GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

ISSN 1612-6793

Master's Thesis
submitted in partial fulfilment of the
requirements for the course "Applied Computer Science"

Post-Processing of Multi-Target Trajectories for Traffic Security Analysis

Thorben Janz

Institute of Computer Science

Bachelor's and Master's Theses
of the Center for Computational Sciences
at the Georg-August-Universität Göttingen

22. December 2017

Georg-August-Universität Göttingen
Institute of Computer Science

Goldschmidtstraße 7
37077 Göttingen
Germany

☎ +49 (551) 39-172000
FAX +49 (551) 39-14403
✉ office@informatik.uni-goettingen.de
www.informatik.uni-goettingen.de

First Supervisor: Jun.-Prof. Dr.-Ing. Marcus Baum
Second Supervisor: Dr.-Ing. Andreas Leich

I hereby declare that I have written this thesis independently without any help from others and without the use of documents or aids other than those stated. I have mentioned all used sources and cited them correctly according to established academic citation rules.

Göttingen, 22. December 2017

Abstract

Existing multi-target tracking systems might be non-modifiable, but still provide qualitatively insufficient trajectory outputs. In this case, improvements are achievable by means of post-processing. This thesis intends to qualitatively improve the multi-target tracking performance at the AiM research crossroads, which is operated by the German Aerospace Center. The focus is set on metrics relevant for traffic security analysis, that is ID switches and track fragmentation specifically. A two-phase track stitching method is proposed, which introduces a novel breaking phase for tracklet creation. Beside track stitching mechanisms, common minimum-cost network flow techniques are utilized. In addition to the evaluation regarding trajectories at the research crossroads, standardized MOTChallenge benchmarks are applied, in order to review the method's general performance. Improvements between 14% and 33% concerning ID switches 40% in terms of track fragmentations are achieved. Post-processing, as applied in the presented setting, is applicable in specific problem scenarios. However, the approach of pipelining a set of specialized, cascaded tracking algorithms is encouraged for in-depth research.

Contents

List of Figures

List of Tables

Chapter 1

Problem Introduction

Tracking is one intensively investigated field of study in the domain of sensor data fusion. Its most intuitive application is tracking the position of some object over time, for example connecting GPS measurements, in order to observe its trajectory. Classic tracking systems have their roots in military appliances, like missile navigation [3], which have been adapted to civil purposes through the years. Nowadays a wide quantity of tracking systems have been developed and great research effort is expended to steadily improve performances. The currently most hyped implementation is autonomous driving [4] and advanced driver assistance systems. Still, the same methods can be applied on problems in various scopes. An example of less prominent applications is the usage of Kalman filter techniques in financial analysis [5].

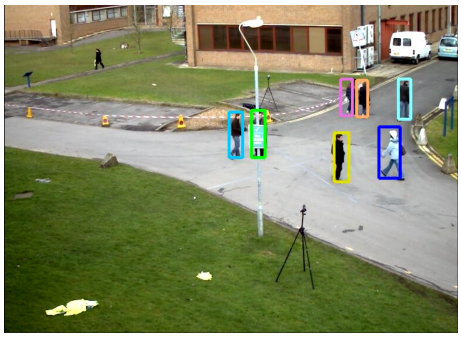


Figure 1.1: Tracking of multiple pedestrians, each color representing separate identity [2].

One of the main focuses of research at the Institute of Transportation Systems at the German Aerospace Center (DLR) is traffic security analysis. In order to evaluate parameters of traffic security, the DLR surveils relevant road sections. For this purpose, the Application Platform for Intelligent Mobility (AiM) research crossroads (figure 1.2) is established, which serves as a platform for multiple traffic studies with different objectives. Traffic parameters are recorded by utilizing video and radar based multi-target tracking systems, which feature common problems among tracking methods. For instance, fragmented trajectories and identity switches impair the quality of generated tracks. The method developed in this thesis intends to improve the reliability of these trajectories, by compensating the most relevant inadequacies regarding the suitability for traffic security analysis. Since the existing tracker is non-modifiable and has to be considered as a black-box system, the proposed approach is designed as post-processing technique, that operates

on the formerly generated trajectories exclusively. In addition, this work discusses the benefit of post-processing steps for generic trackers in general, on basis of the achieved results.

Figure 1.2: Birds eye view of AiM research crossroads.

This problem introduction is followed by an outline of basic multi-target tracking methods, that are applied throughout this thesis. Chapter 3 summarizes important historical and current works on the field of study of multi-target tracking, focusing on utilized methods and approaches. Furthermore, the existing tracking system is presented, by describing the set of provided features and the predominant insufficiencies. Additionally, common evaluation metrics are reviewed, considering specifically the suitability for evaluating traffic security analysis systems. Subsequent chapter 4 explains the developed methods in this work in detail, and advises the usage of the associated tuning parameters. These methods are evaluated in chapter 5, which analyzes their effect on the trajectories at the AiM research crossroads, and utilizes standard benchmarks to check the post-processing method's generic performance. Final chapter 6 briefly recalls the main insights gained during this work, concludes the overall achievements and outlines possible future steps to continue the accomplished results.

Chapter 2

Overview of Multi-Target Tracking Methods

Following the brief problem introduction, this chapter presents intuitive and formal descriptions of the problem domain and the methods, that are used in this work.

2.1 Multi-Target Tracking

Tracking is an increasingly important field within computer vision research. Essentially, tracking is the process of estimating the state of a moving object to compute its trajectory [6]. The input data does not have to be a video stream, as common radar [7] and modern lidar [8] applications prove. Furthermore, the fusion of different sources is frequently applied [9]. The classic principle of tracking-by-detection includes two main stages. In the first place, targets under consideration are detected. In the second place, each detections is associated to a target or clutter, in order to create tracks. Optional additional information about trajectories, such as velocities or orientations of the tracked objects, are extracted, depending on the scope of the tracking application. Thus, the calculation of important metrics for traffic security analysis, as time-to-collision, is enabled by tracking. Moreover, tracking methods can be divided into real-time and batch-processing algorithms. The former create their tracking estimate in each time step separately, based on a (sub-)set of previously estimated states and currently received detections. The latter process all measurements at once, and do not necessarily accomplish runtime requirements, while real-time algorithms inevitably need to be efficient.

Since trackers process detections, they are strongly dependent on the quality of the applied detection algorithm. Higher rates of clutter and missed detections potentially lead trackers to perform worse. Multi-target tracking extends the basic problem to an unknown, time-varying number of targets [10]. Accordingly the data association phase is more complex than in the single target case.

Formally, multi-target tracking is definable as follows. Given a set of detections $Z = (z_1, \ldots, z_N)$ with each z_n containing the measured features, like position, including a time step. Then a trajectory hypothesis can be interpreted as the list $T_k = \{z_{k_1}, \ldots, z_{k_{l_k}}\}$ of distinct detections in temporal order [11]. The estimated set of trajectories, resulting from a tracking algorithm, is a set $X = \{T_1, \ldots, T_K\}$ of individual hypotheses. By assuming that z_n is a Gaussian random variable with mean $\bar{z}_n$ and variance σ_0^2, the data association problem of multi-target tracking can be formulated as maximum a posteriori estimation [12]

$$\hat{X} = \arg\max_X P(X|Z) \tag{2.1}$$

$$= \arg\max_X P(Z|X)P(X) \ . \tag{2.2}$$

Moreover, expecting the trajectories to be mutually independent, and that detections can only be associated with a single trajectory, leads to the expression [11]

$$\hat{X} = \arg\max_X \prod_n P(z_n|X)P(X) \tag{2.3}$$

$$= \arg\max_X \prod_n P(z_n|X) \prod_{T_k \in X} P(T_k) \ , \tag{2.4}$$

with condition

$$T_k \cap T_l = \emptyset, \ \forall l \neq k \ . \tag{2.5}$$

Equation (2.5) defines the exclusivity constraint regarding detection association.

2.2 Minimum-Cost Network Flow

Intuitively, the objective of a minimum-cost network flow is to find an optimal path, in the sense of minimal costs, through a network to transport a certain amount of some goods from one supplying node to one demanding node. Moreover, each edge has a specified capacity that limits the amount of flow on this edge.

Graph $G = (N, E)$ represents a directed network, with edges E connecting nodes N. Each edge $(i, j) \in E$ has an assigned cost c_{ij} and a capacity u_{ij}. For each node $n_i \in N$ the supply $b(i)$ is defined. This indicates the amount of flow that is induced or requested, respectively, by the corresponding node and specifies its characteristic [13]:

Supply node	$b(i) > 0$
Demand node	$b(i) < 0$
Transshipment node	$b(i) = 0$

In addition, each edge is assigned a flow $l_{ij} \leq f_{ij} \leq u_{ij}$ during the optimization, limited by a lower bound l_{ij} and the edge's capacity. Finding the optimal flow $f^\star$ is defined as the optimization problem [14]

$$f^\star = \arg\min_f \sum_{(i,j)\in E} c_{ij} f_{ij} \ , \tag{2.6}$$

constrained by the condition

$$\sum_{\{j:(i,j)\in E\}} f_{ij} - \sum_{\{j:(j,i)\in E\}} f_{ji} = b(i), \text{ for all } i : (i,j) \in E \ . \tag{2.7}$$

This constraint indicates, that the outgoing and incoming flow of each node have to sum up to its supply. In case of transshipment nodes both flows have to compensate for each other.

In order to formulate the problem of multi-target tracking in terms of minimum-cost network flow, the maximum a posteriori estimation, described in section 2.1, is considered. Equation (2.4) consists of two terms, that can be translated as [11]

$$P(z_n|X) = \begin{cases} 1-\beta_n & \exists T_k \in X, z_n \in T_k \\ \beta_n & \text{otherwise} \end{cases} \ , \tag{2.8}$$

$$P(T_k) = P_s(z_{k_1}) \left(\sum_{i=1}^{l_k - 1} P(z_{k_{i+1}}|z_{k_i}) \right) P_t(z_{k_{l_k}}) \ . \tag{2.9}$$

Firstly, by introducing the probability β_n that detection z_n is a false alarm, the likelihood equation (2.8) is $1-\beta_n$ in case the corresponding detection is part of some track. Secondly, equation (2.9) represents the probability, that trajectory T_k starts at detection z_{k_1}, terminates at $z_{k_{l_k}}$, and has transitions between all included detections in temporal order. Taking the logarithm of equation (2.4) reduces the function to

$$\hat{X} = \arg\min_X \sum_{T_k \in X} -\log P(T_k) + \sum_n -\log P(z_n|X) \ . \tag{2.10}$$

This can be interpreted likewise the optimization of finding the optimal flow $f^\star$ in equation (2.6), through designing each detection as a node in the network flow graph [1], resulting in

$$f^\star = \arg\min_f C(f) \tag{2.11}$$

$$= \arg\min_f \sum_i c_s f_{si} + \sum_{(ij)\in E} c_{ij} f_{ij} + \sum_i c_i f_i + \sum_i c_t f_{it} \ , \tag{2.12}$$

where all $f \in \{0,1\}$ are binary variables. Condition (2.7) is applied as well. f_i specifies whether detection z_i is part of some trajectory, with $c_i = \log \frac{\beta_i}{1-\beta_i}$ being the log-likelihood. Thus, the

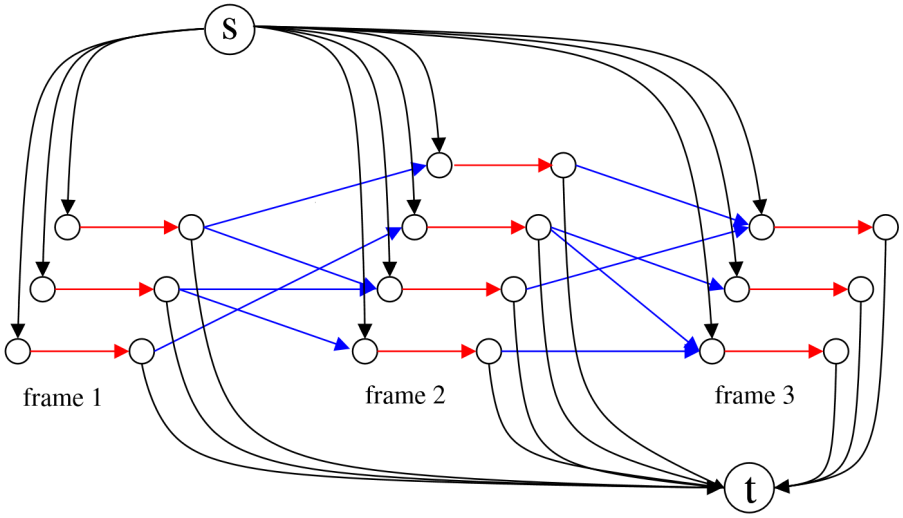


Figure 2.1: Network model for three consecutive frames without occlusion reasoning from [1].

term $\sum_i c_i f_i$ corresponds to equation (2.8) from the initial maximum a posteriori estimation. Accordingly, the other three summations originate from equation (2.9). While $\sum_{(i,j)\in E} c_{ij} f_{ij}$ coincides with the original objective function of network flow, as described in equation (2.6), the graph has to be extended in order to suit the formulation of $C(f)$.

