UNIVERSIDAD POLITÉCNICA DE MADRID
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

DETECTION AND TRACKING OF MULTIPLE TARGETS
USING WIRELESS SENSOR NETWORKS

DETECCIÓN Y SEGUIMIENTO DE MÚLTIPLES BLANCOS
EN REDES INALÁMBRICAS DE SENSORES

TESIS DOCTORAL

Ángel Froilán García Fernández
Ingeniero de Telecomunicación

2011

DEPARTAMENTO DE SEÑALES, SISTEMAS Y RADIOCOMUNICACIONES
ESCUELA TÉCNICA SUPERIOR DE INGENIEROS DE TELECOMUNICACIÓN

UNIVERSIDAD POLITÉCNICA DE MADRID

DETECTION AND TRACKING OF MULTIPLE TARGETS
USING WIRELESS SENSOR NETWORKS

DETECCIÓN Y SEGUIMIENTO DE MÚLTIPLES BLANCOS
EN REDES INALÁMBRICAS DE SENSORES

TESIS DOCTORAL

Autor:

Ángel Froilán García Fernández

Ingeniero de Telecomunicación

Directores:

Jesús Grajal de la Fuente

Profesor titular del Dpto. de Señales, Sistemas y Radiocomunicaciones
Universidad Politécnica de Madrid

Mark Richard Morelande

Senior Research Fellow del Dept. of Electrical and Electronic Engineering
The University of Melbourne

2011

TESIS DOCTORAL

DETECTION AND TRACKING OF MULTIPLE TARGETS USING
WIRELESS SENSOR NETWORKS

DETECCIÓN Y SEGUIMIENTO DE MÚLTIPLES BLANCOS EN REDES
INALÁMBRICAS DE SENSORES

AUTOR: Ángel Froilán García Fernández
DIRECTORES: Jesús Grajal de la Fuente
Mark Richard Morelande

Tribunal nombrado por el Magfco. y Excmo. Sr. Rector de la Universidad
Politécnica de Madrid, el día ____ de _____ de 2011.

PRESIDENTE: Narciso García Santos

SECRETARIO: Santiago Zazo Bello

VOCAL: Antonio Artés Rodríguez

VOCAL: Joaquín Míguez Arenas

VOCAL: Lennart Svensson

SUPLENTE: Rubén Martínez Cantín

SUPLENTE: José María Muñoz Ferreras

Realizado el acto de defensa y lectura de Tesis el día ____ de _____ de 2011.

En la E.T.S. de Ingenieros de Telecomunicación.

Calificación:

EL PRESIDENTE

LOS VOCALES

EL SECRETARIO

*‘His usual words on this subject were
that those who do not sometimes go
outside the rules never go beyond them.’*

Filippo Baldinucci

‘The life of Bernini’

*‘Taking a new step, uttering a new
word, is what [people] fear most.’*

Fyodor Dostoyevsky

‘Crime and punishment’

Acknowledgements

There are a lot of people who, directly or indirectly, have helped me come up with and develop the ideas put forward in this thesis. It was my supervisor Jesús Grajal who talked me into starting the Ph.D. so much of the credit for this thesis is his as well. Also, his positive attitude towards several setbacks has clearly smoothed the path.

The main change in the development of this thesis (also in my life over the last few years) happened when I packed, said goodbye to my beloved summer and flew down under to do research at The University of Melbourne in 2009. What was supposed to be a mere four-month collaboration with Mark Morelande to delve into a single subject ended up being a key period in which the foundations of very important parts of this thesis were laid. This fruitful collaboration has continued and Mark was appointed supervisor of this thesis. Our interesting discussions have helped me open my mind although, much to my regret, I still find it difficult to understand why a cricket match lasts so long.

I also thank my family for their support.

Special thanks go to my friends and former classmates: Antonio, Carol, Cris, Dani, Fran, Rebe and Tere.

My sincere thanks to all the members of the Grupo de Microondas y Radar. I want to specially name: Ana, Bea I, Bea II, Cristina, Diego I, Diego II, Javi, José María, Joseto, Pablo and, finally, for their collaboration to unmask the B.L. issue: Jaime and Mariano. I can only say that the atmosphere in our office would be difficult to beat.

Every inhabitant and regular visitor of my father's town, Pradoluengo, has to be acknowledged. Going there makes me disconnect from the hectic life in the city and have a good time. Special thanks go to all the members of the "peña Yo Ke Sé": Abel, Andoni, Ángel, Arkaitz, Cabal, David, Diego, Iñaki, Javi I, Javi II, Jorge, Peio and Raúl and, also, to Fernando and Nacho.

I cannot forget my neighbourhood friends: Adriana, Ana, Esther, Eva and José.

In my first research stay in Australia, I met very interesting people to have fun with. I have to highlight the “captains” Marco and Ralf, Mia and Sabrina and the Spanish researchers: Ale, Ainhoa, Esther and Gustavo.

From my second research stay in Australia, I want to name: Dev, Kam, Reina and Tomo.

I want to thank Lennart Svensson (Chalmers University of Technology) and Wei Yi (The University of Melbourne) for helping in improving some parts of this thesis. I am also very grateful to Hans Rudolf Künsch (Swiss Federal Institute of Technology Zurich) for mailing me Markus Hürzeler’s thesis even without knowing me.

Also, I would like to mention Corla (for helping me with English), David (for being a great schoolmate and putting me up in Sydney), Henar (for helping me with tedious paperwork) and Raquel (for putting up with me).

Finally, I want to indicate that this thesis has been partially supported by an FPU (Formación de Profesorado Universitario) Fellowship from the Spanish Ministerio de Educación.

Resumen

En esta tesis se desarrollan algoritmos de detección y seguimiento de múltiples blancos en redes inalámbricas de sensores usando un enfoque bayesiano. Esta tarea se lleva a cabo calculando la función densidad de probabilidad (FDP) del estado, compuesto por todas las variables que representan los blancos, condicionada a todas las medidas de los sensores (FDP a posteriori). La FDP a posteriori contiene toda la información de interés de los blancos. La detección y seguimiento de los blancos se consigue mediante el uso de estimadores aplicados a esta FDP a lo largo del tiempo. No obstante, los modelos dinámicos de los blancos y los modelos de medida de los sensores son normalmente no-lineales y no-gaussianos. Esto conlleva que la FDP a posteriori no pueda ser calculada de forma analítica y haya que recurrir a aproximaciones. Las aproximaciones basadas en filtros de partículas son convergentes si el número de partículas tiende a infinito. Sin embargo, en una situación práctica, los recursos computacionales son limitados. Esto implica que el número de partículas está acotado y el rendimiento del filtro de partículas puede ser bajo. Esta pérdida de rendimiento debida a una potencia de cálculo limitada tiene mayor importancia cuando hay múltiples blancos por la alta dimensión del estado.

Esta tesis se centra en el desarrollo de filtros de partículas con menor carga computacional y mayor rendimiento que otros propuestos previamente. Se consideran tres escenarios: detección y seguimiento de un número desconocido y variable de blancos en redes de sensores, seguimiento de múltiples blancos cuando hay incertidumbre en la posición de los sensores y el seguimiento de múltiples blancos cuando la red de sensores no está sincronizada.

En cuanto a la detección y seguimiento de un número desconocido y variable de blancos, se propone un filtro de partículas de dos capas para detectar los blancos y un algoritmo eficiente para seguir los blancos detectados. Además, se propone una técnica para extraer la información de etiquetado cuando hay dos blancos, es decir, el filtro es capaz de decidir qué blanco es cuál y con qué probabilidad de error.

El seguimiento de blancos cuando hay incertidumbre en la posición de los sensores se lleva a cabo a la vez que se localiza la posición de los sensores. Esto se basa en

técnicas de localización y *mapeado* simultáneos (SLAM) tradicionalmente usadas en robótica. Sin embargo, el hecho de que haya múltiples blancos implica que las técnicas tradicionales de SLAM no son apropiadas. Se propone una nueva técnica que supera los problemas de las técnicas de SLAM convencionales. Además, el filtro truncado de Kalman, también desarrollado en esta tesis, es un elemento clave para estimar con precisión la posición de los sensores. En la tesis también se muestra que este filtro es muy útil en una gran variedad de problemas por su alto rendimiento y bajo coste computacional.

Cuando los sensores no están sincronizados, los filtros de partículas convencionales tienen una carga computacional muy alta. Por tanto, se propone el uso del filtro de partículas asíncrono para reducir la carga computacional a la vez que se mantiene un alto rendimiento.

Abstract

This Ph.D. thesis is concerned with the development of algorithms for the detection and tracking of multiple targets using wireless sensor networks from the Bayesian standpoint. This is achieved by calculating the probability density function (PDF) of the multitarget state given the sensor measurements (posterior PDF) as it includes all the useful information to perform these tasks. The models of the target dynamics and the sensor measurements are usually nonlinear/non-Gaussian. Therefore, the posterior PDF cannot be calculated in closed form and approximations need to be made. Particle filters' approximations to the posterior PDF are convergent if the number of particles tends to infinity. However, in a practical situation, the computer power available is limited. As a result, the number of particles is bounded and particle filter performance is not guaranteed to be high. This decrease in performance due to the limited computational power is even more acute in a multiple target situation because of the high dimension of the state.

Therefore, this thesis focuses on the development of particle filtering techniques with lower computational burden and higher performance than previously developed ones. Three different scenarios are considered: the detection and tracking of an unknown and variable number of targets using a sensor network, the tracking of targets when there is uncertainty in the sensor positions and the tracking of targets when a non-synchronised sensor network is used.

As regards the detection and tracking of an unknown and variable number of targets, a particle filter with two layers is proposed to detect targets and an efficient algorithm, called the parallel partition method, is developed to track the detected targets. Also, a technique to extract target labelling information when there are two targets is proposed. That is, the filter is able to decide which target is which and determine the probability of error.

The tracking of targets when there is uncertainty in the sensor positions is carried out by simultaneously localising the sensors and tracking the targets using simultaneous localisation and mapping (SLAM) techniques, traditionally used in the field of robotics. However, the multiple target nature of the problem implies that tra-

ditional SLAM techniques are not suitable and a new technique, which is based on the parallel partition method, is proposed to overcome the problems of conventional SLAM techniques. Additionally, the truncated Kalman filter also presented in this thesis is of great importance to estimate the positions of the sensors and is shown to be a very useful filtering technique that can be applied to a variety of filtering problems.

When the sensors are not synchronised, conventional particle filtering techniques have a large computational load. Therefore, in this thesis, the asynchronous particle filter is proposed to lower their computational burden while providing accurate estimates.

