean distance results for the three different scenarios with the EfficientNetV2-S (384x384) on the four different sequences.

B.2 Affinity Results for the Proposed MOB Approach

The affinity results for the proposed MOB with black and blurred background used to set tracking thresholds. See Section B for details of the different scenarios.

B.2.1 Black

MobileNetV3-L (224x224)

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.9931 \pm 0.01198$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.94107 \pm 0.06465$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.78935 \pm 0.16822$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99616 \pm 0.00887$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.96089 \pm 0.02537$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.8803 \pm 0.06576$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.9579 \pm 0.03032$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.7815 \pm 0.13809$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.66928 \pm 0.17031$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.98632 \pm 0.01719$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.96281 \pm 0.04413$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.91383 \pm 0.07445$

Table B.5: Cosine similarity results for the proposed MOB with black background on the three different scenarios with the MobileNetV3-L (224x224) on the four different sequences.

EfficientNetV2-S

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99126 \pm 0.01302$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.91609 \pm 0.07454$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.76572 \pm 0.14991$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99526 \pm 0.00649$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.93463 \pm 0.0271$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.83789 \pm 0.0679$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.95623 \pm 0.03102$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.77801 \pm 0.12143$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.64691 \pm 0.1641$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.98661 \pm 0.0145$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.94483 \pm 0.04665$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.8754 \pm 0.08081$

Table B.6: Cosine similarity results for the proposed MOB approach with black background on the three different scenarios with the EfficientNetV2-S (384x384) on the four different sequences.

EfficientNetV2-M

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99281 \pm 0.01058$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.93381 \pm 0.05682$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.81151 \pm 0.13022$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99604 \pm 0.00476$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.94874 \pm 0.02354$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.8799 \pm 0.05024$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.96638 \pm 0.02142$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.81447 \pm 0.1109$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.68765 \pm 0.16738$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.98941 \pm 0.01245$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.95621 \pm 0.04053$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.90397 \pm 0.064$

Table B.7: Cosine similarity results for the proposed MOB with black background on the three different scenarios with the EfficientNetV2-M (480x480) on the four different sequences.

B.2.2 Blur

MobileNetV3-L (224x224)

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99452 \pm 0.00591$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.95905 \pm 0.04587$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.8517 \pm 0.12671$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99807 \pm 0.00165$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.98317 \pm 0.01358$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.9461 \pm 0.04728$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.96927 \pm 0.01609$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.8821 \pm 0.05248$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.83671 \pm 0.06413$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.993 \pm 0.00613$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.98316 \pm 0.02286$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.96631 \pm 0.04122$

Table B.8: Cosine similarity results for the proposed MOB with blurred background on the three different scenarios with the MobileNetV3-L (224x224) on the four different sequences.

EfficientNetV2-S

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99231 \pm 0.01069$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.9516 \pm 0.06194$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.88847 \pm 0.10153$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.9988 \pm 0.00149$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.99137 \pm 0.01119$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.9796 \pm 0.02068$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.95417 \pm 0.02716$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.85641 \pm 0.06864$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.79301 \pm 0.09162$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99372 \pm 0.0086$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.989 \pm 0.01878$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.98509 \pm 0.02471$

Table B.9: Cosine similarity results for the proposed MOB with blurred background on the three different scenarios with the EfficientNetV2-S (384x384) on the four different sequences.

EfficientNetV2-M

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99411 \pm 0.00894$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.97084 \pm 0.04756$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.9313 \pm 0.07986$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99899 \pm 0.00074$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.99598 \pm 0.00402$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.99143 \pm 0.00784$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.95781 \pm 0.02428$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.88739 \pm 0.05791$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.84241 \pm 0.07174$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu 0.99521 \pm 0.00613$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu 0.99247 \pm 0.01423$
$\mathbf{Aff}_{\text{mean}}$	$\mu 0.99014 \pm 0.01997$

Table B.10: Cosine similarity results for the proposed MOB with blurred background on the three different scenarios with the EfficientNetV2-S (480x480) on the four different sequences.

B.3 Affinity Results for the SuperPoint Framework

The following section includes the affinity results for SuperPoint used to set tracking thresholds. See Section B for details of the different scenarios.

