Algorithms for Multiple Ground Target Tracking

ALGORITHMS FOR MULTIPLE GROUND TARGET TRACKING

BY

QINGSONG WU

M.Eng. (Electrical and Communication Engineering)

Zhejiang University, Hangzhou, China

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Doctor of Philosophy (2017) McMaster University

(Electrical & Computer Engineering) Hamilton, Ontario, Canada

TITLE:	Algorithms for Multiple Ground Target Tracking
AUTHOR:	Qingsong Wu
	B.Sc. (Information Engineering)
	Xi'an Jiaotong University, Xi'an, China
	M.Eng. (Electrical and Communication Engineering)
	Zhejiang University, Hangzhou, China
SUPERVISOR:	Dr. Thia Kirubarajan
NUMBER OF PAGES:	xvii, 148

Dedicated to my mother, father, and wife

Abstract

In this thesis, multiple ground target tracking algorithms are studied. From different aspects of the ground target tracking, three different types of tracking algorithms are proposed according to the specialties of the ground target motion and sensors employed.

Firstly, the dependent target tracking for ground targets is studied. State dependency is a common assumption in traditional target tracking algorithms, while this may not be the true in ground target tracking as the motion of targets are constraint to certain path. To enhance the tracking algorithm for ground targets, starting with the dependency assumption, Markov Random Field (MRF) based Probabilistic Data Association (PDA) approach is derived to associate motion dependent targets. The driving behavior model is introduced to describe motion relationship among targets. The Posterior Cramer-Rao Lower Bound (PCRLB) is derived for this new motion model. Experiments and simulations show that the proposed algorithm can reduce the false associations and improve the predictions. Eventually, the proposed approach alleviates issues like the track impurity and coalescence problem and achieves better performance comparing to standard trackers assuming state independence.

Ground target tracking using cameras is then studied. To build an efficient multi-target visual tracking algorithm, fast single target visual tracking is an important

component. A novel visual tracking algorithm that has high speed and better or comparable performance to state-of-the-art trackers is proposed. The proposed approach solves the tracking task by using a mixed-motion proposal based particle filter with Ridge Regression observation likelihood calculation. This approach largely reduces the exhaustive searching in common state-of-art trackers while maintains efficient representation of the target appearance change. Experiments on 100 public benchmark videos, as well as a high frame rate benchmark, are carried out to compare the performance with the state-of-art published algorithms. The results of the experiment show the proposed tracker achieves good performance while beats other algorithms in speed with a large margin.

The proposed visual target tracker is integrated into a new multiple ground target tracking algorithm using a single camera. The multi-target tracker addresses the issues in the target detection, data association and track management aside from the single target tracker. A perspective aware detection algorithm utilizing the recent advanced Convolutional Neural Networks (CNN) based detector is proposed to detect multiple ground targets and alleviate the weakness of CNN detectors in detecting small objects. A hierarchical class tree based multi-class data association is presented to solve the multi-class association problem with potential misclassified detections. Track management is also improved utilizing the high efficiency detectors and a Support Vector Machine (SVM) based track deletion is proposed to correctly remove the dead tracks. Benchmarking is presented in experiments and results are analyzed. A case study of applying the proposed algorithm is provided demonstrating the usefulness in real applications.

Acknowledgements

While writing this thesis and summarizing my Ph.D studies at McMaster University, I could not hold my feeling to express my gratitude to a number of people.

First and foremost, I would like to acknowledge my deepest gratitude to my supervisor Dr. Thia Kirubarajan at McMaster University. Thank you for accepting me as a Ph.D student, for providing me invaluable advice and insightful guidance in studies and research, for supporting me at McMaster University both financially and spiritually, and for the assistance of my future career plans.

The people in the Estimation, Tracking and Fusion research laboratory are the greatest team members I have ever been with. Thanks to Dr. R. Tharmarasa. We worked together on several different research projects, and I learned a lot from the inspiring discussions with you. I will never forget these precious experiences. Thanks to all my friends in the lab, in particular, Dr. Yuanhao Yu and Dr. Krishanth Krishnan, for answering of my questions and suggestions. I hope you all have a bright and prosperous future.

I would like to express my gratitude to Dr. James P. Reilly and Dr. Timothy R. Field for being my supervisory committee members. I appreciate all your time spent on my supervisory meetings and reading this thesis and your advisory thoughts. I am also thankful to the administrative staff of the Electrical and Computer Engineering

department, specially Ms. Cheryl Gies, for the administrative support.

Lastly and most importantly, I would like to especially thank my parents, my wife and my uncle and aunt. This thesis would not be done without you all standing behind me. Your endless and unconditional love and support guides me and encourages me for the past, now and future.

Notation and abbreviations

Acronyms

ADAS Advanced Driver Assistance Systems.

ADS Autonomous Driving Systems.

CNN Convolutional Neural Networks.

EKF Extended Kalman Filter.

EO/IR Electro-optical/Infrared.

FOV field of view.

FPS frames per second.

GMTI Ground Moving Target Indicator.

GNN Global Nearest Neighbor.

HOG Histogram of Oriented Gradients.

i.i.d independent and identically distributed.

IHS Intelligence Highway Systems.

IMM Interacting Multiple Model.

IoU Intersection over Union.

JPDA Joint Probabilistic Data Association.

LBP local binary patterns.

LiDAR Light Detection And Ranging.

MHT Multiple Hypothesis Tracking.

MMSE Minimum Mean Square Error.

MRF Markov Random Field.

NCC Normalized Cross Correlation.

NfS Need for Speed.

PCRLB Posterior Cramer-Rao Lower Bound.

PDA Probabilistic Data Association.

PF Particle Filter.

radar RAdio Detection And Ranging or RAdio Direction And Ranging.

ResNet Residual Networks.

RPN Regional Proposal Networks.

SVM Support Vector Machine.

UKF Unscented Kalman Filter.

VTB Visual Tracker Benchmark.

Contents

Abstract **iv**

Acknowledgements **vi**

Notation and abbreviations **viii**

Acronyms viii

1 Introduction **1**

1.1 Backgrounds on Tracking and Ground Target Tracking 2

1.1.1 Object Tracking 2

1.1.2 Ground Target Tracking 2

1.2 Introduction on Object Tracking Concept and Theory 4

1.2.1 The Tracking System 4

1.2.2 Multiple Target Tracking 5

1.2.3 Visual Tracking 8

1.3 Organization of the Thesis 9

2 Joint MRF-PDA with Driving Behavior Modeling for Ground Target Tracking **11**

2.1 Introduction 14
2.2 Problem Formulation for State Dependent Target Tracking 18
2.3 Data Association for State Dependent Ground Targets 19
2.4 Decomposing the Association Probability into Markov Random Fields 22
2.4.1 Markov Random Field and its Factorization 22
2.4.2 Pairwise MRF Representation for Ground Target Data Association 23
2.4.3 Markov Random Field and JPDA 25
2.5 Driving Behavior Modeling for Association Probability 25
2.5.1 Lane-change Modeling 26
2.5.2 Car-following Model 27
2.5.3 Driving Behavior Models for Association Probability 27
2.6 State Estimation and Posterior CRLB 32
2.6.1 State Estimation for State-Dependent Targets 32
2.6.2 Posterior Cramer-Rao Lower Bound 35
2.7 Experiments 38
2.7.1 Experiment Setup 38
2.7.2 Simulation Results 39
2.8 Conclusion 55

3 Mix-state Proposal based Particle Filter with Ridge Regression for High Speed Real-time Visual Target Tracking 57

3.1 Introduction 57
3.2 Related Work 60
3.3 Mixed-motion Proposal based Particle Filtering 63

3.3.1 Bayesian Recursive Filtering for Tracking 63
3.3.2 Particle Filtering 65
3.3.3 Mixed-state Proposals for Particle Filter 68
3.3.4 Bounded Adaptive Mixed-motion Proposals 72
3.4 Observation Likelihood based on Ridge Regression 73
3.4.1 Image Features and Feature Representation 73
3.4.2 Observation Likelihood 76
3.4.3 Observation Likelihood by Ridge Regression 76
3.5 Occlusion and Partial Out-of-scene Handling 82
3.6 Experiments 84
3.6.1 Evaluation Methodology 84
3.6.2 Benchmark Datasets and Result Analysis 85
3.7 Conclusion 96

4 Multiple Ground Target Tracking using a Single Camera in Urban Scene 98

4.1 Introduction 98
4.2 Perspective Aware Object Detection 101
4.2.1 Real-time Object Detection using Convolutional Neural Networks 101
4.2.2 Perspective Aware Ground Target Detection 102
4.3 Multi-Class Data Association using 2D Assignment with Hierarchical Object Class Tree 105
4.3.1 2D Assignment 105
4.3.2 Multi-Class Data Association 107
4.3.3 Hierarchical Object Class Tree for Multi-Class Data Association 108

4.3.4 Association Cost 110
4.4 Track Management 111
4.4.1 Track Creation 112
4.4.2 Tentative Track Update 113
4.4.3 Track Deletion 114
4.4.4 Track Management Post-Process – Visual Tracking Feature Update 115
4.5 Experiments and Analysis 117
4.5.1 Experimental Setup 117
4.5.2 Training Detector for Ground Targets 118
4.5.3 Multiple Target Tracking Performance Evaluation Methods . . 119
4.5.4 Tracking Datasets and Result Analysis 120
4.5.5 Application: Monocular Camera based Vehicle Distance Estimation Using Proposed Tracking Algorithm - A Case Study . 123
4.6 Conclusion 128

5 Conclusions and Future Research **130**

5.1 Research Summary and Conclusions 130
5.2 Future Research Direction and Discussion 132

List of Figures

1.1 Different types of ground target tracking platforms 3

2.1 Vehicle relationship based on their lanes. Note that v_1 is following v_2, so their relationship is represented by $\mathcal{R}^F_{1,2}$. Also v_3 and v_4 are the adjacent neighboring vehicles of v_1, so their relationships are represented as $\mathcal{R}^A_{1,3}$ and $\mathcal{R}^A_{1,4}$, respectively. 28

2.2 Leading-following relationship formulation. 31

2.3 Sample tracking results in Scenario A. Target 2 is following target 1 on a single lane. The estimated distance moved from the origin along the road is shown. This estimated distance of target 2 should always be less than that of target 1. 41

2.4 Average position and velocity RMSE in Scenario A over 500 Monte Carlo runs ($P_d = 1$) 43

2.5 Average position and velocity RMSE in Scenario A over 500 Monte Carlo runs ($P_d = 0.9$) 44

2.6 Sample tracking results in Scenario B. Three targets are on different lanes with no lane changes. Estimated track segments are shown. . . 46

2.7 Average position and velocity RMSE in Scenario B over 500 Monte Carlo runs ($P_d = 0.9$) 48

2.8 Simulation of Scenario C. 50
2.9 Average position and velocity RMSE in Scenario C over 500 Monte Carlo runs ($P_d = 0.9$) 51
2.10 Ground target tracking simulation on a real map scenario (area: 2.3×1.8 km^2) 52
2.11 Average position and velocity RMSE in Scenario D in 100 Monte Carlo runs ($P_d = 0.9$) 54
3.1 Examples of object's self changes (better view in color). Each rows of the images are from the same target (denoted with red rectangle) captured over the time. 59
3.2 Intersection over Union (IoU) scores of sample rectangles (better view in color). IoU is calculated between red and yellow rectangles. Red rectangle is fixed and used as the reference, while yellow rectangles varies in size and position. 79
3.3 Part templates used for occlusion detection. Three different part division is used (better view in color). 83
3.4 Sample tracking results (in red rectangle) on Visual Tracker Benchmark (VTB)-100 datasets (better view in color). From the top to the bottom, sample results are take from sequence 'bolt' (frame #1, #196, #237, #349), 'CarDark' (frame #1, #28, #138, #250), 'deer' (frame #1, #28, #42, #70), 'Tiger1' (frame #1, #106, #177, #267), 'RedTeam'(frame #2, #234, #482, #667) 88
3.5 VTB-100 Tracking Results - Precision plots (better view in color). . . 90
3.6 VTB-100 Tracking Results - Precision plots (better view in color) cont. 91

3.7 VTB-100 Tracking Results - Success plots (better view in color). . . . 92
3.8 VTB-100 Tracking Results - Success plots (better view in color) cont. 93
3.9 Need for Speed (NfS) Tracking Results - Precision plots (better view in color). 95
3.10 NfS Tracking Results - Success plots (better view in color). 96
4.1 Common scaling strategies in traditional detection algorithm (left column) and the proposed perspective aware scaling strategy (right column). 102
4.2 Illustration of a 2D assignment problem. The assignment algorithm aims to find the optimal association pairs between the detections and tracks. 105
4.3 An example of hierarchical object class tree representation based on their common attributes. 109
4.4 Sample tracking results on KITTI dataset sequence '0007', targets are marked with colored rectangles and with numbered ID (better view in color). 122
4.5 Camera setup for distance estimation. 124
4.6 Illustration of the geometry for calculating distance from camera to other vehicles. 124
4.7 Sample result of estimated distance and longitudinal speed to frontal vehicles (better view in color). 127

Chapter 1

Introduction

The rapid development and advancement of today's sensing technologies have led to the generation of large amount of data at all the times from sensors installed on various platforms. Extracting useful information from the data gradually grows into a significant component of today's sensory systems. One major task is tracking any interested objects that could be observed in the sensory data. Tracking algorithms automatically produce the states of objects over time, such as trace and velocity, in the observing regions. This automated processing can handle big amount of data that limited human resources cannot process and vastly improve system efficiency. Tracking algorithms have been employed in a broad area of applications including air traffic control, maritime surveillance, intelligence transportation systems, automotive safety systems, Autonomous Driving Systems (ADS), and national defense and security surveillance. These applications require the tracking algorithms to be fast, robust and accurate, that are the main goals of this thesis to address and improve over existing approaches.

1.1 Backgrounds on Tracking and Ground Target Tracking

1.1.1 Object Tracking

Object tracking has a long history in both the radar and computer vision research communities. The fundamental task of object tracking is sequentially estimating the states of interested objects given observations or measurements from sensors in a scene or multiple scenarios.The states of targets depend on the applications, and typical states include position, velocity, and acceleration. A sequence of states formats the spatial and temporal description of the interested targets. Based on different types of sensors, various algorithms have been proposed to track multiple types of objects.

1.1.2 Ground Target Tracking

Ground target tracking focuses on studying tracking algorithms for ground targets. Ground targets mainly refer to vehicles such as cars, trucks, vans, and motorcycles, but may also include bicycles, and pedestrians. Tracking ground targets provides valuable information for many applications such as intelligent transportation, on-vehicle safety like forward collision warning and blind spot vehicle detection. Comparing to other targets, unique motion and appearance features of ground targets could be utilized to design better tracking algorithms. For example, vehicles travel in a constrained path is an important feature that can be adopted to improve prediction and association between tracks and measurements. Also, the rigidness of the vehicle shape can lead to tracking algorithms exploiting robust features based on this character.

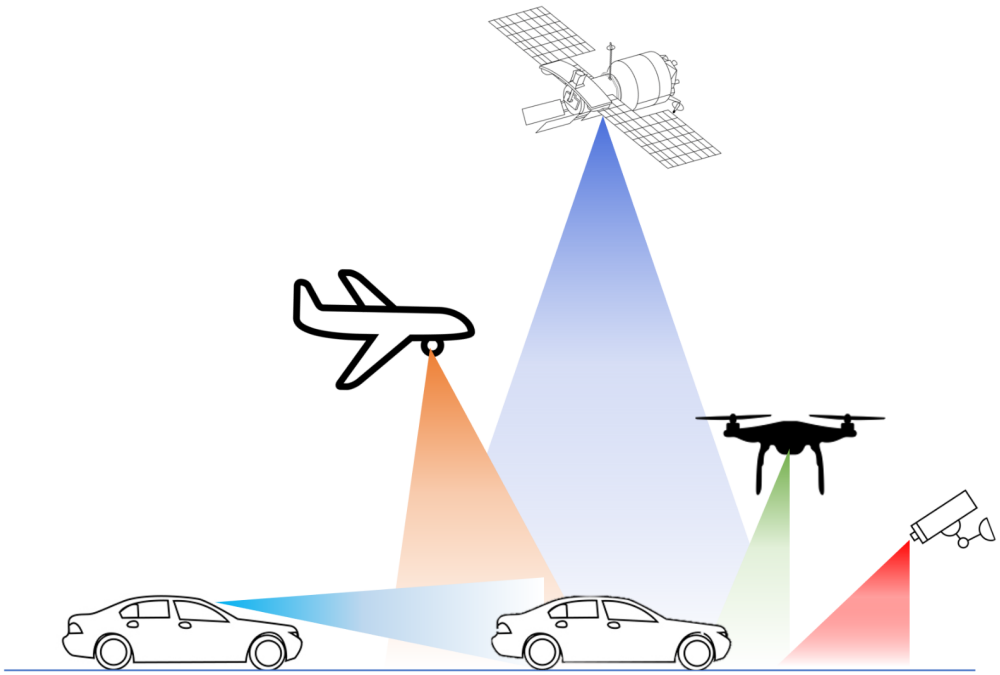


Figure 1.1: Different types of ground target tracking platforms

Multiple types of platforms could be employed to track ground targets. For example, as shown in Figure 1.1, common platforms include satellites, aircrafts, traffic surveillance systems, and automotive safety sensory platforms. Depending on the distance from sensors to the objects of interest to be tracked, the platforms could be divided into two categories.

Long-distance sensory platforms. Long distance surveillance platforms are usually mounted with sensors like long range RAdio Detection And Ranging or RAdio Direction And Ranging (radar), such as Ground Moving Target Indicator (GMTI) radar, or Electro-optical/Infrared (EO/IR) systems. These platforms cover relatively large surveillance area and handle tracking problem with large number of targets. Due to the long-distance from sensors to targets, objects to be tracked are often modeled

as point targets in radars, or small image patches with vague details in images, and the maneuverable movement range of the target is relatively small within the region of sensor's field of view (FOV) compared to the size of the FOV.

Short-distance sensory platforms. Short distance platforms are often referred as cameras, mid-range/short-range radars, or Light Detection And Ranging (LiDAR). For example, one of the common short-distance platforms are traffic cameras. Compared to long-distance sensors, these platforms have smaller range and area of FOV, and less number of targets to be tracked, however, more details of the object can be observed. For example, in radar/LiDAR measurements, multiple reflections that originated from the same target would be received. In image platform, the objects would occupy more pixels, meaning more details are captured for the object. Rich features observed from these sensors would increase the accuracy of tracking and reduce errors.

Before elaborating the proposed algorithms for ground target tracking, a brief review on general object tracking theory is presented in the following section as the fundamental basis for the derived algorithms.

1.2 Introduction on Object Tracking Concept and Theory

1.2.1 The Tracking System

The tracking system is often modeled with two stochastic processes, the state transition and measurement observation. Each process is corrupted with random noises.

The state transition process characterizes the state change of the target and the measurement observation process describes the stochastic behavior of the sensor.

1.2.2 Multiple Target Tracking

Multiple target tracking generally has three common components, data association, state estimation and track management.

Data Association Data association aims to solve the measurement origination uncertainty problems in multiple target tracking. Either deterministic (hard) or probabilistic (soft) decisions on associating corresponding targets and measurements that are made in the data association steps. Missing detections and false alarms by imperfect sensors also need to be addressed in data association. Classical data association technique is briefly discussed as follows:

Global Nearest Neighbor (GNN) The GNN method handles association within one frame of data by optimizing the cost of associating tracks and measurements with assumption that one target can only associate with at most one measurement and vice versa. Within that frame, the global optimal solution is achievable for that cost function and implementation is straightforward. However, with missing detections and existence of clutters, incorrect association can happen and leading to lost tracks. Thus, the GNN method does not perform well in scenarios with high clutter rates or low detection probabilities.

Joint Probabilistic Data Association (JPDA) The JPDA [6] method seeks probabilistic combination of all possible association between tracks and measurements

with consideration of missing detection and false alarms within one frame of data. Classical JPDA algorithm assumes a fixed number of targets and this number is known. However, this assumption could be relaxed [68] by integrating the track existence information. While calculating the association probabilities, the JPDA method enumerates all possible combinations of association events, that can significantly increase the computational load when target number is large.

Multiple Hypothesis Tracking (MHT) The MHT [77] algorithm extends the JPDA method into multiple frames with delayed decision making. MHT assumes all hypothetical possibilities of a track. The hypothesis of each track is described by a posterior probability computed based on Bayes rule. These hypotheses are generated with new observed data and propagated into next time step. However, as the time evolves, the number of hypothesis grows exponentially. Therefore, an efficient pruning algorithm is required. After the pruning procedure, only hypotheses with high probability are kept.

Assignment Algorithms Assignment algorithms convert the data association problem into an integer optimization problem. The GNN method can be viewed as a special case of assignment algorithm and is equivalent to the 2D assignment problem. Multiple frame data can be incorporated with the assignment algorithms, which is called multi-dimensional assignment.

Track Management Track management addresses the track creation, maintenance, and deletion issues. As targets enter and leave the scene, tracks need to be managed to refine the number of effective tracks, both to reduce computational

complexity and to avoid the intervention of the past data. Existing research on track management has proposed methods like probability ratio test [4] track management, and track quality based track management [84].

State Estimation State estimation outputs the estimates of targets' states by using Bayesian filtering algorithms.

Kalman Filter In a linear system with linear state transition and observation model and with Gaussian noise, Kalman filter is an optimal estimator in the sensor of Minimum Mean Square Error (MMSE). Kalman filter recursively estimates the target states by using two alternating steps: the prediction and measurement update steps. The prediction step propagates the target state and covariance in the next time step. When the observations from next time steps arrive, the measurement update step updates the filter gain as well as the states and covariance.

Extended Kalman Filter/Unscented Kalman Filter Extended Kalman Filter (EKF) and Unscented Kalman Filter (UKF) solve the problem of non-linear filtering based on the Kalman filter. EKF linearize the non-linear terms using a Taylor series approximation and then use the linear structure of Kalman filter. For example, the first order EKF introduces the approximation by using Jacobian matrix of the non-linear transition or observation functions. The UKF aims to approximate the probability density function into Gaussian distribution using sigma points. Each of the sigma points is transformed using the non-linear transition or measurement functions, and the transformed states are obtained using the transformed sigma points subsequently.

Particle Filter While Kalman filter and EKF/UKF can only handle linear or moderate non-linearity systems, and higher order of non-linearity can barely be approximated by EKF/UKF, Particle Filter (PF) utilizing a set of finite number of samples (called particles) to approximate the probability density function can implement the Bayesian filter with arbitrary distribution. However, the drawback of PF is the particle degeneration problem caused by the limited number of particles and the increasing variance of the sample weights in the updating steps.

1.2.3 Visual Tracking

Apart from the above summarized algorithms for multiple target tracking, visual tracking algorithm has its own special components.

Track Initialization Different from classical point target/measurement radar tracking systems that assumes measurements carry range/bearing information in each observation directly, the image sensor receives a matrix of color pixels. Depending on the description of the target in the image plane, track usually requires manual initialization or using an object detector to automatically generate detections to initialize and update the tracker.

Feature Extraction Directly using the image patch of the target is not a viable choice for tracking since the appearance of the target may change a lot after a number of frames. This necessitates extracting robust features to represent the target and discriminate it from the backgrounds.

Similarity Evaluation After extracting features, comparing the similarity of two targets using their features is the next task. Both statistical and machine learning based feature extraction are widely employed in the visual tracking research.

Track Generation Commonly the tracks are generated using filter algorithms or directly output the sequence of estimated observations from the images.

1.3 Organization of the Thesis

The scope of this thesis mainly involves three aspects of the tracking problems to enhance the tracking performance, the first problem is dependent ground target tracking on a long-distance surveillance platform, the second problem is real-time single target tracking, and based on the second single target tracking algorithm, multiple ground target tracking on a moving vehicle platform is addressed as the third problem.

In details, the rest of the thesis is organized as follows:

- Chapter 2 explores the dependent ground target tracking problem. By breaking the independence assumption in most of the traditional tracking algorithms, the proposed algorithm models data association with the dependency between targets, and solves it within a JPDA framework. The PCRLB of the algorithms is calculated, and experiments and comparisons with conventional tracking methods show the superiority of the proposed algorithm.

- Chapter 3 presents a fast novel single target tracking algorithm based on a new mixed-state particle filter. Within the particle filter, a new re-sampling strategy is proposed, as well as a new observation likelihood modeling and its

update. The proposed algorithms achieved comparable accuracy with state-of-art trackers, while had significant improvement in speed that is crucial in real-time applications. Benchmark on two public datasets are conducted, and the results are compared with the state-of-art visual trackers.

- Chapter 4 extends the visual target tracking work into a novel multiple target tracking algorithm. The algorithm firstly improves ground object detection using CNN, and proposed a novel hierarchical class specific data association algorithm using 2D assignment. An SVM based track deletion for the track management is also reported in this chapter. The proposed target tracking algorithm is compared with several state-of-art online object tracking algorithms on public datasets captured in real road conditions. In addition, an example of applying the proposed algorithm to one common task in Advanced Driver Assistance Systems (ADAS), estimating distance of forward vehicle, is studied.

- Chapter 5 summarizes most of the important findings and proposed algorithms in the above chapters and provides a discussion of the future research direction and improvements to the developed algorithms.

Chapter 2

Joint MRF-PDA with Driving Behavior Modeling for Ground Target Tracking

Abstract

Tracking multiple closely-spaced ground targets, even if they are resolved in sensor's filed of view, is challenging for traditional tracking approaches. In the JPDA algorithm and the MHT algorithm as well as their variants, data association and prediction are carried out with a mutual independence assumption among target states. This is true for most other target tracking algorithms as well. However, the motion of a ground target has high correlation or interaction with that of its neighbors. Ignoring vehicle state dependency during the association and prediction steps in existing approaches will result in erroneous associations, track impurity and track coalescence. To solve this problem, this paper introduces a novel technique with emphasis on improving data association without the target state independence assumption as well as on improving state prediction with an explicit dependent target motion model. For data association in the presence of targets with explicit state dependence assumption, a MRF graphical probabilistic model is introduced to model the association probabilities for such targets. Using the MRF, driving behavior models, road conditions and motion constraints are integrated into the tracking algorithm. The driving behavior models are used to calculate the joint association probability for two neighboring targets and as well to predict and estimate target states. Then data association and state update are resolved within a coupled JPDA framework. Through Monte Carlo simulations, it is shown that the proposed joint MRF-PDA algorithm can improve tracking results and alleviate issues like track impurity and coalescence in realistic ground target tracking scenarios. The proposed tracker outperforms standard trackers that assume state independence among targets.

Keywords: Ground target tracking, vehicle tracking, data association, state dependent motion, driving behavior models

2.1 Introduction

Tracking multiple ground targets has many applications including traffic control, surveillance, path planning and Intelligence Highway Systems (IHS) [24] [18]. Existing algorithms have been applied to ground target tracking problems [89, 66, 50]. However, due to high target density and constraints on target motion (e.g., lanes, safety distances), traditional algorithms are not accurate and they become computationally expensive [61]. Ground targets exhibit complex behavior due to interactions with their neighbors, which leads to correlated motion patterns and dependent target states. These issues necessitate a robust data association approach along with state dependent target motion models. This paper aims to improve ground targets tracking, especially vehicle tracking, using a novel data association approach with better motion models.

Most of the work on ground target tracking relies on the mutual state independence assumption [7], which means that one target's state does not affect any other targets' states. This assumption is reasonable in certain scenarios like missile or aircraft tracking, but in ground vehicle scenarios existing approaches perform poorly due to this assumption. After data association, estimation is treated as a single target problem, without considering the interaction with and influence of the neighbors. These shortcomings will eventually result in: 1) track crossings, coalescence or swaps with multiple lanes; 2) unstable tracks that jump around within a single lane.

To alleviate these phenomena, in the past, several data association algorithms have been proposed in the literatures for tracking closely-spaced targets. The JPDA Coupled Filter (JPDACF) [7] models the target states jointly with means, covariances, and cross-covariances that represent the correlation of target states given the

past. Although this approach inherits the drawbacks of the JPDA algorithm that shares measurements among tracks [6], which causes the above problems for closely-spaced ground targets, it generalizes a practical systematic Bayesian framework to estimate correlated or dependent target states recursively. To avoid track coalescence for closely spaced targets, the Exact Nearest Neighbour Probabilistic Data Association (ENNPDA) [26] and the Track-coalescence-avoiding Coupled Probabilistic Data Association (CPDA*)/ the Track-coalescence-avoiding Joint Probabilistic Data Association (JPDA*) [11] filters introduced new strategies that select certain measurements based on either the hypothesis with the highest weight or more-likely hypotheses for updating target states, instead of summing over all weighted possible associations. These strategies try to address track coalescence while being robust to missed detections and false alarms. Similar pruning strategies were combined with the Interacting Multiple Model (IMM) in [12] to track maneuvering targets.

To model the vehicle motions under road constraints, IMM/Variable Structure IMM (VS-IMM) algorithms, which capture the maneuvering targets' motion independently utilizing a combination of different types of motion models, were used in [50] [60]. By integrating topology map information, the VS-IMM algorithm was able to adjust the number of models and decide which models to apply under different conditions. For the data association part of the IMM/VS-IMM approaches, a 2-Dimension assignment [50] [73] or a multi-dimension assignment algorithm [21] [75] was used [8]. They formulated the association of measurements and tracks as an optimization problem that maximizes the dimensionless global likelihood ratio of the measurements conditioned on a particular assignment. This makes it possible to find the most probable hypnosis for MHT in quasi-polynomial time [21].

The Markov Random Field [51] [67] is an undirected graphical model that factorizes the joint probability of several nodes as products of local potential functions for each node and intersecting potential functions for neighborhood edges. It is widely applied in many different areas including image processing [13] [59], pattern recognition [62] and remote sensing [88]. For tracking applications, this graphical model was used as motion prior information [47, 93, 48] to track interacting targets. The MRF was also used for data association with multiple sensor network [16] by formulating the association as an inference and optimization problem.