Contents

1	Introduction	1
1.1	Overview of wireless sensor networks	1
1.2	Thesis objectives	4
1.3	Thesis organisation	5
2	Filtering using the Bayesian framework	9
2.1	Introduction	9
2.2	General filtering problem	10
2.3	Notation clarification	12
2.4	Kalman filter for linear/Gaussian models	13
2.5	Particle filters for nonlinear/non-Gaussian models	14
2.5.1	Sequential importance sampling	15
2.5.1.1	Degeneracy problem	18
2.5.1.2	Choice of importance density	19
2.5.2	Marginal particle filter	20
2.5.3	Auxiliary particle filter	22
2.5.4	Rao-Blackwellisation	23
2.6	Kalman filter for nonlinear/non-Gaussian models	25
2.6.1	Approximation 1: extended Kalman filter	29
2.6.2	Approximation 2: unscented Kalman filter	30
2.6.3	Approximation 3: cubature Kalman filter	31
3	Truncated unscented Kalman filtering	33
3.1	Introduction	33

3.2	Kalman filter for nonlinear measurement models	35
3.2.1	Illustration of the KF's deficiencies	36
3.3	Truncated Kalman Filter with bounded noise support	38
3.3.1	Theory	38
3.3.2	Example	39
3.4	Truncated Kalman Filter with unbounded noise support	41
3.4.1	Truncation of the measurement noise	42
3.4.2	Practical approach: use of a mixture	42
3.4.3	Truncated unscented Kalman filter	45
3.4.3.1	Approximation of the first two moments of $p_1(\cdot)$. .	45
3.4.3.2	Approximation of the prior moments	47
3.4.3.3	Selection of α	49
3.4.3.4	Single point TUKF	49
3.4.4	Discussion on the TUKF	51
3.5	Numerical examples	52
3.5.1	Range-bearing case	53
3.5.2	Update analysis in one dimension	57
3.5.3	Target tracking with range-bearing measurements	60
3.5.4	Univariate nonstationary growth model (UNGM)	66
3.6	Conclusions	68
4	Detection and tracking of multiple targets	69
4.1	Introduction	69
4.2	Joint multitrack probability density	72
4.2.1	Multitrack model	74
4.2.2	Sensor models	76
4.3	Particle filter implementation for a fixed and known number of tracks	78
4.3.1	Parallel partition method	80
4.3.1.1	Sampling from the importance density	82
4.3.1.2	Discussion about the PP method	83
4.4	Particle filter implementation for a variable and unknown number of tracks	85

4.4.1	Importance density for removing tracks	87
4.4.2	Importance density for proposing new tracks. Pre-track particle filter	88
4.5	Simulation results	91
4.5.1	Tracking performance	91
4.5.2	Estimation of the number of targets	103
4.6	Conclusions	106
5	Improved detection and extraction of target labelling information	107
5.1	Introduction	107
5.2	Improved multiple target detection algorithms via sequential hypothesis tests	107
5.2.1	Formulation of the problem	110
5.2.2	Sequential multiple hypothesis tests	111
5.2.2.1	Fixed number of particles sequential test	111
5.2.2.2	Sequential test based on the JMPD	113
5.2.2.3	Variable number of particles sequential test	114
5.2.3	Case study	114
5.2.3.1	Implementation for the auxiliary particle filter	114
5.2.3.2	Simulations	116
5.2.3.3	Comparison with the subparticle Neyman-Pearson detector	120
5.3	Extraction of target labelling information	121
5.3.1	Problem statement	124
5.3.2	Information about target labelling: Previous work	125
5.3.2.1	Permutation invariant PDF	125
5.3.2.2	Permutation strictly variant PDF	126
5.3.2.3	Unique decomposition of a PDF	127
5.3.2.4	Regions of interest of the unique decomposition	127
5.3.2.5	Example	128
5.3.3	On the probability of successful labelling	128
5.3.3.1	Example	130

5.3.4	Probability of successful labelling: Targets have separated after being in close proximity	131
5.3.5	Particle filter implementation	132
5.3.5.1	Approximation of the region A^{k+1}	133
5.3.5.2	Exploration of the mirror mode	133
5.3.5.3	Problems with conventional particle filters	134
5.3.5.4	Marginal particle filter	135
5.3.6	Estimation and labelling	136
5.3.6.1	Labelling of the MMOSPA estimator	137
5.3.6.2	Approximation of the labelled MMOSPA estimator using the particle filter with mirror particles	138
5.3.7	Simulations	140
5.4	Conclusions	145

6 Multitarget simultaneous localisation and mapping of a sensor network 147

6.1	Introduction	147
6.2	Problem statement	150
6.3	FSLAM versus Multitarget SLAM	152
6.3.1	FSLAM	153
6.3.2	Multitarget SLAM	154
6.4	Approximation of the target state posterior	157
6.4.1	Sampling the targets' positions	158
6.4.2	Update of the targets' velocities	160
6.4.3	Calculation of the particles' weights	161
6.5	Sensors' positions update	163
6.5.1	Approximation of the first two moments of the truncated prior	163
6.5.1.1	Global minimisation	165
6.5.1.2	Local minimisation	165
6.5.2	Selection of α	166
6.5.3	Implementation of the algorithm	167
6.6	Simulation results	167

6.7	Conclusions	175
7	Tracking of multiple targets using non-synchronised sensor networks	177
7.1	Introduction	177
7.2	Problem statement	179
7.2.1	Dynamic model of targets	180
7.2.2	Sensor model	181
7.3	Asynchronous particle filter	182
7.4	Application to multiple target tracking	185
7.4.1	Asynchronous likelihood	185
7.4.2	Importance density	188
7.5	Simulation results	188
7.6	Conclusions	192
8	Conclusions and future work	197
8.1	Conclusions	197
8.2	Future work	198
8.2.1	Truncated Kalman filtering	199
8.2.2	Detection and tracking of multiple targets in general	199
8.2.3	Detection and tracking of multiple targets using wireless sensor networks	200
A	Additional mathematical derivations	203
A.1	TKF analytical example calculations	203
A.1.1	Kalman filter moments	203
A.1.2	Useful relationships	204
A.1.3	Truncated Kalman filter moments	204
A.1.4	Mean square errors of the estimators	206
A.2	TKF with unbounded noise support: Calculations using AP1 and AP2207	
A.3	Truncated Kalman filter proof	208
A.4	TKF: Application of two Kalman filters to the prior mixture	209
A.5	TUKF: First two moments of the truncated prior (I)	210

A.6	TUKF: First two moments of the truncated prior (II)	212
A.7	SP-TUKF: Approximation of the first two moments of $p_2(\cdot)$	213
A.8	TUKF: First two moments of $p_1(\cdot)$ in the numerical examples	213
A.8.1	Range-bearing case	213
A.8.2	One-dimensional update case	214
A.8.3	Univariate nonstationary growth model	214
A.9	Kullback-Leibler divergence	214
A.10	Multitarget state explanation	215
A.11	Set and vector notation equivalence for the labels	215
A.12	Simplification of the transition density of the JMKPD	216
A.13	Rao-Blackwellisation of target velocities	217
A.14	Sampling states and labels of the surviving targets	219
A.14.1	Explanation using target existences	220
A.14.1.1	Posterior of the target states and labels	220
A.14.1.2	Types of importance densities	221
A.14.1.3	Numerical example	224
A.14.2	Explanation using sets for the labels	228
A.14.3	Discussion on the application to the two-layer PF	229
A.15	Approximating the evidence using particle filters	230
A.16	Kullback-Leibler divergence: Proof	231
A.16.1	Behaviour with the number of targets	231
A.16.2	Behaviour with SNR_0	232
A.17	Example: Unique decomposition of a PDF	232
A.18	Form of the posterior after a target crossing (I)	233
A.18.1	Subsequent encounters	235
A.18.2	Conclusions	236
A.19	Form of the posterior after a target crossing (II)	236
A.20	Importance density for mirror particles	237
A.21	Proof that $\hat{\alpha}^k \leq 1$ using the marginal PF	238
A.22	Alternative importance density for the mirror PF	238
A.23	MSLAM prediction step for the targets' positions	239

A.24 MSLAM: Approximation of the first-stage weights using the unscented transformation	240
A.25 MSLAM: Update of the targets' velocities	242
A.26 MSLAM: Map mixing step	243
A.27 MSLAM: Jacobian of the measurement function	244
A.28 MSLAM-TUKF: moments of the truncated prior for overdetermined measurement equations	244
A.29 Asynchronous PF: Approximation of the asynchronous likelihood . .	246
A.30 Asynchronous PF: Calculation of the PDF of a target position at a sampling time instant	247
B Comparison between set and vector notations for tracking	249
B.1 Fixed and known number of targets	249
B.1.1 Posterior PDF representation	251
B.1.2 Prediction phase: vector notation without labels	252
B.1.3 Prediction phase: vector notation with labels	252
B.1.4 Prediction phase: set notation without labels	252
B.1.5 Prediction phase: set notation with labels	254
B.1.6 PHD filter	255
B.1.6.1 PHD for sets without labels	255
B.1.6.2 PHD for labelled sets	256
B.2 Variable and unknown number of targets	256
B.2.1 Situation at time k	257
B.2.2 Posterior PDF at time k using a vector with existence variables	258
B.2.3 Posterior PDF at time k using a labelled vector	258
B.2.4 Posterior PDF at time k using a vector without labels	259
B.2.5 Posterior PDF at time k using a labelled set	260
B.2.6 Posterior PDF at time k using a set without labels	261
B.2.7 Comments on the mathematical validity of a labelled vector	261
B.3 Conclusions	261
Bibliography	263

List of Figures

1.1	Centralised configuration of a wireless sensor network.	2
1.2	Distributed configuration of a wireless sensor network.	3
2.1	Filtering problem state diagram.	11
3.1	Illustration of Kalman filter's deficiencies.	37
3.2	Bayes' rule with bounded noise support.	40
3.3	Truncated Kalman filter with bounded noise support.	41
3.4	Truncated Kalman filter with unbounded noise support.	43
3.5	Illustration of the TUKF performance.	53
3.6	Labelling of the true target positions around the prior mean.	54
3.7	TUKF: Range-bearing update analysis.	56
3.8	TUKF: Kullback-Leibler divergence for the 1-D case.	58
3.9	TUKF: Posterior mean for the 1-D case.	59
3.10	TUKF: Posterior variance for 1-D case.	59
3.11	TUKF: RMSE for the 1-D case.	60
3.12	TUKF: Scenario for the range-bearing tracking.	61
3.13	TUKF: Performance in the range-bearing tracking scenario.	63
3.14	TUKF: True state for the univariate nonstationary growth model.	67
3.15	TUKF: Performance in the UNGM scenario.	67
4.1	Subparticle crossover.	84
4.2	Two-layer particle filter diagram.	86
4.3	Scenario for multiple target tracking performance.	94
4.4	Time-averaged RMS position error for multiple target tracking.	96

4.5	Performance of the algorithms in multiple target tracking.	98
4.6	RMS position error using K-means partition swapping.	99
4.7	Percentage of surviving subparticles.	99
4.8	Scenario with increasing number of targets.	101
4.9	Scenario to analyse the estimation of the number of targets.	104
4.10	Estimated number of targets with time.	105
5.1	Scenario: Multiple target detection.	117
5.2	Probability of classification for the multiple target detection algorithms.	119
5.3	Stopping time for the multiple target detection algorithms.	119
5.4	Receiver operating characteristic for the subparticle Neyman-Pearson detector and fixed number of particles test.	120
5.5	Target labelling problem.	121
5.6	Variant-invariant decomposition example (I).	129
5.7	Variant-invariant decomposition example (II).	129
5.8	Variant-invariant decomposition example (III).	129
5.9	Scenario for assessing target labelling extraction.	142
5.10	Constellations of labelled particles at different time steps.	142
5.11	Evolution of $\hat{\alpha}^k$ with time.	144
5.12	MOSPA state error for the mirror particle filter.	144
6.1	FSLAM adapted to multiple targets.	154
6.2	Map degeneracy in FSLAM.	155
6.3	Diagram of an MSLAM update step	157
6.4	MSLAM: Scenario of the simulations.	169
6.5	MSLAM: Performance of the algorithms for three targets.	170
6.6	MSLAM: Final RMS position error map for the sensors.	172
6.7	MSLAM: RMS position error of the targets per time step.	173
6.8	MSLAM: RMS position error of the targets for different number of particles.	174
6.9	MSLAM: RMS position error with one target.	174
7.1	Diagram of states with non-synchronised sensor networks.	181

7.2	Particle propagation for different filters.	183
7.3	Scenario for assessing the asynchronous particle filter.	191
7.4	Asynchronous particle filter. Performance in scenario 1.	193
7.5	Asynchronous particle filter. Performance in scenario 2.	193
7.6	Asynchronous particle filter. Performance in scenario 3.	194
7.7	Normalised execution time of the asynchronous particle filter.	194
7.8	Asynchronous PF. Performance in scenario 1 against the number of particles.	195
A.1	Effective sampling size for three sampling methods.	227
A.2	RMS error of the estimated coefficients for three sampling methods.	227