Scenario	MOT17-02
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 138.9 \pm 99.6$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 3.81 \pm 9.55$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 20.98 \pm 24.92$

Scenario	MOT17-04
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 98.73 \pm 64.53$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 4.41 \pm 5.52$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 21.07 \pm 14.04$

Scenario	MOT17-05
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 144.29 \pm 107.41$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 2.87 \pm 5.60$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 6.39 \pm 8.76$

Scenario	MOT17-13
$\mathbf{Aff}_{\text{corresponding}}$	$\mu = 19.08 \pm 18.02$
$\mathbf{Aff}_{\text{mean}}$	$\mu = 2.85 \pm 2.91$
$\mathbf{Aff}_{\text{most\_similar}}$	$\mu = 9.36 \pm 4.18$

Table B.11: Estimate of number of matched keypoints as a sum matching scores for all matches for the SuperPoint and SuperGlure

Appendix C

MOB - Additional Experiments

This chapter explains the experiments for the proposed MOB tracker, which have not worked out. The first two sections investigate the steps that have been attempted to mitigate the problems with small detections. Section C.1 describes the experiments with a larger input shape along with the results. Section C.2 describes the experiments that have been conducted with enlarging small detections. Lastly, Section C.3 describes the training attempts that have been done with siamese networks, attempted to enhance the object description.

C.1 Larger Input Shape

An immediate consequence of the MOB approach, is the rather limited information that can be extracted from the smaller detections in the background shapes. Some of these examples for the MOT17-02 sequence are visualized in Figure C.1. To increase the feature extractor's possibility to extract valuable information for small detections, the input shape of the network is increased. Attempts have been done with the EfficientNetV2-M to double the input shape to (960x960). The best-achieved results are given in Table C.1. Figure C.2 compares the results to the best performing results for the (480x480).

C.1.1 Results

Scaling	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	31.6 %	49.4 %	46.3 %	36.4 %
MOTA	26.1 %	51.6 %	39.5 %	40.0 %
IDF1	30.9 %	50.5 %	43.3 %	33.1 %

Table C.1: Results for 960x960. Cosine similarity and Hungarian Association.

(a) Bounding box shape (40x8), 320 pixels.

(b) Bounding box shape (39x7), 273 pixels.

Figure C.1: Two examples of small background detections with masked out background for MOT17-02.

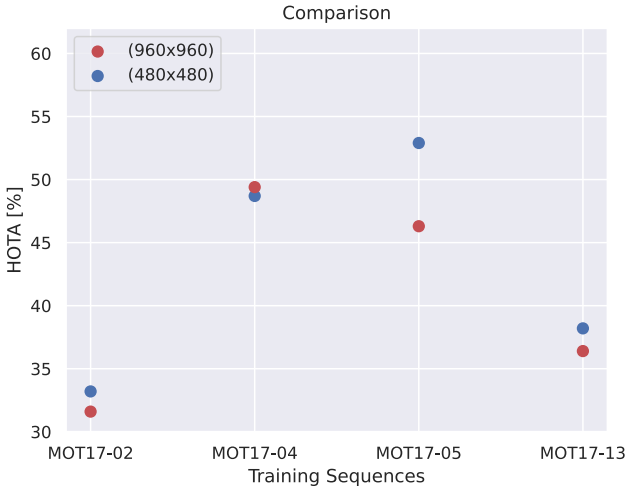


Figure C.2: The best performing results for (960x960) compared to (480x480)

(a) Normal size detection.

(b) Detection enlarged to 0.5 of the image height.

Figure C.3: Example of the image alteration to enlarge the detection size.

As seen in the comparison, the larger input shape does not increase the tracking performance. Most noticeably, the performance is decreased for the MOT17-05 sequence. This sequence contains large close-up detections for a moving camera attached to a pedestrian. Most likely, the bad performance arises since the trained weights in EfficientNetV2-M are unable to extract valuable information.