Figure 2.1 shows such an example network, modeling a multi-target tracking problem. Each detection is modeled by a pair of nodes (u_i, v_i), connected by an edge exclusively. Consequently, f_i indicates the inclusion of this detection in some trajectory, and c_i is the related edge cost, which usually is provided by the deployed detection algorithm by means of a confidence. Furthermore, the starting and termination probabilities, represented by $\sum_i c_s f_{si}$ and $\sum_i c_t f_{it}$ are incorporated by appending a start node s and a termination node t to the graph. An incoming edge (s, u_i) from the start node s and an outgoing edge (v_i, t) to the termination node t are assigned to every detection, which allows trajectories to span partial time sequences of the scenario. In case some trajectory starts at detection z_i, the indicator $f_{si} = 1$. Respectively, the termination of some trajectory at detection z_i is specified by $f_{it} = 1$. Start and termination edges feature costs c_s and c_t accordingly. Capacities of all edges are set to $u_{ij} = 1$, to restrict each detection to a single trajectory. Except for the start and termination node, all supplies are fixed at $b(i) = 0$, while supplies for s and t are undetermined but equal. The optimization strategy attains a set of k optimal paths through the network, by increasing and accordingly decreasing the supply of s and t, in order to increase the overall flow, until a stop criterion is met [1]. This is the case when incrementing the flow will not raise a reasonable track. Thus, the number of targets is equal to the optimal amount of flow through the network. Detailed functioning of the optimization depends on its approach and implementation.

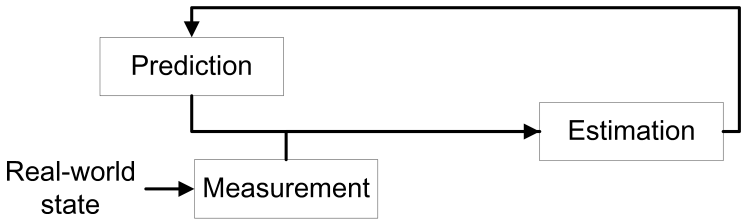


Figure 2.2: Feedback control loop approach of the Kalman filter.

2.3 Kalman Filter

Multi-target tracking methods need to rely on measurements performed by sensors of different kind, to be able to operate accurately. Since sensors underlie inaccuracies due to noise, scientists have developed filtering techniques to gain higher reliability of such systems. One of them is the Kalman filter, which was formulated by Rudolf E. Kálmán in 1960 [15]. Apart from pure noise correction capabilities, it provides an approach for the fusion of multiple sensors. Furthermore proofs exist which demonstrate that the Kalman filter is the minimum mean squared error state estimator [12]. In spite of its mathematical formulation, the filter is easily comprehensible, due to its intuitive, recursive structure. This section describes the linear Kalman filter and follows [16].

The general concept of the Kalman filter is a recursive feedback control loop to estimate the state of a system (see Figure 2.2). It consists of the time update (prediction) and measurement update phase. Since the filter is estimating the state, it involves an uncertainty. This is specified as estimation error P, which is the covariance of the estimation from the real world state. Beside raw sensor measurements, the Kalman filter uses a time-discrete dynamic model of the observed process to predict the state $\mathbf{x}$ of the system in time step k. This is formulated by the linear dynamic system equation

$$\hat{\mathbf{x}}_k = \mathbf{A}\hat{\mathbf{x}}_{k-1} + \mathbf{w}_k \ . \tag{2.13}$$

The hat denotes that the state is estimated and not the real-world state. Transition matrix $\mathbf{A}$ defines the system's dynamics, which projects the last estimate $\hat{\mathbf{x}}_{k-1}$ into the current time step. Term $\mathbf{w}_k$ introduces the process noise, which is assumed to be Gaussian distributed with zero mean and covariance $\mathbf{Q}$. Likewise, the measurement noise $\mathbf{v}$ is defined similarly with covariance $\mathbf{R}$

$$P(\mathbf{w}) \sim \mathcal{N}(0, \mathbf{Q}) \ , \tag{2.14}$$

$$P(\mathbf{v}) \sim \mathcal{N}(0, \mathbf{R}) \ . \tag{2.15}$$

Measurement $\mathbf{z}$ at discrete time k is represented by the measurement equation

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \ . \tag{2.16}$$

Prediction and measurement are combined by a weighted average, introduced by the Kalman gain matrix $\mathbf{G}$. The state estimation equation

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k + \mathbf{G}_k(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k) \tag{2.17}$$

determines the optimal estimate. Consequently, a large gain declares that the estimation mainly depends on sensor measurements, whereas small gains indicate a less abrupt adaption [12]. To calculate the optimal Kalman gain, following equation

$$\mathbf{G}_k = \mathbf{P}_k\mathbf{H}^T(\mathbf{H}\mathbf{P}_k\mathbf{H}^T + \mathbf{R})^{-1} \tag{2.18}$$

is formulated. Accordingly $\mathbf{G}$ depends on the measurement noise covariance and the estimation error. Equation 2.19 represents the adjustment of the latter following the time update, equation 2.20 after measurement update

$$\mathbf{P}_k = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{Q} \ , \tag{2.19}$$

$$\mathbf{P}_k = (\mathbb{1} - \mathbf{G}_k\mathbf{H})\mathbf{P}_k \ . \tag{2.20}$$

Due to the addition of $\mathbf{Q}$ after time update, noisy system dynamics lead to greater estimation error. This contributes to a higher Kalman gain and consequently to a lower estimation error after measurement update.

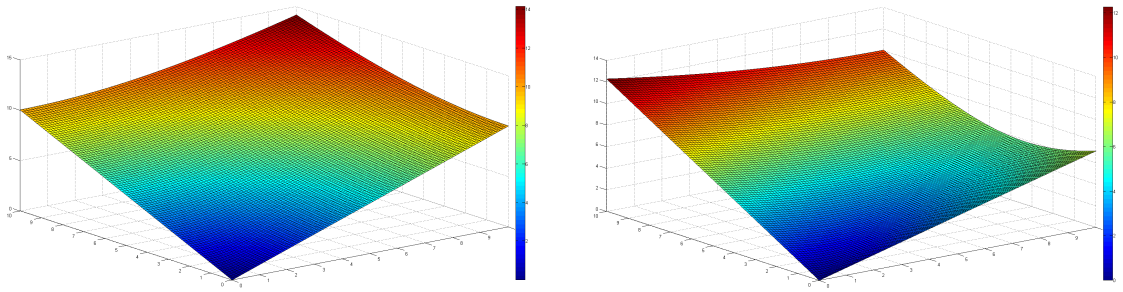
The Kalman filter offers a framework for sensor data processing. By cause of its recursive structure, it is easily implemented and provides fast execution time, being suitable for real-time applications.

2.4 Mahalanobis Distance

Distances between distinct states described by multi-variate parameters, can be calculated by use of the Mahalanobis distance. Mahalanobis introduced this measurement in 1930 [17]. Let $x_1, x_2 \in \mathbb{X}$ be states in the state space $\mathbb{X}$ of some considered system with covariance Σ. The squared Mahalanobis distance is defined as

$$\Delta^2(x_1, x_2) = (x_1 - x_2)^T\Sigma^{-1}(x_1 - x_2) \ , \tag{2.21}$$

as formulated in [18]. In case of uncorrelated state parameters with unit variance, the distance metric reduces to the Euclidean distance. This case is visualized in plot (a) of Figure 2.3, as well as



(a) (b)

Figure 2.3: Mahalanobis distance for state vectors $x_1, x_2 \in \mathbb{X}$. (a) Uncorrelated vectors with unit variance. (b) Correlated vectors.

the Mahalanobis distance for two random states

$$\text{with parameters } \begin{pmatrix} p_1 \\ p_2 \end{pmatrix} \text{ and covariance } \begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix}$$

in plot (b). The distance function is ellipsoidal for correlated state parameters with varied variances. Deviations in parameters with lower variance result in greater increases in distance than deviations in parameters with greater variance.

Chapter 3

Problem Formulation

This chapter presents an overview of related work, and analyses the existing tracking system operated by the DLR and its weaknesses. Moreover, common evaluation metrics are reviewed according to their suitability for traffic security surveillance systems.

3.1 Related Work

Due to the high amount of active research groups in the field of target tracking, there is an equivalent quantity of recent work. This section mainly focuses on modern trends regarding multi-target tracking, with emphasis on methods related to this thesis.

Multi-Target Tracking. Two of the most popular methods from the beginnings of multi-target tracking are the joint probabilistic data association filter (JPDAF) [19] and multiple hypothesis tracking (MHT) [20]. The JPDAF approach assigns measurements to targets, by computing joint posterior probabilities. MHT, on the other side, exhaustively brute-forces all potential associations. Both algorithms are exponential in computational time and memory, whereby they are limited to scenarios of low target occurrences. Recursion is one out of different aspects that categorize modern approaches. Recursive, interchangeably called online, tracking algorithms incorporate measurements at each time step directly, hence estimating trajectories based only on the last estimate and current measurements. Widespread implementations are particle filters [21,22] and the probability hypothesis density filter [23]. One main drawback of recursive approaches is the missing ability of correcting errors resulting from inaccurate measurements, once they are incorporated into estimation, which might prevent them from achieving a globally optimal set of tracks. On the other hand, non-recursive (offline) methods estimate trajectories by processing measurements of multiple time steps at once. Popular algorithms are discrete-continuous energy minimization [24],

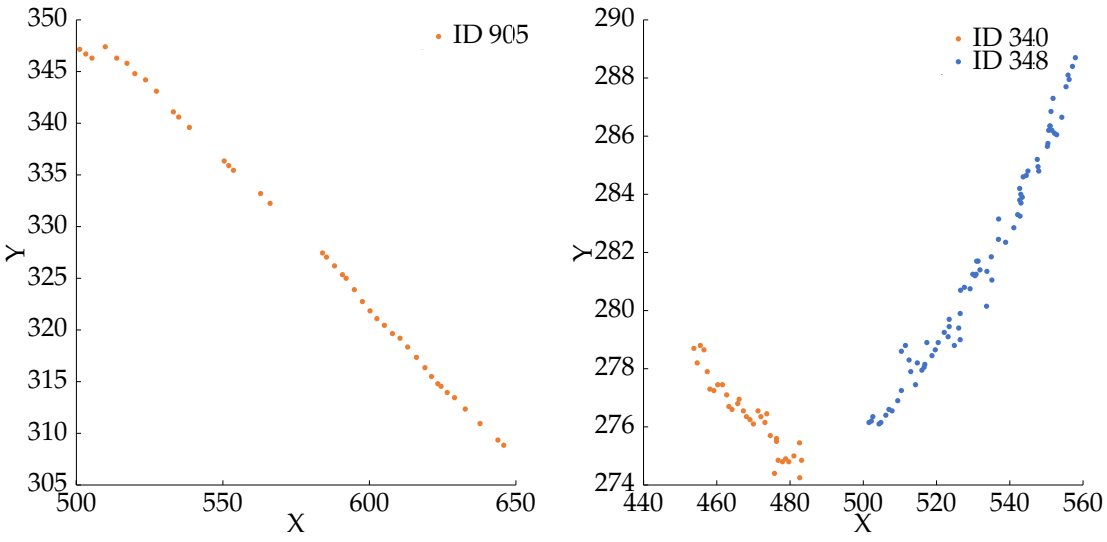
which utilizes a discrete-continuous energy function to model physical constraints and interdependencies between distinct trajectories, and graphical models solved with path finding algorithms like k-shortest paths [1] and the simplex algorithm [25]. Another separating aspect is the kind of affinity model used to distinguish detections of different targets and recognize those originating from the similar target. While the aforementioned methods heavily rely on distances between detections and accordance with some motion model, recent improvements are achieved by using appearance models [26]. Moreover, deep learning methods start to play a role in multi-target tracking, as [27] utilizes recurrent neural networks to embed the problem into neural learning frameworks.

Track Stitching. Track Stitching describes the process of linking entire tracks together, in case they likely origin from the same target. It provides a framework for resolving occlusions and extending fragmented tracks. To deal with short-term occlusions [28] links trajectories by means of bipartite graph matchings, shifting to set cover approximations for long-term occlusions. This method models tracks as nodes in a graph and generates split and merge hypotheses to reason inter-target occlusions. In [29] track stitching is solved by exploiting network flow techniques. One essential assumption of the mentioned work is independence between paths through the network, meaning trajectories. It is stated, that this assumption is valid for long tracks, that are more likely to be disrupted by obscuration than sensor uncertainty. This remark is supported by [30], which concludes a minimal loss in accuracy by path independence assumptions, if the sensors are reliable.

Post-Processing. In contrast to the domain of detection methods, in which cascaded tree-like classification procedures are common [31, 32], the application of different, aggregated tracking algorithms on one specific problem is used rather infrequently. Publicized work exists for iris movement tracking, utilizing two connected Kalman filters to compensate for eye and head movement separately [33]. Further work concerning post-processing of trajectories is provided in [11]. The authors developed an occlusion reasoning scheme, which iteratively generates artificial detections that are likely missing due to occlusions, and applies the original tracking algorithm on the expanded data.

3.2 Existing Multi-Target Tracking System

Starting point of this work is the multi-target tracking system deployed by the DLR at the AiM research crossroads. It is arranged in a setup of two separate stationary towers, while each of them is equipped with radar and video camera units. Radar and video data, as well as the information of both towers, are fused in order to produce trajectories. Despite the multi-sensor structure, there are no separate trajectories produced by each tower, but a conjoined output of the whole system.