List of Tables

2.1	SIS particle filter	18
2.2	Systematic resampling algorithm	19
2.3	SIR particle filter	21
2.4	Auxiliary particle filter steps	24
3.1	Truncated unscented Kalman filter update phase steps	50
3.2	Single point truncated unscented Kalman filter update phase steps .	51
3.3	Averaged RMSE for range-bearing scenarios	57
3.4	Parameters of the 1-D simulation	58
3.5	RMS error averaged over both the parameter and the measurement .	60
3.6	Time-averaged RMS position error for different values of γ (range-bearing)	63
3.7	Time-averaged RMS position error for different measurement noise parameters	65
3.8	Time-averaged RMS position error for different values of γ (UNGM)	67
3.9	Time-averaged RMS position error for different measurement noise parameters	68
4.1	Independent and parallel partition subroutines	84
4.2	Proposed algorithm for detection and tracking of multiple targets .	92
4.3	Properties of the importance densities	93
4.4	Simulation parameters	94
4.5	Execution times in seconds for 500 particles	100
4.6	Required number of particles to obtain an RMS position error lower than 25 m	102

5.1	Adaptation of the number of particles per hypothesis in VNP-ST	115
5.2	Auxiliary marginal PF with mirror particles	137
5.3	Parameters of the simulation	141
5.4	Probability of successful labelling P_{sl} and $\hat{\alpha}^{79}$ against the distance between targets	143
5.5	Time-averaged MOSPA state error	145
6.1	Parallel partition subroutine for multitarget SLAM	160
6.2	Subroutine for updating a sensor position	166
6.3	Proposed algorithm for multitarget SLAM	167
6.4	Model parameters	168
6.5	Algorithm parameters	168
6.6	Execution times in seconds for $\sigma_s = 4$ m.	175
7.1	Independent partition subroutine for the asynchronous particle filter	189
7.2	Proposed algorithm for tracking multiple targets using non-synchronised sensors	189
7.3	Sampling period parameters of the sensors in the scenarios	192
A.1	Likelihood for the numerical example	225

List of Acronyms

CDF	Cumulative density function
CKF	Cubature Kalman filter
CP	Coupled partition
EKF	Extended Kalman filter
FNP-ST	Fixed number of particles sequential test
IP	Independent partition
JMKPD	Joint multitrack probability density
JMPD	Joint Multitarget probability density
JMPD-ST	Joint multitarget probability density sequential test
JPDAF	Joint probabilistic data association filter
KF	Kalman filter
KL	Kullback-Leibler
LRT	Likelihood ratio test
MAP	Maximum a posteriori
MCKF	Monte Carlo Kalman filter
MHT	Multiple hypothesis tracking
MMOSPA	Minimum mean optimal subpattern assignment
MMSE	Minimum mean square error
MOSPA	Mean optimal subpattern assignment
MSLAM	Multitarget simultaneous localisation and mapping

OID	Optimal importance density
OSPA	Optimal subpattern assignment
PDF	Probability density function
PP	Parallel partition
RFS	Random finite set
RMS	Root mean square
RMSE	Root mean square error
SIR	Sampling importance resampling
SIS	Sequential importance sampling
SLAM	Simultaneous localisation and mapping
SP-TUKF	Single point truncated unscented Kalman filter
TKF	Truncated Kalman filter
TUKF	Truncated unscented Kalman filter
UKF	Unscented Kalman filter
UT	Unscented transformation
VNP-ST	Variable number of particles sequential test
WSN	Wireless sensor network

Chapter 1

Introduction

1.1 Overview of wireless sensor networks

A wireless sensor network (WSN) comprises a large number of inexpensive, battery-powered nodes which are deployed in a region of interest. The nodes are equipped with sensors and collect information on phenomena within that region. They also include a radio frequency transceiver to be able to transmit and receive information messages through a wireless channel. Interest in these kinds of networks has considerably increased over the last years due to their great flexibility, robustness and low deployment cost [2, 42, 43, 186, 209].

There are several aspects of WSNs that are worth highlighting to put this thesis in context. The following list includes the topics that are addressed in this introduction without trying to be an exhaustive account of wireless sensor network features:

- Processing of information in the nodes.
- Information fusion.
- Applications.
- Deployment of the network.
- Synchronisation of the nodes.
- Sensor management.

The messages sensors transmit can contain raw or processed data. The objective of processing the information in each node before transmission is to send less information through the communication channel. This way, a lower bandwidth is necessary and battery lasts longer extending the life span of the network. However, this data

Figure 1.1: Centralised configuration of a wireless sensor network. A circle denotes a node and a dashed line ending in an arrow a wireless channel. All the nodes send their information to a fusion centre that performs data fusion. The fusion centre is able to send messages to the nodes.

processing in the nodes is usually simple because of their limited computational capabilities and energy consumption constraints. Additionally, one should note that compressing the information implies that the system's performance inevitably deteriorates.

Depending on how the messages are transmitted and data fusion is carried out, there are two possible configurations of the network: centralised and distributed [166]. In the centralised configuration, the nodes send their data directly to a fusion centre, as shown in Figure 1.1. The fusion centre is in charge of combining all the data coming from the sensors and providing the operator with the required information. The fusion centre does not usually have battery or computational resource constraints and, therefore, sophisticated algorithms of data fusion can be implemented in principle. On the contrary, in the distributed configuration, the fusion centre does not usually exist. Each node communicates with its neighbours and data fusion is accomplished in a distributed manner [36, 104, 193], see Figure 1.2. This implies that the algorithms cannot be as sophisticated as in the centralised configuration because of the lack of resources of the nodes. Moreover, a decentralised algorithm will never perform better than a properly designed centralised one. Nevertheless, the distributed configuration is considered to be more robust than the centralised one as it does not depend on a fusion centre to operate.

The applications of these kinds of networks are manifold, ranging from surveillance to environmental, industrial or health monitoring [209]. Environmental monitoring encompasses gathering information about temperature, atmospheric pressure, rainfall, pollution [211], volcanic activity [204] or animal tracking [87]. An

Figure 1.2: Distributed configuration of a wireless sensor network. A circle denotes a node and a dashed line a wireless channel. The nodes communicate with neighbouring nodes and data fusion is performed in a distributed fashion.

important surveillance application is the detection and tracking of targets that enter a certain region [1,182]. It is of interest to deploy a wireless sensor network where other more sophisticated, expensive sensors such as radars fail or are not economically feasible. For instance, in mountainous regions or with exuberant vegetation, radar's performance might be poor as shadow regions could be extensive and sensor networks are a reasonable alternative to detect and track people or vehicles.

The deployment of the network can be carried out in two ways. One is a carefully planned deployment so that the nodes are placed in known positions. The other is a less planned deployment in which the sensor locations are not accurately known. The sensors can be located practically at random in a certain region, for instance, if they have been thrown from an aeroplane or with high inaccuracy if the network has been deployed quickly. Accurately knowing the sensor positions is rather advantageous as the performance of a network with known sensor positions will always outperform another in which the sensor positions are vaguely known or not known at all. Nonetheless, deploying the network in such a way that the sensor positions are known can be time-consuming. It is clear that equipping the nodes with a built-in global positioning system (GPS) receiver allows an accurate knowledge of the sensor positions even if the network has been deployed quickly. However, the facts that the nodes are battery-powered and inexpensive and, also, that maximising the life span of the network is of vital importance prevent us from adding a GPS module to the nodes. Therefore, extensive research has been done to overcome this difficulty. One approach is to develop fusion algorithms that assume that the sensors are deployed at random within the surveillance region [154–156]. Nonetheless, due to the improvement in performance that can be achieved by knowing the sensor positions, it is advisable that the network should have the intrinsic ability to localise its nodes. A possible way to do it is called self-localisation [173,186]. Based on an accurate knowledge of the positions of some nodes, referred to as anchor nodes, self-localisation can infer the positions of the rest of the nodes by sending information packages

between the nodes. These information packages contain measurements pertained to the relative positions between two nodes, e.g., measurements of power of the signal received by a node that has been transmitted by another one.

Synchronisation of the sensors is also important [65, 125]. Every sensor has its own internal clock that governs its activity and, thus, sets its sampling frequency. Due to the fabrication process, the frequency and phase of these clocks are different. It is possible to estimate and compensate the timing offsets at the sensors using synchronisation protocols. However, it is done at the expense of a significant increase in the communication overhead and, as a result, an undesired reduction of the life of the nodes [194]. Then, it is preferable to develop protocols that synchronise the clocks with minimal communication overheads or, at least, develop algorithms that account for this lack of synchronisation without correcting it.

Sensor management is also a field of intensive research [89]. It deals with the proper administration of the scarce resources of the sensors. For example, it could control the sleeping periods or the sensing modality of the nodes. Sensor management has been approached from very different perspectives based on: Markov decision problems (MDP) or partially observed Markov decision problems (POMDP) [89, 186], information theory techniques such as the Shannon or the Rényi entropies [89, 115, 117, 210], game theory [134, 186] or posterior Cramér-Rao lower bound (PCRLB) [188, 189].

1.2 Thesis objectives

This thesis is mainly concerned with the detection and tracking of multiple targets using a centralised sensor network in the Bayesian framework. From the Bayesian point of view, one is interested in approximating the posterior probability density function (PDF) of the state as it contains all the data about the characteristics of interest of the targets in the surveillance area. This is due to the fact that the posterior PDF includes all the available information about the state [13].

The posterior PDF is very useful as it allows the use of the most commonly used estimators such as the minimum mean square error (MMSE) estimator or the maximum a posteriori (MAP) estimator to infer quantities of interest about the targets such as the number of targets, their positions and velocities. When the system is linear and Gaussian, the posterior PDF admits a closed-form solution that is given by the Kalman filter [13]. However, in general, the dynamic models of the targets and the measurement models are nonlinear/non-Gaussian. In this case, particle filters are a successful technique to perform sequential estimation in the Bayesian framework [9]. Therefore, as most models are nonlinear in reality,

the algorithms proposed in this thesis are mainly improvements related to particle filtering techniques although Kalman filtering also plays a very important role in this thesis.

There are three main subjects that are addressed:

1. Development of a mathematical tool for detecting and tracking an unknown and variable number of targets. The objective is to determine how many targets there are in the surveillance area and where these targets are at each time to build the tracks. Track building in this case is achieved through target labelling. The theory is presented to perform these tasks by means of a sensor network. Nevertheless, it is a theory with general applicability and the results can be applied to perform these tasks with other kinds of sensors, i.e., not necessarily a sensor network.
2. Development of an algorithm for simultaneous localisation of sensors and tracking of multiple targets. This algorithm aims to track multiple targets while at the same time localising the sensors assuming there is some prior knowledge available about the initial position of the targets and sensors.
3. Development of a computationally efficient algorithm for tracking multiple targets using sensor networks in which the sensors are not synchronised.

In all the subjects mentioned above, there were methods for dealing with them prior to the beginning of the research leading to the writing of this thesis. However, the characteristics inherent to sensor networks and multiple target tracking were not fully exploited to achieve such a high performance as the algorithms developed in this thesis. These previous methods are properly acknowledged in each of the chapters and comparisons with the most significant ones are also included.

1.3 Thesis organisation

The rest of this thesis is organised such that it goes from the general problem of Bayesian filtering to its application to multiple target tracking and sensor networks:

- Chapter 2 reviews the fundamentals of Bayesian filtering. Firstly, the general filtering problem with nonlinear/non-Gaussian systems is formulated. Secondly, a survey on the widely used Bayesian filtering algorithms necessary to understand this thesis is provided. These algorithms are based on Kalman and particle filtering.

