



**SINGLE TO MULTIPLE TARGET,
MULTIPLE TYPE VISUAL
TRACKING**

submitted by

Nathanael Lemessa Baisa

for the degree of

Doctor of Philosophy

of the

Department of Electrical, Electronic and Computer Engineering
Heriot-Watt University

October, 2017

This copy of the thesis has been supplied on the condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the prior written consent of the author.

ABSTRACT

Visual tracking is a key task in applications such as intelligent surveillance, human-computer interaction (HCI), human-robot interaction (HRI), augmented reality (AR), driver assistance systems, and medical applications. In this thesis, we make three main novel contributions for target tracking in video sequences.

First, we develop a long-term model-free single target tracking by learning discriminative correlation filters and an online classifier that can track a target of interest in both sparse and crowded scenes. In this case, we learn two different correlation filters, translation and scale correlation filters, using different visual features. We also include a re-detection module that can re-initialize the tracker in case of tracking failures due to long-term occlusions.

Second, a multiple target, multiple type filtering algorithm is developed using Random Finite Set (RFS) theory. In particular, we extend the standard Probability Hypothesis Density (PHD) filter for multiple type of targets, each with distinct detection properties, to develop multiple target, multiple type filtering, N-type PHD filter, where $N \geq 2$, for handling confusions that can occur among target types at the measurements level. This method takes into account not only background false positives (clutter), but also confusions between target detections, which are in general different in character from background clutter. Then, under the assumptions of Gaussianity and linearity, we extend Gaussian mixture (GM) implementation of the standard PHD filter for the proposed N-type PHD filter termed as N-type GM-PHD filter.

Third, we apply this N-type GM-PHD filter to real video sequences by integrating object detectors' information into this filter for two scenarios. In the first scenario, a tri-GM-PHD filter is applied to real video sequences containing three types of multiple targets in the same scene, two football teams and a referee, using separate but confused detections. In the second scenario, we use a dual GM-PHD filter for tracking pedestrians and vehicles in the same scene handling their detectors' confusions. For both cases, Munkres's variant of the Hungarian assignment algorithm is used to associate tracked target identities between frames.

We make extensive evaluations of these developed algorithms and find out that our methods outperform their corresponding state-of-the-art approaches by a large margin.

Dedicated to all my family, especially to my mom!

ACKNOWLEDGMENT

First and foremost, I would like to thank GOD for providing me the chance to pursue my dreams of following this PhD program. Next my deepest gratitude goes to UK Engineering and Physical Sciences Research Council (EPSRC), James Watt Scholarship and all members of the Heriot-Watt University for giving me the opportunity to have these wonderful PhD program in my life. A very special gratitude goes to my supervisor, Prof. Andrew Wallace, who has supported me throughout my PhD program. I appreciate his technical support, and continuous follow up of my progress in designing, implementing as well as writing up my PhD dissertation. I also would like to thank Dr. Daniel Clark for his valuable suggestions and technical support in area of the technique I am using in this dissertation. I must also acknowledge Dr. Deepayan Bhowmik for his ideas and comments he provided me to adapt to the work environment and technical support. My appreciation also goes to all computer vision lab members of Heriot-Watt University for showing their hospitable behaviour of welcoming international students.

Lastly but not least, I would like to thank my family without their love and long support, I would not be here to fulfil my dream. In particular, I must acknowledge my mother, Dashi Merera, without her love and encouragement, I would not have achieved my dreams.

Nathanael Lemessa Baisa
October, 2017



ACADEMIC REGISTRY Research Thesis Submission

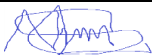
Name:	Nathanael Lemessa Baisa		
School:	School of Engineering and Physical Sciences		
Version: <i>(i.e. First, Resubmission, Final)</i>	Final	Degree Sought:	Doctor of Philosophy (PhD) Electrical Engineering

Declaration

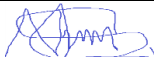
In accordance with the appropriate regulations I hereby submit my thesis and I declare that:

- 1) the thesis embodies the results of my own work and has been composed by myself
- 2) where appropriate, I have made acknowledgement of the work of others and have made reference to work carried out in collaboration with other persons
- 3) the thesis is the correct version of the thesis for submission and is the same version as any electronic versions submitted*.
- 4) my thesis for the award referred to, deposited in the Heriot-Watt University Library, should be made available for loan or photocopying and be available via the Institutional Repository, subject to such conditions as the Librarian may require
- 5) I understand that as a student of the University I am required to abide by the Regulations of the University and to conform to its discipline.
- 6) I confirm that the thesis has been verified against plagiarism via an approved plagiarism detection application e.g. Turnitin.

* Please note that it is the responsibility of the candidate to ensure that the correct version of the thesis is submitted.

Signature of Candidate:		Date:	
-------------------------	---	-------	--

Submission

Submitted By <i>(name in capitals)</i> :	NATHANAEL LEMESSA BAISA
Signature of Individual Submitting:	
Date Submitted:	

For Completion in the Student Service Centre (SSC)

Received in the SSC by <i>(name in capitals)</i> :			
Method of Submission <i>(Handed in to SSC; posted through internal/external mail):</i>			
E-thesis Submitted <i>(mandatory for final theses)</i>			
Signature:		Date:	

Please note this form should be bound into the submitted thesis.
Academic Registry/Version (1) August 2016

Contents

Abstract	ix
Acknowledgement	ix
List of Figures	ix
List of Tables	xiv
List of Publications	xv
Acronyms and Notations	xvii
1 Introduction	1
1.1 Thesis Objectives	2
1.2 Contributions	3
1.3 Thesis Outlines	5
2 Literature Review	7
2.1 Important Steps for Visual Tracking	7
2.1.1 Object Representation	8

2.1.2	Visual Features Selection	10
2.1.2.1	Traditional Visual Features	11
2.1.2.2	Convolutional Neural Networks (CNNs / ConvNets)	13
2.1.3	Object detection	15
2.1.3.1	Object Detection Approaches	16
2.1.3.2	Exemplars of Object Detection Algorithms	17
2.2	Types of Visual Tracking	18
2.3	Challenges in Visual Tracking	20
2.3.1	Coping with appearance changes	20
2.3.2	Handling occlusions	21
2.4	Model-free Tracking	22
2.5	Bayesian Tracking	25
2.5.1	Single Target Tracking	26
2.5.1.1	Kalman Filter	27
2.5.1.2	Particle filter	28
2.5.2	Data Association-based Multi-Target Tracking	30
2.5.2.1	Data Association Algorithms	31
2.5.2.2	Exemplars of Traditional Multi-Target Trackers	34
2.5.3	Multi-Target Tracking using Random Finite Sets	37
2.6	Evaluation Metrics for Visual Tracking	43

2.6.1	Evaluation Metrics for Single Target Tracking	44
2.6.2	Evaluation Metrics for Multi-target Tracking	45
3	Long-term Correlation Tracking using Multi-layer Hybrid Features	48
3.1	Overview of our Algorithm	49
3.2	Proposed Algorithm	50
3.2.1	Correlation Filters for Multi-layer Features	50
3.2.2	Online Detector	54
3.2.3	Temporal Filtering using the GM-PHD Filter	56
3.2.4	Scale Estimation	60
3.3	Implementation Details	62
3.4	Experimental Results	63
3.4.1	Evaluation on OOTB	64
3.4.2	Evaluation on PETS 2009 Data Sets	66
3.5	Summary	71
4	Development of a N-type GM-PHD Filter for Multiple Target, Multiple Type Filtering	74
4.1	Multiple Target, Multiple Type Recursive Bayes Filtering with RFS . . .	75
4.2	Probability Generating Functional (PGFL)	77
4.3	N-type PHD Filtering Strategy	81
4.4	Gaussian Mixture-Based N-type PHD Filter Implementation	83

4.5	Experimental Results	86
4.6	Summary	97
5	Multiple Target, Multiple Type Visual Tracking using a N-type GM- PHD Filter	101
5.1	Multiple Target, Implicit Multiple Type Tracking using a Tri-GM-PHD Filter	102
5.1.1	Object Detection, Training and Evaluation	102
5.1.2	Tracking Football Teams and Referee	104
5.1.3	Data Association	105
5.1.4	Experimental Results	105
5.2	Multiple Target, Explicit Multiple Type Tracking using a Dual GM-PHD Filter	114
5.2.1	Object Detection	116
5.2.1.1	3D orientation	116
5.2.1.2	Visual features	117
5.2.1.3	Pedestrian Detection	118
5.2.1.4	Vehicle Detection	118
5.2.1.5	Detection Parameters Extraction	119
5.2.2	Tracking Pedestrians and Vehicles, and Data Association	119
5.2.3	Experimental Results	120
5.3	Evaluation and Comparison using the MOT Benchmarking Tool	122

5.4 Summary	127
6 Conclusions and Future Work	130
6.1 Conclusions	130
6.2 Future Work	132
7 APPENDIX A	134
Bibliography	137
Index	156

List of Figures

2.1	Examples of target representations: (a) a sample target (rectangular red bounding box) and corresponding (b) colour template and (c) colour histogram (Y-axis is the number of pixels in the image at each intensity value along X-axis) and (d) HOG feature visualization.	10
2.2	One-stream Alex Net [1]. The network's input is $227 \times 227 \times 3 = 154, 587$ -dimensional, and the number of neurons in the network's remaining layers is given by $290,400 - 186,624 - 64,896 - 64,896 - 43,264 - 4096 - 4096 - 1000$	14
2.3	Data association problem: two validated measurements (green boxes) are located in the validation region (grey area) centred on the predicted measurement (red triangle) [2].	31
2.4	Single target to multi-target extension using RFS theory. The black arrows show the extension of single target tracking in which states and observations are represented using random vectors to multi-target tracking though FISST framework in which states and observations are represented using random finite sets.	39
2.5	FISST framework.	39
2.6	Probability hypothesis densities (PHDs) in two consecutive time steps [3].	39
2.7	RFS-based multi-target filters in the FISST framework.	41
3.1	The flowchart of the proposed algorithm. It consists of three main parts: translation estimation, re-detection and scale estimation.	51

3.2	VGG-Net 19 [4].	63
3.3	Distance precision (left) and overlap success (right) plots on OOTB using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better.	65
3.4	Distance precision (left) and overlap success (right) plots on OOTB using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better. The performances of our algorithm (LCMHT) and its version without a re-detection module (LCMHT_WtR) are shown.	65
3.5	Success plots on OOTB using one-pass evaluation (OPE) for 8 challenge attributes: occlusion, scale variation, background clutter, illumination variation, in-plane rotation, out-of-plane rotation, deformation, and fast motion. The legend contains the AUC score of each tracker; the larger, the better.	67
3.6	Qualitative results of our proposed LCMHT algorithm, CF2 [5], MEEM [6], LCT [7] and KCF [8] on some challenging sequences of OOTB (Fleetface, Singer1, Freeman4, and Walking2 from left column to right column, respectively).	68
3.7	Distance precision (left) and overlap success (right) plots on PETS data sets using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better.	69
3.8	Distance precision (left) and overlap success (right) plots on PETS data sets using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better. The performances of our algorithm (LCMHT) and its version without a re-detection module (LCMHT_WtR) are shown.	70
3.9	Qualitative results of our proposed algorithm LCMHT, CF2 [5], MEEM [6], LCT [7] and KCF [8] on PETS 2009 medium density (left column) and dense (right column) data sets.	72

3.10	Qualitative results of our proposed LCMHT algorithm, CF2 [5], MEEM [6], LCT [7] and KCF [8] on PETS 2009 medium density (left, frame 78) and dense (right, frame 85) data sets, just after occlusion by cropping and enlarging. . . .	73
4.1	A sensor setting for our approach. The sensor is equipped with many detectors for each type of multiple targets in which case one detector can also detect other than its own target type resulting in separate but confused measurements which are filtered and discriminated by our approach.	75
4.2	Confusions between four target types (T1, T2, T3 and T4) at the detection stage from detectors 1, 2, 3 and 4.	92
4.3	Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from four independent GM-PHD filters (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.3$	94
4.4	Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from quad GM-PHD filter (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.3$	95
4.5	Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from four independent GM-PHD filters (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.6$	96
4.6	Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from quad GM-PHD filter (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.6$	97

4.7	Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from four independent GM-PHD filters (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.9$	98
4.8	Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from quad GM-PHD filter (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.9$	99
4.9	Cardinality and OSPA error: Ground truth (red for cardinality only), quad GM-PHD filter (green), four independent GM-PHD filters (blue) for $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.6$	100
4.10	OSPA error comparison for quad GM-PHD filter and four independent GM-PHD filters at different probabilities of confusion.	100
5.1	Extracting detection probabilities for three target types (red, white and referee) from ROCs of 3 detectors: red team detector, white team detector and referee detector when tested on red team, white team and referee instances, respectively.	103
5.2	Results of detections, three independent GM-PHD trackers and tri-GM-PHD tracker, respectively, for frame 25.	109
5.3	Results of detections, three independent GM-PHD trackers and tri-GM-PHD tracker, respectively, for frame 57.	110
5.4	Results of detections, three independent GM-PHD trackers and tri-GM-PHD tracker, respectively, for frame 73.	111
5.5	Cardinality and OSPA error: Ground truth (red for cardinality only), tri-GM-PHD filter (green), three independent GM-PHD filters (blue), detections (magenta).	112
5.6	Tracking the red and white teams, and referee from frame 193 to frame 293 . . .	112

5.7	Results of tri-GM-PHD tracker for different values of the confusion parameters, for frame 57, corresponding to detections of Fig. 5.3a.	115
5.8	Quantizing 3D geometric orientation into L labels for obtaining required aspect ratios; for each of the aspect ratio, a specific detector is developed.	116
5.9	Clustering of CNN features.	117
5.10	ROCs using 3D orientation and CNN visual features detector models tested on KITTI sequence 16.	120
5.11	Results of detections, two independent GM-PHD trackers and dual GM-PHD tracker, respectively, for frame 13.	122
5.12	Results of detections, two independent GM-PHD trackers and dual GM-PHD tracker, respectively, for frame 23.	123
5.13	Cardinality and OSPA error: Ground truth (red for cardinality only), dual GM-PHD filter (green), two independent GM-PHD filters (blue), detections (magenta).	124
5.14	Sample results on several sequences of MOT16 datasets, bounding boxes represents the tracking results with their color-coded identities. From left to right: MOT16-01, MOT16-03 (top row), MOT16-06, MOT16-08 (middle row),and MOT16-12, MOT16-14 (bottom row).	127
5.15	Sample results on the sequence MOT16-07, bounding boxes represents the tracking results with their label for their identities, for frames 368, 380 and 396 from top to bottom.	129

List of Tables

2.1	Survey of sample tracking algorithms.	37
3.1	Implementation parameters, Ps for parameters and Vs for values.	63
4.1	OSPA error at different values of probabilities of confusion $p_{12,D}$, $p_{13,D}$, $p_{14,D}$, $p_{21,D}$, $p_{23,D}$, $p_{24,D}$, $p_{31,D}$, $p_{32,D}$, $p_{34,D}$, $p_{41,D}$, $p_{42,D}$ and $p_{43,D}$ (0.3, 0.6 and 0.9) for quad GM-PHD filter and 4 independent GM-PHD filters. Time taken is given in brackets.	96
5.1	Frame-averaged cardinality and OSPA errors, time taken and discrimination rate at the extracted detection probabilities for tri-GM-PHD filter, three independent GM-PHD filters and Detections.	108
5.2	Frame-averaged cardinality and OSPA errors, time taken and discrimination rate at the extracted detection probabilities for dual GM-PHD filter, two independent GM-PHD filters and Detections.	124
5.3	Tracking performance of representative trackers developed using both online and offline methods. All trackers are evaluated on the test dataset of the MOT16 [9] benchmark using public detections. The first and second highest values are highlighted by bold and underline, respectively.	126

List of Publications

Scientific publications produced during this PhD are listed below by categorizing into journals, and conferences and posters.

Journals:

[1]. N. L. Baisa and A. Wallace, "Development of a N-type GM-PHD Filter for Multiple Target, Multiple Type Visual Tracking," IEEE Transactions on Pattern Analysis and Machine Intelligence (under review) and also published on arxiv.org.

[2]. N. L. Baisa, D. Bhowmik, and A. Wallace, "Long-term Correlation Tracking using Multi-layer Hybrid Features in Sparse and Dense Environments," Journal of Visual Communication and Image Representation (under review) and also published on arxiv.org.

Conferences and others:

[3]. N. L. Baisa and A. Wallace, "Multiple Target, Multiple Type Visual Tracking using a Tri-GM-PHD Filter," in Proceedings of the 12th International Conference on Computer Vision Theory and Applications (VISAPP), VISIGRAPP, 2017.

[4]. N. L. Baisa, D. Bhowmik, and A. Wallace, "Long-term Correlation Tracking using Multi-layer Hybrid Features in Dense Environments," in Proceedings of the 12th International Conference on Computer Vision Theory and Applications (VISAPP), VISIGRAPP, 2017.

[5]. N. L. Baisa and A. Wallace, "Multiple Target, Multiple Type Filtering in RFS Framework," Published on arxiv.org, 2017.

[6]. N. L. Baisa, D. Clark, and A. Wallace, "Multi-target Visual Tracking using Random Finite Set-Based Filters," Poster on International Computer Vision Summer School (ICVSS), 2014.

Acronyms

Acronym	Description
HOG	Histogram of Oriented Gradients
SIFT	Scale-Invariant Feature Transform
SURF	Speeded Up Robust Features
KLT	Kanade-Lucas-Tomasi
ACF	Aggregated Channel Features
CNN	Convolutional Neural Networks
EKF	Extended Kalman filter
UKF	Unscented Kalman filter
GNN	Global Nearest Neighbour
JPDAF	Joint Probabilistic Data Association Filter
MHT	Multiple Hypothesis Tracking
PMHT	Probabilistic Multiple Hypothesis Tracking
HMM	Hidden Markov Models
RFS	Random Finite Set
FISST	FInite Set STatistics
PGFL	Probability Generating Functional
PHD	Probability Hypothesis Density
CPHD	Cardinalized Probability Hypothesis Density
GM-PHD	Gaussian Mixture Probability Hypothesis Density
SMC-PHD	Sequential Monte Carlo Probability Hypothesis Density
MeMber	Multi-target Multi-Bernoulli
CB-MeMber	Cardinality Balanced-MeMber
SIS	Sequential Importance Sampling
MCMC	Markov Chain Monte Carlo
RJ-MCMC	Reversible Jump MCMC
MCMCDA	Markov Chain Monte Carlo Data Association
MRF	Markov Random Field
CRF	Conditional Random Field
SVM	Support Vector Machines
LCMHT	Long-term Correlation Multi-layer Hybrid Tracker
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
IFFT	Inverse Fast Fourier Transform
OSPA	Optimal Subpattern Assignment
MOTA	Multi-Object Tracking Accuracy
MOTP	Multi-Object Tracking Precision

Notations

Notation	Description
$\Phi(\mathbf{x})$	Mapping to non-linear feature space (kernel space)
$k(\mathbf{x}, \hat{\mathbf{x}})$	Kernel function
$y_{k k-1}(x_k x_{k-1})$	Single-target transition density
$f_k(z_k x_k)$	Single-target likelihood
$p_{k-1 k-1}(x_{k-1} z_{1:k-1})$	Single-target updated (posterior) density at time k-1
$p_{k k-1}(x_k z_{1:k-1})$	Single-target predicted (prior) density at time k
$p_{k k}(x_k z_{1:k})$	Single-target updated (posterior) density at time k
\mathcal{X}	State space
$\mathcal{F}(\mathcal{X})$	Collection of all the finite subsets of \mathcal{X}
\mathcal{Z}	Measurement space
$\mathcal{F}(\mathcal{Z})$	Collection of all the finite subsets of \mathcal{Z}
$X_{i,k}$	Multi-target state for target type i at time k
$Z_{i,k}$	Multi-target measurement for target type i at time k
$y_{i,k k-1}(x \zeta)$	Single-target transition density at time k given the previous state ζ of target type i
$f_{ji,k}(z x)$	Single-target likelihood for target type $i \in \{1, \dots, N\}$ from detector $j \in \{1, \dots, N\}$
$y_{i,k k-1}(X_{i,k} X_{i,k-1})$	Multi-target transition density of target type i
$f_{ji,k}(Z_{j,k} X_{i,k})$	Multi-target likelihood for target type $i \in \{1, \dots, N\}$ from detector $j \in \{1, \dots, N\}$
$p_{i,k-1 k-1}(X_{i,k-1} Z_{i,1:k-1})$	Multi-target posterior density for target type i at time k-1
$p_{i,k k-1}(X_{i,k} Z_{i,1:k-1})$	Multi-target predicted density for target type i at time k
$p_{i,k k}(X_{i,k} Z_{i,1:k})$	Multi-target posterior density for target type i at time k
$\Xi_{i,k}$	RFS associated with the multi-target state of target type i
$S_{i,k}(X_{i,k-1})$	RFS of survived targets of target type i at time k from the previous set of targets $X_{i,k-1}$
$\Gamma_{i,k}$	RFS of the new-born targets of target type i
$\gamma_{i,k}(x)$	Intensity of the new-born targets RFS $\Gamma_{i,k}$
$\Omega_{i,k}$	RFS associated with the multi-target measurement of target type i
$\Theta_{i,k}(X_{i,k})$	RFS modeling the measurement generated by the targets $X_{i,k}$
$C_{s_{i,k}}$	RFS associated with background clutter for target type i
$c_{s_{i,k}}$	PHD of the clutter RFS $C_{s_{i,k}}$ at time k
$C_{t_{i,J,k}}$	RFS associated with all target types $J = \{1, \dots, N\} \setminus i$ treated as clutter, termed as confusion
$c_{t_{i,k}}$	PHD of the clutter RFS $C_{t_{i,J,k}}$ at time k

Notations

Notation	Description
$p_{i,S,k k-1}(\zeta)$	Probability of survival for target type i at time k
$p_{ii,D}(x)$	Probability of detection for target type i by detector i
$p_{ij,D}(x)$	Confusion detection probability for target type i by detector j
$F_i[g, h]$	Joint probability generating functional (PGFL) for target type i
$G_{T_i}(h)$	Prior PGFL for target type i
$G_{L_{i,j}}(g x)$	Bernoulli detection process for each target of target type i using detector j
$G_{c_i}(g)$	Poisson PGFL of background clutter for target type i
$G_i(h z_1, \dots, z_m)$	Updated PGFL for target type i
$\mathcal{D}_i(\cdot z_1, \dots, z_m)$	Updated PHD for target type i
$\mathcal{D}_{\Xi_i}(x)$	PHD of a RFS Ξ_i associated with target type i
$\mathcal{D}_{i,k-1 k-1}(x)$	Updated PHD for target type i at time $k-1$
$\mathcal{D}_{i,k k-1}(x)$	Predicted PHD for target type i at time k
$\mathcal{D}_{i,k k}(x)$	Updated PHD for target type i at time k

Chapter 1

Introduction

Visual tracking is an active research field in computer vision which has got many applications such as intelligent surveillance, human-computer interaction (HCI), human-robot interaction (HRI), augmented reality (AR), medical applications, visual vehicle navigation, visual servoing, motion-based recognition, video indexing, etc. Recently, there has been a great deal of interest in robust visual tracking algorithms due to the increased need for automated video analysis in the computer vision community. This video analysis has three crucial steps: detection of interesting objects, tracking of these objects in each video frame, and analysis of object tracks to recognize their behaviour. The advantage of using video information is that it is cheap to acquire when compared to radar and lidar, for example.

There are two important things to consider when developing a visual tracker: detection of objects in each frame and associating the detections corresponding to the same object over time. A tracker is not only expected to assign consistent labels to the tracked targets in each video frame to generate a trajectory for each target but also, depending on the tracking problem, it can give object specific information such as the area (size), velocity, shape, or orientation of targets. However, there are many difficulties in visual target¹ tracking such as abrupt changes in target motion, changing appearance patterns of both the target and the scene, target-to-target and target-to-scene occlusions, non-rigid target structures, noise in the image and a cluttered background, and camera motion. Moreover, the uncertain noise-corrupted nature of detections (observations or measurements) are also a great challenge in estimating the number of targets and their positions.

¹Note that the terms *target* and *object of interest* are used interchangeably in this document

In some tracking tasks, making some justifiable assumptions is important to simplify the tracking problem as well as to improve the tracking performance. For example, prior knowledge of object appearance, shape, or motion (e.g. constant velocity, constant acceleration) can help to simplify the tracking problems.

There are many tracking approaches described in the literatures. The key differences [10] [11] between all of these tracking approaches lies in four important things: 1) a suitable object representation (motion, shape and appearance) and the way it is modeled, 2) appropriate and robust image features (histogram of oriented gradients, color, convolutional neural networks, etc), 3) detection or search strategy (off-line or online classifier, matching, etc), and 4) complexity (linear, polynomial, exponential, etc). These key things for visual tracking problems are determined depending on the context or environment, and the the end use of that tracking algorithm.

There are some requirements [12] that are crucial when developing a visual tracking algorithm: robustness, adaptivity to object appearance changes, and real-time processing. Robustness measures the ability of the tracker to track the target of interest even under complicated conditions such as clutter, occlusions, changing illuminations or complex object motion. Adaptivity is mostly concerned with the changes that the object itself undergoes. A system that engages in dealing with live video streams need to have high processing speed which relies on the speed of the target under observation. Generally, a frame-rate of at least 25 frames per second (fps) must be set up in order to attain a smooth output video for human eyes' feeling.

1.1 Thesis Objectives

The main purpose of visual tracking, either single or multiple target, is to determine the positions and velocities (and possibly additional information such as identity and size) in either the image plane or more commonly in a 3D world space. Accordingly, this thesis has two main but related goals.

The first goal is to develop a long-term online visual tracking algorithm that can track an unknown target in both sparse and crowded scenes where the unknown target is initialized by a bounding box in the first frame and then is tracked in subsequent frames. Previously, the tracking of a target of interest in sparse [13] and crowded [14] scenes were treated separately since, in the latter case, it is very challenging due to heavy occlusions, high target densities and cluttered scene, and significant appearance

variations of targets. Therefore, we are interested, in this thesis, in developing a generic long-term model-free tracking algorithm that can be applied to both scenarios. Thus, without taking any constraint on the video scene, we want to develop a long-term online tracking algorithm that can close the research gap between sparse and crowded environment tracking problems.

The second goal is to develop a multiple target, multiple type filtering algorithm using the recently popular Random Finite Set (RFS) theory that can handle the measurement confusions that may occur among target types, and then apply that algorithm to real video sequences. Practically, there are many situations where tracking and discrimination of multiple target types is essential to handle confusions between target types. For example, like many others, situational awareness for driver assistance and vehicle autonomy has been studied [15], in which a vehicle equipped with a sensor suite must detect and track other road users to select the best sensor focus and course of action. In this example, the most numerous objects of interest in urban environments are other road users such as cars, pedestrians and bicycles. In this particular and many other examples, confusion between target types is common, for example a standard pedestrian detection strategy [16] often provides confused detections between pedestrians and cyclists, and even small cars. Moreover, for sports analysis we often want to track and discriminate sub-groups of the same target type such as players in opposing teams [17]. Thus, these types of problems motivate our work to develop a multiple target, multiple type filtering methodology handling target confusions in which a single sensor (e.g. smart camera either static or moving such as camera mounted on a vehicle, Kinect on a robot, or surveillance unmanned aerial vehicle (UAV) also known as a drone) has $N \geq 2$ different detection modes, each with its own probability of detection and a measurement density for N different target types. One main difference between tracking targets in videos from static and moving cameras is that some object detection algorithms which work for a static camera might not work for a moving camera, background subtraction for example.

1.2 Contributions

The contributions of this thesis are 1) developing a long-term model-free single target tracking algorithm, 2) modeling and implementation of a N -type PHD filter for filtering multiple target of different types, and 3) integrating object detectors of different target types in the same scene into the N -type GM-PHD filter to apply to visual tracking on real video sequences.

Long-term model-free Target Tracking

We propose a novel long-term visual tracking algorithm by learning discriminative correlation filters and an online classifier for tracking a target of interest in both sparse and crowded scenes. First, we learn a translation correlation filter using multi-layer hybrid features i.e. features extracted from multiple layers of convolutional neural networks (CNNs) trained on a large amount of object recognition data set (ImageNet) [18] and traditional hand-crafted features, particularly histogram of oriented gradients (HOG) and color-naming. We combine the advantages of both the lower convolutional layer which retains more spatial details for precise localization and the higher convolutional layer which encodes semantic information for handling appearance variations, and then integrate these with HOG and color-naming features. Second, we include a re-detection module for overcoming tracking failures due to long-term occlusions. In this case, we train an incremental (online) support vector machine (SVM) on the most confident frames using traditional features (HOG, LUV color and normalized gradient magnitude). This re-detection module is activated only when the correlation response of the object is below some pre-defined threshold. This generates high score detection proposals which are temporally filtered using a Gaussian mixture probability hypothesis density (GM-PHD) filter for removing clutter. The Gaussian component (detection proposal) with the maximum weight is selected as the state (position) estimate which re-fines the object location when a re-detection module is activated removing the others as clutter. Finally, we train a scale correlation filter for estimating the scale of a target by constructing a target pyramid around the estimated or re-detected position using HOG features. We call this a Long-term Correlation Multi-layer Hybrid Tracker (LCMHT). We make extensive experiments both on a large-scale online object tracking benchmark (OOTB) and on tracking an interesting target in crowded scenes which show our method performs favorably against existing state-of-the-art methods.

Modeling and Implementation of a N-type PHD Filter

A Multiple Target, Multiple Type Filtering (MTMTF) algorithm is developed using RFS theory. First, we extend the standard Probability Hypothesis Density (PHD) filter for multiple types of targets, each with distinct detection properties, to develop a multiple target, multiple type filtering, N-type PHD filter, where $N \geq 2$, for handling confusion. In this approach, we assume that there are confusions between detections, i.e. clutter arises not just from background false positives, but also from target confusion. Then, under the assumptions of Gaussianity and linearity, we extend the Gaussian mixture (GM) implementation of the standard PHD filter to develop a N-type GM-PHD filter. In addition, we analyze the results from simulations to track sixteen targets

of four different types using four-type GM-PHD (quad GM-PHD) filter and compare it with four independent GM-PHD filters using Optimal Subpattern Assignment (OSPA) metric. This shows the improved performance of our strategy that accounts for target confusions by efficiently discriminating them.

Multiple Target, Multiple Type Visual Tracking using a N-type GM-PHD Filter

We integrate the object detectors' information such as probabilities of detections for each target type and confusion detection probabilities among target types at a specific clutter rate into the N-type GM-PHD filter to apply to visual tracking on real video sequences. We investigate two scenarios of real video sequences. In the first case which we consider as implicit multiple type tracking, we apply a tri-GM-PHD filter to real video sequences containing three types of multiple targets in the same scene, two football teams and a referee, using separate but confused detections. In the second case, we use a dual GM-PHD filter for tracking pedestrians and vehicles in the same scene handling their detectors' confusions which is considered as explicit multiple type tracking. For both cases, Munkres's variant of the Hungarian assignment algorithm is used to associate tracked target identities between frames. The trackers on these two scenarios are evaluated and compared to both raw detection and independent GM-PHD filters using the OSPA metric and discrimination rate. This shows the improved performance of our strategy on real video sequences.

1.3 Thesis Outlines

The remainder of this PhD thesis is organised as follows:

Chapter 2 presents the background theory for visual tracking algorithms in general as well as the different kinds of important visual tracking algorithms available in the literature.

Chapter 3 discusses long-term single target tracking which is developed by learning discriminative correlation filters, an online classifier, and using the GM-PHD filter for both sparse and crowded scenes.

Chapter 4 presents the modeling and derivation of our new N-type PHD filter to filter and discriminate multiple target of different types handling their confusions that may occur at the measurement stage. This is a development of the very popular RFS approach. Its implementation scheme and simulation analysis are also discussed in detail.

Chapter 5 demonstrates the application of the developed N-type GM-PHD filter to real video sequences. Two scenarios are considered: tracking of football teams and a referee using a tri-GM-PHD filter, and tracking of pedestrians and vehicles using a dual GM-PHD filter.

Chapter 6 closes the thesis with conclusions of the proposed methods with an outlook on future work.

Chapter 2

Literature Review

In this chapter, we cover some of the important theoretical background used for developing visual tracking algorithms. These visual tracking algorithms can be designed using different approaches such as model-free or model-based, single or multiple targets, static or moving cameras, overlapping or non-overlapping networks of cameras, etc. We review some of the exemplar state-of-the-art tracking algorithms available in the computer vision research community. Moreover, available evaluation metrics for both single and multi-target visual tracking problems are highlighted. Accordingly, important steps for developing visual tracking algorithms are given in section 2.1, different kinds of visual tracking algorithms are briefly described in section 2.2, the challenges visual trackers face are presented in section 2.3, model-free and Bayesian tracking approaches are discussed in sections 2.4 and 2.5, respectively, and evaluation metrics for visual tracking algorithms are given in section 2.6.

2.1 Important Steps for Visual Tracking

There are key steps that should be considered when developing a visual tracker [10] [12]. The first step is to define appropriate object shape, appearance and motion representations. The second step is to select robust image features that can be given as input to the tracking algorithm. Obviously, almost all tracking algorithms require object detection that might be developed using a specific image feature or a combination of image features that can help to detect objects of interest either in the first frame or in every frame of a video sequence. Therefore, depending on the object shapes, object appearances, number of targets (single or multiple), object and camera motions (static

or moving platforms), and camera network (overlapping or non-overlapping), a suitable and robust visual tracking algorithm can be developed.

2.1.1 Object Representation

It is crucial to first represent targets using shape, appearance and motion models before performing visual tracking [10] [2]. Each of these target representation models are discussed below in detail, and in some cases a combination of them are described jointly as appropriate.

Object Shape Representation: Small objects in an image are commonly represented using a point (e.g. centroid) or set of points [19]. Primitive geometric shapes such as a rectangle, ellipse, cuboid and ellipsoid are more appropriate for representing simple rigid objects though they can also be used for non-rigid objects in visual tracking. For example, a sample target is represented using a rectangle in Fig. 2.1a. However, complex non-rigid objects such as the human body need to be represented using contour (boundary of an object) and silhouette (region inside the contour). If targets are composed of connected body parts e.g. human body (torso, legs, hands, head, and feet), articulated shape models are used to represent those targets by modeling the constituent parts using cylinders or ellipses. The kinematic motion models such as joint angle can govern the relationship between these parts. Object skeletons which can be extracted from an object silhouette using a medial axis transform are also used as a shape representation mainly for object recognition as well as to model both articulated and rigid objects.

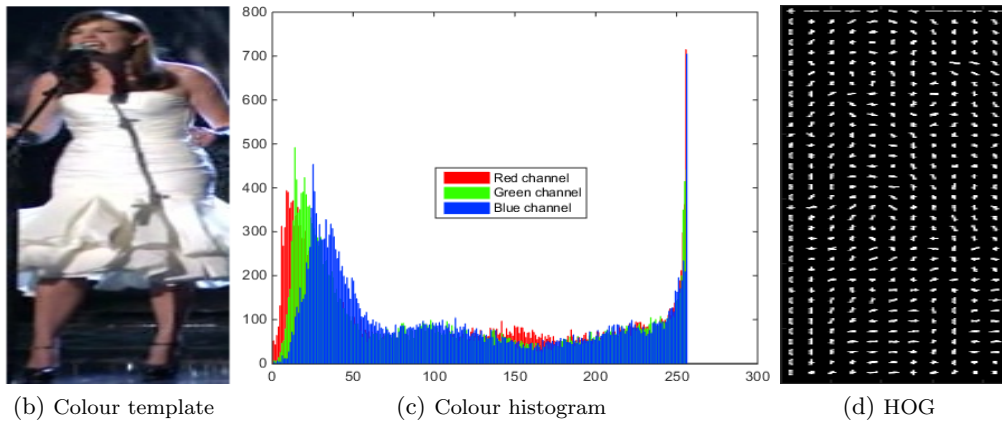
Object Appearance Representation: There are common representations of object appearances in the context of visual tracking: templates, probability density of object appearance, active appearance models and multiview appearance models. Templates are formed using simple geometric shapes or silhouettes and are used for both recognition and tracking of objects whose poses do not change significantly during tracking processes. For instance, a colour template is represented in Fig. 2.1b. One advantage of a template is that it can capture both spatial and appearance information though it has the drawback of handling only a single view of the object appearance. After defining the object shape models, the probability densities of object appearance features such as colour, texture and gradients can be computed from the interior region of these shape models. Basically, the probability density estimates of object appearance can either be parametric such as Gaussian and mixtures of Gaussians or nonparametric such as Parzen windows and histograms (e.g. colour, orientation). A colour histogram where

all the red, green and blue channels are plotted together and HOG feature are shown in Fig. 2.1c and Fig. 2.1d respectively which correspond to the colour template shown in Fig. 2.1b. Histograms lack spatial information that makes it difficult to distinguish targets with similar colour or gradient distribution which can be alleviated by introducing spatial information associated with each bin of the histogram (spatiograms) [20]. Active appearance models are generated by simultaneously modelling the object shape and appearance statistically [21]. In this case, the object shape is represented by a set of landmarks which can reside on the object boundary or inside the object region where an appearance feature vector is stored for each landmark, and then both shape and its associated appearance are learned from a set of samples using, for example, Principal Component Analysis (PCA). Obviously, shape and appearance representations can be combined for improving visual tracking tasks [22] [23]. Multiview appearance models encode different views of an object, and there are two main approaches to do so. The first approach is to generate a subspace from the given views e.g. PCA and Independent Component Analysis (ICA) have been used for both shape and appearance representation in [24]. The second approach is to train a set of classifiers such as SVM [25] or Bayesian networks to learn the different views of an object, however, it has a limitation i.e. the appearances in all views are needed in advance.

Object Motion Representation: The motion of a target over time can be described by a motion model which is used for predicting the likely state of the target between consecutive frames reducing computational complexity [26]. The widely used motion models are constant velocity, constant acceleration, coordinated turn [27] and random walk [26] [28]. When the model properties match the motion characteristics of the target, the motion model can increase the performance of the tracker. For example, when a target moves behind occlusions, motion models such as constant velocity, constant acceleration and coordinated turn can increase the robustness of the tracking algorithm. However, if targets have no predefined motion pattern, the random walk model is the most widely used motion model. While an adaptive velocity motion model with adaptive noise variance is introduced in [28], an observation data driven motion model (exploitation of explicit visual motion measurements in the proposal distribution) is used for overcoming abrupt motion changes of a target in [29]. Generally, the motion models are designed depending on the motion characteristics of targets as well as the adopted shape (state) representation of targets such as rectangle, affine etc taking into account changes in the scale, rotation and deformation properties of the targets. The motion of objects represented using shapes, primitive geometric shapes for instance, need to be modeled by translation, affine, or projective (homography) transformation. Though trackers such as mean-shift [30] and the Kanade-Lucas-Tomasi (KLT)



(a) A sample target



(b) Colour template

(c) Colour histogram

(d) HOG

Figure 2.1: Examples of target representations: (a) a sample target (rectangular red bounding box) and corresponding (b) colour template and (c) colour histogram (Y-axis is the number of pixels in the image at each intensity value along X-axis) and (d) HOG feature visualization.

tracker [31] are designed free of a constrained motion model, they implicitly embed the motion model in their search strategies. Thus, motion models are important ingredients of visual tracking algorithms.

2.1.2 Visual Features Selection

Tracking algorithms use a single or a combination of visual features that can help to uniquely distinguish objects in the feature space. Selecting appropriate features for the tracking problem depends on the object representation. For example, while colour is the appropriate feature for histogram-based appearance representations, edges are a suitable feature for contour-based shape representations. Any visual descriptor can be adapted for visual tracking. Some of the recent widely used visual descriptors for visual detection and tracking are given in [12]. Generally, features to be extracted

from images for target representation can be categorized into three types: low-level (e.g. colour, gradient, motion), mid-level (e.g. edges, corners, regions) and high-level (e.g. (detected) objects) features [2]. The most widely used visual features in the visual detection and/or tracking problems are illustrated below by classification into traditional and CNN features.

2.1.2.1 Traditional Visual Features

Here we present some of the traditional hand-engineered visual features which are crucial in computer vision tasks, particularly in visual tracking problems.

Colour: There are many colour spaces in image processing: RGB (Red, Green, Blue), HSV (Hue, Saturation, Value), Luv, Lab, etc [32], [33]. RGB colour space is not perceptually uniform i.e. the differences in colors in RGB space do not correspond to the color differences perceived by humans. In contrast, while Luv and Lab color spaces are perceptually uniform, HSV color space is only approximately uniform. Color-naming, the linguistic color label assigned by human to describe the color, is also becoming popular [34]. However, there are some physical factors which can influence the color of objects: the spectral distribution of the illuminance, the surface reflectance properties of the object, self-shading, contribution from the colour of surrounding large objects and surfaces, etc. Moreover, these color spaces themselves are sensitive to noise though each of them can be used in visual tracking depending on the scenario. There are two known color descriptors in computer vision: histogram-based color descriptors and SIFT-based color descriptors. Performance evaluation of color descriptors is given in [35].

Edges: Tracking algorithms that aim to track the boundary of objects use edge features which are generated by capturing sudden intensity changes around object boundaries (usually a binary map), and are less sensitive to illumination when compared to color features. Evaluation and comparison of edge detection algorithms are given in [36] where the popular Canny edge detector still has competitive performance due to its accuracy and simplicity.

Texture: Texture is a measure of intensity change of an image which quantifies properties such as smoothness and regularity, and it requires a processing step to generate the descriptors unlike color. These features are less sensitive to illumination changes like edge features when compared to color. There are various texture descriptors: Gray-Level Co-occurrence Matrices (GLCM) which are 2D histogram showing the co-

occurrences of intensities in a specified direction and distance, Gabor wavelet [37] which can be considered as orientation and scale-tunable edge and line detectors, Law's texture measures which are twenty-five 2D filters generated from 1D filters corresponding to level, edge, spot, wave and ripple, and Local Binary Patterns (LBP) [38] and its variants which are very efficient texture descriptors due to their gray scale invariance, tolerance against illumination changes and computational simplicity. These texture features are also used for detection and recognition in addition to visual tracking.

Gradient: Gradient features capture directional change in intensity or color (usually approximated by convolving an image with a kernel such as the Sobel operator or Pre-witt operator), and there are many descriptors which are developed using the statistical summarization of the gradients such as Scale-Invariant Feature Transform (SIFT) [39] for object recognition, Speeded Up Robust Features (SURF) [40] which is a much faster scale and rotation invariant interest point descriptor, Histogram of Oriented Gradients (HOG) descriptor for pedestrian detection [41], etc. Canny edge detector also uses image gradient for edge detection.

Spatio-temporal: The local space-time features capture characteristic salient as well as motion patterns in video, and are a popular representation for action recognition and visual detection [42]. They capture representation of events which are comparatively independent with respect to their spatio-temporal shifts and scales. For example, optical flow defines the translation of each pixel in a region and is used as a feature for developing motion-based segmentation and tracking algorithms. The Histogram of Oriented gradients (HOG) and optical flow (HOF) i.e. HOG/HOF descriptor accumulated in space-time neighbourhoods of detected interest points can characterise the local motion and appearance for learning human actions in movies [43].

Multiple feature fusion: Feature fusion is becoming more important in image and video retrieval, visual tracking, and object detection as it achieves better performance than using only a single kind of feature. For example, the HOG-LBP descriptor for pedestrian detection can handle partial occlusion [44]. HOG in combination with background subtraction is used in [45] for pedestrian detection which meets a real-time demand with high accuracy. The aggregated Channel Features (ACF) detector developed in [16] uses 3 kinds of features in 10 channels: normalized gradient magnitude (1 channel), histogram of oriented gradients (6 channels) and LUV color (3 channels) for pedestrian detection. The PHD-filter-based visual tracker developed in [46] uses colour histograms (CH), HOG and covariance matrices for better observation model. Our work in Chapters 3 and 5 use multiple feature fusion in some parts.

2.1.2.2 Convolutional Neural Networks (CNNs / ConvNets)

Though traditional hand-engineered features are still important, deep learning features have recently demonstrated outstanding results on various recognition tasks. There are various deep learning architectures [47] for feature learning of which the most successful in computer vision is CNN [48]. There are many widely used CNN implementations such as CAFFE [49], MatConvNet [50], Keras (with TensorFlow or Theano backends), etc. CNN gives state-of-the-art results on visual recognition tasks [1] [4] [51], generic object detection [52] and/or specific object detection [53] [54]. Two-dimensional CNN is extended to 3D CNN to extract features from both the spatial and temporal dimensions using 3D convolutions on a stack of frames for action recognition in [55].

The neurons in a ConvNet layer are connected to only a small local region of the layer before it, instead of all of the neurons in a fully-connected manner as in a regular Neural Nets, reducing the number of parameters (the weights and biases of the neurons) by a large margin. The spatial extent of this connectivity is called the receptive field of the neuron (equivalently this is the filter size). Moreover, unlike a regular Neural Network, the layers of a ConvNet have neurons arranged in 3 dimensions (width, height, depth), taking advantage of the fact that the input consists of images (or others such as a speech signal). It is important to emphasize that the connections are local in space (along width and height), but always full along the entire depth of the input volume. Each ConvNet layer can have convolution, nonlinearity Rectified Linear Units (ReLUs, $f(x) = \max(0, x)$), pooling (sub-sampling), and normalization building blocks. A typical ConvNet is composed of one or more locally-connected convolutional layers (often with the other steps such as ReLU, sub-sampling and normalization), followed by one or more fully-connected layers. In practice, it is common to zero pad the border during convolution to preserve size spatially from shrinkage. Thus, the one-stream AlexNet shown in Fig. 2.2 has 8 layers with weights (the first five convolutional layers and the remaining three fully-connected layers). In this AlexNet example, the ReLU non-linearity is applied to the output of every convolutional and fully-connected layer. Max-pooling follows the first, second and fifth convolutional layers (after ReLU), and response-normalization follows the first and second convolutional layers after max-pooling (but normalization is not used any more due to its minimal contribution). The output of the last fully-connected layer is fed to a 1000-way softmax loss function which produces a distribution over the 1000 class labels.

All neurons in each depth column are connected to the same region of the input, but of course with different weights i.e. all neurons in a single depth slice (column) are sharing the same parameters, sets of weights referred as a filter (or a kernel), hence the name

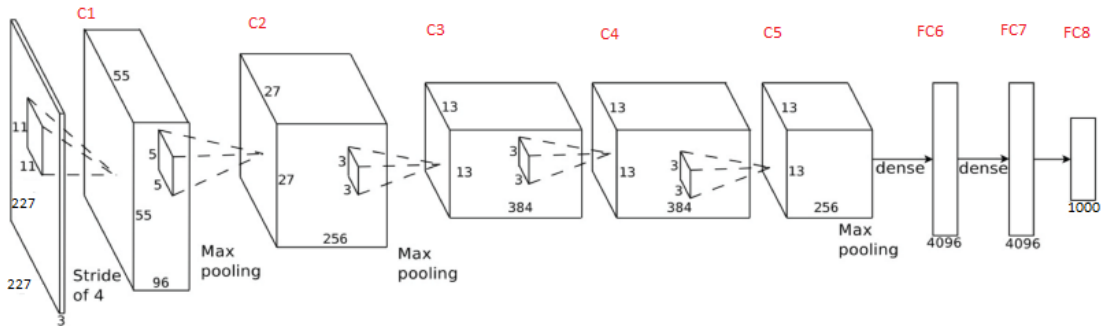


Figure 2.2: One-stream Alex Net [1]. The network’s input is $227 \times 227 \times 3 = 154,587$ -dimensional, and the number of neurons in the network’s remaining layers is given by $290,400 - 186,624 - 64,896 - 64,896 - 43,264 - 4096 - 4096 - 1000$.

parameter sharing. The number of kernels or filters (number of feature maps needed) is user-defined. It is crucial to notice that some layers contain parameters (CONV and FC) and others do not (ReLU and POOL). Similarly, some Layers have one or more additional hyperparameters such as the number of filters \mathbf{K} , their spatial extent \mathbf{F} , the stride \mathbf{S} , and the amount of zero padding \mathbf{P} (for instance, CONV/FC/POOL do, RELU does not). Given these hyperparameters, the spatial size of the output volume can be computed as a function of the input volume size. Parameter sharing and pooling operation contribute to the translation invariance of the ConvNet whereas ReLU increases the nonlinear properties of the decision function and of the overall ConvNet. Obviously, the earlier features of a ConvNet contain more generic low-level features such as edge detectors and color blob detectors whereas later layers of the ConvNet become progressively more specific to the details of the classes (high-level features).

ConvNet trends are towards smaller filters (narrower) and deeper architectures as deeper networks do better, and towards getting rid of POOL/FC layers (just CONV) for not losing information (e.g. due to pooling, though pooling progressively reduces the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting) [4] [51].

Data augmentation and a regularizer method called dropout are the main techniques to overcome overfitting [1]. Dropout, setting the output of each hidden neuron to zero with probability 0.5, is used, for instance, in the first two fully-connected layers of Fig. 2.2.

Supervised training of ConvNet is performed by back-propagating classification error.

Generally, the training process of ConvNet can be summarized as follows:

1. Initialize all filters and parameters (weights) with normal random values.
2. Provide a centered (mean subtracted) training image as input to the network which goes through the forward propagation step, and find the output probabilities for each class.
3. Calculate the total error at the output layer (summation over all classes).
4. Use back-propagation to calculate the gradients of the error with respect to all weights in the network and use gradient descent to update all filter values (weights) to minimize the output error.
5. Repeat steps 2-4 with all images in the training set.

Due to lack of a sufficient size of dataset, very few people train an entire ConvNet from scratch (with random initialization). Instead, it is common to pretrain a ConvNet on a very large dataset such as ImageNet [18], and then use the ConvNet for transfer learning: as a fixed feature extractor [56] or for fine-tuning (as an initialization) [52]. In the first case, the last fully-connected layer (whose outputs are the class scores) is first removed, then the rest of the ConvNet is treated as a fixed feature extractor for the new dataset. It is important for performance that these extracted features are ReLUd (i.e. thresholded at zero) but before pooling. In the second case, the classifier on top of the ConvNet is replaced and retrained on the new dataset as well as the weights of the pre-trained network is fine-tuned by continuing the backpropagation. Though it is possible to fine-tune all the layers of the ConvNet, it is sometimes crucial to keep some of the earlier layers fixed (due to overfitting concerns) and only fine-tune some higher-level portion of the network. Transfer learning is successfully used in our work in Chapters 3 and 5.