C.2 Enlarging the Detection Size

As another attempt to alleviate the problem, it has been decided to scale-up the size of the smaller detections in the background. If a detection height is less than half of the image height, the detection is scaled-up to have a height of 0.5. This decision is a trade-off, which gives the feature extractor more textural information to work with, however it removes its ability to infer a size relation of smaller objects in the background. The width to height ratio is maintained, hence it should not impact that spatial dimension. An example of this image manipulation is given in Figure C.3.

C.2.1 Results

The results for the larger detection size are given in Table C.2 and Figure C.4 compares the results to the normal detection size.

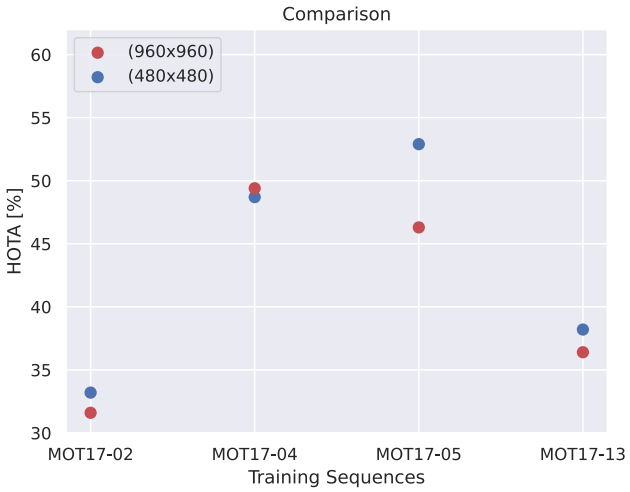


Figure C.4: Normal detections size compared to larger detection size. EfficientNetV2-M with larger detection size up to 0.5. Cosine similarity and Hungarian association.

Scaling	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	32.3 %	47.9 %	51.8 %	43.1 %
MOTA	26.0 %	51.4 %	45.6 %	42.5 %
IDF1	31.3 %	47.5 %	50.0 %	40.2 %

Table C.2: Results for EfficientNetV2-M with larger detection size up to 0.5. Cosine similarity and Hungarian association.

As shown in the comparison, the scaled-up detection size improves the results for the MOT17-13 sequence. This is good, since it is the sequence, with the largest amount of small input shapes. However, the enlarged detection size performs worse on the rest of the sequences. This is probably since the large input shape for small detections becomes the same size as the larger detections. Thus, the MOB approach is no longer able to utilize the size of the detection to separate small detections from large detections.

C.3 Training a Siamese Networks

It has been attempted to train a feature extractor with a siamese network architecture to let corresponding detections be more similar and make non-corresponding detections appear less similar. To accomplish this, a siamese network has been constructed with the structure as given in Figure C.5. The EffientNetV2-M module the pretrained ImageNet version from TensorFlow Hub [53] is used as a backbone. The weights of the backbone are non-trainable Three fully connected (FC) layers are added on top (1024x1024x512) as the trainable weights. Note, that these weights are identical for all three image samples in a