- Chapter 3 indicates the drawbacks of Kalman filtering applied to systems with nonlinear measurement models. Then, the case when the measurement function is bijective and the PDF of the measurement noise is bounded is analysed. Under these conditions, an alternative to conventional Kalman filtering to approximate the first two moments of the posterior PDF is proposed. This algorithm is referred to as truncated Kalman filter (TKF). Practical implementations of the TKF, such as the truncated unscented Kalman filter (TUKF) or the single point truncated unscented Kalman filter (SP-TUKF), do not require a PDF of the measurement noise with a bounded support. This tool proves to be an essential component of the algorithm developed in Chapter 6. Some parts of this chapter have been published in [72] and submitted for publication [74].
- Chapter 4 addresses the first subject of the thesis, i.e., detecting and tracking multiple targets. Firstly, the joint multitrack probability density (JMKPD) is proposed as a way to detect and track an unknown and variable number of targets from the Bayesian perspective. Secondly, a particle filter with low computational burden, especially designed to deal with multitarget situations, is developed to approximate the JMKPD. Two novelties characterise this particle filter implementation. Sampling the current target states is performed using a new method called the parallel partition (PP) method and the creation and deletion of tracks is done via a two-layer particle filter. Some parts of this chapter have been published in [66] and submitted for publication [69].
- Chapter 5 explores more deeply some topics on the first subject of the thesis. Particularly, detection techniques that allow the detection of multiple moving targets in close proximity are proposed. Additionally, how to extract target labelling information from the posterior PDF is explained for a two-target case. The extraction of target labelling information consists of inferring the probability that an estimate of a target state corresponds to a specific target. This is of special interest for targets that have moved in close proximity and then separated. Some parts of this chapter have been published in [68, 73].
- Chapter 6 deals with the second subject of this thesis, i.e., simultaneous localisation of sensor positions and targets. The technique called multitarget simultaneous localisation and mapping (MSLAM) is developed. MSLAM is based on the PP method explained in Chapter 4. In conjunction with the TUKF or the SP-TUKF, MSLAM vastly outperforms traditional techniques for simultaneous localisation and mapping based on particle filtering. Therefore, this chapter builds on some of the ideas of Chapters 3 and 4. Some parts

of this chapter have been published in [71].

- Chapter 7 addresses the third subject of this thesis proposing the asynchronous particle filter. When the sensors are not synchronised, conventional particle filters have a high computational burden. On the contrary, the asynchronous particle filter has a low computation burden with a high performance. Some parts of this chapter have been published in [67].
- Chapter 8 includes the conclusions and future work.

Chapter 2

Filtering using the Bayesian framework

2.1 Introduction

In estimation problems, there are usually two approaches to estimate the quantity of interest (in this thesis scalar or vector valued) from noisy measurements. The non-Bayesian approach considers the quantity to be a deterministic but unknown parameter. On the contrary, the Bayesian approach considers the quantity to be a random variable and assumes that some prior knowledge is available [105]. This prior knowledge is represented by a prior PDF that includes all the knowledge about the quantity of interest before a measurement is taken. Even though in some cases there is some controversy about which approach should be followed, in dynamic state estimation, the Bayesian formulation is commonly accepted although other approaches are also possible [160, 172]. In dynamic state estimation, the prior knowledge refers to the initial value of the quantity and how this quantity evolves with time.

In target tracking, which is a common case of dynamic state estimation, the quantity to estimate is usually a vector that represents the position and velocity of the target. For example, if the position and velocity of the target are known at a certain time, it makes sense to assume that the position and velocity at a near time stamp in the future is linked to this state via the dynamics of the target. One way to model this “prior” evolution of the quantity of interest with time is by means of a probability density function, giving rise to the Bayesian framework.

2.2 General filtering problem

In dynamic state estimation, filtering refers to the estimation of the current state of the system from noisy measurements obtained up to the current time [13]. The estimation of the state is performed by means of estimators. Estimators of interest such as the minimum mean square error (MMSE) estimator [105], maximum a posteriori (MAP) estimator [105] or the minimum mean optimal subpattern assignment (MMOSPA) estimator [83] require the knowledge of the posterior probability density function (PDF), i.e., the PDF of the current state given the measurements. For this reason, it is of great interest to calculate the posterior PDF.

The dynamic state at time k is represented by a state vector $\mathbf{x}^k$ taking values in a state space $\mathcal{X} \subseteq \mathbb{R}^{n_x}$ where n_x is the dimension of $\mathbf{x}^k$. The state is modelled by a discrete Markov process that evolves as [6]

$$\mathbf{x}^k | \mathbf{x}^{k-1} \sim f^k(\cdot | \mathbf{x}^{k-1}), k = 1, 2, \dots \quad (2.1)$$

with

$$\mathbf{x}^0 \sim \pi^0(\cdot) \quad (2.2)$$

where $\sim$ means “is distributed as”, $f^k(\cdot | \mathbf{x}^{k-1})$ is a Markov transition PDF and $\pi^0(\cdot)$ is the prior PDF at time 0. The state at time k is observed through the measurement vector $\mathbf{z}^k$ taking values in an observation space $\mathcal{Z} \subseteq \mathbb{R}^{n_z}$, where n_z is the dimension of $\mathbf{z}^k$, that satisfies

$$\mathbf{z}^k | \mathbf{x}^{0:k}, \mathbf{z}^{1:k-1} \sim l^k(\cdot | \mathbf{x}^k), k = 1, 2, \dots \quad (2.3)$$

where $l^k(\cdot | \mathbf{x}^k)$ is the conditional density of the measurement based on the state $\mathbf{x}^k$, $\mathbf{x}^{0:k} = (\mathbf{x}^0, \mathbf{x}^1, \dots, \mathbf{x}^k)$ is the sequence of states from time 0 to k and similarly for $\mathbf{z}^{1:k-1}$. The system described by (2.1), (2.2) and (2.3) admits a graphical representation, which is shown in Figure 2.1. Once a measurement $\mathbf{z}^k$ has been received, the function $l^k(\mathbf{z}^k | \cdot)$ is called the likelihood.

In the Bayesian framework, the estimation of the current state is based on the posterior PDF $\pi^k(\cdot | \mathbf{z}^{1:k})$, i.e., the PDF of the current state $\mathbf{x}^k$ given the measurements $\mathbf{z}^{1:k}$. The posterior PDF contains all the available information about the state and can be calculated recursively in two stages: prediction and update. The objective is to calculate $\pi^{k+1}(\cdot | \mathbf{z}^{1:k+1})$ based on $\pi^k(\cdot | \mathbf{z}^{1:k})$ and the system equations (2.1) and (2.3).

The prediction stage involves using the process model (2.1) to obtain the prior PDF of the state at time $k + 1$. The prior PDF of the state at time $k + 1$,

Figure 2.1: State diagram that shows the statistical dependencies in the filtering problem: The process is Markov and the measurements are conditionally independent given the states.

$\varpi^{k+1}(\cdot | \mathbf{z}^{1:k})$, denotes the PDF of the state at time $k + 1$ given the measurements $\mathbf{z}^{1:k}$ up to time k . This is found via the Chapman–Kolmogorov equation:

$$\varpi^{k+1}(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \int f^{k+1}(\mathbf{x}^{k+1} | \mathbf{x}^k) \pi^k(\mathbf{x}^k | \mathbf{z}^{1:k}) d\mathbf{x}^k \quad (2.4)$$

The update stage makes use of Bayes’ rule:

$$\pi^{k+1}(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) = \frac{l^{k+1}(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) \varpi^{k+1}(\mathbf{x}^{k+1} | \mathbf{z}^{1:k})}{\int l^{k+1}(\mathbf{z}^{k+1} | \mathbf{x}) \varpi^{k+1}(\mathbf{x} | \mathbf{z}^{1:k}) d\mathbf{x}} \quad (2.5)$$

$$\propto l^{k+1}(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) \varpi^{k+1}(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) \quad (2.6)$$

where $\propto$ means “proportional to”. It should be noted that the denominator in (2.5) does not depend on $\mathbf{x}^{k+1}$ and, consequently, it is a normalising constant whose function is to normalise the posterior such that its integral is one:

$$\int \pi^{k+1}(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) d\mathbf{x}^{k+1} = 1 \quad (2.7)$$

Therefore, in principle, one can calculate the posterior solving recursively (2.4) and (2.6). However, an analytical solution to these equations is rarely found. In fact, Bayesian filtering has an analytical solution if [181]

1. The marginalisation over $\mathbf{x}^k$ in (2.4) is analytically tractable.
2. The resulting posterior $\pi^{k+1}(\cdot | \mathbf{z}^{1:k+1})$ in (2.6) has the same functional form as the previous step $\pi^k(\cdot | \mathbf{z}^{1:k})$, allowing the procedure to be iterated.

These conditions are only met for a very limited class of models [44] and one must resort to approximations in general.

2.3 Notation clarification

It should be noted that the PDFs involved in the previous section, $f^k(\cdot | \mathbf{x}^{k-1})$, $l^k(\cdot | \mathbf{x}^k)$, $\varpi^{k+1}(\cdot | \mathbf{z}^{1:k})$ and $\pi^k(\cdot | \mathbf{z}^{1:k})$, are different. That is the reason why they are identified with different letters and a time index k . In addition, one should distinguish between the PDF itself, e.g., $\pi^k(\cdot | \mathbf{z}^{1:k})$ and the value the PDF takes for a given argument, e.g., $\pi^k(\mathbf{x}^k | \mathbf{z}^{1:k})$.

A common convention, which is used to simplify notation, is to assume that it is the argument that defines the PDF [13, 164]. Accordingly, the following equivalences are met

$$p(\mathbf{x}^{k+1} | \mathbf{x}^k) = f^{k+1}(\mathbf{x}^{k+1} | \mathbf{x}^k) \quad (2.8)$$

$$p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) = l^{k+1}(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) \quad (2.9)$$

$$p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \varpi^{k+1}(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) \quad (2.10)$$

$$p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) = \pi^{k+1}(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) \quad (2.11)$$

and the Chapman-Kolmogorov equation and Bayes' rule become:

$$p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \int p(\mathbf{x}^{k+1} | \mathbf{x}^k) p(\mathbf{x}^k | \mathbf{z}^{1:k}) d\mathbf{x}^k \quad (2.12)$$

$$p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) = \frac{p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k})}{p(\mathbf{z}^{k+1} | \mathbf{z}^{1:k})} \quad (2.13)$$

$$= \frac{p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k})}{\int p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) d\mathbf{x}^{k+1}} \quad (2.14)$$

$$\propto p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) \quad (2.15)$$

This notation will be used in most parts of this thesis as it is always clear what PDF is being referred to. However, there is one part of the thesis, namely Section 5.3, in which evaluating the posterior PDF for different values is necessary, e.g., $\pi^k(\mathbf{a} | \mathbf{z}^{1:k})$ and $\pi^k(\mathbf{b} | \mathbf{z}^{1:k})$ where $\mathbf{a}$ and $\mathbf{b}$ are two different vectors of the proper dimension. This cannot be done with the notation in which a PDF is defined by its argument. However, as the number of PDFs involved in this thesis is large, using the first convention would require the definition of numerous PDFs, each one with

a different symbol, and the resulting notation could be cumbersome. Therefore, the second convention will be used unless otherwise stated.

2.4 Kalman filter for linear/Gaussian models

When the system, which is described by (2.1), (2.2) and (2.3), is linear and Gaussian, the posterior PDF admits a closed-form solution that can be calculated recursively. A linear and Gaussian system is defined by the following state space equations [9]:

$$\mathbf{x}^{k+1} = \mathbf{F}^{k+1}\mathbf{x}^k + \mathbf{w}^{k+1} \quad (2.16)$$

$$\mathbf{z}^{k+1} = \mathbf{H}^{k+1}\mathbf{x}^{k+1} + \mathbf{v}^{k+1} \quad (2.17)$$

where $\mathbf{F}^{k+1}$ and $\mathbf{H}^{k+1}$ are known matrices, $\mathbf{w}^{k+1}$ is the process noise which is Gaussian distributed with zero mean and covariance matrix $\mathbf{Q}^{k+1}$, $\mathbf{v}^{k+1}$ is the measurement noise which is Gaussian distributed with zero mean and covariance matrix $\mathbf{R}^{k+1}$. The noise vectors at each time step are assumed to be independent. Note that the matrices $\mathbf{F}^k$, $\mathbf{H}^k$, $\mathbf{Q}^k$ and $\mathbf{R}^k$ are allowed to be time variant. It is also assumed that the prior at time 0 is:

$$p(\mathbf{x}^0) = \mathcal{N}(\mathbf{x}^0; \bar{\mathbf{x}}^0, \Sigma^0) \quad (2.18)$$

where $\mathcal{N}(\mathbf{x}^0; \bar{\mathbf{x}}^0, \Sigma^0)$ is a Gaussian PDF with mean $\bar{\mathbf{x}}^0$ and covariance matrix Σ^0 evaluated at $\mathbf{x}^0$.