2.1.3 Object detection

Directly or indirectly, every tracking algorithm needs an object detection algorithm either in each frame or when the object first becomes visible in the video. While some object detection methods use a single frame information, others use temporal information which is computed by frame differencing to highlight changing regions in consecutive frames which in turn reduces the number of false detections. Object detection can be generic such as Pascal VOC [57] or specific such as pedestrian [16], vehicles [58], or bicycles.

2.1.3.1 Object Detection Approaches

Some of the object detection methods in the context of visual tracking are given below.

Point detectors: Point detectors are used to locate keypoints in images, and are used for solving problems in tasks such as motion, stereo, and tracking. Invariance to changes in illumination, camera viewpoint, rotation and scale are some of the desirable qualities of interest points. Some of the widely used interest point detectors are the Moravec point detector, Harris point detector, KLT (Kanade-Lucas-Tomasi) detector [31], SIFT detector [39], and SURF detector [40]. Pedestrian detector was developed using (local) interest points (keypoints) in [59]. A review of interest point detectors is given in [60].

Background subtraction: This algorithm builds a representation of a scene, the background model, and any significant change in an incoming frame from this background model is considered as a moving object. A connected component algorithm is applied to the pixels constituting the regions undergoing the change to get connected regions corresponding to the moving objects. A review of background subtraction techniques is given in [61].

Segmentation: Image segmentation algorithms split the image into perceptually homogeneous regions, which can be identified as objects and then are used in object tracking. The criteria for a good partition and the approach for achieving efficient partitioning are the two important problems that every segmentation algorithm needs to address. Some of the recent segmentation algorithms which are used in object tracking are mean-shift clustering, graph-cuts (and normalized cut), Markov random field (MRF), (geodesic) active contours, and level-sets. A review of image segmentation algorithms is found in [62].

Learning: Object detection can be accomplished by learning different object views automatically from a set of positive and negative instances using supervised learning algorithms. Given a set of learning instances, supervised learning algorithms generate a function or model that maps inputs to desired outputs. Some of these learning methods [63] are adaptive boosting (adaboost), support vector machines (SVM), decision trees, random forests, and neural networks (NN). A classifier which combines advantages of both kernel-based classifiers which handles high-dimensional feature spaces such as SVM and graphical model-based classifiers which captures correlations in structured data such as Markov networks has been developed and termed as Maximum Margin Markov Networks (M^3N) [64]. However, a large amount of manually labeled samples from every object class is required for using supervised learning algorithms.

To alleviate this problem, it is crucial to accompany co-training (semi-supervised learning technique) with supervised learning [65]. In co-training, two classifiers are trained using a small set of labeled samples where the features utilized for every classifier are statistically independent. Once trained, the most confident predictions of each classifier on the unlabeled data are used to iteratively construct additional labeled training data for the other classifier. Co-training has been successfully used with adaboost in [66] providing better results.

2.1.3.2 Exemplars of Object Detection Algorithms

We focus on object detection algorithms based on learning approaches as these are more promising than approaches using a keypoint, background subtraction and segmentation; a survey of object (pedestrian) detectors is given in [67] [68].

Integral images for quick feature computation, a combination of increasingly more complex classifiers in a cascade for efficient detection by discarding background regions while spending more computation on promising object-like regions, and Adaboost for automatic feature selection were introduced in [69] using Haar-like features and sliding window approach. Though this algorithm is very fast, it generates high false positive rates. A popular histogram of oriented gradient (HOG) features for object detection was introduced in [41] inspired by the work in [39]. It computes histogram of oriented gradients on a dense grid of uniformly spaced cells (not at sparse, scale-invariant key image points like SIFT [39]). These cells are grouped into larger overlapping blocks and are normalized within each block. The descriptor is the concatenation of these histograms. It is found that four 8×8 pixels cells per block (16×16 pixels per block) with 9 histogram channels (bins) are the optimal parameters. This detector has better performance by reducing false positive rates by a large margin over the Haar wavelet based object detector in [69]. Though HOG is the best performing single hand-engineered feature for object detection, it is possible to obtain a better performing detector by combining with other features. For instance, the HOG was combined with a texture descriptor based on local binary patterns (LBP) for pedestrian detection using an integral image, sliding window and linear SVM classifier to handle partial occlusion in [44]. The performance of the original HOG is improved in [70] which proposed a discriminatively trained part based approach by modeling unknown part positions as latent variables in an SVM framework (latent SVM) for a general object detection. This method discriminatively trains partially labeled data i.e. objects of interest are labeled while part locations are not labeled and are treated as latent (hidden) variables during training. The trained object model is composed of a root model and optionally multiple mix-

ture components. Though this method is quite popular with a very good performance, it is computationally expensive. The detector developed in [16] reduces computation time by approximating multi-resolution image features using extrapolation from nearby scales (usually finely sampled pyramids from coarsely sampled ones) rather than computing them explicitly. This detector, Aggregated Channel Features (ACF), uses three different kinds of features in 10 channels: normalized gradient magnitude (1 channel), histogram of oriented gradients (6 channels), and LUV color (3 channels) and has a better performance than even the detector developed in [70] with a considerably faster speed. This object detector is extended for detecting vehicles by sub-categorization as a means of capturing appearance variations due to varying orientation, truncation and occlusion levels [58]. It trains many models rather than just one as in [16] for handling different aspect ratios of vehicles i.e. one model is trained for each category of vehicles and the category of vehicles is determined by geometrical and visual features. A CNN-based subcategory-aware object detection was proposed [54] in which a region proposal network (RPN) and an object detection network were introduced by exploiting subcategory information. In this approach, a subcategory convolutional layer was introduced in the RPN where each filter in this layer is discriminatively trained for subcategory detection at a specific location and scale. The object detection network classifies region proposals generated by subcategory detection in RPN and refines their locations. Moreover, this detector also handles large scale variation of objects using image pyramids. This detection algorithm has a good performance, however, it is less flexible to adapt to small quantity of data sets. Thus, due to its friendliness, flexibility and reasonable performance, we have adapted the pedestrian detector in [16] and the vehicles detector in [58] for our data sets in Chapter 5.

2.2 Types of Visual Tracking

As mentioned earlier, the goal of visual tracking is to generate the trajectory of a target by finding its location in each video frame in which object detection and tracking can be accomplished either separately or jointly. In the first approach, an object detection algorithm is used to obtain possible object regions in each video frame and then a tracking algorithm is applied to the result of the object detection algorithm to make correspondences between targets across frames. However, in the latter approach, the target region and correspondence is simultaneously estimated by iteratively updating target position and region information obtained from previous frames. The model used to represent the shape of a target has great influence on the type of motion it can undergo when performing visual tracking. For example, while only translation is

necessary for objects represented by points, affine or projective transformations are the appropriate parameter motion models for objects represented using rectangles or ellipses. Accordingly, there are three main tracking categories: point tracking, kernel tracking and silhouette tracking. Each of these tracking categories can be used for either single or multiple target tracking problems. A good survey and book of visual tracking are found in [10], [12], [71], [2].

Point Tracking: This tracking approach can be formulated as the correspondence (association) of detected targets using points representation across frames, and this approach requires an external object detection technique in each frame. This point correspondence is a challenging problem specially in the presence of miss-detections, occlusions, appearance and disappearance of targets, and can be classified into two: deterministic and statistical point correspondence methods.

The deterministic methods define a cost of associating every target in a previous frame(s) to a single target in a current frame using a collection of motion constraints. This correspondence cost is formulated as a combinatorial optimization problem and is solved using optimal linear assignment methods such as the Hungarian algorithm [72] and Munkres algorithm [73] for two frames, or greedy search [74], min-cost network flow [75] [76] [77] and multidimensional assignment [78] for three or more frames.

However, the statistical methods solve the tracking problems by considering the observation and the motion model uncertainties during target state estimation. They use state space approach to model the target properties such as position, velocity and acceleration. For example, the Kalman filter [79] and particle filter or Sequential Monte Carlo (SMC) [80] estimate single target states. To track multiple targets using Kalman or particle filter, it is necessary to deterministically (nearest neighbour [81]) or probabilistically (Joint Probabilistic Data Association Filter (JPDAF) [81] [82], Multiple Hypothesis Tracking (MHT) [83]) associate the most likely observation for a particular target to that target's state; more detail is given in section 2.5. RFS-based filters (PHD, CPHD, CB-MeMBer) [84] do not need data association to filter multiple targets, however, post-processing the filtered results is necessary for labeling each target from frame to frame as discussed in section 2.5.3. Note that these approaches can be employed to estimate the state of any time varying system e.g. they have been used for tracking contours in [85].

Kernel Tracking: In this context, a kernel refers to describing the target shape and appearance e.g. rectangular template or elliptical shape with an associated histogram. Tracking of targets are performed by computing the motion of the kernel which can

be parametric transformation (translation, affine, etc) or the dense flow field in consecutive frames. There are two categories of kernel tracking algorithms based on the utilized appearance representation: templates and density-based appearance models such as mean-shift [30], CAMSHIFT (Continuously Adaptive Mean Shift which adjusts the window size) [86] and Kanade-Lucas Tomasi (KLT) tracker [31], and multiview appearance models such as support vector tracking [25].

Silhouette Tracking: These methods give an accurate shape description for complex shapes such as the human body (hands, head, shoulders, etc). The aim of this type of tracker is to find the target region in every frame using the object model such as color histogram, target edges or target contour generated from the previous frame, and can be divided into two: shape matching and contour tracking. While shape matching methods search for the target silhouette in the current frame [87], contour tracking methods evolve an initial contour to its new position in the current frame by either utilizing the state space models in which the target’s state is defined in terms of the shape and the motion parameters of the contour [85] or direct minimization of some energy functional (either variational such as level set [88], [89] or heuristics [90]). Obviously, both of these methods, shape matching and contour tracking, use the priors generated from the previous frames to segment objects in the temporal domain.

2.3 Challenges in Visual Tracking

The main challenges in visual tracking are the similarity of appearance between a target of interest and other targets (and the background) which produces clutter, and appearance variations of the target itself due to changes in pose (translation, rotation, deformation, etc), changing scene conditions (ambient illumination, weather), sensor noise, or occlusions (partial or total). Although all the listed challenges are very important in solving visual tracking problems, we present how to cope with appearance changes and handle occlusions briefly as follows.

2.3.1 Coping with appearance changes

Appearance variations can be coped with using model update strategies i.e. image measurements and the output of the tracker can be used to adapt the appearance representation of the target to the current scene conditions [2], [12]. The drawback of updating the target model online using the output of the tracker is model drifting i.e.

the gradual impoverishment of information in the model due to the amplification of the tracking error caused by the update feedback loop which can be reduced by including a contribution from the initial model into the update strategy [91].

Generally, there are two online learning methods for handling appearance changes in model-free tracking: generative and discriminative methods [12] as discussed in section 2.4. The model learned in the current frame using these methods should be updated by including the contribution from the model of the previous frame (which in turn takes contribution from the initial frame) to make the tracker more adaptive to appearance variations of a target of interest [7]. However, for model-based approach (the class of the objects to be detected is known beforehand), both single and multiple target tracking, it is important to include a significant number of training samples with appearance variations to be able to detect and track the target(s) under appearance changes.

2.3.2 Handling occlusions

This challenge can be addressed using different approaches depending on the expected level of occlusion:

Partial occlusion: This kind of occlusion which can affect only a small portion of the target area can be handled by either the target appearance model or the target detection algorithm itself. Some global feature representations such as a histogram have invariant properties which can deal with this type of occlusion [30]. Moreover, using multiple localized features which encode information for a small portion of the target instead of a global representation may increase the robustness of the tracking algorithm.

Total occlusion: With this kind of occlusion any information on the target appearance is absent. Track continuity in this type of occlusion can be obtained via higher-level reasoning or through multi-hypothesis methods such as the particle filter that keeps propagating the tracking hypotheses over time. The trajectory of the target in the absence of valid measurements can be propagated using information about pre-existing occlusion patterns and typical motion behaviours i.e. carry on predicting the states of the target until the target reappears. When the target reappears from occlusion, the necessary cues which can be obtained by the propagation of multiple tracking hypotheses and appearance modeling re-initialise the track [2]. Some tracking algorithms use a separate re-detection module which can be trained in parallel and then re-initialize

the tracker in case of tracking failure [13] [7]. Occlusion can also be indirectly resolved while generating tracks of targets. The use of depth information such as from stereo cameras or the use of camera network to cover wider space in multi-view can also solve occlusion problems.

2.4 Model-free Tracking

Model-free tracking is a type of tracking in which a target of any class (arbitrary object) is manually annotated in the first frame of a video sequence and then needs to be tracked throughout the remainder of the video sequence. However, the class of targets to be detected are required to be known in advance in model-based tracking [25] [92], and these objects are first detected based on a pre-trained model and then are tracked. Model-free tracking lifts the design efforts needed in model-based approaches to treat each object class separately. Although it has been studied extensively during past decades as recently surveyed in [93] [12], model-free object tracking is still a difficult problem due to many challenges that cause significant appearance changes of targets such as illumination changes, occlusion, pose variations, deformation, abrupt motion, and background clutter (see section 2.3). Little prior information about the target to be tracked is also another challenging factor. Particularly, tracking an interested target in dense or crowded environments is an important task in some security applications, however, it is very challenging due to heavy occlusion, high target densities, cluttered scenes and significant appearance variations of targets. Robust representation of target appearance is important to overcome these challenges.

Various visual tracking algorithms have been proposed over the past decades to cope with the challenges in visual tracking, and they can be categorized into two types depending on the learning strategies: *generative* and *discriminative* methods. *Generative* methods describe the target appearances using generative models and search for target regions that best-fit the models i.e. search for the best-matching windows (patches). In other words, generative methods model only the appearance of the object without considering the appearance of the background unlike discriminative methods which make them fail when the background is cluttered. Various generative target appearance modelling algorithms have been proposed such as online density estimation [94], sparse representation [95,96], and incremental subspace learning [97].

On the other hand, *discriminative* methods build a model that distinguishes the target from the background. These algorithms typically learn classifiers based on online

boosting [98], multiple instance learning [99], P-N learning [13], transfer learning [100], structured output SVMs [101] and combining multiple classifiers with different learning rates [6]. Background information is important for effective tracking as explored in [93] [102] which means that more competing approaches are discriminative methods [103]. Discriminative methods are sensitive to noise and generative methods fail within cluttered backgrounds, therefore, these two methods can be combined to overcome these problems as used in [104] [105] [106]. Model-free tracking is not only used to track a single unknown target but also used for tracking multiple targets as in [107]. However, all of these mentioned algorithms use hand-crafted features which are less robust to significant appearance variations of targets. Recently, CNN features have demonstrated outstanding results on various recognition tasks [4, 52]. Motivated by this, a few deep learning based trackers [108, 109] were developed which learn online classifier by drawing positive and negative training examples around the estimated target location and then extracting deep learning features from them. However, sampling ambiguity is one of the big problems in discriminative tracking methods which results in drifting.

More recently, correlation filters [8, 110, 111] have been introduced for online visual target tracking that can alleviate the sampling ambiguity in discriminative tracking approaches. Previously, the large training data required to train correlation filters prevented them from application to online visual tracking though correlation filters are effective for localization tasks. However, recently all the circular-shifted versions of input features have been considered with the help of a circulant matrix producing a large number of training samples [8, 110]. Hence, discriminative correlation filter-based trackers have achieved state-of-the-art results as surveyed in [112] in terms of both efficiency and robustness due to three reasons. First, efficient correlation operations are performed by replacing exhausted circular convolutions with element-wise multiplications in the frequency domain which can be computed using the fast Fourier transform (FFT) with very high speed. Second, thousands of negative samples around the target’s environment can be efficiently incorporated through circular-shifting with the help of circulant matrix [113] [8]. Third, training instances are regressed to soft labels of a Gaussian function (Gaussian-weighted labels) instead of binary labels alleviating sampling ambiguity. In fact, regression with class labels can be seen as classification. However, correlation filter-based trackers are susceptible to long-term occlusions.

There are three tracking scenarios that are important to consider: short-term tracking, long-term tracking, and tracking in a crowded scene. If an object is visible over the whole course of the sequence, short-term model-free tracking algorithms are sufficient to track a single object without applying a pre-trained model of target appearance.

There are many short-term tracking algorithms developed in the literature [11] [112] such as online density estimation [94], context-learning [114], scale estimation [111], and using features from multiple CNN layers [5, 115]. However, these short-term tracking algorithms can not re-initialize the trackers once they fail due to long-term occlusions and confusions from background clutter.

Long-term tracking algorithms are important for tracking an unknown object in a video stream that runs for indefinitely long handling long-term occlusions and difficult background clutter. Long-term trackers are usually developed either by introducing a re-detection module [13] [7] or learning/maintaining conservative appearance of the target [116] [117]. A Tracking-Learning-Detection (TLD) algorithm has been developed in [13] which directly decomposes the long-term tracking problem into tracking, learning and detection. In this case, the tracker tracks the target from frame to frame and provides training data for the detector which re-initializes the tracker when it fails. The learning component estimates the detector’s errors and then updates it for correction in the future. This algorithm works well in very sparse video but is sensitive to background clutter. Long-term correlation tracking (LCT), developed in [7], learns three different discriminative correlation filters: translation, appearance and scale correlation filters using hand-crafted features. Even though it includes a re-detection module by learning the random ferns classifier online for re-initializing a tracker in the case of tracking failures, it is not robust to long-term occlusions and background clutter. Multi-domain network (MDNet) [118] pre-trains a CNN network composed of shared layers and multiple domain-specific layers using a large set of videos to get generic target representations in the shared layers. This proposed network has separate branches of domain-specific layers for binary classification to identify the target in each domain. However, when applied to fundamentally different videos other than the related videos on which it was trained, it gives poorer results.

Tracking of a target of interest in a crowded scene is very challenging due to heavy occlusions, high target densities and cluttered scene, and significant appearance variations. Person detection and tracking in crowds is formulated as a joint energy minimization problem by combining crowd density estimation and localization of individual person in [119]. This approach uses a pre-trained model on human heads to initialize targets, thus it is not suitable for tracking a generic target of interest in model-free sense. The method developed in [120] trained Hidden Markov Models (HMMs) on motion patterns within a scene to capture spatial and temporal variations of motion in the crowd which are used for tracking individuals. However, this approach is limited to a crowd with structured pattern i.e. it needs some prior knowledge about the scene. The algorithm developed in [14] uses visual information (prominence) and spatial context (influence

from neighbours) to develop online tracking in crowded scene without using any prior knowledge about the scene, unlike the method in [120] which also uses some training data from the past as well as the future. This algorithm performs well in highly crowded scenes but has a low performance in a less crowded scenes as influence from neighbours (spatial context) decreases. However, our work in chapter 3 tries to close the gap between sparse and crowded scenes tracking problems by proposing an algorithm which can be applied to both scenes, particularly robust to long-term occlusions and challenging clutter from both background scene and other uninterested targets.

2.5 Bayesian Tracking

Target filtering is a state estimation problem which plays a key role in visual, radar and sonar tracking, robot simultaneous localization and mapping (SLAM), and other signal processing applications. In fact, there are three different kinds of state estimation methods: Maximum Likelihood (ML), Maximum a posterior (MAP), and Bayesian estimation (filtering). ML estimates the set of values of the model parameters that maximizes the likelihood function whereas MAP estimates a mode of the posterior distribution. MAP is closely related to ML estimation but employs an augmented optimization objective which incorporates a prior distribution over the quantity one wants to estimate, thus, it can be seen as a regularization of ML estimation (ML lacks prior) [63]. The Bayesian inference-based visual tracking approach has succeeded in computer vision when compared to ML, MAP and exhaustive search-based methods such as Mean-shift and KLT since it offers a systematic way of combining prior knowledge of object positions, modeling assumptions, and measurement information to the problem of tracking target(s) [81]. In Bayesian estimation, states are hidden variables (modelled using Hidden Markov Models (HMM)) i.e. they are obtained indirectly from measurements. This is because the target state space is treated as a hidden layer whereas the observation space is visible. Though Bayesian networks (directed probabilistic graphical models, e.g. HMM) are widely used for tracking applications, Markov networks (undirected probabilistic graphical models or MRF) such as conditional random fields (CRFs) are also used for solving tracking problems. Probabilistic graphical model is a marriage between a probability theory and a graph theory [121].

The Bayesian approach is the main approach for estimating the trajectories of a target in either the image or ground plane as it moves in the scene. The Bayes filter has two steps: the prediction step which predicts the target state based on dynamical model and the update step which updates the resulting density using a newly available

measurement. Two known implementations of this filter are the Kalman filter and its extended versions [79], and the particle filter (PF) [80], both for single-target tracking. Obviously, Bayesian filtering algorithms can be used with model-free as well as model-based tracking approaches.

2.5.1 Single Target Tracking

The single-target tracking task can be modeled using the state and the measurement equations [79].

$$x_k = \mathbf{y}_k(x_{k-1}, u_{k-1}, n_{k-1}) \quad (2.1)$$

and

$$z_k = \mathbf{f}_k(x_k, v_k) \quad (2.2)$$

where \mathbf{y}_k and \mathbf{f}_k are non-linear, time-varying functions, $\{u_{k-1}, k \in \mathbb{N}\}$ is the known control input which is not necessarily available (usually used in robotics), and $\{n_{k-1}, k \in \mathbb{N}\}$ and $\{v_k, k \in \mathbb{N}\}$ are assumed to be independent and identically distributed (i.i.d) stochastic processes. Usually, equation 2.1 is assumed to be a Markov process i.e. state x_{k-1} contains all measurement information z_{k-1} up to time $k - 1$. Most of the time, the function \mathbf{y}_k is obtained using a state-space model.

The goal of tracking is to estimate the states of targets which can be the positions and velocities of targets. The state sequence is assumed to be stochastic and, therefore, it is looking for the probability density function (pdf) of the target states. Thus, tracking is to estimate $p_{k|k}(x_k|z_{1:k})$, the *pdf* of the target being in state x_k , given all the measurements z_k up to time k , based on Eq. (2.1) and Eq. (2.2). The estimation is accomplished recursively in two steps: prediction and update.

The *prediction step* uses the dynamic model defined in Eq. (2.1) to obtain the prior *pdf* using Chapman-Kolmogorov equation given by

$$p_{k|k-1}(x_k|z_{1:k-1}) = \int y_{k|k-1}(x_k|x)p_{k-1|k-1}(x|z_{1:k-1})dx \quad (2.3)$$

with $p_{k-1|k-1}(x_{k-1}|z_{1:k-1})$ known from the previous iteration and the transition density $y_{k|k-1}(x_k|x_{k-1})$ determined by Eq. (2.1).

The *update step* uses Bayes' rule once the measurement z_k is available to get the posterior pdf

$$p_{k|k}(x_k|z_{1:k}) = \frac{f_k(z_k|x_k)p_{k|k-1}(x_k|z_{1:k-1})}{\int f_k(z_k|x)p_{k|k-1}(x|z_{1:k-1})dx} \quad (2.4)$$

where the likelihood $f_k(z_k|x_k)$ is determined by equation (2.2).

2.5.1.1 Kalman Filter

Since equations 2.3 and 2.4 cannot be solved analytically, under the assumption of linearity for equations (2.1) and (2.2) and Gaussianity of the prior $p_{k-1|k-1}(x_{k-1}|z_{1:k-1})$ and of the two noise sources, n_{k-1} and v_k , an optimal solution can be obtained using a Kalman filter [79]. If equations (2.1) and (2.2) are mildly non-linear, it can be solved sub-optimally using the extended Kalman filter (EKF) [79] and unscented Kalman filter (UKF) [122].

EKF: When equations 2.1 and 2.2 are non-linear, the Kalman filter cannot be used directly. Instead, a local linear approximation around the current means m_{k-1} and $m_{k|k-1}$ at each time should be used to get a transition matrix F_k and an observation matrix H_k using first-order Taylor expansion, respectively, and then the standard KF is applied to obtain a sub-optimal overall solution. If the non-linearities become severe, the performance of this filter can decrease drastically, and it is difficult to know this ahead of time. Another drawback is that linear approximation (Jacobian matrix) should exist to apply linearization which is not always the case due to discontinuities and singularities [122].

UKF: This filter is another sub-optimal extension of Kalman filter which uses sigma-points. Its basic idea is that instead of approximating an arbitrary non-linear function or transformation, it is easier to approximate a probability (Gaussian or normal) distribution [122]. This algorithm can be summarized as follows:

1. Choose a minimum number of $2n_x + 1$ sigma points from a Gaussian distribution with a desired mean and covariance where n_x is the dimension of the state.
2. Transform the chosen sigma points using the non-linear equation to obtain a set of transformed points.

3. Re-approximate the non-linearly transformed mean and covariance of the normal distribution using the transformed points.

The UKF can handle severe non-linearities with more accuracy than the EKF with the same computational complexity. However, a Gaussian assumption is still made. If the true density is non-Gaussian, none of KF, EKF and UKF can handle this; only the particle filter [80] can manage such scenarios.

All Kalman filters (KF, EKF and UKF) can only approximate state vectors of a fixed length. Hence, they cannot track a varying number of targets; a separate filter for each target should be applied using data association algorithms.

2.5.1.2 Particle filter

When the posterior is a uni-modal distribution, the Kalman filter can handle it, however, in some real scenarios, this is not the case i.e. the posterior is a multi-modal pdf. In this case another generic method called Sequential Monte Carlo (SMC) is necessary [80] that can approximate or sample the states of non-linear dynamical models and non-Gaussian noise. This SMC approach is known by different names such as the particle filter, bootstrap filter, Condensation algorithm, interacting particle approximations, and survival of the fittest [80]. It is a technique for implementing a recursive Bayesian filter by Monte Carlo simulation in which the key idea is to approximate the posterior pdf with the set of weighted random samples (Dirac δ functions or particles) and to estimate states based on these samples and corresponding weights. As the number of samples becomes large, this filter can approach the optimal Bayesian estimate. Thus, the Kalman filter is a single-hypothesis method i.e. only one track candidate estimate is evaluated at any time whereas particle filter is a multiple-hypothesis method i.e. multiple track candidates are evaluated simultaneously. The performance of a tracker can be improved by propagating multiple hypotheses.

Sequential Importance Sampling (SIS) is a Monte Carlo method that forms the basis for SMC or particle filters. The main idea of importance sampling is to give more emphasis to the high density areas rather than using the inefficient uniform sampling [123]. The term sequential in the SIS algorithm implies that a series of measurements is received sequentially i.e. only the current observation is considered without considering any previous recorded observations though it is possible to parallelize as all particles are generated at the same time. However, this SIS algorithm suffers from the degeneracy problem where all but one particle will have negligible weight after a few iterations. The

brute-force approach to reduce this problem is to use a very large number of particles which is sometimes impractical. There are two widely used approaches to solve this problem: good choice of importance density and use of resampling. The key idea of resampling is to discard particles with negligible weights while focusing on the particles with large weights. There are many resampling techniques that are used in particle filtering, of which systematic resampling performs better as it minimizes Monte Carlo variation [124]. However, resampling creates other practical problems though it can reduce the effects of the degeneracy problem. These problems are:

1. All the particles have to be combined which restricts the chance of parallelization.
2. Large weighted particles are statistically chosen repeatedly resulting in loss of diversity called sample impoverishment which can be severe for small process noise. This problem can be tackled using the resample-move algorithm and regularization [80].
3. Any smoothed estimates based on the paths of the particles degenerate as their diversity is reduced. This effect can be overcome by using Markov Chain Monte Carlo (MCMC) sampling [125].

There are various particle filters proposed in the literature that are regarded as special cases of this general SIS algorithm which are derived from the SIS algorithm by a suitable choice of importance sampling density and/or modification of the resampling step. These special particle filters are Sampling Importance Resampling (SIR), Auxiliary Sampling Importance Resampling (ASIR) and the Regularised Particle Filter (RPF) [80].

A popular process called Rao-Blackwellization can also be integrated into the particle filter with the basic idea of decreasing the size of the state space by marginalizing out some of the variables analytically. Kalman filter is used for analytical updates so that the remaining space that has to be sampled is smaller which leads to a much lower required number of samples [126]. An unknown number of targets are tracked using the Rao-Blackwellized particle filter (RBPF) in [127]. RBPF is more accurate and computationally efficient than the standard particle filter (PF).

MCMC is another sampling technique which applies a Markov chain model to Monte Carlo integration to produce a series of samples called chain [125]. In the chain, the next sample is dependent on the previous sample in the series which enables the MCMC sampling method to explore high dimensional state space by making a chain of samples

around high density regions. There are various closely related MCMC algorithms with different names: Metropolis-Hastings algorithm, Gibbs sampling, Slice sampling, and Multiple-try Metropolis [128]. The MCMC method which uses rejection sampling is termed as Metropolis-Hastings Algorithm which may be trapped in a peak, unable to jump to other peaks, as it rejects all low density regions. An extension of the standard MCMC algorithm to sampling of posterior distributions of varying dimension is called Reversible Jump Markov Chain Monte Carlo (RJ-MCMC) [125], [129] and is used to infer states in trans-dimensional MCMC approaches [130].

Generally, the SIS algorithm and its special particle filters (using importance sampling) are preferred for low dimensional problems. MCMC particle filters (using MCMC sampling) with marginal proposal strategy using the Metropolis Hastings rule are preferred for high dimensional problems. RJ-MCMC particle filters (using RJ-MCMC sampling) can be used to manage states with a varying dimension. When using particle filters such as the joint particle filter, MCMC particle filter and RJ-MCMC particle filter, the identity of targets can be included in state vectors in which case a separate data association algorithm is not necessary for tracking multiple targets [131]. However, if a particle filter is to be applied for each target to track multiple targets, an explicit data association is necessary.

2.5.2 Data Association-based Multi-Target Tracking

While tracking targets, multiple measurements usually appear due to either actual targets or measurement noise (clutter or false alarms). Data association deals with the task of choosing the observation that most probable is generated from the target to be tracked. Actually, the data association task is concerned with both associating targets with correct measurements as well as associating targets temporally (labeling identities of targets across time). The first step in solving data association problem (associating targets with correct measurements) is the choice of a validation region called measurement validation or gating (shown in Fig. 2.3) which is a region in which the next measurement is more likely to appear i.e. gating discards less probable association hypotheses for reducing computational cost. The usual approach of measurement validation is to first predict the current state from the previous states, and then to assess the compatibility of each observation with this predicted state.

Note that the validation procedure has to be repeated for every object and for every tracking hypothesis for the case of multiple targets and multiple tracking hypothesis.

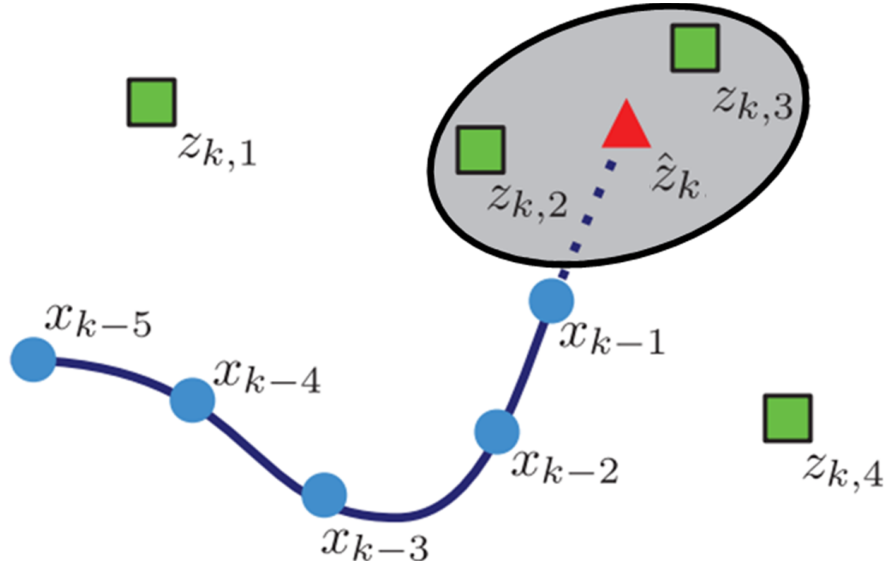


Figure 2.3: Data association problem: two validated measurements (green boxes) are located in the validation region (grey area) centred on the predicted measurement (red triangle) [2].

Another problem is that the number of targets can be unknown and varying since new targets can appear and disappear at any time and anywhere in which case invalidated measurements might be important. Therefore, a data association algorithm should determine whether an observation is correct or incorrect as well as if a new track has to be initialized or an existing track has to be continued. All observations that are not utilized for the continuation of existing tracks are either due to clutter or the appearance of new targets. Therefore, a strategy that can deal with the appearance and disappearance of tracks should be included in the data association algorithms based on gating.

2.5.2.1 Data Association Algorithms

There are various statistical data association algorithms which are widely used for multi-target visual tracking in the computer vision community [81], [132]. Some of them are discussed below.

Nearest Neighbour Filter: This algorithm selects a measurement closest to the predicted measurement \hat{z}_k by computing the Mahalanobis distance d for each received measurement z_k and is given by

$$d(z_k, \hat{z}_k) = \sqrt{(z_k - \hat{z}_k)^T S_k^{-1} (z_k - \hat{z}_k)}, \quad (2.5)$$

where $\hat{z}_k = H_k m_{k|k-1}$ is the predicted observation projected from the predicted state $m_{k|k-1}$, and $S_k = H_k P_{k|k-1} H_k^T + R_k$ is the innovation measurement covariance, H_k is the observation matrix, R_k is the measurement noise covariance and $P_{k|k-1}$ is the predicted covariance. Note that $m_{k|k-1}$ is equivalent to x_{k-1} in Fig. 2.3.

There are various variants of this filter such as probabilistic nearest neighbour, distributed sequential nearest neighbour, suboptimal nearest neighbor, and Global Nearest Neighbour (GNN) as detailed in [81]. Though this filter has low computational complexity, its performance decreases rapidly when the number of false measurements increases. Another problem of this filter is that one observation can be associated with multiple tracks giving unrealistic results, however, the GNN approach solves this problem by the constraint of one observation can only be associated with one track as detailed in [133] where it is stated that it has still a poor performance. Moreover, this filter cannot deal with the appearance and disappearance of targets, therefore, it can only be used in combination with a birth and death model.

Probabilistic Data Association Filter (PDAF): This is a sub-optimal algorithm in which a weighted average of all validated measurements is given as an input to the tracking algorithm [134]. The state is assumed to be normally distributed according to the latest state estimate and covariance matrix, and linear dynamic and measurement models are considered i.e. Kalman filter plays a central role in this filter though linearization can be used for a non-linear model. Moreover, only one target is modeled whose track has been initialized. However, when gating regions overlap, the same observations can contribute to updating of multiple predicted states. The extension of this PDAF called Joint PDAF (JPDAF) enables tracking of multiple targets in clutter in which measurements to targets probabilities (weights) are computed jointly across targets with a quadratic complexity with the number of targets. However, it only considers the last measurement (non-back-scan), and each target is assumed to have its own linear dynamic and measurement models though linearization can also be exploited in case of mildly non-linear models. Moreover, the number of targets is assumed to be known and constant though it can deal with multiple sensors as detailed in [81] [132]. A sample-based version of JPDAF (S-JPDAF) was introduced in [135] in which Kalman filter is replaced by a particle filter, and a varying and unknown number of targets can also be tracked. Another filter called the Monte Carlo JPDAF (MC-JPDAF) was introduced in [136] generalizing it for multiple sensors and arbitrary proposal distributions though a fixed and known number of targets is assumed. A fast and efficient version of JPDAF has been recently proposed and applied to multi-target tracking in [137].

Multiple Hypothesis Tracking (MHT): This algorithm is the most widely used and optimal data association algorithm as it postpones the data association decision till more information becomes available (N-back-scan) [138] [139] [133] as opposed to JPDAF which considers only two frames of a video, and its implementation library is given in [140]. However, it has an exponential complexity with time and cubic with the number of targets. The first step of MHT is the formulation of all feasible hypotheses. Then, when a new measurement is available, each hypothesis is expanded into a set of new feasible hypotheses so that a tree of hypotheses can be generated. The track score is utilized to assess the validity of the track. Since the size of the hypotheses tree grows exponentially as more measurements are available, there are several techniques used to limit the size of the tree such as clustering of measurements, and track and hypothesis pruning as discussed in [133] [139]. This filter handles the appearance and disappearance of targets as well as a varying and unknown number of targets as opposed to JPDAF. Moreover, MHT can tackle the problem of occlusions i.e. continuation of a track in case some of the measurements from a target are missing. The MHT algorithm which makes associations in a deterministic sense is extended to probabilistic MHT (PMHT) to lower its computational complexity considerably from exponential in time [141]. Rather than listing all feasible combinations of hypotheses, PMHT tries to compute the maximum likelihood solutions using an expectation-maximization (EM) algorithm. About ten different PMHT algorithm implementations are compared pointing out their strong and weak sides in [142]. Data association of a similar method to PMHT is used in [143] where a particle filter is used for tracking multiple targets. MHT is recently revisited in [144] by training online appearance models for each track hypothesis which has good performance.

There are also other data association approaches (deterministic) based on combinatorial optimization for multi-target tracking. For example, optimal assignment methods such as the Hungarian algorithm [72] and Munkres algorithm [73] are used in [145], and graph matching [74] is used in [146]. The (2D) linear assignment problem can be interpreted as a weighted bipartite graph. Generally, optimal linear assignment methods such as the Hungarian algorithm [72] and Munkres algorithm [73] are used for two frames whereas greedy algorithm (for graph matching) [74], (min-cost max-flow) network flow [75] [76] [77] and multidimensional assignment [78] methods generalize to three or more frames. The greedy algorithm and network flow have a polynomial-time complexity whereas the multidimensional assignment has NP-hard (non-deterministic polynomial-time hard). Moreover, Siamese CNN for robust data association within the context of pedestrian tracking has been introduced in [147] which uses a two-stage learning scheme to match pairs of detections.

2.5.2.2 Exemplars of Traditional Multi-Target Trackers

There are a plenty of multi-target trackers in the literature as surveyed in [148]. We only describe some of the typical traditional data-association based multi-target tracking algorithms.

To start with, a method developed in [131] incorporated a Markov random field (MRF) motion prior into a MCMC particle filter-based multi-target filter for modeling target interactions which helps to maintain the identity of targets. However, it suffers from computational cost with a number of tracked targets though it reduces the exponential complexity of a joint particle filter and is limited to track only a fixed number of targets. It is extended to handle a variable number of interacting targets from a static camera by extending MCMC sampling to a RJ-MCMC sampling scheme. Similarly, a RJ-MCMC sampling-based multi-target visual tracking algorithm was developed in [149] by modeling the interaction of targets using a MRF motion prior to track pedestrians from a mobile vision platform such as cameras mounted on a vehicle or a Kinect on a robot in which both the camera’s ego motion and pedestrians’ trajectories are estimated in a single coherent framework without using stereo information as in [150] [151]. However, RJ-MCMC sampling based trackers are computationally intensive. In [150] and [151], multiple detectors are combined to estimate the position of camera odometry and track pedestrians using stereo camera for getting stereo depth information. Online multi-target tracking-by-detection from a single, potentially moving, uncalibrated camera was proposed in [92] in a particle filtering framework.

A stable multi-target visual tracker which estimates stable and precise object (head) locations was developed in [152] by combining HOG detections with simultaneous KLT tracking and Markov Chain Monte Carlo Data Association (MCMCDA) within a temporal sliding window i.e. MCMC is used to sample from the huge combinatorial space of hypotheses. The MCMC stochastically explores the search space by proposing a set of ‘moves’ from the current to a next state. Partial occlusion is handled using a part-based model at both detection and tracking stages for tracking multiple persons in [153]. Recently, a variational Bayesian model was introduced for tracking an unknown and varying number of persons in [154] which has closed-form expressions for the posterior distributions of the latent (hidden) variables and for the model parameters, and is implemented via a variational EM (VEM) algorithm. It jointly infers both the state variables and the assignment variables (labels) in the filtering equation and handles both birth and death of targets. However, it is still computationally demanding though better than the trans-dimensional MCMC approaches where the dimensionality of the state space is treated as a state variable and the states are inferred by RJ-MCMC

sampling.

Graph theory has been introduced for multi-target tracking problems [74] [75] [76] [77]. A greedy algorithm for graph matching that generalizes for a multi-frame correspondence was proposed in [74] which is efficient for real-time tracking, however, it is purely causal (does not look-ahead) i.e. if matches were once made, they cannot be corrected if later information shows them to be suboptimal. A network flow based optimization method for data association was proposed for multi-target tracking in [75] by mapping a maximum-a-posteriori (MAP) into a cost-flow network with a non-overlap constraint on trajectories (edge disjoint paths in a graph) and the global optimal solution is obtained by a min-cost flow algorithm in the network. Efficient, greedy but globally optimal algorithm for tracking multiple targets was introduced in [76] in which the number of targets and their track births and deaths are estimated based on a min-cost flow framework. However, both methods in [75] [76] optimize the association globally using all the observations from the whole sequence which limits their applications for online tracking problems. The multi-target tracker in [77] uses the min-cost flow algorithm to solve the data association problem optimally in which the drawbacks such as computational cost, whole video as a batch input assumption and computational scale with the length of the video sequence are overcome making it suitable for online data stream. For non-overlapping camera networks, an approach that jointly optimises single camera object tracking and inter-camera object tracking in an equalised global graphical model was proposed in [155]. This method is an extension of [75] where tracklets are used as input to the network flow rather than object detections. Simultaneous multi-object tracking and classification was proposed in [156] using a graphical probabilistic model and an inference procedure and then solving using a variational approximation, however, it suffers from class switching.

Conditional Random Fields (CRFs) have also been introduced for multi-target tracking [157] [158] [159] [160] where tracking can be solved by an energy minimization or sampling-based inference strategy. A learning-based CRF model for tracking multiple targets was proposed in [157] by progressively associating detection responses into long tracks in which a CRF model considers both tracklet affinities or similarities (node of CRF) and dependencies among them (edge of CRF). In this approach, the best global associations are learned rather than the best local affinities between tracklets by transforming the task into an energy minimization problem, however, this method is computationally intensive. While this method models association dependencies well and works offline, the method in [158] models better discrimination of spatially close targets with similar appearance by learning CRF model online which utilizes unary energy cost (for node of CRF) and pairwise or binary energy cost (for edge of CRF)

based on motion and appearance models i.e. the meanings of edges in CRF are different. Although the approach in [158] is faster than the method in [157], it has still a polynomial complexity with the number of tracklets. Graphical models, particularly factor graphs, were used to model as many tracker components as possible including the observation model, the motion model, the interaction model and the termination procedure within a log-linear CRF to develop a robust multi-target visual tracker in [159], however, it is computationally intensive. In this method, a sampling-based inference strategy is used since the model cannot be globally optimized. The tracker developed in [160] considers long-time connectivity between detections as well as similarity and dissimilarity between them in a time-interval using position, color, motion and SURF cues in a CRF framework.

Online multi-object tracking with moving cameras was proposed in [161] which uses a data association method that exploits structural motion constraints (location and velocity) and event aggregation (assignment ambiguities). A near-online multi-target tracking algorithm was developed in [162] by introducing an affinity measure to associate detections considering the relative motion pattern between a pair of temporally distant detections using long term interest point trajectories and then taking the advantages of both the pure online and global tracking algorithms. More recently, Siamese CNN has also been introduced in multi-target tracking [147] [163]; hand-engineered features are also used in [164]. These approaches use similarity between detections for data association unlike others such as MHT, network flow and CRF which use temporal consistency for data association. A structure of recurrent neural networks (RNNs) is used to jointly reason on multiple cues (motion, appearance and interaction) over a temporal window to develop an online multi-target tracker in [165]. A deep CNN is used in [166] for a reliable affinity measure between pairs of detections which is used to specify a cost of a graph-based optimization problem for multi-person tracking. However, it is computationally expensive.

In practice, though deterministic optimization approaches which require whole video as a batch input outperforms the approaches based on probabilistic inference, for instance, in the case of occlusion among targets, their applications are limited in online tracking problems since they need all frames of the video beforehand. However, to the best of our knowledge, no work was done which can solve the confusion among detectors of different target types for online multi-target tracking as our work in Chapters 4 and 5.

The table given in 2.1 lists and compares a sample of traditional multi-target visual tracking algorithms in terms of targets birth, death, interaction, clutter, missed detection (MD) due to (short term) occlusion, and type of used camera (static or moving).

Ref	Algorithms	Birth	Death	MD	Clutter	Interaction	Camera
[131]	MCMC-PF	no	no	no	no	yes	static
[149]	RJ-MCMC-PF	yes	yes	yes	yes	yes	moving
[159]	CRF-FG	yes	yes	no	no	yes	static
[158]	online-CRF	yes	yes	yes	yes	no	moving
[152]	MCMCDA	yes	yes	yes	yes	no	static
[75]	GlobalNF	yes	yes	yes	yes	no	static
[76]	GlobalOptimal	yes	yes	yes	yes	no	static
[77]	FollowMe	yes	yes	yes	yes	no	static
[78]	Multi-D	yes	yes	yes	yes	no	static
[154]	VEM	yes	yes	yes	yes	no	static
[161]	MOT-EA	yes	yes	yes	yes	no	moving
[144]	MHT-R	yes	yes	yes	yes	no	static
[137]	JPDAF-R	yes	yes	yes	yes	no	static
[166]	MultiCutDM	yes	yes	yes	yes	no	moving
[165]	RNN-T	yes	yes	yes	yes	yes	moving

Table 2.1: Survey of sample tracking algorithms.

2.5.3 Multi-Target Tracking using Random Finite Sets

To extend the single target tracking algorithms for tracking multiple targets, they need to handle the target birth and target death, clutter (false alarms), and missing detections. Traditionally, multi-target filters are based on the concept of finding associations between targets and measurements called Bayesian data associations [81] discussed in section 2.5.2 such as Global Nearest Neighbour (GNN) [167], Joint Probabilistic Data Association Filter (JPDAF) [82], and Multiple Hypothesis Tracking (MHT) [83]. However, these approaches faced challenges not only in the uncertainty caused by the data association but also the computational growth exponentially to the number of objects and measurements. For example, the MHT has an exponential complexity with time and cubic with the number of targets.

Recently, single-target filtering has been extended to handle the varying number of targets due to new targets appearing in the scene and old targets leaving the scene in a Bayesian framework using Finite Set Statistics (FISST). In this approach, one may think of stacking the states of every tracked object in a big vector to fit into a Bayesian framework, however, as the number of targets in the scene are changing, one has to deal with state sizes of non-fixed dimension. Nevertheless, it is impossible to consistently calculate the estimation error between a frame with no target (empty vector) and a consecutive frame with targets (non-empty vector). A suitable method to handle this problem is to represent the state and observation vectors as finite sets which have a defined estimation error computation method (the minimum Euclidean distance over all permutations of the individual states). Therefore, the analogous of a random vector for single-target tracking is a Random Finite Set (RFS) for multi-target tracking which can naturally represent the varying number of non-ordered multi-target state and observation dimensions as shown in Fig. 2.4. A RFS is a set where not only the elements are random stochastic processes but also the set cardinality is a stochastic process. More precisely, RFS is a finite-set-valued random variable, that is a random variable which is random in both the number of elements and the values of the elements themselves [84]. Accordingly, in order to fit into the Bayesian framework, the multi-target states are considered as a single meta-target represented using RFS whose Bayesian propagation is similar to that of single-target case. Similarly, the multi-target observations are treated as a single set of measurements of the meta-sensor represented using RFS. Hence, FISST, the study of statistical properties of RFS, is the first systematic treatment of multi-sensor multi-target filtering as a unified Bayesian framework using random set theory. Similar to single-target filtering, this multi-target filtering approach propagates the multi-target posterior density recursively [168] [169] [170] [171].

The FISST framework consists of two spaces: a state space and an observation space as shown in Fig. 2.5. As can be observed from Fig. 2.5, the observation space contains many observations produced by objects or targets without any knowledge of which object generates which observation, and some of these observations are clutter which do not correspond to real targets. In visual tracking, object detection algorithms extract these measurements from image space which then can help to estimate the targets' states and cardinality jointly.

However, as the number of targets increases, the dimensionality of the target states still increases. Therefore, rather than propagating the full multi-target posterior, it is less computationally intensive to propagate its first-order moment called an intensity function or Probability Hypothesis Density (PHD) [168] [169] [170]. The area inside

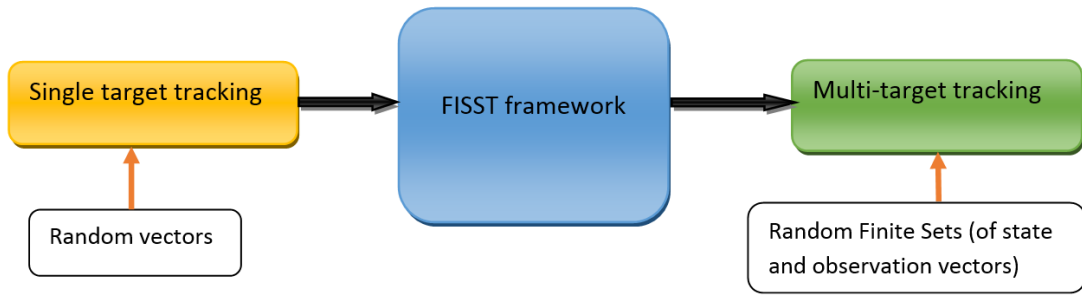


Figure 2.4: Single target to multi-target extension using RFS theory. The black arrows show the extension of single target tracking in which states and observations are represented using random vectors to multi-target tracking through FISST framework in which states and observations are represented using random finite sets.