(a) (b)

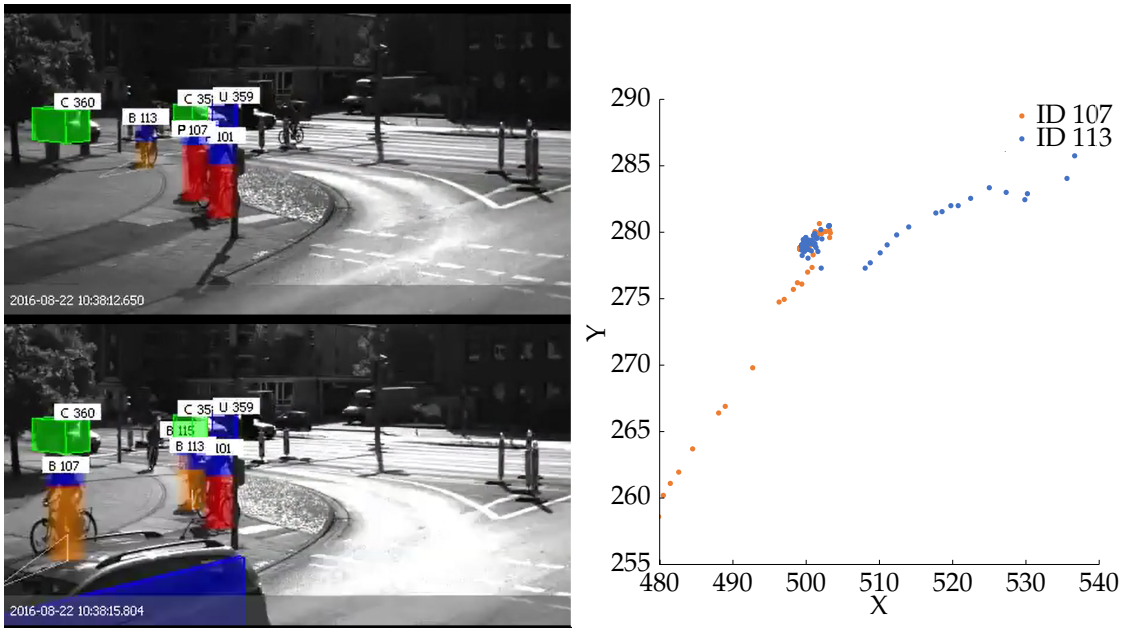
Figure 3.1: Example trajectories. (a) Track 905 loss and recover. (b) Track 340 loss and ID switch to 348

Explicit functioning of the system is unknown in this work. Furthermore, neither sensor variances nor detection confidences are available. Table 3.1 lists provided features of the detections. Each feature is generated for every estimated target with 13 Hz. Considering the system as a black box, the goal is to improve the trajectories by post-processing.

As comparable systems, the tracker encounters typical problems. Plot (a) of figure 3.1 shows the most frequent disturbance, namely fragmented tracks, and visualizes the positions of the related detections on the ground plane. The track of a car with ID 905 is initiated. While the track is

Feature	Description
$id \in \mathbb{N}$	Unique identifier of the estimated target
$t \in \mathbb{N}$	UTC time stamp of the video frame in microseconds
$p = (x, y) \in \mathbb{Q}^2$	Two-dimensional world coordinates on the ground plane
$v = (v_x, v_y) \in \mathbb{Q}^2$	Two-dimensional estimated velocity on the ground plane
$a = (a_x, a_y) \in \mathbb{Q}^2$	Two-dimensional estimated acceleration on the ground plane
$\phi \in \mathbb{N}$	Heading direction in centidegree
$b = (w, h, l) \in \mathbb{N}^3$	Width, length and height of bounding box in image pixels
$c = (p_p, \ldots, p_r) \in \mathbb{Q}^7$	Class probabilities of target for pedestrian, bicyclist, motorcycle, car, van, truck and railway vehicle

Table 3.1: Features provided by existing multi-target tracking system.



(a) (b)

Figure 3.2: ID switch between distinct targets 107 and 113. (a) Frame from video captured at research crossroads. (b) Detections of both involved targets.

lost multiple times, the tracker is able to recover the ID after re-detection, which is apparent by the gaps. Equally, some targets are not tracked at all or initiated with delay, at the expense of information loss. However, both cases are different kinds of false negatives from the perspective of post-processing, given the features from table 3.1. Missed detections before a track is created or after it is ultimately terminated are unrecoverable without a broader feature set, whereas those in between fragments of a track can be recovered by estimation. One further predominant issue is visualized in the second plot (b) of figure 3.1. Track 340 stops after some frames and is retrieved, but the ID switches. Furthermore, the plot indicates to some degree a stuttering behavior of tracks.

Figure 3.2 shows a more complex scenario. The tracker falsely switches IDs between the tracks with IDs 107 and 113, as they come close to each other. Thus, this scene represents a second type of ID switch, as multiple tracked targets are involved, in contrast to the former example.

3.3 Evaluation Metrics for Traffic Security Surveillance Systems

Multi-target tracking systems can be evaluated objectively by metrics, rating different aspects. These measures are calculated by comparing the trajectories generated by the tracker with ground truth tracks. Highly used CLEAR (Classification of Events, Activities and Relationships) metrics are presented in [34]. They value the tracker's performance regarding positional precision, by

the multi-object tracking precision (MOTP; Eq. 3.1), and target consistency, by the multi-object tracking accuracy (MOTA; Eq. 3.2). The equations

$$\text{MOTP} = \frac{\sum_{i,t} \mathrm{d}_t^i}{\sum_t \mathrm{c}_t} \,, \tag{3.1}$$

$$\text{MOTA} = 1 - \frac{\sum_t (\mathrm{fn}_t + \mathrm{fp}_t + \mathrm{mme}_t)}{\sum_t \mathrm{g}_t} \tag{3.2}$$

specify their exact computation. To calculate MOTP and MOTA, each detection in every frame used by the tracker to form a trajectory is either matched against a ground truth detection or classified as false positive. The matching is performed by nearest neighbor algorithms with a distance threshold deciding on false positives. Thereby missed detections, that is false negatives, and identity switches are computable. MOTP averages the positional deviation d_t^i for all matched detections i in each frame t by the number of matches c_t in all frames. MOTA sums up three error rates, namely the number of false negatives fn_t, false positives fp_t and mismatch errors mme_t, that occur due to identity switches, averaged by the sum of the number of ground truth targets g_t over all frames. By subtracting this average from 1, a higher multi-object tracking accuracy signifies better performance.

Another approach to analyze a tracker's effectiveness is used in [35], introducing metrics for mostly tracked (MT), mostly lost (ML) and partially tracked (PT) targets. While MT is the percentage of ground truth tracks that are tracked for at least 80% of their length, ML tracks are tracked for at most 20%. Accordingly, the PT metric includes tracks which fall in between both conditions. Since this interpretation of the tracking output uses this kind of binning, it provides a more general quality measure. In addition, the mentioned work applies the metric of fragmented tracks for evaluation purposes, that is the number of gaps within all tracks compared to their ground truth counterparts.

Common multi-target tracking benchmark data sets, as MOTChallenge [2] and KITTI [36] apply a combination of both approaches for their evaluation task, which supports their usefulness. Considering the specific problem of traffic security analysis at the AiM research crossroads, two metrics are of interest specifically. In the first place, minimization of track fragmentations increases the data's practicality. Although the track identity is correctly determined throughout such sequence, there is a loss in information during the gaps. This potentially leads to omitting critical situations. In the second place, ID switches impair the analysis of critical situations, since the tracing of detected incidents is affected, as paths of involved road users become obscure. Moreover, especially in the case of abrupt ID switches without gaps in between, the changed identity likely provokes the traffic security analysis system to falsely detect a critical situation. Besides, investigating the effect of tracking methods on the MOTA metric is beneficial, in order to beware of an overall decrease in tracking quality. This is reasonable, because improvements in individual metrics might lead to decline concerning other aspects. In contrast, the MOTP calculation is secondary

from the perspective of post-processing, utilizing the listed feature set, as the existing precision of detections is not improvable. Nevertheless, MOTP is of importance regarding principal traffic security analysis, since spatial errors distort both detection and examination of relevant scenes. Undoubtedly, the relative amount of mostly lost, partially tracked and mostly tracked targets are noteworthy. They provide enhanced insight into the tracker's performance, when combined with other metrics. Individually, they do neither consider identity consistency nor the number of fragmentations. In reverse, the latter one does not take the length of gaps into account, as the number of occurrences is counted. However, longer gaps result in higher information loss. Consequently, calculating ML, PT and MT in addition to ID switches, track fragmentations and MOTA is advantageous for the evaluation of traffic security surveillance systems.

Chapter 4

Network Flow Based Post-Processing

This chapter presents the approaches and implementations proposed to qualitatively enhance the trajectories produced at the AiM research crossroads. Two different algorithms are developed. While the first one, discarding the pre-labeled IDs of tracks, only considers spatial distances between detections, the second one deploys track stitching methods to link the existing trajectories. Thus, the distance-based approach provides a baseline to evaluate the benefit of exploiting preassigned IDs. Both use a graphical model of trajectories, solved by the k-shortest paths implementation of [1] with method-dependent changes of the graph building procedure. To be in conformance with the desired data structure, the feature set of the existing trajectories (see table 3.1) is extended, by calculating frame numbers from corresponding time stamps. Moreover, position, velocity and acceleration are transformed from world coordinates into pixels of the top view image, to easily review the results by plotting the trajectories as a video. This does not decrease the precision, since the transformation only involves shifting and scaling by factor 5. Each implementation is written in Matlab 2013a.

Chapter 5 demonstrates the performances of these methods regarding the specific problem at the AiM research crossroads, as well as their general post-processing applicability.

4.1 Network Flow by K-Shortest Paths Optimization

The optimization algorithm described in this section is introduced by [1]. Accordingly each explanation and equation refers to the mentioned work. Changes performed to adapt the algorithm to the problem of this thesis are declared. Furthermore, the authors publish an implementation in Matlab at [37] which is utilized in this work.

The authors exploit two features of graphs constructed for network flow tracking algorithms. First, since each detection can be part of only one single track, each edge (i, j) has capacity $u_{ij} = 1$.

Second, the graph is directed and acyclic, because detections are connected in temporal order, which impedes the appearance of cycles and bidirectional edges. By utilizing these conditions, the authors are able to construct a dynamic programming algorithm, whose runtime $\mathcal{O}(KN)$ is linear in the number of tracks K and processed frames N. For single target tracking the algorithm computes the optimal solution, whereas the resulting paths through the network are approximately optimal in the case of $K > 1$. This behavior results from a greedy approach.

Handling the single target case connects two phases, in order to compute the shortest path, which is the minimum-cost flow of one unit, from the source node to the termination node. Due to the graph's directed, acyclic structure, processing the nodes in temporal order is appropriate. As initialization, the cost $cost(i)$ of each node n_i in the first frame is determined by the birth cost c_s, fixed at some positive value. The costs of successive nodes are calculated recursively, as equation

$$cost(i) = \min(\pi, c_s) \ , \tag{4.1}$$

with

$$\pi = \min_{j \in N(i)} c_{ji} + cost(j) \ , \tag{4.2}$$

demonstrates. The set $N(i)$ contains all nodes, which have an outgoing edge (j, i) to node n_i. Accordingly, at each node the algorithm compares the birth cost c_s, induced by starting a path at the particular node, with the cost caused by extending the shortest path to each node n_j by edge (j, i) towards the current node. In case node n_i is the termination node t, transition cost c_{ji} equals termination cost c_t. Choosing the minimum cost results in $cost(i)$ defining the cost of the optimal path from the start node to node n_i. In addition, the $\arg\min$ value j is cached. This allows for retracing the optimal path from the start node to each other node, by simply going backwards in the graph after all frames are processed. This procedure forms the second phase of the algorithm. By starting at termination node t, the minimum-cost flow of one unit through the network is obtained. As mentioned in section 3.2, the provided data does not include detection confidences. Consequentially, in contrast to the original algorithm in [1], the equations do not include observation costs c_i.

Essentially, solving the optimization problem for an unknown number of targets involves looping the algorithm for the single target case. After each iteration, if the overall cost of the path is negative, it is appended to the result set of tracks, and all nodes and edges contained in this path are removed from the network flow graph. Thus, the following iteration will compute the next best path. The stop criterion is satisfied, if no negative cost flow is found.

4.2 Distance-Based Tracking Method

The multi-target tracking algorithm from [1] uses two-dimensional bounding boxes, in addition to the frame number to describe the state of some target. Transition costs are neglected, which means that no likelihood of correlation between two detections in subsequent frames is calculated. However, the observation cost is set to be the negative of the score of the detector. Furthermore, only detections surpassing a lower bound with respect to the intersection over union of their bounding boxes are connected. According to [26], the algorithm performs comparable to other methods, that do not utilize appearance features, concerning ID switches and track fragmentation. Therefore the main idea is adopted for the distance-based tracking method, with adjustments to suit the available data.

Detections are represented as state vectors $x = (f, p, v)$, containing the belonging frame number, plus two-dimensional position and velocity on the ground plane. Since detection confidences are not available, observation costs are discarded. Consequentially, the transition costs are investigated specifically. They are calculated by

$$c_{ij} = \frac{2 * \text{dist}(\text{predict}(x_i, f_j), p_j)}{\text{maxDist}} - 1 \ , \tag{4.3}$$

$$\forall i, j : 0 < f_j - f_i \leq 26 \ \wedge \ \text{dist}(\text{predict}(x_i, f_j), p_j) < 15 \ ,$$

with dist() being the euclidean distance, predict() estimating the position of a target in some frame moving with constant velocity, and maxDist acting as the distance threshold. Due to normalization by the upper bound, scaling with constant factor 2 and the subtraction by one, the transition costs range from -1 to 1. By predicting the position of a target in further frames, the algorithm is able to span gaps of fragmented tracks. Limiting this temporal distance to 26 frames, equaling two seconds, reduces these extensions to reasonable occasions.

As a result, the example network in figure 2.1 obtains two modifications. In the first place, each pair of nodes (u_i, v_i) is abbreviated to one single edge, hence the observation edge is omitted, since confidences are not available. Secondly, transition edges between single detections that skip multiple frames are introduced, to span fragmentations.