Equations (2.16) and (2.17) can be written as:

$$p(\mathbf{x}^{k+1} | \mathbf{x}^k) = \mathcal{N}(\mathbf{x}^{k+1}; \mathbf{F}^{k+1}\mathbf{x}^k, \mathbf{Q}^{k+1}) \quad (2.19)$$

$$p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) = \mathcal{N}(\mathbf{z}^{k+1}; \mathbf{H}^{k+1}\mathbf{x}^{k+1}, \mathbf{R}^{k+1}) \quad (2.20)$$

The recursive analytical solution to the system defined by (2.18), (2.19) and (2.20) is known and is given by the Kalman filter [13, 84, 88, 102, 105, 179].

Assume the posterior PDF at time k is

$$p(\mathbf{x}^k | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^k; \bar{\mathbf{x}}^k, \Sigma^k) \quad (2.21)$$

In the prediction step, the predicted density is calculated as

$$p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \Sigma^{k+1|k}) \quad (2.22)$$

where

$$\bar{\mathbf{x}}^{k+1|k} = \mathbf{F}^{k+1} \bar{\mathbf{x}}^k \quad (2.23)$$

$$\Sigma^{k+1|k} = \mathbf{F}^{k+1} \Sigma^k \left(\mathbf{F}^{k+1} \right)^T + \mathbf{Q}^{k+1} \quad (2.24)$$

In the update step, the posterior PDF at time $k + 1$ is calculated as

$$p\left(\mathbf{x}^{k+1} \mid \mathbf{z}^{1:k+1}\right) = \mathcal{N}\left(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1}, \Sigma^{k+1}\right) \quad (2.25)$$

where

$$\bar{\mathbf{x}}^{k+1} = \bar{\mathbf{x}}^{k+1|k} + \Psi^{k+1} \left(\mathbf{S}^{k+1} \right)^{-1} \left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1} \right) \quad (2.26)$$

$$\Sigma^{k+1} = \Sigma^{k+1|k} - \Psi^{k+1} \left(\mathbf{S}^{k+1} \right)^{-1} \left(\Psi^{k+1} \right)^T \quad (2.27)$$

and

$$\bar{\mathbf{z}}^{k+1} = \mathbf{H}^{k+1} \bar{\mathbf{x}}^{k+1|k} \quad (2.28)$$

$$\Psi^{k+1} = \Sigma^{k+1|k} \left(\mathbf{H}^{k+1} \right)^T \quad (2.29)$$

$$\mathbf{S}^{k+1} = \mathbf{H}^{k+1} \Sigma^{k+1|k} \left(\mathbf{H}^{k+1} \right)^T + \mathbf{R}^{k+1} \quad (2.30)$$

It should be noted that $\bar{\mathbf{x}}^{k+1}$, given by (2.26), is the MMSE estimator as it is the posterior mean [13].

2.5 Particle filters for nonlinear/non-Gaussian models

A nonlinear/non-Gaussian system is defined by the following state space equations [9]:

$$\mathbf{x}^{k+1} = \mathbf{f}^{k+1} \left(\mathbf{x}^k, \mathbf{w}^{k+1} \right) \quad (2.31)$$

$$\mathbf{z}^{k+1} = \mathbf{h}^{k+1} \left(\mathbf{x}^{k+1}, \mathbf{v}^{k+1} \right) \quad (2.32)$$

where $\mathbf{f}^{k+1}$ and $\mathbf{h}^{k+1}$ are possibly nonlinear functions, $\mathbf{w}^{k+1}$ is the process noise and $\mathbf{v}^{k+1}$ is the measurement noise. The noise vectors at each time step are assumed to be independent. It should be mentioned that the system is nonlinear/non-Gaussian if $\mathbf{f}^k$ or $\mathbf{h}^k$ is not a linear function for some k or, $\mathbf{w}^k$ or $\mathbf{v}^k$ is not Gaussian distributed for some k . In this case, the posterior PDF does not have a closed-form expression in general and one must resort to approximations. Two kinds of approximations are common:

1. Approximation of the posterior PDF at each time step assuming it has a specific form, which is usually Gaussian, propagating its sufficient statistics.

In other words, for the Gaussian case, the algorithm only approximates the first two moments of the posterior: its mean and covariance matrix.

2. Approximation of the posterior PDF without assuming a concrete form.

Approximations of type 1 usually have a much lower computational burden than approximations of type 2. They are often based on Kalman filtering for nonlinear/non-Gaussian models, which will be addressed in Section 2.6, although other approaches are also possible [113]. However, in some cases their performance is poor, for example, when the posterior is multimodal because a multimodal distribution cannot be properly approximated by a unimodal distribution in general, although a mixture of Gaussian PDFs can be used to improve performance in this case [13, 114].

This section deals with approximations of type 2 based on sequential Monte Carlo methods or particle filters (PFs) [31, 52, 56, 85, 169]. Particle filters approximate the posterior PDF by a set of weighted samples drawn from an importance density. An important virtue of PFs is that they provide an asymptotically exact approximation of the posterior PDF as the sample size tends to infinity and convergence results can be applied [38, 92, 93]. Deterministic grid approximations to the posterior such as the point-mass filters have not been considered in this thesis due to their less common use [28, 202] and their problems to deal with high-dimensional problems [26]. A promising approach that has not been considered here is particle flow because of its high complexity [46, 47].

In the following, the important facts about particle filtering that are needed to understand this thesis are described.

2.5.1 Sequential importance sampling

The sequential importance sampling (SIS) algorithm is a Monte Carlo (MC) method, in which one builds the importance density, draws samples from it and computes the weights of the samples recursively [126]. The idea is to represent the posterior PDF by a set of random samples with weights and compute estimates based on these samples and weights.

In a filtering problem, one is interested in approximating the posterior PDF, $p(\mathbf{x}^k | \mathbf{z}^{1:k})$, sequentially, as happened with the Kalman filter. However, SIS achieves this task in an indirect way. SIS approximates the posterior PDF of the whole trajectory, $p(\mathbf{x}^{0:k} | \mathbf{z}^{1:k})$, sequentially and, based on that approximation, the posterior PDF is built by integrating out the states at previous time instants. This is explained in the following.

Let us assume the posterior PDF of the whole trajectory is approximated by

drawing N_{par} samples $\{\mathbf{x}_i^{0:k}\}_{i=1}^{N_{par}}$ from an importance density $q(\mathbf{x}^{0:k} | \mathbf{z}^{1:k})$ such that if $p(\mathbf{x}^{0:k} | \mathbf{z}^{1:k}) > 0 \rightarrow q(\mathbf{x}^{0:k} | \mathbf{z}^{1:k}) > 0$. Then, using the principle of importance sampling:

$$p(\mathbf{x}^{0:k} | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{x}^{0:k} - \mathbf{x}_i^{0:k}) \quad (2.33)$$

where $\delta(\cdot)$ is the Dirac delta and the weight w_i^k is given by [9]

$$w_i^k \propto \frac{p(\mathbf{x}_i^{0:k} | \mathbf{z}^{1:k})}{q(\mathbf{x}_i^{0:k} | \mathbf{z}^{1:k})} \quad (2.34)$$

and

$$\sum_{i=1}^{N_{par}} w_i^k = 1 \quad (2.35)$$

It should be noted that in order to obtain the importance weights, given by (2.34), one only needs to know the posterior PDF of the whole trajectory and the importance density up to a normalising constant as the weights are normalised using (2.35).

Suppose at time step k there is a collection of samples and weights that approximates $p(\mathbf{x}^{0:k} | \mathbf{z}^{1:k})$ as in (2.33). With the reception of the new measurement $\mathbf{z}^{k+1}$ at time $k+1$, SIS obtains an approximation to $p(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k+1})$ of the form (2.33) based on the available approximation to $p(\mathbf{x}^{0:k} | \mathbf{z}^{1:k})$. Therefore, how to get the particles at time $k+1$ and their corresponding weights needs to be determined.

The particles can be obtained recursively if the importance density is chosen to factorise such that

$$q(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k+1}) = q(\mathbf{x}^{k+1} | \mathbf{x}^{0:k}, \mathbf{z}^{1:k+1}) q(\mathbf{x}^{0:k} | \mathbf{z}^{1:k}) \quad (2.36)$$

Therefore, according to (2.36), as $\mathbf{x}_i^{0:k} \sim q(\mathbf{x}^{0:k} | \mathbf{z}^{1:k})$, a sample $\mathbf{x}_i^{0:k+1}$ formed as $\mathbf{x}_i^{0:k+1} = (\mathbf{x}_i^{0:k}, \mathbf{x}_i^{k+1})$ where $\mathbf{x}_i^{k+1} \sim q(\mathbf{x}^{k+1} | \mathbf{x}_i^{0:k}, \mathbf{z}^{1:k+1})$ is a sample from $q(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k+1})$ [9]. That is, a sample from (2.36) can be drawn by augmenting an existing particle at time k , $\mathbf{x}_i^{0:k}$, with the new state $\mathbf{x}_i^{k+1}$ drawn from $q(\mathbf{x}^{k+1} | \mathbf{x}_i^{0:k}, \mathbf{z}^{1:k+1})$.

To derive the weight of the particle at time $k+1$, firstly, one should use Bayes' rule:

$$p(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k+1}) = \frac{p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k})}{p(\mathbf{z}^{k+1} | \mathbf{z}^{1:k})} \quad (2.37)$$

Applying the Markov property that comes from the prediction equation, see

(2.8):

$$\begin{aligned} p(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k+1}) &= \frac{p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{x}^k) p(\mathbf{x}^{0:k} | \mathbf{z}^{1:k})}{p(\mathbf{z}^{k+1} | \mathbf{z}^{1:k})} \\ &\propto p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{x}^k) p(\mathbf{x}^{0:k} | \mathbf{z}^{1:k}) \end{aligned} \quad (2.38)$$

Therefore, using (2.34), (2.36) and (2.38), the weight of the i th particle at time $k+1$ is

$$\begin{aligned} w_i^{k+1} &\propto \frac{p(\mathbf{x}_i^{0:k+1} | \mathbf{z}^{1:k+1})}{q(\mathbf{x}_i^{0:k+1} | \mathbf{z}^{1:k+1})} \\ &\propto \frac{p(\mathbf{z}^{k+1} | \mathbf{x}_i^{k+1}) p(\mathbf{x}_i^{k+1} | \mathbf{x}_i^k) p(\mathbf{x}_i^{0:k} | \mathbf{z}^{1:k})}{q(\mathbf{x}_i^{k+1} | \mathbf{x}_i^{0:k}, \mathbf{z}^{1:k+1}) q(\mathbf{x}_i^{0:k} | \mathbf{z}^{1:k})} \\ &\propto \frac{p(\mathbf{z}^{k+1} | \mathbf{x}_i^{k+1}) p(\mathbf{x}_i^{k+1} | \mathbf{x}_i^k)}{q(\mathbf{x}_i^{k+1} | \mathbf{x}_i^{0:k}, \mathbf{z}^{1:k+1})} w_i^k \end{aligned} \quad (2.39)$$