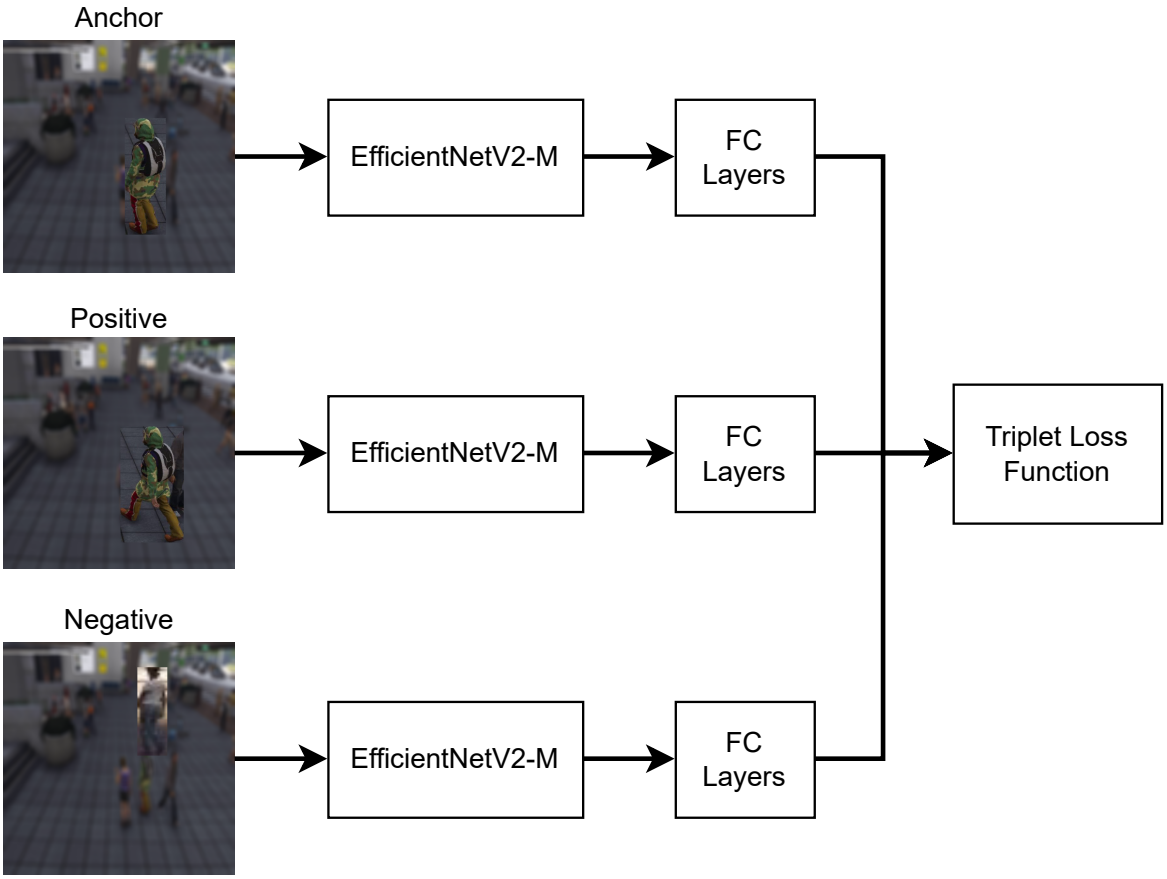


Figure C.5: Example of the network structure used for training of the feature extractor. Three samples, anchor, positive and negative is fed through an EfficientNetV2-M backbone and FC-layers with shared weights. The siamese network uses the triplet loss function for training.

siamese network.

The training procedure uses the triplet loss function as it has previously achieved success for the training of siamese networks in object tracking [62]. This is done since it both enhances the similarity between corresponding detections and increases the dissimilarity between non-corresponding detections. The triplet loss requires three images for training. An anchor, a positive sample which is the corresponding object and a negative sample which is a non-corresponding object.

C.3.1 Data

As data for the network, the MOTSynth dataset is used. Samples are collected by iterating through the dataset and extracting samples of corresponding objects and choosing another random object as a negative sample. Around 15,000 samples for both anchors, positive and

negative samples are used. An example of anchor, positive and negative samples are given in C.5. The example images are for the scaled-up detections. Attempts have been made both with a blurred and black background, with the normal detection size and with the smaller detections scaled-up.

C.3.2 Findings and Problems

As a general trend, the training with the siamese networks made the feature representation from the trained FC-layers of the corresponding objects appear more similar (cosine similarity). However, it also made the feature representations of the non-corresponding objects appear more similar. Thus, the triplet loss function does not take fully into account the loss of the negative samples, to make them appear more dissimilar. Initial thoughts are, that the backbone extracts insufficient information from the small detections (See Figure C.3a) to allow for training of the FC-layers.

Scaling up the detection sizes as described in Section C.2, does not help in training the network sufficiently.

Appendix D

Extended Results

This chapter includes the extended results(MOTA, IDF1 and HOTA scores) for the various proposed frameworks. The results for the baseline tracker are given in Section D.1. The results for the proposed idea are given in Section D.2. The results for the SuperPoint tracker are given in Section D.3. The results for the combined idea and SuperPoint tracker are given in Section D.4.