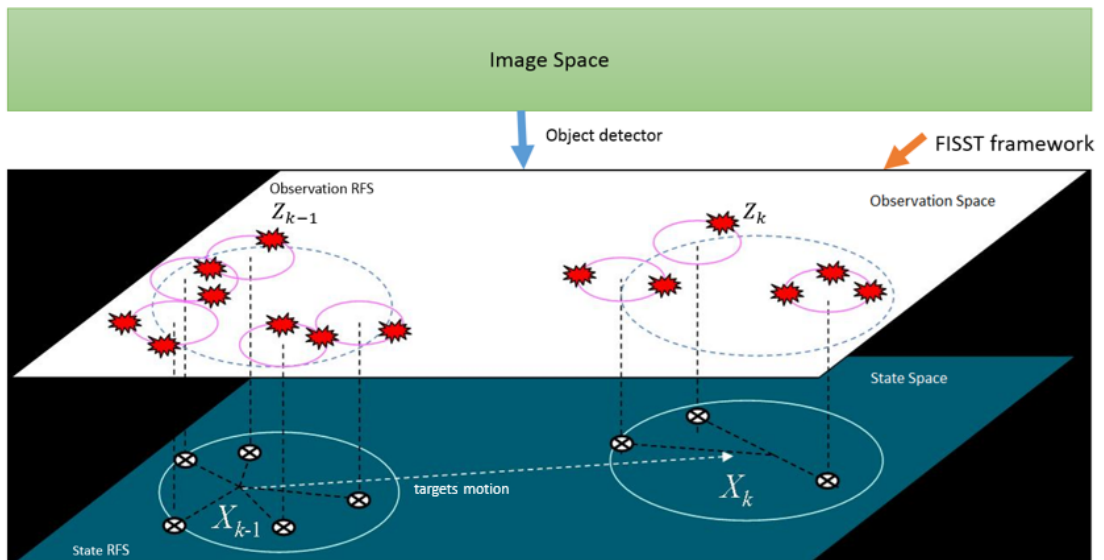


Figure 2.5: FISST framework.

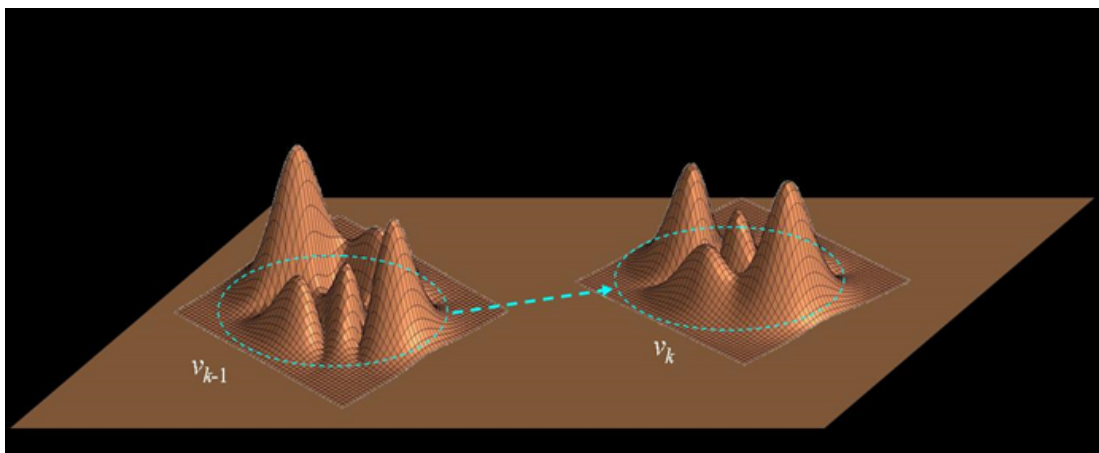


Figure 2.6: Probability hypothesis densities (PHDs) in two consecutive time steps [3].

the circle in Fig. 2.6 shows the most probable region in which targets are located. The PHD filter has linear complexity with the number of targets. However, the PHD recursion involves multiple integrals which requires systematic implementation schemes. Accordingly, there are two PHD implementation methods: by assuming the PHD to be a mixture of Gaussians called a Gaussian Mixture PHD (GM-PHD) [172] for linear (and by extension mildly non-linear) dynamic and observation models and Gaussian stochastic process or by approximating it with the samples generated from a Sequential Monte Carlo process called the Particle-PHD (SMC-PHD) [173] for highly non-linear dynamic and observation models and non-Gaussian stochastic process. The cost for the lower complexity of the PHD filter is the lack of information on the identity of the targets i.e. data association which helps to recognize the same target, among other targets, in consecutive frames, is not included in the PHD filter. While a clustering step is necessary to associate the peaks of the PHD with target states in the Particle-PHD filter [173] [174] though clustering step is avoided in [175] and [176], state extraction in GM-PHD filter is easier as the target state can be associated directly with each Gaussian whose weight is greater than a given threshold [172] [177]. Recently, labeled RFS for multi-target tracking was introduced in [178] [179] [180], however, its computational complexity is high.

A generalization of PHD recursion called Cardinalized PHD (CPHD) recursion jointly propagates the posterior PHD and the posterior cardinality distribution i.e. the posterior distribution of the number of targets [181]. Its Gaussian mixture implementation is given in [182] and its adaptive SMC implementation is given in [176]. Both PHD and CPHD filters and their implementation methods are well discussed with a brief overview of random set theory in [84] [183].

The PHD filter is formulated based on Poisson distributions whereas CPHD filter is formulated using independent and identically distributed (IID) cluster process, an extension of a Poisson (point) process involving the locations of clusters and the locations of elements within a cluster. Another important kind of the RFS-based filters in the FISST framework is represented by Bernoulli distribution called multi-target multi-Bernoulli (MeMber) filter. The cardinality bias of this filter is solved using cardinality balanced multi-target multi-Bernoulli (CB-MeMber) filter in [184] where its implementations using SMC and Gaussian mixture (GM) are given. These RFS-based filters are formulated with the assumption of uncorrelated targets i.e. target interaction is assumed to be negligible. There is an open research for the case of correlated targets in which the interaction of targets can be modeled. A tutorial on Bernoulli filters is given in [185] where its implementation as a SMC or particle filter and Gaussian sum filter are given, and the joint estimation of clutter intensity and detection profile of this

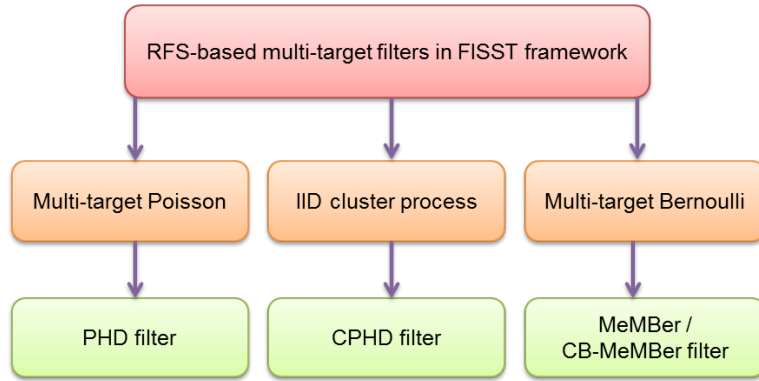


Figure 2.7: RFS-based multi-target filters in the FISST framework.

filter is given in [186]. While PHD and CPHD filters propagate first-order moments (and cardinality distributions for only CPHD filter), the multi-Bernoulli filter propagates the parameters of a multi-Bernoulli distribution which approximate the posterior multi-target density. One advantage of this filter over the SMC-PHD filter is that it avoids the additional clustering step for multi-target state estimation which is its main problem. Multi-Bernoulli filter has the same complexity as the PHD filter i.e. linear in both the number of targets and measurements, however, the complexity of the CPHD filter is linear in the number of targets and cubic in the number of measurements. The general sub-divisions of the RFS-based multi-target filters in the FISST framework is diagrammatically shown in Fig. 2.7.

In a standard model of the above mentioned RFS-based filters, no target generates more than one measurement. The case where one target can generate more than one measurement called extended targets is considered in [187] and its Gaussian mixture implementation is given in [188]. This has got to be relevant in visual tracking as one target may generate more than one (overlapping) detections.

Recently, there have been some multi-target trackers using a PHD filter and its variants in the literature. A SMC-PHD filter is used for visual tracking using graph matching for data association in [146], and scene contextual information is learnt for both birth and clutter intensities and is included in [189], however, a minimal user interaction is required for collecting training data in the scene for learning clutter intensity. In this approach, after the detector is applied to a training set of frames, the user selects the detections that are not associated with objects of interest (clutter) in randomly chosen frames interactively and then learns the clutter intensity using maximum a posterior (MAP). Using a Kalman filter [79] in the prediction step of the particle PHD filter to obtain a new proposal distribution using the latest observation, and using a new observation model by considering both object dynamic states and appearances

for deriving a more precise likelihood for the update step, a more robust multi-target visual tracker was developed in [46].

A method for tracking the movement of multiple cells and their lineage as they spawn was developed using a GM-PHD filter in [190]. The method in [191] uses a GM-PHD filter in which the noise of new birth intensity which may produce short-lived tracks is removed using entropy distribution and coverage-rate. Pedestrians group tracking using a GM-PHD filter was designed in [192] where clustering is applied to the output of the GM-PHD filter and then the level-curves separating the groups in both position and velocity are obtained, however, it only focuses on the group without labeling individual targets in the group. PHD recursion for color measurements was proposed in [193] for tracking pedestrians in a video; a PHD filter is combined with optical flow in [194] to get velocity information rather than estimating indirectly as in the normal PHD filter. PHD filter is robust to clutter but vulnerable to missed detections; this vulnerability is overcome using local observations from an online-trained local detector in addition to observations from off-line trained global detector (HOG) in an RFS of observation model in [195]. These local observations can also provide identity (label information) to each target in the filtering procedure.

The GM-PHD filter is used in [196] for tracking pedestrians in video sequences but there is only one type of target and the motion model is fixed. As an extension, a GM-PHD filter was developed in [197] for maneuvering targets but this employed a Jump Markov System (JMS) that switched between several motion models. The miss-detection problem of the PHD filter is alleviated in the GM-PHD filter implementation [198] and applied to visual tracking in [199]. In contrast, a particle-PHD filter was applied in [146] to allow for more complex motion models, and to cope with variation of scale, which has significant effects not just on object motion but also on the detection process. In addition, the particle PHD filter used in [200] has treated high-confidence (strong) and low-confidence (weak) detections separately where strong detections are used for label propagation and target initialization whereas weak detections only support label propagation (track existing targets). However, this approach produces significant trajectory fragmentation. PHD recursion is reformulated in terms of single-target track hypotheses and a min-cost flow network is solved for trajectory estimation with the addition of computational cost in [201], and it is still susceptible to long-term occlusions though global data association is used.

Multi-target visual tracking was developed by directly processing the whole image data without using an explicit detection (track-before-detect approach) using MeM-Ber filter in [202] [203], however, track-before-detect approaches generally have low

performance when compared to tracking-by-detection approaches (detection + filtering/tracking). Integration of audio and visual information for tracking multiple targets was proposed using a SMC-CB-MeMBeR filter [204] in which the audio and the visual cues are integrated using multiple updates. The performance of RFS-based filters for tracking pedestrians is compared in [205] where a CPHD filter-based visual tracker [145] outperforms the others using Munkres assignment algorithm [73] for data association though it has more computation time. Furthermore, tracking multiple targets from a single camera is extended to multiple cameras to handle occlusion problems and recovering the 3D information of targets using a GM-PHD filter in [206], and similarly a GM-PHD filter is also used for multi-target tracking in a distributed camera network in [207]. Multi-object stereo filtering in disparity space is introduced in FISST framework in [208]. RFS-based filters are not only becoming popular in multi-target visual tracking but also for visual odometry (VO) and simultaneous localization and mapping (SLAM) in robotics as used in [209] and [210], respectively.

Considering extensions to different target types, Yan et al. [211] developed joint detection, tracking and classification (JDTC) of multiple targets in clutter which jointly estimates the number of targets, their kinematic states, and types of targets (classes) from a sequence of noisy and cluttered measurement sets using a SMC-PHD filter. The dynamics of each target type (class) was modeled as a class-dependent model set, and the signal amplitude was included in the multi-target likelihood in the PHD-like filter to enhance the discrimination between targets from different classes and false alarms. Similarly, a joint target tracking and classification (JTC) algorithm was developed in [212] using RFS which takes into account extraneous target-originated measurements (of the same type), i.e. multiple measurements that originated from a target can be modeled as a Poisson RFS using linear and Gaussian assumptions. In these approaches, the augmented state vector of a target comprises the target kinematic state and class label, i.e. the target type (class) is put into the target state vector. However, although multiple target types were considered, no account was taken of the effect of target confusions between target types at the detection stage, as is the case in our work in chapter 4 (modeling and implementation) and chapter 5 (applying to visual tracking).

2.6 Evaluation Metrics for Visual Tracking

In this section, we provide the evaluation metrics for both single and multiple target tracking algorithms that are commonly used in the computer vision community.

2.6.1 Evaluation Metrics for Single Target Tracking

There are two commonly used evaluation metrics for quantitatively evaluating the robustness of single target tracking algorithms, precision and success plots, and are elaborated below. One-pass evaluation (OPE), running the trackers throughout a test sequence with initialization from the ground truth position in the first frame, is the usually used setting for evaluating single target trackers using these metrics.

Precision Plots: Center location error is the widely used metric for tracking precision. It computes the Euclidean distance between the center positions of the tracked targets (\mathbf{x}_t) and the manually labeled ground truth positions (\mathbf{x}_g) of all the frames. The precision plot, percentage of frames where the distance from estimated location to the ground truth is within the given threshold, is a better approach to measure the overall tracking performance [93] than taking the average center location error over all the frames of one sequence as the output location can be random when the tracker loses the target.

Euclidean distance can be computed as

$$D_k(\mathbf{x}_{t,k}, \mathbf{x}_{g,k}) = \sqrt{\sum_{j=1}^2 (x_{t,k,j} - x_{g,k,j})^2}, \quad (2.6)$$

where $k \in \{1, \dots, N_f\}$, N_f is the number of frames of the video sequence.

Then, precision for a series of distance thresholds, for instance from 0 to 50 pixels, can be computed and then plotted to show the overall performance of the single target tracker. Sometimes, it is crucial to show representative precision score for each tracker at a specific threshold, for example, at threshold of 20 pixels, especially for ranking the trackers.

Success Plots: Bounding box overlap is a commonly used metric for evaluation of tracking success [93]. Given the tracked bounding box tb_k and the ground truth bounding box gb_k in frame k , the bounding box overlap score or ratio computes the intersection (\cap) over union (\cup), also known as Jaccard index, of the tracked target and ground truth bounding boxes as

$$S_k = \frac{|gb_k \cap tb_k|}{|gb_k \cup tb_k|}, \quad (2.7)$$

where $|\cdot|$ represents the number of pixels in the region. In this case, the number of successful frames whose overlap S is larger than the given threshold T_o is counted and plotted by varying the thresholds from 0 to 1 to measure the overall tracking performance on frames of a sequence. The tracking algorithms are ranked using area under curve (AUC) of each success plot rather than using one success rate value at a specific threshold, for example $T_o = 0.5$.

2.6.2 Evaluation Metrics for Multi-target Tracking

Evaluating multi-target tracking algorithms is more challenging than single target tracking algorithms. Generally, there are two fundamentally different performance metric types for evaluating multi-target tracking performance: Multi-Object Tracking Accuracy (MOTA), Multi-Object Tracking Precision (MOTP) and others [213]¹ [214], and Optimal SubPattern Assignment (OSPA) [215].

MOTA, MOTP and others: MOTA and MOTP are the two known metrics used for measuring the performance of multi-object visual trackers [213] in the computer vision community. MOTA is given by

$$MOTA = 1 - \frac{\sum_k (m_k + fp_k + mme_k)}{\sum_k g_k}, \quad (2.8)$$

where m_k , fp_k , mme_k and g_k are the number of misses (false negatives), of false positives, of mismatches (identity switches), and of ground truth targets, respectively, for frame k , i.e. MOTA is derived using three error ratios: miss-rate, false positive rate and mismatch rate. The total error is the summation of these error ratios, thus, MOTA is 1 minus this total error.

MOTA considers all the object configuration errors such as misses, false positives and mismatches made by the tracking algorithm, over all frames, and gives a measure of the tracker’s performance at detecting objects as well as keeping track of their trajectories, regardless of the precision of the estimated object locations.

The precision of object locations is given by MOTP which shows the ability of the tracker to estimate precise object positions and can be obtained by computing the

¹Source code is given at <https://github.com/glisanti/CLEAR-MOT>

distance (or bounding box overlap) d_k^i between object (ground truth) o_i and its corresponding hypothesis (true positive) for each of the matched targets

$$MOTP = \frac{\sum_{i,k} d_k^i}{\sum_k c_k}, \quad (2.9)$$

where MOTP is the total error in estimated location for object-hypothesis pairs over all frames averaged by the total number of matches made, and c_k is the number of matched targets found for frame k .

It is crucial to first sum up all errors across frames before a final average or ratio can be computed for both MOTA and MOTP since computing ratios r_k for each frame k independently before computing a global average $(1/n) \sum_k r_k$ for all n frames can lead to nonintuitive results.

There are also other widely used multi-target evaluation metrics, particularly for evaluating the quality of multi-target tracking algorithms along with MOTA and MOTP. These evaluation metrics are Mostly Tracked (MT), Mostly Lost (ML) [214], Fragmented trajectories (Frag), False Positives (FP), False Negatives (FN), False Alarms per frame (FAF), Identity Switches (IDSw), tracker speed (Hz), and are well described and used in [216] [9]. If a target is tracked for at least 80% of its life span regardless of maintaining its ID, it is considered as mostly tracked. However, if it is tracked for less than 20% of its life span, it becomes mostly lost. MT and ML are expressed as a ratio of mostly tracked targets and mostly lost targets to the total number of ground truth trajectories, respectively. Track fragmentation deals with how often a ground truth trajectory is interrupted, and the relative number of fragmentations is given as Frag/Recall. Similar to the fragmentation, the relative number of ID switches is computed as IDSw/Recall.

OSPA: OSPA is a consistent performance evaluation metric for computing miss-distance or error between true and estimated states of multi-object filtering taking into account cardinality error, especially formulated for random finite set (RFS) based filters (e.g. PHD filter) as they jointly estimate both the expected number of targets in the state space and the states of the targets. The OSPA metric is mathematically well-formulated taking both cardinality and distance errors between the true and estimated values of both the number of targets (cardinality) and the states (positions) of the targets. The great advantage of this metric is the ability to compute miss-distance on the space of finite sets, and in turn allows a natural physical interpretation even if the two sets have different cardinality without showing any elements of arbitrariness inherent in ad hoc assignment methods.

Given the distance metric (typically Euclidean metric $d(x, y) = \|x - y\|$), denoted by $d^{(c)}(x, y) := \min(c, d(x, y))$ the distance between $x, y \in \mathcal{X}$ (state space) cut-off at $c > 0$, and by Π_k the set of permutations on $\{1, 2, 3, \dots, k\}$ for any $k \in \mathbb{N} = \{1, 2, 3, \dots\}$. For $1 \leq p < \infty$, $c > 0$, and arbitrary finite subsets $X = \{x_1, x_2, x_3, \dots, x_m\}$ and $Y = \{y_1, y_2, y_3, \dots, y_n\}$ of \mathcal{X} , where $m, n \in \mathbb{N}_0 = \{0, 1, 2, 3, \dots\}$, OSPA metric of order p with cut-off c is defined as

$$d_p^{(c)}(X, Y) := \left(\frac{1}{n} \left(\min_{\pi \in \Pi_n} \sum_{i=1}^m d^{(c)}(x_i, y_{\pi(i)})^p + c^p(n - m) \right) \right)^p, \quad (2.10)$$

There are two adjustable parameters in the OSPA metric, order parameter $p \in [1, \infty)$ for outlier sensitivity and cut-off parameter $c > 0$ for cardinality penalty. There are two important choices of p : $p = 1$ and $p = 2$. For $p = 1$, the OSPA metric measures a first-order per target error in which the sum of the localization and cardinality components equals the total metric. However, $p = 2$ is usually a more practical choice since it gives smooth distance curves. If it is important to estimate the number of targets correctly, choose large values of c whereas if accurate position estimates are necessary and cardinality errors are negligible, small values of c should be chosen. For its details both in concepts and mathematical formulations, it is recommended to refer to [215]. A modified version of this original OSPA metric called Optimal Subpattern Assignment for Tracks (OSPA-T) includes labeling error $\alpha \in [0, c]$ in addition to both distance and cardinality errors for evaluating the performance of multi-target visual tracking algorithms [217] i.e. if $\alpha = 0$, OSPA-T becomes the original OSPA metric. It has been successfully used for evaluating pedestrian visual tracking algorithms in [205].

Chapter 3

Long-term Correlation Tracking using Multi-layer Hybrid Features

In section 2.4, an overview of methods for tracking a target of interest in video sequences was discussed using different visual features and approaches. However, tracking methods which work fine in sparse videos may not work well on dense environments and vice versa as they are mostly designed taking the video scenes into account. In this chapter, we mainly focus on long-term tracking of a target of interest in sparse as well as crowded environments where an unknown target is initialized by a bounding box and then is tracked in subsequent frames. Without allowing any constraint on the video scene, we develop a novel long-term online tracking algorithm that can close the research gap between sparse and crowded scenes tracking problems using the advantages of correlation filters, a hybrid of multi-layer convolutional neural network (CNN) and traditional hand-crafted features, an online support vector machine (SVM) classifier and a Gaussian mixture probability hypothesis density (GM-PHD) filter that is robust to long-term occlusion and challenging clutter from the background scene as well as other targets of no interest. Accordingly, an overview of the proposed algorithm is described in section 3.1, discussed in detail in section 3.2, implementation details with parameter settings are described in section 3.3, and experimental results on both sparse and crowded environment data sets are analyzed and compared in section 3.4. Finally, a brief summary of the developed Long-term Correlation Multi-layer Hybrid Tracker (LCMHT) is given in section 3.5.

3.1 Overview of our Algorithm

CNN features have recently demonstrated outstanding results on various recognition tasks though traditional hand-engineered features are still important. Similarly, correlation filters are giving better results for online tracking problems in both efficiency and accuracy [8]. Besides, the GM-PHD filter [172] has an in-built capability of removing clutter that originates from both the background scene and other targets not of current concern while filtering targets at high speed without the need for explicit data association. Though, this filter is designed for multi-target filtering, it is even preferable for single target filtering. In this work, we develop a unique long-term online tracking algorithm that can be applied to both sparse and dense environments by learning correlation filters using a hybrid of multi-layer CNN and hand-crafted features as well as including a re-detection module using an incremental SVM and GM-PHD filter.

Accordingly, to develop an online long-term tracking algorithm robust to appearance variations in both sparse and crowded scenes, we learn two different correlation filters: a translation correlation filter (\mathbf{w}_t) and a scale correlation filter (\mathbf{w}_s). A translation correlation filter is learned using a hybrid of multi-layer CNN features from VGG-Net [4] and traditional hand-crafted features.

For the CNN part, we combine features from both a lower convolutional layer which retains more spatial details for precise localization and a higher convolutional layer which encodes semantic information for handling appearance variations. This makes layer 1, layer 2, and layer 3 in multi-layer features with multiple channels (512, 512 and 256 dimensions) in each layer, respectively. Since the spatial resolution of the extracted features gradually reduces with the increase of the depth of CNN layers due to pooling operators, it is crucial to resize each feature map to a fixed size using bilinear interpolation.

For the traditional features part, we use a histogram of oriented gradients (HOG), in particular Felzenszwalb’s variant [70] and color-naming [34] features for capturing image gradients and color information, respectively. These integrated traditional features have been used for object detection in [218] [219] giving promising results. Color-naming is the linguistic color label assigned by human to describe the color, hence, the mapping method in [34] is employed to convert the RGB space into the color name space which is an 11 dimensional color representation providing the perception of a target color. This associates RGB pixel values with color labels to transform RGB values into a probabilistic 11 dimensional color representation i.e. the probability of color names corresponding to image pixels is computed using the estimated/learned

generative models (distributions representing the color names) from real-world images (Google image search). These color names contain 11 basic color terms and are given in order as black, blue, brown, grey, green, orange, pink, purple, red, white and yellow. This color naming space is better than RGB space in expressing the color of real-world objects though it is computationally more expensive to process than the RGB data. However, in our case, since correlation operation is performed in frequency domain with a very fast speed, its computational complexity is not a much concern. Thus, we choose this color naming due to these two factors. By aligning the feature size of the HOG variant with 31 dimensions and color-naming with 11 dimensions, they are integrated to make 42 dimensional features which make a 4th layer in our hybrid multi-layer features.

For scale estimation, we learn a scale correlation filter using only HOG features, in particular Felzenszwalb’s variant [70]. Besides, we incorporate a re-detection module by learning an incremental SVM from the most confident frames determined by maximal value of correlation response map using HOG, LUV color and normalized gradient magnitude features for generating high-score detection proposals which are filtered using the GM-PHD filter to re-acquire the target in case of tracking failures. The flowchart of our method is given in Fig. 3.1 and the outline of our proposed algorithm is given in Algorithm 1.

3.2 Proposed Algorithm

This section describes our proposed tracking algorithm which has four distinct functional parts: 1) correlation filters formulated for multi-layer features, 2) an online SVM detector developed for generating high score detection proposals, 3) a GM-PHD filter for finding the detection proposal with the maximum weight to re-initialize the tracker in case of tracking failures by removing the other detection proposals as clutter, and 4) a scale estimation method for estimating the scale of a target by constructing image pyramid at the estimated target position.

3.2.1 Correlation Filters for Multi-layer Features

To track a target using correlation filters, the appearance of the target should be modeled using a correlation filter \mathbf{w} which can be trained on feature vector \mathbf{x} of size $M \times N \times D$ extracted from an image patch where M , N , and D indicates the width,

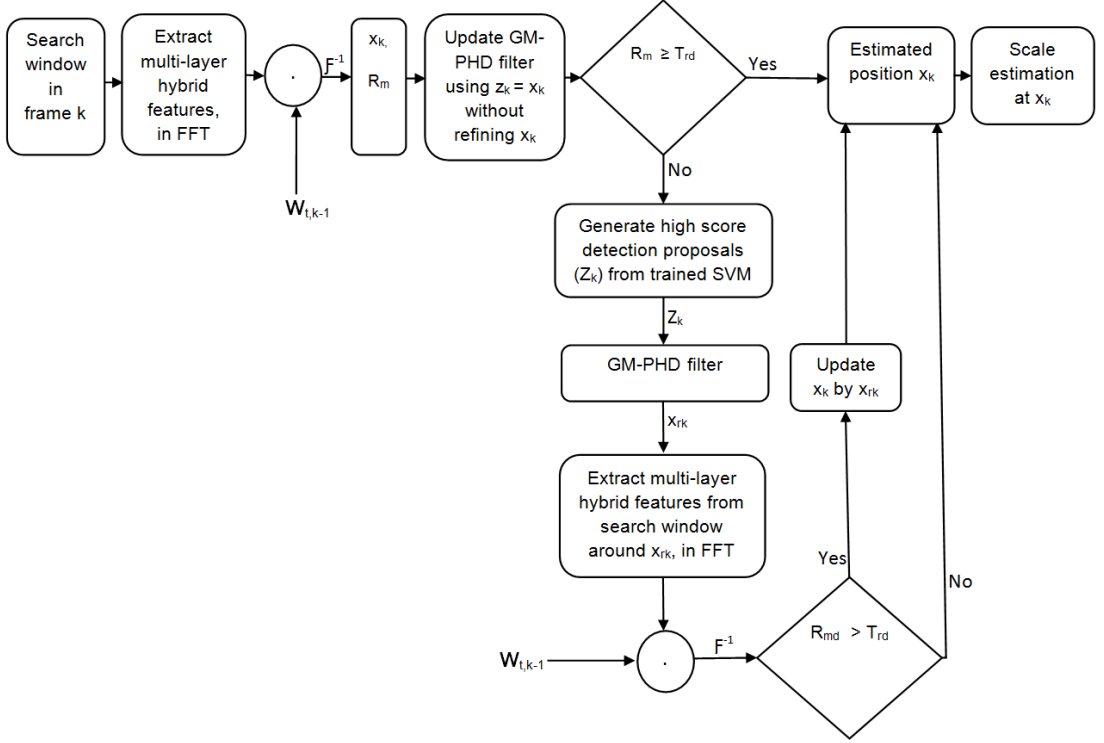


Figure 3.1: The flowchart of the proposed algorithm. It consists of three main parts: translation estimation, re-detection and scale estimation. Given a search window, we extract multi-layer hybrid features (in the frequency domain) and then estimate target position (\mathbf{x}_k) using a translation correlation filter (\mathbf{w}_t). This estimated position (\mathbf{x}_k) is used as a measurement (\mathbf{z}_k) for updating the GM-PHD filter without refining \mathbf{x}_k , just to update its weight for later use during re-detection. Re-detection is activated if the maximum of the response map (R_m) becomes below the pre-defined threshold (T_{rd}). Then, we generate high score detection proposals (Z_k) which are filtered by the GM-PHD filter to estimate the detection with maximum weight as target position (\mathbf{x}_{rk}) removing the others as clutter. If the response map around \mathbf{x}_{rk} (R_{md}) is greater than T_{rd} , the target position \mathbf{x}_k is updated by the re-detected position \mathbf{x}_{rk} . Finally, we estimate the scale of the target by constructing a target pyramid at the estimated position and use the scale correlation filter (\mathbf{w}_s) to find the scale at which the maximum response map is obtained. Note that in frame 1, we only train correlation filters and the SVM classifier using the initialized target; no detection is performed.

height and number of channels, respectively. This feature vector \mathbf{x} can be extracted from multiple layers, for example in the case of CNN features and/or traditional hand-crafted features, therefore, we denote it as $\mathbf{x}^{(l)}$ to designate which layer l it represents in our multi-layer features. All the circular shifts of $\mathbf{x}^{(l)}$ along the M and N dimensions are considered as training examples where each circularly shifted sample $\mathbf{x}_{m,n}^{(l)}$, $m \in \{0, 1, \dots, M-1\}$, $n \in \{0, 1, \dots, N-1\}$ has a Gaussian function label $y^{(l)}(m, n)$ given by

$$y^{(l)}(m, n) = e^{-\frac{(m-M/2)^2 + (n-N/2)^2}{2\sigma^2}}, \quad (3.1)$$

where σ is the kernel width. Hence, $y^{(l)}(m, n)$ is a soft label rather than a binary label. To learn the correlation filter $\mathbf{w}^{(l)}$ for layer l with the same size as $\mathbf{x}^{(l)}$, we extend a ridge regression [220] [63], developed for a single-layer feature vector, to be used for multi-layer hybrid features with layer l , $\mathbf{x}^{(l)}$, as

$$\min_{\mathbf{w}^{(l)}} \sum_{m,n} |\Phi(\mathbf{x}^{(l)}) \cdot \mathbf{w}^{(l)} - y^{(l)}(m, n)|^2 + \lambda |\mathbf{w}^{(l)}|^2, \quad (3.2)$$

where Φ represents the mapping to a kernel space and λ is a regularization parameter ($\lambda \geq 0$). The solution $\mathbf{w}^{(l)}$ can be expressed as

$$\mathbf{w}^{(l)} = \sum_{m,n} \mathbf{a}^{(l)}(m, n) \Phi(\mathbf{x}_{m,n}^{(l)}). \quad (3.3)$$

This alternative representation makes the dual space $\mathbf{a}^{(l)}$ the variable under optimization instead of the primal space $\mathbf{w}^{(l)}$.

Training phase: The training phase is performed in the Fourier domain using the fast Fourier transform (FFT) to compute the coefficient $\mathbf{A}^{(l)}$ as

$$\mathbf{A}^{(l)} = \mathcal{F}(\mathbf{a}^{(l)}) = \frac{\mathcal{F}(\mathbf{y}^{(l)})}{\mathcal{F}(\Phi(\mathbf{x}^{(l)}) \cdot \Phi(\mathbf{x}^{(l)})) + \lambda}, \quad (3.4)$$

where \mathcal{F} denotes the FFT operator.

Detection phase: The detection phase is performed on the new frame given an image patch (search window) which is used as a temporal context i.e. the search window is larger than the target to provide some context. If feature vector $\mathbf{z}^{(l)}$ of size $M \times N \times D$ is extracted from this image patch, the response map ($\mathbf{r}^{(l)}$) is computed as

$$\mathbf{r}^{(l)} = \mathcal{F}^{-1}(\tilde{\mathbf{A}}^{(l)} \odot \mathcal{F}(\Phi(\mathbf{z}^{(l)}) \cdot \Phi(\tilde{\mathbf{x}}^{(l)}))), \quad (3.5)$$

where $\tilde{\mathbf{A}}^{(l)}$ and $\tilde{\mathbf{x}}^{(l)} = \mathcal{F}^{-1}(\tilde{\mathbf{X}}^{(l)})$ denote the learned target appearance model for layer l , operator \odot is the Hadamard (element-wise) product, and \mathcal{F}^{-1} is the inverse FFT. Now, the response maps of all layers are summed according to their weight $\gamma(l)$ element-wise as

$$\mathbf{r}(m, n) = \sum_l \gamma(l) \mathbf{r}^{(l)}(m, n). \quad (3.6)$$

The new target position is estimated by finding the maximum value of $\mathbf{r}(m, n)$ as

$$(\hat{m}, \hat{n}) = \underset{m, n}{\operatorname{argmax}} \mathbf{r}(m, n). \quad (3.7)$$

Model update: The model is updated by training a new model at the new target position and then linearly interpolating the obtained values of the dual space coefficients $\mathbf{A}_k^{(l)}$ and the base data template $\mathbf{X}_k^{(l)} = \mathcal{F}(\mathbf{x}_k^{(l)})$ with those from the previous frame to make the tracker more adaptive to target appearance variations.

$$\tilde{\mathbf{X}}_k^{(l)} = (1 - \eta)\tilde{\mathbf{X}}_{k-1}^{(l)} + \eta\mathbf{X}_k^{(l)}, \quad (3.8a)$$

$$\tilde{\mathbf{A}}_k^{(l)} = (1 - \eta)\tilde{\mathbf{A}}_{k-1}^{(l)} + \eta\mathbf{A}_k^{(l)}, \quad (3.8b)$$

where k is the index of the current frame, and η is the learning rate.

The mappings to the kernel space (Φ) used in Eq. (3.4) and Eq. (3.5) can be expressed using a kernel function as $k(\mathbf{x}_i^{(l)}, \mathbf{x}_j^{(l)}) = \Phi(\mathbf{x}_i^{(l)}) \cdot \Phi(\mathbf{x}_j^{(l)}) = \Phi(\mathbf{x}_i^{(l)})^T \Phi(\mathbf{x}_j^{(l)})$. If the computation is performed in the frequency domain, the normal transpose should be replaced by the Hermitian transpose i.e. $\Phi(\mathbf{X}_i^{(l)})^H = (\Phi(\mathbf{X}_i^{(l)})^*)^T$ where the star ($*$) denotes the complex conjugate.

Thus, for a linear kernel,

$$k(\mathbf{x}_i^{(l)}, \mathbf{x}_j^{(l)}) = (\mathbf{x}_i^{(l)})^T \mathbf{x}_j^{(l)} = \mathcal{F}^{-1} \left(\sum_d (\mathbf{X}_{i,d}^{(l)})^* \odot \mathbf{X}_{j,d}^{(l)} \right), \quad (3.9)$$

where $\mathbf{X}_i^{(l)} = \mathcal{F}(\mathbf{x}_i^{(l)})$.

and for a Gaussian kernel,

$$\begin{aligned} k(\mathbf{x}_i^{(l)}, \mathbf{x}_j^{(l)}) &= \Phi(\mathbf{x}_i^{(l)})^T \Phi(\mathbf{x}_j^{(l)}) = \exp \left(-\frac{1}{\sigma^2} (\|\mathbf{x}_i^{(l)} - \mathbf{x}_j^{(l)}\|^2) \right) \\ &= \exp \left(-\frac{1}{\sigma^2} (\|\mathbf{x}_i^{(l)}\|^2 + \|\mathbf{x}_j^{(l)}\|^2 - \mathcal{F}^{-1}(\sum_d (\mathbf{X}_{i,d}^{(l)})^* \odot \mathbf{X}_{j,d}^{(l)})) \right). \end{aligned} \quad (3.10)$$

This formulation is generic for multiple channel features from multiple layers as in the case of our multi-layer hybrid features, i.e. where $\mathbf{X}_{i,d}^{(l)}$, $d \in \{1, \dots, D\}$, $l \in \{1, \dots, L\}$. This is an extended version of the one given in [8] that takes into account features from multiple layers. The linearity of the FFT allows us to simply sum the individual dot-products for each channel $d \in \{1, \dots, D\}$ in each layer $l \in \{1, \dots, L\}$.

All operations in Eq. (3.4) are element-wise ($O(n)$) with the exception of the DFT which bounds the cost at a nearly-linear $O(n \log n)$ for both linear and other kernel functions, thus, it reduces storage and computation by a large margin when compared to extracting patches explicitly and solving ridge regression, for example, at cost of $O(n^3)$.

3.2.2 Online Detector

We include a re-detection module, D_r , to generate high score detection proposals in case of tracking failures due to long-term occlusions. Instead of using a correlation filter to scan across the entire frame which is computationally expensive and less efficient, we learn an incremental (online) SVM [221] by generating a set of samples in the search window around the estimated target position from the most confident frames, and scan through the window when the re-detection is activated to generate high-score detection proposals. These most confident frames are determined by the maximum translation correlation response in the current frame i.e. if the maximum correlation response of an image patch is above the train detector threshold (T_{td}), we generate samples around this image patch and train the detector. This detector is activated to generate high score detection proposals if the maximum of the correlation response becomes below the activate re-detection threshold (T_{rd}). We use HOG (particularly Felzenszwalb’s variant [70]), LUV color and normalized gradient magnitude features to train this online SVM classifier. We use different visual features for computational feasibility from the ones we use for learning the translation correlation filter since we can select the feature representation for each module independently [7, 111].

We want to update a weight vector \mathbf{w} of the SVM provided a set of samples with associated labels, $\{(\hat{\mathbf{x}}_i, \hat{y}_i)\}$, obtained from the current results. The label \hat{y}_i of a new example $\hat{\mathbf{x}}_i$ is given by

$$\hat{y}_i = \begin{cases} +1, & \text{if } IOU(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_t) \geq \delta_p \\ -1, & \text{if } IOU(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_t) < \delta_n \end{cases} \quad (3.11)$$

where $IOU(.)$ is the intersection over union (overlap ratio) of a new example $\hat{\mathbf{x}}_i$ and the estimated target bounding box in the current most confident frame $\hat{\mathbf{x}}_t$. The samples with the bounding box overlap ratios between the thresholds δ_n and δ_p are excluded from the training set for avoiding drift problem.

Before delving into incremental SVM, it is crucial to give a summary of the optimization procedure for offline SVM learning algorithm. SVM classifiers of the form $f(\mathbf{x}) = \mathbf{w} \cdot \Phi(\mathbf{x}) + b$ are learned from the data $\{(\mathbf{x}_i, \mathbf{y}_i) \in \mathbb{R}^m \times \{-1, +1\} \forall i \in \{1, \dots, N\}\}$ by minimizing

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i^p \quad (3.12)$$

for $p \in \{1, 2\}$ subject to the constraints

$$y_i(\mathbf{w} \cdot \Phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \xi_i \geq 0 \forall i \in \{1, \dots, N\}. \quad (3.13)$$

Hinge loss ($p = 1$) is preferred due to its improved robustness to outliers over the quadratic loss ($p = 2$). Thus, the offline SVM learns a weight vector $\mathbf{w} = (w_1, w_2, \dots, w_N)^T$ by solving this quadratic convex optimization problem (QP) which can be expressed in its dual form as

$$\min_{0 \leq a_i \leq C} W = \frac{1}{2} \sum_{i,j=1}^N a_i Q_{ij} a_j - \sum_{i=1}^N a_i + b \sum_{i=1}^N y_i a_i, \quad (3.14)$$

where $\{a_i\}$ are Lagrange multipliers, b is bias, C is a regularization parameter, and $Q_{ij} = y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$. The kernel function $k(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$ is used to implicitly map into a higher dimensional feature space and compute the dot product. It is not straightforward for conventional QP solvers to handle the optimization problem in Eq. (3.14) for online tracking tasks as the training data are provided sequentially, not at once. Incremental SVM [221] is tailored for such cases which retains Karush-Kuhn-Tucker (KKT) conditions on all the existing examples while updating the model with a new example so that the exact solution at each increment of dataset can be guaranteed. KKT conditions are the first-order necessary conditions for the optimal unique solution of dual parameters $\{a, b\}$ which minimizes Eq. (3.14) and are given by

$$\frac{\partial W}{\partial a_i} = \sum_{j=1}^N Q_{ij} a_j + y_i b - 1 \begin{cases} > 0, \text{ if } a_i = 0 \\ = 0, \text{ if } 0 < a_i < C \\ < 0, \text{ if } a_i = C, \end{cases} \quad (3.15)$$

$$\frac{\partial W}{\partial b} = \sum_{j=1}^N y_j a_j = 0. \quad (3.16)$$

Based on the partial derivative $m_i = \frac{\partial W}{\partial a_i}$ which is related to the margin of the i -th

example, each training example can be categorized into three: \mathcal{S}_1 support vectors lying on the margin ($m_i = 0$), \mathcal{S}_2 support vectors lying inside the margin ($m_i < 0$), and the remaining \mathcal{R} reserve vectors (non-support vectors). During incremental learning, new examples with $m_i \leq 0$ eventually become margin (\mathcal{S}_1) or error (\mathcal{S}_2) support vectors. However, the remaining new training examples become reserve vectors as they do not enter the solution so that the Lagrangian multipliers (a_i) are estimated while retaining the KKT conditions. Given the updated Lagrangian multipliers, the weight vector \mathbf{w} is given by

$$\mathbf{w} = \sum_{i \in \mathcal{S}_1 \cup \mathcal{S}_2} a_i y_i \Phi(\mathbf{x}_i). \quad (3.17)$$

It is important to keep only a fixed number of support vectors with the smallest margins for efficiency during online tracking.

Thus, using the trained incremental SVM, we generate high score detections as detection proposals during the re-detection stage. These are filtered using the GM-PHD filter to find the best possible detection that can re-initialize the tracker.

3.2.3 Temporal Filtering using the GM-PHD Filter

Once we generate high score detection proposals using the learned online SVM classifier during the re-detection stage, we need to find the most probable detection proposal for the target state (position) estimate by finding the detection proposal with the maximum weight using the GM-PHD filter [172]. Though the GM-PHD filter is designed for multi-target filtering with the assumptions of a linear Gaussian system, in our problem (re-detecting a target in cluttered scene), it is used for removing clutter that comes from background scene and other targets not of interest as it is equipped with such a capability. Besides, it provides motion information for the tracking algorithm. More importantly, using the GM-PHD filter to find the detection with the maximum weight from the generated high-score detection proposals is more robust than relying only on the maximum score of the classifier.

The detected position of the target in each frame is filtered using the GM-PHD filter, but without re-fining the position states until the re-detection module is activated. This updates the weight of the GM-PHD filter corresponding to a target of interest giving sufficient prior information to be picked up during the re-detection stage among candidate high score detection proposals. If the re-detection module is activated (cor-

relation response of the target becomes below a pre-defined threshold), we generate a high score detection proposals (in this case 5) from the trained SVM classifier which are then filtered using the GM-PHD filter. The Gaussian component with the maximum weight is selected as position estimate, and if the correlation response of this estimated position is greater than the pre-defined threshold, the estimated position of the target is re-fined.

The GM-PHD filter has two steps: prediction and update. Before stating these two steps, certain assumptions are needed: 1) each target follows a linear Gaussian model:

$$y_{k|k-1}(x|\zeta) = \mathcal{N}(x; F_{k-1}\zeta, Q_{k-1}) \quad (3.18)$$

$$f_k(z|x) = \mathcal{N}(z; H_k x, R_k) \quad (3.19)$$

where $\mathcal{N}(\cdot; m, P)$ denotes a Gaussian density with mean m and covariance P ; F_{k-1} and H_k are the state transition and measurement matrices, respectively. Q_{k-1} and R_k are the covariance matrices of the process and the measurement noises, respectively. 2) A current measurement driven birth intensity inspired by but not identical to [176] is introduced at each time step, removing the need for the prior knowledge (specification of birth intensities) or a random model, with a non-informative zero initial velocity. The intensity of the spontaneous birth RFS is a Gaussian mixture of the form

$$\gamma_k(x) = \sum_{v=1}^{V_{\gamma,k}} w_{\gamma,k}^{(v)} \mathcal{N}(x; m_{\gamma,k}^{(v)}, P_{\gamma,k}^{(v)}) \quad (3.20)$$

where $V_{\gamma,k}$ is the number of birth Gaussian components, $w_{\gamma,k}^{(v)}$ is the weight accompanying the Gaussian component v , $m_{\gamma,k}^{(v)}$ is the current measurement and zero initial velocity used as mean, and $P_{\gamma,k}^{(v)}$ is birth covariance for Gaussian component v . In our case, $V_{\gamma,k}$ equals to 1 unless in re-detection stage at which it becomes 5 as we generate 5 high score detection proposals to be filtered.

3) The survival and detection probabilities are independent of the target state: $p_{S,k}(x_k) = p_{S,k}$ and $p_{D,k}(x_k) = p_{D,k}$.

Prediction: It is assumed that the posterior intensity at time $k - 1$ is a Gaussian

mixture of the form

$$\mathcal{D}_{k-1}(x) = \mathcal{D}_{k-1|k-1}(x) = \sum_{v=1}^{V_{k-1}} w_{k-1}^{(v)} \mathcal{N}(x; m_{k-1}^{(v)}, P_{k-1}^{(v)}), \quad (3.21)$$

where V_{k-1} is the number of Gaussian components of $\mathcal{D}_{k-1}(x)$ and it equals to the number of Gaussian components after pruning and merging at the previous iteration. Under these assumptions, the predicted intensity at time k is given by

$$\mathcal{D}_{k|k-1}(x) = \mathcal{D}_{S,k|k-1}(x) + \gamma_k(x), \quad (3.22)$$

where

$$\mathcal{D}_{S,k|k-1}(x) = p_{S,k} \sum_{v=1}^{V_{k-1}} w_{k-1}^{(v)} \mathcal{N}(x; m_{S,k|k-1}^{(v)}, P_{S,k|k-1}^{(v)}),$$

$$m_{S,k|k-1}^{(v)} = F_{k-1} m_{k-1}^{(v)},$$

$$P_{S,k|k-1}^{(v)} = Q_{k-1} + F_{k-1} P_{k-1}^{(v)} F_{k-1}^T,$$

where $\gamma_k(x)$ is given by Eq. (3.20).

Since $\mathcal{D}_{S,k|k-1}(x)$ and $\gamma_k(x)$ are Gaussian mixtures, $\mathcal{D}_{k|k-1}(x)$ can be expressed as a Gaussian mixture of the form

$$\mathcal{D}_{k|k-1}(x) = \sum_{v=1}^{V_{k|k-1}} w_{k|k-1}^{(v)} \mathcal{N}(x; m_{k|k-1}^{(v)}, P_{k|k-1}^{(v)}), \quad (3.23)$$

where $w_{k|k-1}^{(v)}$ is the weight accompanying the predicted Gaussian component v , and $V_{k|k-1}$ is the number of predicted Gaussian components and it equals to the number of born targets (1 unless in case of re-detection at which it is 5) and the number of persistent components which are actually the number of Gaussian components after pruning and merging at the previous iteration.

Update: The posterior intensity (updated PHD) at time k is also a Gaussian mixture and is given by

$$\mathcal{D}_{k|k}(x) = (1 - p_{D,k})\mathcal{D}_{k|k-1}(x) + \sum_{z \in Z_k} \mathcal{D}_{D,k}(x; z), \quad (3.24)$$

where

$$\mathcal{D}_{D,k}(x; z) = \sum_{v=1}^{V_{k|k-1}} w_k^{(v)}(z) \mathcal{N}(x; m_{k|k}^{(v)}(z), P_{k|k}^{(v)}),$$

$$w_k^{(v)}(z) = \frac{p_{D,k} w_{k|k-1}^{(v)} q_k^{(v)}(z)}{c_{s_k}(z) + p_{D,k} \sum_{l=1}^{V_{k|k-1}} w_{k|k-1}^{(l)} q_k^{(l)}(z)},$$

$$q_k^{(v)}(z) = \mathcal{N}(z; H_k m_{k|k-1}^{(v)}, R_k + H_k P_{k|k-1}^{(v)} H_k^T),$$

$$m_{k|k}^{(v)}(z) = m_{k|k-1}^{(v)} + K_k^{(v)}(z - H_k m_{k|k-1}^{(v)}),$$

$$P_{k|k}^{(v)} = [I - K_k^{(v)} H_k] P_{k|k-1}^{(v)},$$

$$K_k^{(v)} = P_{k|k-1}^{(v)} H_k^T [H_k P_{k|k-1}^{(v)} H_k^T + R_k]^{-1}$$

The clutter intensity due to the scene, $c_{s_k}(z)$, in Eq. (3.24) is given by

$$c_{s_k}(z) = \lambda_t c(z) = \lambda_c A c(z), \quad (3.25)$$

where $c(\cdot)$ is the uniform density over the surveillance region A , and λ_c is the average number of clutter returns per unit volume i.e. $\lambda_t = \lambda_c A$. We set the clutter rate or false positive per image (fppi) $\lambda_t = 4$ in our experiment.

After update, weak Gaussian components with weight $w_k^{(v)} < T = 10^{-5}$ are pruned, and Gaussian components with Mahalanobis distance less than $U = 4$ pixels from each other are merged. These pruned and merged Gaussian components are predicted as

existing targets in the next iteration. Finally, the Gaussian component of the posterior intensity with mean corresponding to the maximum weight is selected as a target state (position) estimate when the re-detection module is activated.

The chosen values of T and U work reasonably in this experiment. The weights corresponding to the real targets are significantly higher than the value used for T usually approaching unity. Only the duplicated and weak Gaussian components that merely add computationally complexity as time progresses have lower value than that of T . We generate only 5 detection proposals when re-detection module is activated. More importantly, we only pick one target with the maximum weight from the posterior intensity, not from the pruned and merged intensity; the merged and pruned Gaussian components are used for prediction as existing targets in the next iteration. Moreover, in case two Gaussian components corresponding to two different targets are merged (less than U), the measurements obtained in the current time-step can influence them to recover in the next time-step as their separation increases (in general case).