Then, the approximation to the posterior PDF of the whole trajectory is

$$p(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k+1}) \approx \sum_{i=1}^{N_{par}} w_i^{k+1} \delta(\mathbf{x}^{0:k+1} - \mathbf{x}_i^{0:k+1}) \quad (2.40)$$

However, in Bayesian filtering, one is interested in the filtering PDF, i.e., the posterior PDF of the state at time $k+1$, whose PF approximation can be obtained by integrating out $\mathbf{x}^{0:k}$ in (2.40):

$$\begin{aligned} p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) &= \int p(\mathbf{x}^{0:k+1} | \mathbf{z}^{1:k+1}) d\mathbf{x}^{0:k} \\ &\approx \int \sum_{i=1}^{N_{par}} w_i^{k+1} \delta(\mathbf{x}^{0:k+1} - \mathbf{x}_i^{0:k+1}) d\mathbf{x}^{0:k} \\ &\approx \sum_{i=1}^{N_{par}} w_i^{k+1} \delta(\mathbf{x}^{k+1} - \mathbf{x}_i^{k+1}) \end{aligned} \quad (2.41)$$

In addition, if the importance density is chosen such that it only depends on the previous state and current measurement

$$q(\mathbf{x}^{k+1} | \mathbf{x}^{0:k}, \mathbf{z}^{1:k+1}) = q(\mathbf{x}^{k+1} | \mathbf{x}^k, \mathbf{z}^{k+1}) \quad (2.42)$$

the trajectories of the particles can be discarded and only the particles at the current time with their weights need to be stored. The steps of the algorithm are shown

Table 2.1: SIS particle filter

- For each particle $i = 1, \dots, N_{par}$
 - Draw $\mathbf{x}_i^{k+1} \sim q(\mathbf{x}^{k+1} | \mathbf{x}_i^k, \mathbf{z}^{k+1})$.
 - Calculate the particle weight w_i^{k+1} using (2.39).
- Normalise the weights so that they sum to one.

in Table 2.1. It should be noted that the complexity of the SIS particle filter is $O(N_{par})$.

2.5.1.1 Degeneracy problem

As k increases, the variance of the importance weights $\{w_i^{k+1}\}_{i=1}^{N_{par}}$ increases if the particle filter is of the form shown in Table 2.1 [126]. Consequently, after a few time steps, many of the weights become very small and some very large and, eventually, a single weight has value one and the rest zero. The fact that the weights are uneven has a negative effect as it is usually interpreted as if the effective number of particles were lower. This is the so-called degeneracy or depletion problem. This implies that a particle filter as the one shown in Table 2.1 does not work in the long run.

A measure of the degeneracy of a particle filter is given by the effective sampling size N_{eff} . There are several ways of calculating N_{eff} in the literature such as [9, 85, 112] but all of them are a decreasing function of the variance of the weights and a reasonable approximation to all of them is

$$N_{eff} \approx \frac{1}{\sum_{i=1}^{N_{par}} (w_i^{k+1})^2} \quad (2.43)$$

Several techniques have been proposed to overcome the degeneracy problem such as pruning, enrichment and resampling [126]. Resampling, first proposed in [78], is the most widely accepted of these methods in the engineering community

Resampling consists of the elimination of particles with low importance weights and the replication of particles with high importance weights. That is, instead of representing the posterior by $\{(\mathbf{x}_i^{k+1}, w_i^{k+1})\}_{i=1}^{N_{par}}$, one uses another PF representation $\{(\mathbf{x}_i^{*k+1}, 1/N_{par})\}_{i=1}^{N_{par}}$ with uniform weights where the new random samples $\{\mathbf{x}_i^{*k+1}\}_{i=1}^{N_{par}}$ are selected from the set $\{\mathbf{x}_i^{k+1}\}_{i=1}^{N_{par}}$. There are many alternatives to implement a resampling step. In this thesis, systematic resampling is used due to

Table 2.2: Systematic resampling algorithm

- Construct the cumulative density function (CDF) of w_i^{k+1} over $i = 1, \dots, N_{par}$:
 $(c_1, \dots, c_{N_{par}})$
 - Assign $c_1 = w_1^{k+1}$
 - For each particle $i = 2, \dots, N_{par}$
 - * $c_i = c_{i-1} + w_i^{k+1}$
- Start at the bottom of the CDF: $j = 1$.
- Draw a starting point: $u_1 \sim U[0, 1/N_{par}]$ (where $U[a, b]$ denotes a uniform PDF in the interval $[a, b]$)
- For each particle $i = 1, \dots, N_{par}$
 - Move along the CDF: $u_i = u_1 + N_{par}^{-1}(i - 1)$
 - While $u_i > c_j$
 - * $j = j + 1$
 - Set $\mathbf{x}_i^{*k+1} = \mathbf{x}_j^{k+1}$

its broad use and low computational complexity [169]. Its steps are shown in Table 2.2.

A problem of resampling is that it implies a loss of diversity among the particles as in the resultant particle approximation there will be repeated samples. This problem is referred to as sample impoverishment. In addition, it introduces a practical problem. Resampling prevents the particle filter from being run in parallel, i.e., propagating a particle and calculating its weight independently of the rest of the particles for all time steps. Therefore, the resampling step is considered to be a bottleneck in the execution time [37] although some parallelisable resampling techniques have been proposed [25, 135].

2.5.1.2 Choice of importance density

The choice of importance density is arguably the most important design aspect that determines the performance of a particle filter. The so-called optimal importance density is the one that minimises the variance of the unnormalised importance weight of the i th particle at time $k + 1$ conditional upon the particle trajectory $\mathbf{x}_i^{0:k}$ and the measurements $\mathbf{z}^{1:k+1}$. It has been shown to be [55]

$$\begin{aligned}
 q_{opt}(\mathbf{x}^{k+1} | \mathbf{x}_i^k, \mathbf{z}^{k+1}) &= p(\mathbf{x}^{k+1} | \mathbf{x}_i^k, \mathbf{z}^{k+1}) \\
 &= \frac{p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{x}_i^k)}{p(\mathbf{z}^{k+1} | \mathbf{x}_i^k)}
 \end{aligned} \tag{2.44}$$

The weight of the particle is calculated by substituting (2.44) into (2.39):

$$w_i^{k+1} \propto p(\mathbf{z}^{k+1} | \mathbf{x}_i^k) w_i^k \tag{2.45}$$

It should be noted that in this case, the importance weights, given by (2.45), can be calculated before the particles are propagated. This implies that the variance of the weights is unaffected by the sampling and that resampling can be done before the propagation of the particles. Other alternatives will add variance to the weights and lower the effective sampling size, which is given by (2.43). Therefore, (2.44) is optimal in the sense that it maximises the effective sampling size [85].

In order to use the optimal importance density, it has to be possible to sample from $p(\mathbf{x}^{k+1} | \mathbf{x}_i^k, \mathbf{z}^{k+1})$ and to evaluate

$$p(\mathbf{z}^{k+1} | \mathbf{x}_i^k) = \int p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{x}_i^k) d\mathbf{x}^{k+1} \tag{2.46}$$

up to a normalising constant. In the general case, either of these is not possible and another importance density should be used.

A widely used importance density is the transitional prior $q(\mathbf{x}^{k+1} | \mathbf{x}_i^k) = p(\mathbf{x}^{k+1} | \mathbf{x}_i^k)$. A particle filter with this importance density and a resampling algorithm performed every time step is called sampling importance resampling (SIR) filter or bootstrap filter and was the first working particle filter [78]. The steps of this algorithm are shown in Table 2.3. However, the performance of a particle filter with this importance density plummets when the likelihood $p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1})$ is narrow compared to $p(\mathbf{x}^{k+1} | \mathbf{x}_i^k)$, as only a few particles will have a non-negligible weight and the variance of the weights will be high [86]. This particle filter is generally outperformed by the auxiliary particle filter to be described in Section 2.5.3.

2.5.2 Marginal particle filter

Sequential importance sampling samples the whole trajectory of the targets $\mathbf{x}^{0:k}$ and then discards the parts of the particles that correspond to previous time steps to approximate the posterior as explained in the previous section. Another approach

Table 2.3: SIR particle filter

- For each particle $i = 1, \dots, N_{par}$
 - Draw $\mathbf{x}_i^{k+1} \sim p(\mathbf{x}^{k+1} | \mathbf{x}_i^k)$
 - Calculate $w_i^{k+1} = p(\mathbf{z}^{k+1} | \mathbf{x}_i^{k+1})$
- Normalise w_i^{k+1} to sum to one over $i = 1, \dots, N_{par}$
- Resample using the algorithm in Table 2.2.

is to sample directly the posterior PDF [110]. Assume the posterior at time k is:

$$p(\mathbf{x}^k | \mathbf{z}^{1:k}) = \sum_{j=1}^{N_{par}} w_j^k \delta(\mathbf{x}^k - \mathbf{x}_j^k) \quad (2.47)$$

Therefore, using the Chapman-Kolmogorov equation and Bayes' rule, the posterior at time $k+1$ is

$$p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) \sum_{j=1}^{N_{par}} w_j^k p(\mathbf{x}^{k+1} | \mathbf{x}_j^k) \quad (2.48)$$

While one is free to choose any importance density that has appropriate support, it is convenient to assume that the proposal takes a similar form to (2.48) [110]:

$$q(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) = \sum_{j=1}^{N_{par}} w_j^k q(\mathbf{x}^{k+1} | \mathbf{x}_j^k, \mathbf{z}^k) \quad (2.49)$$

Therefore, the weight of the i th particle at time $k+1$ is given by

$$w_i^{k+1} \propto \frac{p(\mathbf{z}^{k+1} | \mathbf{x}_i^{k+1}) \sum_{j=1}^{N_{par}} w_j^k p(\mathbf{x}_i^{k+1} | \mathbf{x}_j^k)}{\sum_{j=1}^{N_{par}} w_j^k q(\mathbf{x}_i^{k+1} | \mathbf{x}_j^k, \mathbf{z}^k)} \quad (2.50)$$

The drawback of this method is that its complexity is $O(N_{par}^2)$ as, for each particle, the sums with N_{par} terms in (2.50) have to be evaluated. However, the variance of the importance weights is lower than for a particle filter that samples the whole trajectory as shown in [110]. This suggests that its performance should be higher than SIS performance for a given number of particles.

2.5.3 Auxiliary particle filter

The auxiliary particle filter was originally proposed in [165] as a way to improve the performance of sequential Monte Carlo algorithms in models with peaked likelihoods when sampling from the optimal importance density is not possible or very demanding in terms of computation. The aim is to obtain samples from (2.48). The auxiliary particle filter achieves this objective indirectly by sampling in a higher dimension.

The auxiliary particle filter performs sampling in a higher dimension approximating the posterior PDF

$$p(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) w_a^k p(\mathbf{x}^{k+1} | \mathbf{x}_a^k) \quad (2.51)$$

where $a \in \{1, \dots, N_{par}\}$ is the auxiliary random variable that is an index on the mixture in (2.48). It should be noted that $p(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1}) \geq 0$ and that summing over the discrete auxiliary variable a in (2.51), one gets (2.48):

$$\sum_{a=1}^{N_{par}} p(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{x}^{k+1}) \sum_{a=1}^{N_{par}} w_a^k p(\mathbf{x}^{k+1} | \mathbf{x}_a^k) \quad (2.52)$$

and therefore

$$\sum_{a=1}^{N_{par}} p(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1}) = p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1}) \quad (2.53)$$

Otherwise, the definition of the auxiliary variable and the joint density would not be sound.