D.1 Baseline Results

D.1.1 Cosine Similarity

MobileNetV3-L (224x224) hungarian association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	38.6 %	54.1 %	55.9 %	49.8 %
MOTA	25.7 %	52.0 %	38.8 %	29.0 %
IDF1	36.0 %	55.3 %	53.3 %	45.5 %

Table D.1: Baseline results for the MobileNetV3-L (224x224) using the cosine similarity and hungarian association. $N_{\text{frames}} = 10$ and $\gamma_c = 0.85$

MobileNetV3-L (224x224) greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	38.4 %	53.4 %	54.0 %	50.0 %
MOTA	25.5 %	50.1 %	34.5 %	27.8 %
IDF1	36.3 %	54.9 %	50.8 %	45.5 %

Table D.2: Baseline results for the MobileNetV3-L (224x224) using the cosine similarity and greedy association. $N_{\text{frames}} = 10$ and $\gamma_c = 0.85$

Efficientnetv2-S (384x384) hungarian association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	35.5 %	54.0 %	54.1 %	45.1 %
MOTA	25.9 %	51.9 %	38.3 %	35.6 %
IDF1	34.0 %	55.6 %	51.2 %	40.3 %

Table D.3: Baseline results for the Efficientnetv2-S (384x384) using the cosine similarity and hungarian association. $N_{\text{frames}} = 10$ and $\gamma_c = 0.85$

Efficientnetv2-S (384x384) greedy association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	35.7 %	52.7 %	51.1 %	48.8 %
MOTA	35.5 %	49.4 %	33.6 %	34.6 %
IDF1	34.3 %	54.7 %	47.8 %	45.1 %

Table D.4: Baseline results for the Efficientnetv2-S (384x384) using the cosine similarity and greedy association. $N_{\text{frames}} = 10$ and $\gamma_c = 0.85$.

D.1.2 Euclidean Distance

MobileNetV3-L (224x224) hungarian association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	33.8 %	52.3 %	49.2 %	44.8 %
MOTA	25.1 %	49.9 %	29.6 %	39.0 %
IDF1	32.3 %	53.2 %	45.6 %	41.4 %

Table D.5: Baseline results for the MobileNetV3-L (224x224) using the euclidean distance and hungarian association. $N_{\text{frames}} = 10$ and $\gamma_d = 16$.

MobileNetV3-L (224x224) greedy association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	34.0 %	52.3 %	50.0 %	45.0 %
MOTA	25.1 %	49.9 %	30.0 %	38.2 %
IDF1	31.9 %	53.2 %	46.0 %	41.6 %

Table D.6: Baseline results for the MobileNetV3-L (224x224) using the euclidean distance and greedy association. $N_{\text{frames}} = 10$ and $\gamma_d = 16$.

EfficientNetV2-Small (384x384) Hungarian Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	34.6 %	53.2 %	49.0 %	45.6 %
MOTA	25.1 %	50.2 %	29.2 %	37.5 %
IDF1	33.5 %	54.6 %	45.0 %	41.7 %

Table D.7: Baseline results for the EfficientNetv2-S (384x384) using the euclidean distance and hungarian association. $N_{\text{frames}} = 10$ and $\gamma_d = 14$.

EfficientNetV2-Small (384x384) Greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	33.2 %	52.7 %	48.8 %	43.4 %
MOTA	24.5 %	50.0 %	24.2 %	34.4 %
IDF1	31.9 %	54.0 %	41.6 %	40.6 %

Table D.8: Baseline results for the EfficientNetv2-S (384x384) using the euclidean distance and greedy association. $N_{\text{frames}} = 10$ and $\gamma_d = 15$.