The driving behavior models [35] [97] are a set of models that describe the correlated motion of a vehicle and its neighboring vehicles in a coordinated manner. Driving behaviors have been extensively studied for microscopic traffic simulation and management. Various mathematical models like the Gibbs model [35], Intelligent Driver Model (IDM)/Intelligent Driver Model with Memory (IDMM) model [87], and the Wiedemann model [38] were proposed to characterize drivers' operation of their vehicle and response to other vehicles. These mathematical models provide a theoretical representation of the microscopic traffic reality, and their accuracy has been greatly improved via recently developed calibration and verification algorithms [87]. The driving behavior models have not been exploited or applied in tracking literatures until recently. In [80], the driving behavior models were adopted to avoid track swaps in wide area aerial imagery tracking by using the leading-following relationship as a motion constraint.

The objectives of this paper are to mitigate track crossings, coalescence or swaps with multiple lanes and to reduce unstable tracks that jump around within a single lane. In this paper, a novel approach for tracking multiple state dependent ground

targets is proposed. One of the primary novelties of the proposed approach is the probabilistic graphical model formulation of data association for state dependent ground targets using Markov Random Fields. Ground targets not only move based on their own will or road conditions, but also interact with its neighbor vehicles [35]. To exploit such models, an MRF is created to model their joint association and to describe their motion considering their interaction, which integrates both individual and inter-target information. Then, the association problem is solved within a JPDA framework. The proposed algorithm is called the Joint MRF-PDA (JMRF-PDA) algorithm.

The proposed algorithm utilizes driving behaviors, lanes constraints and safety distance requirements in ground target scenarios. The driving behavior models specify the motion of multiple vehicles with respect to neighboring vehicles and road conditions. With properly calibrated parameters, driving behavior models are integrated into the potential function of the MRF representation of association probabilities, and used for improving prediction and estimation so that the proposed algorithm can model the motion with more precision at the lane level. In contrast, existing approaches use road level constraints. This is the next contribution of this paper.

The proposed method is demonstrated using experiments on a set of scenarios, and can yield better performance than the standard JPDA approach in terms of Root Mean Square Error (RMSE) of estimates. More importantly, the problems of track crossings, coalescence, track swaps with multiple lanes and unstable tracks are alleviated. The PCRLB [86] that quantifies the achievable estimation accuracy with dependent motion model is also derived.

The rest of the paper is organized as follows: The problem of ground target tracking is formulated in Section 2.2. Section 2.3 defines the data association problem and introduces the classical JPDA filter. In Section 2.4, association probability decomposition via the MRF graphical model for state dependent targets is derived. Then, Section 2.5 integrates the driving behavior models into the MRF for association. In Section 2.6, the state estimation algorithm and the corresponding PCRLB are presented. Section 2.7 presents the experiments and simulations and analyze the performance of the proposed algorithm. Section 2.8 summarizes and concludes this paper.

2.2 Problem Formulation for State Dependent Target Tracking

Let $\mathbf{X}_n^t = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ denote the state set of n targets at time t. Assume that the state of a target depends on its own past state and its neighboring targets' states. Then, the state of target i at time t can be modeled as

$$\mathbf{x}_i^t = F_i\left(\mathbf{x}_i^{t-1};\ \mathbf{x}_{i1}^{t-1}, \ldots, \mathbf{x}_{i\kappa}^{t-1}\right) + w_i^t \tag{2.1}$$

where $\mathbf{x}_{i1}^{t-1}, \ldots, \mathbf{x}_{i\kappa}^{t-1}$ are the states of the κ neighbors of target i at time $t-1$. Also, F_i describes the propagation of the target's state according to its past states and its neighbors' past states. Uncorrelated zero-mean Gaussian noise w_i^t is assumed.

Let $\mathbf{Z}_m^t = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ denote the measurement set at time t. For the measurement model, a linear observation model is assumed as

$$\mathbf{z}_i^t = H_i \mathbf{x}_i^t + v_i^t \tag{2.2}$$

where v_i^t is the zero-mean Gaussian measurement noise.

Then the problem is to estimate the states as

$$\hat{\mathbf{X}}^t \triangleq E\left[\mathbf{X}^t | \mathbf{Z}^t\right] \tag{2.3}$$

and the corresponding covariance matrix

$$\mathbf{P}^t \triangleq E\left[\left(\mathbf{X} - \mathbf{X}^t\right)\left(\mathbf{X} - \mathbf{X}^t\right)^T | \mathbf{Z}^t\right] \tag{2.4}$$

from a sequence of measurement sets from time 1 to time t.

2.3 Data Association for State Dependent Ground Targets

The objective of tracking is to estimate the states of ground moving targets using measurements from a single or multiple sensors like Ground Moving Target Indicator (GMTI) radars or imaging sensors. Multiple measurements originating from multiple targets are generated by the sensor, in addition to false alarms due to noise or unrelated objects. In this case, the association of received measurements with existing targets becomes a challenging step that needs to be resolved before using a

measurement to estimate the corresponding target state.

Suppose at time t, there are n tracks with states $\hat{\mathbf{X}}_n^{t-1} = \{\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_n\}$ estimated from time $t-1$, along with observed m received measurements $\mathbf{Z}_m^t = \{\mathbf{z}_1, \ldots, \mathbf{z}_m\}$ at time t. To describe the association between measurement $\mathbf{z}_j$ and track k, an indicator variable θ_{jk} is introduced. These association variables compose a single joint association event $\Theta_l^t = \{\theta_{jk_j} | j \in [0, m], k \in [0, n]$, where k_j indicates that measurement $\mathbf{z}_j$ originated from track $k\}$. Then the problem of data association can be defined as the evaluation of the probability of joint association events, $P\left(\Theta_l^t | \mathbf{Z}_m^t\right)$, given measurement set $\mathbf{Z}_m^t$.

To estimate the probability of a joint association event $P\left(\Theta_l^t | \mathbf{Z}_m^t\right)$ using Bayes formula, the probability of a joint association event could be rewritten as

$$\begin{aligned} P\left(\Theta_l^t | \mathbf{Z}_m^t\right) &= \frac{1}{c} p\left(\mathbf{Z}_m^t | \Theta_l^t, m, \mathbf{Z}^{t-1}\right) \cdot P\left(\Theta_l^t | m\right) \\ &= \frac{1}{c} p\left(\mathbf{z}_1, \ldots, \mathbf{z}_m | \theta_{1k_1}, \ldots, \theta_{mk_m}, m, \mathbf{Z}^{t-1}\right) \cdot P\left(\theta_{1k_1}, \ldots, \theta_{mk_m} | m\right) \end{aligned} \tag{2.5}$$

where $p\left(\mathbf{Z}_m^t | \Theta_l^t, m, \mathbf{Z}^{t-1}\right)$ is the likelihood function of the measurements, $P\left(\Theta_l^t | m\right)$ is the prior probability of a joint association event, m is the number of measurement and c is the normalization constant.

In the standard JPDA filter, the mutual independence among the states of the targets given the past measurements is assumed in addition to the mutual independence among individual associations [7, 6]. Then, the probability of a joint association event could be decomposed as a product of mutually independent individual associations

as

$$\begin{aligned} P\left(\Theta_l^t|\mathbf{Z}_m^t\right) &= \frac{1}{c}p\left(\mathbf{z}_1,\ldots,\mathbf{z}_m|\theta_{1k_1},\ldots,\theta_{mk_m},m,\mathbf{Z}^{t-1}\right)\cdot P\left(\theta_{1k_1},\ldots,\theta_{mk_m}|m\right) \\ &= \frac{1}{c}\frac{\phi!}{m!}\mu_F(\phi)V^{-\phi}\prod_j\left\{f_{k_j}\left[\mathbf{z}_j(k)\right]\right\}^{\tau_j}\cdot\prod_k\left(P_D^k\right)^{\delta_k}\cdot\left(1-P_D^k\right)^{1-\delta_k} \end{aligned} \tag{2.6}$$

with

$$f_{k_j}\left[\mathbf{z}_j^t\right] = \mathcal{N}\left[\mathbf{z}_j^t;\hat{\mathbf{z}}_{k_j}^{t|t-1},S_{k_j}^t\right] \tag{2.7}$$

where $\hat{\mathbf{z}}_{k_j}^{t|t-1}$ is the predicted measurement for the track k_j and $S_{k_j}^t$ is the corresponding innovation covariance.

The association events can be divided into different classes based on target detection, and association with tracks or false alarms. Then, (2.6) can be rewritten as

$$P\left(\Theta_l^t|\mathbf{Z}_m^t\right) = P\left\{\theta_{j'k'_{j'},\delta_{k'_{j'}}\neq 0}|\mathbf{Z}_m^t\right\}\cdot P\left\{\theta_{\delta_{k''}=0}|\mathbf{Z}_m^t\right\}\cdot P\left\{\theta_{j'''0}|\mathbf{Z}_m^t\right\}. \tag{2.8}$$

This equation divides the event space into three partitions. The first term on the right-hand side is the event probability that tracks $\{k'_{j'},k'=1,\ldots,\sum_{j=1}\sum_{k=1}\hat{\omega}_{jt}(\theta)\}$ are associated with measurements $\{j'\}$, the second term is the event probability that tracks $\{k'',k''=1,\ldots,n-m-\phi\}$ have no measurement, and the last term is the event probability that false alarm ($k=0$) is associated with measurements $\{j''',j'''=1,\ldots,\phi\}$.

With the state/association independence assumption, the above items are decomposed into products of individual association probabilities in (2.6). Without the mutual independence assumption of the states or association events, the key challenge in data association is to formulate the association probability $P\left(\Theta_t|\mathbf{Z}_m^t\right)$. As for the

second term in (2.8) with missed detections and the last term with false alarms, independence is still assumed with other association events. For the first term, a novel framework, which decomposes the association probabilities into undirected probabilistic graphical models (known as Markov Random Fields [51]) is used. This joint association probability is then used in state estimation. With the Markov Random Fields, the driving behavior models are integrated into the framework, which improves both association and state estimation accuracies. In the rest of the paper, the association probability, if not specified, means the first term in (2.8).

2.4 Decomposing the Association Probability into Markov Random Fields

2.4.1 Markov Random Field and its Factorization

An undirected graph $\mathcal{G} = (V, E)$ contains a set of nodes V and a set of edges E connecting the nodes. For a node $s \in V$, a random variable $\mathbf{x}_s$ is defined, and an edge indicates dependency between the nodes. If the edges satisfy the Markov property [51], then this graphical model is called a Markov Random Field.

According to the Hammersley-Clifford Theorem [28], the joint probability density of N nodes $p(\mathbf{x}_1, \ldots, \mathbf{x}_N)$ can be represented by a product of factors as

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \frac{1}{Z(\theta)} \prod_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c) \tag{2.9}$$

where $\mathcal{C}$ is the set of all cliques of $\mathcal{G}$, $\psi_c : \mathcal{X}^{\mathbf{x}_c} \rightarrow \mathbb{R}^+$ is the clique non-negative

potential function that depends only on $\mathbf{x}_c$, and

$$Z(\theta) = \sum_{\mathbf{x}} \prod_{c \in \mathcal{C}} \psi_c(\mathbf{x}_c) \tag{2.10}$$

is the partition function that ensures that all probability densities sum up to 1.

If the cliques are limited to the edges of the graph, one has a pairwise MRF [54]. For pairwise MRFs, the probability density can be factorized as

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N) = \frac{1}{Z} \prod_{s \in V} \psi_s(\mathbf{x}_s) \prod_{s,t \in E} \psi_{s,t}(\mathbf{x}_s, \mathbf{x}_t) \tag{2.11}$$

where $\psi_s(\mathbf{x}_s)$ is the node potential function that depends only on node s, and $\psi_{s,t}(\mathbf{x}_s, \mathbf{x}_t)$ is the edge potential function that depends only on the jointed nodes $\mathbf{x}_s$ and $\mathbf{x}_t$.

2.4.2 Pairwise MRF Representation for Ground Target Data Association

Pairwise MRF is a natural way to formulate association event probabilities with dependency. In ground target tracking scenarios, the vehicle states and their measurement associations are assumed to be affected only by their neighboring vehicles, which satisfies the Markov property for an MRF. This neighborhood relationship is assumed to be determined by the distance from a vehicle to other vehicles. Therefore, similar to JPDA, by using pairwise MRF, the first term in (2.8) can be factorized

into

$$\begin{aligned} P\left\{\theta_{j'k'_{j'},\delta_{k'_{j'}}\neq 0}|\mathbf{Z}_m^t\right\} &= \frac{1}{c}P\left\{\mathbf{Z}_m^t|\theta_{j'k'_{j'},\delta_{k'_{j'}}\neq 0}\right\}\cdot P\left\{\theta_{j'k'_{j'},\delta_{k'_{j'}}\neq 0}\right\} \\ &= \frac{1}{Z_c}\prod_{k=1}^{n}\psi_j\left(\mathbf{z}_j^t|\theta_{jk_j}\right)\cdot \\ &\quad \prod_{p,q\in\mathcal{C},\ p\neq q}\psi_{p,q}\left(\mathbf{z}_p^t,\mathbf{z}_q^t|\theta_{pk_p},\theta_{ql_q}\right)\cdot P\left\{\theta_{j'k'_{j'},\delta_{k'_{j'}}\neq 0}\right\} \end{aligned} \tag{2.12}$$

where $\mathcal{C}$ indicates that target p and q are neighboring vehicles on the road.

The node potential function $\psi_j\left(\mathbf{z}_j^t|\theta_{jk_j}\right)$ is defined as the association likelihood θ_{jk_j} for track k_j with observation $\mathbf{z}_j$ given by

$$\psi_k\left(\mathbf{z}_j^t|\theta_{jk_j}\right) = P\left(\mathbf{z}_j^t|\theta_{jk_j}\right). \tag{2.13}$$

The edge potential function $\psi_{p,q}\left(\mathbf{z}_p^t,\mathbf{z}_q^t|\theta_{pk_p},\theta_{ql_q}\right)$ is defined as the joint likelihood of measurement $\mathbf{z}_p$ being associated with track k_p and measurement $\mathbf{z}_q$ being associated with track l_q given by

$$\psi_{p,q}\left(\mathbf{z}_p^t,\mathbf{z}_q^t|\theta_{pk_p},\theta_{ql_q}\right) = P\left(\mathbf{z}_p^t,\mathbf{z}_q^t|\theta_{pk_p},\theta_{ql_q}\right). \tag{2.14}$$

With a pairwise MRF, a spatial relationship is modeled so that both inner-target and inter-target association probabilities are integrated. These inter-target association probabilities will re-weight the existing association probabilities for closely-spaced targets sharing the same measurements and eliminate impossible associations in motion patterns. This means that more information can be added into the association probability to resolve the ambiguity, which the classic JPDA is unable to handle well. In Section 2.5, a family of high-level driving behavior models are introduced to

calculate the association likelihoods.

2.4.3 Markov Random Field and JPDA

There are certain connections between the MRF association probability decomposition and the JPDA:

1. The MRF decomposition or representation of association probability is compatible with the classic JPDA. If one assumes independence among target states and individual associations, which empties clique set $\mathcal{C}$ and vanishes all edge potentials $\psi_{k,l}$, the MRF model will yield a form that is exactly the same as the JPDA.

2. Since the Markov Random Field representation is compatible with the classic JPDA, existing state estimation, track management and other components of a tracking algorithm can be modified to work within the new framework.

3. The MRF factorizes the association likelihood into spatial models within the JPDA framework, which can integrate additional information (e.g., driving behaviors, target features) into data association.

2.5 Driving Behavior Modeling for Association Probability

The driving behaviors are usually modeled in two ways: i) *Lane-change model* that describes how drivers choose a certain lane on the road [35] [1]; ii) *Car-following model* that describes how drivers control their vehicle velocity in response to vehicles in front

and back of them [35] [87]. With the MRF decomposition of association probabilities, the relationship of the motion between a vehicle and those of its neighbors on its left/right lane are modeled using the lane-change model while the relationship with those of the vehicles ahead/behind are modeled using the car-following model.

2.5.1 Lane-change Modeling

The lane-change models aim to specify drivers' choice of switching to or remaining on a certain lane. The reasons for a lane change can be mandatory due to ending of a lane or an incident, or discretionary due to higher utility of other lanes or routing requirements. Besides, an acceptable gap based on its velocity on the desired lane is also required for safe lane change.

Based on Ahmed's decision tree [1] for lane change, the probability of a lane change (LC) for vehicle k given its location, velocity and acceptable gap can be formulated as

$$P(\text{LC}|v_n) = [P(\text{MLC}|v_n) + (1 - P(\text{MLC}|v_n)) \cdot P(\text{DLC}|v_n)] \cdot P(\text{acceptable gap}|v_n) \tag{2.15}$$

where

- P(MLC|v_n) is the probability of mandatory lane change (MLC) given current speed
- P(DLC|v_n) is the probability of discretionary lane change (DLC) given current speed
- P(acceptable gap|v_n) is the probability of existence of acceptable gap on desired lane given current speed.

Suppose a lane change contains LC $= \{L, R, C\}$, which represents changes to the left, right or remaining on the current lane, respectively, one has a set of six neighbor nodes $\{\mathbf{x}_{k-n} | n = L^{lead}, L^{lag}, R^{lead}, R^{lag}, C^{lead}, C^{lag}\}$ for one vehicle to build the MRF.

2.5.2 Car-following Model

The car-following models describe the driving behavior in a single lane. If a driver decides to remain on the same lane, they will control the vehicle based on their own desired velocity, the gap to the leading/following vehicles and the velocity of the leading/following vehicles.

The car-following models proposed in the literature are parametric models, which establish the correlation of velocity, acceleration and gap between a vehicle and its leading/following vehicles on the same lane.

In our proposed algorithm, the car-following model is applied for better prediction before association and the leading-following relationship also influences the association to reduce track swaps on the same lane. Besides the influence of the leading vehicle, the driver will also try to reach their desired velocity v_{desire} and remain at that velocity. This is called the desired driving mode [23].

2.5.3 Driving Behavior Models for Association Probability

Association Probability with Driving Behaviors

For node potentials, the association probability is determined based on its own information, which can be calculated by

$$P\left(\mathbf{z}_j | \theta_{k_j}\right) = g\left[\mathbf{z}_j; \tilde{\mathbf{z}}_{k_j}^t, S_j^t\right] \tag{2.16}$$

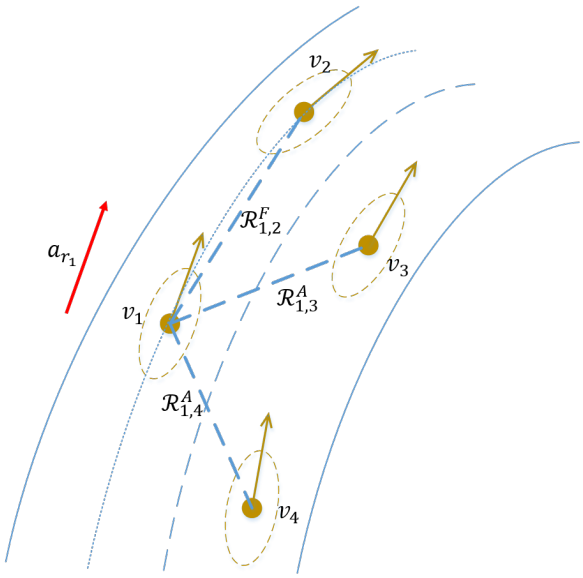


Figure 2.1: Vehicle relationship based on their lanes. Note that v_1 is following v_2, so their relationship is represented by $\mathcal{R}^F_{1,2}$. Also v_3 and v_4 are the adjacent neighboring vehicles of v_1, so their relationships are represented as $\mathcal{R}^A_{1,3}$ and $\mathcal{R}^A_{1,4}$, respectively.

where the likelihood function $g\,[*]$ is defined as in (2.6).

In real scenarios, a vehicle can either change lanes or follow a predecessor (if there is one). Let $\mathcal{C}^{l,r}$ represent the lane change to the left or right, and $\mathcal{F}$ for the car-following event. Based on the lanes, the relationship of two vehicles can be either inner-lane leading-following $\mathcal{R}^F$ or inter-lane adjacent $\mathcal{R}^A$, and this relationship can be changed if either of the two vehicles changes to the adjacent lane. An example of these relationships is given in Figure 2.1.

The association probability for edge potentials with different relationships can be

evaluated as

$$\begin{aligned} P\left(\mathbf{z}_p^t, \mathbf{z}_q^t | \theta_{pk_p}, \theta_{ql_q}\right) =& P\left(\mathbf{z}_p^t, \mathbf{z}_q^t | \theta_{pk_p}, \theta_{ql_q} | \mathcal{R}_{k_p l_q}^F\right) \cdot P\left(\mathcal{R}_{k_p l_q}^F\right) + \\ & P\left(\mathbf{z}_p^t, \mathbf{z}_q^t | \theta_{pk_p}, \theta_{ql_q} | \mathcal{R}_{k_p l_q}^A\right) \cdot P\left(\mathcal{R}_{k_p l_q}^A\right) \end{aligned} \tag{2.17}$$

where

$$P\left(\mathbf{z}_p^t, \mathbf{z}_q^t | \theta_{pk_p}, \theta_{ql_q} | \mathcal{R}_{k_p l_q}\right) = \mathcal{N}\left\{ \begin{bmatrix} \mathbf{z}_p \\ \mathbf{z}_q \end{bmatrix}; \begin{bmatrix} \tilde{\mathbf{z}}_{k_p}^t \\ \tilde{\mathbf{z}}_{l_q}^t \end{bmatrix}, \begin{bmatrix} \mathbf{0} & S_{pq}^t \\ S_{qp}^t & \mathbf{0} \end{bmatrix} \right\} \cdot Q_{\mathcal{R}}\left(\mathbf{z}_p^t, \mathbf{z}_q^t\right) \tag{2.18}$$

is the joint likelihood of measurements (p, q) associated to target (k_p, l_q), respectively, given the relationship $\mathcal{R}_{k_p l_q}^{\{F,A\}}$. In the above, $Q_{\mathcal{R}}\left(\mathbf{z}_p^t, \mathbf{z}_q^t\right)$ is a penalty function that ensures that measurements (p, q) have relationship $\mathcal{R}_{k_p l_q}^{\{F,A\}}$. This penalty term will reduce the likelihood of wrong associations that may result in track swaps. The joint likelihood is evaluated on inter-target innovation covariance corresponding to measurements, excluding inner-target innovation to reduce redundant information. In the above, $P\left(\mathcal{R}_{k_p l_q}^{\{F,A\}}\right)$ is the prior probability of the relationship $R_{k_p l_q}^{\{F,A\}}$.

The penalty term in (2.18) can be defined for $\mathcal{R}^F$ and $\mathcal{R}^A$ as

$$Q_{\mathcal{R}^F}\left(\mathbf{z}_p^t, \mathbf{z}_q^t\right) = \begin{cases} 1, & \text{if } p \text{ is ahead of } q \\ \gamma\ (0 \leq \gamma < 1), & \text{otherwise} \end{cases} \tag{2.19}$$

$$Q_{\mathcal{R}^A}\left(\mathbf{z}_p^t, \mathbf{z}_q^t\right) = \begin{cases} 1, & \text{if } p \text{ and } q \text{ are on correct lanes} \\ \gamma\ (0 \leq \gamma < 1), & \text{otherwise} \end{cases} \tag{2.20}$$

If target pair (k_q, l_q) has relationship $\mathcal{R}^F$ at the previous time step, the prior probability $P\left(\mathcal{R}^F_{k_p l_q}\right)$ is calculated as

$$P\left(\mathcal{R}^F_{k_p l_q}\right) = \left[1 - P\left(\mathcal{C}_p^{l,r}\right)\right]\left[1 - P\left(\mathcal{C}_q^{l,r}\right)\right] + P\left(\mathcal{C}_p^{l}\right) P\left(\mathcal{C}_q^{l}\right) + P\left(\mathcal{C}_p^{r}\right) P\left(\mathcal{C}_q^{r}\right) \tag{2.21}$$

and

$$P\left(\mathcal{R}^A_{k_p l_q}\right) = \left[1 - P\left(\mathcal{C}_p^{l,r}\right)\right] P\left(\mathcal{C}_q^{l,r}\right) + P\left(\mathcal{C}_p^{l,r}\right)\left[1 - P\left(\mathcal{C}_q^{l,r}\right)\right] + P\left(\mathcal{C}_p^{l}\right) P\left(\mathcal{C}_q^{l}\right) + P\left(\mathcal{C}_p^{r}\right) P\left(\mathcal{C}_q^{r}\right) \tag{2.22}$$

where $P\left(\mathcal{C}^{l,r}\right) = P\left(\mathcal{C}^{l}\right) + P\left(\mathcal{C}^{r}\right)$.

Similarly, the relationship prior probabilities can be evaluated if target pair (k_q, l_q) has relationship $\mathcal{R}^A$.

Constructing Vehicle Relationships for MRF models

Based on the Markovian assumption in Section 2.4.2, a vehicle's states and its measurement associations are only affected by its neighboring vehicles. Therefore, a vehicle's relationship can be constructed and updated to calculate association likelihood based on its physical positional relationship with neighboring vehicles.

Assuming that the tracking scenarios are in a two dimensional space, the 2D Delaunay triangulation [57] is first used to construct the positional neighborhoods. Then, a lane width is used to identify which lane is used by a vehicle. To build the

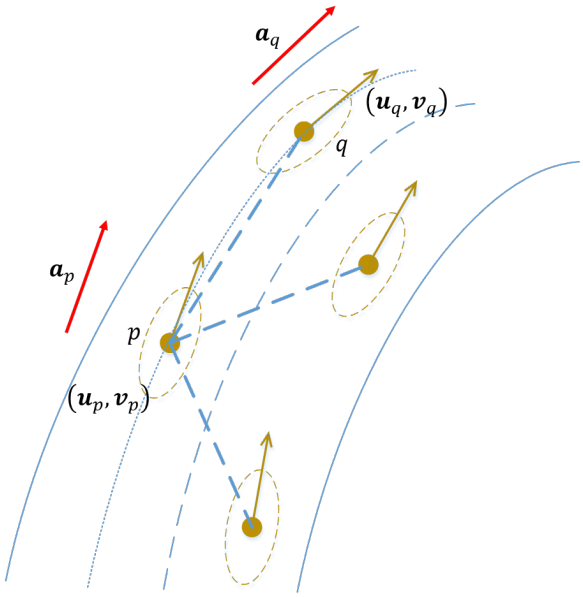


Figure 2.2: Leading-following relationship formulation.

leading-following relationship, a penalty function is defined as

$$\Phi_{p,q}\left(\mathbf{x}_p, \mathbf{x}_q\right) = \begin{cases} \cos\left[\left(\cos^{-1}\dfrac{\left(\mathbf{u}_p - \mathbf{u}_q\right)\cdot\left(\mathbf{u}_p - \mathbf{u}_q\right)}{\|\mathbf{u}_p - \mathbf{u}_q\|^2}\right) - \left(\cos^{-1}\mathbf{a}_p^r \cdot \mathbf{a}_p^r - \cos^{-1}\mathbf{a}_q^r \cdot \mathbf{a}_q^r\right)\right] \cdot \\ \quad \exp\left(-\beta\|\mathbf{u}_p - \mathbf{u}_q\|\right), \qquad \text{if } \dfrac{\mathbf{v}_p \cdot \mathbf{v}_q}{\|\mathbf{v}_p\|\|\mathbf{v}_q\|} > 0 \\ -\infty, \qquad \text{if } \dfrac{\mathbf{v}_p \cdot \mathbf{v}_q}{\|\mathbf{v}_p\|\|\mathbf{v}_q\|} \leq 0 \end{cases} \tag{2.23}$$

based on the angle between two vehicles and their inter-distance as illustrated in Figure 2.2, with $\mathbf{u}_*$ and $\mathbf{v}_*$ being the location and velocity, respectively, $\mathbf{a}_*$ being the unit tangent vector of the road obtained from the road network data, and β being the weight parameter. This classification function will penalize edges with large inter-angles and long inter-distances. Other penalty functions could be used as well. If $\Phi_{p,q}$ is greater than a certain threshold $\mathcal{T}_{\text{th}}$, targets (p, q) will be a lead-following pair.

If $\frac{\mathbf{v}_p \cdot (\mathbf{u}_p - \mathbf{u}_q)}{\|\mathbf{v}_p\| \|\mathbf{u}_p - \mathbf{u}_q\|} > 0$, target p will be the leading vehicle and q will be the following vehicle.

After partitioning the states and associations, a complete MRF model is built. The traffic models can be used for state and measurement prediction, given the inter-vehicle relationship, and the association likelihood is calculated based on this. Also, this relationship is updated and maintained at each step after state estimation.

2.6 State Estimation and Posterior CRLB

2.6.1 State Estimation for State-Dependent Targets

Since the target states are dependent and the state equations are nonlinear, the target states are updated via an EKF in a coupled manner [34]. The state equation (2.1) is linearized as

$$\mathbf{X}^t = \mathbf{F}' \mathbf{X}^{t-1} + \mathbf{W}^t \tag{2.24}$$

where

$$\mathbf{F}' = \begin{bmatrix} \frac{\partial F_1}{\partial \mathbf{x}_1} & \frac{\partial F_1}{\partial \mathbf{x}_2} & \cdots & \frac{\partial F_1}{\partial \mathbf{x}_n} \\ \frac{\partial F_2}{\partial \mathbf{x}_1} & \frac{\partial F_2}{\partial \mathbf{x}_2} & \cdots & \frac{\partial F_2}{\partial \mathbf{x}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial F_n}{\partial \mathbf{x}_1} & \frac{\partial F_n}{\partial \mathbf{x}_2} & \cdots & \frac{\partial F_n}{\partial \mathbf{x}_n} \end{bmatrix} \tag{2.25}$$

and $\mathbf{W}^T = \text{diag}\,(w_1^t, \cdots, w_n^t)$. The covariance matrix is initialized as

$$\mathbf{P}^0 = \begin{bmatrix} P_{11}^0 & P_{12}^0 & \cdots & P_{1n}^0 \\ P_{21}^0 & P_{22}^0 & \cdots & P_{2n}^0 \\ \vdots & \vdots & \ddots & \vdots \\ P_{n1}^0 & P_{n2}^0 & \cdots & P_{nn}^0 \end{bmatrix} \tag{2.26}$$

and the inter-target covariance matrix $P_{ij,i\neq j}^0$ is set to zero if two targets are independent according to the Markov property.