4.3 Two-Phase Track Stitching Method

Two existing track stitching methods are analyzed and reviewed, regarding their appropriateness for the requirements at the AiM research crossroads. The approach of [28] is designed to resolve occlusion, that occur due to interactions between targets. Consequently, only ID switches that

result from occlusions between road users are eliminated. Plot (b) in figure 3.1 shows, that this limitation does not cover all situations of changing identities, that are apparent in the processed data set. Therefore this work cannot be applied directly in this thesis. Moreover, the method creates tracklets out of plain detections by nearest neighbor matching, without considering prelabeled identities, which contradicts the intended approach of exploiting ID information. In contrast, [29] relies entirely on the distinctiveness of existing IDs, assuming that each tracklet origins from exactly one single target. Identity switches caused by fragmentations, as plot (b) in figure 3.1 shows, are solvable, by estimating the association likelihood between tracklets. However, the problem of exchanging IDs between different targets, which is illustrated in figure 3.2, remains unresolved. Consequently, this limitation prevents the method from being utilized unchanged. Nevertheless, the two-staged concept of tracklet creation, followed by tracklet linking, which is developed in [28], is adopted in this work. Though, the creation step is modified to allow for the usage of prelabeled identities, as well as the resolution of both types of ID switches.

In comparison to the distance-based approach of the preceding section, the track stitching method extends the state vectors. Beside frame number, position and velocity, the state $x = (f, p, v, a, id)$ is enhanced by the two-dimensional acceleration and the ID, assigned by the existing tracking system. As reasoned by the previous paragraph, the tracks contained in the tracking output cannot be stitched directly, as they possibly involve an undetected identity switch from one target onto another. Therefore, the initial phase breaks the trajectories built by the existing tracker into separate, shorter tracklets. Figure 4.1 illustrates the procedure of this first stage. The breaking is performed at points, where the algorithm estimates unpredicted behavior of the tracked target, since this is evidence to suggest the detection of suchlike situations. In order to determine these locations for a single track, all detections with the specific ID are isolated and supplied to a linear Kalman filter. The filter uses nearly constant acceleration motion as process model and traverses the entire trajectory. Following each prediction step, as long as the end of the track is not reached, the algorithm looks up whether a detection exists in the corresponding frame. If no detection is found, the filter continues predicting. In the opposite case, the Mahalanobis distance between the predicted and detected state is calculated, as plot (b) of figure 4.1 shows. The covariance, required for computing the distance, is given by the estimation error matrix of the Kalman filter. Since a great Mahalanobis distance signifies discrepancy between the target's expected motion and its observed behavior, the track is broken into two tracklets, if the distance exceeds a certain threshold. To continue, the filter is reinitialized with the detected state.

For the purpose of defining this threshold value, the Mahalanobis distances are analyzed for certain trajectories, which evidentially are either correctly continuous or containing an unnoticed ID switch, proofed by investigating the associated video sequence. Figure 4.2 exemplarily plots the corresponding Mahalanobis distances of the track with ID 113 from figure 3.2 and ID 905 from figure 3.1. By comparing to the positional plots of the related detections, referenced from chapter 3, it is noticeable, that the peak in frame 20 of plot (a) results from the apparent but undetected ID

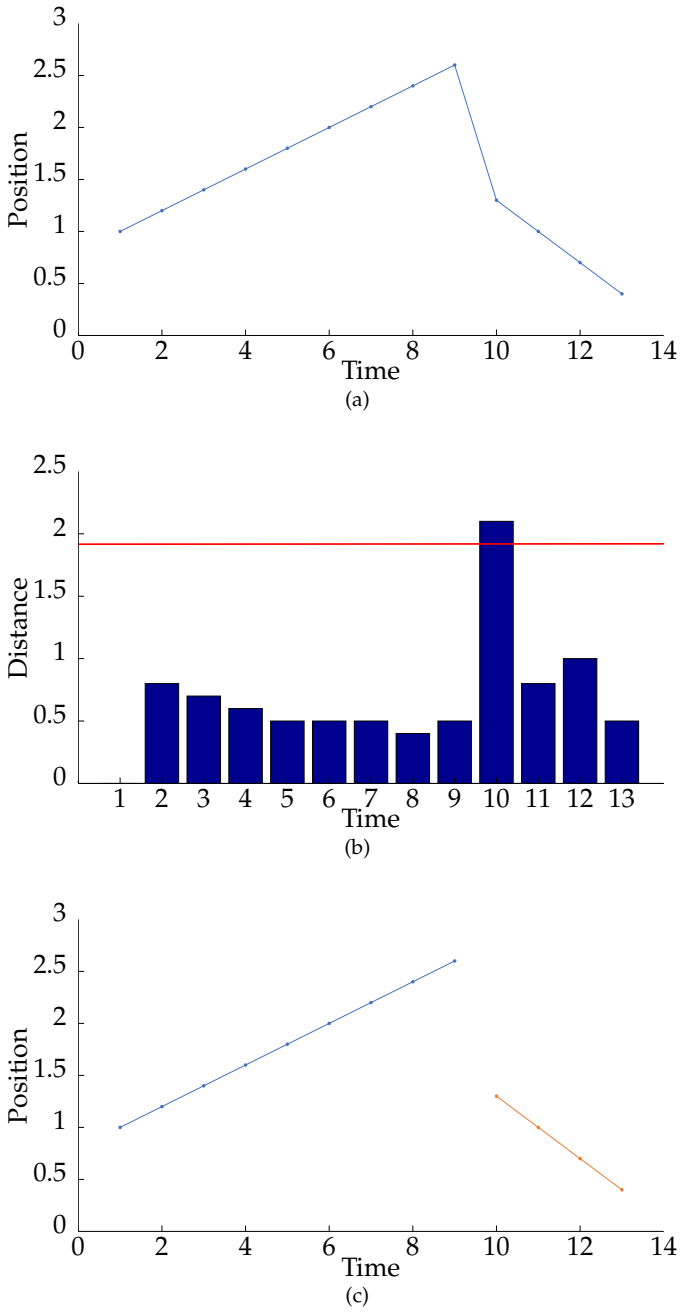


Figure 4.1: Demonstration of the breaking phase with artificial, one-dimensional detections. (a) Trajectory containing detections from multiple targets. (b) Artificial Mahalanobis distances with threshold. (c) Trajectory broken into two distinct tracklets.

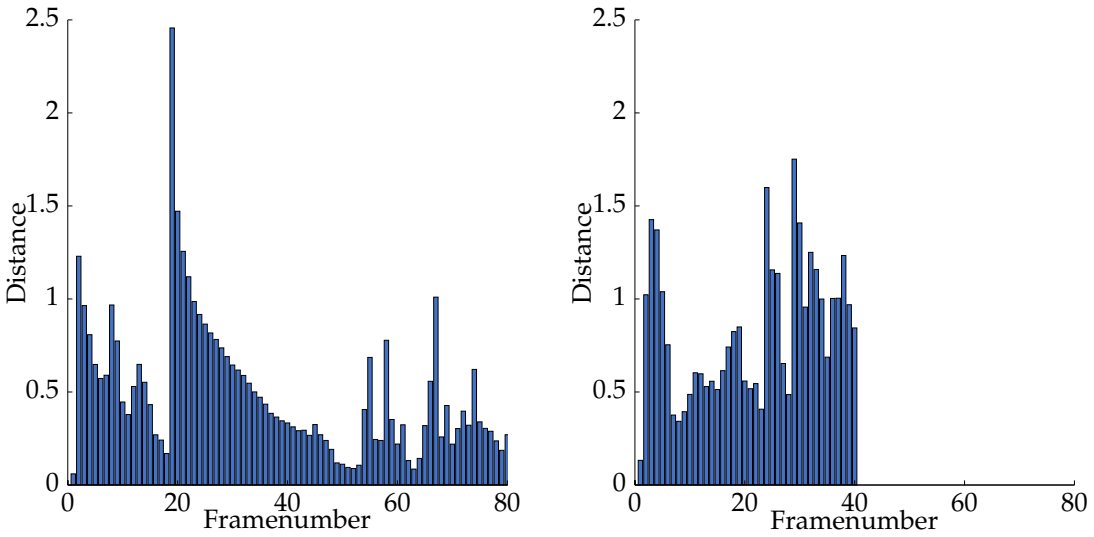


Figure 4.2: Mahalanobis distances. (a) ID 113 from figure 3.2. (b) ID 905 from figure 3.1.

switch. In contrast, the distances in plot (b) do not feature remarkable peaks, which is consistent with the tracker's ID estimation. Thus, splitting this track into separate tracklets is not reasonable, hence the threshold has to be greater than its maximum Mahalanobis distance. This analysis proposes to fix the distance threshold at 1.9. By applying the breaking step on a sample sequence of $1\,\mathrm{min}\ 30\,\mathrm{s}$, formerly containing 28 trajectories, the tracks are split into 83 tracklets, which is roughly equivalent to a factor 3 increase.

The second phase is visualized in figure 4.3. After the breaking process, the linking phase stitches the tracklets and builds the track graph. Once more a Kalman filter is used, processing one tracklet at a time. Reaching the last detection of one tracklet, the filter predicts the state of the target into subsequent frames. In case another tracklet starts in one of these frames, the Mahalanobis distance between the predicted state at this point in time and the first detection of such tracklet is calculated. Its value decides, whether a relation between the tracklets is plausible. Exemplary distances for the two possible relations in plot (a) of figure 4.3 are recorded in corresponding plot (b). If the distance comes below an upper limit, an edge is inserted into the track graph, connecting the associated detections. Edge cost c_{ij} is determined by normalizing the distance with the upper limit and scaling to the interval $[-1, 1]$.

Figure 4.4 illustrates the resulting network flow graph, modified compared to the original graph model in figure 2.1. Observation edges are discarded, as apparent by the absence of corresponding red edges. Consequently, each detection is modeled as a single node, that is connected to both start and terminal node. Moreover, only boundaries of tracklets are connected, relying on the

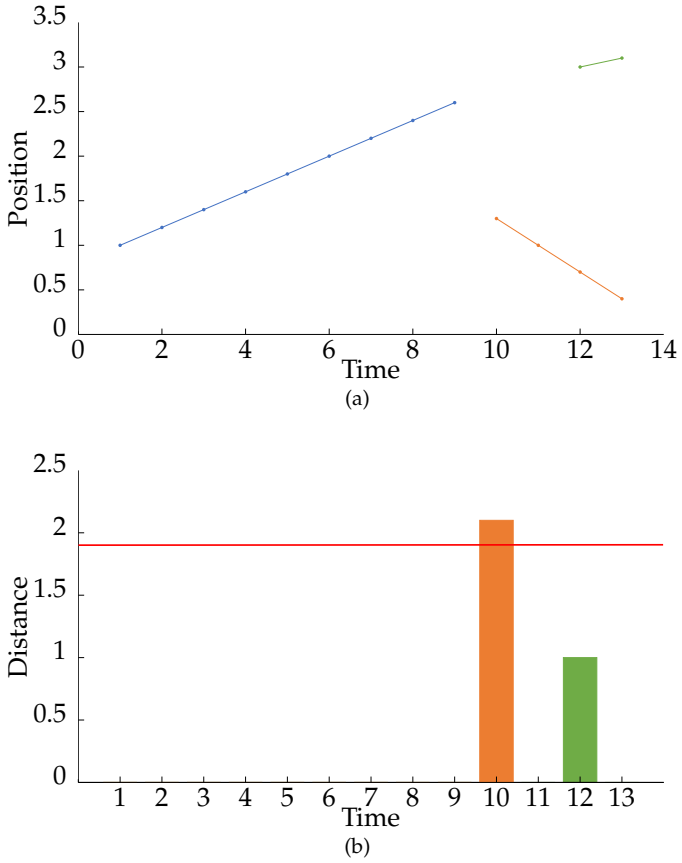


Figure 4.3: Demonstration of the linking phase with artificial, one-dimensional detections. (a) Three distinct tracklets, with fragment and ID switch between blue and green. (b) Artificial Mahalanobis distances with upper limit.

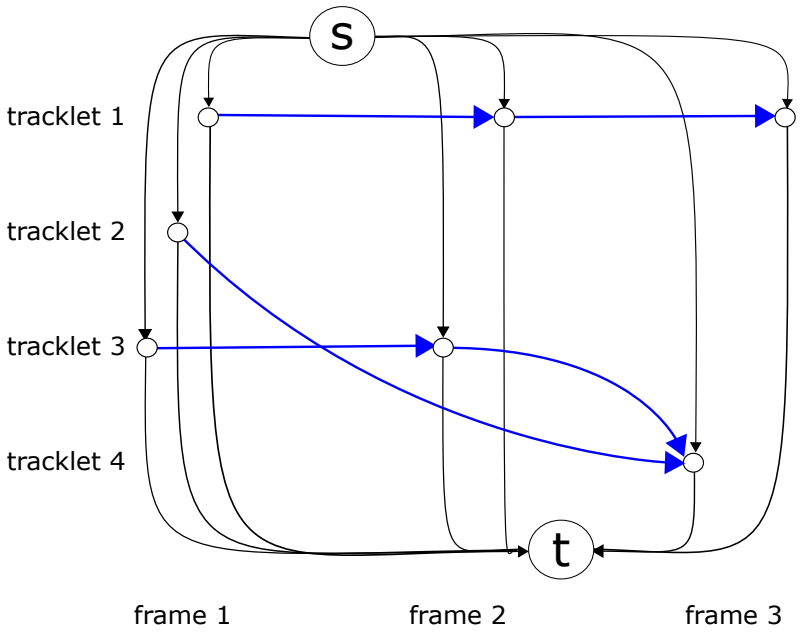


Figure 4.4: Modification of original network flow graph.

tracker's accuracy within each tracklet, created by the foregoing breaking phase. This is visualized in the mentioned figure by edges, that connect all nodes within each horizontal grouping. This alignment is performed according to affiliation of the detections to their corresponding tracklets. Beside this, the linking of separate tracklets is obvious by edges spanning across distinct groupings. In addition, resolving fragmentations is modeled by edges that skip frames. This allows for the compensation of missed detections between single track fragments.