D.1.3 Validation Sequences

Input Shape	MOT17-09	MOT17-10	MOT17-11
HOTA	52.0 %	34.8 %	56.1 %
MOTA	56.5 %	16.0 %	53.2 %
IDF1	51.0 %	26.0 %	52.6 %

Table D.9: Results on the validation sequences for the MobileNetV3-L (224x224) using the cosine similarity and hungarian association. $N_{\text{frames}} = 10$ and $\gamma_c = 0.85$

D.2 Results for the MOB Approach

D.2.1 Black Backgrounds

MobileNetV3-L (224x224) Hungarian Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	28.3 %	34.8 %	49.4 %	30.5 %
MOTA	25.2 %	50.5 %	43.2 %	34.0 %
IDF1	26.5 %	32.2 %	47.7 %	25.6 %

Table D.10: Results for proposed idea with black background, MobileNetV3-L (224x224) and hungarian association. $N_{\text{frames}} = 5$, γ_c =0.85

MobileNetV3-L (224x224) Greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	30.3 %	33.7 %	49.0 %	26.9 %
MOTA	25.1 %	49.0 %	37.9 %	32.0 %
IDF1	28.1 %	32.0 %	47.7 %	21.2 %

Table D.11: Results for proposed idea with black background, MobileNetV3-L (224x224) and Greedy association. $N_{\text{frames}} = 5$, γ_c =0.85

EfficientNetV2-S (384x384) Hungarian Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	31.4 %	49.6 %	50.4 %	42.4 %
MOTA	25.7 %	51.3 %	44.4 %	40.7 %
IDF1	29.6 %	50.6 %	48.4 %	38.0 %

Table D.12: Results for proposed idea with black background, EfficientNetV2-S (384x384) and hungarian association. $N_{\text{frames}} = 5$, γ_c =0.85

EfficientNetV2-S (384x384) Greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	31.9 %	45.6 %	48.7 %	36.5 %
MOTA	25.4 %	45.8 %	39.7 %	37.3 %
IDF1	31.6 %	45.0 %	46.8 %	33.9 %

Table D.13: Results for proposed idea with black background, EfficientNetV2-S (384x384) and Greedy association. $N_{\text{frames}} = 5$, γ_c =0.85

EfficientNetV2-M (480x480) Hungarian Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	33.2 %	48.7 %	53.9 %	38.2 %
MOTA	26.2 %	51.4 %	47.7 %	41.3 %
IDF1	33.2 %	48.4 %	51.1 %	34.0 %

Table D.14: Results for proposed idea with black background, EfficientNetV2-M (480x480) and hungarian association. $N_{\text{frames}} = 5$, γ_c =0.85

EfficientNetV2-M (480x480) Greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	32.1 %	41.0 %	47.0 %	29.2 %
MOTA	25.6 %	41.0 %	38.3 %	33.5 %
IDF1	32.4 %	35.3 %	44.5 %	22.3 %

Table D.15: Results for proposed idea with black background, EfficientNetV2-M (480x480) and greedy association. $N_{\text{frames}} = 5$, γ_c =0.85

D.2.2 Results for the Blurred Background

MobileNetV3-L (224x224) Hungarian Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	31.6 %	46.3 %	53.0 %	35.7 %
MOTA	26.0 %	51.3 %	45.6 %	41.2 %
IDF1	29.0 %	45.9 %	52.2 %	30.7 %

Table D.16: Results for proposed idea with blurred background, MobileNetV3-L (224x224) and hungarian association. $N_{\text{frames}} = 5$, γ_c =0.85

MobileNetV3-L (224x224) Greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	32.7 %	38.5 %	51.2 %	30.7 %
MOTA	25.4 %	40.8 %	39.4 %	35.1 %
IDF1	31.8 %	34.2 %	50.2 %	23.9 %

Table D.17: Results for proposed idea with blurred background, MobileNetV3-L (224x224) and greedy association. $N_{\text{frames}} = 5$, γ_c =0.85

EfficientNetV2-S (384x384) Hungarian Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	31.8 %	49.4 %	49.0 %	34.8 %
MOTA	25.9 %	51.4 %	43.4 %	41.5 %
IDF1	29.6 %	50.7 %	46.1 %	31.0 %

Table D.18: Results for proposed idea with blurred background, EfficientNetV2-S (384x384) and hungarian association. $N_{\text{frames}} = 5$, γ_c =0.88

EfficientNetV2-S (384x384) Greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	27.8 %	39.4 %	48.0 %	16.9 %
MOTA	24.6 %	40.5 %	38.3 %	28.3 %
IDF1	25.8 %	34.1 %	44.9 %	11.1 %