The auxiliary particle filter operates by obtaining samples $\{(\mathbf{x}_i^{k+1}, a_i)\}_{i=1}^{N_{par}}$ of the joint density $p(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1})$. Then, it omits¹ a in the pair $(\mathbf{x}^{k+1}, a)$ to produce samples $\{\mathbf{x}_i^{k+1}\}_{i=1}^{N_{par}}$ of the posterior $p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k+1})$. As sampling is realised in the joint space, the importance density used to draw the samples $\{(\mathbf{x}_i^{k+1}, a_i)\}_{i=1}^{N_{par}}$ needs to be of the form $q(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1})$. A common way to define the importance density is [169]

$$q(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \boldsymbol{\mu}_a^{k+1}) p(\mathbf{x}^{k+1} | \mathbf{x}_a^k) w_a^k \quad (2.54)$$

where $\boldsymbol{\mu}_a^{k+1}$ is some characterisation of $\mathbf{x}^{k+1}$ given $\mathbf{x}_a^k$, such as the predicted mean $\boldsymbol{\mu}_a^{k+1} = \mathbb{E}[\mathbf{x}^{k+1} | \mathbf{x}_a^k]$ or a sample $\boldsymbol{\mu}_a^{k+1} \sim p(\mathbf{x}^{k+1} | \mathbf{x}_a^k)$.

¹The reason why removing a from the samples of the joint density produces samples of the marginal density is the same as why the SIS particle filter produces samples of the posterior by dropping the particle trajectories. This was explained through (2.40) and (2.41).

One can write

$$q(\mathbf{x}^{k+1}, a | \mathbf{z}^{1:k+1}) = q(a | \mathbf{z}^{1:k+1}) q(\mathbf{x}^{k+1} | a, \mathbf{z}^{1:k+1}) \quad (2.55)$$

and defining

$$q(\mathbf{x}^{k+1} | a, \mathbf{z}^{1:k+1}) \triangleq p(\mathbf{x}^{k+1} | \mathbf{x}_a^k) \quad (2.56)$$

it follows from (2.54) and (2.55) that

$$b_a \triangleq q(a | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \boldsymbol{\mu}_a^{k+1}) w_a^k \quad (2.57)$$

where $\{b_a\}_{a=1}^{N_{par}}$ are referred to as the first-stage weights [165].

The final weight or second-stage weight of the i th particle is

$$\begin{aligned} w_i^{k+1} &\propto \frac{p(\mathbf{x}_i^{k+1}, a_i | \mathbf{z}^{1:k+1})}{q(\mathbf{x}_i^{k+1}, a_i | \mathbf{z}^{1:k+1})} \\ &\propto \frac{p(\mathbf{z}^{k+1} | \mathbf{x}_i^{k+1}) p(\mathbf{x}_i^{k+1} | \mathbf{x}_{a_i}^k) w_{a_i}^k}{p(\mathbf{z}^{k+1} | \boldsymbol{\mu}_{a_i}^{k+1}) p(\mathbf{x}_i^{k+1} | \mathbf{x}_{a_i}^k) w_{a_i}^k} \\ &\propto \frac{p(\mathbf{z}^{k+1} | \mathbf{x}_i^{k+1})}{p(\mathbf{z}^{k+1} | \boldsymbol{\mu}_{a_i}^{k+1})} \end{aligned} \quad (2.58)$$

To sum up, the steps of the algorithm are given in Table 2.4. It should be noted that the computational complexity of the algorithm is $O(N_{par})$. It should also be mentioned that the auxiliary particle filter has also been interpreted as a sampling importance resampling algorithm in [31, 98]. This is of use to understand some of its theoretical properties such as convergence results [85].

2.5.4 Rao-Blackwellisation

It is known that the performance of a particle filter severely deteriorates when the dimension of the state increases [45]. Therefore, it is desired to reduce the dimension of the state represented by the particles whenever is possible. The approach presented here has taken several names such as Rao-Blackwellised particle filter or marginalised particle filter (not to be confused with the marginal particle filter addressed in Section 2.5.2) [6, 141, 157, 175]. The former name is the one that will be used throughout this thesis.

Rao-Blackwellisation can be applied when the state can be partitioned as $\mathbf{x}^k =$

Table 2.4: Auxiliary particle filter steps

- For each particle $i = 1, \dots, N_{par}$
 - Calculate $\boldsymbol{\mu}_i^{k+1} = \mathbb{E} \left[\mathbf{x}^{k+1} \mid \mathbf{x}_i^k \right]$ or $\boldsymbol{\mu}_i^{k+1} \sim p \left(\mathbf{x}^{k+1} \mid \mathbf{x}_i^k \right)$.
 - Calculate b_i using (2.57).
- Normalise the first stage weights $(b_1, \dots, b_{N_{par}})$ so that they sum to one.
- For each particle $i = 1, \dots, N_{par}$
 - Sample an index a from the distribution defined by $(b_1, \dots, b_{N_{par}})$.
 - Draw $\mathbf{x}_i^{k+1} \sim p \left(\mathbf{x}^{k+1} \mid \mathbf{x}_a^k \right)$.
 - Assign weight w_i^{k+1} using (2.58).
- Normalise $\left(w_i^{k+1} \right)_{i=1}^{N_{par}}$ so that they sum to one.

$\left[\left(\mathbf{x}_l^k \right)^T, \left(\mathbf{x}_n^k \right)^T \right]^T$ such that

$$p \left(\mathbf{x}_l^k, \mathbf{x}_n^{0:k} \mid \mathbf{z}^{1:k} \right) = p \left(\mathbf{x}_l^k \mid \mathbf{x}_n^{0:k}, \mathbf{z}^{1:k} \right) p \left(\mathbf{x}_n^{0:k} \mid \mathbf{z}^{1:k} \right) \quad (2.59)$$

where the conditional PDF $p \left(\mathbf{x}_l^k \mid \mathbf{x}_n^{0:k}, \mathbf{z}^{1:k} \right)$ admits a closed-form expression for example because the relations among $\mathbf{x}_l^k$, $\mathbf{x}_n^{0:k}$ and $\mathbf{z}^{1:k}$ are linear and Gaussian. In this case, this conditional PDF can be calculated by means of a Kalman filter. Therefore, Rao-Blackwellisation consists of approximating $p \left(\mathbf{x}_n^{0:k} \mid \mathbf{z}^{1:k} \right)$ using a particle filter and calculating $p \left(\mathbf{x}_l^k \mid \mathbf{x}_n^{0:k}, \mathbf{z}^{1:k} \right)$ analytically using a Kalman filter for each particle. As a result, the dimension of the state represented by the particles is lower and performance increases.

The posterior PDF at time k can be approximated dropping the particle states from time 0 to $k - 1$. If a Kalman filter is used to calculate the linear part, the posterior at time k is approximated as

$$p \left(\mathbf{x}_l^k, \mathbf{x}_n^k \mid \mathbf{z}^{1:k} \right) \approx \sum_{i=1}^{N_{par}} w_i^k \delta \left(\mathbf{x}_n^k - \mathbf{x}_{n,i}^k \right) \mathcal{N} \left(\mathbf{x}_l^k; \bar{\mathbf{x}}_{l,i}^k, \boldsymbol{\Sigma}_{l,i}^k \right) \quad (2.60)$$

where $\mathbf{x}_{n,i}^k$ is the i th particle of the nonlinear part at time k with weight w_i^k , and $\bar{\mathbf{x}}_{l,i}^k$ and $\boldsymbol{\Sigma}_{l,i}^k$ are the mean and the covariance matrix of the output of the Kalman filter applied to $\mathbf{x}_{n,i}^{0:k}$ and $\mathbf{z}^{1:k}$, respectively. Therefore, the posterior PDF is characterised by the set of the so-called Rao-Blackwellised particles $\left\{ \left(\mathbf{x}_{n,i}^k, \bar{\mathbf{x}}_{l,i}^k, \boldsymbol{\Sigma}_{l,i}^k \right) \right\}_{i=1}^{N_{par}}$, which

are augmented particles that include the particles themselves plus the means and covariance matrices of the Kalman filters applied to them, with their corresponding weights w_i^k .

The prior at time $k + 1$ is

$$p(\mathbf{x}_l^{k+1}, \mathbf{x}_n^{k+1} | \mathbf{z}^{1:k}) \approx \int p(\mathbf{x}_l^{k+1}, \mathbf{x}_n^{k+1} | \mathbf{x}_l^k, \mathbf{x}_n^k) \cdot \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{x}_n^k - \mathbf{x}_{n,i}^k) \mathcal{N}(\mathbf{x}_l^k; \bar{\mathbf{x}}_{l,i}^k, \Sigma_{l,i}^k) d\mathbf{x}_n^k d\mathbf{x}_l^k \quad (2.61)$$

$$= \sum_{i=1}^{N_{par}} w_i^k \int p(\mathbf{x}_l^{k+1}, \mathbf{x}_n^{k+1} | \mathbf{x}_l^k, \mathbf{x}_{n,i}^k) \mathcal{N}(\mathbf{x}_l^k; \bar{\mathbf{x}}_{l,i}^k, \Sigma_{l,i}^k) d\mathbf{x}_l^k \quad (2.62)$$

Using a slight abuse of notation, in this thesis, (2.62) is sometimes written as

$$p(\mathbf{x}_l^{k+1}, \mathbf{x}_n^{k+1} | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} w_i^k p(\mathbf{x}_l^{k+1}, \mathbf{x}_n^{k+1} | \mathbf{x}_{n,i}^k, \bar{\mathbf{x}}_{l,i}^k, \Sigma_{l,i}^k) \quad (2.63)$$

where

$$p(\mathbf{x}_l^{k+1}, \mathbf{x}_n^{k+1} | \mathbf{x}_{n,i}^k, \bar{\mathbf{x}}_{l,i}^k, \Sigma_{l,i}^k) = \int p(\mathbf{x}_l^{k+1}, \mathbf{x}_n^{k+1} | \mathbf{x}_l^k, \mathbf{x}_{n,i}^k) \mathcal{N}(\mathbf{x}_l^k; \bar{\mathbf{x}}_{l,i}^k, \Sigma_{l,i}^k) d\mathbf{x}_l^k \quad (2.64)$$

There are several reasons for doing this. One is that it simplifies notation. Instead of carrying the whole particle trajectories in all the equations, one only needs to carry the mean and covariance matrix of the Kalman filter. Another advantage is that from a practitioner's point of view, this way it becomes clearer that the particle trajectories do not need to be stored. The last reason for using this notation is that it is not confusing, i.e., every time a PDF is conditioned on a mean and covariance matrix, it represents an integration of the form of (2.64).

2.6 Kalman filter for nonlinear/non-Gaussian models

In Section 2.4, the Kalman filter (KF) was introduced as the closed-form solution of the posterior PDF when the system is linear and Gaussian. However, as pointed out in [100, 191], Kalman's original derivation [102] was not proposed as the recursive solution of the Chapman–Kolmogorov equation and Bayes' rule, see (2.12) and (2.15), when the system is linear and Gaussian.

In fact, Kalman filtering can be applied to nonlinear/non-Gaussian models. In

this case, the Kalman filter provides a recursion that is used to approximate the first two moments of the posterior PDF. In the prediction step, the first two moments of the prior are approximated based on the Chapman-Kolmogorov equation using the transition PDF and the posterior PDF at the previous time. In the update step, the first two moments of the posterior PDF are approximated using a linear relation in the measurement at the current time. This is explained in the following.

For clarity of presentation, additive noise will be assumed in the process and measurement equations as in [7, 95]. Therefore,

$$\mathbf{x}^{k+1} = \mathbf{f}^{k+1}(\mathbf{x}^k) + \mathbf{w}^{k+1} \quad (2.65)$$

$$\mathbf{z}^{k+1} = \mathbf{h}^{k+1}(\mathbf{x}^{k+1}) + \mathbf{v}^{k+1} \quad (2.66)$$

It is also assumed that the first two moments of $\mathbf{w}^k$ and $\mathbf{v}^k$ are known: $\mathbb{E}[\mathbf{w}^k] = 0$, $\text{cov}[\mathbf{w}^k] = \mathbf{Q}^k$, $\mathbb{E}[\mathbf{v}^k] = 0$ and $\text{cov}[\mathbf{v}^k] = \mathbf{R}^k$. The Kalman filter recursively approximates the first two moments of the posterior via two phases: prediction and update. It is assumed that the approximation of the posterior PDF at time k is $p(\mathbf{x}^k | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^k; \bar{\mathbf{x}}^k, \Sigma^k)$.