The maximum distance value is chosen likewise as for the breaking phase. Therefore, situations of tracklets that should and should not be linked are analyzed, to examine the behavior of the Mahalanobis distance in suchlike scenarios. Accordingly, the upper limit is set to 2.6, which is greater than the breaking threshold. Consequently, the system re-links tracklets that are split before, in case no better match is present and the discrepancy does not result in a distance, which exceeds the upper limit.

For causing the filter to approach convergence, each tracklet is followed by the Kalman filter entirely, instead of starting at the last detection. Thus, the covariance that is used to compute the Mahalanobis distance is estimated by the Kalman filter, by means of the estimation error covariance. This results in more proper distance measurements. Using the Mahalanobis distance metric allows the algorithm to incorporate differences in velocity and acceleration, as well as the Kalman filter's uncertainty about its prediction. Consequently, in contrast to the distance-based method from section 4.2, detections which are close to each other, but their corresponding velocity

vectors are pointing remarkably in different directions, are not necessarily connected in the track graph or get assigned a greater cost. This is in line with intuition.

In addition to these two tracking phases, linear interpolation is applied to fill the gaps of fragmented tracks.

4.3.1 Details of Kalman Filter Implementation

Besides establishing the optimal values of the Mahalanobis distance, for both breaking threshold and upper limit of the linking phase, the Kalman filter is the second essential part of the presented tracking approach. Due to the available data of each detection, a nearly constant acceleration model is utilized. Since the tracker operates in 2D, the state vector is constructed as

$$x_k = \begin{pmatrix} p_x \\ p_y \\ v_x \\ v_y \\ a_x \\ a_y \end{pmatrix} . \tag{4.4}$$

While the measurement matrix $\mathbf{H} = \mathbb{1}$ equals the identity, the state transition matrix $\mathbf{A}$ is equivalent to Newton's equation of motion, that is

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} . \tag{4.5}$$

The motion model assumes continuous white noise affecting the acceleration, which contradicts with the time-discrete design of the linear Kalman filter. Therefore, as shown in [12], the noise is approximately discretized and the state transition equation changes to

$$\hat{\mathbf{x}}_k = \mathbf{A}\hat{\mathbf{x}}_{k-1} + \mathbf{J}\mathbf{w}_k \, , \tag{4.6}$$

with

$$\mathbf{J} = \begin{pmatrix} \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 \\ \Delta t \\ \Delta t \\ 1 \\ 1 \end{pmatrix} \tag{4.7}$$

and $\mathbf{w}_k$ being white noise of the acceleration. Matrix $\mathbf{J}$ applies this noise to both position and velocity as well. According to state transition matrix $\mathbf{A}$ they are dependent on the acceleration. Therefore they are affected by the noise with corresponding time-dependent derivation. Consequently, covariance $\mathbf{Q}$ of the noise term is calculated by [38]

$$\mathbf{Q} = \mathbf{J}\sigma_{\mathbf{w}}^2\mathbf{J}^T \tag{4.8}$$

$$= \sigma_{\mathbf{w}}^2 \begin{pmatrix} \frac{1}{4}\Delta t^4 & 0 & \frac{1}{2}\Delta t^3 & 0 & \frac{1}{2}\Delta t^2 & 0 \\ 0 & \frac{1}{4}\Delta t^4 & 0 & \frac{1}{2}\Delta t^3 & 0 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^3 & 0 & \Delta t^2 & 0 & \Delta t & 0 \\ 0 & \frac{1}{2}\Delta t^3 & 0 & \Delta t^2 & 0 & \Delta t \\ \frac{1}{2}\Delta t^2 & 0 & \Delta t & 0 & 1 & 0 \\ 0 & \frac{1}{2}\Delta t^2 & 0 & \Delta t & 0 & 1 \end{pmatrix}, \tag{4.9}$$

with

$$\sigma_{\mathbf{w}} = \mathbb{E}[\mathbf{w}_k] \tag{4.10}$$

being the expectation value of the acceleration noise, assuming it is constant during the discretized time interval. By analyzing this expected noise, the process noise covariance is calculable. For this reason, the mean acceleration of cars is considered. Following [39], typical midrange cars accelerate by 3–7 $\frac{\mathrm{m}}{\mathrm{s}^2}$ in the range of 0–60 $\frac{\mathrm{km}}{\mathrm{h}}$, which is a common velocity range at crossroads. Since the tracking system at the AiM research crossroads also tracks pedestrians and bicyclists, which accelerate less rapidly, values at the lower end of the range are tested. Moreover, especially in green-light phases, road users are not constantly accelerating, which confirms the adequacy of lower values. Fine-tuning results in setting $\sigma = 3$.

Determining the measurement noise covariance matrix $\mathbf{R}$ is done likewise, because no sensor variances are provided. Thus, the existing dataset is analyzed, examining the parameters regarding position, velocity and acceleration of tracks, whose respective parameter seems to be constant during its occurrence in the video. The fluctuations are evaluated in order to estimate the sensor's

standard deviations, formulating the covariance:

$$R = \begin{pmatrix} 16 & 0 & 0 & 0 & 0 & 0 \\ 0 & 16 & 0 & 0 & 0 & 0 \\ 0 & 0 & 200 & 0 & 0 & 0 \\ 0 & 0 & 0 & 200 & 0 & 0 \\ 0 & 0 & 0 & 0 & 40 & 0 \\ 0 & 0 & 0 & 0 & 0 & 40 \end{pmatrix} . \tag{4.11}$$

Interpreting this noise intuitively indicates, that the sensor is noisiest with respect to the target's velocity, while the deviation in acceleration and position, respectively, is pronounced orders of magnitude less. Furthermore, the noise is not dependent on the direction the target is moving, as both belonging x and y parameters are assigned the same variance. As a result of the black-box behavior of the existing tracking system, no assumptions on the sensor's functioning are made. Hence no covariances are introduced, causing the matrix to be diagonal.

4.3.2 Tuning Parameters

As the preceding content reveals, the method involves fine-tuning of certain parameters. In the following, the effects of changing these variables on the algorithm's output is explained.

Increasing the cost of starting and terminating a track, causes paths through the network to require a larger number of transition edges with negative cost, to obtain an overall negative path cost, thus being a feasible track. As a result, greater start and termination costs lead to longer trajectories. Moreover, in combination with edge costs between $[-1, 1]$, negative start and termination costs result in unintended behavior, as detections, that are not included into tracks due to excessive edge costs, are likely to produce separate tracks, since they are compensated by the constant negative costs of track initialization and termination.

The second type of tuning parameters is represented by the Mahalanobis distance threshold, applied to break tracks and stitch tracklets. Lowering the breaking threshold results in increasing the number of tracklets, created from one single track. Consequently, confidence in the input data is withdrawn. On the other side, decreasing the upper limit for the linking phase, causes the algorithm to link only reliable associations. Therefore, both limits, as well as their ratio, affect the method's behavior and have to be selected according to the desired goal. In case the system requires to prevent trajectories explicitly to consist of detections from multiple objects, it is suggested to chose a low breaking threshold, which is still greater than the linking limit. This results in a larger amount of short tracks and minimizes the probability, that one track contains an undetected ID switch. On the contrary, using a relatively high breaking threshold, that is lower than the linking limit, induces long trajectories with continuous labeling.

In addition, by adjusting the measurement noise matrix of the Kalman filter, it is achievable to specify the sensitivity of the Mahalanobis distance, regarding deviations in particular parameters. In order to increase the influence of one parameter on the distance calculation, the corresponding variance component of the matrix is reduced. This will prompt the distance to greater increases for derivations in the specific parameter, since the measurement noise matrix indirectly controls the estimation error covariance. However, this also alters the Kalman filter's estimation process, hence results in different state predictions.

Chapter 5

Application on Existing Tracking Systems

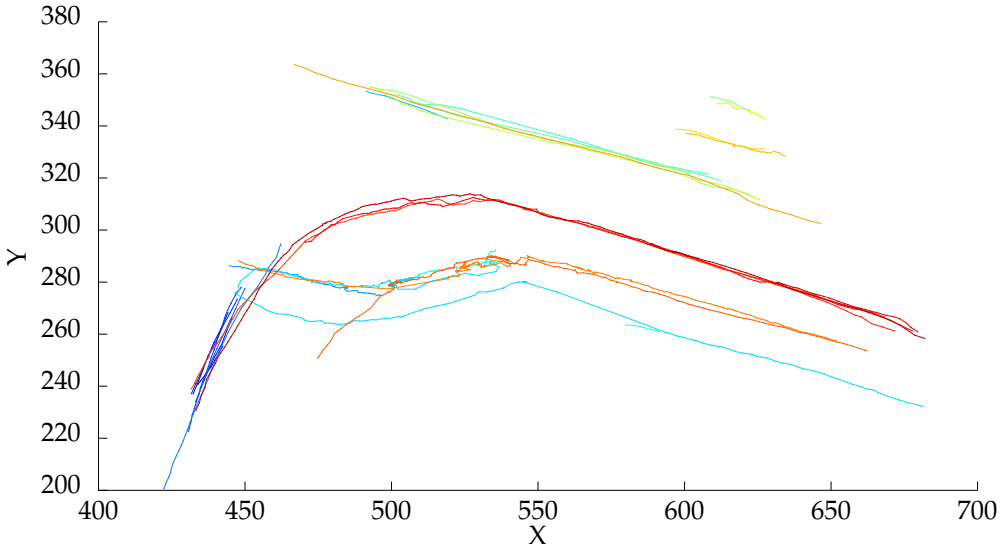
This chapter evaluates the method developed in this thesis regarding two different aspects. The first section 5.1 compares the post-processed trajectories at the AiM research crossroads with the output of the original tracking system. Section 5.2 uses the standardized 3DMOT2015 [2] benchmark in order to discuss the general benefits and drawbacks of the proposed method, applied on generic trackers. In addition, the algorithm's performance regarding computation time is reviewed.

5.1 AiM Research Crossroads Tracking System

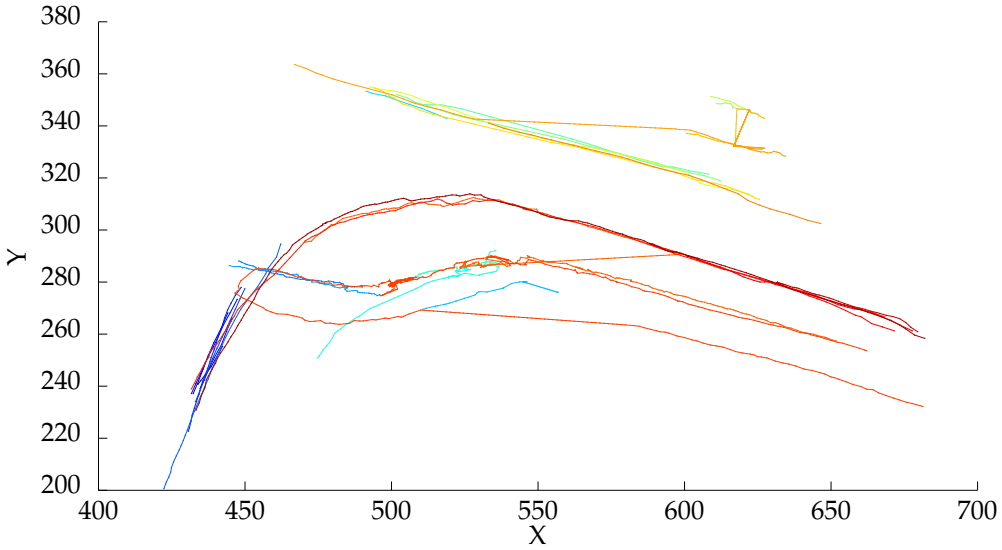
At the time of writing this thesis, no annotated ground truth tracks for the AiM research crossroads are available. Moreover, valid camera calibration parameters are not provided, with the result that evaluation for this particular setting is executed without standardized benchmark processes. Instead, the post-processed trajectories are projected onto the top view image of the crossroads, one detection at a time, in order to create a video, that serves the purpose of assessing the resulting quality by visual observation. The evaluation is performed by matching these videos with the targets' motion, that is apparent in the original video, recorded by the camera sensors at the crossroads. By reducing each scenario to scenes of obvious behavior and problems, this task becomes manageable. In addition, all trajectories of these scenes are plotted collectively, to examine the strategy of the algorithm in general. Both methods combined facilitate the identification of capabilities and incapabilities of the post-processing method.

5.1.1 Distance-Based Tracking Method

For evaluation of the distance-based tracking method, the part plotted in figure 5.1 is selected from the complete data set. The scenario represents an ordinary scene and contains 28 individual tracks. Post-processing the data by means of the distance-based algorithm increases the number



(a)



(b)

Figure 5.1: Performance of distance-based tracking method with dots representing detections. (a) Tracks generated by existing tracking system at AiM research crossroads. (b) Post-processed track output.