Table D.19: Results for proposed idea with blurred background, EfficientNetV2-S (384x384) and greedy association. $N_{\text{frames}} = 5$, γ_c =0.88

EfficientNetV2-M (480x480) Hungarian Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	29.8 %	48.8 %	48.2 %	37.7 %
MOTA	25.9 %	51.3 %	44.4 %	41.5 %
IDF1	29.5 %	50.8 %	46.0 %	33.7 %

Table D.20: Results for proposed idea with blurred background, EfficientNetV2-M (480x480) and hungarian association. $N_{\text{frames}} = 5$, γ_c =0.88

EfficientNetV2-M (480x480) Greedy Association

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	25.1 %	35.4 %	41.7 %	16.4 %
MOTA	23.5 %	40.1 %	36.5 %	26.7 %
IDF1	22.2 %	29.8 %	39.4 %	11.2 %

Table D.21: Results for proposed idea with blurred background, EfficientNetV2-M (480x480) and hungarian association. $N_{\text{frames}} = 5$, γ_c =0.88

D.2.3 Validation Sequences

Input Shape	MOT17-09	MOT17-10	MOT17-11
HOTA	53.0 %	39.0 %	56.4 %
MOTA	56.6 %	41.8 %	55.7 %
IDF1	54.4 %	34.5 %	53.3 %

Table D.22: Validation Sequences for the Black Background, EfficientNetV2-S (384x384) and Hungarian Association. $N_{\text{frames}} = 5$, γ_c =0.88

Input Shape	MOT17-09	MOT17-10	MOT17-11
HOTA	51.1 %	36.8 %	55.4 %
MOTA	56.8 %	42.2 %	56.2 %
IDF1	51.8 %	32.8 %	52.3 %

Table D.23: Validation Sequences for the Black Background, EfficientNetV2-M (480x480) and Hungarian Association. $N_{\text{frames}} = 5$, γ_c =0.88

D.3 Results for SuperPoint Tracking

D.3.1 Training Sequences

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	38.4 %	53.8 %	56.9 %	48.4 %
MOTA	26.7 %	51.6 %	46.9 %	38.4 %
IDF1	27.6 %	54.9 %	55.8 %	44.8 %

Table D.24: Results for the SuperPoint tracker on the Training Sequences.

D.3.2 Validation Sequences

Input Shape	MOT17-09	MOT17-10	MOT17-11
HOTA	54.1 %	37.5 %	55.6 %
MOTA	57.0 %	29.7 %	55.1 %
IDF1	56.1 %	30.7 %	52.5 %

Table D.25: Results for the SuperPoint tracker on the Validation Sequences

D.4 Combined MOB and SuperPoint Results

This section provides the extended results for the combined trackers with the Idea and SuperPoint, given as Combined A and B.

Combined A

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	37.7 %	54.9 %	56.7 %	41.9 %
MOTA	26.5 %	52.2 %	47.6 %	39.1 %
IDF1	35.1 %	57.2 %	55.2 %	37.3 %

Table D.26: Training Sequences

Input Shape	MOT17-09	MOT17-10	MOT17-11
HOTA	59.7 %	40.9 %	60.4 %
MOTA	59.2 %	41.2 %	58.1 %
IDF1	61.6 %	35.6 %	56.9 %

Table D.27: Validation Sequences

Combined B

Input Shape	MOT17-02	MOT17-04	MOT17-05	MOT17-13
HOTA	39.2 %	57.9 %	56.0 %	53.1 %
MOTA	26.0 %	51.7 %	43.5 %	39.5 %
IDF1	37.4 %	58.3 %	53.1 %	47.1 %

Table D.28: Training Sequences

Input Shape	MOT17-09	MOT17-10	MOT17-11
HOTA	55.3 %	47.8 %	59.5 %
MOTA	58.2 %	38.9 %	56.8 %
IDF1	57.7 %	42.3 %	57.7 %

Table D.29: Validation Sequences

D.5 Ablation Results

The section contains the extended results for the ablation study.

Sequence	02	04	05	09	10	11	13	Avg/Total
HOTA[%]	40.3	62.4	54.4	54.0	38.3	58.2	62.9	52.9
MOTA[%]	42.1	59.2	40.3	61.9	34.1	60.7	59.8	51.2
IDF1 [%]	42.9	62.5	51.9	63.0	29.8	54.5	63.0	52.5

Table D.30: Ablation results for the baseline tracker.