3.2.4 Scale Estimation

At the new estimated target position (or re-fined target position after re-detection in case of tracking failure), we construct an image pyramid for estimating its scale. Given a target size of $P \times Q$ in a test frame, we generate S number of scale levels at the new estimated position i.e. for each $n \in \{\lfloor -\frac{S-1}{2} \rfloor, \lfloor -\frac{S-3}{2} \rfloor, \dots, \lfloor \frac{S-1}{2} \rfloor\}$, we extract an image patch I_s of size $sP \times sQ$ centered at the new estimated target position, where scale $s = a^n$ and a is the scale factor between the generated image pyramids. We uniformly resize all the generated image pyramids to $P \times Q$ again unlike [111], and extracted HOG features particularly Felzenszwalb’s variant [70] to construct the scale feature pyramid. Then, the optimal scale \hat{s} of a target at the estimated new position can be obtained by computing the correlation response maps $\hat{\mathbf{r}}_s$ of the scale correlation filter \mathbf{w}_s to I_s and find the scale at which the maximum response map can be obtained as

$$\hat{s} = \underset{s}{\operatorname{argmax}}(\hat{\mathbf{r}}_s). \quad (3.26)$$

The scale correlation filter is updated using the new training sample at the estimated scale $I_{\hat{s}}$ by Eq. (3.8).

Algorithm 1 Proposed tracking algorithm

```
1: Input : Image  $\mathbf{I}_k$ , previous target position  $\mathbf{x}_{k-1}$  and scale  $s_{k-1}$ , previous correlation filters  $\mathbf{w}_{t,k-1}^{(l)}$ 
    and  $\mathbf{w}_{s,k-1}$ , previous SVM detector  $D_r$ 
2: Output : Estimated target position  $\mathbf{x}_k = (x_k, y_k)$  and scale  $s_k$ , updated correlation filters  $\mathbf{w}_{t,k}^{(l)}$ 
    and  $\mathbf{w}_{s,k}$ , updated SVM detector  $D_r$ 
3: repeat
4:   Crop out the searching window in frame k according to  $(x_{k-1}, y_{k-1})$  and  $s_{k-1}$ , and then extract
    multi-layer hybrid features and resize them to a fixed size;

    // Translation estimation
5:   for each layer l do
6:     compute response map  $\mathbf{r}^{(l)}$  using  $\mathbf{w}_{t,k-1}^{(l)}$  and Eq. (3.5);
7:   end for
8:   Sum up the response maps of all layers element-wise according to their weight  $\gamma(l)$  to get
     $\mathbf{r}(m, n)$  using Eq. (3.6);
9:   Estimate the new target position  $(x_k, y_k)$  by finding the maximum response of  $\mathbf{r}(m, n)$ 
    using Eq. (3.7);

    // Apply GM-PHD filter
10:  Update GM-PHD filter using the estimated target position  $(x_k, y_k)$  as measurement but
    without re-fining it, just to update weight of GM-PHD filter for later use;

    // Target re-detection
11:  if  $\max(\mathbf{r}(m, n)) < T_{rd}$  then
12:    Use the detector  $D_r$  to generate detection proposals  $Z_k$  from high scores of incremental
    SVM;

    // Filtering using GM-PHD filter
13:  Filter the generated candidate detections  $Z_k$  using GM-PHD filter and select the detection
    with maximum weight as a re-detected target position  $(x_{rk}, y_{rk})$ . Then crop out the
    searching window at this re-detected position and compute its response map using Eq. (3.5)
    and Eq. (3.6), and call it  $\mathbf{r}_{rd}(m, n)$ ;
14:  if  $\max(\mathbf{r}_{rd}(m, n)) > T_{rd}$  then
15:     $(x_k, y_k) = (x_{rk}, y_{rk})$  i.e. re-fine by the re-detected position;
16:  end if
17: end if

    // Scale estimation
18:  Construct target image pyramid around  $(x_k, y_k)$  and extract HOG features (resized to same
    size), and then compute the response maps  $\hat{\mathbf{r}}_s$  using  $\mathbf{w}_{s,k-1}$  and Eq. (3.5), and then estimate
    its scale  $s_k$  using Eq. (3.26);

    // Translation correlation model update
19:  Crop out new patch centered at  $(x_k, y_k)$  and extract multi-layer hybrid features and resize
    them to a fixed size;
20:  for each layer l do
21:    Update translation correlation filter  $\mathbf{w}_{t,k}^{(l)}$  using Eq. (3.8);
22:  end for

    // Scale correlation model update
23:  Crop out new patch centered at  $(x_k, y_k)$  with estimated scale  $s_k$  and extract HOG features
    and then update correlation filter  $\mathbf{w}_{s,k}$  using Eq. (3.8);

    // Detector update
24:  if  $\max(\mathbf{r}(m, n)) \geq T_{td}$  then
25:    Generate positive and negative samples around  $(x_k, y_k)$  and then extract HOG, LUV color
    and normalized gradient magnitude features to train incremental SVM for updating its
    weight vector using Eq. (3.17);
26:  end if
27: until End of video sequences;
```

3.3 Implementation Details

The main steps of our proposed algorithm are presented in Algorithm 1. More implementation details with parameter settings are given as follows. For learning the translation correlation filter, we extract features from VGG-Net [4], shown in Fig. 3.2, trained on a large amount of object recognition data set (ImageNet) [18] by first removing fully-connected layers. Particularly, we use outputs of *conv3-4*, *conv4-4* and *conv5-4* convolutional layers as features ($l \in \{1, 2, 3\}$ and $d \in \{1, \dots, D\}$), i.e. the outputs of rectilinear units (inputs of pooling) layers must be used to keep more spatial resolution. Hence, the CNN features we use has 3 layers ($L = 3$) and multiple channels ($D = 512$) for *conv5-4* and *conv4-4* layers and ($D = 256$) for *conv3-4* layer. For hand-crafted features, HOG variant with 31 dimensions and color-naming with 11 dimensions are integrated to make 42 dimensional features which make a 4th layer in our hybrid multi-layer features. The components of HOG variant with 31 dimensions are given as $3 * Norientations + 4$. We used number of orientations $Norientations = 9$ where $2 * Norientations$ are contrast sensitive orientation channels, $Norientations$ are contrast insensitive orientation channels, and 4 texture channels. Given an image frame with a search window size of $\tilde{M} \times \tilde{N}$ which is about 2.8 times the target size to provide some context, we resize the multi-layer hybrid features to a fixed spatial size of $M \times N$ where $M = \frac{\tilde{M}}{4}$ and $N = \frac{\tilde{N}}{4}$. These hybrid features from each layer are weighted by a cosine window [8] to remove the boundary discontinuities, and then combined later on in Eq. (3.6) for which we set γ as 1, 0.4, 0.02 and 0.1 for the *conv5-4* as layer 1, *conv4-4* as layer 2, *conv3-4* as layer 3 and hand-crafted features as layer 4 in our hybrid features, respectively. We set the regularization parameter of the ridge regression in Eq. (3.2) to $\lambda = 10^{-4}$, and a kernel bandwidth of the Gaussian function label in Eq. (3.1) to $\sigma = 0.1$. The learning rate for model update in Eq. (3.8) is set to $\eta = 0.01$.

For learning the scale correlation filter, we use the same parameter settings as above with some exceptions as follows. In this case we use HOG features [70] with 31 bins i.e. it is treated as a single layer ($L = 1$) but with multiple channels ($D = 31$). The number of scale spaces is set to $S = 31$ and the scale factor is set to $a = 1.04$. We use a linear kernel Eq. (3.9) for learning both translation and scale correlation filters.

HOG, LUV color and normalized gradient magnitude features are used to train an incremental (online) SVM classifier for the re-detection module. For the objective function given in Eq. (3.14), we use a Gaussian kernel, particularly for $Q_{ij} = y_i y_j k(\mathbf{x}_i, \mathbf{x}_j)$, and the regularization parameter C is set to 2. Empirically, we set the activate re-detection threshold to $T_{rd} = 0.15$ and the train detector threshold to $T_{td} = 0.40$. The parameters in Eq. (3.11) are set as $\delta_p = 0.9$ and $\delta_n = 0.3$. For negative samples, we randomly

sampled 3 times the number of positive samples satisfying $\delta_n = 0.3$ within the maximum search area of 4 times the target size. In the re-detection phase, we generated 5 high-score detection proposals from the trained online SVM around the estimated position within the maximum search area of 6 times the target size which are filtered using the GM-PHD filter to find the detection with the maximum weight removing the others as clutter. The implementation parameters are summarized in Table 3.1.

Ps	λ	σ	η	C	T_{rd}	T_{td}	δ_p	δ_n	S	a	λ_t	U	T
Vs	10^{-4}	0.1	0.01	2	0.15	0.40	0.9	0.3	31	1.04	4	4 pixels	10^{-5}

Table 3.1: Implementation parameters, Ps for parameters and Vs for values.

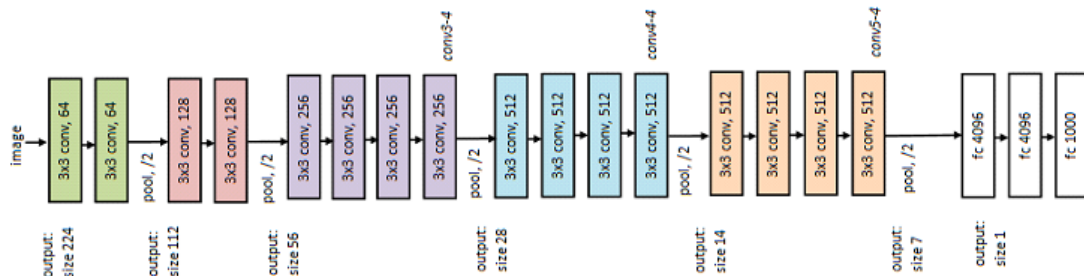


Figure 3.2: VGG-Net 19 [4].

3.4 Experimental Results

We evaluate our proposed tracking algorithm on both a large-scale online object tracking benchmark (OOTB) [93] and crowded scenes (medium and dense PETS 2009 data sets¹), and compare its performance with state-of-the-art trackers using the same parameter values for all the sequences. We quantitatively evaluate the robustness of the trackers using two metrics, precision and success rate based on center location error and bounding box overlap ratio, respectively, using one-pass evaluation (OPE) setting, running the trackers throughout a test sequence with initialization from the ground truth position in the first frame. The center location error computes the average Euclidean distance between the center locations of the tracked targets and the manually labeled ground truth positions of all the frames whereas bounding box overlap ratio computes the intersection over union of the tracked target and ground truth bounding boxes.

¹<http://www.cvg.reading.ac.uk/PETS2009/a.html>

Our proposed tracking algorithm is implemented in MATLAB on a 3.0 GHz Intel Xeon CPU E5-1607 with 16 GB RAM. We also use the MatConvNet toolbox [50] for CNN feature extraction where its forward propagation computation is transferred to a NVIDIA Quadro K5000, and our tracker runs at 5 fps on this setting. The re-detection step and forward propagation for feature extraction step are the main computational load steps of our tracking algorithm. We analyze our algorithm and then compare it with the state-of-the-art trackers both quantitatively and qualitatively on OOTB and PETS 2009 data sets separately as follows.

3.4.1 Evaluation on OOTB

OOTB [93] contains 50 fully annotated videos with substantial variations such as scale, occlusion, illumination, etc and is currently a popular tracking benchmark available in the computer vision community. In this experiment, we compare our proposed tracking algorithm with 6 state-of-the-art trackers including CF2 [5], LCT [7], MEEM [6], DLT [108], KCF [8] and SAMF [219], as well as 4 more top trackers included in the Benchmark [93], particularly SCM [106], ASLA [96], TLD [13] and Struck [101] both quantitatively and qualitatively.

Quantitative Evaluation: We evaluate our proposed tracking algorithm quantitatively and compare with other algorithms as summarized in Fig. 3.3 using precision plots (left) and success plots (right) based on center location error and bounding box overlap ratio, respectively. Our proposed tracking algorithm, denoted by LCMHT, outperforms the state-of-the-art trackers in both precision and success measures by rankings given in the legends using a distance precision of threshold scores at 20 pixels and overlap success of area-under-curve (AUC) score for each tracker, respectively. This is because a hybrid of multi-layer CNN, HOG and color-naming features is more effective to represent the target than their individual features separately i.e. our proposed tracking algorithm integrates a hybrid of multi-layer CNN and traditional (HOG and color-naming) features for learning a translation correlation filter, and uses the GM-PHD filter for temporally filtering generated high score detection proposals during a re-detection phase for removing clutter so that it can re-detect the target even in a cluttered environment.

We also evaluate quantitatively the performance of our algorithm without a re-detection module (LCMHT.WtR) in Fig. 3.4 and show with the top 3 ranked algorithms shown in Fig. 3.3. As can be observed from this figure, the re-detection module has a significant contribution for the performance of our algorithm on OOTB.

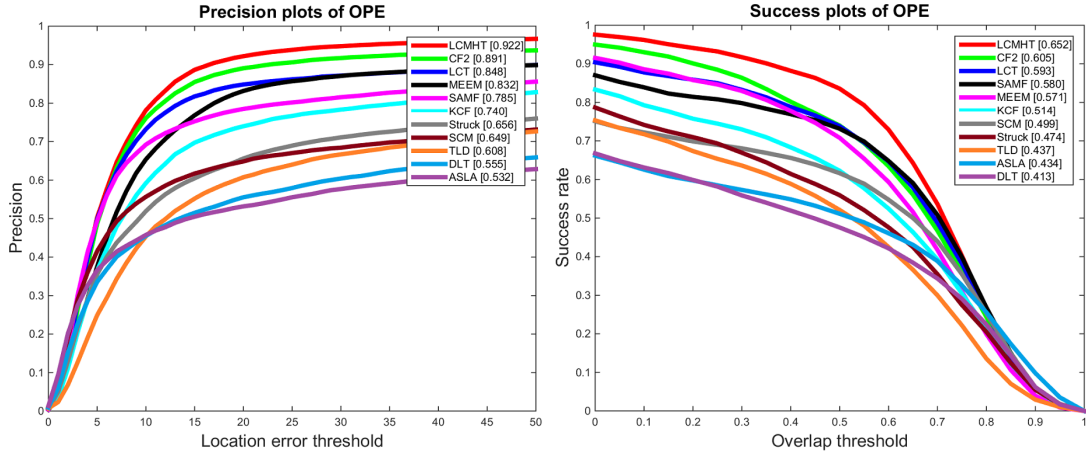


Figure 3.3: Distance precision (left) and overlap success (right) plots on OOTB using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better.

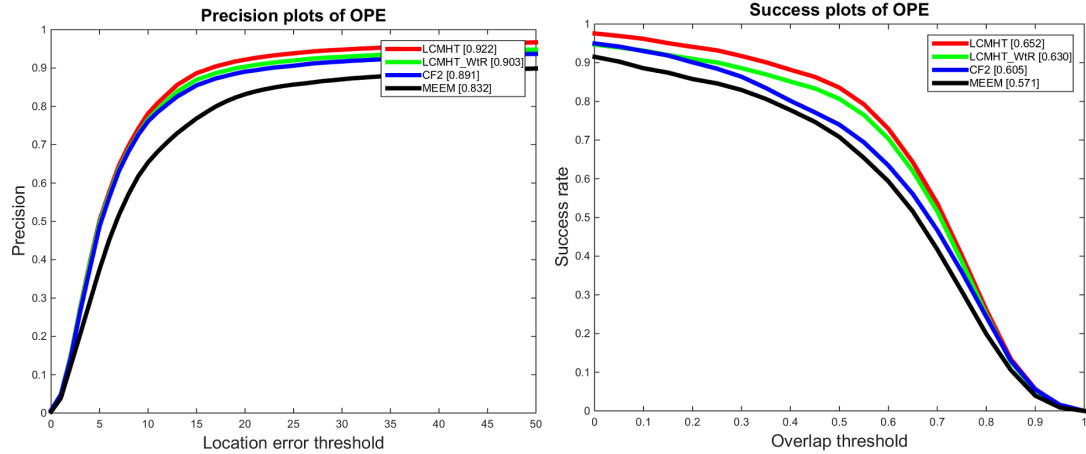


Figure 3.4: Distance precision (left) and overlap success (right) plots on OOTB using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better. The performances of our algorithm (LCMHT) and its version without a re-detection module (LCMHT_WtR) are shown.

Attribute-based Evaluation: For the detailed performance analysis of each of the trackers, we also report the results on various challenge attributes in OOTB [93] such as occlusion, scale variation, illumination variation, etc. As shown in Fig. 3.5, our proposed tracker outperforms the state-of-the-art trackers in almost all challenge attributes. In particular, our proposed tracker (LCMHT) performs significantly better than all trackers on the occlusion attribute since it includes a re-detection module which can re-acquire the target in case the tracker fails even in cluttered environments

by removing clutter using GM-PHD filter. Similarly, our tracker also outperforms other trackers on the scale variation attribute since our tracker elegantly estimates the scale of the tracker at the newly estimated target positions. The LCT algorithm includes both re-detection and scale estimation modules, however, our proposed tracker still outperforms the LCT algorithm by a large margin as shown in Fig. 3.5 since our tracker uses better visual features for translation estimation and re-detection. Furthermore, our proposed algorithm applies scale estimation after translation and re-detection steps (if activated) rather than only after the translation estimation step as in the LCT algorithm, though both methods use similar visual features (HOG) to learn the scale correlation filter.

Qualitative Evaluation: We compare our proposed tracking algorithm (LCMHT) with four other state-of-the-art trackers namely CF2 [5], MEEM [6], LCT [7] and KCF [8] on some challenging sequences of OOTB qualitatively as shown in Fig. 3.6. CF2 uses hierarchical CNN features but is not as effective as our tracker which combines hierarchical CNN features with HOG and color-naming traditional features as can be observed on the sequence *Fleetface* (first column on Fig. 3.6). LCT and KCF also use correlation filters using traditional features but still they are not as accurate as our tracker. MEEM uses many classifiers together to re-initialize the tracker in case of tracking failures but it can not re-detect the target on this sequence. Similarly, it can not re-detect the target on sequences *Singer1* (second column), *Freeman4* (third column) and *Walking2* (forth column) as well. LCT includes re-detection and scale estimation components, however, it can not handle large scale changes as in sequence *Singer1* (second column), and it can not re-initialize the tracker as in sequence *Walking2* (forth column). More importantly, the sequence *Freeman4* undergoes not only heavy occlusion in a cluttered environment but also scale variation, in-plane and out-of-plane rotations. The LCT algorithm which is equipped with both re-detection and scale estimation modules is not effective on this sequence like the other algorithms. However, only our proposed tracker tracks the target till the end of the sequence not only handling the scale change but also re-detecting the target when it fails. This sequence is a typical example which is related to our next evaluation on PETS 2009 data sets on which our proposed algorithm outperforms the other trackers by a large margin.

3.4.2 Evaluation on PETS 2009 Data Sets

We label the upper part (head + neck) of representative targets in both medium and dense PETS 2009 data sets to analyze our proposed tracking algorithm. In this ex-

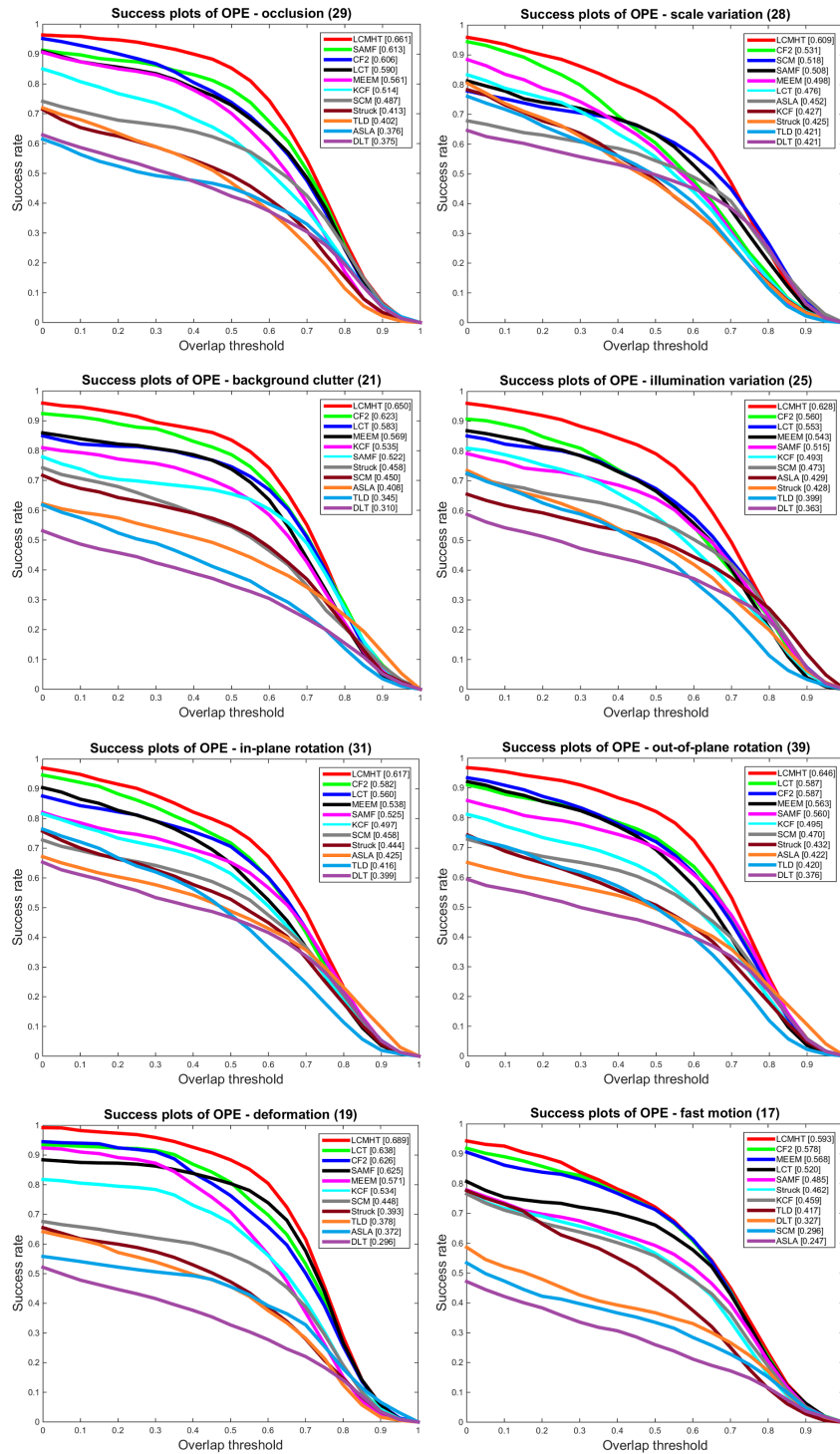


Figure 3.5: Success plots on OOTB using one-pass evaluation (OPE) for 8 challenge attributes: occlusion, scale variation, background clutter, illumination variation, in-plane rotation, out-of-plane rotation, deformation, and fast motion. The legend contains the AUC score of each tracker; the larger, the better.

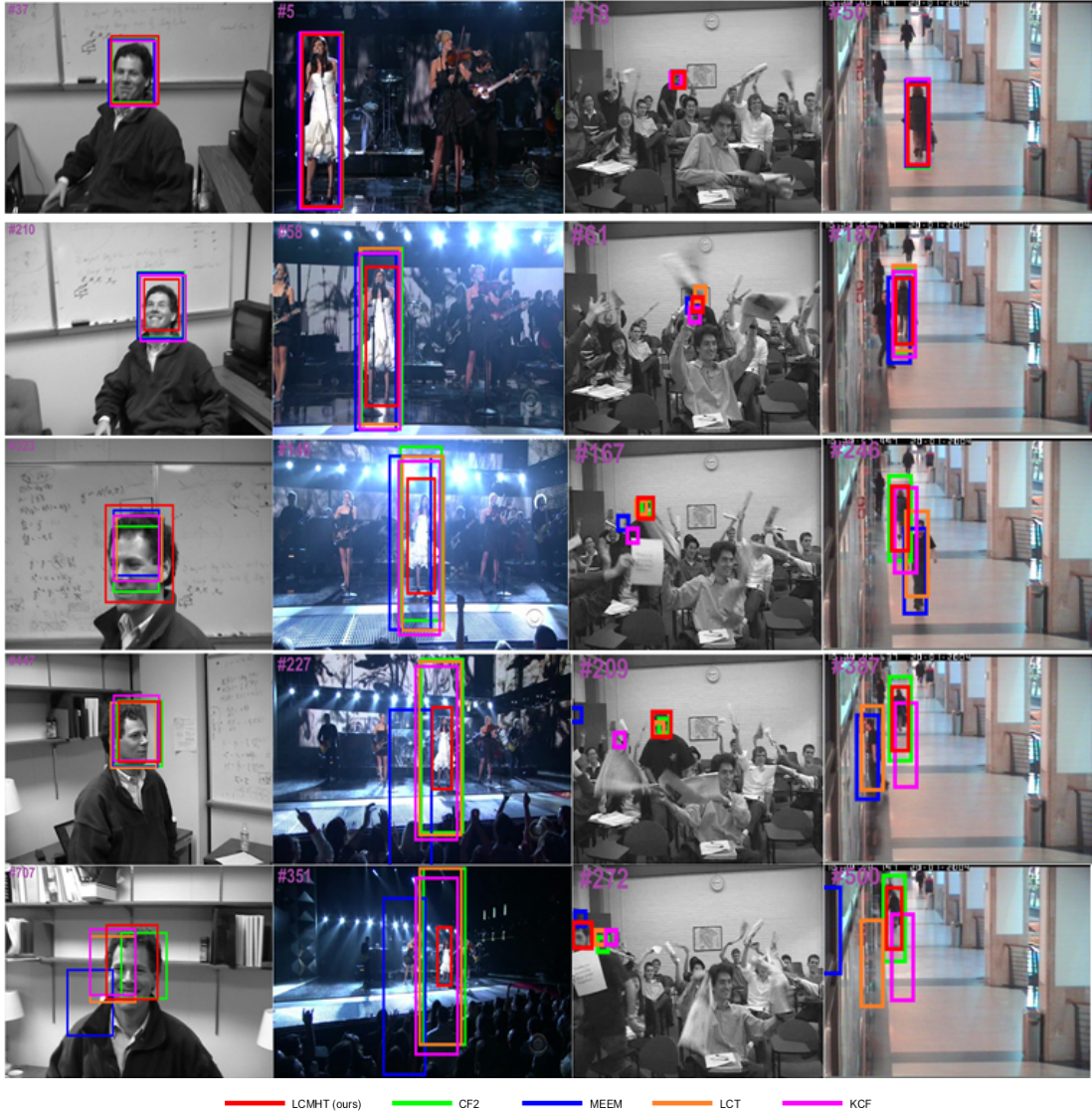


Figure 3.6: Qualitative results of our proposed LCMHT algorithm, CF2 [5], MEEM [6], LCT [7] and KCF [8] on some challenging sequences of OOTB (Fleetface, Singer1, Freeman4, and Walking2 from left column to right column, respectively).

periment, our goal is to analyze our proposed tracking algorithm and other available state-of-the-art tracking algorithms to see whether they can successfully be applied for tracking a target of interest in occluded and cluttered environments. Accordingly, we compare our proposed tracking algorithm with 6 state-of-the-art trackers including CF2 [5], LCT [7], MEEM [6], DSST [111], KCF [8] and SAMF [219], as well as 4 more top trackers included in the Benchmark [93], particularly SCM [106], ASLA [96], CSK [110] and IVT [97] both quantitatively and qualitatively.

Quantitative Evaluation: The evaluation results of precision plots (left) and success

plots (right) based on center location error and bounding box overlap ratio, respectively, are shown in Fig. 3.7. Our proposed tracking algorithm, denoted by LCMHT, outperforms the state-of-the-art trackers by a large margin on PETS 2009 data sets in both precision and success rate measures. The rankings are given in distance precision of threshold scores at 20 pixels and overlap success of AUC score for each tracker as given in the legends.

Similar to OOTB case, we also evaluate the performance of our algorithm without a re-detection module (LCMHT_WtR) on PETS data sets quantitatively in Fig. 3.8 and show with the top 3 ranked algorithms shown in Fig. 3.7. As can be observed from this figure, the re-detection module has much more contribution for the performance of our algorithm than the traditional features and scale estimation combined when compared to CF2, for instance.

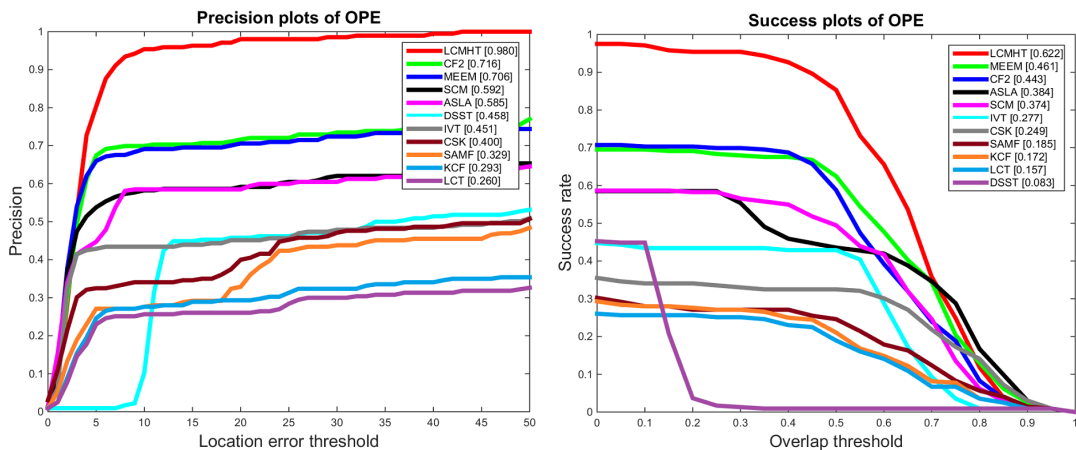


Figure 3.7: Distance precision (left) and overlap success (right) plots on PETS data sets using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better.

The second and third ranked trackers are CF2 [5] and MEEM [6] for precision plots, respectively, and viceversa for success plots on PETS 2009 data sets. However, on OOTB, CF2 outperforms MEEM significantly being second to our proposed tracking algorithm. The most important thing to give attention is on the performance of LCT [7]. This algorithm is ranked third on the OOTB as shown in Fig. 3.3, however, it performs least well on the precision plots and second from last on success plots on PETS 2009 data sets. Surprisingly, this algorithm was developed by learning three different discriminative correlation filters and even included a re-detection module for long-term

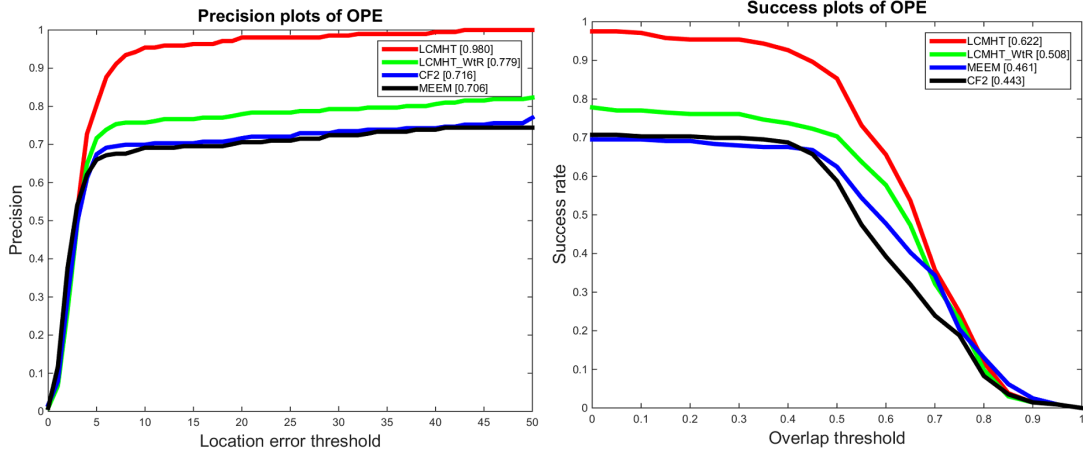


Figure 3.8: Distance precision (left) and overlap success (right) plots on PETS data sets using one-pass evaluation (OPE). The legend for distance precision contains threshold scores at 20 pixels while the legend for overlap success contains the AUC score of each tracker; the larger, the better. The performances of our algorithm (LCMHT) and its version without a re-detection module (LCMHT.WtR) are shown.

tracking problems. Though it performs reasonably on the OOTB, its performance on occluded and cluttered environments such as PETS 2009 data sets is poor due to using less robust visual features in such environments. Even CF2 which uses CNN features has low performance compared to our proposed algorithm on the PETS 2009 data sets. Since our proposed tracking algorithm integrates a hybrid of multi-layer CNN and traditional features for learning the translation correlation filter and GM-PHD filter for temporally filtering generated high score detection proposals during a re-detection phase for removing clutter, it outperforms all the available trackers significantly. This closes the model-free tracking research gap between sparse and crowded environments.

Qualitative Evaluation: Fig. 3.9 presents the performance of our proposed tracker qualitatively compared to the state-of-the-art trackers. In this case, we show the comparison of four representative trackers to our proposed algorithm: CF2 [5], MEEM [6], LCT [7], and KCF [8] as shown in Fig 3.9. On the medium density PETS 2009 data set (left column), LCT and KCF lose the target even on the first 16 frames. Though the CF2 and MEEM trackers track the target well, they could not re-detect the target after the occlusion i.e. only our proposed tracking algorithm tracks the target till the end of the sequence by re-initializing the tracker after the occlusion. We show the cropped and enlarged re-detection just after occlusion in Fig. 3.10. On the dense PETS data set (right column), all trackers track the target on the first 20 frames but LCT and KCF lose the target before 73 frames. Similar to the medium density PETS data set, the CF2 and MEEM trackers track the target before they lose it due to occlusion. Only our proposed tracking algorithm, LCMHT, re-detects the target and tracks it till the end

of the sequence in such dense environments due to two reasons. First, it incorporates both lower and higher CNN layers in combination with traditional features (HOG and color-naming) in a multi-layer to learn the translation correlation filter that is robust to appearance variations of targets. Second, it includes a re-detection module which generates high score detection proposals during a re-detection phase and then filter them using GM-PHD filter to remove clutter due to background and other uninterested targets so that it can re-detect the target in such cluttered and dense environment. These make our proposed tracking algorithm outperform the other state-of-the-art trackers.

3.5 Summary

In this chapter, we propose a novel long-term online model-free tracking algorithm that can be applied to track a target of interest in both sparse and crowded environments. The advantages of the correlation filters, a hybrid of multi-layer CNN and hand-engineered (HOG and color-naming) features, an incremental (online) SVM and a GM-PHD filter are exploited to make the developed algorithm robust to not only long-term occlusions but also challenging clutter from the background scene and other uninterested targets. A ridge regression is extended for multi-layer hybrid CNN and hand-crafted features for learning a translation correlation filter for estimating a target location. These CNN and hand-crafted features are appropriately combined for better performance. Both lower and higher convolutional layers are combined to capture more spatial details and semantic information for localizing a target precisely and handling its appearance variations respectively. When confidence of the translation correlation filter is not large enough, we activate a re-detection module, an online trained SVM on the most confident frames using traditional features (HOG, LUV color and normalized gradient magnitude), to re-initialize the tracker in case of tracking failure due to long-term occlusion. When this re-detection module is activated, it generates high score detection proposals which are then temporally filtered using the GM-PHD filter to find the detection proposal with the maximum weight as the target position estimate by removing the others as clutter. We also learn a scale correlation filter by constructing a target pyramid around the estimated or re-detected position using HOG features for estimating the scale of a target. The integration of each of them for the different parts of the algorithm is relevant for robustness and accuracy i.e. learning correlation filters using an appropriate combination of CNN and traditional features as well as including a re-detection module using incremental SVM and GM-PHD filter can give better results. Extensive experiments both on a large-scale online object tracking benchmark



Figure 3.9: Qualitative results of our proposed algorithm LCMHT, CF2 [5], MEEM [6], LCT [7] and KCF [8] on PETS 2009 medium density (left column) and dense (right column) data sets.



Figure 3.10: Qualitative results of our proposed LCMHT algorithm, CF2 [5], MEEM [6], LCT [7] and KCF [8] on PETS 2009 medium density (left, frame 78) and dense (right, frame 85) data sets, just after occlusion by cropping and enlarging.

(OOTB) and on tracking an interesting target in crowded scenes (PETS 2009) show our proposed method performs favorably against existing state-of-the-art methods.

Chapter 4

Development of a N-type GM-PHD Filter for Multiple Target, Multiple Type Filtering

A unified framework which directly extends single target tracking to multi-target tracking by representing multi-target states and observations as random finite sets (RFS) is developed as discussed in chapter 2 (section 2.5.3) which estimates both the states and cardinality of an unknown and time varying number of targets in a scene. This includes target birth, death, clutter (false alarms), and missing detections with a much lower computational complexity in a single state space rather than in a joint-state space as in most traditional methods. However, all existing RFS-based multiple target filters were developed for either a single or multiple target types but without taking any account of target confusion between target types at the measurement stage i.e. measurements originate not only from the same target type but are also confused with other target types. In this chapter, we model a new filter which can solve these problems.

First, we present recursive Bayes filtering with RFS for multiple targets of different types in section 4.1. Second, we derive our new N-type PHD where $N \geq 2$ using a probability generating functional (PGFL) in section 4.2. Then we present the N-type PHD filtering recursion and its Gaussian mixture implementation scheme in sections 4.3 and 4.4, respectively. We demonstrate by simulation the performance of this proposed N-type GM-PHD filter, specifically for quad GM-PHD filter ($N = 4$) as an example in section 4.5. Finally, we summarize the key concepts behind the N-type GM-PHD filter in section 4.6.

4.1 Multiple Target, Multiple Type Recursive Bayes Filtering with RFS

When different detectors are run on the same scene to detect different target types, there is no guarantee that these detectors detect only their own type. It is possible to run an independent PHD filter for each target type, but this is not correct in most cases, as the likelihood of a positive response to a target of the wrong type is in general different from, usually higher than, the likelihood of a positive response to the scene background. Hence, we want to account for this difference between background clutter and target type confusion. This is equivalent to a single sensor (e.g. a smart camera) that has N different detection modes, each with its own probability of detection and a measurement density for N different target types. The sensor setting for this approach is given in Figure 4.1 in a generic form for N different target types.

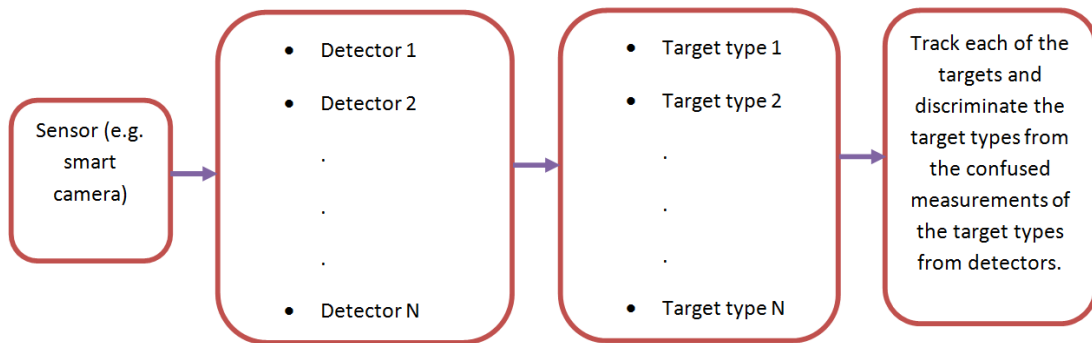


Figure 4.1: A sensor setting for our approach. The sensor is equipped with many detectors for each type of multiple targets in which case one detector can also detect other than its own target type resulting in separate but confused measurements which are filtered and discriminated by our approach.

So we model a N -type PHD filter to filter N -types of multiple targets in such a way that the first PHD update filters the first target type treating the others as potential, additional clutter in addition to background clutter, and vice versa. In the joint tracking and classification approaches such as in [211], [212], the target type (class) is put into the target state vector, however, here we follow a different approach which is convenient for handling target confusions from different target types. Accordingly, to derive the N -type PHD filter, it is necessary to first give its RFS representation extended from a single type single-target Bayes framework to a multiple type multi-target Bayes framework.

Let the multi-target state space $\mathcal{F}(\mathcal{X})$ and the multi-target observation space $\mathcal{F}(\mathcal{Z})$ be the respective collections of all the finite subsets of the state space \mathcal{X} and observation space \mathcal{Z} , respectively. If $L_i(k)$ is the number of targets of target type i in the scene at time k , then the set of multiple states for target type i , $X_{i,k}$, is

$$X_{i,k} = \{x_{i,k,1}, \dots, x_{i,k,L_i(k)}\} \in \mathcal{F}(\mathcal{X}) \quad (4.1)$$

where $i \in \{1, \dots, N\}$. Similarly, if $M_i(k)$ is the number of received observations for target type i , then the corresponding multiple target measurements for that target type is the set

$$Z_{i,k} = \{z_{i,k,1}, \dots, z_{i,k,M_i(k)}\} \in \mathcal{F}(\mathcal{Z}) \quad (4.2)$$

where $i \in \{1, \dots, N\}$. As stated above, some of these observations may be false, i.e. due to clutter (background) or confusion (response due to another target type).

The uncertainty in the state and measurement spaces is introduced by modeling the multi-target state and the multi-target measurement using RFS. Let $\Xi_{i,k}$ be the RFS associated with the multi-target state of target type i , then

$$\Xi_{i,k} = S_{i,k}(X_{i,k-1}) \cup \Gamma_{i,k}, \quad (4.3)$$

where $S_{i,k}(X_{i,k-1})$ denotes the RFS of surviving targets of target type i , and $\Gamma_{i,k}$ is the RFS of the new-born targets of target type i . We do not consider target spawning in this formulation as this has no meaning in our context.

Further, the RFS $\Omega_{i,k}$ associated with the multi-target measurements of target type i is

$$\Omega_{i,k} = \Theta_{i,k}(X_{i,k}) \cup C_{s_{i,k}} \cup C_{t_{i,J,k}}, \quad (4.4)$$

where $J = \{1, \dots, N\} \setminus i$ and $\Theta_{i,k}(X_{i,k})$ is the RFS modeling the measurements generated by the targets $X_{i,k}$, and $C_{s_{i,k}}$ models the RFS associated with the clutter (false alarms) for target type i which comes from the scene background. However, we now must also include $C_{t_{i,J,k}}$ which is the RFS associated with all target types $J = \{1, \dots, N\} \setminus i$ being treated as clutter while filtering target type i i.e. all target types are included into clutter except target type i while filtering target type i . We term this detection of the wrong type of target as confusion, just to distinguish from background clutter.

Analogous to the single-target case, the dynamics of $\Xi_{i,k}$ are described by the multi-target transition density $y_{i,k|k-1}(X_{i,k}|X_{i,k-1})$, while $\Omega_{i,k}$ is described by the multi-target likelihood $f_{j,i,k}(Z_{j,k}|X_{i,k})$ for multiple target type $i \in \{1, \dots, N\}$ from detector $j \in \{1, \dots, N\}$. The recursive equations are

$$p_{i,k|k-1}(X_{i,k}|Z_{i,1:k-1}) = \int y_{i,k|k-1}(X_{i,k}|X)p_{i,k-1|k-1}(X|Z_{i,1:k-1})\mu(dX) \quad (4.5)$$

$$p_{i,k|k}(X_{i,k}|Z_{i,1:k}) = \frac{f_{j,i,k}(Z_{j,k}|X_{i,k})p_{i,k|k-1}(X_{i,k}|Z_{i,1:k-1})}{\int f_{j,i,k}(Z_{j,k}|X)p_{i,k|k-1}(X|Z_{i,1:k-1})\mu(dX)} \quad (4.6)$$

where μ is an appropriate dominating measure on $\mathcal{F}(\mathcal{X})$ [170]. Though a Monte Carlo approximation of this optimal multi-target type Bayes recursion is possible according to multi-target for single type [173], the number of particles required is exponentially related to the number of targets and their types in the scene. To make it computationally tractable, we extend Mahler's method of propagating the first-order moment of the multi-target posterior instead of the full multi-target posterior. This approximation is called the N-type Probability Hypothesis Density (N-type PHD), $\mathcal{D}_{i,k|k}(x|Z_{i,k}) = \mathcal{D}_{i,k|k}(x) = \int \delta_x(x)p_{i,k|k}(X|Z_{i,1:k})\delta X$ where $\delta_x(x) = \sum_{w \in x} \delta_w(x)$, for $N \geq 2$ types of multiple targets by deriving the updated PHDs from Probability Generating Functionals (PGFLs) starting from the standard predicted PHDs of each target type for our new filter which is termed as N-type PHD filter.

4.2 Probability Generating Functional (PGFL)

A probability generating functional is a convenient representation for stochastic modelling with a point process [170], a type of random process for which any one realisation consists of a set of isolated points either in time or space. Now, we model joint (probability generating) functionals which take into account the clutter due to the other target types (confusion) in addition to the background clutter for deriving the updated PHDs. Starting from the standard predicted PHDs [170], we derive novel extensions for the updated PHDs of a N-type PHD filter from PGFLs of each target type, handling confusions among target types.

The joint functional for target type i treating all other target types as clutter is given

by

$$F_i[g, h] = G_{T_i}(hG_{L_{i,i}}(g|\cdot))G_{c_i}(g) \prod_{j=1 \setminus i}^N G_{T_j}(G_{L_{j,i}}(g|\cdot)), \quad (4.7)$$

where $i \in \{1, \dots, N\}$ denotes target type, g is related to the target measurement process and h is related to the target state process.

$$G_{c_i}(g) = \exp(\lambda_i(c_i[g] - 1)), \quad (4.8)$$

where $G_{c_i}(g)$ is the Poisson PGFL for background false alarms where λ_i is the average number of false alarms for target type i and the functional $c_i[g] = \int g(z)c_i(z)dz$ where $c_i(\cdot)$ is the uniform density over the surveillance region;

$$G_{T_i}(h) = \exp(\mu_i(s_i[h] - 1)), \quad (4.9)$$

where $G_{T_i}(h)$ is the prior PGFL (also Poisson-distributed) and μ_i is the average number of targets, each of which is distributed according to $s_i(x)$ for target type i ; and

$$G_{L_{j,i}}(g|x) = 1 - p_{ji,D}(x) + p_{ji,D}(x) \int g(z)f_{ji}(z|x)dz, \quad (4.10)$$

where $G_{L_{j,i}}(g|x)$ is the Bernoulli detection process for each target of target type i using detector j with probability of detection for target type i by detector j , $p_{ji,D}$, and $f_{ji}(z|x)$ is a likelihood defining the probability that z is generated by the target type i conditioned on state x from detector j . Expanding $s_i[hG_{L_{i,i}}(g|x)]$ and $s_j[G_{L_{j,i}}(g|x)]$ as

$$s_i[hG_{L_{i,i}}(g|x)] = \int s_i(x)h(x)(1 - p_{ii,D}(x) + p_{ii,D}(x) \int g(z)f_{ii}(z|x)dz)dx, \quad (4.11)$$

and

$$s_j[G_{L_{j,i}}(g|x)] = \int s_j(x)(1 - p_{ji,D}(x) + p_{ji,D}(x) \int g(z)f_{ji}(z|x)dz)dx. \quad (4.12)$$

Accordingly, $F_i[g, h]$ is expanded as

$$F_i[g, h] = \exp\left(\lambda_i(\int g(z)c_i(z)dz - 1) + \sum_{j=1 \setminus i}^N \mu_j[\int s_j(x)(1 - p_{ji,D}(x) + p_{ji,D}(x) \int g(z)f_{ji}(z|x)dz)dx - 1] + \mu_i[\int s_i(x)h(x)(1 - p_{ii,D}(x) + p_{ii,D}(x) \int g(z)f_{ii}(z|x)dz)dx - 1]\right). \quad (4.13)$$

Now, the extended derivation is based on $F_i[g, h]$ for target type i . Accordingly, the M_i^{th} functional derivative of $F_i[g, h]$ with respect to g in the directions of $\varphi_{z_1}, \dots, \varphi_{z_{M_i}}$ is given by

$$\frac{\delta^{M_i}}{\delta\varphi_{z_1} \dots \delta\varphi_{z_{M_i}}} F_i[g, h] = F_i[g, h] \prod_{m=1}^{M_i} \left[\lambda_i c_i(z_m) + \sum_{j=1 \setminus i}^N \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx + \mu_i \int s_i(x) h(x) p_{ii,D}(x) f_{ii}(z_m|x) dx \right], \quad (4.14)$$

where $\varphi_{z_i} = \delta_{z_i}$ for all z_i .