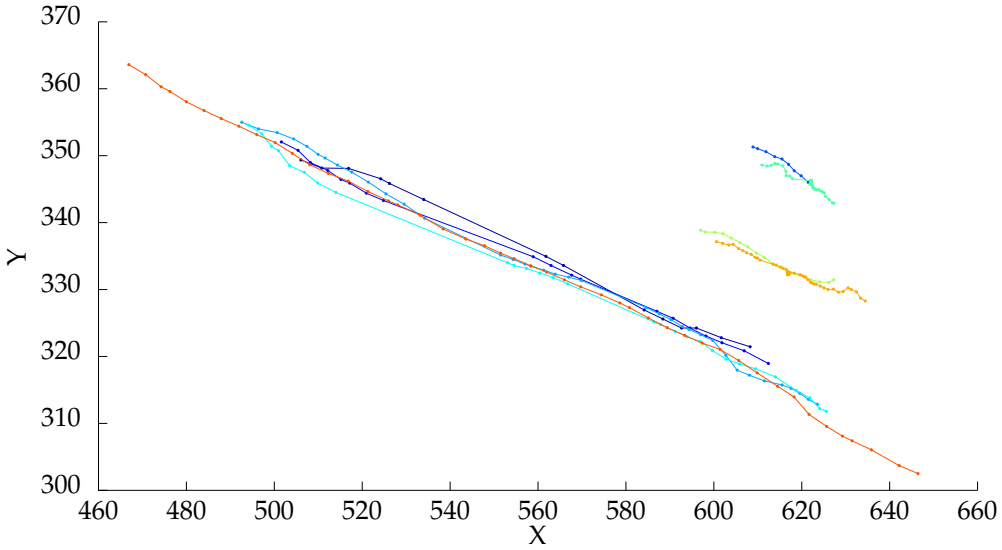
of trajectories by two. Considering figure 5.1 shows the modifications on the original tracks, introduced by post-processing. The paths' overall shape is consistent, but one specific issue reduces the tracking quality. In detail, as apparent mainly in the upper right-hand side of plot (b), cross connections between different lanes are created. Consulting the camera video of this occasion reveals, that the generated lane changes are incorrect and the number of mistakenly established ID switches exceeds the number of managed issues. Therefore, the distance-based tracking method performs inadequately, thus cannot be applied properly on trajectories of the AiM research crossroads.

5.1.2 Two-Phase Track Stitching Method

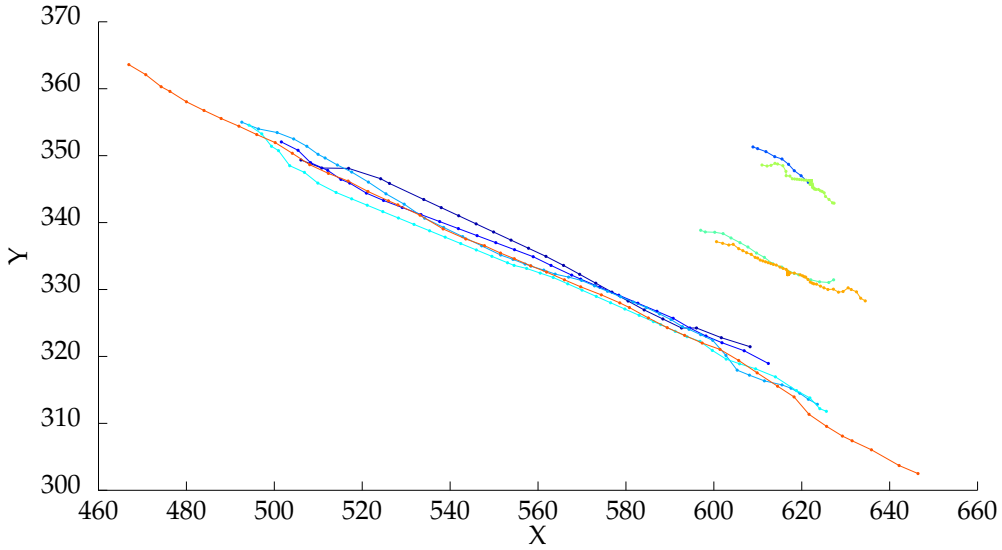
By intuition, the two-phase track stitching method is expected to achieve more accurate results, since it involves more sophisticated approaches. This anticipation is validated by analyzing the same scenario as before. In order to gain a more detailed insight, figure 5.2 and 5.3 show the trajectories of two different aspects. The first one demonstrates the influence of the post-processing method on targets, which move straightforward, while figure 5.3 points out the behavior at curves, with multiple tracks moving close to each other. Both plotted scenes are part of the same sample sequence, that was utilized also to evaluate the distance-based algorithm, but each track is assigned to the respective figure according to their spatial separation. The number of tracks, initially contained in the scene, remains constant despite post-processing.

As apparent from the plotted straight-line trajectories in figure 5.2, the post-processing algorithm merely performs marginal changes, in the case of reliable tracks, like in straight motion scenarios as the illustrated one. These tracks are handled appropriately by the existing tracking system, regarding precise identity assignments. Since no changes appear due to applying the two-phase track stitching method, it is evident, that the post-processing does not cause a considerable decline in tracking quality. Instead, the final interpolation step introduces virtual detections inside gaps between track fragments, thus recovers information lost, and consequently increases the data's usefulness. This is displayed by the cyan an blue colored trajectories. While the original tracks in plot (a) exhibit large lines without dots, which visualize detections, the post-processed trajectories in plot (b) contain evenly distributed detections in between these spaces.

Figure 5.3 principally exposes similar properties, when post-processing is applied on curvy moving tracks. Because of the higher density of targets, as well as the changes in direction, this setting is more challenging. Once more, the trajectories appear to be mainly identical. However, there is a number of adjustments from the original to the post-processed tracks. The most obvious modification is present in the center of the plot. In part (a) the tracks colored in cyan and orange cross each other two times, but the upper crossing is discarded due to partitioning of the cyan path by the breaking phase and differently linked tracklets, as visible in part (b). This part is also described in figure 3.2, revealing that the method developed in this thesis successfully resolves the

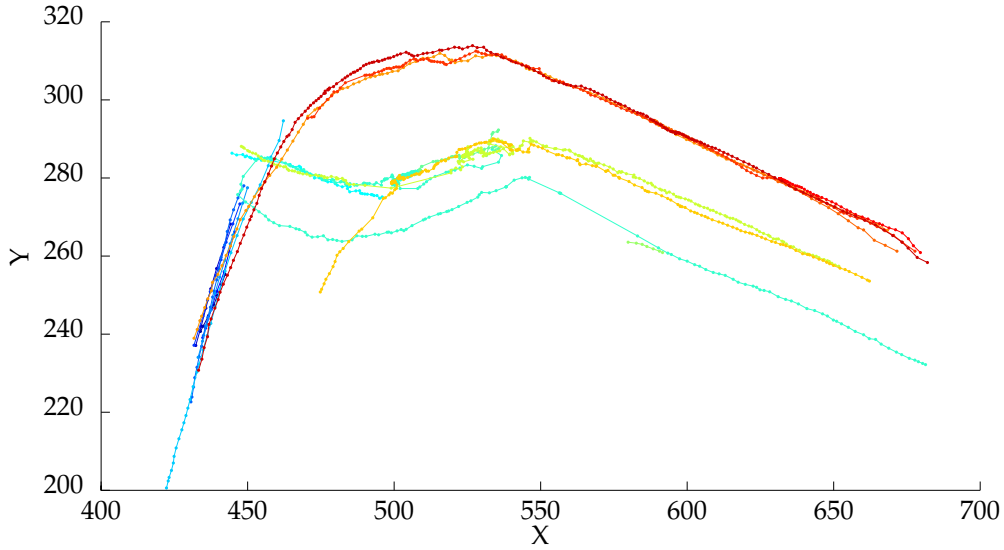


(a)

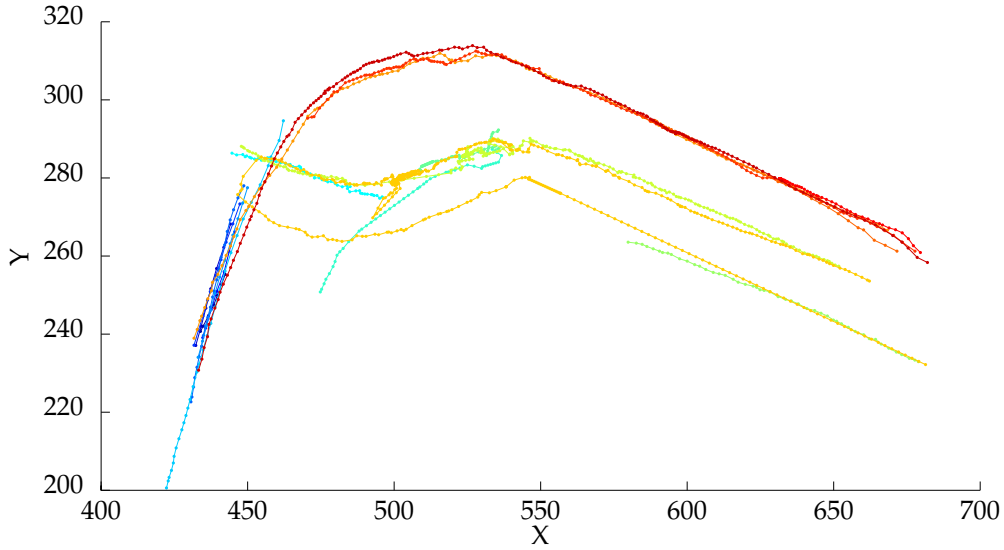


(b)

Figure 5.2: Performance of two-phase track stitching method at straight motions with dots representing detections. (a) Tracks generated by existing tracking system at AiM research crossroads. (b) Post-processed track output.



(a)



(b)

Figure 5.3: Performance of two-phase track stitching method at curvy and crossing motions with dots representing detections. (a) Tracks generated by existing tracking system at AiM research crossroads. (b) Post-processed track output.

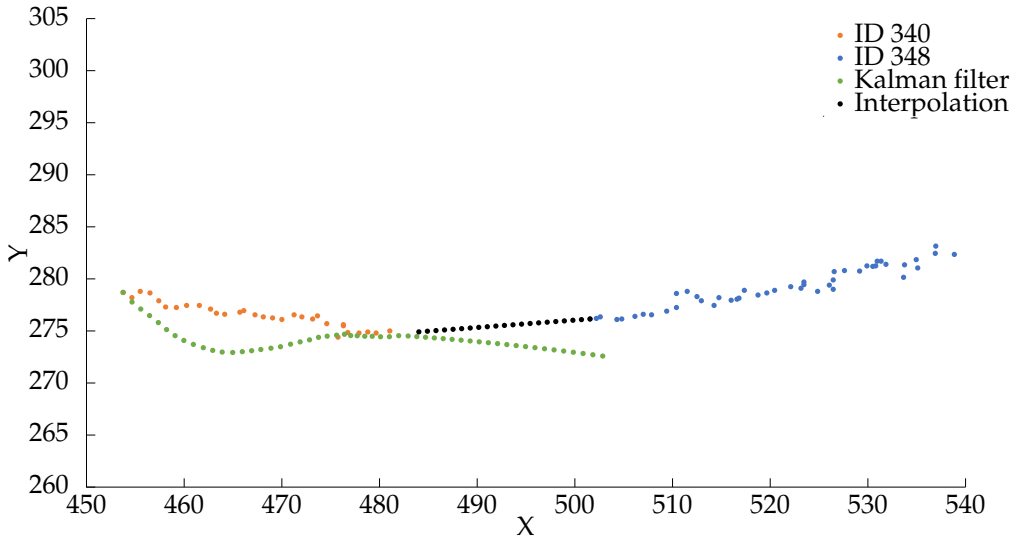


Figure 5.4: Working example of the algorithm, linking tracks with ID 340 and 348, including linear interpolation.

ID switch.

To outline the method's capabilities in more specific situations, particular tracks are analyzed isolated. Figure 5.4 visualizes the trajectories of one sample scenario. Simultaneously, it demonstrates how the described track stitching algorithm works on real data. The plotted detections are the same as in the right plot in figure 3.1. Resulting from noisy initialization data for velocity and acceleration, the Kalman filter initially performs a curve, which differs strongly from the measured positions. However, both paths merge together after 20 frames. By reaching the tracklet's end, the Kalman filter predicts its state into the start frame of the second tracklet. The associated Mahalanobis distance is 1.50 rounded, which is below the upper limit for the linking step. Accordingly, the corresponding edge cost $c_{ij} = 0.15$ is computed. Optimizing the constructed network flow graph leads to stitching both tracklets together. This causes the algorithm to interpolate linearly, in order to fill the gap between both fragments. Consequently, the final trajectory consists of the detections from the tracklet with ID 340, the interpolated detections and those from tracklet 348. Analyzing the related video from the camera sensor at the crossroads reveals, that the displayed event represents a formerly undetected ID switch, which the method is able to correct.