Together with data association, the steps for updating vehicle states using JMRF-PDA are as follows:

Step 1 The state is predicted based on driving behavior models.

$$\tilde{\mathbf{x}}_i^t = F_i\left(\hat{\mathbf{x}}_i^{t-1}; \hat{\mathbf{x}}_{i1}^{t-1}, \ldots, \hat{\mathbf{x}}_{i\kappa}^{t-1}\right), \quad i = 1, \ldots, n \tag{2.27}$$

$$\tilde{\mathbf{P}}^t = \mathbf{F}'\hat{\mathbf{P}}^{t-1}\mathbf{F}'^T + \mathbf{W}\mathbf{W}^T. \tag{2.28}$$

Step 2 Gating is done as in the standard JPDA.

Step 3 Evaluation of the association probabilities $P\left(\Theta^t|\mathbf{Z}_m^t\right)$ is carried out using the JMRF-PDA.

Step 4 Estimates and vehicle neighborhood relationships are updated.

- The state is updated as

$$\hat{\mathbf{X}}^t = \tilde{\mathbf{X}}^t + \mathbf{K}^t \sum_{\Theta^t} P\left(\Theta^t|\mathbf{Z}_m^t\right) \times \left[\mathbf{Z}^t(\Theta) - \hat{\mathbf{Z}}^t\right] \tag{2.29}$$

where $\mathbf{K}^t$ is the Kalman gain given by

$$\mathbf{K}^t = \tilde{\mathbf{P}}^t \mathbf{H} \left[\mathbf{S}^t\right]^{-1}. \tag{2.30}$$

- The convariance matrix is updated as

$$\begin{aligned}\hat{\mathbf{P}}^t =& \tilde{\mathbf{P}}^t - \sum_{\Theta^t:\Theta^t\neq\Theta_0} \beta(\Theta^t)\mathbf{K}^t\mathbf{S}^t\left(\mathbf{K}^t\right)^T \\ &+ \sum_{\Theta^t} \beta(\Theta^t)\mathbf{K}^t\left(\mathbf{Z}^t(\Theta^t) - \hat{\mathbf{Z}}^t\right)\left(\mathbf{Z}^t(\Theta^t) - \hat{\mathbf{Z}}^t\right)^T \cdot \left(\mathbf{K}^t\right)^T \\ &- \left[\sum_{\Theta^t} \beta(\Theta^t)\mathbf{K}^t\left(\mathbf{Z}^t(\Theta^t) - \hat{\mathbf{Z}}^t\right)\right] \cdot \\ &\left[\sum_{\Theta^t} \beta(\Theta^t)\mathbf{K}^t\left(\mathbf{Z}^t(\Theta^t) - \hat{\mathbf{Z}}^t\right)\right]^T\end{aligned} \tag{2.31}$$

  where Θ_0 represents the association events that $\delta_r(\Theta) = 0, \forall r \in \{1, \ldots, n\}$, which means all measurements are false alarms.

- The vehicle neighborhood relationship is reconstructed using the method in Section 2.5.3. If the neighborhood relationship is changed, then the motion models for these targets will be updated and the affected inter-target covariance matrix $P^t_{ij,i\neq j}$ will be re-initialized for new relationship and reset to zero for removed relationships.

2.6.2 Posterior Cramer-Rao Lower Bound

PCRLB for Multiple Target Tracking

In this section, the PCRLB is derived for dependent state estimation. Suppose $P(t)$ is the covariance of state estimate $\hat{X}(t)$, then the lower bound known as the PCRLB is given by [86] as

$$P(t) = E\left[\left(\hat{X}(t) - X(t)\right)\left(\hat{X}(t) - X(t)\right)'\right] \geq J(t)^{-1} \tag{2.32}$$

where $J(t)$ is the Fisher Information Matrix (FIM). For multiple target tracking, a recursive evaluation is provided by [85] as

$$J(t) = J_x(t) + J_z(t). \tag{2.33}$$

The first term, the posterior information matrix for target states, can be evaluated using

$$J_x(t+1) = D_t^{22} - D_t^{21}\left(J_x(t) + D_t^{11}\right)D_t^{12} \tag{2.34}$$

where

$$D_t^{11} = \mathbb{E}\left[-\Delta_{X_t}^{X_t}\ln p(X_{t+1}|X_t)\right] \tag{2.35}$$

$$D_t^{12} = \mathbb{E}\left[-\Delta_{X_t}^{X_{t+1}}\ln p(X_{t+1}|X_t)\right] \tag{2.36}$$

$$D_t^{21} = \left(D_t^{12}\right)' \tag{2.37}$$

$$D_t^{22} = \mathbb{E}\left[-\Delta_{X_{t+1}}^{X_{t+1}}\ln p(X_{t+1}|X_t)\right]. \tag{2.38}$$

The second term, the measurement contribution, is given by

$$J_z(t+1) = \mathbb{E}\left[-\Delta_{X_{t+1}}^{X_{t+1}} \ln p(Z_{t+1}|X_{t+1})\right]. \tag{2.39}$$

State Dependent Target Tracking

A target's state involves its neighboring targets' states, so the first term in (2.33) is calculated using the Jacobian matrix $\mathbf{F}'$ as

$$J_x(t+1) = \left[\mathbf{\Lambda}^t + \mathbf{F}' J_x(t)^{-1} \mathbf{F}'^T\right]^{-1} \tag{2.40}$$

where $\mathbf{\Lambda}^t = [\Lambda_1^t, \cdots, \Lambda_n^t]^T$, and Λ_i^t $(i = 1, \ldots, n)$ is the covariance of the additive noise ω_i^t in (2.1). The state contribution to the PCRLB depends on different target motion patterns and can be evaluated using their specific motion models and substituted into (2.40).

Since the proposed algorithm assumes the same measurement model as [85], the evaluation of the measurement contribution to the FIM remains unchanged.

Example: PCRLB for Leading-following Targets

Consider a scenario with only two vehicles on a single lane with one following the other. Their leading-following relationship can be modeled using the IDM [87]. The IDM models the acceleration of a vehicle as a continuous function based on its current velocity v_α, the gap s_α between to leading vehicle, and the velocity difference Δv_α with the leading vehicle as

$$\dot{v}_\alpha = a_{\max}\left[1 - \left(\frac{v_\alpha}{v_0}\right)^\delta\right] - a_{\max}\left(\frac{s^\star(v_\alpha, \Delta v_\alpha)}{s_\alpha}\right)^2. \tag{2.41}$$

The first term in (2.41) represents the acceleration that drives the vehicle to its desired speed, and the second term is the deceleration that maintains a desired gap $s^\star$ to its leading vehicle to avoid collision. The desired gap $s^\star$ is a function of velocity v_α and velocity difference Δv_α giving by

$$s^\star\left(v_\alpha, \Delta v_\alpha\right) = s_0 + s_1\sqrt{\frac{v_\alpha}{v_0}} + Tv + \frac{v_\alpha \Delta v_\alpha}{2\sqrt{a_{\max} b}}. \tag{2.42}$$

The remaining parameters and their typical values are given in Section 2.7.1.

Based on (2.41), the motion of two targets p and q, assuming that p is following q, is given by

$$x_p^{t+1} = F_p^t + \mathcal{R}(x_p^t, x_q^t) + w_p^t \tag{2.43}$$

$$x_q^{t+1} = F_q^t x_q^t + w_q^t \tag{2.44}$$

where the nearly constant velocity model [5] for target q is used for simplicity and the transition matrix F can be found in [5]. Note that the IDM assumes that the leading vehicle is not affected by the following vehicles. The motion of target p is assumed to be a mixture of independent motion F_p, where a nearly constant velocity model is applied, and the inter-target impact on acceleration. The inter-target impact on the acceleration of the following vehicle p is represented as $\mathcal{R}(x_p^t, x_q^t)$, which can be derived according to (2.41).

Therefore the stacked transition matrix in (2.40) can be written as

$$\mathbf{F}' = \begin{bmatrix} F_p^t + \dfrac{\partial R(x_p^t, x_q^t)}{\partial x_p^t} & \dfrac{\partial R(x_p^t, x_q^t)}{\partial x_q^t} \\ 0 & F_q^t \end{bmatrix}. \tag{2.45}$$

Table 2.1: The parameters of the IDM model

Parameter	Value
Desired velocity v_0 [m/s]	speed limit
Safe time headway T [s]	1.6
Maximum acceleration a [m/s^2]	1.4
Desired deceleration b [m/s^2]	2.0
Acceleration exponent δ	4
Jam distance s_0 [m]	2
Jam distance s_1 [m]	0

2.7 Experiments

2.7.1 Experiment Setup

In this section, the proposed JMRF-PDA tracker is implemented and compared with the standard JPDA tracker. The traffic models used in the JMRF-PDA are based on the IDM [87] and the freeway lane changing models [1].

The values of the IDM parameters [87] are listed in Table 2.1.

As for the lane change model, a model with fixed values of $P(\text{MLC}|v_n)$ and $P(\text{DLC}|v_n)$ is implemented. Also, $P(\text{acceptable gap}|v_n)$ is evaluated according to the freeway model [1] as

$$\begin{aligned} P(\text{acceptable gap}|v_n) = &\; P\left(G^{\text{lead}} \text{ acceptable and } G^{\text{lag}} \text{ acceptable } |v_n\right) \\ = &\; P\left(G^{\text{lead}} > G^{\text{lead}}_{\text{critical}} \text{ and } G^{\text{lag}} > G^{\text{lag}}_{\text{critical}} | v_n\right) \\ = &\; \Psi\left(\frac{\ln\left(G^{\text{lead}}\right) - \beta^{\text{lead}} X^{\text{lead}} - \alpha^{\text{lead}} v_n}{\sigma_{\epsilon,\text{lead}}}\right) \cdot \\ & \Psi\left(\frac{\ln\left(G^{\text{lag}}\right) - \beta^{\text{lag}} X^{\text{lag}} - \alpha^{\text{lag}} v_n}{\sigma_{\epsilon,\text{lag}}}\right) \end{aligned} \tag{2.46}$$

Table 2.2: The additional parameters in the lane-change model [1]

Parameter	Value
α^{lead}	-0.055
$\sigma_{\epsilon,\text{lead}}$	1.61
Constant $\beta^{\text{lead}} X^{\text{lead}}$	2.72
α^{lag}	0.190
$\sigma_{\epsilon,\text{lag}}$	1.31
Constant $\beta^{\text{lag}} X^{\text{lag}}$	-9.32
Normal distributed random term v_n	$N(0;1)$

where $G^{\text{lead, lag}}$ is the actual gap between the lead and lag vehicles, $G^{\text{lead, lag}}_{\text{critical}} = \beta^{\text{lead, lag}} X^{\text{lead, lag}} + v_n + \epsilon^{\text{lead, lag}}$ is the minimum acceptable gap for lane change, called the critical gap, and $\Psi(\cdot)$ is the cumulative distribution function of a normal random variable. The rest of the parameters that characterize the random variables as well as non-random in the lane-change model are assumed with default values in [1] (as listed in Table 2.2).

2.7.2 Simulation Results

As stated in Section 2.4.3, the JMRF-PDA tracker is an extension of the standard JPDA tracker. Thus, the parameters for the JMRF-PDA and the JPDA are the same in the simulations, with the additional parameters in the new algorithm being given in Section 2.7.1.

The simulations are carried out in a 2D Cartesian space. The state of the target is modeled as

$$\mathbf{x} = \begin{bmatrix} \xi & \dot{\xi} & \zeta & \dot{\zeta} & \ddot{\xi} & \ddot{\zeta} \end{bmatrix}^T \tag{2.47}$$

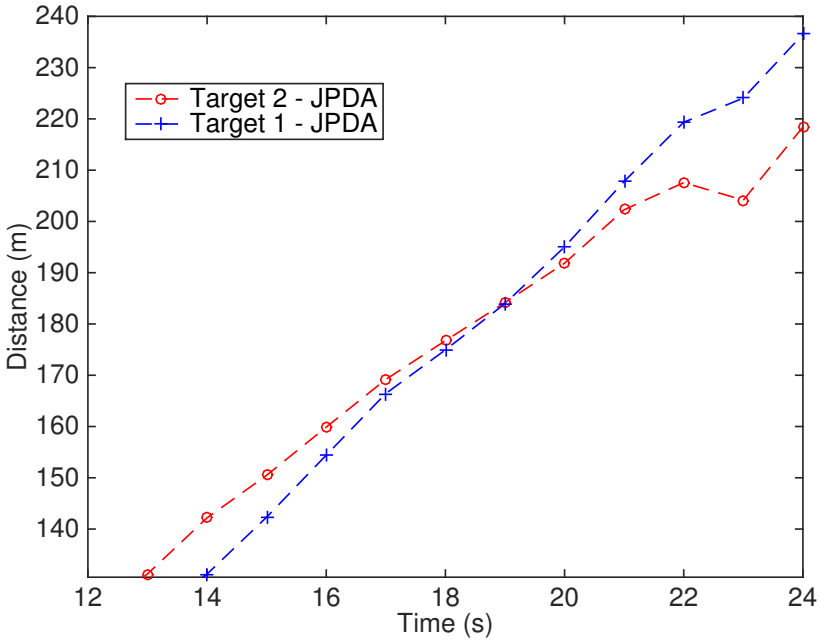
where (ξ, ζ) denotes the Cartesian coordinates, while $(\dot{\xi}, \dot{\zeta})$ and $(\ddot{\xi}, \ddot{\zeta})$ represent the velocity and acceleration components, respectively. The process noise is white Gaussian with covariance matrix given by

$$Q = 0.5 \times \begin{bmatrix} \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 & 0 & 0 & \frac{1}{2}\Delta t^2 & 0 \\ \frac{1}{2}\Delta t^3 & \Delta t^2 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & \frac{1}{4}\Delta t^4 & \frac{1}{2}\Delta t^3 & 0 & \frac{1}{2}\Delta t^2 \\ 0 & 0 & \frac{1}{2}\Delta t^3 & \Delta t^2 & 0 & \Delta t \\ \frac{1}{2}\Delta t^2 & \Delta t & 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{2}\Delta t^2 & \Delta t & 0 & 1 \end{bmatrix}. \tag{2.48}$$

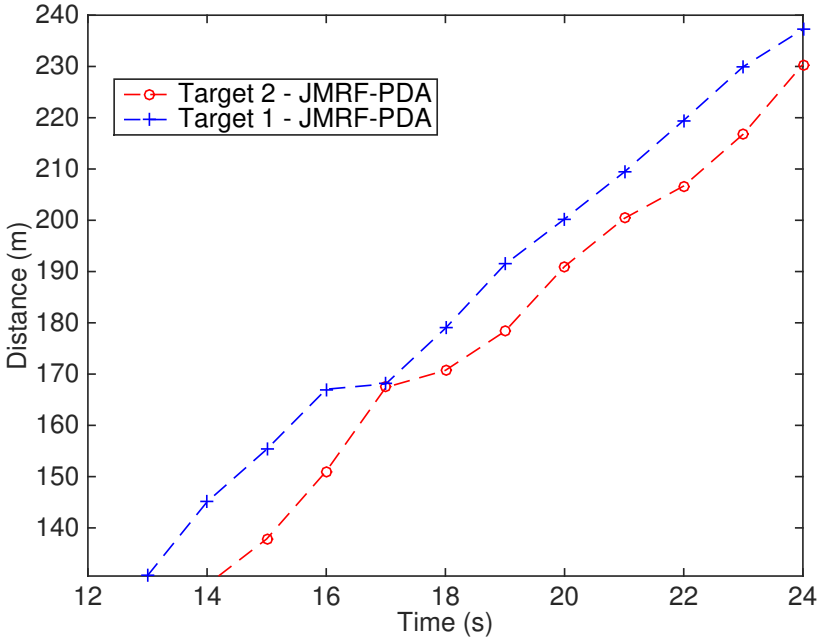
The measurement noise is zero-mean white Gaussian with standard deviation 2.5m in both X and Y axes. The number of false alarm measurements is Poisson distributed with mean $\lambda = 0.0003$. In the validation step, the gate to validate measurements is set using a 99.99% acceptance probability.

The JMRF-PDA tracker is same as the JPDA for single vehicle tracking and sparse non-interacting target tracking with no driving behavior based prediction and no edge potentials in the association step. To compare the performance of the proposed algorithm with that of the JPDA, the following scenarios for simulation are used:

1. Scenario A: *Vehicles on a single lane* — In this scenario, two targets moving at a nearly constant velocity in the same direction are simulated. Target 1 is the leading vehicle and target 2 is the following one, with initial gap 10m.



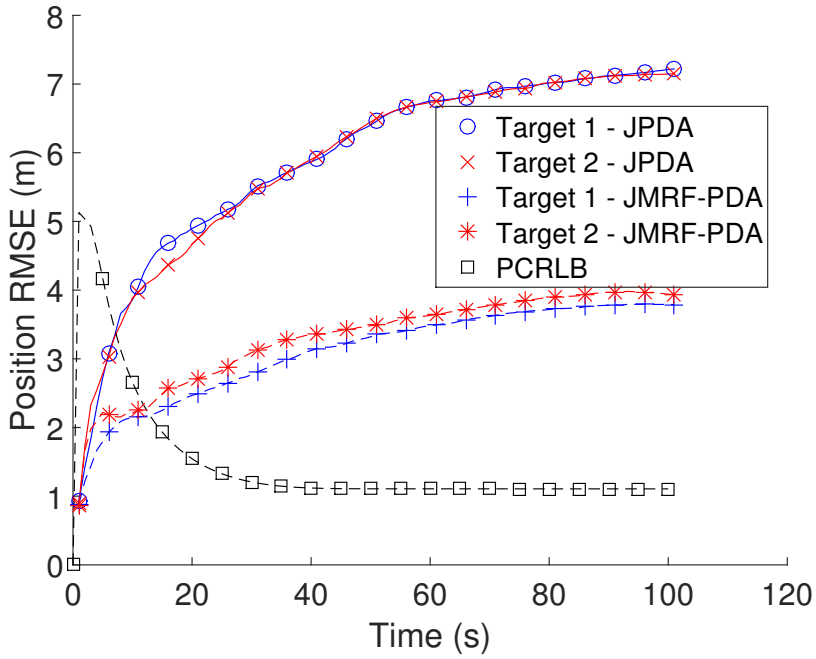
(a) JPDA



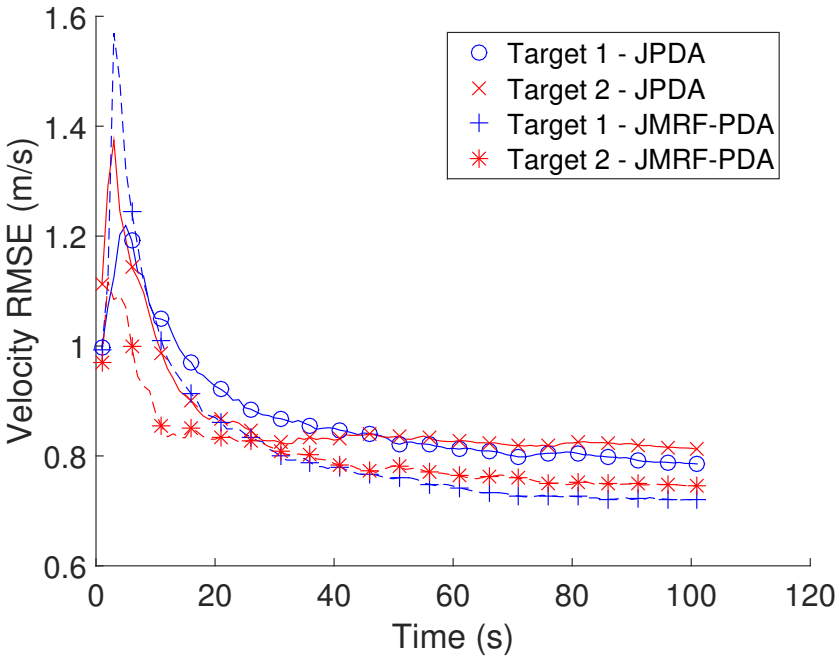
(b) JMRF-PDA

Figure 2.3: Sample tracking results in Scenario A. Target 2 is following target 1 on a single lane. The estimated distance moved from the origin along the road is shown. This estimated distance of target 2 should always be less than that of target 1.

Representative tracking results obtained using the JPDA and the JMRF-PDA tracker are presented in Figure 2.3. The Root Mean Squared Error (RMSE) of the estimates of the JMRF-PDA and that of the JPDA algorithms in 500 Monte Carlo runs over 100 scans are compared in Figure 2.4 and Figure 2.5, with P_d 1 and 0.9, respectively. The PCRLB of estimated position error is also shown in the figures. The JMRF-PDA can generate accurate tracks in the sense of avoiding track swaps in a single lane, and, as a result, the RMSE is less than that of the JPDA. Besides, the RMSE of the following target 2 is less than that of the leading vehicle 1, which confirms that the inter-target information can in fact reduce the uncertainty during association and estimation. The track swaps between target 1 and 2 are compared in Table 2.3, where the number of runs (out of 500 runs) with no track swaps over 100 scans and the percentage of track swaps are listed. This table shows that the JMRF-PDA performs better by reducing track swaps.

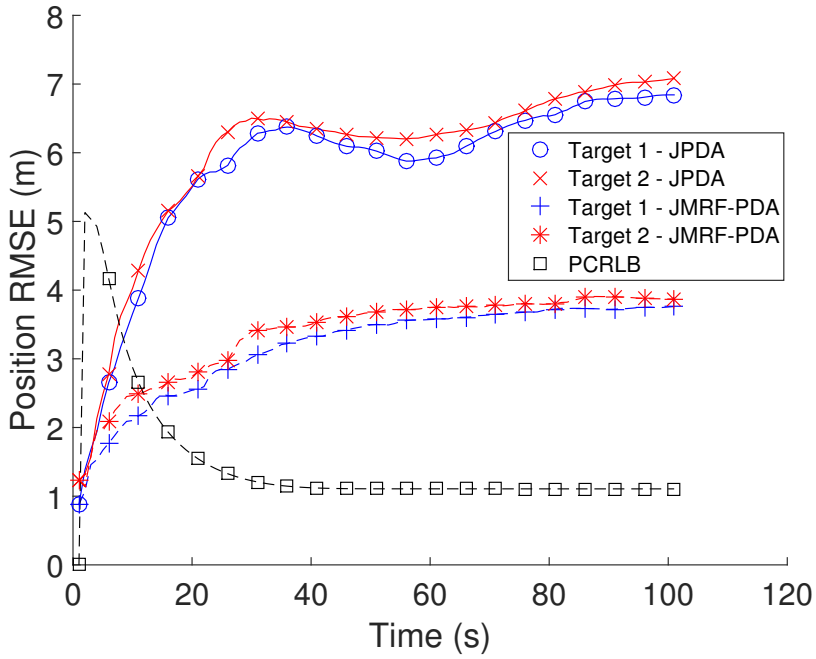


(a) Position RMSE (m)

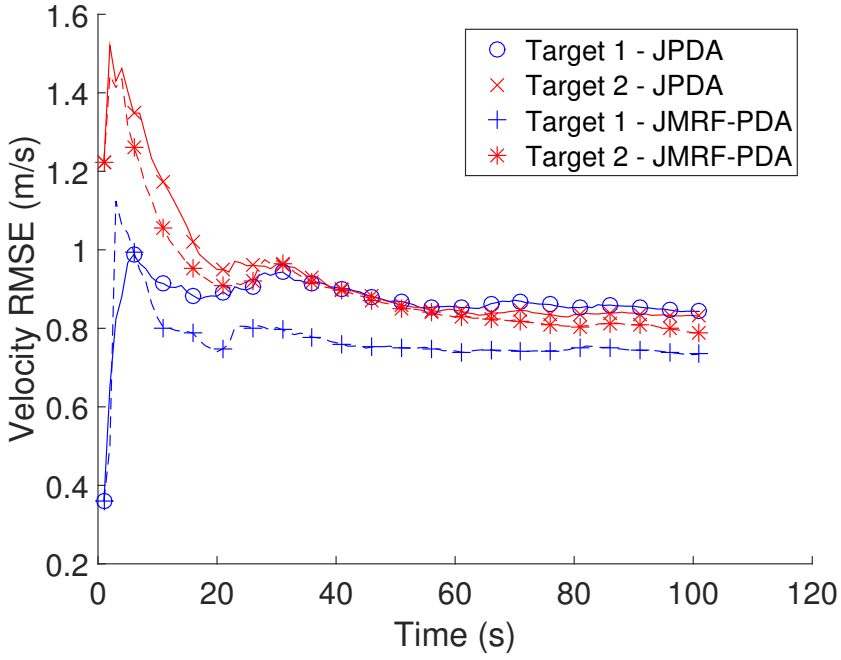


(b) Velocity RMSE (m/s)

Figure 2.4: Average position and velocity RMSE in Scenario A over 500 Monte Carlo runs ($P_d = 1$)



(a) Position RMSE (m)



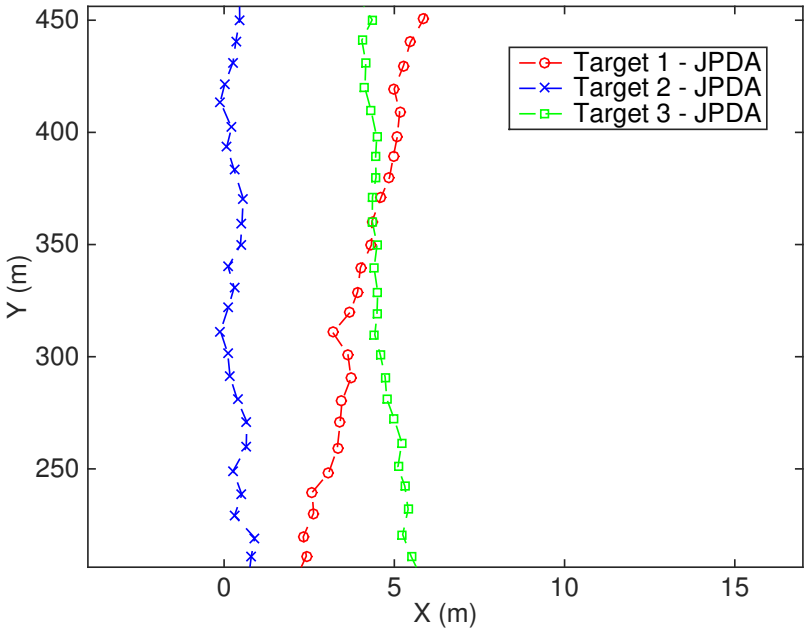
(b) Velocity RMSE (m/s)

Figure 2.5: Average position and velocity RMSE in Scenario A over 500 Monte Carlo runs ($P_d = 0.9$)

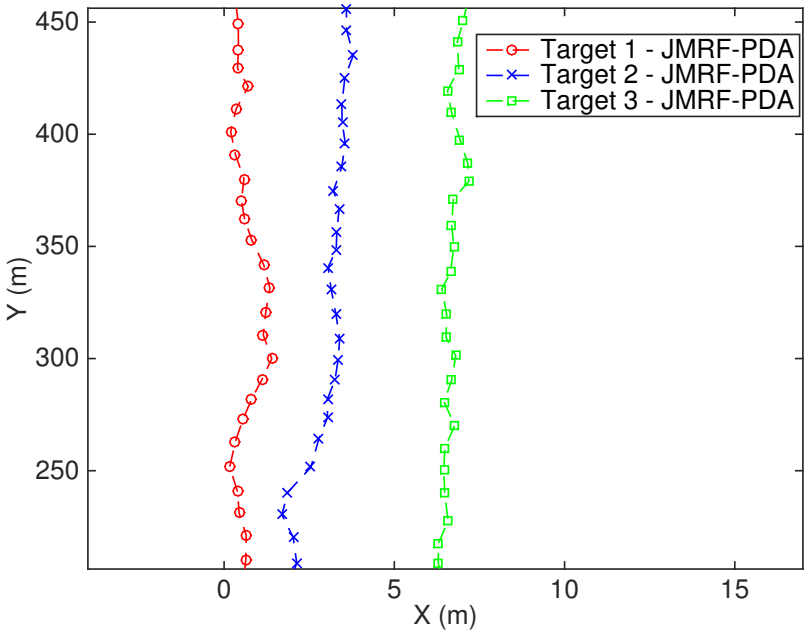
Table 2.3: Single lane tracking results over 500 Monte Carlo runs and 100 scans

P_d	Scenario	Runs with no track swaps (%)		Scans with track swaps (%)	
		JPDA	JMRF-PDA	JPDA	JMRF-PDA
1	Nearly constant velocity	40.0	57.6	26.9	12.0
0.9	Nearly constant velocity	34.0	52.0	32.6	12.6

2. Scenario B: *Parallel vehicles on multiple lanes* — In the multiple lane simulation, three vehicles are moving at a nearly constant velocity in the same direction with no lane changes. Target 1 is on the left lane, target 2 is on the center lane and target 3 is on the right lane.