The PGFL Baye's update $G_i(h|z_1, \dots, z_{M_i})$ for target type i is obtained by finding the M_i^{th} functional derivative of $F_i[g, h]$ and then setting $g = 0$ in the numerator and setting $g = 0, h = 1$ in the denominator as

$$\begin{aligned} G_i(h|z_1, \dots, z_{M_i}) &= \frac{\frac{\delta^{M_i}}{\delta\varphi_{z_1} \dots \delta\varphi_{z_{M_i}}} F_i[g, h] \Big|_{g=0}}{\frac{\delta^{M_i}}{\delta\varphi_{z_1} \dots \delta\varphi_{z_{M_i}}} F_i[g, 1] \Big|_{g=0}}, \\ &= \exp \left(\mu_i \int s_i(x) h(x) (1 - p_{ii,D}(x)) dx - \mu_i \int s_i(x) (1 - p_{ii,D}(x)) dx \right) * \\ &\quad \prod_{m=1}^{M_i} \left[\frac{\lambda_i c_i(z_m) + \sum_{j=1 \setminus i}^N \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx + \mu_i \int s_i(x) h(x) p_{ii,D}(x) f_{ii}(z_m|x) dx}{\lambda_i c_i(z_m) + \sum_{j=1 \setminus i}^N \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx + \mu_i \int s_i(x) p_{ii,D}(x) f_{ii}(z_m|x) dx} \right]. \end{aligned} \quad (4.15)$$

Now, for simplification, Eq. 4.15 can be represented as

$$G_i(h|z_1, \dots, z_{M_i}) = r(x) \prod_{m=1}^{M_i} q_m(x), \quad (4.16)$$

where $r(x)$ and $q_m(x)$ are given by

$$r(x) = \exp \left(\mu_i \int s_i(x) h(x) (1 - p_{ii,D}(x)) dx - \mu_i \int s_i(x) (1 - p_{ii,D}(x)) dx \right), \quad (4.17)$$

$$q_m(x) = \frac{\lambda_i c_i(z_m) + \sum_{j=1 \setminus i}^N \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx + \mu_i \int s_i(x) h(x) p_{ii,D}(x) f_{ii}(z_m|x) dx}{\lambda_i c_i(z_m) + \sum_{j=1 \setminus i}^N \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx + \mu_i \int s_i(x) p_{ii,D}(x) f_{ii}(z_m|x) dx}. \quad (4.18)$$

Now, take the first-order moment (mean) of Eq. 4.16 using the product rule,

$$\frac{\delta}{\delta\varphi_x} G_i(h|z_1, \dots, z_{M_i}) = \frac{\delta}{\delta\varphi_x} r(x) \prod_{m=1}^{M_i} q_m(x) + r(x) \frac{\delta}{\delta\varphi_x} \left[\prod_{m=1}^{M_i} q_m(x) \right], \quad (4.19)$$

Using $\varphi_x = \delta_x$,

$$\frac{\delta}{\delta\varphi_x} r(x) = r(x) \mu_i [s_i(x) - p_{ii,D}(x) s_i(x)], \quad (4.20)$$

Using the product rule for more than two factors,

$$\frac{\delta}{\delta\varphi_x} \left[\prod_{m=1}^{M_i} q_m(x) \right] = \left(\prod_{m=1}^{M_i} q_m(x) \right) \left(\sum_{m=1}^{M_i} \frac{q'_m(x)}{q_m(x)} \right), \quad (4.21)$$

where the derivative of $q_m(x)$ in Eq. 4.18 is obtained using the quotient rule and is given in a simplified form using $\varphi_x = \delta_x$ as

$$\begin{aligned} q'_m(x) &= \frac{\delta}{\delta\varphi_x} q_m(x) \\ &= \frac{\mu_i s_i(x) p_{ii,D}(x) f_{ii}(z_m|x)}{\lambda_i c_i(z_m) + \sum_{j=1 \setminus i}^N \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx + \mu_i \int s_i(x) p_{ii,D}(x) f_{ii}(z_m|x) dx}, \end{aligned} \quad (4.22)$$

where the following equational facts are important for this simplification:

$$\begin{aligned} \frac{\delta}{\delta\varphi_x} \lambda_i c_i(z_m) &= 0, \quad (\text{No process variable } h(x)) \\ \frac{\delta}{\delta\varphi_x} \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx &= 0, \quad (\text{No process variable } h(x)) \\ \frac{\delta}{\delta\varphi_x} \mu_i \int s_i(x) h(x) p_{ii,D}(x) f_{ii}(z_m|x) dx &= \mu_i s_i(x) p_{ii,D}(x) f_{ii}(z_m|x). \end{aligned} \quad (4.23)$$

Now combining Eqs. 4.17, 4.18, 4.20, 4.21, 4.22, i.e. taking the first-order moment (mean) and setting $h = 1$, the updated PHD for target type i treating all other target types as clutter can be obtained and is given by

$$\begin{aligned} \mathcal{D}_i(x|z_1, \dots, z_{M_i}) &= \frac{\delta}{\delta\varphi_x} G_i(h|z_1, \dots, z_{M_i}) \Big|_{h=1}, \\ &= \mu_i s_i(x) (1 - p_{ii,D}(x)) + \\ &\quad \sum_{m=1}^{M_i} \frac{\mu_i s_i(x) p_{ii,D}(x) f_{ii}(z_m|x)}{\lambda_i c_i(z_m) + \sum_{j=1 \setminus i}^N \mu_j \int s_j(x) p_{ji,D}(x) f_{ji}(z_m|x) dx + \mu_i \int s_i(x) p_{ii,D}(x) f_{ii}(z_m|x) dx}. \end{aligned} \quad (4.24)$$

This, $\mathcal{D}_i(x|z_1, \dots, z_{M_i})$ in Eq. (4.24), is the updated PHD for target type i treating all other target types as clutter in the case of the N-type PHD filter.. The term $\mu_i s_i(x)$ in Eq. (4.24) is the predicted PHD for target type i .

4.3 N-type PHD Filtering Strategy

Here we state PHD recursions in a generic form for multiple target, multiple type filtering with $Z_{1,k}, \dots, Z_{N,k}$ separate but confused multi-target measurements between different target types, a N-type PHD filter, where $N \geq 2$. For N types of multiple targets, the PHDs, $\mathcal{D}_{\Xi_1}(x), \mathcal{D}_{\Xi_2}(x), \dots, \mathcal{D}_{\Xi_N}(x)$, are the first-order moments of RFSs, $\Xi_1, \Xi_2, \dots, \Xi_N$, and they are intensity functions on a single state space \mathcal{X} whose peaks identify the likely positions of the targets. For any region $R \subseteq \mathcal{X}$

$$E[|(\Xi_1 \cup \Xi_2 \dots \cup \Xi_N) \cap R|] = \sum_{i=1}^N \int_R \mathcal{D}_{\Xi_i}(x) dx \quad (4.25)$$

where $|\cdot|$ is used to denote the cardinality of a set. Practically, Eq. (4.25) means that by integrating the PHDs on any region R of the state space, we get the expected number of targets (cardinality) in R .

The recursion of the N-type PHD filter is based on 3 assumptions:

- The targets evolve and generate measurements independently of one another.
- The clutter RFS, $C_{s_{i,k}}$, is Poisson-distributed and is independent of target-originated measurements, and
- The predicted multi-target RFS governed by $p_{i,k|k-1}$ is Poisson-distributed.

The third assumption is specific to the derivation of the PHD update where interactions amongst targets are negligible, however, the first two assumptions are common to most Bayesian multi-target trackers [81] [143] [222] [136] [127].

Accordingly, the Bayesian iterative prediction and update of the N-type PHD filtering strategy is given as follows.

The PHD *prediction* for target type i is defined as

$$\mathcal{D}_{i,k|k-1}(x) = \int p_{i,S,k|k-1}(\zeta) y_{i,k|k-1}(x|\zeta) \mathcal{D}_{i,k-1|k-1}(\zeta) d\zeta + \gamma_{i,k}(x), \quad (4.26)$$

where $\gamma_{i,k}(\cdot)$ is the intensity function of a new target birth RFS $\Gamma_{i,k}$, $p_{i,S,k|k-1}(\zeta)$ is the probability that the target still exists at time k given that its previous state is ζ ,

$y_{i,k|k-1}(\cdot|\zeta)$ is the single target state transition density at time k given the previous state ζ for target type i .

Thus, following Eq. (4.24), the final updated PHD for target type i is obtained by setting $\mu_i s_i(x) = \mathcal{D}_{i,k|k-1}(x)$ as

$$\mathcal{D}_{i,k|k}(x) = \left[1 - p_{ii,D}(x) + \sum_{z \in Z_{i,k}} \frac{p_{ii,D}(x) f_{ii,k}(z|x)}{c_{s_{i,k}}(z) + c_{t_{i,k}}(z) + \int p_{ii,D}(\xi) f_{ii,k}(z|\xi) \mathcal{D}_{i,k|k-1}(\xi) d\xi} \right] \mathcal{D}_{i,k|k-1}(x). \quad (4.27)$$

The confused clutter intensity $c_{t_{i,k}}(z)$ due to all types of targets $j = \{1, \dots, N\}$ except target type i in Eq. (4.27) is given by

$$c_{t_{i,k}}(z) = \sum_{j=\{1, \dots, N\} \setminus i} \int p_{ji,D}(y) \mathcal{D}_{j,k|k-1}(y) f_{ji,k}(z|y) dy. \quad (4.28)$$

This means that when we are filtering target type i , all the other target types are included into clutter (confusion). Eq. (4.28) converts state space to observation space by integrating the PHD estimator $\mathcal{D}_{j,k|k-1}(y)$ and likelihood $f_{ji,k}(z|y)$ which defines the probability that z is generated by detector j conditioned on state x of the target type i taking into account the confusion probability $p_{ji,D}(y)$, the detection probability for target type i by detector j .

The clutter intensity due to the background for target type i , $c_{s_{i,k}}(z)$, in Eq. (4.27) is given by

$$c_{s_{i,k}}(z) = \lambda_i c_i(z) = \lambda_{c_i} A c_i(z), \quad (4.29)$$

where $c_i(\cdot)$ is the uniform density over the surveillance region A , and λ_{c_i} is the average number of clutter returns per unit volume for target type i i.e. $\lambda_i = \lambda_{c_i} A$. While PHD filter has linear complexity with the current number of measurements (m) and with the current number of targets (n) i.e. computational order of $O(mn)$, N-type PHD filter has linear complexity with the current number of measurements (m), with the current number of targets (n) and with the total number of target types (N) i.e. computational order (run-time) of $O(mnN)$ as can be seen from Eq. (4.27).

In general, the clutter intensity due to the background for target type i , $c_{s_{i,k}}(z)$, can be different for each target type as they depend on the receiver operating characteristic (ROC) curves of the detection processes. Moreover, the probabilities of detection

$p_{ii,D}(x)$ and $p_{ji,D}(x)$ may all be different although assumed constant across both the time and space continua.

4.4 Gaussian Mixture-Based N-type PHD Filter Implementation

The Gaussian mixture implementation of the standard PHD (GM-PHD) filter [172] is a closed-form solution of the PHD filter with the assumption of a linear Gaussian system. In this section, this standard implementation is extended for the N-type PHD filter, more importantly solving Eq. (4.28). We assume each target follows a linear Gaussian model.

$$y_{i,k|k-1}(x|\zeta) = \mathcal{N}(x; F_{i,k-1}\zeta, Q_{i,k-1}) \quad (4.30)$$

$$f_{ji,k}(z|x) = \mathcal{N}(z; H_{ji,k}x, R_{ji,k}) \quad (4.31)$$

where $\mathcal{N}(\cdot; m, P)$ denotes a Gaussian density with mean m and covariance P ; $F_{i,k-1}$ and $H_{ji,k}$ are the state transition and measurement matrices, respectively. $Q_{i,k-1}$ and $R_{ji,k}$ are the covariance matrices of the process and the measurement noises, respectively, where $i \in \{1, \dots, N\}$ and $j \in \{1, \dots, N\}$. The intensity of the spontaneous birth RFS is $\gamma_{i,k}(x)$ for target type i

$$\gamma_{i,k}(x) = \sum_{v=1}^{V_{\gamma_{i,k}}} w_{i,\gamma,k}^{(v)} \mathcal{N}(x; m_{i,\gamma,k}^{(v)}, P_{i,\gamma,k}^{(v)}) \quad (4.32)$$

where $V_{\gamma_{i,k}}$ is the number of birth Gaussian components for target type i where $i \in \{1, \dots, N\}$, $m_{i,\gamma,k}^{(v)}$ is the current measurement (noisy version of position) and zero initial velocity used as mean (peak of the spontaneous birth intensity), $P_{i,\gamma,k}^{(v)}$ is birth covariance for Gaussian component v of target type i and determines the spread of the birth intensity in the vicinity of the peak $m_{i,\gamma,k}^{(v)}$, and the weight $w_{i,\gamma,k}^{(v)}$ gives the expected number of new targets originating from $m_{i,\gamma,k}^{(v)}$.

It is assumed that the posterior intensity for target type i at time $k - 1$ is a Gaussian

mixture of the form

$$\mathcal{D}_{i,k-1}(x) = \mathcal{D}_{i,k-1|k-1}(x) = \sum_{v=1}^{V_{i,k-1}} w_{i,k-1}^{(v)} \mathcal{N}(x; m_{i,k-1}^{(v)}, P_{i,k-1}^{(v)}), \quad (4.33)$$

where $i \in \{1, \dots, N\}$ and $V_{i,k-1}$ is the number of Gaussian components of $\mathcal{D}_{i,k-1}(x)$. Under these assumptions, the predicted intensity at time k for target type i is given following Eq. (4.26) by

$$\mathcal{D}_{i,k|k-1}(x) = \mathcal{D}_{i,S,k|k}(x) + \gamma_{i,k}(x), \quad (4.34)$$

where

$$\mathcal{D}_{i,S,k|k-1}(x) = p_{i,S,k} \sum_{v=1}^{V_{i,k-1}} w_{i,k-1}^{(v)} \mathcal{N}(x; m_{i,S,k|k-1}^{(v)}, P_{i,S,k|k-1}^{(v)}),$$

$$m_{i,S,k|k-1}^{(v)} = F_{i,k-1} m_{i,k-1}^{(v)},$$

$$P_{i,S,k|k-1}^{(v)} = Q_{i,k-1} + F_{i,k-1} P_{i,k-1}^{(v)} F_{i,k-1}^T,$$

where $\gamma_{i,k}(x)$ is given by Eq. (4.32).

Since $\mathcal{D}_{i,S,k|k-1}(x)$ and $\gamma_{i,k}(x)$ are Gaussian mixtures, $\mathcal{D}_{i,k|k-1}(x)$ can be expressed as a Gaussian mixture of the form

$$\mathcal{D}_{i,k|k-1}(x) = \sum_{v=1}^{V_{i,k|k-1}} w_{i,k|k-1}^{(v)} \mathcal{N}(x; m_{i,k|k-1}^{(v)}, P_{i,k|k-1}^{(v)}), \quad (4.35)$$

where $w_{i,k|k-1}^{(v)}$ is the weight accompanying the predicted Gaussian component v for target type i and $V_{i,k|k-1}$ is the number of predicted Gaussian components for target type i where $i \in \{1, \dots, N\}$.

Now, assuming the probabilities of detection to be constant i.e. $p_{ji,D}(x) = p_{ji,D}$, the final updated PHD for target type i is given as follows. Accordingly, the posterior

intensity for target type i at time k (updated PHD), considering incorrect detection of target types as confusion, is also a Gaussian mixture which corresponds to Eq. (4.27) and is given by

$$\mathcal{D}_{i,k|k}(x) = (1 - p_{ii,D,k})\mathcal{D}_{i,k|k-1}(x) + \sum_{z \in Z_{i,k}} \mathcal{D}_{i,D,k}(x; z), \quad (4.36)$$

where

$$\mathcal{D}_{i,D,k}(x; z) = \sum_{v=1}^{J_{i,k|k-1}} w_{i,k}^{(v)}(z) \mathcal{N}(x; m_{i,k|k}^{(v)}(z), P_{i,k|k}^{(v)}),$$

$$w_{i,k}^{(v)}(z) = \frac{p_{ii,D,k} w_{i,k|k-1}^{(v)} q_{i,k}^{(v)}(z)}{c_{s_{i,k}}(z) + c_{t_{i,k}}(z) + p_{ii,D,k} \sum_{l=1}^{V_{i,k|k-1}} w_{i,k|k-1}^{(l)} q_{i,k}^{(l)}(z)},$$

$$q_{i,k}^{(v)}(z) = \mathcal{N}(z; H_{ii,k} m_{i,k|k-1}^{(v)}, R_{ii,k} + H_{ii,k} P_{i,k|k-1}^{(v)} H_{ii,k}^T),$$

$$m_{i,k|k}^{(v)}(z) = m_{i,k|k-1}^{(v)} + K_{i,k}^{(v)}(z - H_{ii,k} m_{i,k|k-1}^{(v)}),$$

$$P_{i,k|k}^{(v)} = [I - K_{i,k}^{(v)} H_{ii,k}] P_{i,k|k-1}^{(v)},$$

$$K_{i,k}^{(v)} = P_{i,k|k-1}^{(v)} H_{ii,k}^T [H_{ii,k} P_{i,k|k-1}^{(v)} H_{ii,k}^T + R_{ii,k}]^{-1}.$$

$c_{s_{i,k}}(z)$ is given in Eq. (4.29). Therefore, all that is left is to formulate the implementation scheme for $c_{t_{i,k}}(z)$ which is given in Eq. (4.28) and is given again as

$$c_{t_{i,k}}(z) = \sum_{j=\{1,\dots,N\} \setminus i} p_{ji,D} \int \mathcal{D}_{j,k|k-1}(y) f_{ji,k}(z|y) dy, \quad (4.37)$$

where $\mathcal{D}_{j,k|k-1}(y)$ is given in Eq. (4.35), $f_{ji,k}(z|y)$ is given in Eq. (4.31) and $p_{ji,D}$ is assumed constant. Since $w_{j,k|k-1}^{(v)}$ is independent of the integrable variable y , Eq. (4.37)

becomes

$$c_{t_{i,k}}(z) = \sum_{j=\{1,\dots,N\}\setminus i} \sum_{v=1}^{V_{j,k|k-1}} p_{j,i,D} w_{j,k|k-1}^{(v)} \int \mathcal{N}(y; m_{j,k|k-1}^{(v)}, P_{j,k|k-1}^{(v)}) \mathcal{N}(z; H_{j,i,k} y, R_{j,i,k}) dy. \quad (4.38)$$

This can be simplified using the following equality given that P_1 and P_2 are positive definite

$$\int \mathcal{N}(y; m_1 \zeta, P_1) \mathcal{N}(\zeta; m_2, P_2) d\zeta = \mathcal{N}(y; m_1 m_2, P_1 + m_1 P_2 m_2^T). \quad (4.39)$$

Therefore, Eq. (4.38) becomes,

$$c_{t_{i,k}}(z) = \sum_{j=\{1,\dots,N\}\setminus i} \sum_{v=1}^{V_{j,k|k-1}} p_{j,i,D} w_{j,k|k-1}^{(v)} \mathcal{N}(z; H_{j,i,k} m_{j,k|k-1}^{(v)}, R_{j,i,k} + H_{j,i,k} P_{j,k|k-1}^{(v)} H_{j,i,k}^T), \quad (4.40)$$

where $i \in \{1, \dots, N\}$.

The key steps of the N-type GM-PHD filter are summarised in Algorithms 2, 3 and 4. The number of Gaussian components in the posterior intensities may increase without bound as time progresses. Therefore, it is necessary to prune weak and duplicated components in Algorithm 3. First, weak components with weight $w_{i,k}^{(v)} < T = 10^{-5}$ are pruned. Further, Gaussian components with Mahalanobis distance less than $U = 4m$ from each other are merged. These pruned and merged Gaussian components, output of Algorithm 3, are predicted as existing targets in the next iteration. Finally, Gaussian components of the posterior intensity, output of Algorithm 2, with means corresponding to weights greater than 0.5 as a threshold are selected as multi-target state estimates as given in Algorithm 4.

4.5 Experimental Results

Simulation filtering example using a quad GM-PHD filter for four different types of multiple targets is analyzed in this section. We also apply a dual GM-PHD filter and a tri-GM-PHD filter for visual tracking applications in Chapter 5 for two types and three types of targets, respectively, after making their experimental simulation analyses. In this experiment, we demonstrate the quad GM-PHD filter ($N = 4$) with detailed

Algorithm 2 Pseudocode for the N-type GM-PHD filter

```

1: given  $\{w_{i,k-1}^{(v)}, m_{i,k-1}^{(v)}, P_{i,k-1}^{(v)}\}_{v=1}^{V_{i,k-1}}$ , and the measurement set  $Z_{i,k}$  for target type
    $i \in \{1, \dots, N\}$ 
2: step 1. (prediction for birth targets)
3: for  $i = 1, \dots, N$  do ▷ for all target type  $i$ 
4:    $e_i = 0$ 
5:   for  $u = 1, \dots, V_{\gamma_i,k}$  do
6:      $e_i := e_i + 1$ 
7:      $w_{i,k|k-1}^{(e_i)} = w_{i,\gamma,k}^{(u)}$ 
8:      $m_{i,k|k-1}^{(e_i)} = m_{i,\gamma,k}^{(u)}$ 
9:      $P_{i,k|k-1}^{(e_i)} = P_{i,\gamma,k}^{(u)}$ 
10:  end for
11: end for
12: step 2. (prediction for existing targets)
13: for  $i = 1, \dots, N$  do ▷ for all target type  $i$ 
14:   for  $u = 1, \dots, V_{i,k-1}$  do
15:      $e_i := e_i + 1$ 
16:      $w_{i,k|k-1}^{(e_i)} = p_{i,S,k} w_{i,k-1}^{(u)}$ 
17:      $m_{i,k|k-1}^{(e_i)} = F_{i,k-1} m_{i,k-1}^{(u)}$ 
18:      $P_{i,k|k-1}^{(e_i)} = Q_{i,k-1} + F_{i,k-1} P_{i,k-1}^{(u)} F_{i,k-1}^T$ 
19:   end for
20: end for
21:  $V_{i,k|k-1} = e_i$ 
22: step 3. (Construction of PHD update components)
23: for  $i = 1, \dots, N$  do ▷ for all target type  $i$ 
24:   for  $u = 1, \dots, V_{i,k|k-1}$  do
25:      $\eta_{i,k|k-1}^{(u)} = H_{ii,k} m_{i,k|k-1}^{(u)}$ 
26:      $S_{i,k}^{(u)} = R_{ii,k} + H_{ii,k} P_{i,k|k-1}^{(u)} H_{ii,k}^T$ 
27:      $K_{i,k}^{(u)} = P_{i,k|k-1}^{(u)} H_{ii,k}^T [S_{i,k}^{(u)}]^{-1}$ 
28:      $P_{i,k|k}^{(u)} = [I - K_{i,k}^{(u)} H_{ii,k}] P_{i,k|k-1}^{(u)}$ 
29:   end for
30: end for
31: step 4. (Update)
32: for  $i = 1, \dots, N$  do ▷ for all target type  $i$ 
33:   for  $u = 1, \dots, V_{i,k|k-1}$  do
34:      $w_{i,k}^{(u)} = (1 - p_{ii,D,k}) w_{i,k|k-1}^{(u)}$ 
35:      $m_{i,k}^{(u)} = m_{i,k|k-1}^{(u)}$ 
36:      $P_{i,k}^{(u)} = P_{i,k|k-1}^{(u)}$ 
37:   end for
38:    $l_i := 0$ 
39:   for each  $z \in Z_{i,k}$  do
40:      $l_i := l_i + 1$ 
41:     for  $u = 1, \dots, V_{i,k|k-1}$  do

```

```

42:      $w_{i,k}^{(l_i V_{i,k|k-1} + u)} = p_{ii,D,k} w_{i,k|k-1}^{(u)} \mathcal{N}(z; \eta_{i,k|k-1}^{(u)}, S_{i,k}^{(u)})$ 
43:      $m_{i,k}^{(l_i V_{i,k|k-1} + u)} = m_{i,k|k-1}^{(u)} + K_{i,k}^{(u)}(z - \eta_{i,k|k-1}^{(u)})$ 
44:      $P_{i,k}^{(l_i V_{i,k|k-1} + u)} = P_{i,k|k}^{(u)}$ 
45:   end for
46:   for  $u = 1, \dots, V_{i,k|k-1}$  do
47:      $c_{s_{i,k}}(z) = \lambda_{c_i} A c_i(z)$ 
48:      $c_{t_{i,k}}(z) = \sum_{j=\{1, \dots, N\} \setminus i} \sum_{e=1}^{V_{j,k|k-1}} p_{ji,D} w_{j,k|k-1}^{(e)} \mathcal{N}(z; H_{ji,k} m_{j,k|k-1}^{(e)}, R_{ji,k} +$ 
49:        $H_{ji,k} P_{j,k|k-1}^{(e)} H_{ji,k}^T)$ 
50:      $c_{i,k}(z) = c_{s_{i,k}}(z) + c_{t_{i,k}}(z)$ 
51:      $w_{i,k,N} = \sum_{e=1}^{V_{i,k|k-1}} w_{i,k}^{(l_i V_{i,k|k-1} + e)}$ 
52:      $w_{i,k}^{(l_i V_{i,k|k-1} + u)} = \frac{w_{i,k}^{(l_i V_{i,k|k-1} + u)}}{c_{i,k}(z) + w_{i,k,N}}$ 
53:   end for
54:    $V_{i,k} = l_i V_{i,k|k-1} + V_{i,k|k-1}$ 
55: end for
56: output  $\{w_{i,k}^{(v)}, m_{i,k}^{(v)}, P_{i,k}^{(v)}\}_{v=1}^{V_{i,k}}$ 

```

Algorithm 3 Pruning and merging for the N-type GM-PHD filter

```

1: given  $\{w_{i,k}^{(v)}, m_{i,k}^{(v)}, P_{i,k}^{(v)}\}_{v=1}^{V_{i,k}}$  for target type  $i \in \{1, \dots, N\}$ , a pruning weight
   threshold  $T$ , and a merging distance threshold  $U$ .
2: for  $i = 1, \dots, N$  do ▷ for all target type  $i$ 
3:   Set  $\ell_i = 0$ , and  $I_i = \{v = 1, \dots, V_{i,k} | w_{i,k}^{(v)} > T\}$ 
4:   repeat
5:      $\ell_i := \ell_i + 1$ 
6:      $u := \arg \max_{v \in I_i} w_{i,k}^{(v)}$ 
7:      $L_i := \{v \in I_i | (m_{i,k}^{(v)} - m_{i,k}^{(u)})^T (P_{i,k}^{(v)})^{-1} (m_{i,k}^{(v)} - m_{i,k}^{(u)}) \leq U\}$ 
8:      $\tilde{w}_{i,k}^{(\ell_i)} = \sum_{v \in L_i} w_{i,k}^{(v)}$ 
9:      $\tilde{m}_{i,k}^{(\ell_i)} = \frac{1}{\tilde{w}_{i,k}^{(\ell_i)}} \sum_{v \in L_i} w_{i,k}^{(v)} m_{i,k}^{(v)}$ 
10:     $\tilde{P}_{i,k}^{(\ell_i)} = \frac{1}{\tilde{w}_{i,k}^{(\ell_i)}} \sum_{v \in L_i} w_{i,k}^{(v)} (P_{i,k}^{(v)} + (\tilde{m}_{i,k}^{(\ell_i)} - m_{i,k}^{(v)}) (\tilde{m}_{i,k}^{(\ell_i)} - m_{i,k}^{(v)})^T)$ 
11:     $I_i := I_i \setminus L_i$ 
12:  until  $I_i = \emptyset$ 
13: end for
14: output  $\{\tilde{w}_{i,k}^{(v)}, \tilde{m}_{i,k}^{(v)}, \tilde{P}_{i,k}^{(v)}\}_{v=1}^{\ell_i}$  as pruned and merged Gaussian components
   for target type  $i$ .

```

Algorithm 4 Multi-target state extraction for the N-type GM-PHD filter

```

1: given  $\{w_{i,k}^{(v)}, m_{i,k}^{(v)}, P_{i,k}^{(v)}\}_{v=1}^{V_{i,k}}$  for target type  $i \in \{1, \dots, N\}$ .
2: for  $i = 1, \dots, N$  do ▷ for all target type  $i$ 
3:   Set  $\hat{X}_{i,k} = \emptyset$ 
4:   for  $v = 1, \dots, V_{i,k}$  do
5:     if  $w_{i,k}^{(v)} > 0.5$  then
6:       for  $j = 1, \dots, \text{round}(w_{i,k}^{(v)})$  do
7:         update  $\hat{X}_{i,k} := [\hat{X}_{i,k}, m_{i,k}^{(v)}]$ 
8:       end for
9:     end if
10:  end for
11: end for
12: output  $\hat{X}_{i,k}$  as the multi-target state estimate for target type  $i$ .

```

analysis as a typical simulation example. Accordingly, we define a sequence of 120 frames with sixteen trajectories that emanate from four types of targets that appear in the scene at different positions of the first frame, as shown in Fig. 4.3. This is a typical example of not only a higher number of target types (four) but also an example of a dense scene i.e. it consists of trajectories of 16 targets in the same scene with many crossings. Obviously, the goal of a N-type PHD filter is to handle confusions among $N \geq 2$ different target types; not to deal with sparse or dense targets in the scene. With regards to sparse or dense targets in the scene, it has the same characteristics as the standard PHD filter.

The initial locations and covariances for all target types are given by Eq. 4.41 and Eq. 4.42 as follows.

$$\begin{aligned}
m_{1,k}^{(1)} &= [-100, 700, 0, 0]^T, \\
m_{1,k}^{(2)} &= [-750, -100, 0, 0]^T, \\
m_{1,k}^{(3)} &= [-200, 400, 0, 0]^T, \\
m_{1,k}^{(4)} &= [-700, -400, 0, 0]^T, \\
m_{2,k}^{(5)} &= [-400, 600, 0, 0]^T, \\
m_{2,k}^{(6)} &= [-800, -600, 0, 0]^T, \\
m_{2,k}^{(7)} &= [-500, -200, 0, 0]^T, \\
m_{2,k}^{(8)} &= [700, 600, 0, 0]^T, \\
P_{1,k} &= P_{2,k} = \text{diag}([100, 100, 25, 25]).
\end{aligned} \tag{4.41}$$

$$\begin{aligned}
m_{3,k}^{(9)} &= [-900, 100, 0, 0]^T, \\
m_{3,k}^{(10)} &= [-800, 500, 0, 0]^T, \\
m_{3,k}^{(11)} &= [-900, -200, 0, 0]^T, \\
m_{3,k}^{(12)} &= [400, -600, 0, 0]^T, \\
m_{4,k}^{(13)} &= [800, -600, 0, 0]^T, \\
m_{4,k}^{(14)} &= [500, -700, 0, 0]^T, \\
m_{4,k}^{(15)} &= [-700, -600, 0, 0]^T, \\
m_{4,k}^{(16)} &= [900, -100, 0, 0]^T, \\
P_{3,k} &= P_{4,k} = \text{diag}([100, 100, 25, 25]).
\end{aligned} \tag{4.42}$$

The state vector $x_k = [p_{x,xk}, p_{y,xk}, \dot{p}_{x,xk}, \dot{p}_{y,xk}]^T$ consists of position $(p_{x,xk}, p_{y,xk})$ and velocity $(\dot{p}_{x,xk}, \dot{p}_{y,xk})$, and the measurement is a noisy version of the position, $z_k = [p_{x,zk}, p_{y,zk}]^T$. Each of the target trajectories follows a linear Gaussian dynamic model of Eq. (4.30) with matrices

$$\begin{aligned}
F_{i,k-1} &= \begin{bmatrix} I_2 & \Delta I_2 \\ 0_2 & I_2 \end{bmatrix}, \\
Q_{i,k-1} &= \sigma_{v_i}^2 \begin{bmatrix} \frac{\Delta^4}{4} I_2 & \frac{\Delta^3}{2} I_2 \\ \frac{\Delta^3}{2} I_2 & \Delta^2 I_2 \end{bmatrix},
\end{aligned} \tag{4.43}$$

where I_n and 0_n denote the $n \times n$ identity and zero matrices, respectively. $\Delta = 1s$ is the sampling period. $\sigma_{v_i} = 5m/s^2$ where $i \in \{1, 2, 3, 4\}$ is the standard deviation of the process noise for target type i .

For the algorithm, we assume each target has a survival probability $p_{1,S} = p_{2,S} = p_{3,S} = p_{4,S} = 0.99$. The probabilities of detection are $p_{11,D} = 0.90$, $p_{22,D} = p_{33,D} = 0.92$, $p_{44,D} = 0.91$, and different values of confusion detection probabilities (0.0, 0.3, 0.6 and 0.9) are analyzed for $p_{12,D}$, $p_{13,D}$, $p_{14,D}$, $p_{21,D}$, $p_{23,D}$, $p_{24,D}$, $p_{31,D}$, $p_{32,D}$, $p_{34,D}$, $p_{41,D}$, $p_{42,D}$ and $p_{43,D}$.

The measurement follows the observation model of Eq. (4.31) with matrices

$$\begin{aligned} H_{ii,k} = H_{ji,k} &= [I_2 \ 0_2], \\ R_{ii,k} &= \sigma_{r_{ii}}^2 I_2, \\ R_{ji,k} &= \sigma_{r_{ji}}^2 I_2, \end{aligned} \tag{4.44}$$

where $\sigma_{r_{ii}} = \sigma_{r_{ji}} = 6m$ ($i \in \{1, 2, 3, 4\}$ and $j \in \{1, 2, 3, 4\}$) is the standard deviation of the measurement noise.

Since there are many targets in the scene, we use about 16 clutter returns (4 for each target type) over the surveillance region. A current measurement driven birth intensity inspired by but not identical to [176] is introduced at each time step, removing the need for the prior knowledge (specification of birth intensities) or a random model, with a non-informative zero initial velocity. At birth, Gaussian components of each target type has a corresponding initial weight $w_{1,\gamma,k}^{(i)} = w_{2,\gamma,k}^{(i)} = w_{3,\gamma,k}^{(i)} = w_{4,\gamma,k}^{(i)} = 3 \times 10^{-6}$. This very small initial weight is assigned to the Gaussian components for new births as this is effective for high clutter rates. This is basically equivalent to the average number of appearing (birth) targets per time step (n_b) divided uniformly across the surveillance region (A).

The configuration of the detectors is shown Fig. 4.2. As shown in this figure, detector 1 detects target type 1 (targets 1, 2, 3 and 4) with probability of detection $p_{11,D}$, target 5 which is of target type 2 with probability of detection $p_{21,D}$, target 9 which is of target type 3 with probability of detection $p_{31,D}$ and target 13 which is of target type 4 with probability of detection $p_{41,D}$. Detector 2 detects target type 2 (targets 5, 6, 7 and 8) with probability of detection $p_{22,D}$, target 1 which is of target type 1 with probability of detection $p_{12,D}$, target 10 which is of target type 3 with probability of detection $p_{32,D}$ and target 14 which is of target type 4 with probability of detection $p_{42,D}$. Similarly, detector 3 detects target type 3 (targets 9, 10, 11 and 12) with probability of detection $p_{33,D}$, target 2 which is of target type 1 with probability of detection $p_{13,D}$, target 6 which is of target type 2 with probability of detection $p_{23,D}$ and target 15 which is of target type 4 with probability of detection $p_{43,D}$. Moreover, detector 4 detects target type 4 (targets 13, 14, 15 and 16) with probability of detection $p_{44,D}$, target 3 which is of target type 1 with probability of detection $p_{14,D}$, target 7 which is of target type 2 with probability of detection $p_{24,D}$ and target 11 which is of target type 3 with probability of detection $p_{34,D}$. This means that targets 1, 2 and 3 from target type 1, targets 5, 6 and 7 from target type 2, targets 8, 9 and 10 from target type 3, and targets 13, 14 and 15 from target type 4 are detected two times. Our main goal is

to filter out confused measurements which correspond to a specific target i.e. doubly detected targets are estimated once, not twice. Therefore, the number of targets in the scene is 16, not 28.

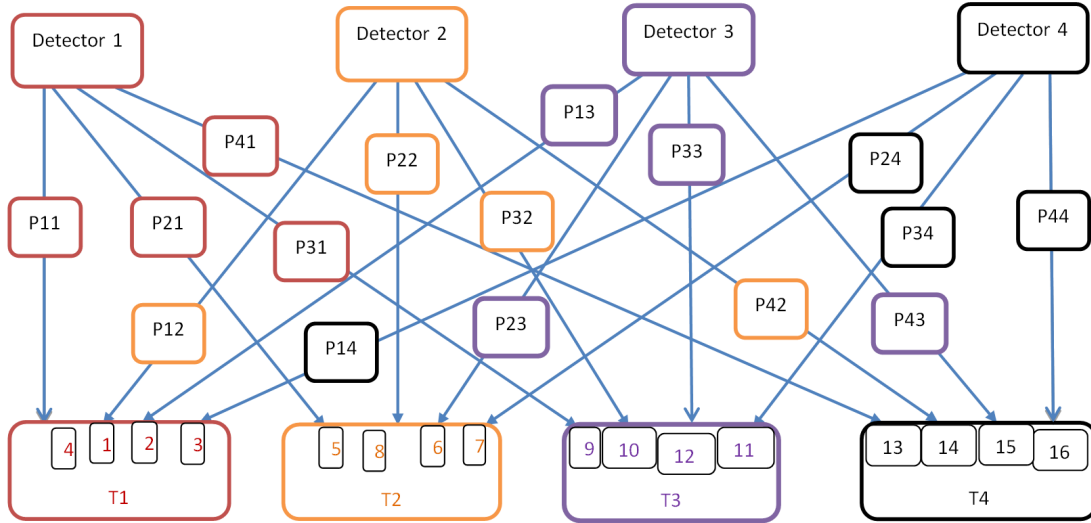


Figure 4.2: Confusions between four target types (T1, T2, T3 and T4) at the detection stage from detectors 1, 2, 3 and 4.

The figures shown in Fig. 4.3, Fig. 4.4, Fig. 4.5 and Fig. 4.6 show the comparisons of the outputs of both the quad GM-PHD filter (Fig. 4.4 for detection probability of confusion of 0.3 and Fig. 4.6 for detection probability of confusion of 0.6) and four independent GM-PHD filters (Fig. 4.3 for detection probability of confusion of 0.3 and Fig. 4.5 for detection probability of confusion of 0.6). For both approaches, the simulated ground truths are shown in red for target type 1, black for target type 2, yellow for target type 3 and magenta for target type 4 while the estimates are shown in blue circles for target type 1, green triangles for target type 2, cyan asterisks for target type 3 and black circles for target type 4. Accordingly, for simulated measurements, the quad GM-PHD filter outputs estimates of target type 1 (targets 1, 2, 3 and 4), target type 2 (targets 5, 6, 7 and 8), target type 3 (targets 9, 10, 11 and 12) and target type 4 (targets 13, 14, 15 and 16) being well differentiated which can not be handled even by intelligent track management as shown in Fig. 4.4 and Fig. 4.6. However, using four independent GM-PHD filters, estimates of targets 1, 2, 3, 4, 5, 9 and 13 are obtained from GM-PHD filter 1, estimates of targets 1, 5, 6, 7, 8, 10 and 14 from GM-PHD filter 2, estimates of targets 2, 9, 10, 11, 12 and 15 from GM-PHD filter 3, and estimates of targets 3, 7, 11, 13, 14, 15 and 16 from GM-PHD filter 4 i.e. targets 1, 2, 3, 5, 6, 7, 9, 10, 11, 13, 14 and 15 are estimated twice though intermittently, as shown in Fig. 4.3 and Fig. 4.5 overlaid. Even if the confusion rates increase to 0.6, the proposed method filters

out the target confusions effectively discriminating the target types. This confusion problem is solved by using our proposed approach with computation time of 264.41 seconds for 120 iterations when compared to 158.82 seconds for using four independent GM-PHD filters experimented on a Core i7 2.30 GHz processor and 8 GB RAM laptop using MATLAB when setting detection probabilities of confusion to 0.6, for example, as given in Table 4.1. In Fig. 4.3, Fig. 4.4, Fig. 4.5, Fig. 4.6, Fig. 4.7 and Fig. 4.8, E stands for end point whereas the other side of each target is starting point of the simulation.

When we set the probabilities of confusion to 0.0 i.e. no target confusions, the quad GM-PHD filter performs similar to four GM-PHD filters i.e. it degrades to four GM-PHD filters. However, if the values of confusion detection probabilities approach the values of the true detection probabilities as shown in Fig. 4.7 and Fig. 4.8, the quad GM-PHD filter still effectively filters the confusion in the detection of targets though it sometimes fails to discriminate the target types. This is illustrated in Fig. 4.8 which shows targets 2 and 3 (target type 1) are estimated as target type 3 (cyan asterisks) and target type 4 (black circles) respectively when setting the probabilities of confusion to 0.9 which is very close to the values of $p_{11,D} = 0.9$, $p_{22,D} = p_{33,D} = 0.92$ and $p_{44,D} = 0.91$. Similarly, targets 6 and 7 (target type 2) are estimated as target type 3 (cyan asterisks) and target type 4 (black circles), respectively. Target 11 (target type 3) is also filtered as target type 4 (black circles). If the probability of confusion is the same as of true detection, then the result is random on first guess (sometimes fails to discriminate the target types) though it still filters out the confusions effectively. Therefore, the values of the confusion probabilities ($p_{12,D}$, $p_{13,D}$, $p_{14,D}$, $p_{21,D}$, $p_{23,D}$, $p_{24,D}$, $p_{31,D}$, $p_{32,D}$, $p_{34,D}$, $p_{41,D}$, $p_{42,D}$ and $p_{43,D}$) should be less than the values of the true detection probabilities ($p_{11,D}$, $p_{22,D}$, $p_{33,D}$ and $p_{44,D}$) to discriminate the target types. However, in real applications (visual tracking), this does not happen i.e. the confusion detection probabilities can never become equal in values to the true detection probabilities as object detectors are at least becoming more accurate than random guessing i.e. nobody would employ a random detector.

On the other hand, if each target is regarded as a type (e.g. each of the four target types in this example has only one target), the N-type GM-PHD filter is used as a labeler of each target i.e. it discriminates those targets from frame to frame whether or not confusions between targets exist rather than simply degrading to the standard GM-PHD filter(s).

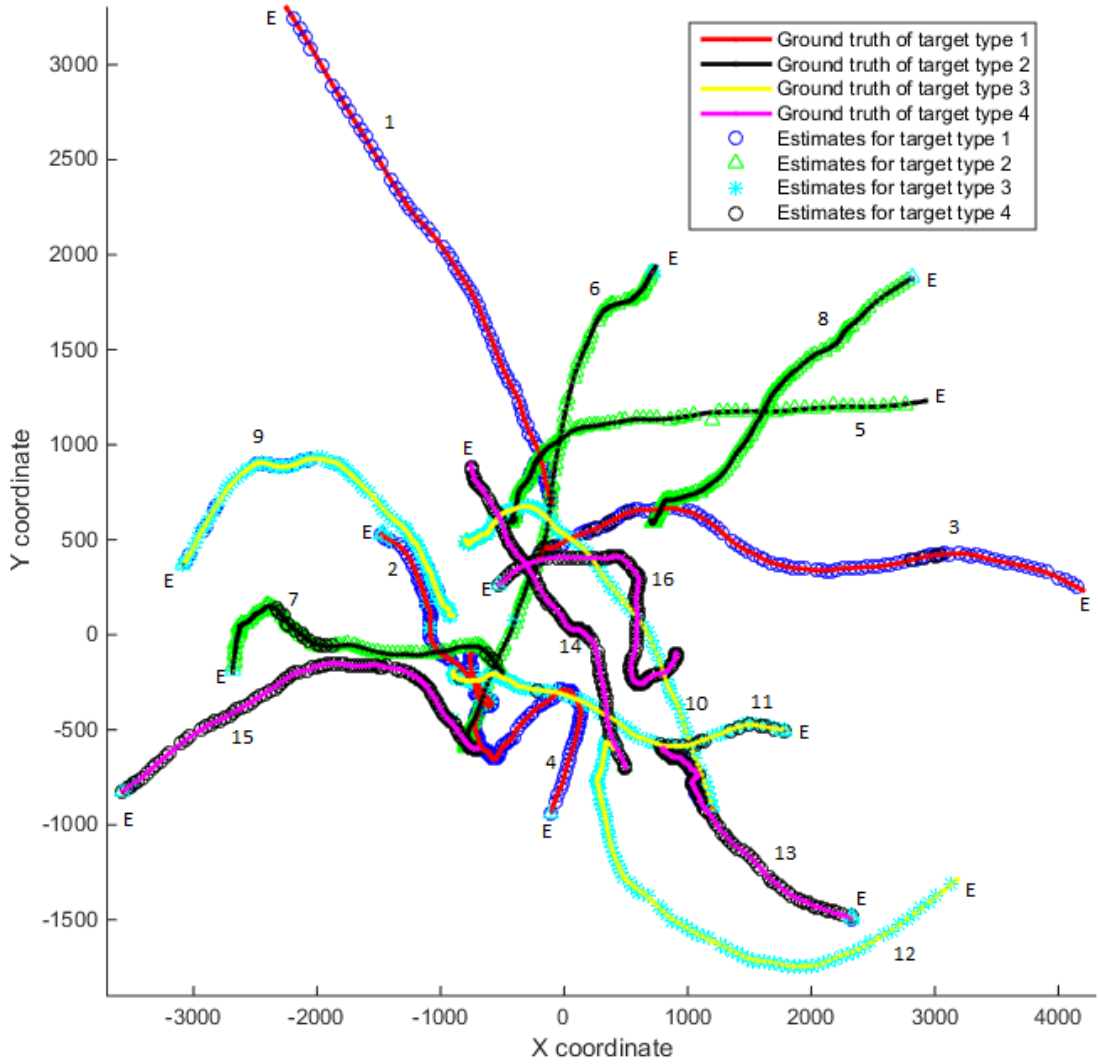


Figure 4.3: Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from four independent GM-PHD filters (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.3$.

Furthermore, we assess tracking accuracy using the cardinality (number of targets) and Optimal Subpattern Assignment (OSPA) metric [215]. From Fig. 4.9a (when setting the probabilities of confusion to 0.6), we observe that the cardinality of targets estimated using four independent GM-PHD filters (blue) has much more deviation from the ground truth (16 in red) when compared to the one obtained using our proposed quad GM-PHD filter (green). Similarly, the OSPA error of using four independent GM-PHD filters (blue) is much greater than that of using the quad GM-PHD filter (green) as shown in Fig. 4.9b. The overall average value of the OSPA error for four independent GM-PHD filters is 48.38m compared to 33.32m when using our proposed quad GM-

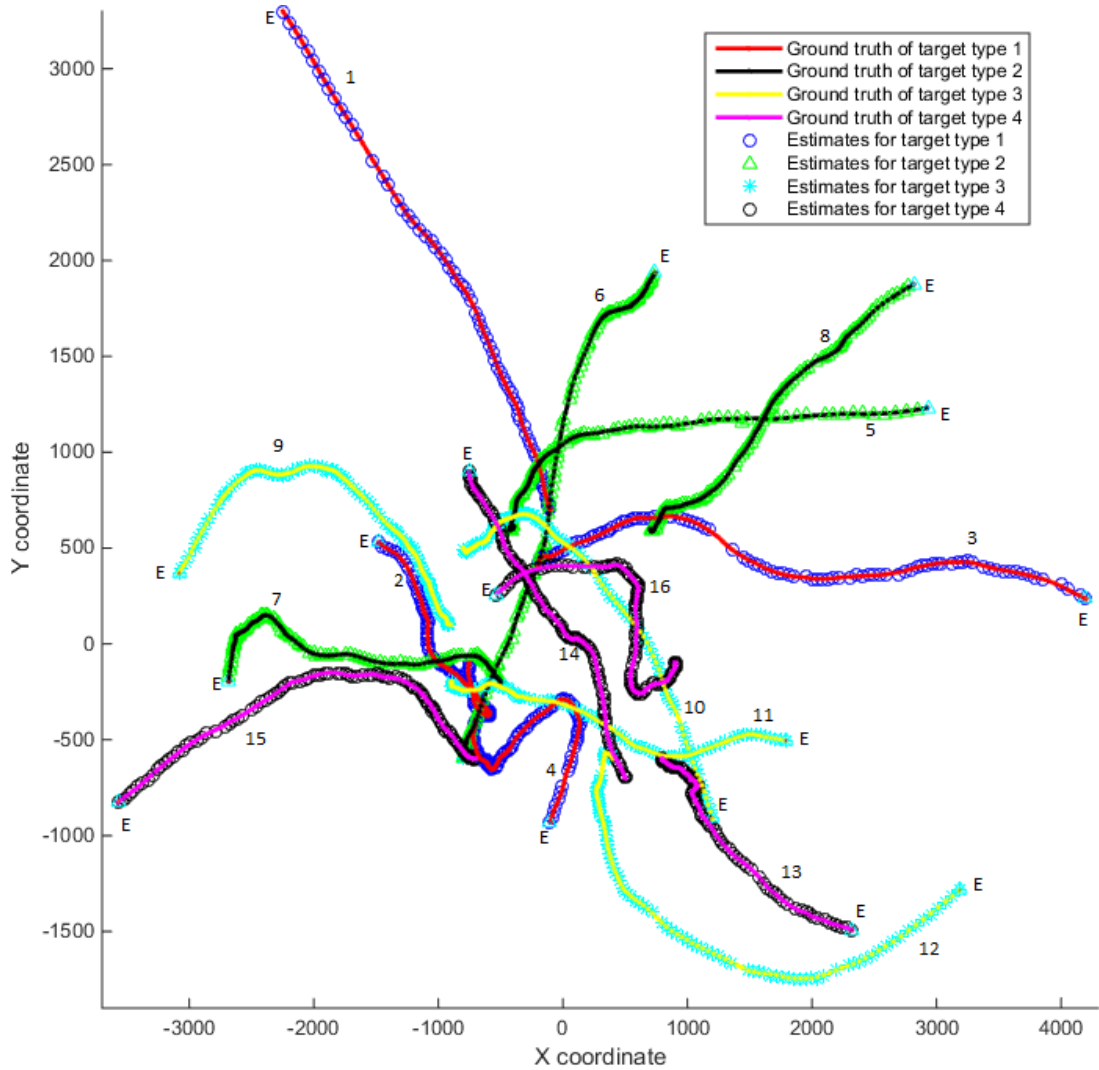


Figure 4.4: Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from quad GM-PHD filter (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.3$.

PHD filter as given in Table 4.1. The OSPA error and time taken (in brackets) when using probabilities of confusion of 0.3 and 0.9 are also given in Table 4.1. As can be observed from Fig. 4.10 and Table 4.1, as we increase the probabilities of confusions from 0.0 to 0.9, the OSPA error for quad GM-PHD filter is almost constant which shows how efficient the quad GM-PHD filter is in handling target confusions. However, for the case of using four independent GM-PHD filters, the OSPA error increases significantly as we increase the probabilities of confusions from 0.0 to 0.9 which is due to the increment of target confusions. The time taken (given in Table 4.1) also increases slightly for both methods with the increment of target confusions.