Likewise behavior is visualized in figure 5.5, although the scenario involves more complex computation, as the breaking phase is relevant for successfully compensating the present issue. The corresponding video sequence shows, that the track with ID 113 stops erroneously when crossing a traffic light pole, while the target proceeds continuously. The track is resumed with the new ID 119, before the initial track terminates. Therefore, simple linking of both tracklets is not possible nor

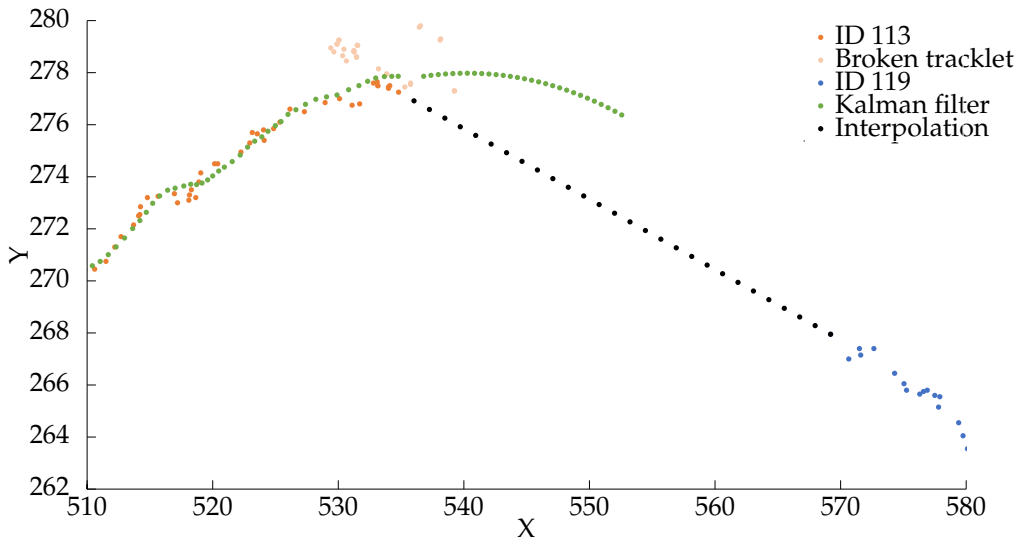


Figure 5.5: Working example of the algorithm, breaking track with ID 113 and linking with ID 119, including linear interpolation.

would it be sufficient. As the plotted detections show, the post-processing algorithm resolves this problem by breaking track 113 at the time step, when the track gets stuck. Due to this abrupt stop, the Mahalanobis distance in this time step adds up to 2.10 rounded, which is above the breaking threshold. Thus, the method is able to detect abnormal movement, resulting from falsely formed trajectories. The subsequent linking phase stitches the first tracklet, resulting from breaking the track 113, and tracklet with ID 119 together. The related Mahalanobis distance equals roughly 2.47, resulting in an edge cost of $c_{ij} = 0.9$. As the plot points out, the euclidean distance between the state, predicted by the Kalman filter, and the real state at the start of tracklet 119 is large. Still, the accordance in velocity and acceleration is sufficiently high to compensate this radius. Nonetheless, as the large Mahalanobis distance indicates, this example represents a borderline case of the method. Moreover, this instance implies the possibility of maliciously linked tracklets, due to the ability of spanning suchlike gaps of large spatial distances. However, this problem does not occur to an mentionable amount, on account of the post-processing method's design, as the aforementioned figure 5.2 and 5.3 demonstrate. In the first place, costs of edges are designed to cover positive values, beside negative costs. Consequently, edges representing such long-range connections feature costs above zero. Therefore, tracks are more likely to be terminated by the optimization algorithm before such edges. By decreasing start and termination costs, this property is emphasized. In the second place, the k-shortest paths optimization algorithm, that is utilized for optimizing the network flow problem, is designed greedily, thus tends to incorporate preferably many detections, to achieve a more negative overall path cost. As a result, longer tracks are favored over those, containing long lasting gaps.

Figure 5.6: Screenshot of AVG-TownCentre sequence from 3DMOT2015 benchmark.

To conclude the method's performance in general, the trajectories at the AiM research crossroads remain constant overall. Nevertheless, selective alterations are accomplished, mainly in order to correct falsely changed identities and to reconstruct missing detections between fragments. Consequently, the two-phase track stitching method suits the requirements of the designated setting.

5.2 Post-Processing of Generic Multi-Target Trackers

To make a contribution that is of further interest to the broad field of study of multi-target tracking, the effect of post-processing in general is reviewed in this section. As chapter 3 shows, only little investigation of such approaches and cascaded tracking algorithms is available. Due to the inappropriateness of the distance-based tracking algorithm regarding the designated setting at the AiM research crossroads, reasoned in section 5.1, the present part focuses on evaluating the track stitching method.

In order to produce comparative results, a standardized evaluation method is utilized. Since the algorithm in this thesis operates in world coordinates instead of image pixel positions, the 3DMOT2015 benchmark is selected, which is in conformance with this requirement. Another fundamental advantage of this data set is the existence of tracking results of one certain test sequence, produced by two different algorithms [40]. Usually, tracking benchmarks provide a set of detections for each sequence, but to analyze the behavior of post-processing methods, only tracking results are of interest. These tracking systems are described briefly in the following subsection, followed by the evaluation results of the trackers themselves and their post-processed counterpart.

Figure 5.6 shows a screenshot of the test sequence AVG-TownCentre that is chosen for evaluation purposes. Apparently, only pedestrians are contained in this data set. However, their movements encompass typical behavior of road users that challenges multi-target tracking algorithms. For instance, pedestrians cross and occlude each other. In addition occurring changes in velocity and direction are abrupt. Confirming the appropriateness of the sequence for evaluation, this benchmark sequence is classified as test set by the initiators of MOTChallenge. This indicates a higher complexity in comparison to training set sequences. Typically, ground truth information is not available for test scenarios. Still, the authors of the trackers, whose results are used in this chapter, provide annotated ground truth trajectories.

In its entirety the data set includes a video of $5\,\mathrm{min}$ length, containing 230 annotated individual tracks. However, the data does not contain velocity and acceleration data of targets. Therefore, to match the desired data structure of the post-processing algorithm, both are computed from the spatial differences between two consecutive detections which are labeled with the same ID. To produce more proper data, these artificial measurements are generated from a virtual sensor, simulated by a random number generator, which draws numbers according to a Gaussian distribution. Parameters of the distribution are chosen as mean being the value computed from spatial difference, and standard deviation is set according to the measurement noise covariance matrix of the Kalman filter, that is deployed in the two-phase track stitching method in this thesis. Introducing this disturbance results in more reasonable, thus more realistic velocity and acceleration data.

5.2.1 Tracking Algorithms Used for Evaluation

Both trackers used for evaluating the post-processing method developed in this thesis, are introduced by Benfold and Reid. The first one [41] is published at the British Machine Vision Conference in 2009, the second one [42] at the Conference on Computer Vision and Pattern Recognition in 2011. In the following, they are referred to according to the conference's acronyms, that is BMVC2009 and CVPR2011, respectively.

Firstly, the BMVC2009 system intends to estimate the viewing directions of moving humans, to surveil their attention. This involves detecting humans in the area of interest and their correlated head, as well as tracking their position. Benfold and Reid utilize Histograms of Oriented Gradients (HOG) for detecting heads, which analyzes appearance features in the image [43]. By making use of Kanade-Lucas-Tomasi methods (KLT) [44], the target's velocity is estimated. KLT approaches this problem by tracking a set of nearby pixels in subsequent frames, that exhibits a sufficiently strong distinguishable pattern. The resulting velocity estimate is integrated into a modified Kalman filter implementation, substituting the state transition equation. The Kalman filter estimate, supplemented by the state covariance, defines the area in the image, where the head of the target is expected in the next frame. Consequently, the HOG detector is applied only on this specific

region. In addition, the pose of heads is inspected. Both information are considered in order to determine each target's center of attention. The corresponding data set, that is used as input for the post-processing method of this thesis, additionally includes each target's position on the ground. Assuming a typical height of humans of $1.70\,\mathrm{m}$, the ground plane position is derived directly from head detections.

Secondly, the tracking algorithm from CVPR2011 is designed to obtain head images, which are sufficiently stable to facilitate activity recognition or biometric analysis. The authors emphasize the specific focus of their work on bounding box stability, rather than retaining target identities. Consequently a higher amount of ID switches might be expected. Parallel to the BMVC2009 tracker, head detections are gathered by an HOG detector and tracking is performed by applying a KLT tracker on a sliding window of frames. Moreover, multiple KLT tracks with different features are generated for each target, in order to increase robustness. Data association is implemented as Monte-Carlo Markov-Chain sampling [45], which establishes track hypotheses dependent on past and present detections. Ground position of each target's feet are calculated from camera calibration parameters.

Both trackers are applied on the identical video sequence, for which full camera calibration is available. Therefore, all spatial features refer to ground plane coordinates and the results are entirely comparable.

5.2.2 Benchmark Realization

The benchmark is executed using the development kit, provided by the initiators of the 3DMOT2015 challenge [2]. They supply Matlab scripts for different use cases, as evaluation of detections or tracking results, with automatically carrying out the problem of association between ground truth annotations and track detections. Moreover, all metrics described in section 3.3 are calculated by the script. In addition, whether to use ground plane or image pixel coordinates, is determined by analyzing the file, which contains the output trajectories, produced by the tracking algorithm, that is to be tested. The desired structure of the result file is strictly specified, so that in case of absence of bounding box information, ground plane coordinates are reviewed.

Regarding the post-processing method developed in this thesis, the implementation is entirely inherited from the application at the AiM research cross roads. However, the Mahalanobis distance for both the threshold value of the breaking phase and the upper limit of the linking phase are adapted. Manual fine-tuning proposes each to be set to 4 to achieve best results.

Metric	BMVC2009	Post-Processed	Relative Change
Mostly Tracked	84	88	4.76%
Partially Tracked	104	100	-3.85%
Mostly Lost	42	42	0.00%
False Positives	4370	4644	6.27%
False Negatives	23258	22655	-2.59%
ID Switches	100	86	-14.00%
Fragmentations	153	93	-39.22%
MOTA	61.2	61.7	0.82%
MOTP	77.4	77.2	-0.26%

Table 5.1: Benchmark results for BMVC2009 tracker and effect of post-processing by network flow.

Metric	CVPR2011	Post-Processed	Relative Change
Mostly Tracked	144	150	4.17%
Partially Tracked	60	54	-10%
Mostly Lost	26	26	0.00%
False Positives	11099	14159	27.57%
False Negatives	13758	12259	-10.90%
ID Switches	273	182	-33.33%
Fragmentations	186	111	-40.32%
MOTA	64.8	62.8	-3.09%
MOTP	79.2	79.0	-0.25%

Table 5.2: Benchmark results for CVPR2011 tracker and effect of post-processing by network flow.

5.2.3 Benchmark Results

Table 5.1 demonstrates the relative difference between the benchmark performance of the BMVC2009 tracker and its post-processed correspondent. While a shift of four tracks appears from partially tracked to mostly tracked, no change in mostly lost tracks is evident. This is in line with the functioning of the post-processing method. Since the algorithm is not designed to generate detections on itself, but rather processes only the detections included in trajectories by a tracker, it is not capable of recovering highly deficient tracks. As mentioned in section 3.3, the mostly lost metric includes all tracks with completeness below 20%. Nevertheless, the tracking result benefits from post-processing, if substantial parts of the trajectories are accessible. In this case, the algorithm is able to identify related tracklets and retrieve missed detections. This statement is fortified by the benchmark results for CVPR2011 tracker, shown in table 5.2. The relative increase in mostly tracked tracks is comparable to the BMVC2009 benchmark, even though the absolute proportion of partially tracked tracks is lower and the ratio of mostly tracked tracks is greater. This concludes to some degree an independence of the initial tracking quality.

As proven by the number of ID switches, the related but unconnected parts of partially tracked tracks do not necessarily have to be labeled with the same identity, in order to be associated with each other. The method of this thesis achieves an improvement of up to one third less switched identities, in the case of the CVPR2011 review. Regarding BMVC2009, the achievement still equals half the advancement roughly. In the same course, fragmentations are reduced by about 40% roughly, in both benchmarks. Both metrics are correlated with each other, when applying the post-processing algorithm, since resolving an ID switch entails interpolation of missed detections between both tracklets. Consequently, enhancing the maintenance of target identities decreases the number of fragmented tracks implicitly. However, the absolute change of both metrics is not inevitably identical, because ID switches can occur without the existence of gaps between trajectories. Such situations are not considered as track fragmentation in the first place. The remarkable discrepancy between changes in ID switches for CVPR2011 on the one hand, and for BMVC2009 on the other, is justified by the tracker's design, since CVPR2011 consciously aims at stable head bounding boxes. Accordingly the number of identity changes is higher, whereas the amount of tracks that are at most partially tracked is significantly lower. Thus, the post-processing method takes advantage of accurate trajectories, with respect to the number of missed detections and the detections' spatial precision, as present in CVPR2011's tracks, since small gaps are spanned more easily. This is reasonable because of only short phases that need to be bridged by means of prediction, which prevents the Kalman filter's estimate from drifting heavily.

In the same context, the absolute number of false negatives and their change as a result of post-processing is explainable. Firstly, the number of missed detections produced by CVPR2011 is lower than those of BMVC2009, due to the explained difference in design. Moreover, the metric of false negatives does not implicate identity switches, as only the occurrence of ground truth detections in

the tracking result is checked. Secondly, since the post-processing algorithm resolves ID switches more efficiently for CVPR2011 on the one hand, and the fragmented tracks of CVPR2011 are larger, hence more missed detections are recoverable, on the other hand, the decrease of false negatives is more pronounced in the mentioned data set. The latter argument is founded by the fact, that interpolation is only possible between tracklets, unlike at the boundaries. As a consequence, largely covered tracks, in other words mostly or partially tracked ones, potentially give more reason to the spanning of gaps.

All metrics mentioned so far benefit from the post-processing method. In contrast, the number of false positives is raised. To analyze the cause of this gain, different phases of the algorithm are disabled, and the benchmark results are checked against each other. According to this, the increasing number of false positives results directly from the interpolation between maliciously connected tracklets. This becomes evident, when the trajectories produced by interpolation only are evaluated and compared with the results of the complete algorithm. Applying just linear interpolation on the CVPR2011 data set gives identical results as the tracker itself. This signifies, that the CVPR2011 tracker does not produce gaps in estimated tracks. Fragments only exist, if ground truth trajectories are split by the tracker into distinct parts, labeled with separate IDs. Consequently, the number of false positives does not result from interpolation on the original tracks. This outcome indicates, that falsely linked tracklets have to occur due to post-processing. Still, recapitulating the overall impact of the post-processing method, the number of resolved identity switches vastly exceeds the newly introduced ones, as the corresponding reduction by -33.33% demonstrates. The significance of interpolation for the worsening is apparent, when breaking and linking phases are applied on the CVPR2011 trajectories as before, but the interpolation is disabled. The corresponding relative changes alter as follows. The number of false positives drops by -3.25%, compared to an earlier increase of 27.57%, and the number of false negatives rises by 13.68%, while it decreases by -10.90% when interpolation is implemented. Moreover, one track shifts from mostly tracked to partially tracked, representing a direct effect of the increase in false negatives. As a consequence it is necessary to trade the requirement of less false positives off against minimizing false negatives.