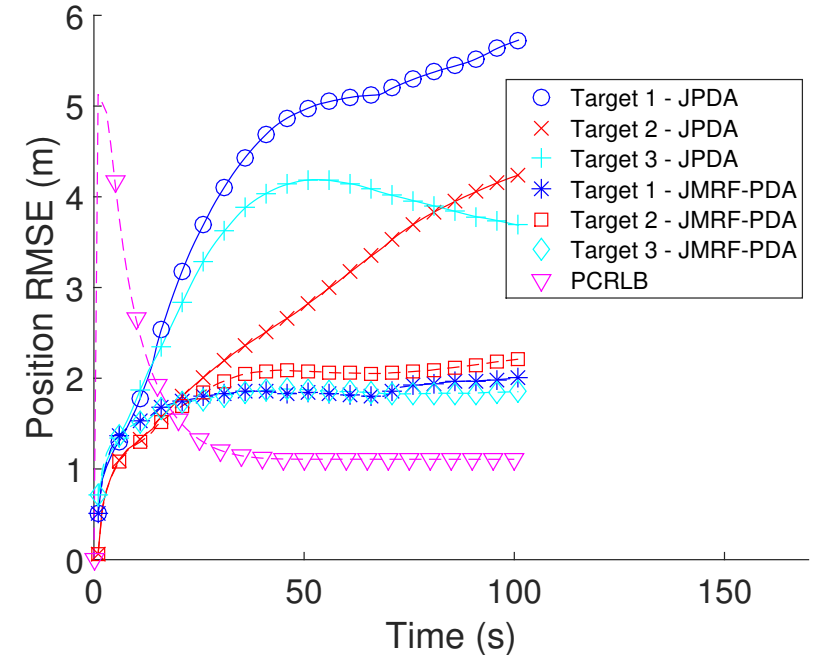
(a) JPDA



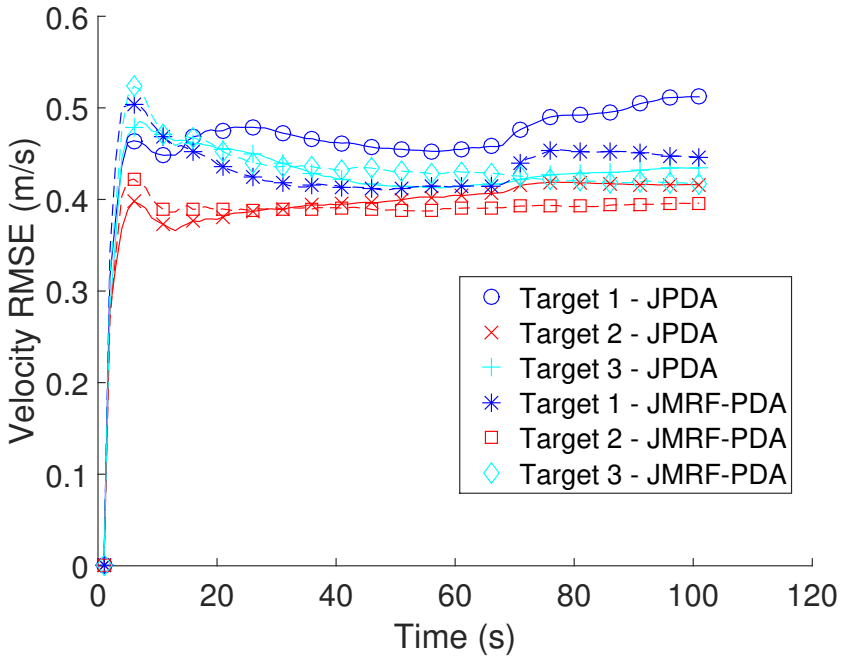
(b) JMRF-PDA

Figure 2.6: Sample tracking results in Scenario B. Three targets are on different lanes with no lane changes. Estimated track segments are shown.

Representative tracks obtained using the JPDA and the JMRF-PDA algorithms are shown in Figure 2.6. The RMSE of the estimates from the JMRF-PDA and the JPDA trackers over 500 Monte Carlo runs and 100 scans are compared in Figure 2.7. The PCRLB is also shown in the figure. The RMSE of the estimates corresponding to the left and right vehicles from the JPDA is higher than that of the center vehicle. This is due to the fact that the motion of the center vehicle is constrained by the vehicles on adjacent lanes, which is exploited by the proposed tracker to improve results. The position RMSE of the JMRF-PDA is lower for all three targets compared to that of the JPDA. That is, the proposed association approach reduces the interference from measurements generated by targets on neighboring lanes. The velocity RMSE of the JMRF-PDA and that of the JPDA are similar. This is because the velocity along the road direction is unrestricted for all targets with no leading vehicles. Table 2.4 compares the performances of the two algorithms in terms of lane swaps. Simulation results show that the JPDA can hardly maintain the track on the correct lanes while the JMRF-PDA reduces track swaps significantly.



(a) Position RMSE (m)



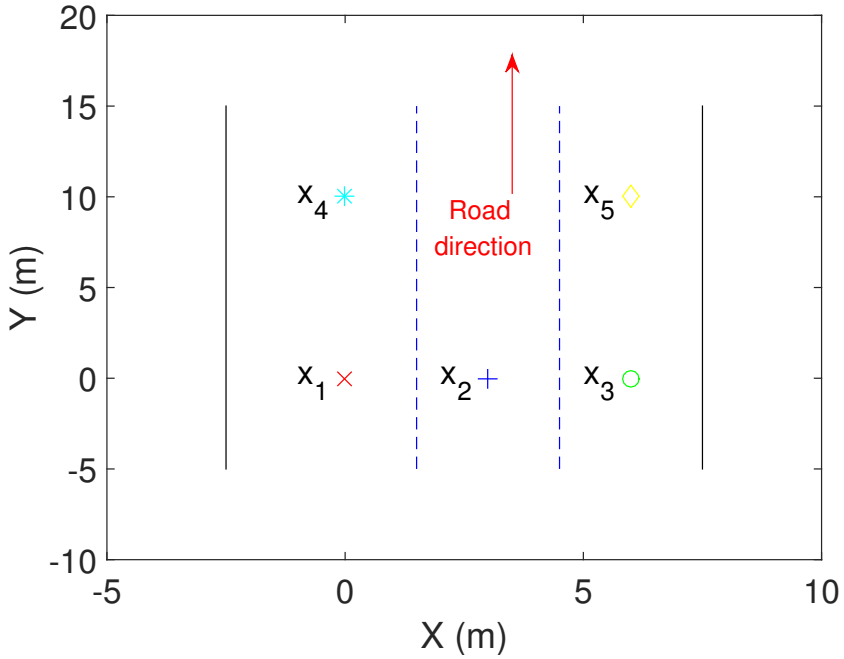
(b) Velocity RMSE (m/s)

Figure 2.7: Average position and velocity RMSE in Scenario B over 500 Monte Carlo runs ($P_d = 0.9$)

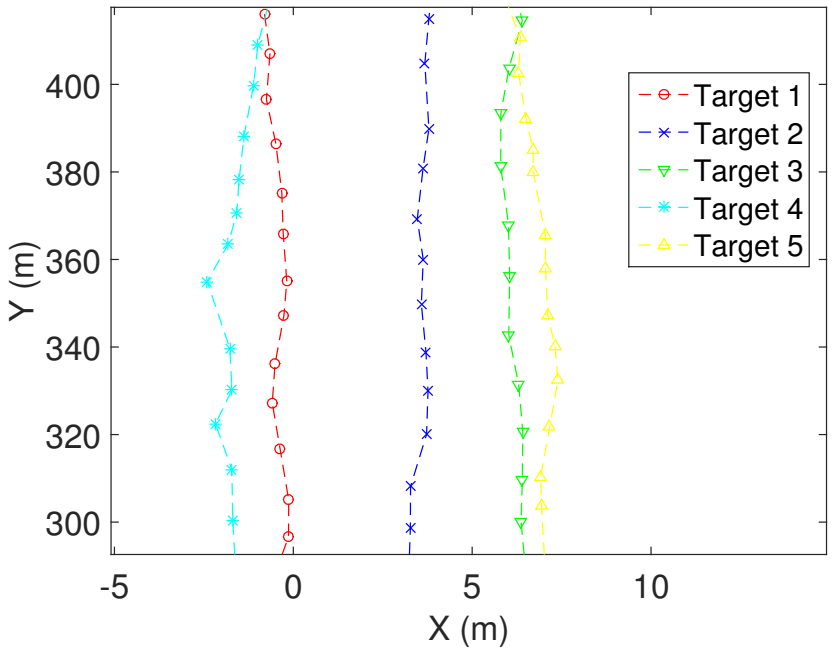
Table 2.4: Multiple lane tracking results in 500 Monte Carlo runs

P_d	Scenario	Runs with no lane swaps (%)		Scans with lane swaps (%)	
		JPDA	JMRF-PDA	JPDA	JMRF-PDA
1	Nearly constant velocity	2.0	46.2	90.6	45.7
0.9	Nearly constant velocity	0.0	44.0	92.1	40.8

3. Scenario C: *Group of vehicles on multiple lanes* — In this scenario, 5 targets are simulated and their positions are shown in Figure 2.8a, where targets on multiple lanes with leading-following situations are illustrated.

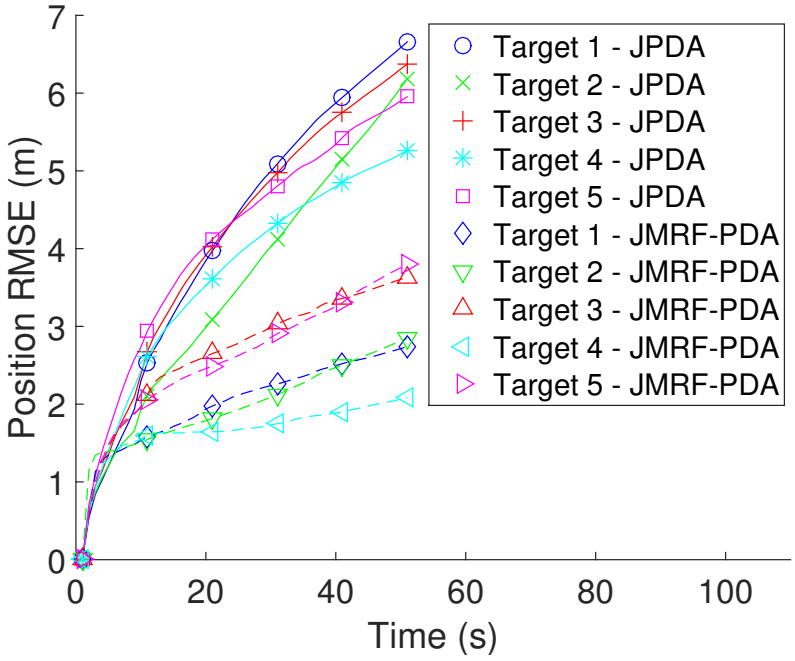


(a) Experimental setup in Scenario C.

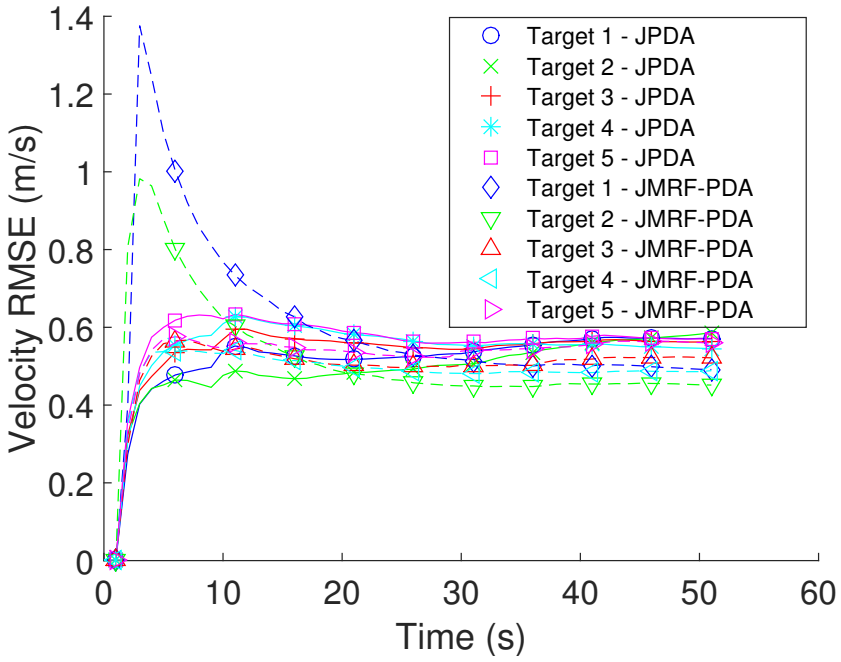


(b) Sample tracks from the JMRF-PDA

Figure 2.8: Simulation of Scenario C.



(a) Position RMSE (m)



(b) Velocity RMSE (m/s)

Figure 2.9: Average position and velocity RMSE in Scenario C over 500 Monte Carlo runs ($P_d = 0.9$)

Representative tracks obtained by the JMRF-PDA tracker are shown in Figure 2.8b while the position and velocity RMSE values are shown in Figure 2.9. Compared with scenario B, the RMSE of both position and velocity estimates from the two methods are higher due to the increased number of objects and the ensuing association uncertainty. The JMRF-PDA performs significantly better than the JPDA in terms of position RMSE while the velocity error is slightly lower than that of the JPDA.

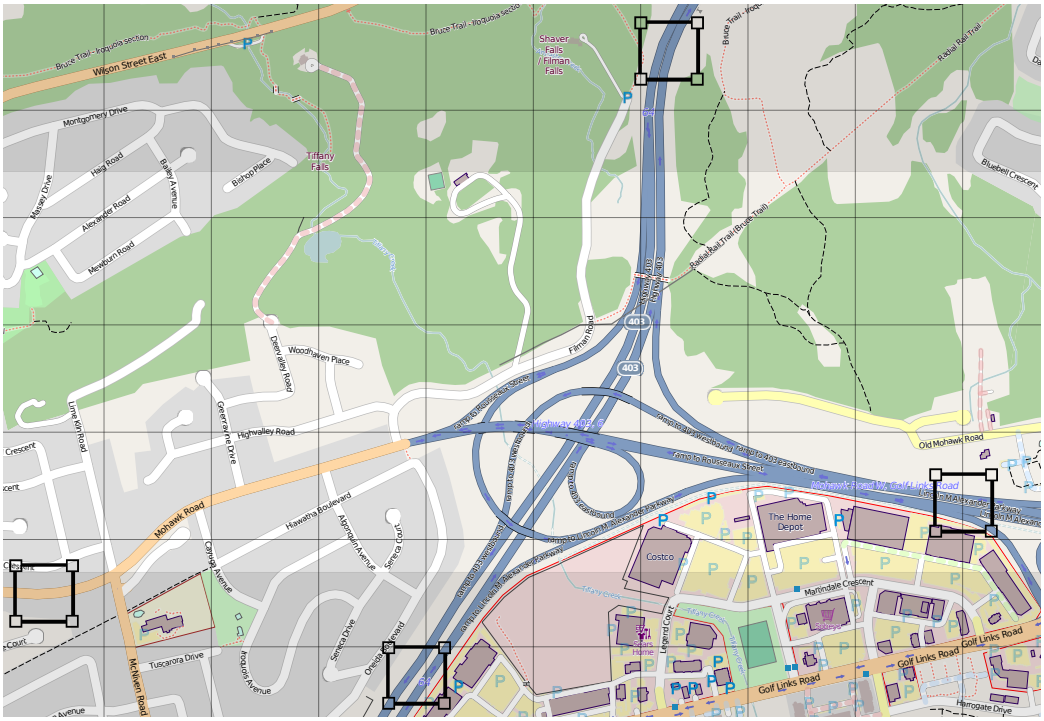
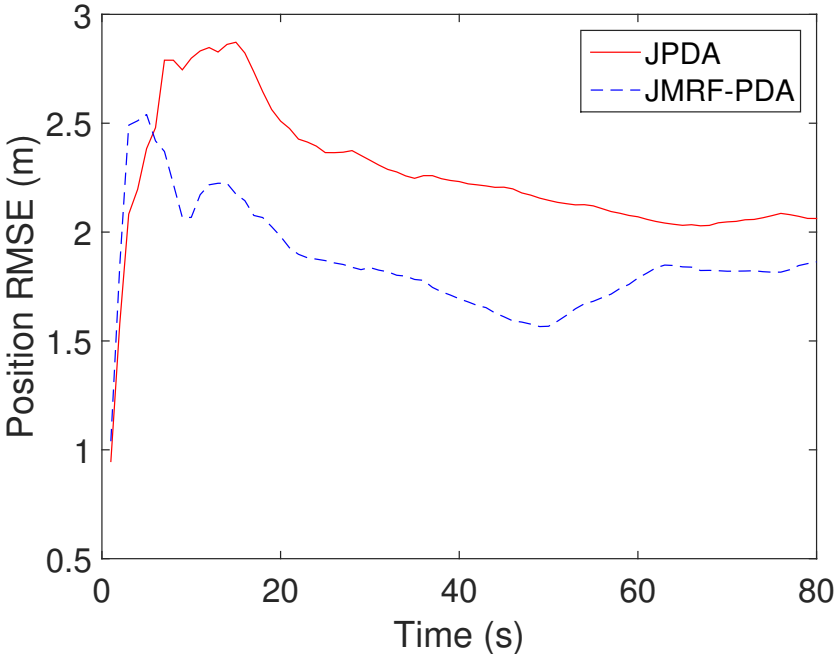


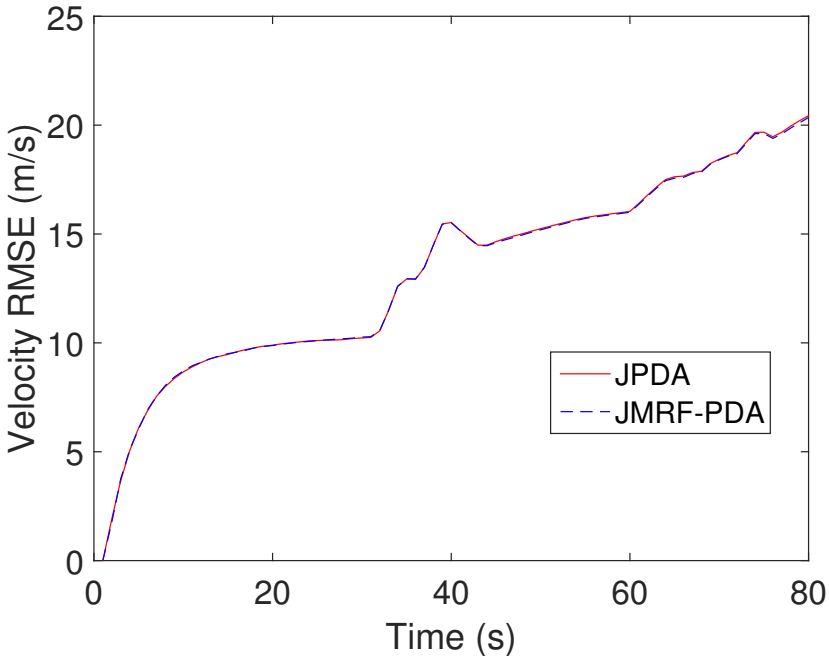
Figure 2.10: Ground target tracking simulation on a real map scenario (area: 2.3×1.8 km^2)

4. Scenario D: *Simulation on a real map* — To verify the performance of the proposed algorithm in real world problems, a combination of highway and

local traffic is simulated on a real map (see Figure 2.10) of Hamilton, Ontario, Canada. The traffic is generated using the Simulation of Urban MObility (SUMO) tool [52] with a map imported from OpenStreetMap [70]. The entry points into and exits out of the simulation area are located along the perimeter of the map marked by the black square, and the routes are generated from each point to a random exit. In this scenario, 48 vehicles are simulated, and the departure times of the vehicles are random and the leading-following order of the vehicles are also random. In SUMO, the vehicles on this map follow a nearly constant velocity model. The scenario lasts 90 scans, after which most of the vehicles are out of the area of interest.



(a) Position RMSE (m)



(b) Velocity RMSE (m/s)

Figure 2.11: Average position and velocity RMSE in Scenario D in 100 Monte Carlo runs ($P_d = 0.9$)

The RMSE of position and velocity estimates over 100 Monte Carlo runs are compared in Figure 2.11. It can be observed that the JMRF-PDA yields better performance in terms of position RMSE while the velocity RMSE are almost the same. The ground truth generated by SUMO reveals that the leading and following vehicles on highways are not closely-spaced, which means that the driving behavior model has less impact on the JMRF-PDA. However, on local roads, where targets are closely-spaced, the impact is significant.

The overall CPU times of the JMRF-PDA and the JPDA are almost the same. In this scenario, without any optimization, the Matlab code runs at 0.1354s and 0.1223s per scan for the JMRF-PDA and the JPDA, respectively, on an Intel 2.7 GHz i7 processor. That is, the JMRF-PDA requires 10% more CPU time.

2.8 Conclusion

In this paper a new technique called the Joint MRF-PDA algorithm was proposed to track closely-spaced ground targets with strongly correlated motion patterns. Without the standard mutual independence assumption about targets' state, as in the case of most existing approaches, the Markov Random Field based PDA filter with integrated driving behavior models was developed. The MRF graphical model was constructed by mapping individual association probabilities into nodes inside a graphical joint association probability. By integrating driving behavior models, including a lane change model and a lead-following model, into the MRF representation, the joint association probabilities are evaluated within the Probabilistic Data Association framework. Other traffic models can be integrated into the proposed framework as well. The driving behavior model was also applied as a novel motion model to describe

dependent target states. Both the MRF decomposition and the integrated driving behavior models improved predictions and associations based on a target's neighbors, which significantly improves track purity. Monte Carlo simulations showed that the proposed Joint MRF-PDA algorithm reduces track swaps caused by measurement-origin ambiguity and the mutual independence assumption in the presence of closely-spaced targets. With modeling target state dependencies using the MRF and driving behavior models, the proposed algorithm is suitable for realistic ground target tracking problems with minimal extra computational burden.

Acknowledgements

We thank Toyota Motor Engineering & Manufacturing North America (TEMA) and, in particular, Yasuo Uehara at TEMA, Ann Arbor, Michigan for their technical and financial support for this work.

Chapter 3

Mix-state Proposal based Particle Filter with Ridge Regression for High Speed Real-time Visual Target Tracking

3.1 Introduction

Object tracking is one of the central problems in computer vision research. The task of single target visual object tracking is to sequentially estimate the states of the target in the video sequences given only the object's initial state in the first frame. It has been an essential part of a variety of vision based applications, including video surveillance, human computer interaction, robot path planning and navigation, and crowd analysis. A robust and fast tracking algorithm can significantly improve the

overall performance and quality of these applications.

Visual object tracking has been actively studied for decades, however, many problems still remain unsolved due to many complex conditions and challenges. Those challenging situations can be categorized into mainly two aspects:

1. **Environmental condition changes.** Common environmental condition changes include illumination change, shadow, occlusions, camera shaking and abrupt motion, where these changes may trigger the camera's auto-settings too, and subsequently affect the captured image content.

2. **Object's self appearance changes.** The tracking algorithms have to deal with object's self changes too, such as rotation, scale changes, deformation, switching backside to the front side. Some pairs of examples[1] reflecting these changes are shown in Figure 3.1.

[1]Images and ground truth are from OTB−100 datasets [95]

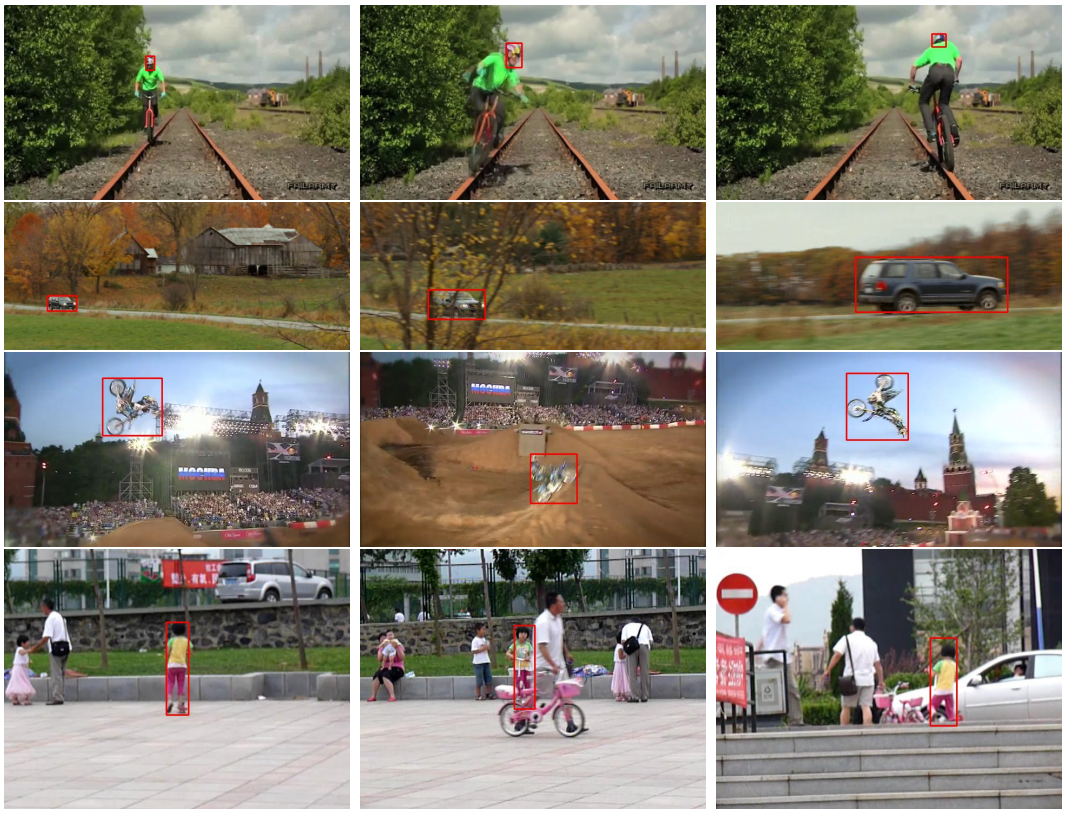


Figure 3.1: Examples of object's self changes (better view in color). Each rows of the images are from the same target (denoted with red rectangle) captured over the time.

If only single change is presented and known in advance, the algorithms could be specially designed to handle specific topics. However, in a real video data, one sequence of video may expose many different combinations of those changes, necessitating tracking algorithms to handle multiple challenges simultaneously.

3.2 Related Work

Visual target tracking can be addressed in different aspects. This section briefly reviews and summarizes the common issues and solutions in existing algorithms designed for real-time single object video tracking in literatures, specialties of each algorithm is beyond the scope of this thesis.

Generally visual tracking algorithms sequentially localize the target in the images. To localize the object, the observation likelihood needs to be calculated. Finding and improving the observation likelihood calculation is a main active research area that many new algorithms have been proposed in recent years. The observation likelihood calculation involves both object representation by appearance features, object comparison based on its appearance feature, and updating of the observation model during the tracking process according to the object's visual appearance feature change. Object representation describes how the target state is modeled, and based on which, the features used to characterize the object is chosen. A bounding box or surrounding rectangle that encloses the target is most widely used for object representation due to its simplicity and suitability for direct feature calculation. Other than bounding boxes, tilted bounding boxes, quadrilaterals, ellipsoids, and closed curves have also been studied for representing objects for tracking. Besides the holistic representation, part models and keypoint sets are also broadly used to describe more detailed shape information about the object.

Searching object in image is through potential candidate comparison. One way is direct object comparison. Normalized Cross Correlation (NCC) is a fast and efficient approach to implement the template based matching. However, direct comparison also suffers drawbacks of sensitivity to object appearance change, including changes

by object itself, such as object scale change and deformation, as well as influence of its environment and sensor, such as varying illumination and shadow, partial occlusion, and motion blur. Various feature resisting or partially resisting to these changes are proposed. Color intensity histograms, Local Binary Patterns (LBPs) [69], and Histogram of Oriented Gradients (HOG) [20] [25] are three typical and commonly used features, which represent color, local texture, and structural information respectively.

Object representation affects the choice of appearance feature. More important factor to consider while choosing features is how effectively the feature can be used to distinguish the object and the background, and invariant to certain self appearance change. Single feature or more effective combination of multiple features are widely used in the tracking research.

To achieve more discrimination between object and the background, learning based algorithms are introduced. The major family of the learning algorithms is supervised learning. The supervised learning based algorithms try to learn the parameter of a mapping from given features to either a binary value (for example $0, 1$ or $1, -1$) representing target or non-target, or continuous values representing how likely the features belong to the object. Different learning algorithms such as linear regression based [40], SVM based [36], random forest/ferns based [46], are commonly used.

Another aspect of the tracking problems is how to search or localize the object in the image using the appearance model with associated comparison algorithm. The localization part is a significant step impacting the computational load as well as the precision. As most object feature extraction and comparison has fixed number of calculations, the amount of search operations determines computational time, and

only through proper searching region can result in correct tracking results. Typically, the localization methods can be categorized in three families [83], the gradient descent based methods, exhaustively brutal force searching methods, and stochastic searching methods. All these three methods balance simplicity, computational load and accuracy.

Overall, the contribution of this chapter is a novel single target tracking algorithm that runs at approximately 400 frames per second (FPS) with better or comparable performance to recently published existing real-time trackers compared on a publicly available benchmark. Aiming high speed that the algorithm is started with, the proposed algorithm replaced the exhaustive search method employed widely in the common tracking framework with the particle filter. To further effectively use the limited number of particles, a mixed-motion proposal distribution is adopted to better handle noise, abrupt motion and recovery from failure fragments. The proposed tracking algorithm also addresses the abrupt motion and scale change problems as many state-of-the-art trackers still employ only a fixed scale model despite the target's self scale change. A ridge regression based observation likelihood is used to fast learn the appearance model and update with the changes in it. Occlusion and partial out-of-scene is also handled using part matching.

3.3 Mixed-motion Proposal based Particle Filtering

3.3.1 Bayesian Recursive Filtering for Tracking

In a conventional single target tracking problem, the system is modeled as a discrete time Markov process with hidden states $\mathbf{x}_{1:t} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t\}$ and emitted observations $\mathbf{z}_{1:t} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t\}$ of the states measured by the sensor for time 1 to t. The discrete state evolution is characterized as

$$\mathbf{x}_t = \mathbf{f}_t(\mathbf{x}_{t-1}, \mathbf{v}_t) \tag{3.1}$$

where $\mathbf{f}_t$ is the transition function, and $\mathbf{v}_t$ is an independent and identically distributed (i.i.d) noise. The observations are given by

$$\mathbf{z}_t = \mathbf{h}_t(\mathbf{x}_t, \mathbf{w}_t) \tag{3.2}$$

where $\mathbf{h}_t$ is the sensor measurement function depicting how the target state is observed or measured mathematically, and $\mathbf{w}_t$ is another i.i.d noise.