Sequence	02	04	05	09	10	11	13	Avg/Total
HOTA[%]	34.2	55.4	48.9	53.7	39.2	56.9	35.1	46.2
MOTA[%]	39.7	58.4	48.8	60.4	66.0	64.0	59.7	56.7
IDF1 [%]	33.4	54.8	46.3	54.9	36.5	54.1	29.8	44.3

Table D.31: Ablation results for the MOB tracker.

Sequence	02	04	05	09	10	11	13	Avg/Total
HOTA[%]	40.8	61.4	55.2	57.6	40.4	59.0	58.0	53.2
MOTA[%]	40.8	58.7	48.7	61.0	51.6	62.5	60.7	54.9
IDF1 [%]	44.2	61.9	54.6	59.2	35.3	56.7	56.1	52.6

Table D.32: Ablation results for the SuperPoint tracker.

Sequence	02	04	05	09	10	11	13	Avg/Total
HOTA[%]	37.6	56.2	55.4	58.6	45.7	62.8	39.8	50.9
MOTA[%]	40.5	59.0	50.0	62.0	65.7	66.3	59.2	57.5
IDF1 [%]	37.3	55.1	56.0	58.3	41.3	63.3	34.8	49.4

Table D.33: Ablation results for the Combined A tracker.

Sequence	02	04	05	09	10	11	13	Avg/Total
HOTA[%]	47.4	61.3	51.2	58.4	57.0	65.4	58.6	57.0
MOTA[%]	43.3	59.3	46.2	62.4	66.2	65.6	63.8	58.1
IDF1 [%]	48.0	60.1	48.2	60.7	53.0	64.0	53.0	55.3

Table D.34: Ablation results for the Combined B tracker.

D.6 Results on the Test Sequences

This section contains the extended results for the combined tracking framework B on the MOT17 test sequences. The results are placed in Table D.35.

Sequence	01	03	06	07	08	12	14	Avg/Total
HOTA[%]	34.1	41.6	34.0	30.8	29.3	35.2	22.7	32.5
MOTA[%]	27.4	58.3	37.8	30.8	24.1	33.7	14.7	32.4
IDF1 [%]	40.3	51.8	40.6	35.4	31.3	43.3	28.0	38.7
Prcn [%]	68.1	95.5	75.2	79.1	88.6	83.8	66	79.5
Rcll[%]	52.9	61.6	62.1	43.6	28.3	43.4	35.2	43.0

Table D.35: Extended results for the Combined B tracker on the MOT17-test (FRCNN) sequences. Precision(Prcn), Recall(Rcll), TP, FP and FN are metrics mostly for the detections.

D.7 Results for Other MOT Sequences

This section presents the extended results for the Combined B tracker on the Zebrafish, Ants Indoor and Outdoor sequences.

D.7.1 Zebrafish

Input Shape	01	02	03	04
HOTA	58.1 %	80.0 %	100 %	62.5 %
MOTA	99.5 %	99.3 %	100 %	95.8 %
IDF1	54.3 %	79.9 %	100 %	57.3 %

Table D.36: Results for the Combined B tracker on the Zebrafish sequences.

D.7.2 Ants Indoor

Input Shape	01	02	03	04	05
HOTA	97.1 %	94.4 %	99.7 %	100 %	100 %
MOTA	99.9 %	99.8 %	99.5 %	100 %	100 %
IDF1	97.7 %	94.2 %	99.4 %	100 %	100 %

Table D.37: Results for the Combined B tracker on the Ants Indoor sequences.

D.7.3 Ants Outdoor

Input Shape	06	07	08	09	10
HOTA	78.9 %	73.7 %	74.8 %	66.6 %	77.2 %
MOTA	93.7 %	91.9 %	83.1 %	89.2 %	94.9 %
IDF1	75.2 %	65.6 %	73.7%	62.0 %	71.7 %

Table D.38: Results for the Combined B tracker on the Ants Outdoor sequences.