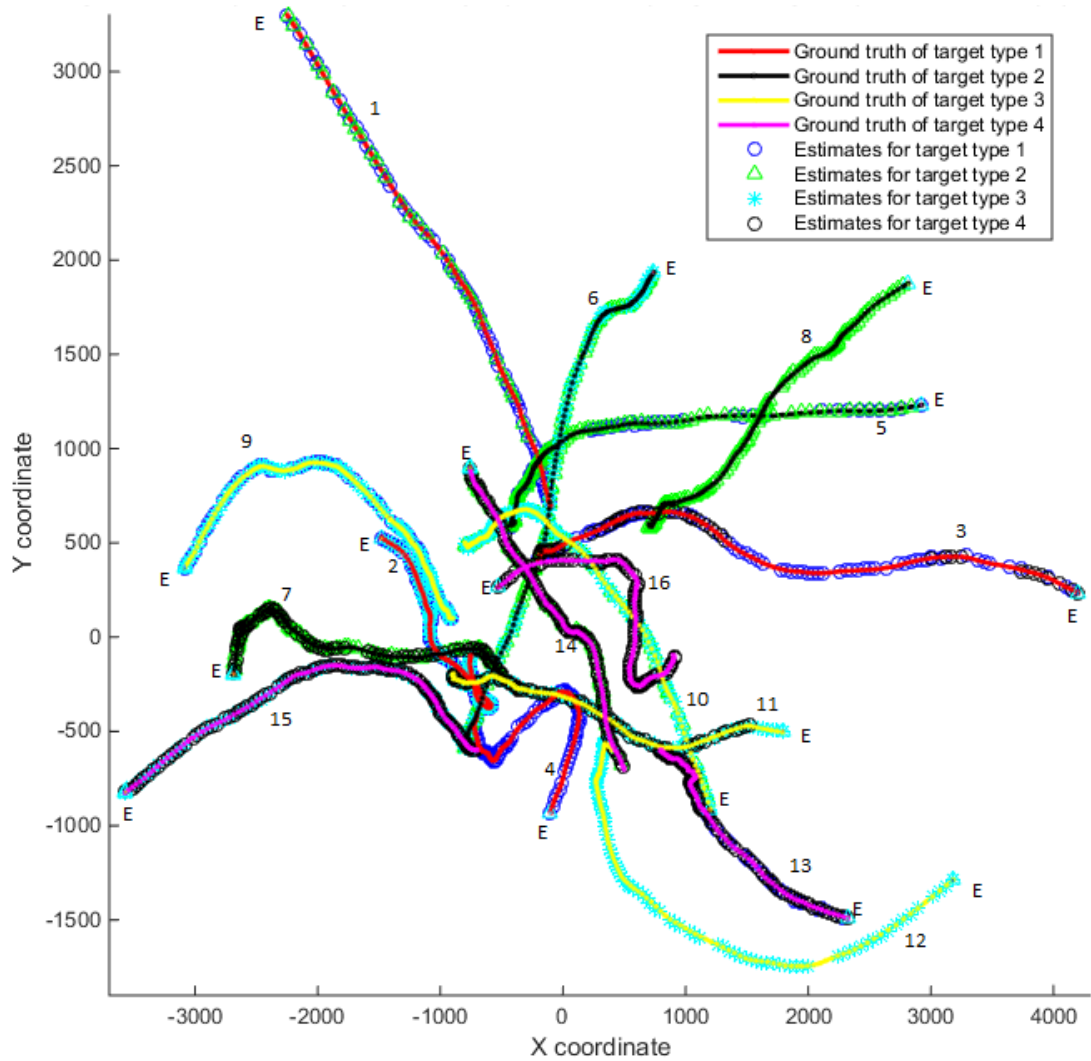


Figure 4.5: Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from four independent GM-PHD filters (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.6$.

Method	0.3	0.6	0.9
Quad GM-PHD filter	33.15m (253.97sec)	33.32m (264.41sec)	34.22m (291.27sec)
4 GM-PHD filters	35.57m (154.86sec)	48.38m (158.82sec)	56.53m (167.28sec)

Table 4.1: OSPA error at different values of probabilities of confusion $p_{12,D}, p_{13,D}, p_{14,D}, p_{21,D}, p_{23,D}, p_{24,D}, p_{31,D}, p_{32,D}, p_{34,D}, p_{41,D}, p_{42,D}$ and $p_{43,D}$ (0.3, 0.6 and 0.9) for quad GM-PHD filter and 4 independent GM-PHD filters. Time taken is given in brackets.

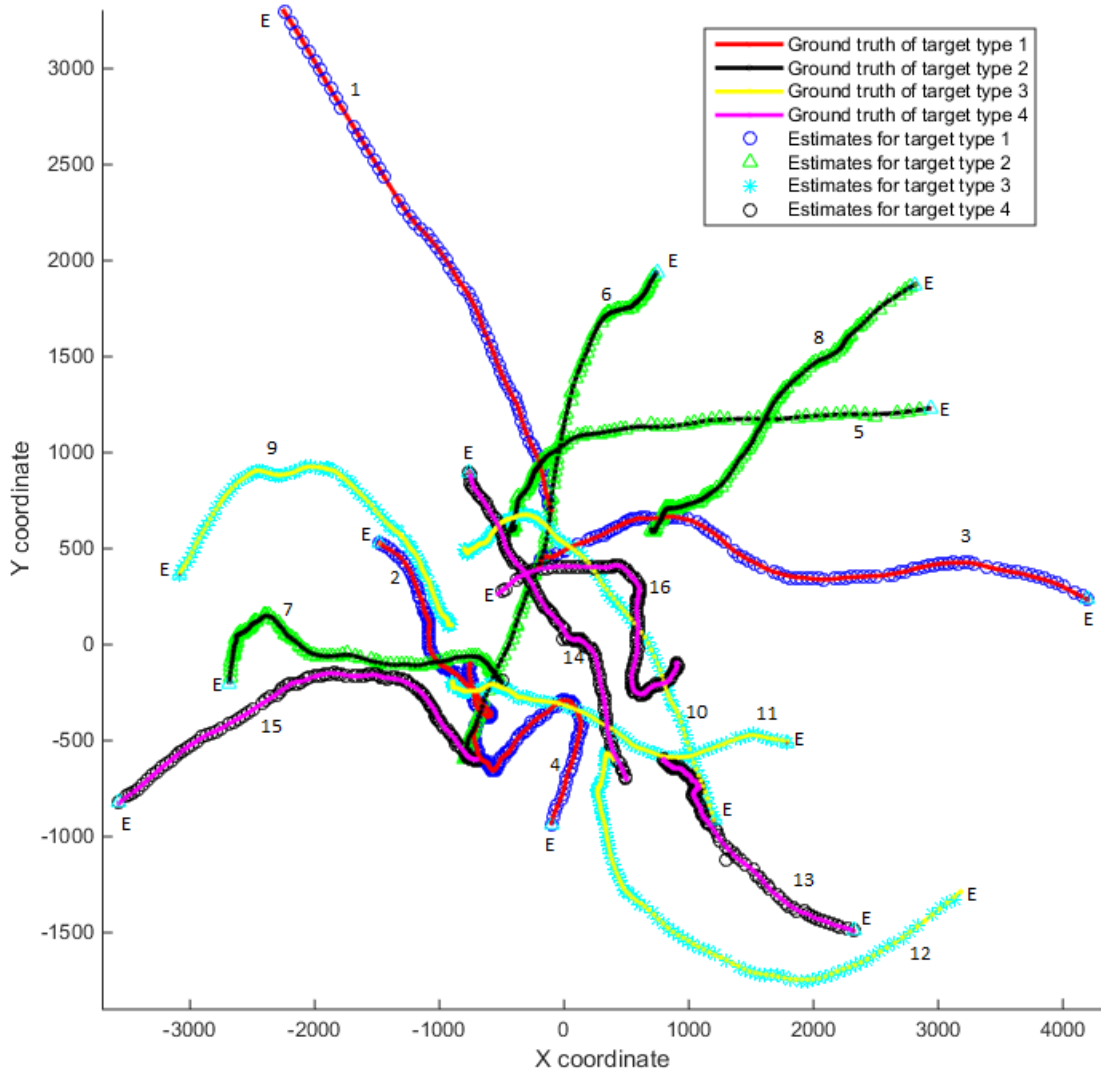


Figure 4.6: Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from quad GM-PHD filter (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.6$.

4.6 Summary

In this chapter, we propose a novel filter, N-type PHD filter where $N \geq 2$, in the RFS framework. In this approach, we assume that there are confusions between detections, i.e. clutter arises not just from background false positives, but also from target confusion. Under the Gaussianity and linearity assumptions, the Gaussian mixture (GM) implementation is proposed for N-type PHD filter, N-type GM-PHD filter. We evaluate the quad GM-PHD filter and compare to four independent GM-PHD filters, indicating that our approach shows better performance determined using cardinality, OSPA met-

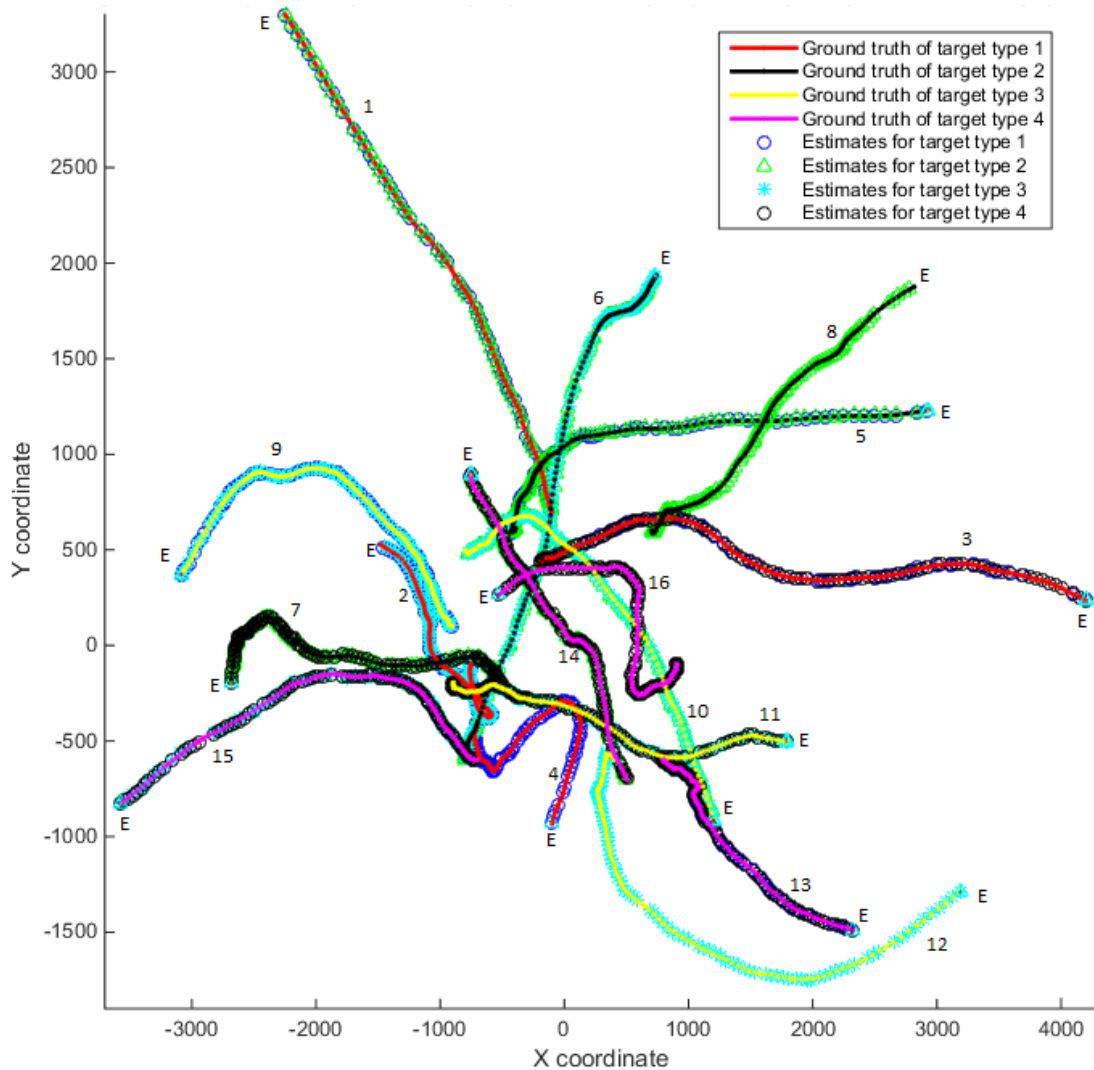


Figure 4.7: Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from four independent GM-PHD filters (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.9$.

ric and discrimination rate among the different target types. Even though, we show the simulation analysis for $N = 4$, in principle the methodology can be applied to N types of targets where N is a variable in which the number of possible confusions may rise as $N(N - 1)$. For instance, after experimenting the dual GM-PHD filter ($N = 2$) and the tri-GM-PHD filter ($N = 3$) by simulation and making sure that they show similar behaviour as for $N = 4$, they are applied for visual tracking applications in Chapter 5. In case there is no target confusion, the N -type GM-PHD filter performs similar to N independent GM-PHD filters. On the other hand, if each target is regarded as a type, the N -type GM-PHD filter is used as a labeler of each target i.e. it discriminates those

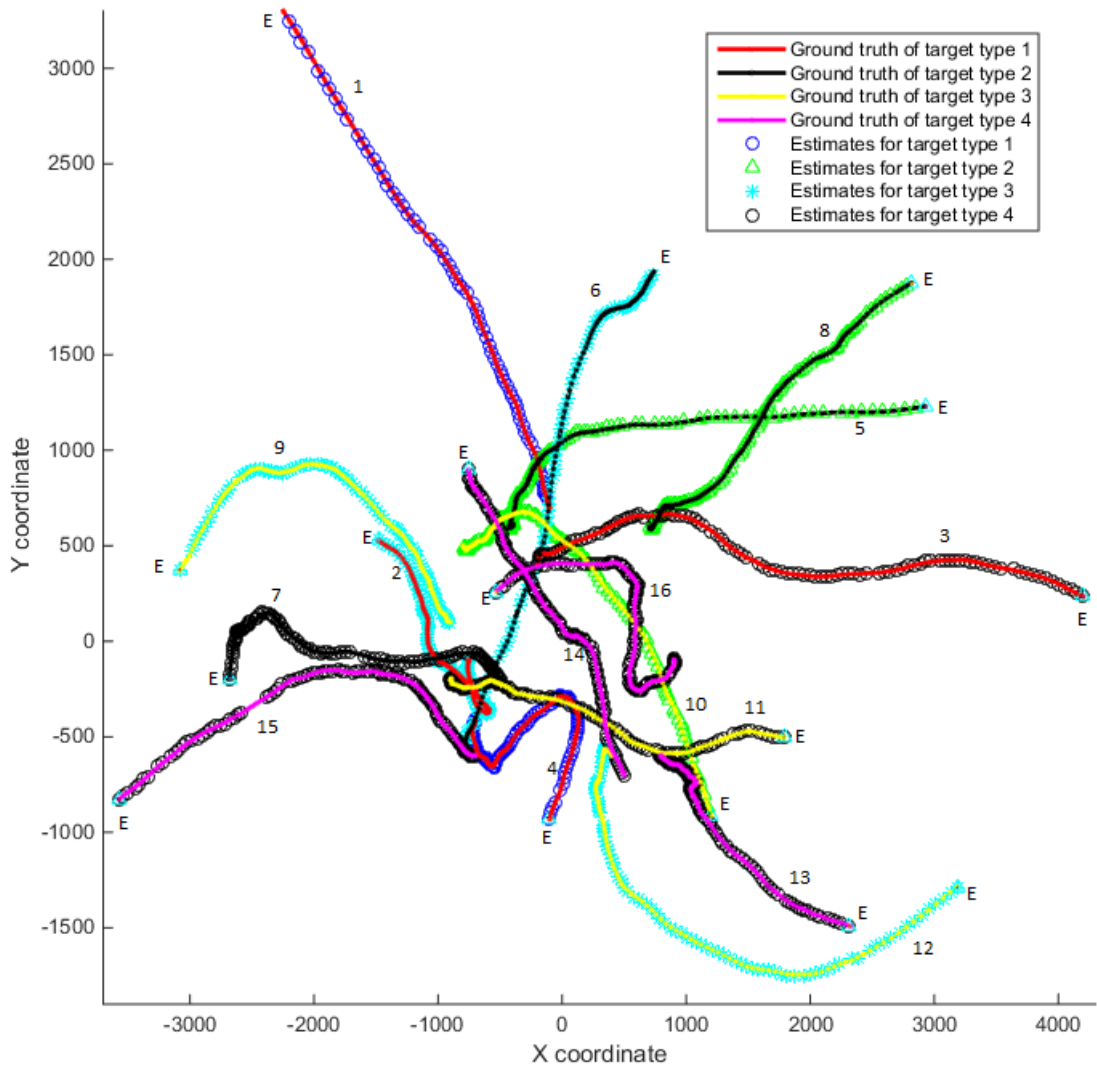


Figure 4.8: Simulated ground truth (red, black, yellow and magenta for target type 1, 2, 3 and 4, respectively) and position estimates from quad GM-PHD filter (blue circles, green triangles, cyan asterisks and black circles for target type 1, 2, 3 and 4, respectively) using $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.9$.

targets from frame to frame rather than simply degrading to the standard GM-PHD filter(s).

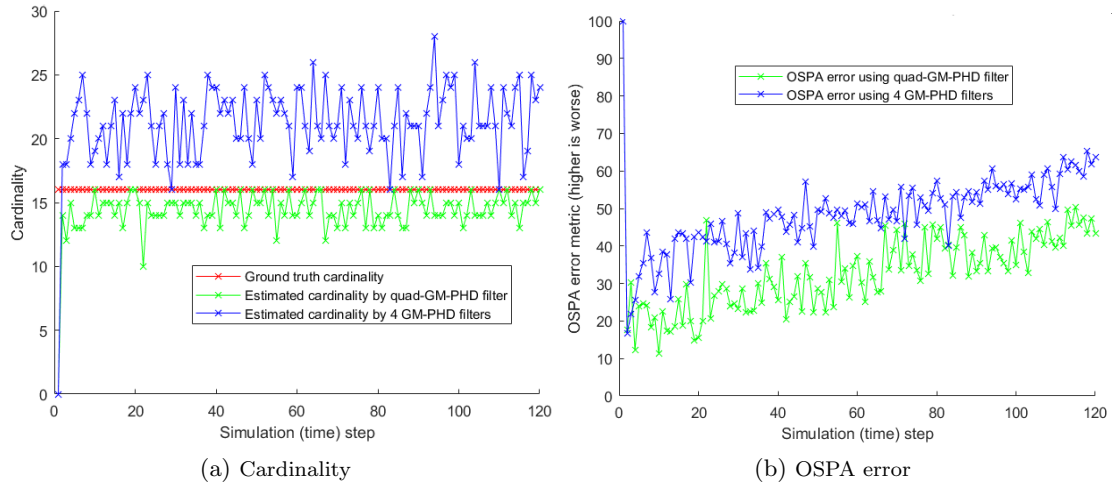


Figure 4.9: Cardinality and OSPA error: Ground truth (red for cardinality only), quad GM-PHD filter (green), four independent GM-PHD filters (blue) for $p_{12,D} = p_{13,D} = p_{14,D} = p_{21,D} = p_{23,D} = p_{24,D} = p_{31,D} = p_{32,D} = p_{34,D} = p_{41,D} = p_{42,D} = p_{43,D} = 0.6$.

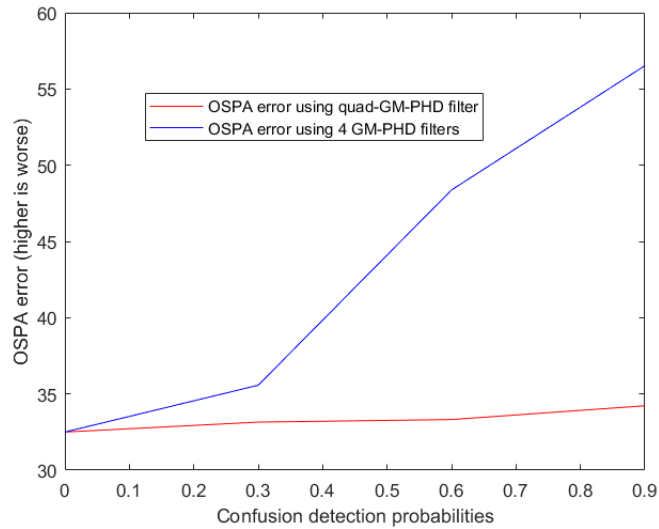


Figure 4.10: OSPA error comparison for quad GM-PHD filter and four independent GM-PHD filters at different probabilities of confusion.

Chapter 5

Multiple Target, Multiple Type Visual Tracking using a N-type GM-PHD Filter

In the previous chapter, a new N-type GM-PHD filter was developed which takes into account the effect of target confusion among different target types at the detection stage. In this chapter, we integrate the object detectors' information such as the probabilities of detections for each target type and the confusion detection probabilities among target types at a specific background clutter rate into the N-type GM-PHD filter to apply to visual tracking on real video sequences. Accordingly, we consider two scenarios. In the first scenario, we consider a sports analysis application where we want to track and discriminate sub-groups of the same target type, football players in opposing teams (and a referee), as discussed in section 5.1. In the second scenario, we want to distinguish between different target types, vehicles and more vulnerable road users such as pedestrians and bicycles, which is crucial for situational awareness, driver assistance and vehicle autonomy. In this case we focus on pedestrians and vehicles as two different target types and this is presented in section 5.2. Important components in visual tracking such as object detection and data association for each case are also discussed. We also evaluate the performance of our tracker and compare to both raw detection and independent GM-PHD trackers for each scenario. The building block of the N-type GM-PHD tracker (the N-type GM-PHD tracker for $N = 1$) is evaluated using Multi-Object Tracking (MOT) benchmarking tool and compared to the state-of-the-art algorithms in the MOT challenge in section 5.3. Finally, a brief summary of the application of the N-type GM-PHD filter to real video sequences is given in section 5.4.

5.1 Multiple Target, Implicit Multiple Type Tracking using a Tri-GM-PHD Filter

In this part, we consider tracking of football teams and a referee in the same scene handling their confusions using a tri-GM-PHD filter. We call it implicit multiple type since the multiple types we are dealing with are fundamentally the same target type but grouped into sub-groups which we try to track and discriminate by handling their confusions. Accordingly, we discuss the football teams and referee detectors in subsection 5.1.1, apply a tri-GM-PHD filter in subsection 5.1.2, apply data association algorithm in subsection 5.1.3, and analyze experimental results in subsection 5.1.4.

5.1.1 Object Detection, Training and Evaluation

The RFS methodology post-processes a set of detections with parameters defining the probabilities of detection and clutter (false alarms). For the tri-PHD filter, we also need parameters for confusion. We employ the existing, state-of-the-art, Aggregated Channel Features (ACF) pedestrian detection algorithm [16] adapted to our data set due to its computational efficiency and ease of use. This uses three different kinds of features in 10 channels: normalized gradient magnitude (1 channel), histograms of oriented gradients (6 channels), and LUV color (3 channels). It is applied to detect the actors (football teams and a referee) using a sliding window at multiple scales. The Adaboost classifier [223] is used to learn and classify the feature vectors acquired by the ACF detector.

For training, evaluation and parameter setting we use the VS-PETS'2003 football video data¹. This data set consists of 2500 frames which have players from the red and white teams and the referee. We trained three separate detectors for each target type (red, white, referee). We used every 10'th frame, i.e. 240 frames taken from the last 2400 frames. From these 240 frames, we collected 2000 positive samples for each footballer type, 240 samples for the referee, and 5000 randomly selected negative samples. This captures the appearance variation of players due to articulated motion. The correct player type or referee positions and windows were labeled manually for training as positive samples. The first 100 frames (video) are used to evaluate and test the tri-GM-PHD filtering process in comparison with repeated detection and three separate GM-PHD filters in subsection 5.1.4.

¹<http://www.cvg.reading.ac.uk/slides/pets.html>

The RFS methodology assumes point detections and a Gaussian error distribution on locations accuracy. However, humans in a video sequence are extended targets and the ACF detector has a bounding box that encloses the target. Therefore, overlapping detections are merged using a greedy non-maximum suppression (NMS) overlap threshold (intersection over union of two detections) of 0.05 (we made the overlap threshold very tight to ignore multiple bounding boxes on the same object). However, when evaluating the detectors, an overlap threshold (intersection over union of detection and ground truth bounding boxes) of 0.5 is used to identify true positives vs false positives. The receiver operating characteristic (ROC) curves for each of the detectors evaluated on the test video are given in Fig. 5.1.

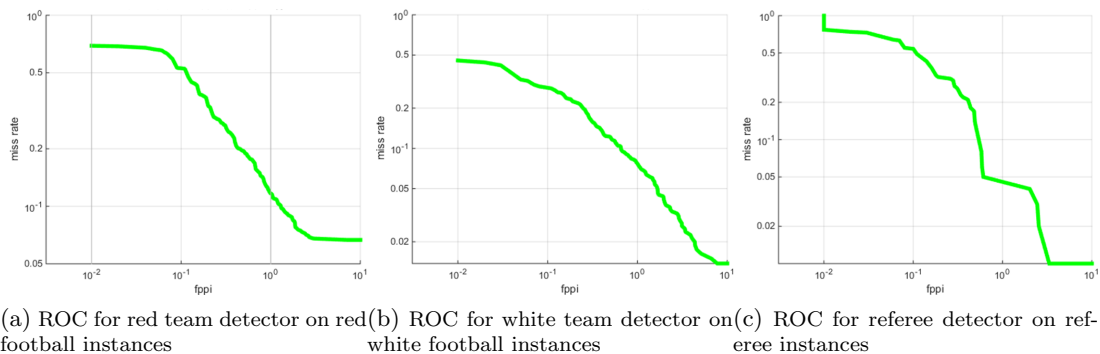


Figure 5.1: Extracting detection probabilities for three target types (red, white and referee) from ROCs of 3 detectors: red team detector, white team detector and referee detector when tested on red team, white team and referee instances, respectively.

For the tri-GM-PHD strategy, we must set the thresholds on detection from the ROC curves in Fig. 5.1, taking into account the probabilities of confusion that arise from the corresponding ROC curves (not shown) of each detector applied to targets of a confusing type. From our own simulations and the published literature, e.g. [172, 176], we know that the RFS methodology is most effective when applied with a high probability of detection, albeit with a higher clutter rate, and in our case a higher confusion rate. Obviously, for a target detection to be useful, the probability of true detection must be higher than the probability of confusion. Therefore, from Fig. 5.1, we standardise a clutter rate of 10 false positive per image (fppi), which gives probabilities of detection of 0.93 (p_{11}), 0.99 (p_{22}) and 0.99 (p_{33}) for red, white and referee, respectively. With these values, the corresponding confusion parameters are 0.24 (white footballer detected as red, p_{21}), 0.5 (referee as red, p_{31}), 0.24 (red as white, p_{12}), 0.18 (referee as white, p_{32}), 0.19 (red as referee, p_{13}) and 0.17 (white as referee, p_{23}).

5.1.2 Tracking Football Teams and Referee

The key steps of the tri-GM-PHD filter are summarised in Algorithms 2, 3 and 4 where $N = 3$. These are expressed in terms of frames k and $k - 1$; for the first frame, $k = 1$, of a sequence there is only detection and target birth, but no prediction and update for existing targets. For subsequent frames, we have chosen measurement driven target birth, rather than a random or a-priori birth model, inspired by but not identical to [176] in this visual tracking context. Maggio et al. [146] also assume that targets are born in a limited volume around measurements. The advantage of random birth is in the potential detection of weak target signatures, but in these examples the presence of a human should, in general, generate a strong probability of detection provided the target is in view. This is borne out by experiments and parameter setting in subsection 5.1.4. A further disadvantage of random birth is the increased complexity of processing a large number of incorrect targets. For targets moving in video sequences there is no spawn process, but occlusions do result anywhere in the field of view, and may be caused either by other targets or other obstacles. Re-emerging targets are detected and constitute births, are not spawned because they may be occluded by obstacles other than targets, and have no a-priori location.

The prediction and update, steps 2 to 4 (of Algorithm 2), follow the standard procedures for the GM-PHD filter [172] but are extended to take into account the three detection processes and the subsequent confusion between detections. In the proposed algorithm, birth and prediction both precede the construction and update of the PHD components, so the total number at the conclusion of step 4 is the sum of the persistent and birthed components. When applying to visual tracking, the weights update in the update step (step 4 of Algorithm 2) need to be carefully tuned unlike in the simulation case. Thus, we updated the weights using only the position component of the measurement since updating the weights over all components (position and bounding box) produces unacceptably low weights which in turn provides unexpected results. The number of Gaussian components in the posterior intensities may increase without bound as time progresses, particularly as a birth at this stage may be due to an existing target that has moved from the previous frame and then is re-detected in the current frame. Therefore, it is necessary to prune weak and duplicated components in Algorithm 3. First, weak components with weight $w_{i,k}^{(v)} < T = 10^{-5}$ are pruned. Further, Gaussian components with Mahalanobis distance less than $U = 4$ pixels from each other are merged. These pruned and merged Gaussian components, output of Algorithm 3, are predicted as existing targets in the next iteration. Finally, Gaussian components of the posterior intensity, output of Algorithm 2, with means corresponding to weights greater than 0.5 as a threshold are selected as multi-target state estimates as shown in Algorithm 4.

5.1.3 Data Association

The tri-GM-PHD filter distinguishes between true and false targets of each type. However, this does not distinguish between two different targets of the same type, so an additional step can be applied if we wish to identify different targets of the same type between consecutive frames. Although not part of the tri-GM-PHD strategy, this is commonly required so we include results from this post-labeling process for completeness in subsection 5.1.4. It does not affect our error metrics but is a post-process to label individuals from frame to frame. For data association (labeling), the Euclidean distance between each previous filtered centroid (track) and the current filtered centroids is computed and we compute an assignment which minimizes the total cost returning assigned tracks to current filtered outputs (min-cost matchings). This assignment problem represented by the cost matrix is solved using Munkres's variant of the Hungarian algorithm [73] which has time complexity of $O(n^3)$ i.e. cubic with the number of targets, and it is used for associating only two frames.

This also returns the unassigned tracks and unassigned current filtered results. The unassigned tracks are deleted and the unassigned current filtered outputs create new tracks if the targets are not created earlier. If some targets are miss-detected and incorrectly labeled, labels are uniquely re-assigned by re-identifying them using the approach in [224].

It is important to notice here that other combinatorial optimization based data association algorithms which generalize for three or more frames such as the Greedy algorithm (for graph matching) [74], min-cost network flow [75] [76] [77], and multi-dimensional assignment [78] can also be used.

5.1.4 Experimental Results

Referring to Eq.(4.1), our state vector includes the centroid positions, velocities, and the width and height of the bounding boxes, i.e. $x_k = [p_{cx,xk}, p_{cy,xk}, \dot{p}_{x,xk}, \dot{p}_{y,xk}, w_{xk}, h_{xk}]^T$. Similarly, the measurement is the noisy version of the target area in the image plane approximated with a $w \times h$ rectangle centered at $(p_{cx,zk}, p_{cy,zk})$ i.e. $z_k = [p_{cx,zk}, p_{cy,zk}, w_{zk}, h_{zk}]^T$.

As stated above, the detection and confusion probabilities are set by experimental evaluation of the ACF detection processes. Additional parameters are set from simulation and previous experience. For each target type, we set survival probabilities $p_{1,S} = p_{2,S} = p_{3,S} = 0.99$, and we assume the linear Gaussian dynamic model of

Eq. (4.30) with matrices taking into account the box width and height at the given scale.

$$F_{i,k-1} = \begin{bmatrix} I_2 & \Delta I_2 & 0_2 \\ 0_2 & I_2 & 0_2 \\ 0_2 & 0_2 & I_2 \end{bmatrix},$$

$$Q_{i,k-1} = \sigma_{v_i}^2 \begin{bmatrix} \frac{\Delta^4}{4} I_2 & \frac{\Delta^3}{2} I_2 & 0_2 \\ \frac{\Delta^3}{2} I_2 & \Delta^2 I_2 & 0_2 \\ 0_2 & 0_2 & \Delta^2 I_2 \end{bmatrix}, \quad (5.1)$$

where I_n and 0_n denote the $n \times n$ identity and zero matrices, respectively and Δ is the sampling period defined by the time between frames (we use 1 second). $\sigma_{v_i} = 5$ pixels/ s^2 is the standard deviation of the process noise for target type i where $i \in \{1, 2, 3\}$ i.e. type 1 (red football team), target type 2 (white football team) and target type 3 (a referee).

Similarly, the measurement follows the observation model of Eq. (4.31) with matrices taking into account the box width and height,

$$H_{ii,k} = H_{ji,k} = \begin{bmatrix} I_2 & 0_2 & 0_2 \\ 0_2 & 0_2 & I_2 \end{bmatrix},$$

$$R_{ii,k} = \sigma_{r_{ii}}^2 \begin{bmatrix} I_2 & 0_2 \\ 0_2 & I_2 \end{bmatrix},$$

$$R_{ji,k} = \sigma_{r_{ji}}^2 \begin{bmatrix} I_2 & 0_2 \\ 0_2 & I_2 \end{bmatrix}, \quad (5.2)$$

where $i \in \{1, 2, 3\}$, $j \in \{1, 2, 3\}$, and $\sigma_{r_{ii}}$ and $\sigma_{r_{ji}}$ are the measurement standard deviations taken from the distribution of distance errors of the centroids from ground truth in the evaluation of the detection process though truncated due to true detection overlap criterion, effectively 6 pixels.

Accordingly, in our approach, positive detections specify the possible birth locations with the initial covariance given in Eq.(5.3). The current measurement and zero initial velocity are used as a mean of the Gaussian distribution using a pre-determined initial

covariance for birthing of targets, i.e. new targets are born in the region of the state space for which the likelihood will have high values. Precisely, the birthing of targets is completely automatic using the very recent measurements obtained from object detectors. Very small initial weight (e.g. 10^{-4}) is assigned to the Gaussian components for new births as this is effective for high clutter rates. This is basically equivalent to the average number of appearing (birth) targets per frame (n_b) divided uniformly across the frame resolution (A).

$$P_{i,\gamma,k} = \text{diag}([100, 100, 25, 25, 20, 20]). \quad (5.3)$$

where $i \in \{1, 2, 3\}$.

We evaluate the tracking methodology of the tri-GM-PHD tracker in comparison with first, repeated independent detection on each frame, and second, with three independent GM-PHD trackers. Using the football video sequence, the examples shown in Fig. 5.2, Fig. 5.3 and Fig. 5.4 are for repeated detection (no tracking), three independent GM-PHD trackers, and the tri-GM-PHD tracker for frames 25, 57 and 73, respectively. Hence, Fig. 5.3a designates detections in which the red footballers, white footballers and the referee are detected both correctly and incorrectly, i.e. one object may be detected by many detectors. In this example the referee is detected 3 times: by the red team detector (red), by the white team detector (yellow) and the referee detector (black). Moreover, there are many background false positives (clutter) in the scene that arise from our choice to set the detection probability high at the expense of higher clutter as this is the detection scenario that is favored by the PHD process. Using the three independent GM-PHD trackers to effectively eliminate false positives, confused detections are not resolved as shown in Fig. 5.3b. However, our proposed tri-GM-PHD tracker effectively eliminates the false positives as well as confused detections as shown in Fig. 5.3c.

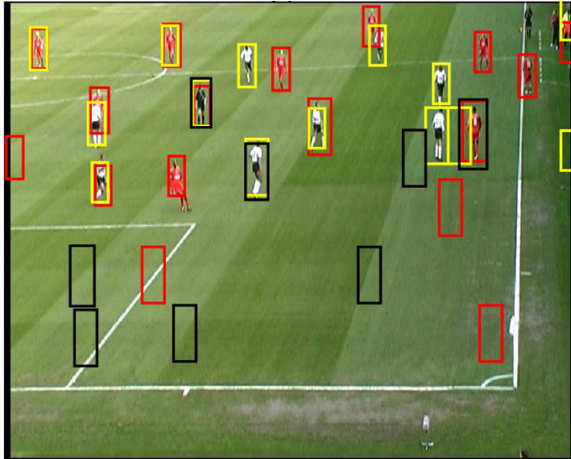
The tri-GM-PHD filter is evaluated quantitatively for the whole test sequence and compared with three independent GM-PHD filters and repeated raw detections using cardinality, OSPA metric [215], discrimination rate and time taken. We use the OSPA metric which is designed for evaluating RFS-based filters rather than multi-object tracking accuracy (MOTA) [213] which is widely used for evaluating other traditional multi-target tracking algorithms [161, 162]. Our algorithm is developed not only for tracking but also for discriminating different target types overcoming their confusions unlike algorithms such as [161, 162]. Therefore, the OSPA is the right evaluation metric to compare our approach with repeated raw detection and three independent GM-PHD trackers. The computational figures arise from experiments on a i5 2.50 GHz core processor with 6

GB RAM laptop using MATLAB and we acknowledge that these are not definitive and give a rough guide only to implementation costs. Though labeling of the targets using Munkres’s variant of the Hungarian assignment algorithm works well as shown in Figs. 5.2c, 5.3c and 5.4c, we did not include this in our evaluation as it is not part of the quantitative comparison of the filtering and type labeling of either the detections or distinct GM-PHD filters. We present the cardinality and OSPA error plots in Fig. 5.5a and Fig. 5.5b respectively, in red for ground truth (cardinality), green for the tri-GM-PHD filter, blue for the three independent GM-PHD filters and magenta for repeated detections. As summarised in Table 5.1 the average absolute cardinality error using raw detections is 10.22, reduced to 5.76 using the standard GM-PHD filters and to 0.11 (below 1 target) using the tri-GM-PHD filter. The overall frame-averaged value of OSPA error for the tri-GM-PHD filter is 10.59 pixels, compared to three independent GM-PHD filters of 30.86 pixels, and repeated detections of 37.61 pixels. The proposed approach reduces the cardinality and OSPA errors by a large margin over three independent GM-PHD filters and repeated detections, although this has more computational cost as also shown in Table 5.1.

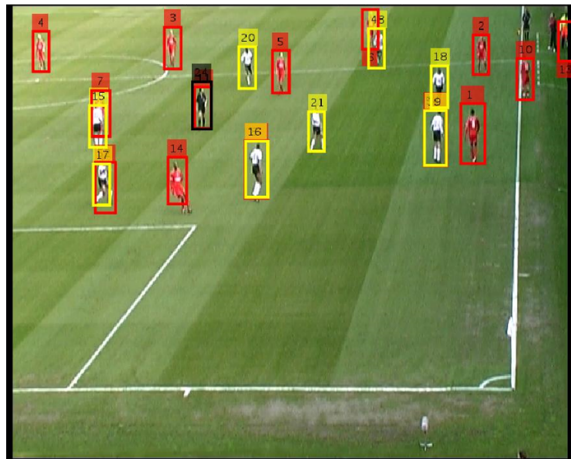
Independent GM-PHD trackers do not take confusion into account, so treat such confusion as ‘background’ clutter; the problem is that such confused detections are not likely to be accurately modeled by random detections distributed uniformly in space as is commonly the case. Our approach can effectively discriminate true positives from clutter, while eliminating confused detections with a discrimination rate of 99.20%. The mis-discrimination rate of 0.80% occurs primarily during the initial frames (e.g. the first 7 frames) until the prediction-update process stabilises and the true detections are confirmed by the motion between adjacent frames.

Method	cardinality error	OSPA error	time taken	discrimination rate
Detections	10.22	37.61 pixels	0.59 sec/frame	0%
3 GM-PHDs	5.76	30.86 pixels	0.80 sec/frame	0%
Tri-GM-PHD	0.11	10.59 pixels	3.00 sec/frame	99.20%

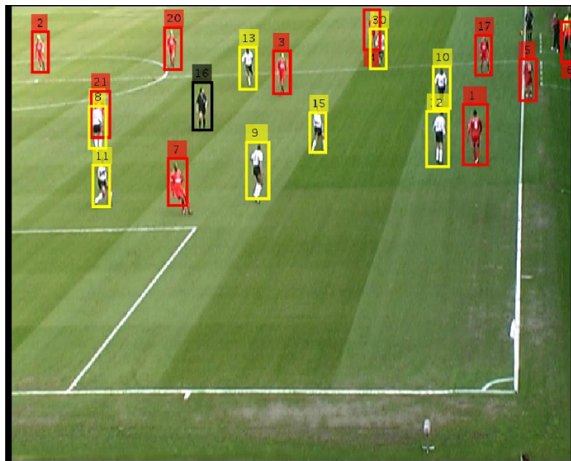
Table 5.1: Frame-averaged cardinality and OSPA errors, time taken and discrimination rate at the extracted detection probabilities for tri-GM-PHD filter, three independent GM-PHD filters and Detections.



(a) Detections, frame 25

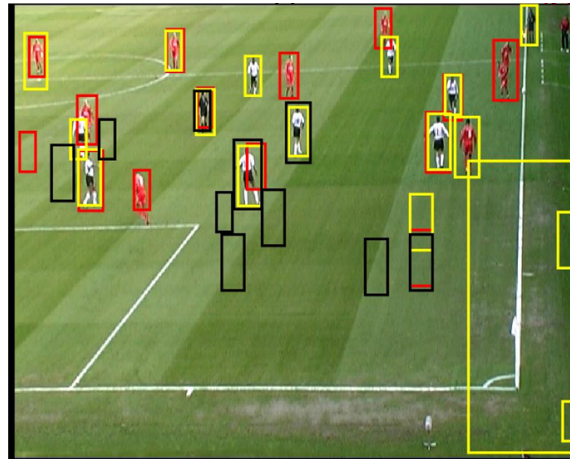


(b) Three independent GM-PHD trackers, frame 25

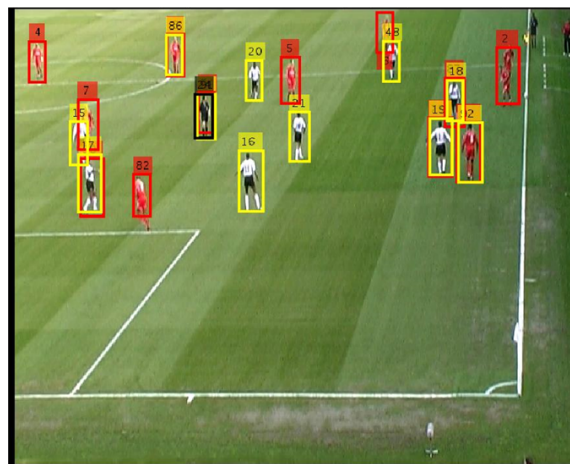


(c) Tri-GM-PHD Tracker, frame 25

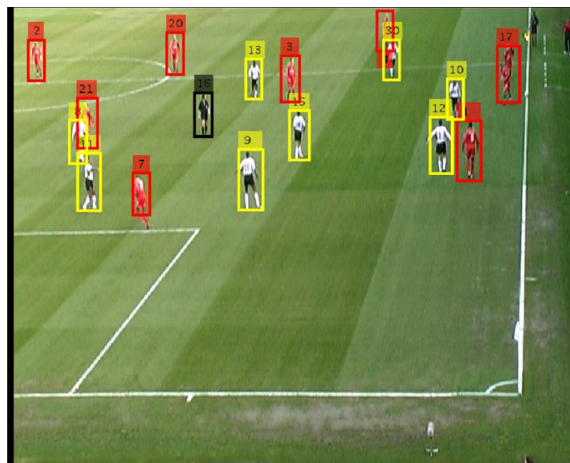
Figure 5.2: Results of detections, three independent GM-PHD trackers and tri-GM-PHD tracker, respectively, for frame 25.



(a) Detections, frame 57

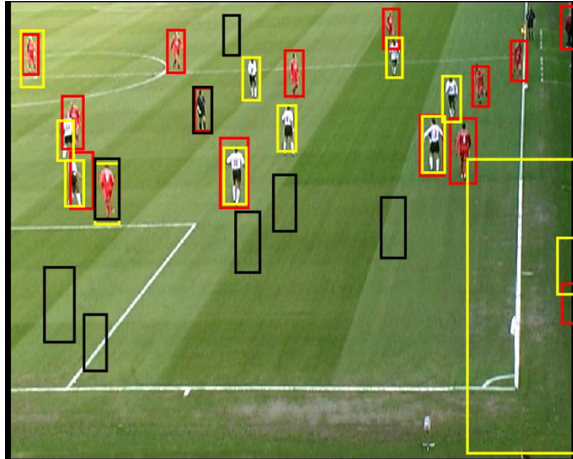


(b) Three independent GM-PHD trackers, frame 57

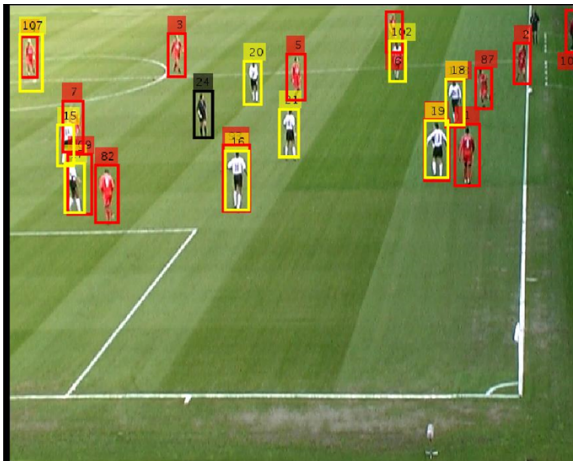


(c) Tri-GM-PHD Tracker, frame 57

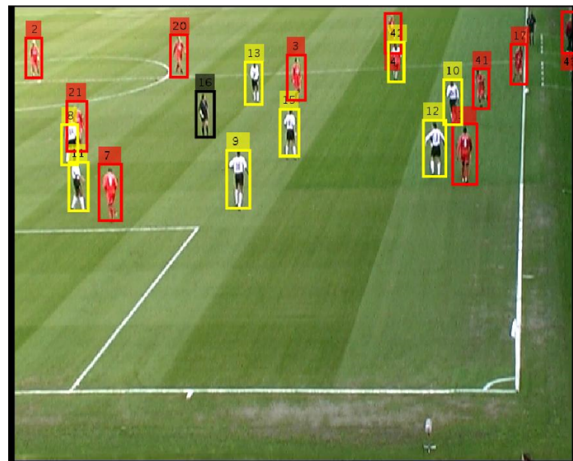
Figure 5.3: Results of detections, three independent GM-PHD trackers and tri-GM-PHD tracker, respectively, for frame 57.



(a) Detections, frame 73



(b) Three independent GM-PHD trackers, frame 73



(c) Tri-GM-PHD Tracker, frame 73

Figure 5.4: Results of detections, three independent GM-PHD trackers and tri-GM-PHD tracker, respectively, for frame 73.

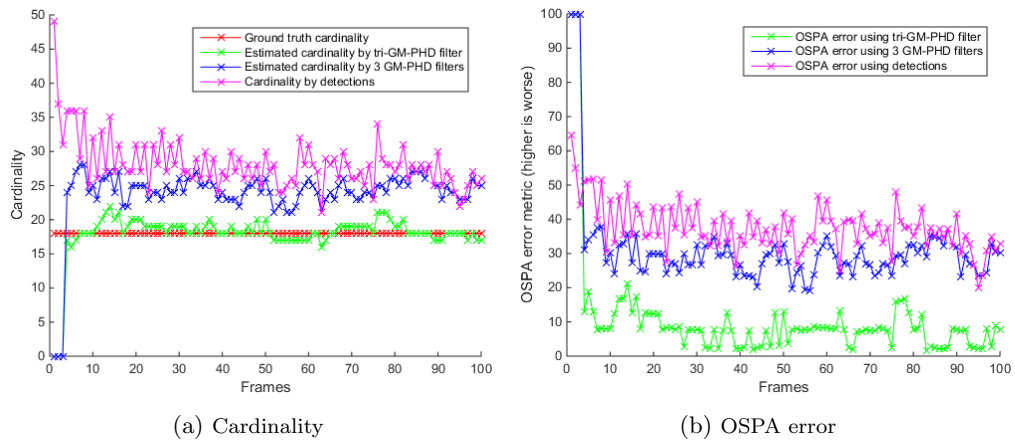
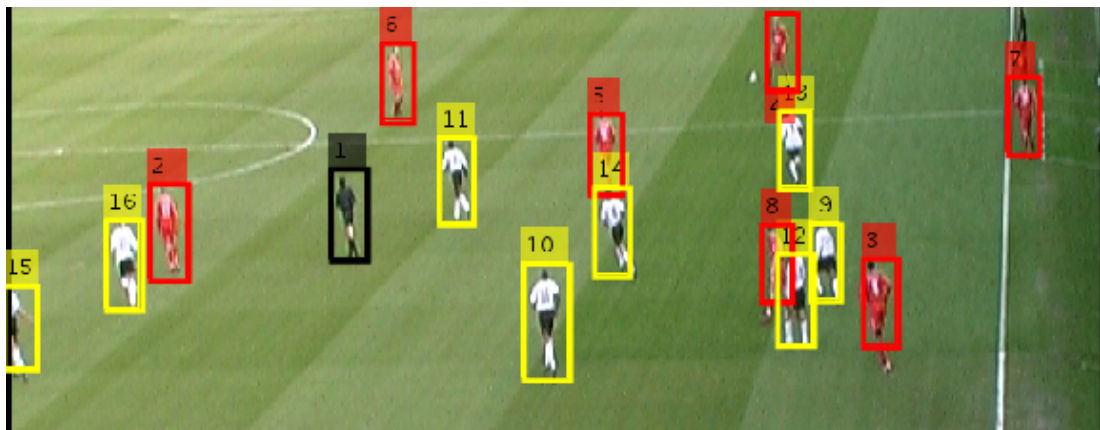
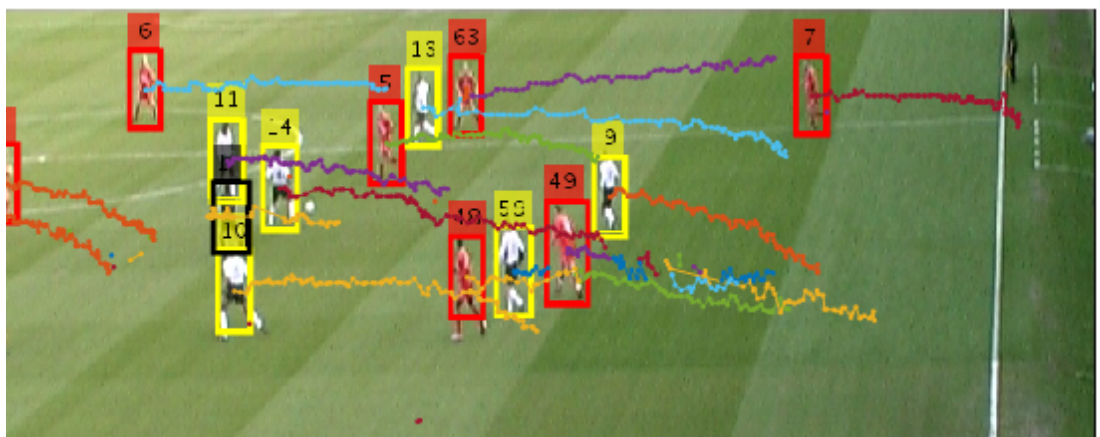


Figure 5.5: Cardinality and OSPA error: Ground truth (red for cardinality only), tri-GM-PHD filter (green), three independent GM-PHD filters (blue), detections (magenta).