The objective of single target tracking is to recursively estimate the state $\mathbf{x}$ from time 1 to t based on the received sequence of measurements $\mathbf{z}_{1:t}$ up to time t.

Given the following assumptions,

1. State prior is given by $p(\mathbf{x}_0)$

2. The probabilistic state transition and measurement observation of the system

are modeled as

$$\mathbf{x}_t \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{3.3}$$

$$\mathbf{z}_t \sim p(\mathbf{z}_t|\mathbf{x}_t) \tag{3.4}$$

3. A sequence of measurements received from the sensor $\mathbf{z}_{1:t} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t\}$

Bayesian recursive filter finds the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ of the hidden state at time t. By using the Bayes's rule, the posterior $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ can be computed as

$$\begin{aligned} p(\mathbf{x}_t|\mathbf{z}_{1:t}) &= \frac{p(\mathbf{z}_{1:t}|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{z}_{1:t})} \\ &= \frac{p(\mathbf{z}_t, \mathbf{z}_{1:t-1}|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{z}_t, \mathbf{z}_{1:t-1})} \\ &= \frac{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_t)p(\mathbf{z}_{1:t-1}|\mathbf{x}_t)p(\mathbf{x}_t)}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1})} \\ &= \frac{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_t)\frac{p(\mathbf{x}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1})}{p(\mathbf{x}_t)}p(\mathbf{x}_t)}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_{1:t-1})} \\ &= \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})}. \end{aligned} \tag{3.5}$$

In the above equation, the prior $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ represents the knowledge of the model. Assume from the last iteration at $t-1$, the posterior distribution $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ is known. The joint distribution of $\mathbf{x}_t, \mathbf{x}_{t-1}$ given $\mathbf{z}_{1:t-1}$ is

$$\begin{aligned} p(\mathbf{x}_t, \mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) &= p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) \\ &= p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}). \end{aligned} \tag{3.6}$$

Integrating over $\mathbf{x}_{t-1}$ leads to the Chapman-Kolmogorov equation

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}. \tag{3.7}$$

Calculating $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is often referred as the prediction step in the Bayesian recursive filtering.

The likelihood $p(\mathbf{z}_t|\mathbf{x}_t)$ depicts the noise corruption in the sensor. After receiving measurement $\mathbf{z}_t$, the posterior is then calculated by (3.5), which is often denoted as the update step in the Bayesian recursive filtering.

The denominator $p(\mathbf{z}_t|\mathbf{z}_{1:t-1})$ in (3.5) is a constant given by

$$p(\mathbf{z}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})d\mathbf{x}_t. \tag{3.8}$$

For a system with linear transition and observation model, and additive Gaussian noise, the above recursion could be computed exactly using a Kalman filter. And the state estimated by the Kalman filter is optimal in the sense of MMSE. However, for other models, the posterior may not be easily evaluated, sometimes computationally intractable.

3.3.2 Particle Filtering

Particle filtering method, also known as sequential Monte Carlo method, is an approximation approach to solve the exact solution of the Bayesian recursive estimation. A brief review of particle filtering is presented as follows:

Many integration problems, including the Bayesian Recursive Filtering, have the

form of computing expectation

$$E[f(x)] = \int f(x)dp(x) = \int f(x)p(x)dx \tag{3.9}$$

on a probability density $p(x)$. For a generic probability density $p(x)$, with N independent random samples (also called particles) $X_{1:N} = \{x^{(1)}, \ldots, x^{(N)}\}$ drawn from $p(x)$, the Monte Carlo approximation $\hat{p}(x)$ of $p(x)$ is

$$\hat{p}(x) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x^{(i)}) \approx p(x) \tag{3.10}$$

where $\delta(\cdot)$ is the Dirac delta function.

Sometimes, directly evaluating the integral in (3.9) can be computationally intractable. Using the Monte Carlo approximation, the integral in (3.9) can be numerically evaluated as

$$E[f(x)] = \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)}). \tag{3.11}$$

However, to achieve a reasonable good approximation, a big number of samples are needed, which will result in a huge computing burden in real applications. Especially in high dimension space where the data is sparse, samples are mostly wasted on low probability locations, that have negligible contribution to the final result. To increase the computational efficiency, importance sampling that targeting to generate samples as concentrating to the high weights location as possible was proposed. Importance sampling uses an additional known distribution $\pi(x)$ to sample from, instead of drawing samples directly from the true distribution $p(x)$ that is often difficult. With a finite set of N samples $X_{1:N} = \{x^{(1)}, \ldots, x^{(N)}\}$ drawn from importance density $\pi(x)$,

the approximation in (3.9) can be computed as

$$E[f(x)] = \sum_{i=1}^{N} W^i \ f(x^{(i)}) \tag{3.12}$$

$$W^i = \frac{w_i(x^{(i)})}{\sum_{j=1}^{N} w_j(x^{(j)})} \tag{3.13}$$

where $w_i(x^{(i)}) = \frac{p(x^{(i)})}{\pi(x^{(i)})}$ is the importance weight of the i-th sample, and W^i is the normalized weight.

Sequential Importance Sampling

Considering the target tracking problem, a set of N samples $\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^{N}$ could be used to approximate the posterior probabilistic density

$$p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(\mathbf{x}_{0:t} - \mathbf{x}_{0:k}^{i}). \tag{3.14}$$

If the samples are drawn from an importance density proposal $\pi(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$, then the importance weights $w_t^{(i)}$ are

$$w_t^{(i)} = \frac{p(\mathbf{x}_{0:t}^{(i)}|\mathbf{z}_{1:t})}{\pi(\mathbf{x}_{0:t}^{(i)}|\mathbf{z}_{1:t})} \tag{3.15}$$

By choosing proposal distribution $\pi(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ that satisfies

$$\pi(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = \pi(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})\pi(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}) \tag{3.16}$$

(3.15) can be expanded into

$$\begin{aligned} w_t^{(i)} &= \frac{p(\mathbf{x}_{0:t}^{(i)}|\mathbf{z}_{1:t})}{\pi(\mathbf{x}_{0:t}^{(i)}|\mathbf{z}_{1:t})} \\ &= \frac{1}{Z_t}\frac{p(\mathbf{z}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{\pi(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)},\mathbf{z}_{1:t})}\frac{p(\mathbf{x}_{0:t-1}^{(i)}|\mathbf{z}_{1:t-1})}{\pi(\mathbf{x}_{0:t-1}^{(i)}|\mathbf{z}_{1:t-1})} \\ &\approx \frac{p(\mathbf{z}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{\pi(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)},\mathbf{z}_{1:t})} \cdot w_{t-1}^{(i)} \end{aligned} \tag{3.17}$$

where $Z_t = p(\mathbf{z}_t|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{z}_{1:t-1})$ is a normalizing constant.

Choosing state transition prior as importance density proposal distribution, $\pi(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$, leads to the conventional bootstrap filter [2] or known as the CONDENSATION method [42] in the computer vision community, and then the weights could be calculated as

$$w_t^{(i)} \approx p(\mathbf{z}_t|\mathbf{x}_t^{(i)}) \cdot w_{t-1}^{(i)} \tag{3.18}$$

3.3.3 Mixed-state Proposals for Particle Filter

Traditional particle filters used in image tracking tends to adopt single motion model, leading to a single proposal distribution for $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. However, visual targets have complex motion patterns and occasionally abrupt motions, as well as various appearance change such as light condition change, deformation, and occlusion. To keep tracking the object, using only single motion model requires an increasing number of particles and increased variance of the processing noise distribution to cover the possible location of the target in the next frame. Therefore, a lot of the particles are wasted when target exposes only certain pattern of motions, and this results in

inefficiency in computation.

In maneuvering target tracking, a common strategy is to utilize multiple models to model the target motion, such as the Interacting Multiple Model (IMM) filter [5] that uses multiple Kalman filters with different motion model in parallel to estimate the target maneuver. Mixed motion models are also introduced to particle filtering [15] [98] [53] [43].

Let $\{\kappa_m(\mathbf{x})\}_{m=1}^{M}$ represents a set of M different proposal distributions. A mixed-motion proposal distribution could be denoted as

$$\begin{aligned} \Lambda(\mathbf{x}) &= \sum_{m=1}^{M} \alpha_m \kappa_m(\mathbf{x}) \\ \text{with } \sum_{m=1}^{M} \alpha_m &= 1, \quad \alpha_m \geq 0 \end{aligned} \tag{3.19}$$

where α_m is the m-th proposal's weight, and the weights sum to 1.

Specifically for the tracking problem, by using a set of N samples drawn from $\Lambda(\mathbf{x})$, with n_p samples in each proposal for $p = 1, \ldots, M$, the posterior in (3.14) can be approximated by

$$p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = \sum_{p=1}^{M} \sum_{m=1}^{n_p} \alpha_m^p \kappa_m^p(\mathbf{x}_m^p) \delta(\mathbf{x} - \mathbf{x}_m^p) \tag{3.20}$$

Compared with other proposal distributions merely based on state transition density or observation density, a mixed-motion proposal distribution integrates and incorporates multiple sources of information including both multiple motion model as well as observation information into the proposal distribution. The weights in the mixed-motion proposals determine the number of particles used in each individual proposal.

The weights could be either fixed to empirical values or dynamically adaptive called *balance heuristic* setup [90] by

$$\alpha_m = \frac{n_p \kappa_m(\mathbf{x})}{\sum_{m=1}^{M} n_p \kappa_m(\mathbf{x})}. \tag{3.21}$$

This setup of weights could not be the worst any other weights settings.

To handle the motion and appearance changes of the targets, several proposal distribution based on different motion model is used. In the rest of this section, the dynamic and measurement model is described and subsequently the proposal distributions are presented.

Dynamic Model and Measurement Model

The state of the target $\mathbf{x}$ is defined as

$$\mathbf{x} = \begin{bmatrix} u_c & v_c & w & h & \dot{u}_c & \dot{v}_c & \dot{w} & \dot{h} \end{bmatrix}^T \tag{3.22}$$

where (u_c, v_c) is the 2D center position of the target, w, h is target's width and height respectively, and $\dot{u}_c, \dot{v}_c, \dot{w}, \dot{h}$ is the velocity. And the observation from the image sensor is defined as

$$\mathbf{z} = \begin{bmatrix} u_c & v_c & w & h \end{bmatrix}^T \tag{3.23}$$

The dynamic model of the system is

$$\mathbf{x}_t = F_t(\mathbf{x}_{t-1}) + w_t \tag{3.24}$$

where F_t is the state transition function, and w_t is Gaussian additive noise.

The observation model of the system is

$$\mathbf{z}_t = H\mathbf{x}_t + v_t \tag{3.25}$$

with measurement matrix

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{3.26}$$

and v_t as Gaussian noise.

Constant velocity model. The constant velocity model is one of the most widely used simple motion models for tracking objects. It assumes that the target moves at a constant velocity. In most of the image tracking scenes, the velocity is not observable and is often recovered by using filters such as Kalman filter. The state transition

matrix for the above state models is

$$F = \begin{bmatrix} 1 & 0 & 0 & 0 & T & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & T & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & T & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & T \\ 0 & 0 & 0 & 0 & T & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & T & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & T \end{bmatrix} \tag{3.27}$$

where T is the sampling period.

Stop model. The stop model refers to when the target is stopped at some location. The velocity of the target is set to zero, and the stop model is often used to keep the estimate of the target from previous frame.

Re-birth model. Due to fully or partial occlusions, or target out-of-view reasons, a re-birth model is necessary to cover the possible location where the target might show up again. Potential location could be the last known position, also the entrance and exit location of the scene could also be employed.

3.3.4 Bounded Adaptive Mixed-motion Proposals

Observing that 1) when the confidence of the tracker is high, most of the particles are wasted. 2) When the tracker confidence is low, that usually caused by occlusion on the target or target out of the scenario or significant appearance change on the target,

the particles needs to cover the possible location to rediscover the target. To more efficiently allocate particles for each motion mode, instead of choosing a fixed number of particles to represent a motion component, the number can be adaptive according to the weights of each mode in the proposed mixed-motion proposal distribution.

However, similar to particle filtering, this adaptive mode weight could lead to mode impoverishment that only one mode has significant larger weights than the rest of the weights. Once this happens, the tracker would hardly recover from disruptions caused by noised or occluded observations. To alleviate this problem, a bounded adaptive mixed-motion proposal strategy is introduced. Each motion mode has a predefined weight lower bound. Once the weight of the mode drops below the bound, a cut-off would be used to prevent disappearance of that mode. The bounded weight will ensure all motion mode remains active throughout the tracking process, and helps to recover from noise corruption.

3.4 Observation Likelihood based on Ridge Regression

3.4.1 Image Features and Feature Representation

To calculate the proposal likelihood $f(\mathbf{z}_t^i|\hat{\mathbf{z}}_t)$, multiple image features could be employed. These features include human hand-crafted and machine learned.

1. Hand-crafted features. Hand-crafted features cover different aspects of the characteristics of the target appearance. For example, the color histogram contains statistics of the object color distribution. The Histogram of Oriented Gradients

(HOG) describes the structural distribution of the edge/contour information. The local binary patterns (LBP) represents the local texture information.

2. Machine learned features. Deep convolutional networks boost the performance of many existing machine learning research fields, such as natural language processing, computer vision, and speech recognition. Convolutional networks are mainly constructed based on convolution operations. The convolution results from different level of layers extracts a hierarchical level of features of the input images. These learned features could also be applied into calculate the proposal likelihood.

As the CNN based feature extraction consumes significant amount of computations and may not meet the real-time processing goal without a descent GPU to speedup [44], traditional hand-crafted features are chosen for feature extraction.

To describe different characteristics of the input image, a common approach is to combine different features. The combination can be done by concatenating features together or using weighted combination after feature comparison.

Image Resizing

An important step in the aforementioned feature extraction steps is image resizing. Since in the particle filtering steps, random sampled particles have different sizes, extracting features that have structure information, like HOG or raw image as feature, will result in different feature vector sizes. These features with different dimensions would cause difficulties for comparison or calculating likelihood, because most of the distance metric requires same input size of two candidates.

One important technique to solve this problem is resizing the particle image into a

fixed size, and then extract features based on the resized image, that eventually leads to same fixed size feature dimensions. In order to resize an image, a target size should be chosen. For tracking purposes, either a fixed size or the initial size of the target at first frame could be adopted. The former option is finally chosen as the resizing parameter. The reason for choosing a fixed size is based on three aspects. Firstly, fixed resizing size avoids extra padding steps for some feature extraction methods since target size does not always fit the size required by some features such as HOG feature. Secondly, fixed resizing size that is often chosen to be smaller than target size that reduces computational load. Thirdly, by using a smaller template size, some information will be discarded, and will result in robustness to some appearance changes, as too detailed features would be very sensitive to appearance changes.

Since targets to be tracked may have different sizes and mismatching of width/height ratio would reduce misleading spatial resolution in the extracted feature, an adaptive template size is proposed. For a given template size (w, h), based on the ratio of width and height, the resizing size $\mathcal{S}$ is chosen from a set of empirical ratios $\{0.5, 1.0, 2.0\}$ based on the following thresholds:

$$\mathcal{S} = \begin{cases} (w/2, h) \text{ if } r \leq 0.75 \\ (w, h/2) \text{ if } r \geq 1.5 \\ (w, h) \text{ otherwise} \end{cases} \tag{3.28}$$

where r is the ratio of width over height from the initial rectangle, and the thresholds is experimentally determined to make sure the ratio of target width and height is close to the given three ratios.

3.4.2 Observation Likelihood

Most of the dimensions of the aforementioned features depend on the input image size. Comparing features with different dimensions is not easy. One solution is normalizing the input image into same fixed size using image transformation (such as resizing, affine transform, or perspective transform).

To obtain the likelihood of the observations, a feature similarity comparison is calculated between target feature template f_i and the observation feature f_j. Traditional distance metric like correlation $\frac{cov(f_i,f_j)}{\sqrt{var(f_i)var(f_j)}}$, or cosine similarity $\frac{\langle f_i,f_j\rangle}{\|f_i\|\|f_j\|}$ where $\langle\cdot,\cdot\rangle$ is the inner product, could be adopted. In order to maximize the dissimilarity between object and background and increase the feature discrimination, an online learning algorithm based on logistic regression with ℓ_2 regularization is introduced.

3.4.3 Observation Likelihood by Ridge Regression

Learning based observation models have been widely adopted for their robustness to various conditions and discrimination against noisy background.

Ridge regression is an $\ell 2$-regularized linear regression method. Given a set of sample data $\mathcal{X}_{m\times n}$, with m samples and n features for each sample, and a set of corresponding labels $\mathcal{Y}_{m\times 1}$ for each sample. Linear regression assumes a linear relationship between the sample data and their labels

$$y_i = \mathbf{w}^T\mathbf{x}_i \tag{3.29}$$

where $\{\mathbf{x}_i, y_i\}$ is a pair of sample and its corresponding label, and $\mathbf{w}$ is the unknown parameter to be solved using the sample data.

The Ridge regression aims to find the parameter $\mathbf{w}$ that minimizes the $\ell 2$ regularized linear regression objective function

$$\begin{aligned} \min_{\mathbf{w}} \frac{1}{2} \sum_{k=1}^{m} \|y_i - \mathbf{w}^T \mathbf{x}_i\|^2 + \frac{\lambda}{2} \|\mathbf{w}\|_2^2 \\ = \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{y} - X\mathbf{w}\|^2 + \frac{\lambda}{2} \|\mathbf{w}\|_2^2 \end{aligned} \tag{3.30}$$

where λ is the $\ell 2$-regularization parameter, and in the second matrix form, X is the stacked feature matrix with size $m \times n$. Each row of X corresponds to the features $\mathbf{x}_i$ of sample i. And $\mathbf{y}$ is the stacked label vector $\{y_i\}_{i=1}^m$.

The above objective function is convex; therefore the problem is an unconstrained convex optimization problem. A closed-form solution to the problem in (3.30) can be obtained as

$$\mathbf{w} = (X^T X + \lambda I)^{-1} X^T \mathbf{y} \tag{3.31}$$

where I is the identity matrix.

As shown above, the ridge regression problem has a closed-form solution that can be quickly solved. This is crucial for real-time tracking purposes. Ridge regression is also fast to evaluate or test for applying the model on new data after solving the parameters in the training phase, and parallel programming can be applied to make use of multi-core machines.

Given a set of target image patch samples, a proper label for the samples should be designed and chosen to correctly interpret the sample similarity to the true object and dissimilarity to the background. The labels will influence the trained parameters that re-weight the feature importance to reward the similar samples and penalize dissimilar samples. In the literatures, most common label is based on Gaussian Distribution.

By fixing the mean to the position of the true object, the sample label is calculated based on that mean and a pre-defined variance parameter. A proper chosen variance parameter can result in a good performance, however, choosing such a parameter can be tricky for different object with different sizes. If variance is too large, samples around the true object will have similar labels that may bring in high objective score for too much background patches. Also, if the variance is too small, the feature will be too restricting to the true object that decreases the robustness to the change of object appearance itself.

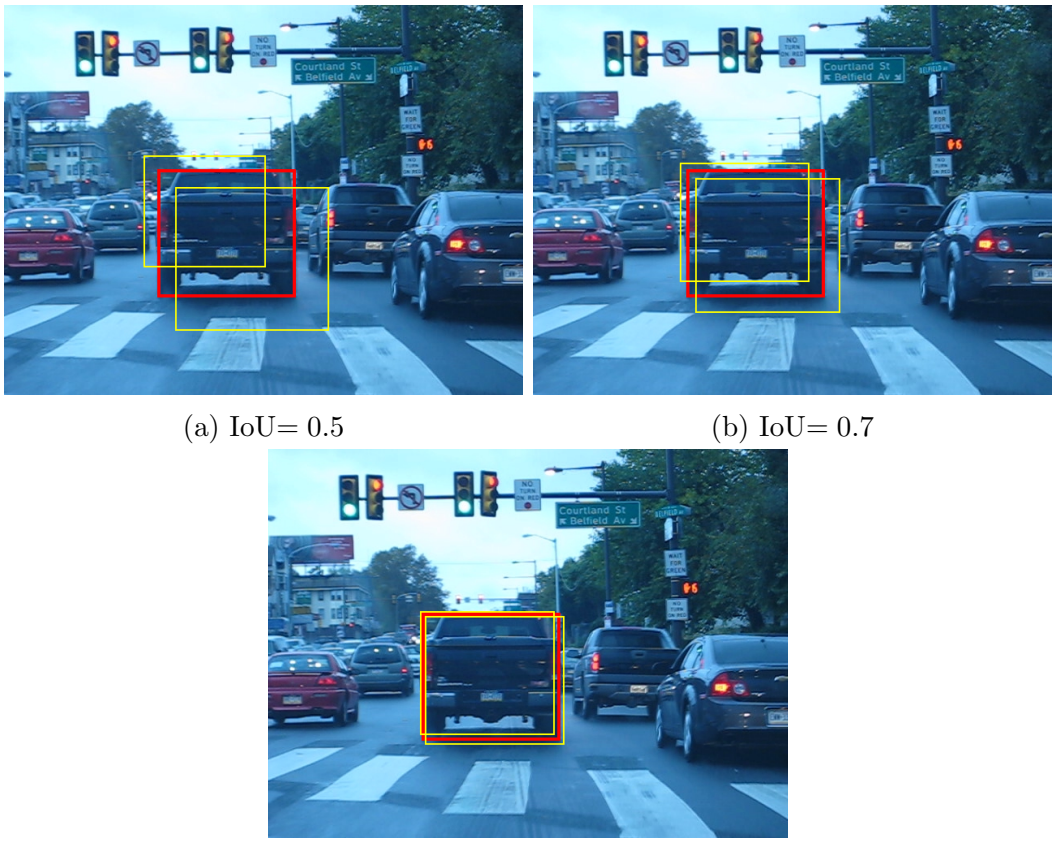
To better label the samples, an IoU based label is proposed. IoU describes both position and size information of two rectangle image patches. In the computer vision community, it's been widely used to measure many rectangle based tasks, such as the accuracy measurement for object detection as well as object tracking. The IoU of two rectangles $\mathcal{R}_1, \mathcal{R}_2$ is defined as

$$\text{IoU} = \frac{|\mathcal{R}_1 \cap \mathcal{R}_2|}{|\mathcal{R}_1 \cup \mathcal{R}_2|} \tag{3.32}$$

where $|\cdot|$ represents the area of the intersected ($\cap$) region or union ($\cup$) region of two rectangles.

Examples[2] showing the IoU score are presented in Figure 3.2. IoU score penalizes well on loose overlapping rectangles without requiring tuning parameters on sizes and position variances.

[2]Images are from OTB−100 datasets [95]



(a) IoU= 0.5 (b) IoU= 0.7

(c) IoU= 0.9

Figure 3.2: IoU scores of sample rectangles (better view in color). IoU is calculated between red and yellow rectangles. Red rectangle is fixed and used as the reference, while yellow rectangles varies in size and position.

Model Training

To calculate the observation likelihood of a sample, first the parameter $\mathbf{w}$ should be learned through training in the first frame when object location is given by the initializer. To train the model parameter, samples are drawn densely in the neighbor regions of the initial position with different scales and tilt transform as well. The sampled image patches are then augmented with different brightness level to adapt to possible further appearance changes. Having the training samples ready, the model

parameter $\mathbf{w}$ is solved by using (3.31).

After that, the model parameter is used to evaluate the observation likelihood in the rest of the frames. If a sample's appearance is closer to the initial object appearance, the sample will have higher score calculated by (3.29).

Model Update

As described above, the model parameter is learned at the first frame, and remain unchanged in the rest of the tracking process. If the object appearance does not change much, the model can represent the object's appearance well. However, to handle the appearance change that is common in real data, the model parameters need to be adaptive and updated after initialization. Observing that the object may be occluded or affected by sudden changes in illumination, model update should be only applied when the tracking confidence at current time frame is still reasonably high. This may limit the model's capability of learning new appearance of the model, but can prevent drift to the background when background information is included in the model in the wrong updates. Unless trained offline with large number of samples, the online learned models still lack the capability of adapting to various appearance changes of the object.

To update the model parameter, one problem is to decide when to update, and another one is how to update the model parameters. For the first problem, similar to [92], an evaluation score is utilized to determine whether to update parameters. The evaluation score $\mathcal{T}_S$ is calculated as

$$\mathcal{T}_S = \frac{F_{\max} - F_{\min}}{\frac{1}{N_s}\sum_{i=1}^{N_s}(F_i - F_{\min})} \tag{3.33}$$

where $F_{\max}$ is the maximum likelihood, and $F_{\min}$ is the minimum likelihood. N_s is the number of valid particles, and F_i is the likelihood of a valid particle i. Only valid particles are used for calculation, others like out-of-bound particles are discarded.

When the tracking confidence is above the threshold, the model is updated by using a mini-batch gradient descent method with learning rate η described below

$$\mathbf{w}(t) \leftarrow \mathbf{w}(t-1) - \eta \frac{\partial E}{\partial \mathbf{w}} \tag{3.34}$$

with $E = \frac{1}{2}(\mathbf{y}_t - X_t\mathbf{w})^T(\mathbf{y}_t - X_t\mathbf{w}) + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w}$, and $\{X_t, \mathbf{y}_t\}$ is the mini-batch constructed by a small set of samples in time frame t. This gradient descent update step adds new information in frame t, while keeping the old model information. The ratio of learning new and keeping old information is governed by the learning rate and number of iterations. This update step neither guarantees a global optimal of minimizing the loss function E, nor it needs to be. A global optimal may not be useful because the label created in this step is based on current tracking result instead of from external input. This means that the labels could be wrong or misleading, so a global optimal solution may be harmful to the eventual performance. Another worth mentioning merits of the gradient descent update comparing to closed-form solution in (3.31) is that it does not need to store the training data all the time. Only new information is used thereby freeing memory of the initial training data. And the computational load is much less comparing to the relatively expense matrix inversion with increasing number of dimensions that grow with the number of samples.

3.5 Occlusion and Partial Out-of-scene Handling

As the observation model employed in the last section is a holistic model that captures the overall description of the target. It is prone to fail when the target is partially visible due to occlusion or partial out-of-scene. Existing algorithms often handle these problems using part-based models that creates multiple part models simultaneously while creating the entire target model. This would linearly increase the processing time according to the number of parts used and barely contributes to the final tracking performance when no occlusion or partial out-of-scene happens.

Without using the part model solution, the occlusion and partial out-of-scene is handle separately in the proposed tracker. The occlusion is assumed to happen only within the scene and partial out-of-scene happens close to the image boundary. The image plane is divided into two regions: boundary region where object boundary coincide with the image boundary and inner region when no coincidence happens. And occlusion is only handled in inner region and partial out-of-scene is only considered in boundary region.

Since in a 2D image depth information is missing due to the perspective transformation, detecting occlusion requires aligning and comparing existing image patch with historical appearance. There is no easy solution to this. As the tracking confidence may indicates the similarity matching from current observation to the history, changing in tracking confidence can be utilized to indicate an occlusion investigation. Noting occlusion may lead tracking confidence drop but not verse versa, because other appearance change would also cause the tracking confidence to drop. In the inner region, if a significant drop of confidence happens, an occlusion detection is performed

by using the part templates shown in Figure 3.3[3]. For each divided part, features are extracted and compared with corresponding part from last high confidence tracking result. The parts with highest response is chosen to be used for the observation likelihood re-calculation.

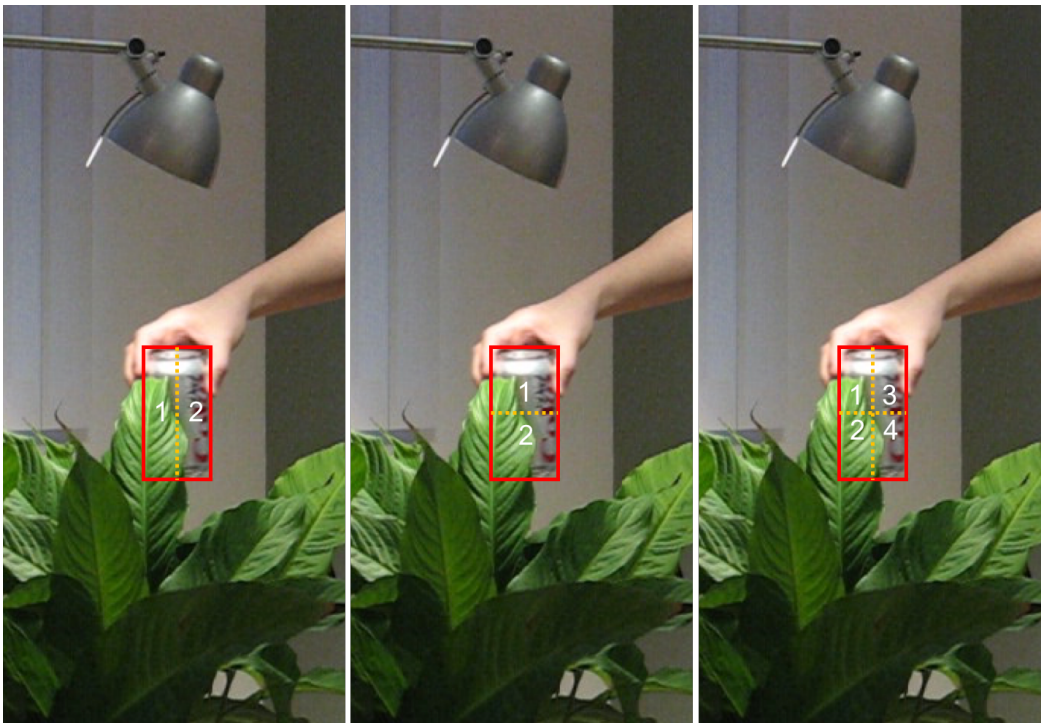


Figure 3.3: Part templates used for occlusion detection. Three different part division is used (better view in color).