The balance between improvement in ID switches and worsening in false positives is pointed out by the MOTA metric. While it shows slightly worse overall performance results for the CVPR2011 data set, little advancement is asserted in post-processing the BMVC2009 tracker. However, again it is important to consider, that the post-processing method is developed specifically to resolve changes in track identities and decrease the number of fragmented tracks. Hence, the combination of all metrics demonstrates, that the method accomplishes this particular objective, without interfering with other aspects to an extent, that lowers the overall quality significantly. Likewise, the spatial precision of generated tracks basically remains stable, as proven by the relative change of MOTP. Besides, the minor decrease is reasonable, since linear interpolation within gaps does not simulate road users' motion correctly. Consequently, its usage does not capture adjustments in velocity,

Length	Targets	Breaking	Linking	K-Shortest Paths	Time per Target	Time per Second
10 s	11	0.120 s	0.108 s	0.006 s	0.021 s	0.023 s
88 s	28	1.372 s	1.458 s	0.053 s	0.103 s	0.033 s
273 s	53	3.986 s	3.832 s	0.139 s	0.150 s	0.029 s
558 s	101	8.045 s	6.503 s	0.288 s	0.147 s	0.027 s

Table 5.3: Computational time measurements of two-phase track stitching method for sequences of different lengths. Time per Target and Time per Second are calculated over summarized runtime entire algorithm.

leading to slightly less precise trajectories in general.

5.3 Computational Complexity and Runtime

Intentionally, the algorithm is not designed to fulfill real-time requirements, because this prerequisite is explicitly not demanded for the application at the AiM research crossroads. Nevertheless, the computation complexity and time to execute the post-processing is evaluated, in order to demonstrate the efficiency, thus usability of the algorithm.

Considering the breaking phase, a constant amount of operations is executed for each detection of every target, as the Kalman filter is applied and the Mahalanobis distance calculated. Assuming a constant number of detections per frame, the phase's complexity $\mathcal{O}(N)$ is linear in the number of processed frames. The same is true for the linking phase, because Kalman filter estimation, Mahalanobis distance computation and possibly edge creation involve constant operations. Following the former assumption of fixed detections per frame, the computational complexity is $\mathcal{O}(N)$, as well. Although the last two procedures are only applied on a subset of detections and the Kalman filter prediction is executed more often than the number of detections per track, this argument is still valid. This is justified, as constant factors are discarded, meaning that $\mathcal{O}(2N)$ equals $\mathcal{O}(N)$. Similarly, linear interpolation is linear in the number of frames. Additionally, the k-shortest-paths approach for optimizing the minimum-cost network flow problem has an $\mathcal{O}(KN)$ runtime [1]. Altogether the two-phase track stitching algorithm's computational complexity equals $\mathcal{O}(3N + KN) = \mathcal{O}(KN)$.

To assess the real-world computational time, table 5.3 compares the runtime of the individual phases for sequences of different length. In addition, the required time for each target, as well as for one second of the data set, is given. As the results show, especially the runtime per second, the required computational time scales linearly with the data set, which supports the previous computational complexity analysis. Moreover, the declared linear complexity of the k-shortest-paths optimization is verified by the measurements.

All measurements are performed using one core of an Intel i7–7500U processor, running at 2.7 GHz

base clock. The current implementation is not capable of multi-threading. However, the performance will benefit drastically from using multiple cores in parallel, although the k-shortest path optimization is restricted to single-core execution. Nevertheless, computations of both breaking and linking on the trajectories are independent from other tracks, thus can be parallelized easily. According to table 5.3, these phases require most computational time. Consequently, the overall runtime would affected beneficially.

Chapter 6

Final Conclusion of Results

Initially, this work is intended to qualitatively improve the multi-target tracking performance at the AiM research crossroads, which is operated by the German Aerospace Center. The focus is set on metrics relevant for traffic security analysis. Particularly, the biggest expense is spend on improving ID switches and track fragmentations, since these mainly interfere with the detection and analysis of critical situations. As the existing tracker is treated like a black-box system, an immediate modification is impossible. Therefore, the method proposed in this thesis is designed as batch post-processing. The developed approach is two-phased and applies track stitching mechanisms. In principle, tracklets created by the existing tracking system are linked, according to common track stitching procedures, that are combined with network flow optimization. However, by introducing a novel breaking phase, the algorithm is able to identify and recover situations of maliciously formed trajectories.

As the evaluation results demonstrate, the primary goal of improving multi-target trajectories with respect to identity switches and fragmentations is achieved. The benchmark performances provide objective evidence for proving this statement. Likewise, comparing original sequences from the AiM research crossroads data set with their post-processed counterparts, shows a remarkable increase in tracking quality. Even more, the method demonstrates its strength, by being suitable for a wider range of tracking systems, beside the one it is designed for initially. At this point, future investigation of the performance of the approach are of interest, when applied on more modern tracking algorithms. Moreover, potential for improving the two-phase track stitching method further exists. For instance, more sophisticated interpolation procedures, like Kalman smoothers, might improve tracking accuracy. In addition, the introduction of class dependent motions models for different road users encourages continuing work.

Concluding the usability of post-processing in general, it is reasonable to state, that it is a valuable tool in case of insufficiently performing, but non-modifiable tracking systems. These scenarios are specific, but still of importance, as the real-world application by the German Aerospace Center

demonstrates. Nevertheless, post-processing is no universally appropriate method, since most of the time its capabilities can be integrated directly into multi-target tracking systems. However, the approach of pipelining a set of specialized, cascaded tracking algorithms might be auspicious for in-depth research. In the field of classification this structure is frequently applied, and post-processing features multiple intersection points with classic cascading. Since this thesis emphasizes the accomplishments achieved by post-processing, cascaded tracking is promising, in parallel.

Bibliography

[1] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, "Globally-optimal greedy algorithms for tracking a variable number of objects," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 1201–1208.

[2] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "MOTChallenge 2015: Towards a benchmark for multi-target tracking," *arXiv:1504.01942 [cs]*, apr 2015, arXiv: 1504.01942. [Online]. Available: http://arxiv.org/abs/1504.01942

[3] J. S. Barnett and W. J. Karplus, "Missile tracking system," U.S. Patent 3 010 024, Nov. 21, 1961.

[4] Gartner. (2017) Top trends in the gartner hype cycle for emerging technologies, 2017. Last accessed 2017-10-05. [Online]. Available: http://www.gartner.com/smarterwithgartner/top-trends-in-the-gartner-hype-cycle-for-emerging-technologies-2017/

[5] R.-R. Chen and L. Scott, "Multi-factor cox-ingersoll-ross models of the term structure: Estimates and tests from a kalman filter model," *The Journal of Real Estate Finance and Economics*, vol. 27, no. 2, pp. 143–172, Sep 2003.

[6] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm computing surveys (CSUR)*, vol. 38, no. 4, p. 13, 2006.

[7] Y. Bar-Shalom, "Multitarget-multisensor tracking: advanced applications," *Norwood, MA, Artech House, 1990, 391 p.*, 1990.

[8] A. Dewan, T. Caselitz, G. D. Tipaldi, and W. Burgard, "Motion-based detection and tracking in 3d lidar scans," in *International Conference on Robotics and Automation*. IEEE, 2016, pp. 4508–4513.

[9] C. Premebida, G. Monteiro, U. Nunes, and P. Peixoto, "A lidar and vision-based approach for pedestrian and vehicle detection and tracking," in *Intelligent Transportation Systems Conference*. IEEE, 2007, pp. 1044–1049.

[10] E. Maggio and A. Cavallaro, *Video tracking: theory and practice*. John Wiley & Sons, 2011.

[11] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[12] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[13] M. S. Bazaraa, J. J. Jarvis, and H. D. Sherali, *Linear programming and network flows*. John Wiley & Sons, 2011.

[14] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, "Chapter iv network flows," *Handbooks in operations research and management science*, vol. 1, pp. 211–369, 1989.

[15] R. E. Kalman *et al.*, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[16] G. Welch and G. Bishop, "An introduction to the kalman filter," *Proceedings of the Siggraph Course, Los Angeles*, 2001.

[17] P. C. Mahalanobis, "On test and measures of group divergence, part i: Theoretical formulae," *Journal and Proceedings of Asiatic Society of Bengal (New series)*, vol. 26, no. 4, pp. 541–588, 1930.

[18] G. J. McLachlan, "Mahalanobis distance," *Resonance*, vol. 4, no. 6, pp. 20–26, 1999.

[19] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Multi-target tracking using joint probabilistic data association," in *Conference on Decision and Control including the Symposium on Adaptive Processes*, vol. 19. IEEE, 1980, pp. 807–812.

[20] D. Reid, "An algorithm for tracking multiple targets," *IEEE transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, 1979.

[21] P. Del Moral, "Nonlinear filtering: Interacting particle resolution," *Markov Processes and Related Fields*, vol. 2, no. 4, pp. 555–580, 1996.

[22] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. A. Wan, "The unscented particle filter," in *Advances in neural information processing systems*, 2001, pp. 584–590.

[23] B.-N. Vo and W.-K. Ma, "The gaussian mixture probability hypothesis density filter," *IEEE Transactions on signal processing*, vol. 54, no. 11, pp. 4091–4104, 2006.

[24] A. Milan, K. Schindler, and S. Roth, "Multi-target tracking by discrete-continuous energy minimization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 10, pp. 2054–2068, 2016.

[25] L. Leal-Taixé, G. Pons-Moll, and B. Rosenhahn, "Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker," in *International Conference on Computer Vision Workshops*. IEEE, 2011, pp. 120–127.

[26] L. Leal-Taixé, A. Milan, K. Schindler, D. Cremers, I. Reid, and S. Roth, "Tracking the trackers: An analysis of the state of the art in multiple object tracking," *CoRR*, vol. abs/1704.02781, 2017.

[27] A. Milan, S. H. Rezatofighi, A. R. Dick, I. D. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks." in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*. Association for the Advancement of Artificial Intelligence (AAAI), 2017, pp. 4225–4232.

[28] Z. Wu, T. H. Kunz, and M. Betke, "Efficient track linking methods for track graphs using network-flow and set-cover techniques," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 1185–1192.

[29] G. Castañón and L. Finn, "Multi-target tracklet stitching through network flows," in *Aerospace Conference, 2011 IEEE*. IEEE, 2011, pp. 1–7.

[30] S. Mori and C.-Y. Chong, "Performance analysis of graph-based track stitching," in *International Conference on Information Fusion*, vol. 16. IEEE, 2013, pp. 196–203.

[31] G. Heitz, S. Gould, A. Saxena, and D. Koller, "Cascaded classification models: Combining models for holistic scene understanding," in *Advances in Neural Information Processing Systems 21*, D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, Eds. Curran Associates, Inc., 2009, pp. 641–648.

[32] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2001, pp. I–511–I–518.

[33] X. Xie, R. Sudhakar, and H. Zhuang, "A cascaded scheme for eye tracking and head movement compensation," *Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 28, no. 4, pp. 487–490, 1998.

[34] K. Bernardin, A. Elbs, and R. Stiefelhagen, "Multiple object tracking performance metrics and evaluation in a smart room environment," in *International Workshop on Visual Surveillance, in conjunction with European Conference on Computer Vision*, vol. 6. IEEE, 2006, p. 91.

[35] Y. Li, C. Huang, and R. Nevatia, "Learning to associate: Hybridboosted multi-target tracker for crowded scene," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 2953–2960.

[36] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2012.

[37] H. Pirsiavash. (2011) Globally-optimal greedy algorithms for tracking a variable number of objects. Last accessed 2017-05-23. [Online]. Available: http://www.csee.umbc.edu/~hpirsiav/papers/tracking_release_v1.0.tar.gz

[38] X. R. Li and V. P. Jilkov, "A survey of maneuvering target tracking: Dynamic models," in *Proceedings of SPIE Conference on signal and data processing of small targets*, vol. 6, no. 4048, 2000, pp. 212–235.

[39] H. Burg and A. Moser, *Handbuch Verkehrsunfallrekonstruktion: Unfallaufnahme, Fahrdynamik, Simulation*. Springer-Verlag, 2009.

[40] B. Benfold and I. Reid. (2009) Coarse gaze estimation. Last accessed 2017-09-17. [Online]. Available: http://www.robots.ox.ac.uk/ActiveVision/Research/Projects/2009bbenfold_headpose/project.html#datasets

[41] ——, "Guiding visual surveillance by tracking human attention," in *Proceedings of the 20th British Machine Vision Conference*, September 2009.

[42] ——, "Stable multi-target tracking in real-time surveillance video," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 3457–3464.

[43] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2005, pp. 886–893.

[44] C. Tomasi and T. Kanade, "Detection and tracking of point features," 1991, School of Computer Science, Carnegie Mellon Univ. Pittsburgh.

[45] S. Oh, S. Russell, and S. Sastry, "Markov chain monte carlo data association for general multiple-target tracking problems," in *Conference on Decision and Control*, vol. 43. IEEE, 2004, pp. 735–742.