(a) Frame 193



(b) Tracking both teams and referee, frame 293

Figure 5.6: Tracking the red and white teams, and referee from frame 193 to frame 293

Fig. 5.6 shows another example in which the individual footballers are detected, filtered, tracked and labeled for 100 frames. The image has been cropped as the action is confined to the top half of the image, and immediately follows a throw-in as the players move away left from the touchline. The examples also show the individual tracks and labels of the footballers and referee as small numbers over the targets. From this sequence, we see for example that the red player number 6 and the white player number 10, and several others, are consistently tracked through the sequence. However the labeling does occasionally make mistakes, for example red player 3 who starts near the touchline is finally labeled as red player number 49 in frame 293. In this instance the mislabeling is due to occlusion and lack of persistence in the detection and tracking as it uses successive frames only, so that if a player disappears then re-appears after several frames, he is treated as a new target. Nevertheless, although this evaluation is not part of the tri-GM-PHD filter, the labeling that we apply has good performance with a mean label switch error of only 0.43%.

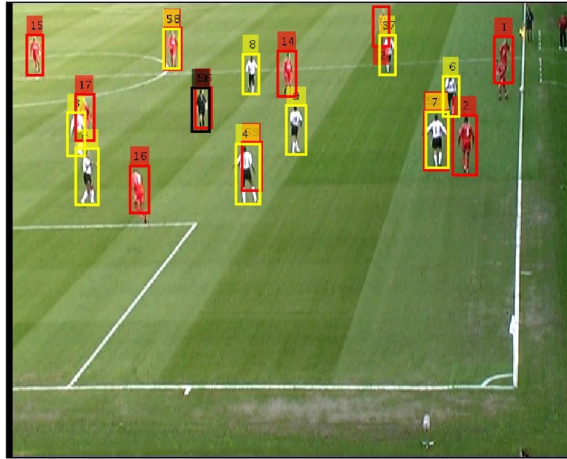
We also give a sensitivity analysis of the proposed method to variations of the parameters: low, medium and high confusion parameters. For the low confusion parameters, we standardise a clutter rate of 6 fppi from Fig. 5.1, which gives probabilities of detection of 0.93 (p_{11}), 0.98 (p_{22}) and 0.99 (p_{33}) for red, white and referee, respectively. With these values, the corresponding confusion parameters are 0.12 (white footballer detected as red, p_{21}), 0.2 (referee as red, p_{31}), 0.12 (red as white, p_{12}), 0.09 (referee as white, p_{32}), 0.1 (red as referee, p_{13}) and 0.08 (white as referee, p_{23}). Using these parameters, the output of the tri-GM-PHD tracker is given in Fig. 5.7a which corresponds to the detections of Fig. 5.3a (frame 57). As can be seen from this figure, these low confusion probabilities are not enough for fully handling the confusions of the detectors though the confusions of a significant number of targets are handled while discriminating them. The standardized parameters corresponding to fppi of 10 in Fig. 5.3c handle the confusions as well as discriminate the target types efficiently.

For medium confusion parameters, we standardise a clutter rate of 15 fppi, which gives almost the same probabilities of true detection as the one for fppi of 10 (given above) for all red, white and referee, however, with higher confusion parameters: 0.38 (white footballer detected as red, p_{21}), 0.53 (referee as red, p_{31}), 0.37 (red as white, p_{12}), 0.32 (referee as white, p_{32}), 0.33 (red as referee, p_{13}) and 0.31 (white as referee, p_{23}). This parameter setting efficiently handles the confusions caused by the detectors as shown in Fig. 5.7b. These values of confusion parameters are higher than the ones we used for Fig. 5.3c and performs similarly in handling confusions as well as discriminating the target types. However, as we increase these confusion probabilities to almost equal to the true detection probabilities (0.93 (p_{11}), 0.99 (p_{22}) and 0.99 (p_{33}) for red, white

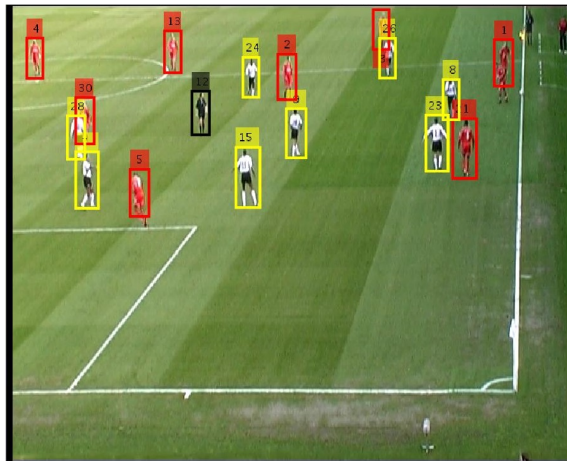
and referee, respectively), the tri-GM-PHD tracker handles the confusions efficiently but fails to discriminate some of the target types as shown in Fig. 5.7c, for instance, targets labelled by 35 and 38 are tracked as a red type. For this, we set the confusion parameters as 0.93 (white footballer detected as red, p_{21}), 0.99 (referee as red, p_{31}), 0.93 (red as white, p_{12}), 0.93 (referee as white, p_{32}), 0.93 (red as referee, p_{13}) and 0.93 (white as referee, p_{23}). We did not standardize fppi for this case (we simply set 16) as practically the confusion parameters cannot be as high as these values as the object detectors are much better than such random guessing i.e. our object detectors cannot give such high values of confusion parameters. We set these values only for analysis though not practical. From this experiment, we can observe that very high values of confusion parameters can sometimes cause the discrimination of target types to fail even in visual tracking application as it is observed in the simulation case in section 4.5. This is because the result is random on first guess for some targets as the true and confusion detection probabilities are almost equal in values which results in failure of discrimination of some targets (occurs only sometimes), however, the confusions of targets are still handled effectively. Therefore, we need to standardise a clutter rate in such a way that a reasonable true and confusion detection probabilities can be extracted from the ROC curves of their respective detectors for efficiently handling confusions of target types as well as discriminating them.

5.2 Multiple Target, Explicit Multiple Type Tracking using a Dual GM-PHD Filter

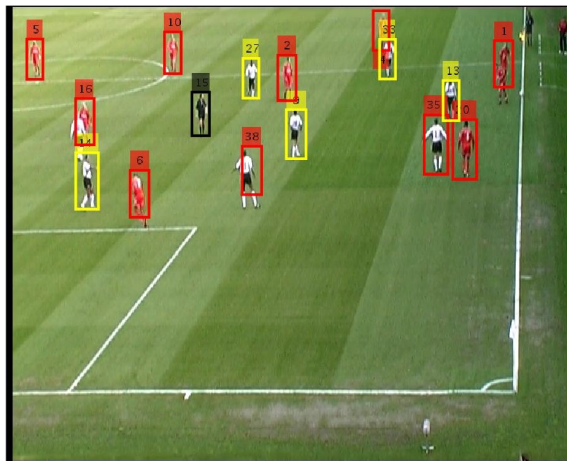
In this part, we consider tracking of pedestrians and vehicles in the same scene handling their confusions using a dual GM-PHD filter. We call it explicit multiple type since the multiple types we are dealing with are fundamentally different target types which we try to track and discriminate handling their confusions. The main difference between implicit and explicit target types is that in the latter case they can have different aspect ratios which can affect the means to differentiate the true and false positives in evaluating probabilities of confusion. Accordingly, we discuss the pedestrian and vehicles detectors in subsection 5.2.1, apply a dual GM-PHD filter and data association algorithm in subsection 5.2.2, and analyze experimental results in subsection 5.2.3.



(a) For low confusion parameters, frame 57



(b) For medium confusion parameters, frame 57



(c) For high confusion parameters, frame 57

Figure 5.7: Results of tri-GM-PHD tracker for different values of the confusion parameters, for frame 57, corresponding to detections of Fig. 5.3a.

5.2.1 Object Detection

Measurements for the dual GM-PHD filter are obtained using object detectors. Accordingly, we adapted the state-of-the-art pedestrian detection algorithm, ACF [16], which uses three different kinds of features in 10 channels: normalized gradient magnitude (1 channel), histogram of oriented gradients (6 channels), and LUV color (3 channels) by considering appearance variations. Similarly, we adapted the vehicle detection algorithm in [58] which uses the same type of features as the ACF pedestrian detector considering additional geometrical features such as truncation level, occlusion level and occlusion type features in addition to 3D orientation which depends on the ground truth information. However, we only consider 3D geometric orientation from the ground truth information available in the KITTI dataset [225]. Similar to [58], we also consider visual features to capture appearance variations due to varying orientation, truncation and occlusion degree for detecting both pedestrians and vehicles.

5.2.1.1 3D orientation

Appearance variation due to observation angle is common when detecting vehicles in different driving settings. Accordingly, the observation angle i.e. relative orientation of the object with respect to the camera is used by considering the angle of the vector joining the camera center in 3D and an object which takes into account the ego-vehicle. This 3D geometric orientation is available in KITTI dataset ranging from $-\pi$ (-3.14) to π (3.14) and is quantized into L labels ($L = 5$ for pedestrians and $L = 20$ for vehicles) as shown in Fig. 5.8. The mean of the aspect ratios of the image instances (samples) with the specific quantized label is used as an aspect ratio for which a specific detector model is trained on that specific image instances.

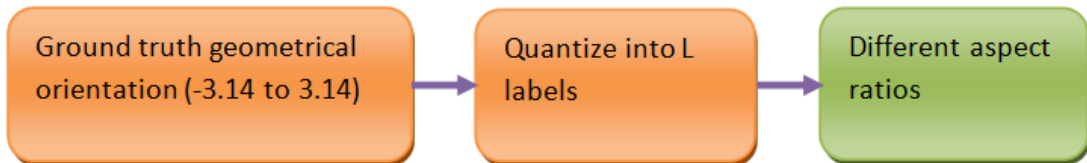


Figure 5.8: Quantizing 3D geometric orientation into L labels for obtaining required aspect ratios; for each of the aspect ratio, a specific detector is developed.

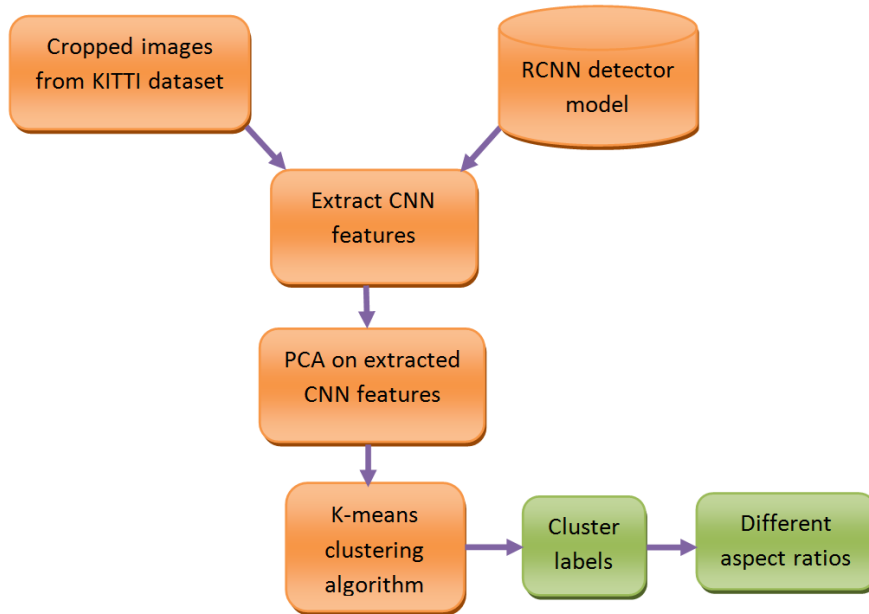


Figure 5.9: Clustering of CNN features.

5.2.1.2 Visual features

We use visual features by clustering them as a means of capturing appearance variations of objects to detect them under challenging appearance changes. This approach is very generic as it does not depend on the availability of ground truth orientation though it gives slightly less accuracy when compared to 3D geometrical orientation.

Though color and gradient features (HOG, LUV color and normalized gradient magnitude) can also be used [58], in our case, high quality convolutional neural network (CNN) features which give state-of-the-art classification results on ImageNet [18] as in [1] implemented efficiently in Caffe [49] are used to learn appearance variations of objects. First, ImageNet trained Caffemodel is fine-tuned to Pascal VOC data set [57] which is then used for extracting CNN features from each region of Pascal VOC data set to build a R-CNN object detector [52] in which category-specific SVMs are used. This is used as a transfer learning [56]. The R-CNN object detector model is then used to extract 4096-dimensional CNN features from cropped KITTI image samples which is then reduced dimensionally using PCA as shown in Fig. 5.9. This dimensionally reduced features are then clustered using k-means clustering giving cluster labels. The mean of the aspect ratios of the image instances assigned the same label is used to learn a specific detector on the image instances with that specific label.

5.2.1.3 Pedestrian Detection

The ACF pedestrian detector [16] uses 3 feature types in 10 channels to detect pedestrians at multiple scales using the Adaboost classifier. However, unlike the original ACF [16], we consider the appearance variations of pedestrians due to varying orientation, truncation and occlusion levels. The KITTI benchmark [225] consists of 7481 training frames from which around 3583 pedestrian instances are extracted in moderate setting. From these 7481 training frames, 6501 frames are used as the training set and the rest (980 frames) as the testing or validation set. To capture the appearance variations, we trained 5 geometrical orientations-based detection models and 5 visual CNN features clustering-based detection models which perform better than using only one model as in [16]. This is necessary as using only one model, it is not possible to get the required detection performance for extracting detection probabilities on this data set. Overlapping detections are merged using a greedy NMS overlap threshold (intersection over union of two detections) of 0.1. However, when evaluating the detector, an overlap threshold (intersection over union of detection and ground truth bounding boxes) of 0.5 is used to identify true positives vs false positives.

5.2.1.4 Vehicle Detection

Capturing appearance variations of vehicles due to changing observation angle, illumination variability, vehicle shape and type, truncation, out of camera view, different occlusion levels, etc is very important for developing a vehicle detector [58]. Unlike the approach considered in [58], we consider only 3D orientation rather than other geometrical features such as truncation level, occlusion level and occlusion type features from the ground truth available in KITTI dataset. Moreover, we used visual features which can also capture appearance variations due to varying orientation, truncation and occlusion. The KITTI benchmark [225] consists of 7481 training frames from which around 16105 car instances are extracted in moderate setting. From these 7481 training frames, 6501 frames are used as the training set and the rest (980 frames) as the testing or validation set. Accordingly, we trained 20 geometrical orientations-based detection models and 20 visual CNN features clustering-based detection models which perform better than using only one model. We use a greedy NMS overlap threshold (intersection over union of two detections) of 0.2 to merge overlapping detections, and an overlap threshold (intersection over union of detection and ground truth bounding boxes) of 0.5 is used to identify true positives vs false positives when evaluating the detector.

5.2.1.5 Detection Parameters Extraction

The detection parameters, detection probabilities ($p_{11,D}$, $p_{22,D}$) and confusion detection probabilities ($p_{12,D}$, $p_{21,D}$), are extracted as follows. The detection probability for pedestrians by a pedestrian detector, $p_{11,D}$, can be extracted from the ROC curve of the pedestrian detector when it is tested on pedestrian instances. Similarly, the detection probability for vehicles by a vehicle detector, $p_{22,D}$, is obtained from the ROC curve of the vehicle detector when it is tested on vehicle instances. The confusion detection probabilities, detection probability for pedestrians by a vehicle detector, $p_{12,D}$, and detection probability for vehicles by a pedestrian detector, $p_{21,D}$, can also be obtained when the vehicle detector is evaluated on pedestrian instances and when the pedestrian detector is evaluated on vehicle instances, respectively (refer to Fig. ??).

Accordingly, when we want to track on video sequences, we first run both pedestrian and vehicle detectors on that specific video sequence to obtain their ROC curves. We used a combination of CNN-based and 3D orientation-based detectors. Thus, the ROC curve of the pedestrian detector when applied to pedestrian instances in the KITTI video tracking sequence 16 [225] is shown in Fig. 5.10a from which $p_{11,D}$ of 0.83 is obtained at clutter rate (false positive per image - fppi) of 10. Similarly, $p_{22,D}$ of 0.86 is obtained at fppi of 10 from the ROC curve of the vehicle detector when it is applied to vehicle instances in KITTI video tracking sequence 16 as shown in Fig. 5.10b. However, the values of $p_{12,D}$ and $p_{21,D}$ are very low, around 0.03 for $p_{12,D}$ and 0.01 for $p_{21,D}$, this happens because even if the vehicle detector detects pedestrian instances, for example, the intersection of the detected bounding boxes by vehicle detector on pedestrian instances and the ground truth of the pedestrian instances is very low as the two bounding boxes have very much different aspect ratios, therefore, it can be classified as false positive though it is detected. Hence, we try to fine-tune values of $p_{12,D}$ and $p_{21,D}$ to some higher values e.g. $p_{12,D} = 0.3$ and $p_{21,D} = 0.1$ in this case.

5.2.2 Tracking Pedestrians and Vehicles, and Data Association

In this case, we use a dual GM-PHD filter by setting $N = 2$ in the Algorithms 2, 3 and 4 derived for the N-type GM-PHD filter. This dual GM-PHD tracker follows the same fashion discussed in subsection 5.1.2 including automated birthing of targets with the exception that we consider here two types of targets (pedestrians and vehicles) by using $N = 2$. The dual GM-PHD filter handles sensor noise, clutter and confusion between target types, therefore, labeling of targets can be applied at the end of the filter i.e. on

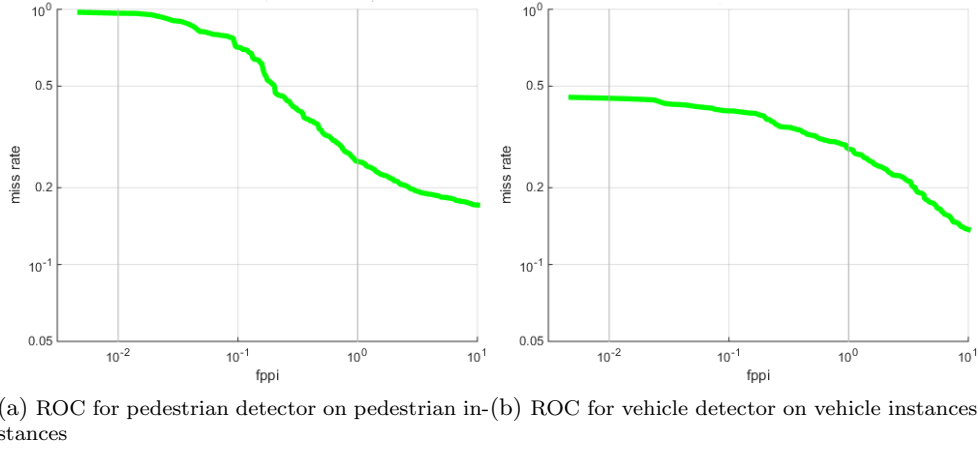


Figure 5.10: ROCs using 3D orientation and CNN visual features detector models tested on KITTI sequence 16.

the filtered outputs. Hence, Munkres’s variant of the Hungarian assignment algorithm is used to associate tracked target identities between two consecutive frames [73] in the same fashion discussed in subsection 5.1.3. In this case, if a target disappears and then reappears, a new label is given without using any re-identification algorithm.

5.2.3 Experimental Results

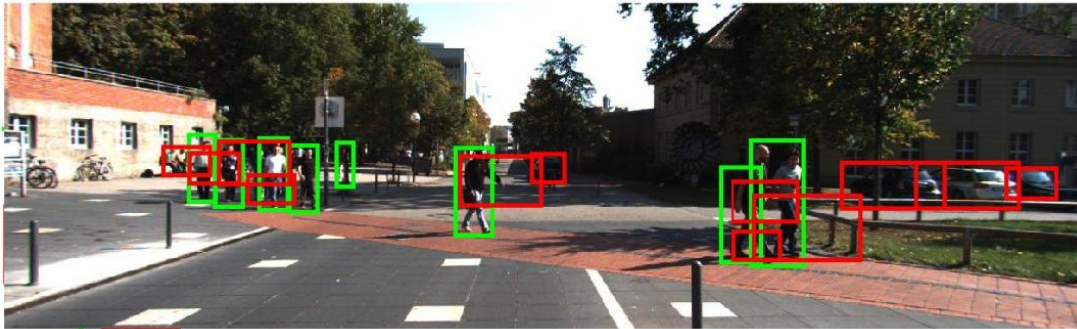
The state vector includes the centroid positions, velocities, and the width and height of the bounding boxes; the measurement is the noisy version of the target area, in the same fashion as in section 5.1.4. A dynamic model, an observation model and a birth covariance follow Eqs. (5.1), (5.2) and (5.3) respectively with the exception of setting $i \in \{1, 2\}$. We set $\sigma_{v_1} = 5 \text{ pixels}/s^2$ and $\sigma_{v_2} = 6 \text{ pixels}/s^2$ for target type 1 (pedestrians) and target type 2 (vehicles), respectively. We also set survival probabilities $p_{1,S} = p_{2,S} = 0.99$ for each target of both types, and the measurement standard deviations $\sigma_{r_{ii}}$ and $\sigma_{r_{ij}}$ ($i \in \{1, 2\}$ and $j \in \{1, 2\}$) are evaluated to 7 pixels.

This proposed visual tracking approach is analyzed using the KITTI tracking video sequence 16 [225]. A multi-target measurement $Z_{1,k}$ for pedestrians is obtained using a pedestrian detector. Similarly, a multi-target measurement $Z_{2,k}$ for vehicles is obtained using a vehicle detector. The sample frames of results of detections, two independent GM-PHD trackers and dual GM-PHD tracker are shown in Fig. 5.11 and Fig. 5.12. For instance, for the sample frame 23 in Fig. 5.12, many clutter responses from detections in Fig. 5.12a are removed by 2 independent GM-PHD trackers as shown in Fig. 5.12b.

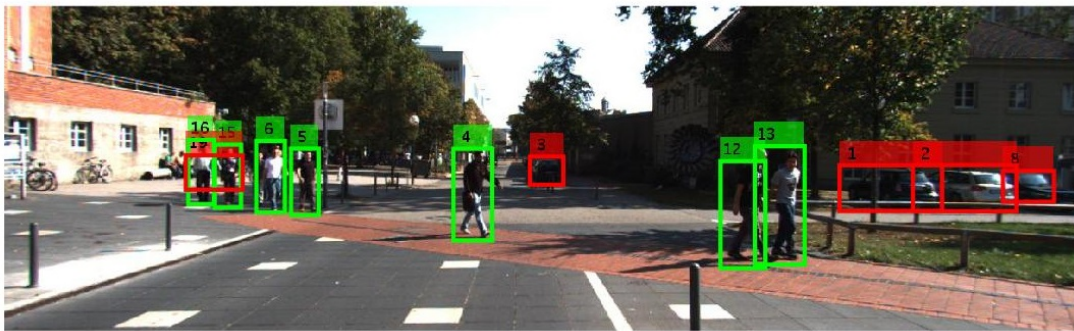
However, the pedestrian targets with labels 10, 30, 34 and 31 are confused by the vehicle detector and then tracked by standard GM-PHD trackers as shown in Fig. 5.12b. These are removed by our dual GM-PHD tracker as shown in Fig. 5.12c. Hence, our approach eliminates the wrong tracking of vehicles or pedestrians which are confused at detection.

The dual GM-PHD filter is evaluated quantitatively and compared with two independent GM-PHD filters and raw detection using the cardinality error, OSPA metric [215], time taken and discrimination rate in Table 5.2. We also show the cardinality and OSPA error plots as shown in Fig. 5.13a and Fig. 5.13b, respectively, in red for ground truth (cardinality), green for dual GM-PHD filter, blue for two independent GM-PHD filters and magenta for detections. As shown in Table 5.2, the overall average value of the OSPA error for the dual GM-PHD filter is 20.74 pixels compared to using two independent GM-PHD filters of 35.29 pixels and raw detection of 49.81 pixels. Our proposed approach reduces the OSPA error by a large margin over both using two independent GM-PHD filters and raw detection.

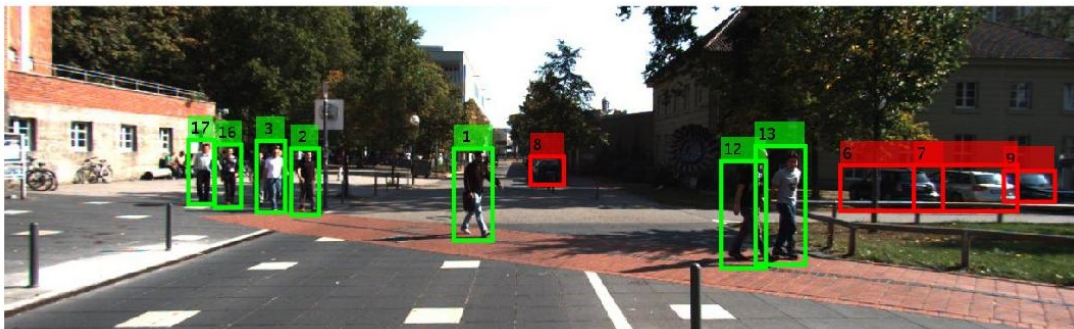
The time taken for the 209 video frames of the KITTI tracking sequence 16 is shown in Table 5.2 where the dual GM-PHD tracker takes 4.03 seconds per frame (both detection and tracking), two independent GM-PHD trackers take 1.61 seconds per frame and raw detection takes 1.25 seconds per frame on a i5 2.50 GHz core processor with 6 GB RAM laptop using MATLAB. Since we use many detection models for each actor (10 for pedestrians, 40 for vehicles), it takes more computational time than the scenario considered in section 5.1.4. As shown in Table 5.2, the dual GM-PHD tracker has only 0.32 (below 1 target) cardinality error and 1.81% discrimination rate error when compared to 3.82 cardinality error and 100% discrimination rate error using 2 independent GM-PHD trackers as well as 9.86 cardinality error and 100% discrimination rate error using raw detection. As can be seen from Fig. 5.11c and Fig. 5.12c, labels of some of the actors (cars - labels 6 and 9; pedestrians - labels 1, 2, 16, 17, etc) are consistent from frame to frame. Since we are using two frames to associate the targets, a new label is given to a target which disappears and then reappears as well as for a newly appearing target. For example, pedestrians labeled 12 and 13 in frame 13 are re-detected as one target in frame 23 and is given a label 26. The car labeled 7 in frame 13 is miss-detected, and then is re-detected in frame 23 and is given a new label, 25. Still, the labeling approach we use has a reasonable performance with a mean label switch error of only 1.07%, and it is obviously not part of the dual GM-PHD filter.



(a) Detections, frame 13



(b) Two independent GM-PHD trackers, frame 13

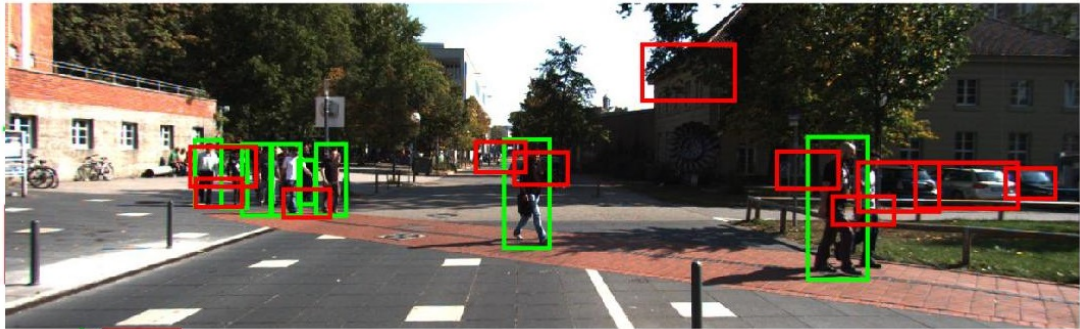


(c) Dual GM-PHD Tracker, frame 13

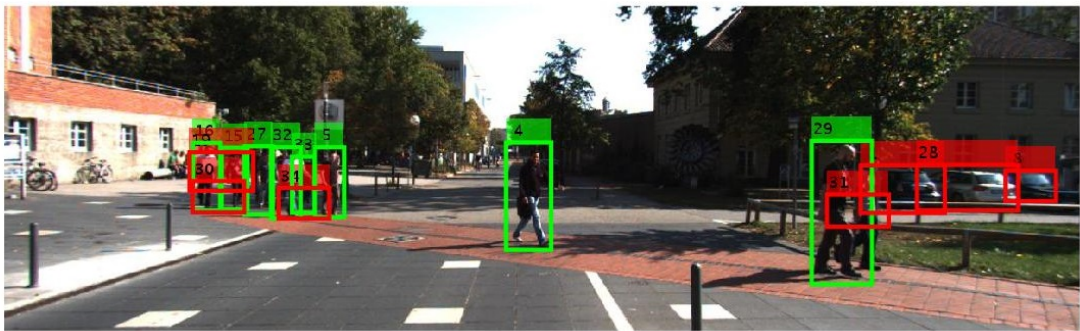
Figure 5.11: Results of detections, two independent GM-PHD trackers and dual GM-PHD tracker, respectively, for frame 13.

5.3 Evaluation and Comparison using the MOT Benchmarking Tool

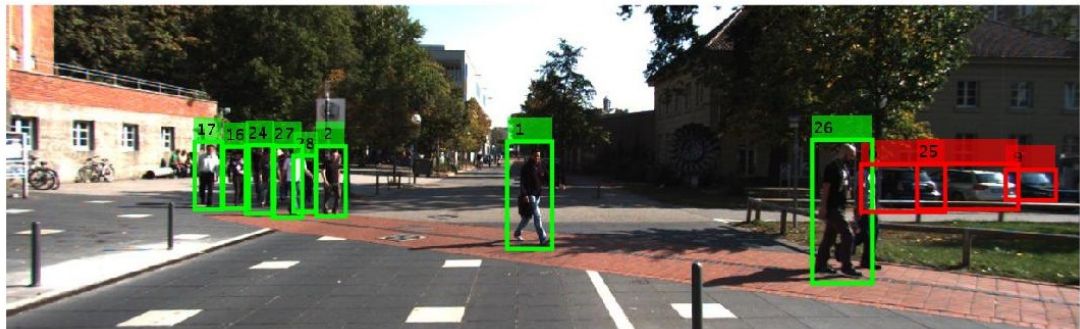
N-type GM-PHD tracker is developed for handling confusions among detectors of different target types. However, all of the state-of-the-art multi-tracking algorithms in MOT challenge have been developed for tracking a single type of multiple targets



(a) Detections, frame 23



(b) Two independent GM-PHD trackers, frame 23



(c) Dual GM-PHD Tracker, frame 23

Figure 5.12: Results of detections, two independent GM-PHD trackers and dual GM-PHD tracker, respectively, for frame 23.

($N = 1$). Thus, the N -type GM-PHD tracker cannot be directly evaluated using the MOT benchmarking tool to be compared to the state-of-the-art algorithms listed on the MOT challenge. To make a good understanding of the performance of our developed algorithm, we use the building block of the N -type GM-PHD tracker (the N -type GM-PHD tracker for $N = 1$, denoted by GM-PHD-N1T) to evaluate using the MOT benchmarking tool and compare to the state-of-the-art algorithms in the MOT challenge.

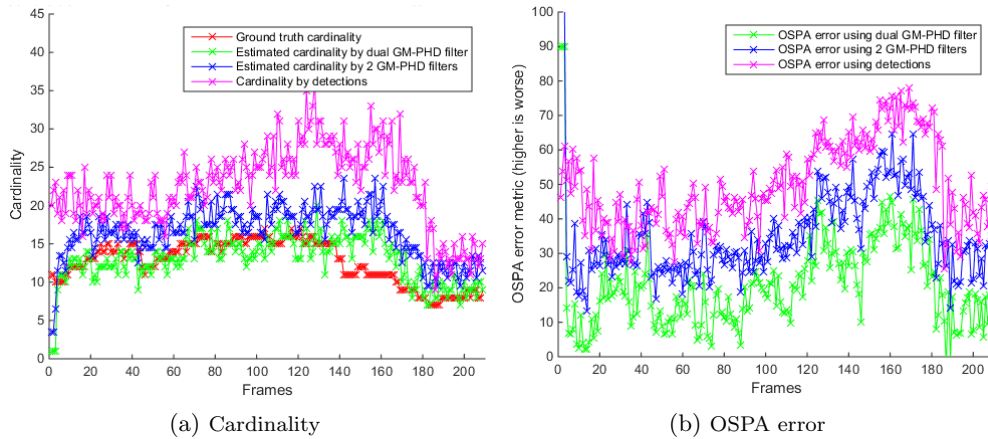


Figure 5.13: Cardinality and OSPA error: Ground truth (red for cardinality only), dual GM-PHD filter (green), two independent GM-PHD filters (blue), detections (magenta).

Method	cardinality error	OSPA error	time taken	discrimination rate
Detections	9.86	49.81 pixels	1.25 sec/frame	0%
2 GM-PHDs	3.82	35.29 pixels	1.61 sec/frame	0%
Dual GM-PHD	0.32	20.74 pixels	4.03 sec/frame	98.19%

Table 5.2: Frame-averaged cardinality and OSPA errors, time taken and discrimination rate at the extracted detection probabilities for dual GM-PHD filter, two independent GM-PHD filters and Detections.

Thus, we validate our proposed tracker (for $N = 1$ version) and compare it against state-of-the-art online and offline tracking methods (GM-PHD-MA [226], DP-NMS [227], SMOT [228], CEM [229] and JPDA-m [137]) on the MOT16 benchmark datasets [9]. We use the *public detections* provided by the MOT benchmark. We use the following evaluation measures: Multiple Object Tracking Accuracy (MOTA), Multiple Object Tracking Precision (MOTP) [213], Mostly Tracked targets (MT), Mostly Lost targets (ML) [214], Fragmented trajectories (Frag), False Positives (FP), False negatives (FN), Identity Switches (IDSw) and speed of the tracker (Hz). For detailed description of each metric, please refer to [9] and section 2.6.

Quantitative evaluation of our proposed method with other trackers is compared in Table 5.3. The Table shows that our algorithm outperforms both online and offline trackers listed in the table in terms of MOTP. In terms of MOTA, our tracker outperforms the online tracker(s) and the offline trackers such as GM-PHD-MA, SMOT and JPDA-m. The number of MT percentage is overall higher than many of the online and offline trackers except one offline tracker (i.e. second to CEM). The number of ML, FP and FN percentage are also lower than many of the online and offline trackers. The

higher number of IDSw and Frag compared to the other online tracker and some of the offline trackers is due to the fact that our tracker relies only on the position and size of the bounding box of the detections; we are not using any appearance models to discriminate nearby targets. Spawning targets are also not modelled in our tracker, therefore, identity switches are more likely to occur in such crowded scenes. Our tracker runs about 9.9 frames per second (fps). The computational costs arise from experiments on a i7 2.30 GHz core processor with 8 GB RAM using Matlab (not well optimized).

The most important to note here is that the comparison of our algorithm to the GM-PHD-MA [226]. These both trackers use a GM-PHD filter but with different approaches for labelling of targets from frame to frame. While our tracker uses the Hungarian algorithm for labelling of targets by postprocessing the output of the filter, the GM-PHD-MA uses the approach in [177] by also including appearance features for the labelling of targets. Our tracker outperforms the GM-PHD-MA tracker in many of the evaluation metrics except IDSw and Frag. The GM-PHD-MA performs better in terms of IDSw and Frag since it includes appearance features in addition to the position and size of the bounding box of the detections; appearance models are not used in our tracker. Thus, the building block of the N-type GM-PHD tracker performs reasonably when evaluated and compared to the MOT challenge.

Examples of tracking results of all MOT16 test sequences except MOT16-07 are shown in Figure 5.14; from left to right: MOT16-01, MOT16-03 (top row), MOT16-06, MOT16-08 (middle row), and MOT16-12, MOT16-14 (bottom row). Three frames from MOT16-07 are shown in Figure 5.15. In all figures, the bounding boxes represent the tracking results with their color-coded identities except MOT16-07 for which we use labels of the targets as small numbers over their bounding boxes. The MOT16-07 shown in Figure 5.15 contains 54 tracks recorded by a moving camera in a sequence of 500 frames. Tracking in this sequence is a very challenging task, not only because the density of pedestrians is quite high, but also because significant camera motion makes the person trajectories to be both rough and discontinuous. Our tracker reasonably performs even on this sequence though some identity switches occur due to significant camera motion, detection failures and lack of appearance model in our approach.

Tracker	Tracking Mode	MOTA \uparrow	MOTP \uparrow	MT (%) \uparrow	ML (%) \downarrow	FP \downarrow	FN \downarrow	IDS $w\downarrow$	Frag \downarrow	H $z\uparrow$
CEM [229]	offline	33.2	75.8	7.8	<u>54.4</u>	6,837	<u>114,322</u>	642	<u>731</u>	0.3
DP-NMS [227]	offline	<u>32.2</u>	<u>76.4</u>	5.4	62.1	1,123	<u>121,579</u>	972	944	212.6
SMOT [228]	offline	29.7	75.2	5.3	47.7	17,426	107,552	3,108	4,483	0.2
JPDF-m [137]	offline	26.2	76.3	4.1	67.5	3,689	130,549	365	638	22.2
GM-PHD-MA [226]	online	30.5	75.4	4.6	59.7	<u>5,169</u>	120,970	<u>539</u>	<u>731</u>	13.6
GM-PHD-N1T (ours)	online	31.6	76.6	<u>5.5</u>	55.2	4,767	115,645	4,348	3,986	9.9

Table 5.3: Tracking performance of representative trackers developed using both online and offline methods. All trackers are evaluated on the test dataset of the MOT16 [9] benchmark using public detections. The first and second highest values are highlighted by bold and underline, respectively.



Figure 5.14: Sample results on several sequences of MOT16 datasets, bounding boxes represents the tracking results with their color-coded identities. From left to right: MOT16-01, MOT16-03 (top row), MOT16-06, MOT16-08 (middle row), and MOT16-12, MOT16-14 (bottom row).

5.4 Summary

In this chapter, our proposed N-type GM-PHD filter is applied to real video sequences considering two scenarios: tracking different football teams and a referee by treating them as three different types of targets, and tracking pedestrians and vehicles by treating them as two different types of targets. When applying to real video sequences, there are many control parameters that exist in this proposed algorithm which are very important to select carefully. Many of these parameters are related to the experimental evaluation of the detection processes, particularly, true and confusion detection probabilities, clutter rate, and measurement standard deviation. True and confusion detection probabilities for each target type are extracted from ROC curves of each detector at a specific background clutter rate (false positive per image). These param-

eters can be fine-tuned depending on the aspect ratios of the the target types under consideration. Similarly, the measurement standard deviations can be estimated from the distribution of distance errors of the centroids from ground truth in the evaluation of the detection process. However, the other parameters such as survival probabilities and standard deviations of the process noise for each target type are set from simulation and previous experience. Thus, using these parameters, we apply the N-type GM-PHD filter for visual tracking on real video sequences achieving improved performance over raw detection and independent standard GM-PHD filters. We also evaluate the building block of the N-type GM-PHD tracker (for $N = 1$) using the MOT benchmarking tool to compare to the state-of-the-art algorithms in the MOT challenge and find out that our algorithm performs against other state-of-the-art trackers.

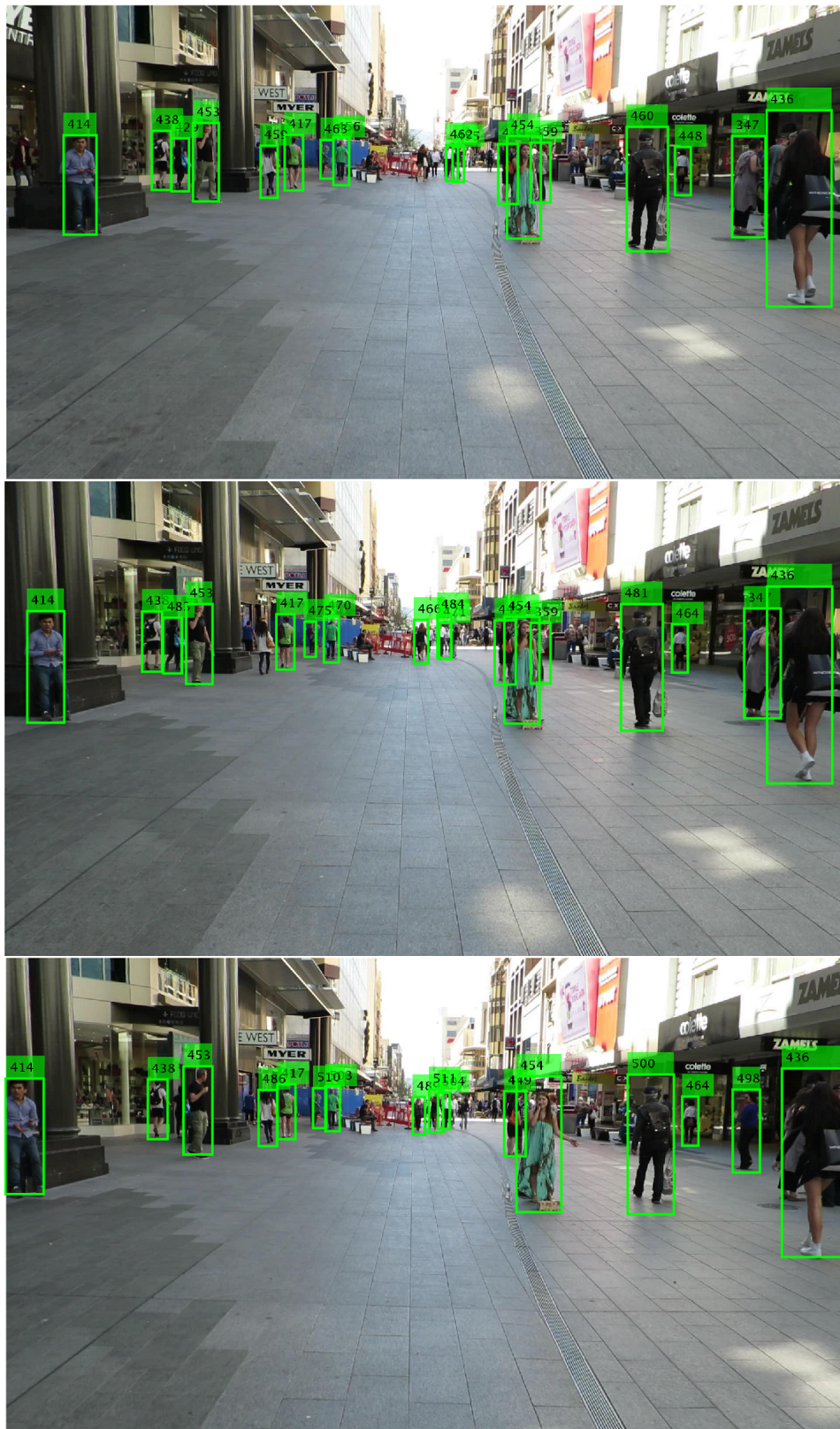


Figure 5.15: Sample results on the sequence MOT16-07, bounding boxes represents the tracking results with their label for their identities, for frames 368, 380 and 396 from top to bottom.

Chapter 6

Conclusions and Future Work

In this chapter, the thesis is summarized with conclusions of the proposed methods with an outlook on future work.

6.1 Conclusions

In this thesis, novel tracking algorithms are proposed, both for model-free single target tracking and multi-target tracking with different target types. Accordingly, the central contributions of this thesis can be briefly stated by categorizing into three as follows.

First, we developed a novel long-term visual tracking algorithm by learning discriminative correlation filters and an incremental SVM classifier that can be applied for tracking of a target in sparse as well as in crowded environments. We learn two different discriminative correlation filters: translation and scale correlation filters. For the translation correlation filter, we combined multi-layer CNN features trained on a large amount of object recognition data set (ImageNet) and traditional (HOG and color-naming) features in proper proportion. For the CNN part, we combined the advantages of both lower and higher convolutional layers to capture spatial details for precise localization and semantic information for handling appearance variations, respectively. We also include a re-detection module using HOG, LUV color and normalized gradient magnitude features for re-initializing the tracker in case of tracking failures due to long-term occlusions by training an incremental SVM from the most confident frames. The re-detection module, when activated (the correlation response of the object is below some pre-defined threshold), generates high score detection pro-

posals which are temporally filtered using a GM-PHD filter for removing clutter. The Gaussian component with the maximum weight is selected as a state estimate which re-fines the object location when a re-detection module is activated. For the scale correlation filter, we use HOG features to construct a target pyramid around the estimated or re-detected position for estimating the scale of the target. Extensive experiments on both OOTB and PETS 2009 data sets show that our proposed algorithm significantly outperforms state-of-the-art trackers by 3.48% in distance precision and 7.77% in overlap success on sparse (OOTB) data sets, and by 36.87% in distance precision and 34.92% in overlap success on dense (PETS 2009) data sets. We conclude learning correlation filters using an appropriate combination of CNN and traditional features as well as including a re-detection module using incremental SVM and GM-PHD filter can give better results on both sparse and dense environments.

Second, we developed an extension of the PHD filter in the RFS framework to account for many different types of targets with separate observations of the same scene, allowing for different probabilities of detection, scene clutter and possible confusions between targets of different types at the detection stage. This extends the standard PHD filter to a N -type PHD filter i.e. in principle the methodology can be applied to N types of targets where N is a variable, with the caveat that the number of possible confusions rises as $N(N - 1)$. Then, under the Gaussianity and linearity assumptions, we extend the Gaussian mixture (GM) implementation of the standard PHD filter for the proposed N -type PHD filter, N -type GM-PHD filter. We evaluate the quad GM-PHD filter and compare to four independent GM-PHD filters, indicating that our approach shows better performance determined using cardinality and OSPA metric by efficiently discriminating the target types.

Third, our proposed N -type GM-PHD filter is tested and evaluated using real video sequences by integrating object detectors' information such as the probabilities of detections for each target type and confusion detection probabilities among target types at a specific clutter rate into this filter by considering two scenarios. In the first case, a tri-GM-PHD filter is applied to sport analysis applications, particularly, tracking different football teams and a referee by treating them as three different types of targets. In the second case, we use a dual GM-PHD filter for tracking pedestrians and vehicles by treating them as two different types of targets which is important for building intelligent static/mobile vehicles for detecting, tracking and discriminating different target types in the same scene handling their detectors' confusions. For each scenario, we also apply Munkres's variant of the Hungarian assignment algorithm as data association on the filtered results of the filter as a post-process to label each target from frame to frame. The key finding of this work is that by considering and modeling confusions

between the different types of targets we can improve the target discrimination rate, demonstrated by quantitative measurement of cardinality and OSPA error. Application of the Hungarian labeling method shows reasonable data association so that we are able to track individual targets over the sequence. Although the process is applied here to 3 and 2 types of targets on real video sequences, in principle the methodology can be applied to visually track N types of targets where N is any variable integer in which the number of possible confusions may rise as $N(N - 1)$. We also observe that other assumptions about background clutter, target location and birth follow the same random models as the standard PHD filter. Hence we assume that our background clutter, and the detection and confusion probabilities are uniform across the image field, which is not unreasonable in the football data, but is less likely to be true when identifying pedestrians in an urban environment, where particular street furniture may generate repeated false alarms. As the detections are represented as points (centroids of bounding detection boxes), the filtering process does not explicitly consider scale, and as the boxes and humans/vehicles within the boxes have finite extent, this makes occlusions possible such that targets may disappear for several frames. Notwithstanding these imperfections, the work we have done has shown that the N-type GM-PHD filter has the potential both to track targets in video data, and to better address multiple target confusions than the standard method. The evaluations and comparisons of these trackers on these two scenarios to both raw detections and independent GM-PHD filters using cardinality, OSPA metric and discrimination rate show the improved performance of our strategy on real video sequences.

6.2 Future Work

In this thesis, three aspects of future work are identified as the possible extensions of the proposed methods, and are presented as follows.

1. Extending model-free single unknown target tracking to model-free multi-target tracking using discriminative correlation filters and a hybrid of multi-layer CNN and traditional hand-crafted (e.g. HOG + color naming) features. Model-free multi-target tracking is presented in [107], however, the potential of correlation filters and CNN features are not explored so far in the model-free multi-target tracking problems.
2. Parallelizing of a N-type PHD filter (either Gaussian Mixture or SMC implementation) using multi-core CPU, GPU, and/or FPGA platforms. This is important

since the N-type PHD filter is slightly more computationally expensive than the standard PHD filter to apply for real-time applications.

3. Integrating a parallelized N-type PHD filter implementation into either static or moving (automotive) embedded platforms (e.g. cars or robots) for tracking multiple actors (pedestrian, vehicles, bicycles, etc) by fusing information from multiple sensors (video camera, radar (RADio Detection And Ranging), lidar (LIght Detection And Ranging), etc). Similarly, it might also be crucial to integrate the parallelized N-type PHD filter for real-time sports analysis such as real-time tracking and performance (e.g. using trajectory) analysis of football teams and a referee, or basketball or hockey teams while discriminating the target types.