In boundary region, the partial out-of-scene is easily judged by checking the intersection of object and image boundaries. If out-of-scene happens, while calculating the observation likelihood, the holistic model does not apply due to the missing part. A replacement solution is used to calculate the likelihood by correlation coefficients on extracted feature from the available part.

If an occlusion or partial out-of-scene is detected, the model is not updated when

[3]Images are from OTB−100 datasets [95]

it happens, avoiding involving background into the model. The occlusion and partial out-of-scene can be further handled using occupancy grid [27] in the multiple target tracking applications.

3.6 Experiments

3.6.1 Evaluation Methodology

To evaluate the tracking performance, labeling ground truths of the object in the video sequences is required. Based on this ground truths, several evaluation methodologies can be applied.

Precision Plot. Precision plot is one of the widely used evaluation metric for center location error. The center location error is calculated by the Euclidean distance between the center points of tracking result and the ground truths.

$$e = \|x_{\text{center}} - x_{\text{ground truth center}}\| \tag{3.35}$$

By thresholding the allowable center location error, the overall precision performance could be plotted as precision plot by a normalized count of the number of frames that center location error falls within the error threshold. Overall ranking for comparing different trackers usually adopts the precision counts at error threshold = 20 pixels [3] [94].

Success Plot. Since precision plot only gives the performance on center locations, while missing the size information, the IoU overlap score (3.32) was proposed to

measure the bounding box overlap between the tracking result and the labeled ground truths. Similar to precision score, the success plot is drawn by thresholding the overlap score on x-axis and plotting the normalized counts of the number of frames that overlap score falls within the threshold. Typical ranking of the trackers is based on the overlapping score = 0.5 [14].

Frame Rates. Despite the above accuracy measures, the frame rate or the processing time is also a significant factor of measuring a tracker. And there is often a trade-off between the accuracy and speed. The computational time of the proposed tracker are compared with several different trackers on a same computing platform based on open-source implementations, and the accuracy/speed trade-off is also discussed in the following experiments.

Experimental Setup

All the experiments are carried out on a PC equipped with an Intel Core i7-6700K 4.0 GHz CPU and 32 GB RAM memory, the GPU calculations (if there are) are executed on a single NVIDIA GeForce GTX 1080 GPU with 1607 MHz clock speed and 8 GB of GPU RAM, and the algorithms are implemented using C++11/14 with OpenCV 3.1.

3.6.2 Benchmark Datasets and Result Analysis

In the past years, several versatile datasets have been published to the tracking research community for publicly benchmarking the single target tracker performance. These datasets have not only helped researchers compare the performance of the trackers, but also enabled people discover the strength and weakness of existing trackers

and improve them.

VTB-100 Datasets VTB-100[4] [95] is a public dataset collection that contains 100 sequences of images captured in different lighting and camera conditions. The datasets consist of different types of objects to be tracked, such as pedestrian, birds, face, toy, vehicle, and etc. Variety of the objects, rich conditions of the environment, and different types of camera performance and camera motion create different problems for the trackers to handle, and therefore VTB-100 is a reasonable testbed for benchmarking the proposed tracking algorithm. To characterize and summarize the challenges in tracking objects in the datasets, the authors proposed several different attributes as listed in Table 3.1.

[4] `http://www.visual-tracking.net`

Table 3.1: Attributes of VTB-100 datasets representing challenges for tracking.

Attributes	Description
Illumination Variation (IV)	Significant change in illumination condition on the target.
Scale Variation (SV)	Target size varies in different frames.
Occlusion(OCC)	Full or partial occlusion on the target.
Deformation(DEF)	Target deformation.
Motion Blur (MB)	Blurred target appearance due to motion.
Fast Motion (FM)	The movement of the target is larger than tm pixels (tm=20).
In-Plane Rotation (IPR)	The target rotates within the image plane.
Out-of-Plane Rotation (OPR)	The target rotates outside the image plane.
Out-of-View (OV)	The target is partially out of field of view.
Low Resolution (LR)	The area of the target's bounding box is less than tr pixels (tr =400).

Attribute-based Benchmarking To address the capability of the proposed tracker and better analyze the performance in terms of advantages and disadvantages, benchmarking is carried out by groups of video sequences classified based on different attributes, as well as an overall averaged performance comparison. The exact grouping of the video sequences to different attributes can be found in [95]. The tracker is initialized in the first frame using the ground truth position and run sequentially frame by frame without extra input except for the images after that.

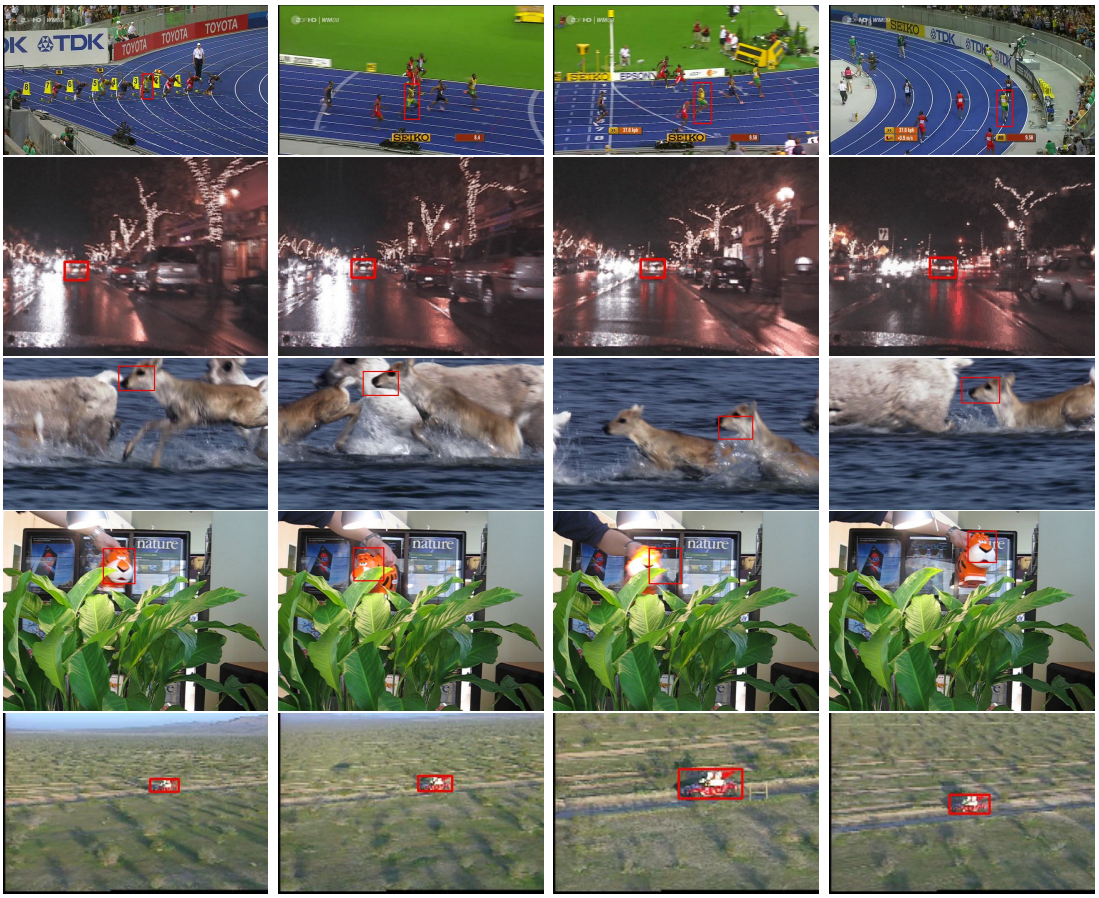


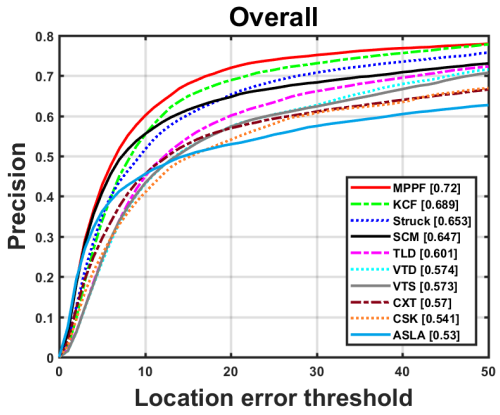
Figure 3.4: Sample tracking results (in red rectangle) on VTB-100 datasets (better view in color). From the top to the bottom, sample results are take from sequence 'bolt' (frame #1, #196, #237, #349), 'CarDark' (frame #1, #28, #138, #250), 'deer' (frame #1, #28, #42, #70), 'Tiger1' (frame #1, #106, #177, #267), 'RedTeam'(frame #2, #234, #482, #667)

Five typical scenarios in the benchmark is presented in Figure 3.4. The 'bolt' data shows changes in object's own appearance change, from different viewpoint, the proposed tracker adapts to the changes of the objects. In the second sequence 'CarDark', under complex illumination conditions in the dark, the tracker follows the vehicle correctly. In third 'deer' sequence, abrupt motion happens throughout the data, and the mixed-motion proposal helps the tracker correctly track the deer head.

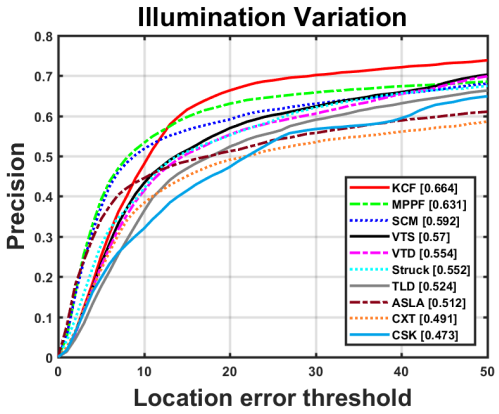
In the fourth 'Tiger1' data, the toy tiger is occluded frequently by the leaves. In frame #177, the occlusion and illumination change caused the tracker drift, however, the tracker successfully recovers from the drift. The last example shows the tracker adaptively follows the scale change of the object.

Using the VTB-100 benchmark suite, the proposed algorithm is evaluated quantitatively with recent published algorithms' results on the benchmark, namely, ASLA [45], cpf [74], CSK [39], CXT [22], DFT [81], KCF [40], LOT [71], LSK [63], MIL [3], MTT [99], OAB [32], sbt [33], Struck [36], SCM [100], TLD [46], TM [95], VTD [55], and VTS [56]. The proposed tracker name is abbreviated as MMPF. All the experiments adopt the exact same parameters meaning the algorithm is not tuned for any specific data sequences.

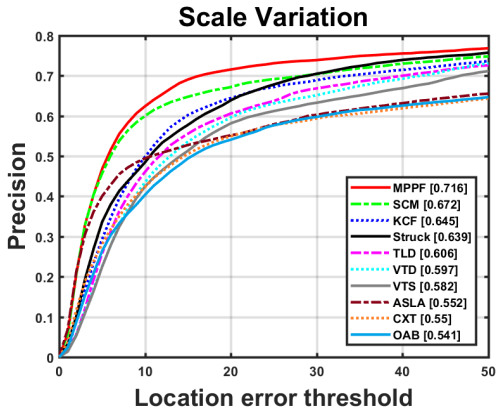
The precision plot is shown in Figure 3.5 and Figure 3.6. Note that only the evaluations of the top 10 performed algorithm is drawn in the figures.



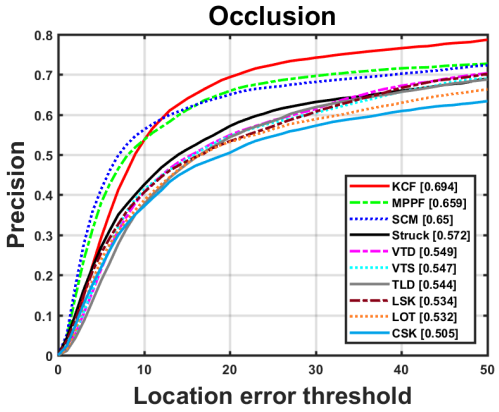
(a) Overall



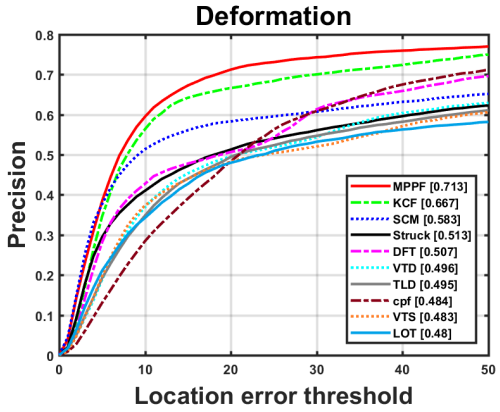
(b) Illumination Variation



(c) Scale Variation

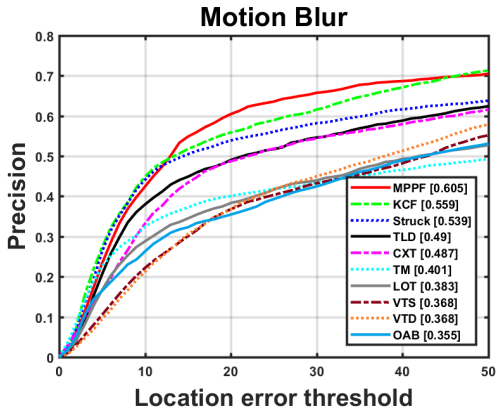


(d) Occlusion

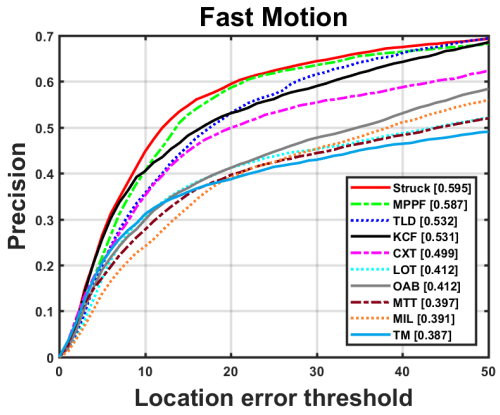


(e) Deformation

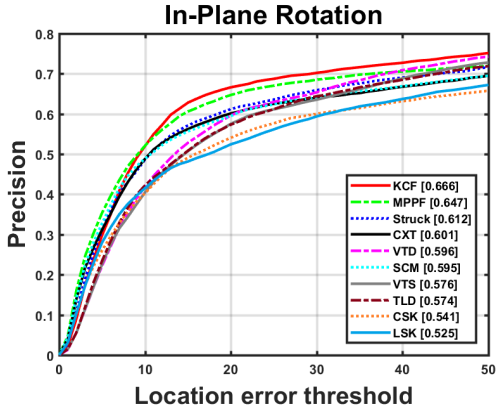
Figure 3.5: VTB-100 Tracking Results - Precision plots (better view in color).



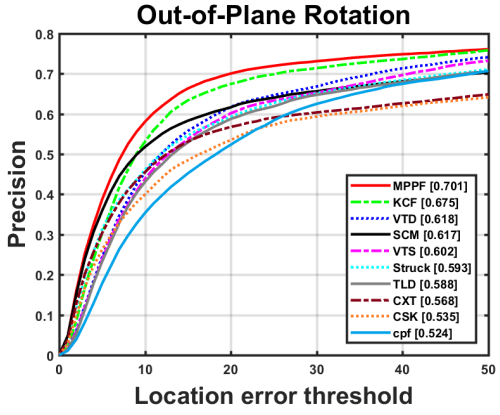
(a) Motion Blur



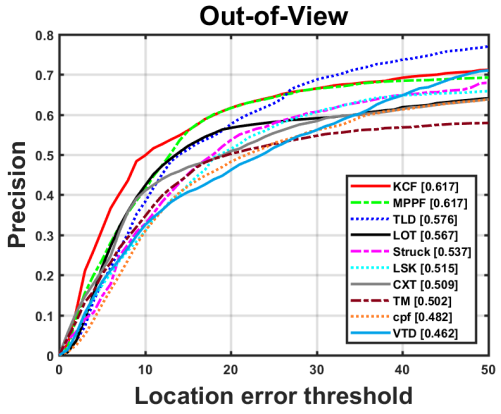
(b) Fast Motion



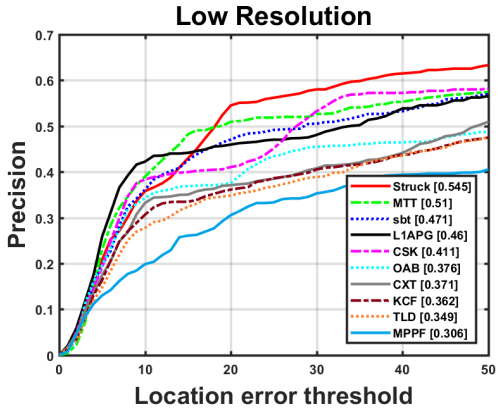
(c) In-Plane Rotation



(d) Out-of-Plane Rotation



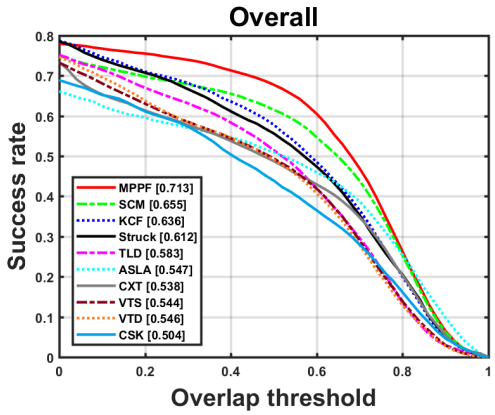
(e) Out-of-View



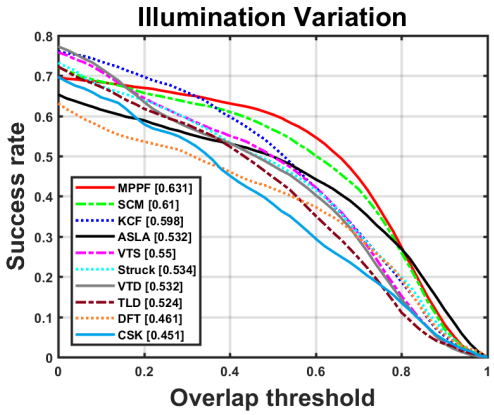
(f) Low Resolution

Figure 3.6: VTB-100 Tracking Results - Precision plots (better view in color) cont.

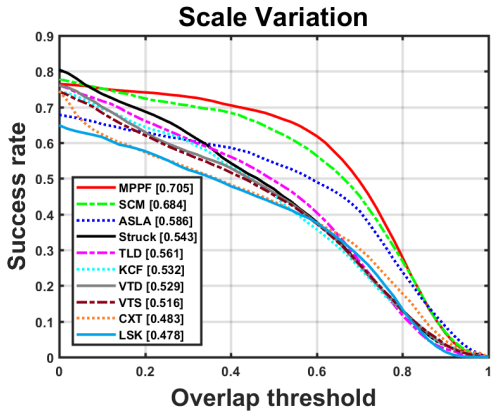
The succession plot is shown in Figure 3.7 and Figure 3.8.



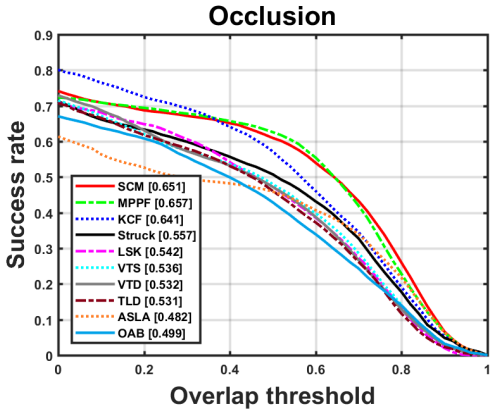
(a) Overall



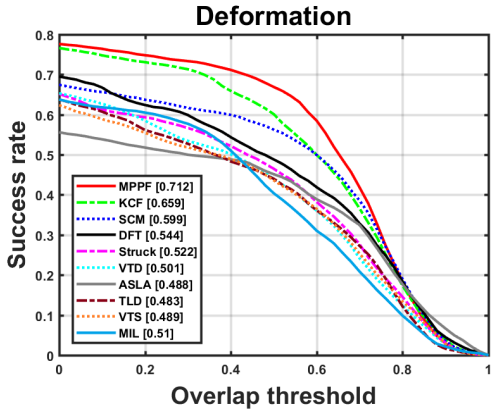
(b) Illumination Variation



(c) Scale Variation

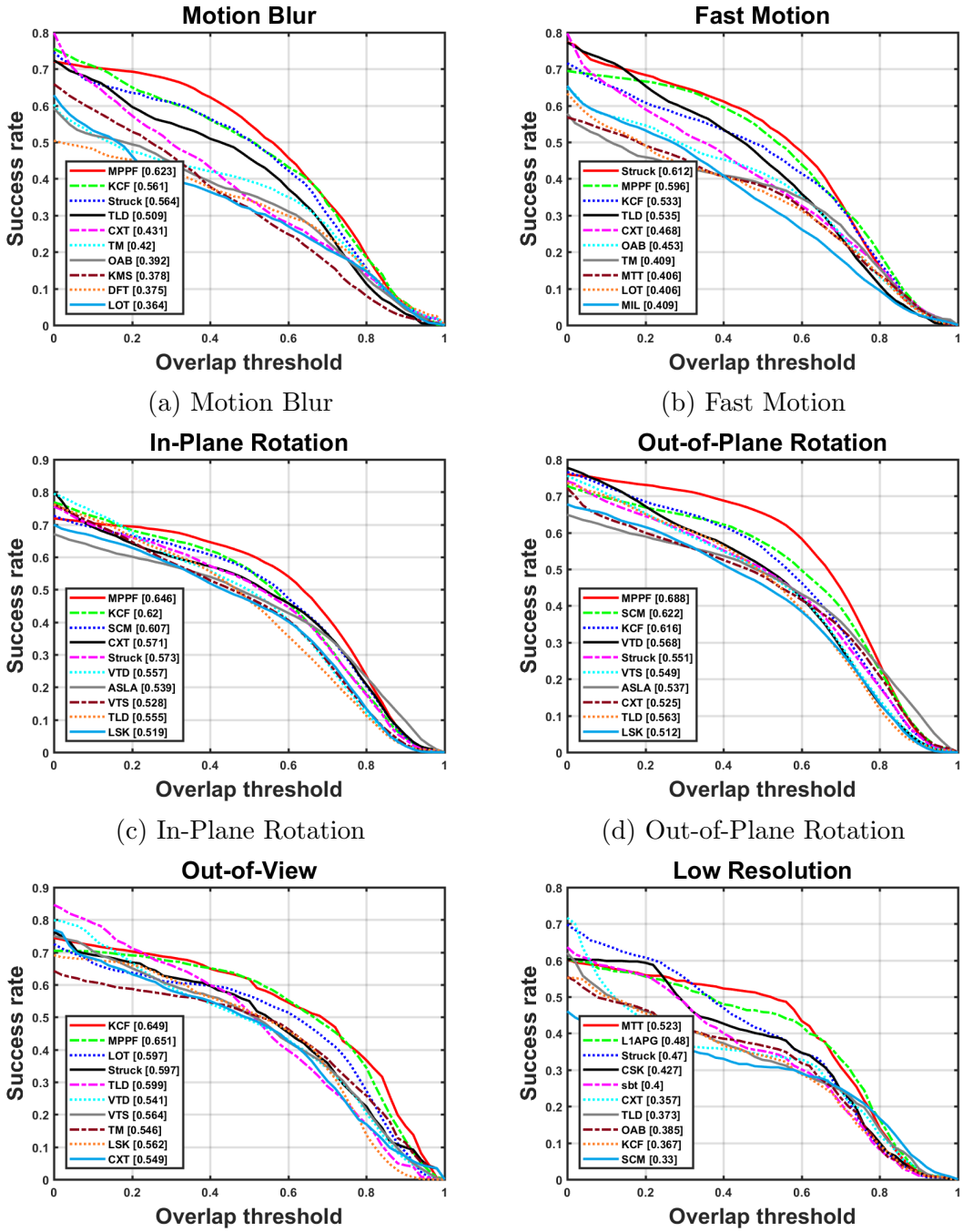


(d) Occlusion



(e) Deformation

Figure 3.7: VTB-100 Tracking Results - Success plots (better view in color).



(a) Motion Blur (b) Fast Motion

(c) In-Plane Rotation (d) Out-of-Plane Rotation

(e) Out-of-View (f) Low Resolution

Figure 3.8: VTB-100 Tracking Results - Success plots (better view in color) cont.

Overall, the proposed algorithm performs best in terms of center error and bounding box overlap as the ranks shown in the precision and success plots. The proposed algorithm achieves highest rank in 6 different sub-categories and 3 second ranks for fast motion in the success plots, and the algorithm performs poorly in the low resolution sequences. However, comparing to other categories, all trackers suffer from low resolution and perform worse than in other categories. The reason behind this is because less information can be extracted to localize the targets when the image resolution is low. This quantitative benchmarking shows the good accuracy of the proposed tracker comparing to the state-of-art published algorithms.

The proposed tracker is designed to achieve high speed without losing the accuracy. The speed of the algorithm is compared with the classical KCF algorithm from the popular correlation filter family on the test machine. The proposed algorithm can achieve an average of 400 FPS while KCF algorithm can only perform 41 FPS on average, showing the proposed tracker is almost 10x faster than the popular KCF algorithm. The speed is crucial to apply the algorithm in multiple target tracking scenarios presented in the following chapter, which will increase the processing time linearly to the number of targets.

NfS Datasets Since high frame rate is achievable by the proposed algorithm, a high frame rate video dataset called NfS [29] is being also tested with. Each sequence in the NfS datasets contains a low and a high frame rate video with 30 FPS and 240 FPS respectively. The 240 FPS sequences are utilized for benchmarking with the proposed algorithm to test the high frame rate performance, and according to the processing time from previous tests, with such a high frame rate video, the algorithm should still achieve real-time processing throughput with no delay.

Result Analysis The tracking algorithm is compared with algorithms, namely KCF [40], Staple [10] and CFwLB [49], that processing more than 50 FPS according to the claimed benchmark results provided in [29]. The precision and success plots are shown in Figure 3.9 and Figure 3.10 respectively.

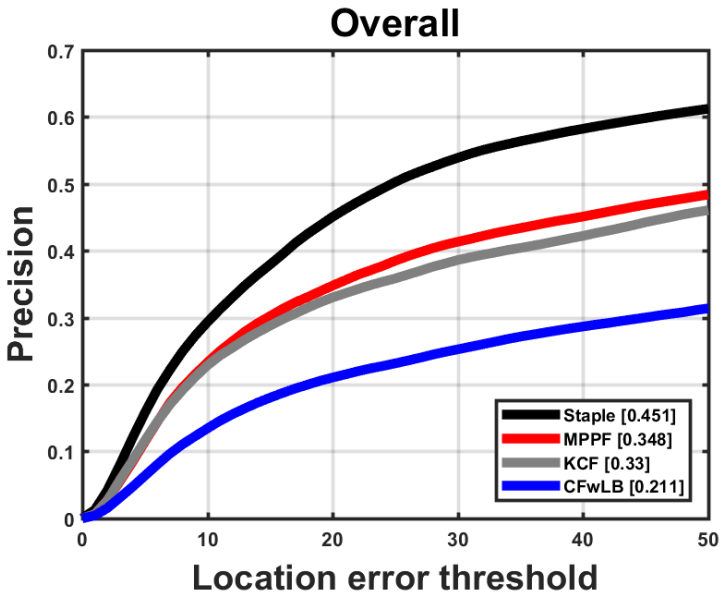


Figure 3.9: NfS Tracking Results - Precision plots (better view in color).

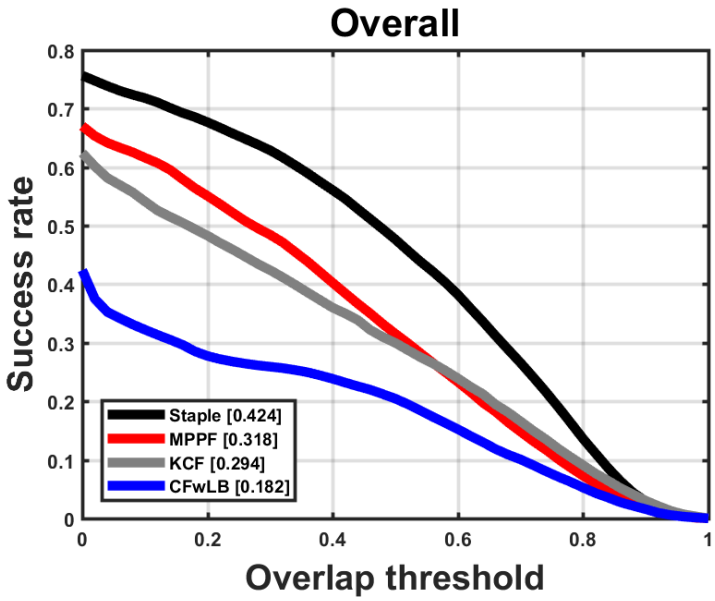


Figure 3.10: NfS Tracking Results - Success plots (better view in color).

According to [29], the speed for the given algorithms are claimed as KCF (170.4 FPS), CFwLB (83 FPS) and Staple (50.8 FPS) in their tests. Still, the proposed tracker (400 FPS) outperforms the mentioned algorithms in speed with a large margin while maintaining the comparable accuracy.

3.7 Conclusion

A high speed visual target tracker is presented in this chapter, without losing the performance in terms of accuracy. The proposed tracker utilizes a mixed-motion proposal particle filter, and employs the Ridge Regression to learn and update a robust appearance model. Occlusion and partial out-of-scene is also handled in the tracker. Experiments and comparison on public dataset shows the proposed tracker achieves best performance and outperforms other algorithms in speed with a large

margin. Further extension of the proposed method to a multiple target algorithm is presented in Chapter 4.

Chapter 4

Multiple Ground Target Tracking using a Single Camera in Urban Scene

4.1 Introduction

Autonomous driving is expected to be the next big change to the world and people's lifestyle. Perception is the key to many ADAS and ADS. Higher level of understanding of other vehicle's behaviors and path planning could be done without perception of the surrounding ground targets. Thus detection and tracking ground targets in these scenarios is one significant part of the perception system.

To tackle the multiple ground target visual tracking problem, various algorithms addressing different aspects have been proposed. There are online and offline algorithms solving the tracking problems in different ways: offline algorithms that make

use of full sequence of the images to generate tracks usually perform better than online algorithms that can only use current and previous inputs. As this thesis mainly aims to develop algorithms for the real-time applications of visual target tracking, only online algorithms are investigated.

An online multiple target tracking system usually divides into four parts: object detection, data association, filtering/estimation, and track management. The object detection module localizes the object in images utilizing prior knowledge or information, like background models in still cameras or machine learning algorithms based on offline training. For detecting ground targets in a moving platform, typically the machine learning algorithms is the preferred solution. As the breakthrough made by the deep learning algorithms with the help of modern GPU technologies, detection algorithms could be divided into traditional shallow learning based and deep learning based. Traditional algorithms that are used for ground target detection include Haar/AdaBoost [91], HOG/SVM [20], DPM [25] and their various variations. These traditional algorithms are often designed with hand-crafted features and then incorporate the features with certain machine learning algorithms to train a binary classifier to predict whether the feature is an object or not. The feature is the key to those algorithms and single model of these algorithms can only detect single class of objects or even single pose of the objects. Applying these algorithms to detect multiple types of ground targets would linearly or near linearly increase the computation.

With the era of deep learning that started in 2012, the CNN based detections algorithms are proposed and keep improving the detection performance and increasing the processing speed. Regional proposal based methods adopt only a subset of the sub-windows of the image and classify them using CNN, comparing the need to scan

every sub-window by traditional algorithms. Fast R-CNN [31] uses selctive-search and Faster R-CNN [78] uses a subnetwork inside the detection network to generate the proposals and then classify detected target types. End-to-End algorithms, such as Yolo9000 [76], SSD [64], learn to predict the detections directly without using the regional proposals.

The CNN based detection algorithms outperform the traditional ones and output the class information together with the object locations, this necessitates the design of new data association algorithms for multi-class multi-target tracking. The data association performs measurement to existing track association when multiple potential detection for one target is observed. The GNN or 2D assignment algorithm is most widely used in online ground target tracking due to its simplicity and reasonable performance with good detection. However, noise and missing detections would lead to wrong associations that needs to be improved.

The filtering/estimation utilize the measurements and output the target states. This part is similar to traditional radar tracking.

The track management maintains the tracking system, making the judgment of when to create new tracks and remove disappeared tracks. In ground target tracking in urban scene, targets enter and exiting the scene frequently. A track management algorithm would make significant different on the overall performance.

The proposed algorithm improves the multiple ground target tracking in the following way. The detection is generated using a CNN based detector and improved by using perspective information. Multi-class data association is proposed using a new hierarchical object class tree based 2D assignments. The track management is enhanced to handle the frequent entering and exiting of objects by using the multiple

extracted features in the tracks and using a novel SVM based deletion prediction.

4.2 Perspective Aware Object Detection

4.2.1 Real-time Object Detection using Convolutional Neural Networks

Recent advancement of parallel computation hardwares and explosively generated visual data on the Internet boost the neural networks into deeper structures. For the CNNs commonly used for vision tasks, with deeper structures or layers, the networks are capable of learning better representation of various object and utilizing large amount of data, usually causing overfitting problem in shallowing learning algorithms. Like the traditional detection algorithms such as Haar/AdaBoost, HOG/SVM or DPM, the CNN based detectors perform the same tasks of detecting interested objects in the images and output the location.

A Regional Proposal Networks (RPN) [78] based algorithms is employed in this paper. The RPN is the proposal network that generates anchors at different scale with different width/height ratio. The base network before the RPN can be different types of networks, such as VGG16/VGG19 [82] and Residual Networks (ResNet) [37]. Comparing to the traditional CNN detectors that use CNN as a classifier to classify object image patches proposed by an external proposal algorithm. This method moves the proposal module into the neural network, and trains an end to end solution for object detection with real-time or near real-time (depends on different base networks) performance on a GPU.

Figure 4.1: Common scaling strategies in traditional detection algorithm (left column) and the proposed perspective aware scaling strategy (right column).

4.2.2 Perspective Aware Ground Target Detection

Object detection algorithm introduced in aforementioned section uses RPN to generate proposals of potential object location. This type of detection algorithm performs poorly on small targets in road scenarios. One potential improvement is to adopt the multi-scale strategy (left column in Figure 4.1) as used in transitional object detection algorithms, by building a pyramid of images resized from original image sizes to detect the objects and then convert back into the original image scale. However, scaling may not help detect small objects far away since most of the scaling strategy is based on center point of the image plane, while small object may not always locate in the center as illustrated in Figure 4.1. The middle column shows scaling using center point may cut out the interested small targets that are supposed to be detected. To make better use of this strategy, a perspective aware scaling is proposed.

Vanishing Point based Perspective Information Estimation

To obtain a good perspective information of the scenario, a vanishing point based perspective estimation is used. In road scenarios, lane markings and guard rails are parallel to each other, and they will pass the vanishing point on the image plane. Also, a roughly plain surface of the road is assumed. Single image vanishing point estimation is implemented based on [58] and updating vanishing point position is done periodically instead of every frame to save computational load. Using of vanishing point in the image, the plain surface could be located on the image plane. And the perspective information of the objects on this surface would be inferred. Based on this perspective geometrical information, the scaling is performed as examples shown in the right column in Figure 4.1, that the small objects will not be cropped as illustrated in the middle column.

The number of scales and the size of each scale can be determined through statistical analysis of the performance of each network structures on a given dataset and choosing best of them. However, this trivial process involves a lot of retraining and testing that requires plenty of computational resources and time, to simplify the process and demonstrate the effectiveness of the proposed perspective aware strategy, heuristic numbers are used as detailed in the following experiment section.

Non-maximum Suppression with Non-Holistic Suppression

Detection outputs in different scales have to be converted back into the original image scale as final outputs. In traditional detection algorithms, detection models are usually trained for a holistic appearance model, for example, a whole-car model can only output detections on holistic image of the car, and partial occluded cars could

be not detected. However, this is not applicable since CNN based detection could output partial objects. So, converting different scales in to one scale introduces a new problem that multiple detections including partial detections could be generated on same targets.

Noticing that only objects on near the boundary of the image could have such a problem, a non-maximum suppression with non-holistic suppression is proposed to solve this problem. The algorithm is described in Algorithm 1.

Data: Raw detections $\mathcal{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$ in each scale $1, \ldots, n$ sorted from small to large.

Result: Raw detections without non-holistic objects.

```
while d in d_i for i = 1 to n do
    while d_k ∈ d_j for j = i + 1 to n do
        if d inside d_k AND d's edge close to any edge of d_k then
            delete d.
        end
    end
end
```

Algorithm 1: Non-holistic suppression.

The output of non-holistic suppression is then fed into a regular non-maximum suppression algorithm which then outputs the final detections.

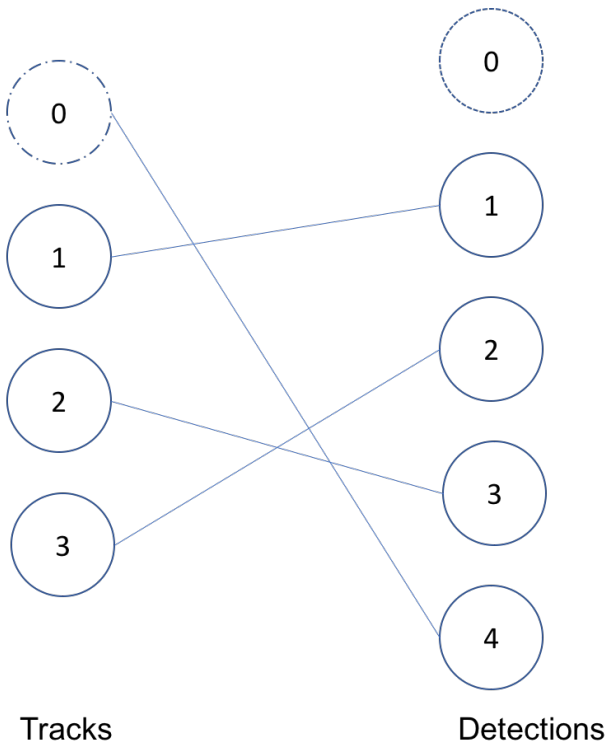


Figure 4.2: Illustration of a 2D assignment problem. The assignment algorithm aims to find the optimal association pairs between the detections and tracks.

4.3 Multi-Class Data Association using 2D Assignment with Hierarchical Object Class Tree

Detection algorithms generates new detections in the new frame, to associate newly detected objects with the existing tracks, a data association is needed.

4.3.1 2D Assignment

Given a set of detections $\mathcal{D} = \{d_i\}, i = 1, \ldots, N_R$, and a set of existing tracks $\mathcal{T} = \{t_j\}, j = 1, \ldots, N_C$, an association indicator variable $x_{i,j} \in \{0, 1\}$ is used to describe whether detection i is associated with track j. In mathematical form, the

2D assignment algorithms can be expressed as the optimization of the following problem:

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{i}^{N_R} \sum_{j}^{N_C} c_{i,j} x_{i,j} \tag{4.1}$$

$$\text{subject to } \sum_{i}^{N_R} x_{i,j} = 1 \quad \forall i \tag{4.2}$$

$$\sum_{j}^{N_C} x_{i,j} = 1 \quad \forall j \tag{4.3}$$

$$x_{i,j} \in \{0, 1\} \quad \forall i, j \tag{4.4}$$

where $c_{i,j}$ is the associate cost.

Solving the above optimization problem will give the optimal assignment pairs of given detections and tracks, and the lowest cost assignment is the solution of it. The optimization algorithm could be solved using the JVC algorithm [19], that is a faster variant of the $O(N^4)$ complexity Hungarian algorithm [19].

As shown in Figure 4.2, dummy nodes could be added into the detection and track column, representing false alarm and missing detection respectively. However, different to radar or sonar sensors, which have reasonable statistical distribution assumption for random noises, in the image tracking, noise is hard to characterize using known random noise distributions. Therefore, the assignment cost for linking detections or tracks to the dummy nodes could not be calculated. Without using the dummy node to represent false alarm and missing detections, good track management and prediction algorithm could help alleviate their effects to the tracking performance.

4.3.2 Multi-Class Data Association

In most of the traditional object detection algorithms as mentioned in Section 4.2.1, only one type of objects could be detected. In order to detect multi-class or multiple types of objects, the models for each different type of objects need to be evaluated one at a time, which will significantly increase the detection processing time. Thus the detection part will limit the capability of tracking multiple different types of targets in videos. However, one of the benefits from adopting the CNN based detectors is that different type of targets could be detected with only one pass evaluation of the model. Therefore, multiple different types of object could be detected without increasing any computational load by using CNN based detectors.

To track multiple targets with different class information, one could either use an object class agnostic way or class specific way to handle feed the data association. The former method treats all the detections as type agnostic targets and tracks them indistinctively. The second method treats targets separately and conduct data association for each separate classes of objects by solving multiple 2D assignment optimization described in Equation 4.1 with subsets of association pairs.

Although the second method may increase the difficulties in designing the system for multiple classes, there are also benefits of associating detections and tracks for each class separately. The first benefit is the reduction in association computational load. Building an association pairs of detections and tracks of all the types of targets would result in a huge association matrix therefore increasing unnecessary computations. With the availability of class information, the association matrix could be divided into blocks and the association could be handled individually. Second, with the class information, inter-class association error could be avoided. However this does

introduce an extra step for handling occlusion, since only considering single class of object would blind that class's tracking algorithm about occlusions between different types of objects. Thirdly, based on class specific association algorithms, separate tracking algorithm including different track management methods could be utilized to maximizing the performance for tracking multiple types of targets.

4.3.3 Hierarchical Object Class Tree for Multi-Class Data Association

One drawback of the aforementioned multi-class data association algorithm is that the object class is not always classified correctly by the detector, especially when the object is being partial occluded. Also, the same object may have different class labels, that causes missing detections for some targets and broken tracks. A hierarchical object class tree is proposed to handle this problem.

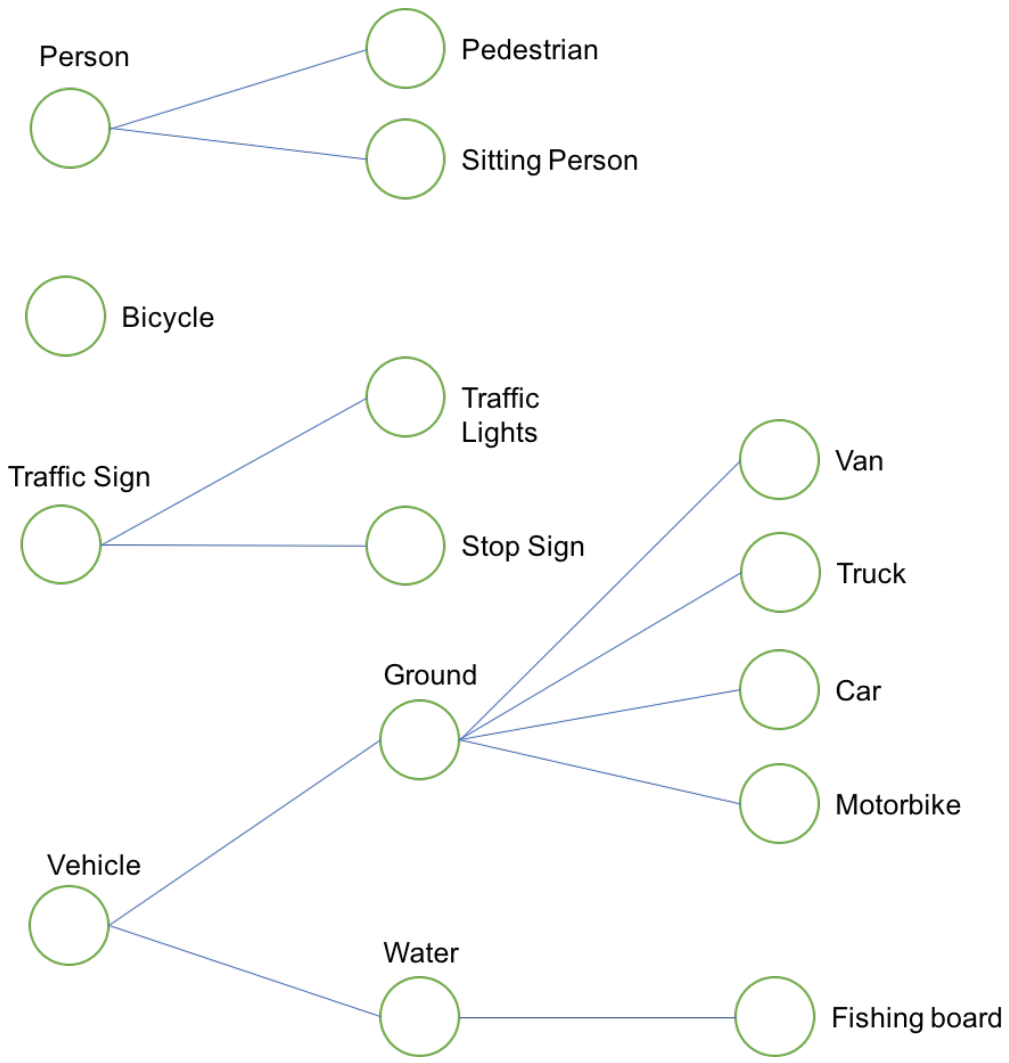


Figure 4.3: An example of hierarchical object class tree representation based on their common attributes.

An example of hierarchical object class tree is given in Figure 4.3. In the detection framework, only the end leaf nodes are used in the last softmax loss function to predict class information, their relationship is not addressed. Handling misclassification problems in the detections, based on the hierarchical representation, depends on the classes that are similar to each other and prone to have misclassification error, the

division of subsets based on classes is modified to include some of their parent nodes. For example, trucks, vans and cars could share a same subset, same for people and bicycle. How to build the hierarchical tree of different classes depends on available classes in different training datasets, and the principal is to include visual similar classes and classes that are potentially incorrectly classified.

4.3.4 Association Cost

In Equation 4.1, association cost $c_{i,j}$ is used to encourage correct association of the detection to the true target and penalize the wrong association. The set of $c_{i,j}$ can be formated as an assignment matrix $\mathcal{C}^{R\times C} = [c_{i,j}]$. The assignment matrix is defined as the combination of motion and appearance feature is adopted to compute the assignment cost,

$$\begin{aligned} \mathcal{C} &= \mathbf{w} \cdot \mathbf{\Phi}(\mathcal{D}, \mathcal{T}) \\ &= \begin{bmatrix} w_m \\ w_f \end{bmatrix} \cdot \begin{bmatrix} \mathbf{\Phi_m}(\mathcal{D}, \mathcal{T}) \\ \mathbf{\Phi_f}(\mathcal{D}, \mathcal{T}) \end{bmatrix} \end{aligned} \tag{4.5}$$

where $\mathbf{\Phi}_m(\mathcal{D}, \mathcal{T})$ is the motion information calculated based on the bounding overlap of $\mathcal{D}$ and $\mathcal{T}$, $\mathbf{\Phi_f}(\mathcal{D}, \mathcal{T})$ is the appearance feature similarity between the detection and targets, and $\mathbf{w}$ is the predefined weight vector that balances the contribution of each components.

The JVC algorithm yields the global optimal solution to the assignment problem, it will be easily mis-led by the missing detections and false alarms. Therefore the assignment cost should reject those potential association pairs before affecting computing optimal assignment solutions. A gating function is applied to calculate $\mathbf{\Phi_m}$

and $\mathbf{\Phi_f}$,

$$\mathbf{\Phi_m}(\mathcal{D},\mathcal{T}) = \begin{cases} \mathbf{\Phi_m}(\mathcal{D},\mathcal{T}) & \text{if } \mathbf{\Phi_m}(\mathcal{D},\mathcal{T}) < \phi_m^{th}, \\ \infty, & \text{otherwise.} \end{cases} \tag{4.6}$$

$$\mathbf{\Phi_f}(\mathcal{D},\mathcal{T}) = \begin{cases} \mathbf{\Phi_f}(\mathcal{D},\mathcal{T}) & \text{if } \mathbf{\Phi_f}(\mathcal{D},\mathcal{T}) < \phi_f^{th}, \\ \infty, & \text{otherwise.} \end{cases} \tag{4.7}$$

where ϕ_m^{th} and ϕ_f^{th} are the gating thresholds.

4.4 Track Management

Track management is an important component in a multi-target tracking algorithm. It aims to manage states of the tracks, such as the creation of new tracks and the deletion of dead tracks.

In radar-like sensor based tracking, adding a confirmation step instead of creating tracks immediately is for reducing number of false alarms, caused by random noise, being treated as targets. However, this may not be effective in other sensor applications in which false alarms are not caused by random noise, like in image tracking, a false alarm caused by mis-classification may not always randomly show up in detections, in other words, they can be detected like a target continuously in several consecutive frames. This difference requires new approaches to handle the track management in visual tracking.

On a moving platform, due to the relative motion between the camera base platform (the ego-vehicle) and other objects, object showing and disappearing happens almost all the time. Therefore, the quality of the track management will considerably impact the entire tracking performance, like false alarms rate, missing tracks, track

identity switch, and track fragments.

The track states are often defined as,

1. Tentative tracks. A tentative track is the initial state when a track is created, and it remains tentative until it is being confirmed.

2. Confirmed tracks. A confirmed track means the track has enough detections and high detection confidence to be confirmed as a true object track, otherwise it will be tentative. A confirmed track will be outputted as tracking result while tentative tracks are only maintained internally.

3. Dead tracks. A track will be marked as dead track if it is considered as out of the scope of tracking scenario.

Track management algorithm manages the above states by creating new tracks, updating tentative states, and eliminate dead tracks. The track creation and deletion determines the number of missing tracks and false tracks, a good management algorithm should minimize these numbers.

4.4.1 Track Creation

Track creation initializes new targets and adds them into the existing tracks for maintenance. After the data association, the measurements left unassociated will be considered for creating a new track. Two criteria are examined to determine if the unassociated measurement is a new target, one is the detection confidence. Since the detector is trained based on manual labeled data, higher confidence means more likely the object is the target. A target is created immediately when the unassociated measurement's detection confidence exceed certain threshold. For the rest lower threshold

measurements, a tentative track is created and normal tracking is performed thereafter except tentative track is not outputted. The tentative tracks will be upgraded to confirmed tracks only if the first criteria is satisfied or consecutively associated with lower threshold detections for a certain number of times.

4.4.2 Tentative Track Update

The tentative tracks are created when the detection confidence associated to the tracks is not high enough to confirm the track as a true track. The states of tentative tracks could either be confirmed as a confirmed track or be deleted as a dead track. Observing that if a target has been detected multiple times with low confidence, it could be an actual target. Based on this, a threshold count based confirmation of tentative track is utilized. If the times of a tentative track has been associated with low confidence detections are above the threshold, the track will be updated to be a confirmed track. Tentative tracks will remain their states until this threshold is satisfied. A proper threshold would balance between resisting to the false alarms and reducing missing tracks.

Another threshold for the number of no detection associated to the track is applied for deleting a tentative track. If the unassociated number is beyond the threshold, the track will be deleted, assuming the previous detections associated come from false alarms. This is useful to reduce the computational burden and maintain only few eligible tentative tracks that could be an actual target trajectory.

4.4.3 Track Deletion

Track deletion removes the tracks that are no longer receiving updates from the measurements (i.e. no measurement associated with them) or being out of field of view. Main reason of the track deletion is long term occlusion or target moved out of the scenario. In some applications that require re-identify objects if it disappeared for certain time, such as video surveillance, track deletion should be paid more attention and adapted to these special needs. In the purpose of developing tracking algorithms on ADAS or ADS platforms, maintaining a long history of identities is neither needed, nor computationally efficient.

Learning to Predict Track Deletion using a Support Vector Machine Different from the task of track creating, more features accumulated in the tracking process, can be adopted for predicting the disappearance of a track. To utilizing these features, a SVM based prediction algorithm is proposed to predict the existence of a track and handle the track removal if it is a dead track.

To help classify whether the track should be deleted, multiple features could be adopted. These features are summarized in Table 4.1. M, N are empirical parameters set in experiments. All of these features are normalized values in between 0 and 1.

The track deletion is then converted into a soft-margin optimization problem (as known as SVM),

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} C\sum_{i=1}^{n}\xi_i + \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{subject to: } y_i(\mathbf{w}^T f_i + b) \geq 1 - \xi_i \text{ and,} \\ &\xi_i \geq 0, \forall i \end{aligned} \tag{4.8}$$

where f_i is the feature vector for sample i, ξ_i are slack variables, and $y_i = +1$ and

Table 4.1: Features used to classify track deletion.

Feature	Description
Appearance	Mean normalized correlation coefficients (NCC) between associated detection and visual tracking output
Discrimination	Mean NCC between visual tracking output and M neighbor samples with bounding box overlap < 0.1.
Boundary	Normalized distance to the closest boundaries
Detection Overlap	Bounding box overlap between the associated detection and track prediction
Detection Confidence	Confidence of the associated detection
Tracking Confidence	Visual tracking confidence of last frame
Detection Rate	Percentage of detections in last N frame window
Missing Detection Rate	Percentage of detections in last N frame window
Mean Detection Confidence	Mean of detection confidence in last N frame window

$y_i = -1$ represents correct prediction of deletion and wrong prediction according to ground truth respectively.

4.4.4 Track Management Post-Process – Visual Tracking Feature Update

A post-process step is inserted after track management, the visual tracking feature update.

As in popular visual object tracking algorithms, especially for single target tracking without incorporating additional detection algorithm, feature update is a significant part to ensure continuously tracking of the object. The online learned algorithm only represents existing knowledge from the past, while new un-seen information needs to be added into existing features. For example, in single object setup, the

initial position is usually given through manual labeling by human. In order to adapt to the target motion and appearance change in a long period of the tracking process, the trackers need to use as much prior knowledge or assumptions on the object motion or appearance change as possible, to correct the feature template while running the tracker. However, if the assumptions do not strictly hold or the prior information is not accurate enough, the feature templates will be corrupted by the background or noise and eventually cause the tracker drifting or deviating from the actual object.

One benefit of applying single target tracking with the multiple target tracking framework is that there is another prior information to be used, the offline trained detectors, commonly not available in single target trackers. The detectors are usually supervised learning models assembling prior information about whether an image patch is a vehicle or not from human, and this information can supervise and guide the tracker to update its appearance model under different appearance changes, or even re-create a new appearance model according to the new detection. This post-processing step helps the tracker adapt to new changes in object appearance and increases the robustness.

As a summary, the entire tracking framework is listed in Algorithm 2.

Data: Video frames at time step t, and existing track set $\mathcal{T}_{t-1}$ from last frame

Result: Track set $\mathcal{T}_t$ in current frame.

1: Detecting objects in the new frame t

2: Individual tracking of each object in $\mathcal{T}_{t-1}$

3: Associating the detected objects and tracks

4: Update features according to association result

5: Update track management

6: State estimation for remaining targets and formulate $\mathcal{T}_t$

Algorithm 2: Proposed Tracking Algorithm.

Step 1 and 2 have the most computational tasks. Since they have no data dependency, these two steps can be parallelized. As the detection algorithms is implemented on GPU and tracking algorithm could run on CPU, this hardware parallelization effectively reduces the processing time and achieves real-time processing as presented in the experiment section.

4.5 Experiments and Analysis

4.5.1 Experimental Setup

The experimental setup is same as described in Section 3.6.1, except the detector training part was run with two NVIDIA GTX 1080 GPUs, and the testing part is executed with single GPU.

4.5.2 Training Detector for Ground Targets

To train a CNN detector for ground target detection, enough labeled ground truths of objects in the images are needed. For road environment applications, the KITTI datasets [30] are chosen for the experiment. The KITTI datasets contain several different challenging benchmark data for multiple vision tasks, like stereo matching, object detection, scene segmentation, object tracking, as well as road/lane detection. The object detection dataset is used for training the proposed object detector and the object tracking dataset is adopted for evaluating and benchmarking the proposed tracking algorithm.

The CNN detector is constructed and trained based on one of the popular deep learning framework MXNet [17] developed by the Distributed (Deep) Machine Learning Community (DMLC). The parameters of the network were initialized using a pre-trained ResNet-101 model[1] trained on ImageNet [79] for image classification purposes. The model was trained for 10 epochs, that took approximately 8 hours on the aforementioned machine using two GPUs.

The detector generalized well on both KITTI's testing datasets and unseen datasets for the trained target types. Using KITTI's detection evaluation tool, the mean average precision (mAP) for 'Car' category by the proposed detector is 86.42%, 89.27%, 72.69% for 'Moderate', 'Easy', 'Hard', while the baseline method Faster R-CNN [78] is 82.11%, 88.70%, 71.19% respectively, showing the effectiveness of the improvements to the original detection algorithm by the proposed detector. However, the detector still has problems with missing some objects and false alarms as well, which remains to be solved by the proposed tracker.

[1] `https://github.com/tornadomeet/ResNet`

4.5.3 Multiple Target Tracking Performance Evaluation Methods

Similar to the single object tracking, benchmarking multiple target tracking algorithms require labeled object locations and IDs in the video sequences as ground truths. With the availability of these ground truth labels, different types of evaluation metric methodologies can be conducted.

Evaluation Methodology

To evaluate multiple aspects of a multi-target tracker's performance, the CLEAR MOT metrics [9] are widely used. The evaluation components in CLEAR MOT are summarized as follows, and each performance metric has an arrow with it indicates whether higher (↑) or lower (↓) score means better performance.

Multi-Object Tracking Accuracy (MOTA ↑). MOTA is an overall summary of the tracking accuracy in terms of false positves, false negatives and identity switches. MOTA is defined as

$$\begin{aligned} MOTA &= 1 - \frac{\sum_t (m_t + fp_t + mme_t)}{\sum_t g_t} \\ &= 1 - \left(\underbrace{\frac{\sum_t m_t}{\sum_t g_t}}_{\text{missing ratio}} + \underbrace{\frac{\sum_t fp_t}{\sum_t g_t}}_{\text{false alarm ratio}} + \underbrace{\frac{\sum_t mme_t}{\sum_t g_t}}_{\text{mismatch ratio}} \right), \end{aligned} \tag{4.9}$$

where m_t, fp_t , mme_t, and g_t are number of misses, false positives, mismatches, and total objects at time t respectively.

Multi-Object Tracking Precision (MOTP ↑). MOTP is an overall summary of the tracking precision in terms of bounding box overlap between the tracking result and the labeled ground truths. MOTP is defined as

$$MOTP = \frac{\sum_{i,t} d_{i,t}}{\sum_t c_t} \tag{4.10}$$

where $d_{i,t}$ is the bounding box overlap between ground truth position of object i and it's estimated location, and c_t is the number of matches found at time t.

Fragmentation (FM ↓). FM measures the number of frames the tracker is interrupted by a missing detection.

Identity switches (IDS ↓). IDS depicts how many times the identity of a ground truth track are changed.

Mostly Tracked (MT ↑). MT describes the percentage of the ground truth tracks that are tracked more than 80% of the track lifetime in length.

Mostly Lost (ML ↓). ML describes the percentage of the ground truth tracks that are tracked less than 20% of the track lifetime in length.