Chapter 7

APPENDIX A

PHD filter for multi-target tracking of a single target type was proposed by Mahler in [170] where its derivation is given. We have derived novel extensions for the updated PHDs of a N-type PHD filter from PGFLs of each target type handling confusions among target types starting from the standard proved predicted PHDs in Chapter 4. In this case we assume the standard predicted PHD can be applied to each target type as influence of the measurement confusions among the target types we are solving occurs at the update stage of the PHD recursion. To make this thesis self-sufficient, we derive the predicted PHDs for each target type $i \in \{1, \dots, N\}$ starting from the PGFLs of the predicted processes. We use the product rule and chain rule for functional derivatives to find PHD prediction equation for target type $i \in \{1, \dots, N\}$ (differentiate and set $h = 1$) as follows:

$$G_{i,k|k-1}(h) = G_{i,\gamma}(h)G_{i,k-1}(G_{i,S}(h|\cdot)), \quad (7.1)$$

where (taking $\varphi_x = \delta_x$)

$$\left. \frac{\delta}{\delta \varphi_x} G_{i,\gamma}(h) \right|_{h=1} = \gamma_{i,k}(x). \quad (7.2)$$

$$\left. \frac{\delta}{\delta \varphi_x} G_{i,k-1}(h) \right|_{h=1} = \mathcal{D}_{i,k-1}(x). \quad (7.3)$$

and

$$G_{i,S}(h|x) = 1 - p_{i,S,k}(x) + p_{i,S,k}(x) \int h(y) y_{i,k|k-1}(y|x) dy. \quad (7.4)$$

the aim is to compute the PHD of the predicted process $D_{i,k|k-1}(x)$ for target type $i \in \{1, \dots, N\}$.

Deriving Eq. 7.1 in $\varphi_x = \delta_x$ and using the product rules yields:

$$\frac{\delta}{\delta\varphi_x} G_{i,k|k-1}(h) \Big|_{h=1} = \underbrace{\frac{\delta}{\delta\varphi_x} G_{i,\gamma}(h) \Big|_{h=1}}_{=\gamma_{i,k}(x)} \underbrace{G_{i,k-1}(G_{i,S}(h|\cdot)) \Big|_{h=1}}_{=1} + \underbrace{G_{i,\gamma}(h) \Big|_{h=1}}_{=1} \underbrace{\frac{\delta}{\delta\varphi_x} G_{i,k-1}(G_{i,S}(h|\cdot)) \Big|_{h=1}}_{=A} \quad (7.5)$$

Now, using the chain rule A reads:

$$A = \frac{\delta G_{i,k-1}}{\delta \left(\frac{\delta G_{i,S}(h|\cdot)}{\delta\varphi_x} \right)} (G_{i,S}(h|\cdot)) \Big|_{h=1} \quad (7.6)$$

where $\frac{\delta G_{i,S}(h|\cdot)}{\delta\varphi_x}$ is found by deriving Eq. 7.4 and using the linearity of $h \rightarrow \int h(y) y_{i,k|k-1}(y|\cdot) dy$ as follows:

$$\frac{\delta G_{i,S}(h|\cdot)}{\delta\varphi_x} = \underbrace{\frac{\delta}{\delta\varphi_x} (1 - p_{i,S,k}(\cdot))}_{=0} + p_{i,S,k}(\cdot) \underbrace{\frac{\delta}{\delta\varphi_x} \int h(y) y_{i,k|k-1}(y|\cdot) dy}_{=y_{i,k|k-1}(x|\cdot)} \quad (7.7)$$

Let us write $g(\cdot) = p_{i,S,k}(\cdot) y_{i,k|k-1}(x|\cdot)$ for simplicity's sake. Now all we need to do is to compute $\frac{\delta G_{i,k-1}}{\delta g(\cdot)} (G_{i,S}(h|\cdot)) \Big|_{h=1}$. Using the Janossy densities we can write:

$$G_{i,k-1}(h) = \sum_{n=0}^{\infty} \frac{1}{n!} \int_{\mathcal{X}^n} \left(\prod_{l=1}^n h(x_l) \right) j_{i,k-1}^{(n)}(x_1, \dots, x_n) dx_1 \dots dx_n. \quad (7.8)$$

Deriving $G_{i,k-1}$ in $g(\cdot)$ gives:

$$\begin{aligned} \frac{\delta G_{i,k-1}}{\delta g(\cdot)} (h) \Big|_{h=1} &= \sum_{n=1}^{\infty} \frac{1}{n!} \int_{\mathcal{X}^n} \frac{\delta}{\delta g(\cdot)} \left(\prod_{l=1}^n h(x_l) \right) \Big|_{h=1} j_{i,k-1}^{(n)}(x_1, \dots, x_n) dx_1 \dots dx_n, \\ &= \sum_{n=1}^{\infty} \frac{1}{n!} \int_{\mathcal{X}^n} \sum_{l=1}^n \left(\frac{\delta}{\delta g(\cdot)} h(x_l) \prod_{j \neq l} h(x_j) \right) \Big|_{h=1} j_{i,k-1}^{(n)}(x_1, \dots, x_n) dx_1 \dots dx_n. \end{aligned} \quad (7.9)$$

Using the definition of functional derivatives with $F[h] = h(x_l)$ gives:

$$\begin{aligned} \left. \frac{\delta G_{i,k-1}}{\delta g(\cdot)}(h) \right|_{h=1} &= \sum_{n=1}^{\infty} \frac{1}{n!} \int_{\mathcal{X}^n} \left(\sum_{l=1}^n g(x_l) \right) j_{i,k-1}^{(n)}(x_1, \dots, x_n) dx_1 \dots dx_n, \\ &= \sum_{n=1}^{\infty} \frac{1}{n!} \sum_{l=1}^n \int_{\mathcal{X}^n} g(x_l) j_{i,k-1}^{(n)}(x_1, \dots, x_n) dx_1 \dots dx_n. \end{aligned} \quad (7.10)$$

That is, since the Janossy densities are symmetric functions:

$$\begin{aligned} \left. \frac{\delta G_{i,k-1}}{\delta g(\cdot)}(h) \right|_{h=1} &= \sum_{n=1}^{\infty} \frac{1}{(n-1)!} \int_{\mathcal{X}^n} g(x_1) j_{i,k-1}^{(n)}(x_1, \dots, x_n) dx_1 \dots dx_n, \\ &= \int_{\mathcal{X}} g(x_1) \left(\sum_{n=1}^{\infty} \frac{1}{(n-1)!} \int_{\mathcal{X}^{n-1}} j_{i,k-1}^{(n)}(x_1, \dots, x_n) dx_2 \dots dx_n \right) dx_1, \\ &= \int_{\mathcal{X}} g(x) \left(\sum_{n=0}^{\infty} \frac{1}{n!} \int_{\mathcal{X}^n} j_{i,k-1}^{(n+1)}(x, x_1, \dots, x_n) dx_1 \dots dx_n \right) dx. \end{aligned} \quad (7.11)$$

Or, using the characterization of the PHD $\mathcal{D}_{i,k-1}(x)$ as the first-moment density of the process:

$$\left. \frac{\delta G_{i,k-1}}{\delta g(\cdot)}(h) \right|_{h=1} = \int_{\mathcal{X}} g(x) \mathcal{D}_{i,k-1}(x) dx. \quad (7.12)$$

Combining Eq. 7.5, Eq. 7.6, Eq. 7.7 and Eq. 7.12 yields the result:

$$\mathcal{D}_{i,k|k-1}(x) = \left. \frac{\delta G_{i,k|k-1}}{\delta \varphi_x}(h) \right|_{h=1} = \gamma_{i,k}(x) + \int_{\mathcal{X}} p_{i,S,k}(\zeta) y_{i,k|k-1}(x|\zeta) \mathcal{D}_{i,k-1}(\zeta) d\zeta. \quad (7.13)$$

Bibliography

- [1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., 2012, pp. 1097–1105. [x](#), [13](#), [14](#), [117](#)
- [2] E. Maggio and A. Cavallaro, *Video tracking: theory and practice*. Wiley, 2011. [x](#), [8](#), [11](#), [19](#), [20](#), [21](#), [31](#)
- [3] D. Clark and B.-N. Vo, “The random set filtering website,” September 2016. [Online]. Available: <http://randomsets.eps.hw.ac.uk/tutorial.html> [x](#), [39](#)
- [4] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *ICLR*, 2015. [xi](#), [13](#), [14](#), [23](#), [49](#), [62](#), [63](#)
- [5] C. Ma, J. B. Huang, X. Yang, and M. H. Yang, “Hierarchical convolutional features for visual tracking,” in *IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 3074–3082. [xi](#), [xii](#), [24](#), [64](#), [66](#), [68](#), [69](#), [70](#), [72](#), [73](#)
- [6] J. Zhang, S. Ma, and S. Sclaroff, “MEEM: robust tracking via multiple experts using entropy minimization,” in *Proc. of the European Conference on Computer Vision (ECCV)*, 2014. [xi](#), [xii](#), [23](#), [64](#), [66](#), [68](#), [69](#), [70](#), [72](#), [73](#)
- [7] C. Ma, X. Yang, C. Zhang, and M. H. Yang, “Long-term correlation tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 5388–5396. [xi](#), [xii](#), [21](#), [22](#), [24](#), [54](#), [64](#), [66](#), [68](#), [69](#), [70](#), [72](#), [73](#)
- [8] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filters,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2015. [xi](#), [xii](#), [23](#), [49](#), [53](#), [62](#), [64](#), [66](#), [68](#), [70](#), [72](#), [73](#)
- [9] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, “MOT16: A benchmark for multi-object tracking,” *arXiv:1603.00831 [cs]*, Mar. 2016, arXiv:

- 1603.00831. [Online]. Available: <http://arxiv.org/abs/1603.00831> xv, 46, 124, 126
- [10] A. Yilmaz, O. Javed, and M. Shah, “Object tracking: A survey,” 2006. 2, 7, 8, 19
- [11] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, “Visual tracking: An experimental survey,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1442–1468, July 2014. 2, 24
- [12] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, “Recent advances and trends in visual tracking: A review,” *Neurocomputing*, vol. 74, no. 18, pp. 3823 – 3831, 2011. 2, 7, 10, 19, 20, 21, 22
- [13] Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012. 2, 22, 23, 24, 64
- [14] H. Idrees, N. Warner, and M. Shah, “Tracking in dense crowds using prominence and neighborhood motion concurrence,” *Image and Vision Computing*, vol. 32, no. 1, pp. 14 – 26, 2014. 2, 24
- [15] P. Matzka, A. Wallace, and Y. Petillot, “Efficient resource allocation for automotive attentive vision systems,” *IEEE Trans. on Intelligent Transportation Systems*, vol. 13, no. 2, pp. 859–872, 2012. 3
- [16] P. Dollar, R. Appel, P. Perona, and S. Belongie, “Fast feature pyramids for object detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 99, p. 14, 2014. 3, 12, 15, 18, 102, 116, 118
- [17] J. Liu and P. Carr, “Detecting and tracking sports players with random forests and context-conditioned motion models,” in *Computer Vision in Sports*. Springer, 2014, pp. 113–132. 3
- [18] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *Computer Vision and Pattern Recognition. IEEE Conference on*, June 2009, pp. 248–255. 4, 15, 62, 117
- [19] D. Serby, E. K. Meier, and L. V. Gool, “Probabilistic object tracking using multiple features,” in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 2, Aug 2004, pp. 184–187 Vol.2. 8
- [20] S. T. Birchfield and S. Rangarajan, “Spatiograms versus histograms for region-based tracking,” in *Proceedings of the IEEE Computer Society Conference on*

- Computer Vision and Pattern Recognition (CVPR) - Volume 2 - Volume 02*, 2005, pp. 1158–1163. [9](#)
- [21] G. J. Edwards, C. J. Taylor, and T. F. Cootes, “Interpreting face images using active appearance models.” 1998, pp. 300–305. [9](#)
- [22] N. Birkbeck and M. Jagersand, “Visual tracking using active appearance models,” in *Computer and Robot Vision, 2004. Proceedings. First Canadian Conference on*, 2004, pp. 2–9. [9](#)
- [23] T. Cootes, G. Edwards, and C. Taylor, “Active appearance models,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 23, no. 6, pp. 681–685, Jun 2001. [9](#)
- [24] M. J. Black and A. D. Jepson, “Eigentracking: Robust matching and tracking of articulated objects using a view-based representation,” *Int. J. Comput. Vision*, vol. 26, no. 1, pp. 63–84, Jan. 1998. [9](#)
- [25] S. Avidan, “Support vector tracking,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 8, pp. 1064–1072, Aug 2004. [9](#), [20](#), [22](#)
- [26] J. Kwon and K. M. Lee, “Tracking by sampling and integrating multiple trackers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1428–1441, July 2014. [9](#)
- [27] P. Tissainayagam and D. Suter, “Visual tracking and motion determination using the IMM algorithm,” in *14th International Conference on Pattern Recognition - ICPR’98*, vol. 1, 1998, pp. 289–291. [9](#)
- [28] S. Zhou, R. Chellappa, and B. Moghaddam, “Visual tracking and recognition using appearance-adaptive models in particle filters,” *Image Processing, IEEE Transactions on*, vol. 13, no. 11, pp. 1491–1506, Nov 2004. [9](#)
- [29] J. M. Odobez and D. Gatica-Perez, “Embedding motion in model-based stochastic tracking,” in *Pattern Recognition, Proceedings of the 17th International Conference on*, vol. 2, Aug 2004, pp. 815–818. [9](#)
- [30] D. Comaniciu, V. Ramesh, and P. Meer, “Kernel-based object tracking,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2003. [9](#), [20](#), [21](#)
- [31] J. Shi and C. Tomasi, “Good features to track,” in *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, Jun 1994, pp. 593–600. [10](#), [16](#), [20](#)
- [32] A. Ford and A. Roberts, “Colour space conversions,” Westminster University, London, Tech. Rep., August 1998. [11](#)

- [33] T. Gevers, J. van de Weijer, and H. Stokman, *Color image processing: methods and applications: color feature detection: an overview*. CRC press, 2006, ch. 9, pp. 203–226. [11](#)
- [34] J. van de Weijer, C. Schmid, J. Verbeek, and D. Larlus, “Learning color names for real-world applications,” *Trans. Img. Proc.*, vol. 18, no. 7, pp. 1512–1523, Jul. 2009. [11](#), [49](#)
- [35] G. J. Burghouts and J. M. Geusebroek, “Performance evaluation of local colour invariants,” *Computer Vision and Image Understanding*, vol. 113, pp. 48–62, 2009. [11](#)
- [36] S. Bhardwaj and A. Mittal, “A survey on various edge detector techniques,” *Procedia Technology*, vol. 4, pp. 220 – 226, 2012. [11](#)
- [37] B. Manjunath and W. Ma, “Texture features for browsing and retrieval of image data,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 18, no. 8, pp. 837–842, Aug 1996. [12](#)
- [38] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 24, no. 7, pp. 971–987, Jul 2002. [12](#)
- [39] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [12](#), [16](#), [17](#)
- [40] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, “Speeded-up robust features (SURF),” *Comput. Vis. Image Underst.*, vol. 110, no. 3, pp. 346–359, Jun. 2008. [12](#), [16](#)
- [41] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 886–893, 2005. [12](#), [17](#)
- [42] H. Wang, M. M. Ullah, A. Klser, I. Laptev, and C. Schmid, “Evaluation of local spatio-temporal features for action recognition,” in *University of Central Florida, U.S.A*, 2009. [12](#)
- [43] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, “Learning realistic human actions from movies,” in *Computer Vision and Pattern Recognition. IEEE Conference on*, June 2008, pp. 1–8. [12](#)
- [44] X. Wang, T. Han, and S. Yan, “An HOG-LBP human detector with partial occlusion handling,” in *Computer Vision, IEEE 12th International Conference on*, Sept 2009, pp. 32–39. [12](#), [17](#)

- [45] B. Bing Wang, Z. Xin Chen, J. Wang, and L. Zhang, "Pedestrian detection based on the combination of HOG and background subtraction method," in *Proc. International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE)*, 2011, pp. 527–531. [12](#)
- [46] W. Ma, B. Ma, and X. Zhan, "Kalman particle PHD filter for multi-target visual tracking," in *Proceedings of the Second Sino-foreign-interchange Conference on Intelligent Science and Intelligent Data Engineering*, ser. IScIDE'11, 2012, pp. 341–348. [12](#), [42](#)
- [47] Y. B. Ian Goodfellow and A. Courville, "Deep learning," 2016, book in preparation for MIT Press. [Online]. Available: <http://www.deeplearningbook.org> [13](#)
- [48] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 5 2015. [13](#)
- [49] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014. [13](#), [117](#)
- [50] A. Vedaldi and K. Lenc, "MatConvNet – convolutional neural networks for matlab," in *Proceedings of the 25th annual ACM international conference on Multimedia*, 2015. [13](#), [64](#)
- [51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015. [13](#), [14](#)
- [52] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Computer Vision and Pattern Recognition*, 2014. [13](#), [15](#), [23](#), [117](#)
- [53] P. Sermanet, K. Kavukcuoglu, S. Chintala, and Y. LeCun, "Pedestrian detection with unsupervised multi-stage feature learning," in *CVPR*, 2013, pp. 3626–3633. [13](#)
- [54] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," *CoRR*, vol. abs/1604.04693, 2016. [13](#), [18](#)
- [55] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, Jan 2013. [13](#)

- [56] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, “Decaf: A deep convolutional activation feature for generic visual recognition,” *CoRR*, vol. abs/1310.1531, 2013. [15](#), [117](#)
- [57] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (VOC) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun. 2010. [15](#), [117](#)
- [58] E. Ohn-Bar and M. M. Trivedi, “Learning to detect vehicles by clustering appearance patterns,” *IEEE Transactions on Intelligent Transportation Systems*, 2015. [15](#), [18](#), [116](#), [117](#), [118](#)
- [59] E. Seemann, B. Leibe, K. Mikolajczyk, and B. Schiele, “An evaluation of local shape-based features for pedestrian detection,” in *In Proc. BMVC*, 2005. [16](#)
- [60] T. Tuytelaars and K. Mikolajczyk, “K.: Local invariant feature detectors: A survey,” *Fnt Comp. Graphics and Vision*, pp. 177–280, 2008. [16](#)
- [61] T. Bouwmans, F. E. Baf, and B. Vachon, “Background modeling using mixture of gaussians for foreground detection: A survey,” in *Recent Patents on Computer Science*, 2008, pp. 219–237. [16](#)
- [62] S. Tripathi, K. Kumar, B. K. Singh, and R. P. Singh, “Image segmentation: A review,” *International Journal of Computer Science and Management Research*, vol. 1, November 2012. [16](#)
- [63] K. P. Murphy, *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012. [16](#), [25](#), [52](#)
- [64] D. Munoz, J. A. D. Bagnell, N. Vandapel, and M. Hebert, “Contextual classification with functional max-margin markov networks,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009. [16](#)
- [65] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” in *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, ser. COLT’ 98. ACM, 1998, pp. 92–100. [17](#)
- [66] A. Levin, P. Viola, and Y. Freund, “Unsupervised improvement of visual detectors using cotraining,” in *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, Oct 2003, pp. 626–633 vol.1. [17](#)
- [67] P. Dollar, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: An evaluation of the state of the art,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 4, pp. 743–761, April 2012. [17](#)

- [68] R. Benenson, M. Omran, J. Hosang, and B. Schiele, *Ten Years of Pedestrian Detection, What Have We Learned?* Cham: Springer International Publishing, 2015, pp. 613–627. [17](#)
- [69] P. Viola and M. J. Jones, “Robust real-time face detection,” *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004. [17](#)
- [70] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part based models,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010. [17](#), [18](#), [49](#), [50](#), [54](#), [60](#), [62](#)
- [71] N. Haering, P. L. Venetianer, and A. J. Lipton, “The evolution of video surveillance: an overview.” *Mach. Vis. Appl.*, vol. 19, no. 5-6, pp. 279–290, 2008. [19](#)
- [72] H. W. Kuhn and B. Yaw, “The hungarian method for the assignment problem,” *Naval Res. Logist. Quart.*, pp. 83–97, 1955. [19](#), [33](#)
- [73] F. Bourgeois and J.-C. Lassalle, “An extension of the munkres algorithm for the assignment problem to rectangular matrices,” *Commun. ACM*, vol. 14, no. 12, Dec. 1971. [19](#), [33](#), [43](#), [105](#), [120](#)
- [74] K. Shafique and M. Shah, “A non-iterative greedy algorithm for multi-frame point correspondence,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2003, pp. 51–65. [19](#), [33](#), [35](#), [105](#)
- [75] L. Zhang, Y. Li, and R. Nevatia, “Global data association for multi-object tracking using network flows,” in *Computer Vision and Pattern Recognition. IEEE Conference on*, June 2008, pp. 1–8. [19](#), [33](#), [35](#), [37](#), [105](#)
- [76] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, “Globally-optimal greedy algorithms for tracking a variable number of objects,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, June 2011, pp. 1201–1208. [19](#), [33](#), [35](#), [37](#), [105](#)
- [77] P. Lenz, A. Geiger, and R. Urtasun, “FollowMe: Efficient online min-cost flow tracking with bounded memory and computation,” in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015. [19](#), [33](#), [35](#), [37](#), [105](#)
- [78] R. T. Collins, “Multitarget data association with higher-order motion models,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, June 2012, pp. 1744–1751. [19](#), [33](#), [37](#), [105](#)
- [79] G. Welch and G. Bishop, “An introduction to the kalman filter,” 2006. [19](#), [26](#), [27](#), [41](#)

- [80] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking,” *Signal Processing, IEEE Transactions on*, vol. 50, no. 2, pp. 174–188, 2002. [19](#), [26](#), [28](#), [29](#)
- [81] Y. Bar-Shalom and T. E. Fortmann, *Tracking and data association*. Academic Press Boston, 1988, vol. 179. [19](#), [25](#), [31](#), [32](#), [37](#), [81](#)
- [82] C. Rasmussen and G. D. Hager, “Probabilistic data association methods for tracking complex visual objects,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, pp. 560–576, 2001. [19](#), [37](#)
- [83] T.-J. Cham and J. M. Rehg, “A multiple hypothesis approach to figure tracking.” in *CVPR*. IEEE Computer Society, 1999, pp. 2239–2245. [19](#), [37](#)
- [84] B. T. Vo, “Random finite sets in multi-object filtering,” Thesis (Ph.D), School of Electrical, Electronic and Computer Engineering, University of Western Australia, 2008. [19](#), [38](#), [40](#)
- [85] M. Isard and A. Blake, “Condensation - conditional density propagation for visual tracking,” *International Journal of Computer Vision*, vol. 29, pp. 5–28, 1998. [19](#), [20](#)
- [86] G. R. Bradski, “Computer vision face tracking for use in a perceptual user interface,” 1998. [20](#)
- [87] K. Sato and J. K. Aggarwal, “Temporal spatio-velocity transform and its application to tracking and interaction,” *Comput. Vision Image Understand*, vol. 96, pp. 100–128, 2004. [20](#)
- [88] M. Bertalmio, G. Sapiro, and G. Randall, “Morphing active contours,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 22, no. 7, pp. 733–737, Jul 2000. [20](#)
- [89] W. Gulyanov, C. Morand, N. Robertson, and A. Wallace, “Real-time active visual tracking with level sets,” in *Imaging for Crime Detection and Prevention (ICDP), 4th International Conference on*, Nov 2011, pp. 1–6. [20](#)
- [90] R. Ronfard, “Region-based strategies for active contour models,” *Int. J. Comput. Vision*. [20](#)
- [91] I. Matthews, T. Ishikawa, and S. Baker, “The template update problem,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 6, pp. 810–815, June 2004. [21](#)

- [92] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, “Online multiperson tracking-by-detection from a single, uncalibrated camera,” *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, [34](#)
- [93] Y. Wu, J. Lim, and M. H. Yang, “Online object tracking: A benchmark,” in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, June 2013, pp. 2411–2418. [22](#), [23](#), [44](#), [63](#), [64](#), [65](#), [68](#)
- [94] B. Han, D. Comaniciu, Y. Zhu, and L. S. Davis, “Sequential kernel density approximation and its application to real-time visual tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 7, pp. 1186–1197, July 2008. [22](#), [24](#)
- [95] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, “Robust visual tracking via multi-task sparse learning,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, June 2012, pp. 2042–2049. [22](#)
- [96] X. Jia, H. Lu, and M. H. Yang, “Visual tracking via adaptive structural local sparse appearance model,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, June 2012, pp. 1822–1829. [22](#), [64](#), [68](#)
- [97] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, “Incremental learning for robust visual tracking,” *International Journal of Computer Vision*, vol. 77, no. 1, pp. 125–141, 2008. [22](#), [68](#)
- [98] H. Grabner, C. Leistner, and H. Bischof, “Semi-supervised on-line boosting for robust tracking,” in *Proceedings of the 10th European Conference on Computer Vision: Part I*, ser. ECCV ’08, 2008, pp. 234–247. [23](#)
- [99] B. Babenko, M. H. Yang, and S. Belongie, “Robust object tracking with online multiple instance learning,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011. [23](#)
- [100] J. Gao, H. Ling, W. Hu, and J. Xing, “Transfer learning based visual tracking with Gaussian processes regression,” in *Computer Vision - ECCV, 2014*, pp. 188–203. [23](#)
- [101] S. Hare, A. Saffari, and P. H. S. Torr, “Struck: Structured output tracking with kernels,” in *2011 International Conference on Computer Vision*, Nov 2011, pp. 263–270. [23](#), [64](#)
- [102] Y. Wu, J. Lim, and M. H. Yang, “Object tracking benchmark,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, Sept 2015. [23](#)

- [103] T. Minka, “Discriminative models, not discriminative training,” Tech. Rep., October 2005. [23](#)
- [104] T. Woodley, B. Stenger, and R. Cipolla, “Tracking using online feature selection and a local generative model,” 2007. [23](#)
- [105] T. B. Dinh, Q. Yu, and G. Medioni, “Co-trained generative and discriminative trackers with cascade particle filter,” *Comput. Vis. Image Underst.*, vol. 119, pp. 41–56, Feb. 2014. [23](#)
- [106] W. Zhong, H. Lu, and M. H. Yang, “Robust object tracking via sparsity-based collaborative model,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, June 2012, pp. 1838–1845. [23](#), [64](#), [68](#)
- [107] L. Zhang and L. van der Maaten, “Preserving structure in model-free tracking,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 4, pp. 756–769, April 2014. [23](#), [132](#)
- [108] N. Wang and D.-Y. Yeung, “Learning a deep compact image representation for visual tracking,” in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds., 2013, pp. 809–817. [23](#), [64](#)
- [109] S. Hong, T. You, S. Kwak, and B. Han, “Online tracking by learning discriminative saliency map with convolutional neural network,” in *Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6-11 July, 2015*. [23](#)
- [110] J. a. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “Exploiting the circulant structure of tracking-by-detection with kernels,” in *Proceedings of the 12th European Conference on Computer Vision - Volume Part IV*, ser. ECCV’12, 2012, pp. 702–715. [23](#), [68](#)
- [111] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, “Accurate scale estimation for robust visual tracking,” in *Proceedings of the British Machine Vision Conference*. BMVA Press, 2014. [23](#), [24](#), [54](#), [60](#), [68](#)
- [112] Z. Chen, Z. Hong, and D. Tao, “An experimental survey on correlation filter-based tracking,” *CoRR*, vol. abs/1509.05520, 2015. [23](#), [24](#)
- [113] R. M. Gray, “Toeplitz and circulant matrices: A review,” *Commun. Inf. Theory*, vol. 2, no. 3, pp. 155–239, 2006. [23](#)
- [114] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M. Yang, “Fast visual tracking via dense spatio-temporal context learning,” in *Computer Vision - ECCV 2014 - 13th*

- European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, 2014, pp. 127–141. [24](#)
- [115] L. Wang, W. Ouyang, X. Wang, and H. Lu, “Visual tracking with fully convolutional networks,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 3119–3127. [24](#)
- [116] J. S. S. III and D. Ramanan, “Self-paced learning for long-term tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pp. 2379–2386. [24](#)
- [117] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, “Multi-store tracker (MUSTer): A cognitive psychology inspired approach to object tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 749–758. [24](#)
- [118] H. Nam and B. Han, “Learning multi-domain convolutional neural networks for visual tracking,” *CoRR*, vol. abs/1510.07945, 2015. [24](#)
- [119] M. Rodriguez, J. Sivic, I. Laptev, and J.-Y. Audibert, “Density-aware person detection and tracking in crowds,” in *roceedings of the International Conference on Computer Vision (ICCV)*, 2011. [24](#)
- [120] L. Kratz and K. Nishino, “Tracking pedestrians using local spatio-temporal motion patterns in extremely crowded scenes,” *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 5, pp. 987–1002, May 2012. [24](#), [25](#)
- [121] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009. [25](#)
- [122] S. Julier and J. Uhlmann, “Unscented filtering and nonlinear estimation,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, Mar 2004. [27](#)
- [123] A. Doucet, S. Godsill, and C. Andrieu, “On sequential monte carlo sampling methods for bayesian filtering,” *STATISTICS AND COMPUTING*, vol. 10, no. 3, pp. 197–208, 2000. [28](#)
- [124] T. Li, M. Bolic, and P. M. Djuric, “Resampling methods for particle filtering: Classification, implementation, and strategies,” *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 70–86, May 2015. [29](#)
- [125] C. Andrieu, “An introduction to MCMC for machine learning,” 2003. [29](#), [30](#)

- [126] A. Doucet, N. De Freitas, and N. Gordon, *Sequential Monte Carlo methods in practice*, ser. Statistics for engineering and information science. New York: Springer, 2001. [29](#)
- [127] S. Särkkä, A. Vehtari, and J. Lampinen, “Rao-blackwellized particle filter for multiple target tracking,” *Inf. Fusion*. [29](#), [81](#)
- [128] Z. Chen, “Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond,” McMaster University, Tech. Rep., 2003. [30](#)
- [129] P. J. Green, “Reversible jump markov chain monte carlo computation and bayesian model determination,” *Biometrika*, vol. 82, pp. 711–732, 1995. [30](#)
- [130] P. J. Green and J. Heikkinen, “Trans-dimensional markov chain monte carlo,” in *in Highly Structured Stochastic Systems*. University Press, 2003. [30](#)
- [131] Z. Khan, T. Balch, and F. Dellaert, “MCMC-based particle filtering for tracking a variable number of interacting targets,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 11, pp. 1805–1819, Nov 2005. [30](#), [34](#), [37](#)
- [132] I. J. Cox, “A review of statistical data association techniques for motion correspondence,” *International Journal of Computer Vision*, vol. 10, pp. 53–66, 1993. [31](#), [32](#)
- [133] S. Blackman, “Multiple hypothesis tracking for multiple target tracking,” *Aerospace and Electronic Systems Magazine, IEEE*, vol. 19, no. 1, pp. 5–18, Jan 2004. [32](#), [33](#)
- [134] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, “Multi-target tracking using joint probabilistic data association,” in *Decision and Control including the Symposium on Adaptive Processes, 19th IEEE Conference on*, Dec 1980, pp. 807–812. [32](#)
- [135] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers, “People tracking with a mobile robot using sample-based joint probabilistic data association filters,” 2003. [32](#)
- [136] J. Vermaak, S. Godsill, and P. Perez, “Monte carlo filtering for multi target tracking and data association,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 41, no. 1, pp. 309–332, 2005. [32](#), [81](#)
- [137] S. H. Rezatofighi, A. Milan, Z. Zhang, Q. Shi, A. Dick, and I. Reid, “Joint probabilistic data association revisited,” in *ICCV*, 2015. [32](#), [37](#), [124](#), [126](#)

- [138] D. Reid, “An algorithm for tracking multiple targets,” *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843–854, Dec 1979. [33](#)
- [139] I. J. Cox and S. L. Hingorani, “An efficient implementation of reid’s multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.* [33](#)
- [140] D. M. Antunes, D. M. de Matos, and J. A. Gaspar, “A library for implementing the multiple hypothesis tracking algorithm,” *CoRR*, vol. abs/1106.2263, 2011. [33](#)
- [141] R. L. Streit and T. E. Luginbuhl, “Maximum likelihood method for probabilistic multi-hypothesis tracking,” in *Signal and Data Processing of Small Targets*, ser. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, O. E. Drummond, Ed., vol. 2235, Jul 1994, pp. 394–405. [33](#)
- [142] P. Willett, Y. Ruan, and R. Streit, “PMHT: problems and some solutions,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 38, no. 3, pp. 738–754, Jul 2002. [33](#)
- [143] C. Hue, J.-P. Le Cadre, and P. Pérez, “Tracking Multiple Objects with Particle Filtering,” INRIA, Tech. Rep., 2002. [33](#), [81](#)
- [144] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, “Multiple hypothesis tracking revisited,” in *ICCV*, 2015. [33](#), [37](#)
- [145] Y. Petetin, D. Clark, B. Ristic, and D. Maltese, “A tracker based on a CPHD filter approach for infrared applications,” pp. 80 500N–80 500N–12, 2011. [33](#), [43](#)
- [146] E. Maggio, M. Taj, and A. Cavallaro, “Efficient multi-target visual tracking using random finite sets,” *IEEE Transactions On Circuits And Systems For Video Technology*, pp. 1016–1027, 2008. [33](#), [41](#), [42](#), [104](#)
- [147] L. Leal-Taixé, C. Canton-Ferrer, and K. Schindler, “Learning by tracking: Siamese CNN for robust target association,” *CoRR*, vol. abs/1604.07866, 2016. [33](#), [36](#)
- [148] W. Luo, X. Zhao, and T. Kim, “Multiple object tracking: A review,” *CoRR*, vol. abs/1409.7618, 2015. [34](#)
- [149] W. Choi, C. Pantofaru, and S. Savarese, “A general framework for tracking multiple people from a moving camera,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 7, pp. 1577–1591, July 2013. [34](#), [37](#)
- [150] A. Ess, B. Leibe, K. Schindler, and L. van Gool, “A mobile vision system for robust multi-person tracking,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR’08)*. IEEE Press, June 2008. [34](#)

- [151] A. Ess, B. Leibe, K. Schindler, and L. Van Gool, “Robust multiperson tracking from a mobile platform,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 10, pp. 1831–1846, Oct 2009. [34](#)
- [152] B. Benfold and I. Reid, “Stable multi-target tracking in real-time surveillance video,” in *CVPR*, June 2011, pp. 3457–3464. [34](#), [37](#)
- [153] G. Shu, A. Dehghan, O. Oreifej, E. Hand, and M. Shah, “Part-based multiple-person tracking with partial occlusion handling,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, June 2012, pp. 1815–1821. [34](#)
- [154] Y. Ban, S. Ba, X. Alameda-Pineda, and R. Horaud, “Tracking Multiple Persons Based on a Variational Bayesian Model,” in *ECCV Workshop on Benchmarking Multiple Object Tracking*, Amsterdam, Netherlands, Oct. 2016. [34](#), [37](#)
- [155] L. Cao, W. Chen, X. Chen, S. Zheng, and K. Huang, “An equalised global graphical model-based approach for multi-camera object tracking,” *CoRR*, vol. abs/1502.03532, 2015. [35](#)
- [156] V. Romero-Cano, G. Agamennoni, and J. Nieto, “A variational approach to simultaneous multi-object tracking and classification,” *Int. J. Rob. Res.*, vol. 35, no. 6, pp. 654–671, May 2016. [35](#)
- [157] B. Yang, C. Huang, and R. Nevatia, “Learning affinities and dependencies for multi-target tracking using a CRF model,” in *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, June 2011, pp. 1233–1240. [35](#), [36](#)
- [158] B. Yang and R. Nevatia, “An online learned CRF model for multi-target tracking.” in *CVPR*. IEEE Computer Society, 2012, pp. 2034–2041. [35](#), [36](#), [37](#)
- [159] S. Pellegrini and L. V. Gool, “Tracking with a mixed continuous-discrete conditional random field,” *Computer Vision and Image Understanding*, January 2013. [35](#), [36](#), [37](#)
- [160] N. Le, A. Heili, and J.-M. Odobez, *Long-Term Time-Sensitive Costs for CRF-Based Tracking by Detection*. Cham: Springer International Publishing, 2016, pp. 43–51. [35](#), [36](#)
- [161] J. H. Yoon, C.-R. Lee, M.-H. Yang, and K.-J. Yoon, “Online multi-object tracking via structural constraint event aggregation.” in *CVPR*, 2016. [36](#), [37](#), [107](#)
- [162] W. Choi, “Near-online multi-target tracking with aggregated local flow descriptor,” in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015. [36](#), [107](#)

- [163] B. Wang, L. Wang, B. Shuai, Z. Zuo, T. Liu, K. Luk Chan, and G. Wang, “Joint learning of convolutional neural networks and temporally constrained metrics for tracklet association,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2016. [36](#)
- [164] H. Kieritz, S. Becker, W. Hbner, and M. Arens, “Online multi-person tracking using integral channel features,” in *13th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Aug 2016, pp. 122–130. [36](#)
- [165] A. Sadeghian, A. Alahi, and S. Savarese, “Tracking the untrackable: Learning to track multiple cues with long-term dependencies,” *CoRR*, vol. abs/1701.01909, 2017. [36](#), [37](#)
- [166] S. Tang, B. Andres, M. Andriluka, and B. Schiele, “Multi-person tracking by multicut and deep matching,” *CoRR*, vol. abs/1608.05404, 2016. [36](#), [37](#)
- [167] N. d. F. Yizheng Cai and J. J. Little, “Robust visual tracking for multiple targets,” in *IN ECCV*, 2006, pp. 107–118. [37](#)
- [168] R. P. S. Mahler, “A theoretical foundation for the stein-winter probability hypothesis density (PHD) multitarget tracking approach,” in *In Processings of the 2002 MSS National Symposium of Sensor and Data Fusion, Vo.I (unclassified)*, San Antonio, TX, June 2000. [38](#)
- [169] R. Mahler, “Multitarget moments and their application to multi-target tracking,” in *In proceedings of the workshop on Estimation, Tracking and Fusion: A tribute to Yaakov Bar-Shalom, Monterey, CA*, 2001, pp. 134–166. [38](#)
- [170] R. P. Mahler, “Multitarget bayes filtering via first-order multitarget moments,” *IEEE Trans. on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, 2003. [38](#), [77](#), [134](#)
- [171] R. P. S. Mahler, *Statistical Multisource-Multitarget Information Fusion*. Norwood, MA, USA: Artech House, Inc., 2007. [38](#)
- [172] B.-N. Vo and W.-K. Ma, “The gaussian mixture probability hypothesis density filter,” *IEEE Trans. on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, 2006. [40](#), [49](#), [56](#), [83](#), [103](#), [104](#)
- [173] B.-N. Vo, S. Singh, and A. Doucet, “Sequential monte carlo methods for multi-target filtering with random finite sets,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1224–1245, 2005. [40](#), [77](#)

- [174] K. Panta, B.-N. Vo, and S. Singh, “Novel data association schemes for the probability hypothesis density filter,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 43, no. 2, pp. 556–570, 2007. [40](#)
- [175] B. Ristic, D. Clark, and B.-N. Vo, “Improved SMC implementation of the PHD filter,” in *Information Fusion FUSION, 13th Conference on*, July 2010, pp. 1–8. [40](#)
- [176] B. Ristic, D. E. Clark, B.-N. Vo, and B.-T. Vo, “Adaptive target birth intensity for PHD and CPHD filters,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 2, pp. 1656–1668, 2012. [40](#), [57](#), [91](#), [103](#), [104](#)
- [177] K. Panta, D. E. Clark, and B.-N. Vo, “Data association and track management for the gaussian mixture probability hypothesis density filter,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 3, pp. 1003–1016, 2009. [40](#), [125](#)
- [178] B. N. Vo, B. T. Vo, and D. Phung, “Labeled random finite sets and the Bayes multi-target tracking filter,” *IEEE Transactions on Signal Processing*, vol. 62, no. 24, pp. 6554–6567, Dec 2014. [40](#)
- [179] M. Beard, S. Reuter, K. Granström, B. T. Vo, B. N. Vo, and A. Scheel, “Multiple extended target tracking with labeled random finite sets,” *IEEE Transactions on Signal Processing*, vol. 64, no. 7, pp. 1638–1653, April 2016. [40](#)
- [180] D. Y. Kim, B. Vo, and B. Vo, “Online visual multi-object tracking via labeled random finite set filtering,” *CoRR*, vol. abs/1611.06011, 2016. [40](#)
- [181] R. Mahler, “PHD filters of higher order in target number,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 43, no. 4, pp. 1523–1543, October 2007. [40](#)
- [182] B.-T. Vo, B.-N. Vo, and A. Cantoni, “Analytic implementations of the cardinalized probability hypothesis density filter,” *Signal Processing, IEEE Transactions on*, vol. 55, no. 7, pp. 3553–3567, 2007. [40](#)
- [183] B. Ristic, *Particle Filters for Random Set Models*. Springer, 2013. [40](#)
- [184] B.-T. Vo, B.-N. Vo, and A. Cantoni. [40](#)
- [185] B. Ristic, B.-T. Vo, B.-N. Vo, and A. Farina, “A tutorial on bernoulli filters: Theory, implementation and applications,” *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3406–3430, 2013. [40](#)
- [186] B.-T. Vo, B.-N. Vo, R. Hoseinnezhad, and R. P. S. Mahler, “Robust multi-bernoulli filtering,” *J. Sel. Topics Signal Processing*, vol. 7, no. 3, pp. 399–409, 2013. [41](#)

- [187] R. Mahler, “PHD filters for nonstandard targets, I: extended targets,” in *Proc. International Conference on Information Fusion (FUSION '09)*, 2009, pp. 915–921. [41](#)
- [188] K. Granstrom, C. Lundquist, and O. Orguner, “Extended target tracking using a Gaussian-Mixture PHD filter,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 3268–3286, 2012. [41](#)
- [189] E. Maggio and A. Cavallaro, “Learning scene context for multiple object tracking,” *Image Processing, IEEE Transactions on*, vol. 18, no. 8, pp. 1873–1884, Aug 2009. [41](#)
- [190] R. Juang, A. Levchenko, and P. Burlina, “Tracking cell motion using GM-PHD,” in *Biomedical Imaging: From Nano to Macro. IEEE International Symposium on*, June 2009, pp. 1154–1157. [42](#)
- [191] X. Zhou, Y. F. Li, and B. He, “Entropy distribution and coverage rate-based birth intensity estimation in GM-PHD filter for multi-target visual tracking,” *Signal Process.* [42](#)
- [192] V. Edman, A. Maria, K. Granström, and F. Gustafsson, “Pedestrian group tracking using the GM-PHD filter,” in *Proceedings of the 21st European Signal Processing Conference* :, 2013. [42](#)
- [193] N. T. Pham, W. Huang, and S. Ong, “Tracking multiple objects using probability hypothesis density filter and color measurements,” in *Multimedia and Expo, IEEE International Conference on*, July 2007, pp. 1511–1514. [42](#)
- [194] M. Schikora, W. Koch, and D. Cremers, “Multi-object tracking via high accuracy optical flow and finite set statistics,” in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*, May 2011, pp. 1409–1412. [42](#)
- [195] J. H. Yoon, K.-J. Yoon, and D. Y. Kim, “Multi-object tracking using hybrid observation in PHD filter,” in *Image Processing (ICIP), 20th IEEE International Conference on*, Sept 2013, pp. 3890–3894. [42](#)
- [196] X. Zhou, Y. Li, B. He, and T. Bai, “GM-PHD-based multi-target visual tracking using entropy distribution and game theory,” *Industrial Informatics, IEEE Transactions on*, vol. 10, no. 2, pp. 1064–1076, May 2014. [42](#)
- [197] S. Pasha, B.-N. Vo, H. D. Tuan, and W.-K. Ma, “A gaussian mixture PHD filter for jump markov system models,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 45, no. 3, pp. 919–936, July 2009. [42](#)

- [198] M. Yazdian-Dehkordi and Z. Azimifar, “Refined GM-PHD tracker for tracking targets in possible subsequent missed detections,” *Signal Processing*, vol. 116, pp. 112 – 126, 2015. [42](#)
- [199] —, “Adaptive visual target detection and tracking using weakly supervised incremental appearance learning and RGM-PHD tracker,” *Journal of Visual Communication and Image Representation*, 2016. [42](#)
- [200] R. Sanchez-Matilla, F. Poiesi, and A. Cavallaro, *Online Multi-target Tracking with Strong and Weak Detections*. Cham: Springer International Publishing, 2016, pp. 84–99. [42](#)
- [201] N. Wojke and D. Paulus, “Global data association for the probability hypothesis density filter using network flows,” in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 567–572. [42](#)
- [202] R. Hoseinnezhad, B.-N. Vo, and B.-T. Vo, “Visual tracking in background subtracted image sequences via multi-bernoulli filtering,” *IEEE Transactions on Signal Processing*, vol. 61, no. 2, pp. 392–397, 2013. [42](#)
- [203] R. Hoseinnezhad, B.-N. Vo, B.-T. Vo, and D. Suter, “Visual tracking of numerous targets via multi-bernoulli filtering of image data,” *Pattern Recognition*, vol. 45, no. 10, pp. 3625–3635, 2012. [42](#)
- [204] R. Hoseinnezhad, V. Ba-Ngu, V. Ba-Tuong, and D. Suter, “Bayesian integration of audio and visual information for multi-target tracking using a CB-MeMBeR filter,” in *Acoustics, Speech and Signal Processing (ICASSP), IEEE International Conference on*, May 2011, pp. 2300–2303. [43](#)
- [205] B. Ristic, J. Sherrah, and Á. F. García-Fernández, “Performance evaluation of random set based pedestrian tracking algorithms,” *CoRR*, vol. abs/1211.0191, 2012. [43](#), [47](#)
- [206] N. T. Pham, W. Huang, and S. H. Ong, in *ACCV (1)*. [43](#)
- [207] M. Khazaei and M. Jamzad, “Multiple human tracking using PHD filter in distributed camera network,” in *Computer and Knowledge Engineering (ICCKE), 4th International eConference on*, Oct 2014, pp. 569–574. [43](#)
- [208] D. C. S. Ivekovic, “Multi-object stereo filtering in disparity space,” *COGIS*, 2009. [43](#)
- [209] F. Zhang, H. Sthle, A. Gaschler, C. Buckl, and A. Knoll, “Single camera visual odometry based on random finite set statistics,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2012, pp. 559–566. [43](#)

- [210] M. Adams, B. N. Vo, R. Mahler, and J. Mullane, “SLAM gets a PHD: New concepts in map estimation,” *IEEE Robotics Automation Magazine*, vol. 21, no. 2, pp. 26–37, June 2014. [43](#)
- [211] Y. Wei, F. Yaowen, L. Jianqian, and L. Xiang, “Joint detection, tracking, and classification of multiple targets in clutter using the PHD filter,” *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 48, no. 4, pp. 3594–3609, October 2012. [43](#), [75](#)
- [212] W. Yang, Y. Fu, and X. Li, “Joint target tracking and classification via RFS-based multiple model filtering,” *Information Fusion*, vol. 18, pp. 101–106, Jul. 2014. [43](#), [75](#)
- [213] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: The CLEAR MOT metrics,” *J. Image Video Process.*, pp. 1:1–1:10, Jan 2008. [45](#), [107](#), [124](#)
- [214] Y. Li, C. Huang, and R. Nevatia, “Learning to associate: Hybridboosted multi-target tracker for crowded scene,” in *In CVPR*, 2009. [45](#), [46](#), [124](#)
- [215] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, “A consistent metric for performance evaluation of multi-object filters,” *Signal Processing, IEEE Transactions on*, vol. 56, no. 8, pp. 3447–3457, Aug 2008. [45](#), [47](#), [94](#), [107](#), [121](#)
- [216] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, “MOTChallenge 2015: Towards a benchmark for multi-target tracking,” *arXiv:1504.01942 [cs]*, Apr. 2015, arXiv: 1504.01942. [Online]. Available: <http://arxiv.org/abs/1504.01942> [46](#)
- [217] B. Ristic, B.-N. Vo, D. Clark, and B.-T. Vo, “A metric for performance evaluation of Multi-Target tracking algorithms,” *IEEE Transactions on Signal Processing*, vol. 59, no. 7, pp. 3452–3457, Jul. 2011. [47](#)
- [218] F. S. Khan, R. M. Anwer, J. van de Weijer, A. D. Bagdanov, M. Vanrell, and A. M. López, “Color attributes for object detection,” in *IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pp. 3306–3313. [49](#)
- [219] Y. Li and J. Zhu, *A Scale Adaptive Kernel Correlation Filter Tracker with Feature Integration*. Cham: Springer International Publishing, 2015, ch. Computer Vision - ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part II, pp. 254–265. [49](#), [64](#), [68](#)

- [220] R. Rifkin, G. Yeo, and T. Poggio, “Regularized least-squares classification,” *Nato Science Series Sub Series III Computer and Systems Sciences*, vol. 190, pp. 131–154, 2003. [52](#)
- [221] C. P. Diehl and G. Cauwenberghs, “SVM incremental learning, adaptation and optimization,” in *Neural Networks. Proceedings of the International Joint Conference on*, vol. 4, July 2003, pp. 2685–2690 vol.4. [54](#), [55](#)
- [222] A. Doucet, B.-N. Vo, C. Andrieu, and M. Davy, “Particle filtering for multi-target tracking and sensor management,” in *Information Fusion. Proceedings of the Fifth International Conference on*, vol. 1, 2002, pp. 474–481 vol.1. [81](#)
- [223] R. Appel, T. Fuchs, P. Dollar, and P. Perona, “Quickly boosting decision trees – pruning underachieving features early,” in *ICML*, vol. 28, no. 3, May 2013, pp. 594–602. [102](#)
- [224] E. Ahmed, M. Jones, and T. K. Marks, “An improved deep learning architecture for person re-identification,” June 2015. [105](#)
- [225] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The KITTI dataset,” *International Journal of Robotics Research (IJRR)*, 2013. [116](#), [118](#), [119](#), [120](#)
- [226] Y. Song and M. Jeon, “Online multiple object tracking with the hierarchically adopted GM-PHD filter using motion and appearance,” in *IEEE/IEIE The International Conference on Consumer Electronics (ICCE) Asia*, 2016. [124](#), [125](#), [126](#)
- [227] H. Pirsiavash, D. Ramanan, and C. C. Fowlkes, “Globally-optimal greedy algorithms for tracking a variable number of objects,” in *CVPR 2011*, June 2011, pp. 1201–1208. [124](#), [126](#)
- [228] C. Dicle, O. I. Camps, and M. Sznaiier, “The way they move: Tracking multiple targets with similar appearance,” in *2013 IEEE International Conference on Computer Vision*, Dec 2013, pp. 2304–2311. [124](#), [126](#)
- [229] A. Milan, S. Roth, and K. Schindler, “Continuous energy minimization for multi-target tracking,” *IEEE TPAMI*, vol. 36, no. 1, pp. 58–72, 2014. [124](#), [126](#)