4.5.4 Tracking Datasets and Result Analysis

KITTI Object Tracking Evaluation Dataset The KITTI tracking datasets contains a 21 sequences of training data and a 29 sequences of testing data [30]. KITTI detection dataset are captured in urban scene with camera mounted on a moving vehicle. The dataset contains typical ground targets including car, van, truck,

tram, and also pedestrians and cyclists. As a moving platform, objects enter and exiting the scene very frequently, therefore the creating and deleting tracks is a very challenging tasks. For the data association, as the KITTI dataset contains only a small number of different categories of objects, a small set of hierarchical class tree is used. The car, van and truck classes share a same parent node. Pedestrian and cyclist categories share the same parent node. More complex hierarchical tree could be constructed when available labeled dataset is provided.

Since ground truths are only available in the training part of the object tracking section in the KITTI datasets. The proposed tracking algorithms is only evaluated on the 21 training image sequences. Sample vehicle tracking results are shown in Figure 4.4.

Figure 4.4: Sample tracking results on KITTI dataset sequence '0007', targets are marked with colored rectangles and with numbered ID (better view in color).

Result Analysis The evaluation is carried out on the 'car' category. Results are presented in Table 4.2. The results are compared with online published tracking algorithms that ranked top on the list on KITTI website only, including MDP [96], SCEA [41], CIWT [72]. The proposed algorithm performs good in terms of FM, IDS, MT, and ML, which indicates the effectiveness of the track management. However, the MOTA and MOTP are not as good as other state-of-art algorithms. Improving

Table 4.2: KITTI benchmark results.

Algorithms	MOTA(%)↑	MOTP(%)↓	FM↓	IDS↓	MT(%)↑	ML(%)↓
MDP	76.6	82.1	387	130	52.1	13.0
SCEA	75.6	79.4	448	104	53.1	11.5
CIWT	75.4	79.3	660	165	49.8	10.3
Ours	65.9	76.7	429	89	50.2	10.8

the precision and accuracy is one of the direction of future research.

The overall processing time of the proposed algorithm including detection and tracking is 11 FPS on average, however, comparing this with other tracking algorithms is hard due to different implementation and executing machine, and the proposed algorithm parallelized the detection and tracking. Therefore there is no easy way to only compare the tracking processing time with other algorithms.

4.5.5 Application: Monocular Camera based Vehicle Distance Estimation Using Proposed Tracking Algorithm - A Case Study

In the last part of the experiment section, an application of the proposed tracking algorithm for estimating vehicle distance based on a monocular camera is demonstrated.

Distance Estimation based on a Monocular Camera Assume the camera is mounted and fixed on the windshield of the ego-vehicle and is used to estimate the distance to the frontal target. Camera setup is described in Figure 4.5.

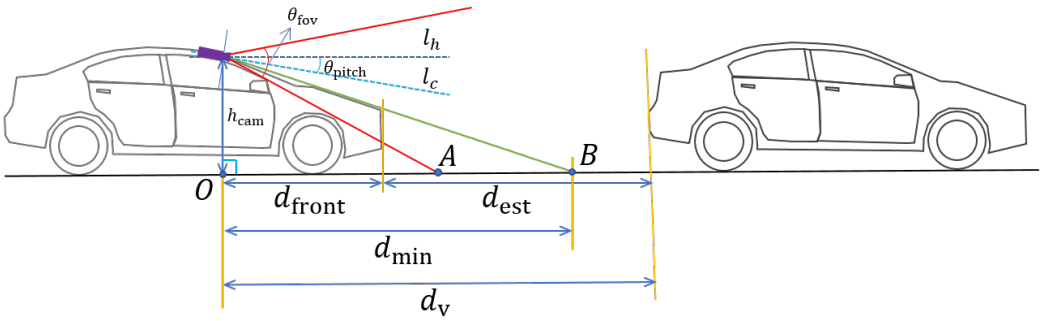


Figure 4.5: Camera setup for distance estimation.

Assuming a flat ground plane, the geometry behind the distance estimation based on a monocular camera is illustrated Figure 4.6.

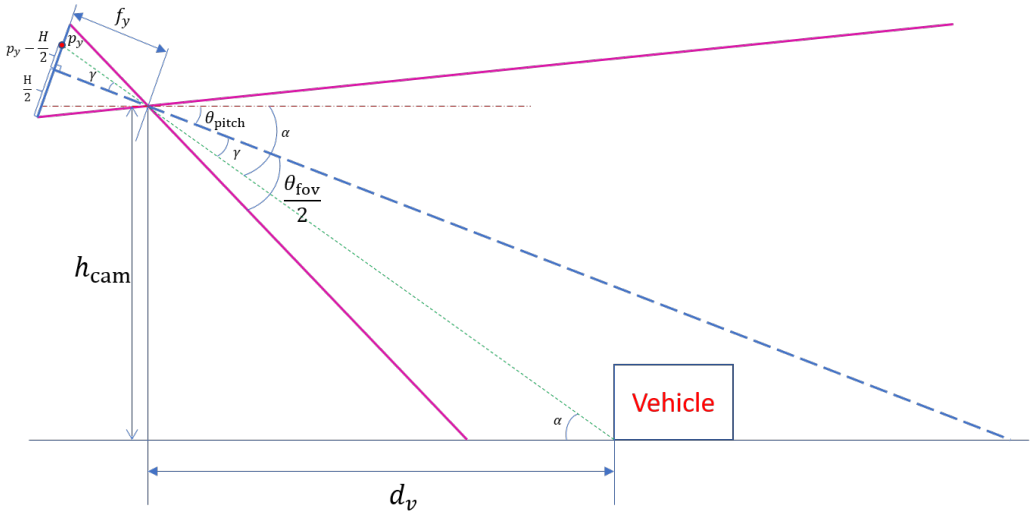


Figure 4.6: Illustration of the geometry for calculating distance from camera to other vehicles.

The parameters in Figure 4.6 depict,

- θ_{pitch}: camera pitch angle.
- θ_{fov}: camera vertical field of view (FOV).
- l_h: horizontal line.

- l_c: center line of FOV.
- d_{min}: minimum detectable range. The minimum detectable range is determined by $\max(d_{\text{OA}}, d_{\text{OB}})$, where A is the point of camera bottom FOV bound that touches the ground, and B is the touching point on the ground of extended line between principal point to bottom most pixel that is not blocked by the ego-vehicle hood.
- h_{cam}: camera height from ground to camera's principal point. This height is obtained from calibration.
- d_{front}: longitudinal distance from camera principal point to front bumper. This distance is obtained from calibration.
- d_{v}: longitudinal distance from camera principal point to the preceding vehicle rear bumper. This distance is estimated using camera measurements.
- d_{est}: longitudinal distance from ego-vehicle front bumper to the preceding vehicle rear bumper, this distance is estimated using,

$$d_{\text{est}} = d_{\text{v}} - d_{\text{front}}. \tag{4.11}$$

if $\theta_{\text{pitch}} < 0$, the longitudinal distance can be calculated as,

$$d_v = \frac{h_{\text{cam}}}{\tan \alpha}, \tag{4.12}$$

$$\alpha = \gamma - \theta_{\text{pitch}} \quad (\theta_{\text{pitch}} < 0), \tag{4.13}$$

and

$$\tan\gamma = \frac{p_y - \frac{H}{2}}{f_y}, \tag{4.14}$$

where p_y is the projected position in the image plane of the bottom part of the detected preceding vehicle, H is the camera height resolution, and f_y is the focal length in y axis.

Using the above equation, distance from camera to the frontal vehicle can be calculated using the bottom pixel position as,

$$d_v = \frac{h_{\text{cam}}}{\tan\left[\theta_{\text{pitch}} + \text{atan}\left(\frac{p_y - \frac{H}{2}}{f_y}\right)\right]} \tag{4.15}$$

Note that same formulation can be derived if $\theta_{\text{pitch}} \geq 0$. Using the distance calculated above, the longitudinal speed could be estimated using a linear Kalman filter assuming a constant velocity motion model.

Estimation Results The sample results of the estimated distance as well as the longitudinal speed are presented in Figure 4.7. The data was collected on road near McMaster University.

Figure 4.7: Sample result of estimated distance and longitudinal speed to frontal vehicles (better view in color).

To evaluate the performance of the above distance estimation algorithm, the estimated results are compared with the range generated from LiDAR sensors that provides relatively more accurate range information using the Oxford RobotCar dataset [65]. The stereo center camera and the LD-MRS front LiDAR data (sequence name '2015-08-20-12-00-47') is chosen. After registration of both camera and LiDAR, the ground truth range is generated using average distance of 3 laser reflection on the back side of the vehicle using the detection from the image to cluster the LiDAR measurements. On average the distance error is $0.8m$. The error would increase with distance, since the meter per pixel is growing with the distance.

The estimated range using the monocular camera supplies valuable information and could be directly applied for forward collision warning in ADAS or path planning in ADS. Further improvements of the estimation precision could be using the size and lane information to auto calibrate the camera and reject wrong estimates.

4.6 Conclusion

This chapter proposed an efficient multiple ground target tracking algorithm using a single camera in the urban scene. The algorithm adopts the CNN based object detector and improves the detection by using a perspective information. The data association is solved by a new hierarchical object class tree based multi-class association algorithm. Track management is enhanced with the help of accurate detector and a new track deletion algorithm is proposed by using a SVM based deletion prediction.

Experiment on public dataset shows the proposed algorithm outperforms existing real-time tracking algorithms. Also an example of applying the proposed algorithm for depth estimation of ground targets in urban driving scenario is studied, showing

the usefulness of the tracking algorithm.

Chapter 5

Conclusions and Future Research

5.1 Research Summary and Conclusions

In this thesis, multiple ground target tracking related topics are studied. Depending on the platforms, three different types of tracking algorithms are proposed according to the specialties of these platforms.

The dependent target tracking for ground targets is studied in Chapter 2. Traditional target tracking algorithms assume a state independence among targets, this will result in erroneous tracks for ground target that have dependent motion due to scenario constraints. The proposed algorithm models the states of the target jointly and derives the MRF based probabilistic data association filter. The state dependency is modeled using the driving behavior models that captures the specialty of ground target motion pattern and dependency among the targets. The experiments and simulations show the effectiveness of such a modeling that improves the predictions, and the proposed behavior model combined with MRF-PDA can reduce the false associations and improve performance evaluated using RMSE metric compared

with classical JPDA.

To track ground targets in the urban scene, camera is an effective and low-cost sensor choice that can record high detailed information about the targets. To build a multiple ground target tracking algorithm using a single camera, the single target tracking is firstly studied in Chapter 3. To achieve real-time processing on multiple targets, the speed of single target tracking is one of the significant part. With speed requirement in mind, the proposed algorithm adopts particle filtering with resizing templates to replace the commonly used exhaustive searching in different scales to localize the potential target candidates. To further increase the effectiveness of the limited number of particles, a mix-state proposal distribution is adopted. The robust appearance model is based on Ridge Regression that learns to discriminate the target and background. The appearance model is fast to train and easily update according to the track confidence. Extensive experiments are carried out with public benchmarking datasets. The results of the experiment show the proposed targets achieves good performance while beats other algorithms in speed with a large margin.

Despite the single target tracking algorithm developed in Chapter 3, the multiple target tracking algorithm still needs to solve the detection generation, data association and track management problems. A new tracking algorithm dealing with these problems are addressed in Chapter 4. Recent advances of CNN have drastically improved the real-time object detection tasks in images. The proposed algorithm combines the ResNet with RPN to train an object detector and generate the detections of interested ground targets. The detector is further improved by using perspective information in the urban scene. Since the detector generates not only the detections, but also the class information, a new multi-class data association algorithm using a hierarchical

object class tree is proposed. The high accuracy CNN detectors help improve track management by using the detection confidence and a new SVM based track deletion is also proposed to correctly remove the dead tracks.

5.2 Future Research Direction and Discussion

Some future work is born from the current research topics, and there are still space to improve the presented work to deploy them into real world systems. Also adapting to a real world system has many requirements and limitations that is different from the experimental setups, such as processing time limits, computational resource constraints, and even power consumption restrictions. Several improvements could be made to expand the proposed research:

Fusing with More Sensors

In all three works, only single sensor model was considered. To improve the accuracy, stability, coverage, and robustness, multiple sensors could be employed. Potential sensors to be fused could be cameras, stereo cameras, mid-range radars and LiDARs. Comparing to using single sensor for vehicle tracking, fusion of multiple sensors would increase the performance by combining different source of information. Fusion algorithms introduces new problems to be handled, such as sensor registration, sensor bias estimation, and fusion algorithm design. These problems need to be solved in order to achieve the benefits of fusing multiple sensors.

Optimizing for Embedded Systems

In real world applications, such automotive, video surveillance and drone navigation, many platforms are implemented on embedded systems. Such systems usually have limited resources for computing and restricted power supply. Comparing to a regular PC machine, the CPU/RAM/GPU of embedded systems are far less powerful in terms of clock speed, speed-up intrinsics, cache/memory size, bus speed and computational cores, and the power consumption may also be limited. These resource limits would slow the proposed algorithms down and eventually reduce the speed as well as accuracy, making the algorithms not deployable.

Trade-offs needs to be made for optimizing the usage of the proposed algorithms in embedded systems. Potential improvements would be developing special hardware speedup units for some modules, such as CNN based detection algorithms. In the algorithm side, designing new detection methods would significantly reduce the computational load.

Application of Generated Tracks

Application of generated tracks is the next steps of ground target tracking algorithm. Tracks themselves contains useful information like speed and location that can be utilized immediately, however, more useful information can be extracted based on the tracks. And applications of these tracks for other tasks could be studied.

ADAS and ADS In the automotive area, the output of extracted tracks can be utilized to help implement forward collision warning, perform adaptive cruise control, and avoid crash in ADAS and ADS.

Intelligent Transportation System and Traffic Control Using traffic surveillance cameras or large area surveillance platforms on helicopters or drones, the proposed algorithms could be applied to automatically generate traffic reports and jam alerts, traffic predictions, and intelligent traffic controls.

Robotics and Its Navigation Vision based navigation in both indoor and outdoor scenarios requires correct tracking and prediction of surrounding obstacle targets. The generated tracks from the camera mounted on robots helps path planning, obstacle avoidance and navigation.

Bibliography

[1] Ahmed, K., Ben-Akiva, M., Koutsopoulos, H., and Mishalani, R. (1996). Models of freeway lane changing and gap acceptance behavior. In *Internaional Symposium on Transportation and Traffic Theory*, pages 501–515.

[2] Andrieu, C., Doucet, A., and Punskaya, E. (2001). Sequential Monte Carlo methods for optimal filtering. In *Sequential Monte Carlo Methods in Practice*, pages 79–95. Springer.

[3] Babenko, B., Yang, M.-H., and Belongie, S. (2011). Robust object tracking with online multiple instance learning. *IEEE transactions on pattern analysis and machine intelligence*, **33**(8), 1619–1632.

[4] Bar-Shalom, Y. and Li, X.-R. (1995). *Multitarget-multisensor tracking: principles and techniques*, volume 19. YBs London, UK:.

[5] Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. (2004). *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, New York, NY, USA.

[6] Bar-Shalom, Y., Daum, F., and Huang, J. (2009). The probabilistic data association filter. *IEEE Control Systems Magazine*, **6**(29), 82–100.

[7] Bar-Shalom, Y., Willett, P. K., and Tian, X. (2011). *Tracking and Data Fusion*. YBS Publishing, Storrs, CT.

[8] Benameur, K., Pannetier, B., and Nimier, V. (2005). A comparative study on the use of road network information in GMTI tracking. In *8th International Conference on Information Fusion*, volume 1.

[9] Bernardin, K. and Stiefelhagen, R. (2008). Evaluating multiple object tracking performance: The CLEAR MOT metrics. *EURASIP Journal on Image and Video Processing*, **2008**(1), 246309.

[10] Bertinetto, L., Valmadre, J., Golodetz, S., Miksik, O., and Torr, P. H. (2016). Staple: Complementary learners for real-time tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1401–1409.

[11] Blom, H. and Bloem, E. (2000). Probabilistic data association avoiding track coalescence. *IEEE Transactions on Automatic Control*, **45**(2), 247–259.

[12] Blom, H. A. and Bloem, E. A. (2002). Interacting multiple model joint probabilistic data association avoiding track coalescence. In *Proceedings of the 41st IEEE Conference on Decision and Control*, volume 3, pages 3408–3415.

[13] Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **23**(11), 1222–1239.

[14] Čehovin, L., Leonardis, A., and Kristan, M. (2016). Visual object tracking performance measures revisited. *IEEE Transactions on Image Processing*, **25**(3), 1261–1274.

[15] Chang, D.-C. and Fan, M.-W. (2014). Interacting multiple model particle filtering using new particle resampling algorithm. In *Global Communications Conference (GLOBECOM), 2014 IEEE*, pages 3215–3219. IEEE.

[16] Chen, L., Wainwright, M. J., Cetin, M., and Willsky, A. S. (2006). Data association based on optimization in graphical models with application to sensor networks. *Mathematical and Computer Modelling*, **43**(9), 1114–1135.

[17] Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. (2015). MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*.

[18] Cheng, H.-Y. and Hsu, S.-H. (2011). Intelligent highway traffic surveillance with self-diagnosis abilities. *IEEE Transactions on Intelligent Transportation Systems*, **12**(4), 1462–1472.

[19] Crouse, D. F. (2016). On implementing 2D rectangular assignment algorithms. *IEEE Transactions on Aerospace and Electronic Systems*, **52**(4), 1679–1696.

[20] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE.

[21] Deb, S., Yeddanapudi, M., Pattipati, K., and Bar-Shalom, Y. (1997). A generalized SD assignment algorithm for multisensor-multitarget state estimation. *IEEE Transactions on Aerospace and Electronic Systems*, **33**(2), 523–538.

[22] Dinh, T. B., Vo, N., and Medioni, G. (2011). Context tracker: Exploring supporters and distracters in unconstrained environments. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1177–1184. IEEE.

[23] Ehlert, P. A. and Rothkrantz, L. J. (2001). Microscopic traffic simulation with reactive driving agents. In *Proceedings of 2001 IEEE Intelligent Transportation Systems*, pages 860–865.

[24] El Faouzi, N.-E., Leung, H., and Kurian, A. (2011). Data fusion in intelligent transportation systems: Progress and challenges–A survey. *Information Fusion*, **12**(1), 4–10.

[25] Felzenszwalb, P. F., Girshick, R. B., McAllester, D., and Ramanan, D. (2010). Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, **32**(9), 1627–1645.

[26] Fitzgerald, R. J. (1986). Development of practical PDA logic for multitarget tracking by microprocessor. In *American Control Conference*, pages 889–898, Seattle, WA, USA.

[27] Fleuret, F., Berclaz, J., Lengagne, R., and Fua, P. (2008). Multicamera people tracking with a probabilistic occupancy map. *IEEE transactions on pattern analysis and machine intelligence*, **30**(2), 267–282.

[28] Frank, O. and Strauss, D. (1986). Markov graphs. *Journal of the American Statistical Association*, **81**(395), 832–842.

[29] Galoogahi, H. K., Fagg, A., Huang, C., Ramanan, D., and Lucey, S. (2017).

Need for speed: A benchmark for higher frame rate object tracking. *arXiv preprint arXiv:1703.05884*.

[30] Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[31] Girshick, R. (2015). Fast R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448.

[32] Grabner, H., Grabner, M., and Bischof, H. (2006). Real-time tracking via on-line boosting. In *Bmvc*, volume 1, page 6.

[33] Grabner, H., Leistner, C., and Bischof, H. (2008). Semi-supervised on-line boosting for robust tracking. *Computer Vision–ECCV 2008*, pages 234–247.

[34] Gu, B. and Hong, L. (2001). Tracking 2-D rigid targets with invariant constraints. *Information Sciences*, **138**(1), 79–97.

[35] Hamdar, S. (2012). Driver behavior modeling. In *Handbook of Intelligent Vehicles*, pages 537–558. Springer London.

[36] Hare, S., Golodetz, S., Saffari, A., Vineet, V., Cheng, M.-M., Hicks, S. L., and Torr, P. H. (2016). Struck: Structured output tracking with kernels. *IEEE transactions on pattern analysis and machine intelligence*, **38**(10), 2096–2109.

[37] He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

[38] Helbing, D., Hennecke, A., Shvetsov, V., and Treiber, M. (2002). Micro- and macro-simulation of freeway traffic. *Mathematical and Computer Modelling*, **35**(56), 517 – 547.

[39] Henriques, J. F., Caseiro, R., Martins, P., and Batista, J. (2012). Exploiting the circulant structure of tracking-by-detection with kernels. In *European conference on computer vision*, pages 702–715. Springer.

[40] Henriques, J. F., Caseiro, R., Martins, P., and Batista, J. (2015). High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **37**(3), 583–596.

[41] Hong Yoon, J., Lee, C.-R., Yang, M.-H., and Yoon, K.-J. (2016). Online multi-object tracking via structural constraint event aggregation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1392–1400.

[42] Isard, M. and Blake, A. (1998a). Condensation–Conditional density propagation for visual tracking. *International Journal of Computer Vision Publ*, pages 5–28.

[43] Isard, M. and Blake, A. (1998b). A mixed-state condensation tracker with automatic model-switching. In *Computer Vision, 1998. Sixth International Conference on*, pages 107–112. IEEE.

[44] Janai, J., Güney, F., Behl, A., and Geiger, A. (2017). Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art. *arXiv preprint arXiv:1704.05519*.

[45] Jia, X., Lu, H., and Yang, M.-H. (2012). Visual tracking via adaptive structural local sparse appearance model. In *Computer vision and pattern recognition (CVPR), 2012 IEEE Conference on*, pages 1822–1829. IEEE.

[46] Kalal, Z., Mikolajczyk, K., and Matas, J. (2012). Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, **34**(7), 1409–1422.

[47] Khan, Z., Balch, T., and Dellaert, F. (2003). Efficient particle filter-based tracking of multiple interacting targets using an MRF-based motion model. In *Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems. (IROS 2003).*, volume 1, pages 254–259 vol.1.

[48] Khan, Z., Balch, T., and Dellaert, F. (2005). MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **27**(11), 1805–1819.

[49] Kiani Galoogahi, H., Sim, T., and Lucey, S. (2015). Correlation filters with limited boundaries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4630–4638.

[50] Kirubarajan, T., Bar-Shalom, Y., Pattipati, K., and Kadar, I. (2000). Ground target tracking with variable structure IMM estimator. *IEEE Transactions on Aerospace and Electronic Systems*, **36**(1), 26–46.

[51] Koller, D. and Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT press, Cambridge, MA, USA.

[52] Krajzewicz, D., Erdmann, J., Behrisch, M., and Bieker, L. (2012). Recent development and applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, **5**(3&4), 128–138.

[53] Kreucher, C., Hero, A., and Kastella, K. (2004). Multiple model particle filtering for multitarget tracking. In *the twelfth annual workshop on adaptive sensor array processing. Lexington, MA*.

[54] Kumar, S. and Hebert, M. (2003). Discriminative random fields: A discriminative framework for contextual interaction in classification. In *Proceedings of the Ninth IEEE International Conference on Computer Vision*, pages 1150–1157.

[55] Kwon, J. and Lee, K. M. (2010). Visual tracking decomposition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1269–1276. IEEE.

[56] Kwon, J. and Lee, K. M. (2011). Tracking by sampling trackers. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1195–1202. IEEE.

[57] Lee, D.-T. and Schachter, B. J. (1980). Two algorithms for constructing a Delaunay triangulation. *International Journal of Computer & Information Sciences*, **9**(3), 219–242.

[58] Lezama, J., Randall, G., and von Gioi, R. G. (2017). Vanishing point detection in urban scenes using point alignments. *Image Processing On Line*, **7**, 131–164.

[59] Li, S. Z. (2009). *Markov Random Field Modeling in Image Analysis*. Springer, New York, NY, USA.

[60] Li, X. and Jilkov, V. (2005). Survey of maneuvering target tracking, Part V: Multiple-model methods. *IEEE Transactions on Aerospace and Electronic Systems*, **41**(4), 1255–1321.

[61] Liangqun, L., Hongbing, J., and Xinbo, G. (2006). Maximum entropy fuzzy clustering with application to real-time target tracking. *Signal processing*, **86**(11), 3432–3447.

[62] Liao, S., Shen, D., and Chung, A. C. (2014). A Markov Random Field groupwise registration framework for face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **36**(4), 657–669.

[63] Liu, B., Huang, J., Kulikowski, C., and Yang, L. (2013). Robust visual tracking using local sparse appearance model and k-selection. *IEEE transactions on pattern analysis and machine intelligence*, **35**(12), 2968–2981.

[64] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). SSD: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer.

[65] Maddern, W., Pascoe, G., Linegar, C., and Newman, P. (2017). 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, **36**(1), 3–15.

[66] Mazor, E., Averbuch, A., Bar-Shalom, Y., and Dayan, J. (1998). Interacting multiple model methods in target tracking: A survey. *IEEE Transactions on Aerospace and Electronic Systems*, **34**(1), 103–123.

[67] Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT press, Cambridge, MA, USA.

[68] Musicki, D. and Evans, R. (2004). Joint integrated probabilistic data association: JIPDA. *IEEE Transactions on Aerospace and Electronic Systems*, **40**(3), 1093–1099.

[69] Ojala, T., Pietikainen, M., and Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, **24**(7), 971–987.

[70] OpenStreetMap (2014). `http://www.openstreetmap.org/`.

[71] Oron, S., Bar-Hillel, A., Levi, D., and Avidan, S. (2015). Locally orderless tracking. *International Journal of Computer Vision*, **111**(2), 213–228.

[72] Osep, A., Mehner, W., Mathias, M., and Leibe, B. (2017). Combined image-and world-space tracking in traffic scenes. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1988–1995. IEEE.

[73] Pattipati, K., Deb, S., Bar-Shalom, Y., and Washburn, R. B. (1992). A new relaxation algorithm and passive sensor data association. *IEEE Transactions on Automatic Control*, **37**(2), 198–213.

[74] Pérez, P., Hue, C., Vermaak, J., and Gangnet, M. (2002). Color-based probabilistic tracking. *Computer visionECCV 2002*, pages 661–675.

[75] Popp, R. L., Pattipati, K. R., and Bar-Shalom, Y. (2001). m-best SD assignment algorithm with application to multitarget tracking. *IEEE Transactions on Aerospace and Electronic Systems*, **37**(1), 22–39.

[76] Redmon, J. and Farhadi, A. (2016). YOLO9000: better, faster, stronger. *arXiv preprint arXiv:1612.08242*.

[77] Reid, D. (1979). An algorithm for tracking multiple targets. *IEEE transactions on Automatic Control*, **24**(6), 843–854.

[78] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

[79] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, **115**(3), 211–252.

[80] Saleemi, I. and Shah, M. (2013). Multiframe many–many point correspondence for vehicle tracking in high density wide area aerial videos. *International Journal of Computer Vision*, **104**(2), 198–219.

[81] Sevilla-Lara, L. and Learned-Miller, E. (2012). Distribution fields for tracking. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1910–1917. IEEE.

[82] Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[83] Singh, A. and Jagersand, M. (2016). Modular tracking framework: A unified approach to registration based tracking. *arXiv preprint arXiv:1602.09130*.

[84] Sinha, A., Ding, Z., Kirubarajan, T., and Farooq, M. (2012). Track quality based multitarget tracking approach for global nearest-neighbor association. *IEEE Transactions on Aerospace and Electronic Systems*, **48**(2), 1179–1191.

[85] Tharmarasa, R., Kirubarajan, T., Hernandez, M., and Sinha, A. (2007). PCRLB-based multisensor array management for multitarget tracking. *IEEE Transactions on Aerospace and Electronic Systems*, **43**(2), 539–555.

[86] Tichavsky, P., Muravchik, C. H., and Nehorai, A. (1998). Posterior Cramér-Rao bounds for discrete-time nonlinear filtering. *IEEE Transactions on Signal Processing*, **46**(5), 1386–1396.

[87] Treiber, M., Hennecke, A., and Helbing, D. (2000). Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, **62**(2), 1805.

[88] Tso, B. C. and Mather, P. M. (1999). Classification of multisource remote sensing imagery using a genetic algorithm and Markov Random Rields. *IEEE Transactions on Geoscience and Remote Sensing*, **37**(3), 1255–1260.

[89] Ulmke, M. and Koch, W. (2006). Road-map assisted ground moving target tracking. *IEEE Transactions on Aerospace and Electronic Systems*, **42**(4), 1264–1274.

[90] Veach, E. and Guibas, L. J. (1995). Optimally combining sampling techniques for Monte Carlo rendering. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 419–428. ACM.

[91] Viola, P. and Jones, M. J. (2004). Robust real-time face detection. *International journal of computer vision*, **57**(2), 137–154.

[92] Wang, M., Liu, Y., and Huang, Z. (2017). Large margin object tracking with circulant feature maps. *arXiv preprint arXiv:1703.05020*.

[93] Wang, Z., Zhang, H., and Ray, N. (2009). Tracking of multiple interacting objects using a novel prediction model. In *16th IEEE International Conference on Image Processing (ICIP)*, pages 869–872, Cairo.

[94] Wu, Y., Lim, J., and Yang, M.-H. (2013). Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2411–2418.

[95] Wu, Y., Lim, J., and Yang, M.-H. (2015). Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **37**(9), 1834–1848.

[96] Xiang, Y., Alahi, A., and Savarese, S. (2015). Learning to track: Online multi-object tracking by decision making. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4705–4713.

[97] Yang, Q. and Koutsopoulos, H. N. (1996). A microscopic traffic simulator for evaluation of dynamic traffic management systems. *Transportation Research Part C: Emerging Technologies*, **4**(3), 113–129.

[98] Zhai, Y., Yeary, M. B., Cheng, S., and Kehtarnavaz, N. (2009). An object-tracking algorithm based on multiple-model particle filtering with state partitioning. *IEEE Transactions on instrumentation and measurement*, **58**(5), 1797–1809.

[99] Zhang, T., Ghanem, B., Liu, S., and Ahuja, N. (2012). Robust visual tracking via multi-task sparse learning. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2042–2049. IEEE.

[100] Zhong, W., Lu, H., and Yang, M.-H. (2014). Robust object tracking via sparse collaborative appearance model. *IEEE Transactions on Image Processing*, **23**(5), 2356–2368.