Prediction:

The first two moments of $\mathbf{x}^{k+1}$ given the past measurements $\mathbf{z}^{1:k}$ are calculated based on the Gaussian assumption of the posterior at time k .

$$\begin{aligned} \bar{\mathbf{x}}^{k+1|k} &= \mathbb{E}[\mathbf{x}^{k+1} | \mathbf{z}^{1:k}] = \int (\mathbf{f}^{k+1}(\mathbf{x}^k) + \mathbf{w}^{k+1}) p(\mathbf{x}^k | \mathbf{z}^{1:k}) p(\mathbf{w}^{k+1}) d\mathbf{x}^k d\mathbf{w}^{k+1} \\ &= \int \mathbf{f}^{k+1}(\mathbf{x}^k) p(\mathbf{x}^k | \mathbf{z}^{1:k}) d\mathbf{x}^k \\ &= \int \mathbf{f}^{k+1}(\mathbf{x}^k) \mathcal{N}(\mathbf{x}^k; \bar{\mathbf{x}}^k, \Sigma^k) d\mathbf{x}^k \end{aligned} \quad (2.67)$$

$$\begin{aligned} \Sigma^{k+1|k} &= \text{cov}[\mathbf{x}^{k+1} | \mathbf{z}^{1:k}] = \int (\mathbf{f}^{k+1}(\mathbf{x}^k) + \mathbf{w}^{k+1} - \bar{\mathbf{x}}^{k+1|k}) \\ &\quad \cdot (\mathbf{f}^{k+1}(\mathbf{x}^k) + \mathbf{w}^{k+1} - \bar{\mathbf{x}}^{k+1|k})^T \\ &\quad \cdot p(\mathbf{x}^k | \mathbf{z}^{1:k}) p(\mathbf{w}^{k+1}) d\mathbf{x}^k d\mathbf{w}^{k+1} \\ &= \int \mathbf{f}^{k+1}(\mathbf{x}^k) (\mathbf{f}^{k+1}(\mathbf{x}^k))^T p(\mathbf{x}^k | \mathbf{z}^{1:k}) d\mathbf{x}^k \\ &\quad - \bar{\mathbf{x}}^{k+1|k} (\bar{\mathbf{x}}^{k+1|k})^T + \mathbf{Q}^{k+1} \end{aligned}$$

$$\begin{aligned}
 &= \int \mathbf{f}^{k+1}(\mathbf{x}^k) \left(\mathbf{f}^{k+1}(\mathbf{x}^k) \right)^T \mathcal{N}(\mathbf{x}^k; \bar{\mathbf{x}}^k, \Sigma^k) d\mathbf{x}^k \\
 &\quad - \bar{\mathbf{x}}^{k+1|k} \left(\bar{\mathbf{x}}^{k+1|k} \right)^T + \mathbf{Q}^{k+1}
 \end{aligned} \tag{2.68}$$

Update:

The first two moments of $\mathbf{x}^{k+1}$ given the measurements $\mathbf{z}^{1:k+1}$ are approximated based on a Gaussian assumption of the prior at time $k+1$, i.e., $p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \Sigma^{k+1|k})$. There are two possible equivalent interpretations about what Kalman filtering does at this step. One is that it assumes that the probability density $p\left(\left[\left(\mathbf{x}^{k+1}\right)^T, \left(\mathbf{z}^{k+1}\right)^T\right]^T | \mathbf{z}^{1:k}\right)$ is Gaussian and then Bayes' rule is applied to give a Gaussian approximation of the posterior [7, 95]. The other interpretation is that it applies the linear MMSE estimator of $\mathbf{x}^{k+1}$ conditional on the measurements $\mathbf{z}^{1:k+1}$ based on the Gaussian assumption of the prior at time $k+1$, $p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \Sigma^{k+1|k})$. Then, it approximates the first two moments of the posterior by the linear MMSE estimator and its mean square error matrix [105]. Therefore, the Kalman filter update stage does not make a direct use of Bayes' rule, see (2.15). This can result in a dramatic decrease in performance when the measurement equation is highly nonlinear as will be explained in Chapter 3.

The updated mean of the KF provides the linear MMSE estimator $\bar{\mathbf{x}}_u^{k+1}$ of $\mathbf{x}^{k+1}$ given $\mathbf{z}^{1:k+1}$ assuming that $p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \Sigma^{k+1|k})$ [105]:

$$\bar{\mathbf{x}}_u^{k+1} = \bar{\mathbf{x}}^{k+1|k} + \Psi^{k+1} \left(\mathbf{S}^{k+1} \right)^{-1} \left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1|k} \right) \tag{2.69}$$

where

$$\bar{\mathbf{z}}^{k+1|k} = \mathbb{E} \left[\mathbf{z}^{k+1} | \mathbf{z}^{1:k} \right] \tag{2.70}$$

$$= \int \mathbb{E} \left[\mathbf{z}^{k+1} | \mathbf{x}^{k+1} \right] p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) d\mathbf{x}^{k+1} \tag{2.71}$$

$$= \int \mathbb{E} \left[\mathbf{z}^{k+1} | \mathbf{x}^{k+1} \right] \mathcal{N}(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \Sigma^{k+1|k}) d\mathbf{x}^{k+1} \tag{2.72}$$

$$= \int \mathbf{h}^{k+1}(\mathbf{x}^{k+1}) \mathcal{N}(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \Sigma^{k+1|k}) d\mathbf{x}^{k+1} \tag{2.73}$$

$$\mathbf{S}^{k+1} = \text{cov} \left[\mathbf{z}^{k+1} | \mathbf{z}^{1:k} \right] \tag{2.74}$$

$$= \int \mathbb{E} \left[\left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1|k} \right) \left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1|k} \right)^T | \mathbf{x}^{k+1} \right] p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) d\mathbf{x}^{k+1} \tag{2.75}$$

$$\begin{aligned}
 &= \int \mathbb{E} \left[\left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1|k} \right) \left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1|k} \right)^T \middle| \mathbf{x}^{k+1} \right] \\
 &\quad \cdot \mathcal{N} \left(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \boldsymbol{\Sigma}^{k+1|k} \right) d\mathbf{x}^{k+1}
 \end{aligned} \tag{2.76}$$

$$\begin{aligned}
 &= \int \left(\mathbf{h}^{k+1} \left(\mathbf{x}^{k+1} \right) - \bar{\mathbf{z}}^{k+1|k} \right) \left(\mathbf{h}^{k+1} \left(\mathbf{x}^{k+1} \right) - \bar{\mathbf{z}}^{k+1|k} \right)^T \\
 &\quad \cdot \mathcal{N} \left(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \boldsymbol{\Sigma}^{k+1|k} \right) d\mathbf{x}^{k+1} + \mathbf{R}^{k+1}
 \end{aligned} \tag{2.77}$$

$$\boldsymbol{\Psi}^{k+1} = \text{cov} \left[\mathbf{x}^{k+1}, \mathbf{z}^{k+1} \middle| \mathbf{z}^{1:k} \right] \tag{2.78}$$

$$\begin{aligned}
 &= \int \mathbb{E} \left[\left(\mathbf{x}^{k+1} - \bar{\mathbf{x}}^{k+1|k} \right) \left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1|k} \right)^T \middle| \mathbf{x}^{k+1} \right] p \left(\mathbf{x}^{k+1} \middle| \mathbf{z}^{1:k} \right) d\mathbf{x}^{k+1}
 \end{aligned} \tag{2.79}$$

$$\begin{aligned}
 &= \int \mathbb{E} \left[\left(\mathbf{x}^{k+1} - \bar{\mathbf{x}}^{k+1|k} \right) \left(\mathbf{z}^{k+1} - \bar{\mathbf{z}}^{k+1|k} \right)^T \middle| \mathbf{x}^{k+1} \right] \\
 &\quad \cdot \mathcal{N} \left(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \boldsymbol{\Sigma}^{k+1|k} \right) d\mathbf{x}^{k+1}
 \end{aligned} \tag{2.80}$$

$$\begin{aligned}
 &= \int \left(\mathbf{x}^{k+1} - \bar{\mathbf{x}}^{k+1|k} \right) \left(\mathbf{h}^{k+1} \left(\mathbf{x}^{k+1} \right) - \bar{\mathbf{z}}^{k+1|k} \right)^T \\
 &\quad \cdot \mathcal{N} \left(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \boldsymbol{\Sigma}^{k+1|k} \right) d\mathbf{x}^{k+1}
 \end{aligned} \tag{2.81}$$

The updated covariance matrix given by the Kalman filter, $\boldsymbol{\Sigma}_u^{k+1}$, is

$$\boldsymbol{\Sigma}_u^{k+1} = \boldsymbol{\Sigma}^{k+1|k} - \boldsymbol{\Psi}^{k+1} \left(\mathbf{S}^{k+1} \right)^{-1} \left(\boldsymbol{\Psi}^{k+1} \right)^T \tag{2.82}$$

As shown in [105], $\boldsymbol{\Sigma}_u^{k+1}$ is the mean square error matrix averaged over all possible measurements at time $k+1$ assuming $p \left(\mathbf{x}^{k+1} \middle| \mathbf{z}^{1:k} \right) = \mathcal{N} \left(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \boldsymbol{\Sigma}^{k+1|k} \right)$:

$$\begin{aligned}
 \boldsymbol{\Sigma}_u^{k+1} &= \mathbb{E} \left[\left(\mathbf{x}^{k+1} - \bar{\mathbf{x}}_u^{k+1} \right) \left(\mathbf{x}^{k+1} - \bar{\mathbf{x}}_u^{k+1} \right)^T \middle| \mathbf{z}^{1:k} \right] \\
 &= \int \left(\mathbf{x}^{k+1} - \bar{\mathbf{x}}_u^{k+1} \right) \left(\mathbf{x}^{k+1} - \bar{\mathbf{x}}_u^{k+1} \right)^T \\
 &\quad \cdot \mathcal{N} \left(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \boldsymbol{\Sigma}^{k+1|k} \right) p \left(\mathbf{z}^{k+1} \middle| \mathbf{x}^{k+1} \right) d\mathbf{x}^{k+1} d\mathbf{z}^{k+1}
 \end{aligned} \tag{2.83}$$

where it should be noted that $\bar{\mathbf{x}}_u^{k+1}$ depends on $\mathbf{z}^{k+1}$.

The main drawbacks of Kalman filtering in nonlinear/non-Gaussian systems are:

- It does not apply Bayes' rule directly: Its update phase can be interpreted as the approximation of the first two moments of the posterior by the linear MMSE estimator with its corresponding mean square error matrix averaged over all possible measurements. It should be noted that the Kalman filter update equations can be regarded as the application of Bayes' rule if the pro-

bability density $p\left(\left[\left(\mathbf{x}^{k+1}\right)^T, \left(\mathbf{z}^{k+1}\right)^T\right]^T \mid \mathbf{z}^{1:k}\right)$ is approximated as a Gaussian [95]. However, it does not use Bayes' rule directly unless the system is linear and Gaussian, as in this case, the Kalman filter update equations are equivalent to solving Bayes' rule.

- It is rarely tractable: The integrals that appear above do not have a closed-form solution in general and must be approximated. However, in some cases a closed-form solution exists, e.g., when the system involves polynomial functions assuming that the prior is Gaussian [127].

In the following, several widely used approximations to the Kalman filter recursion will be presented: the extended Kalman filter, the unscented Kalman filter and the cubature Kalman filter. Other approximations to the Kalman filter recursion that are not considered in this thesis are linear regression KF (LRKF) [122] and the Gauss-Hermite filter [95] among others [123, 205].

2.6.1 Approximation 1: extended Kalman filter

The extended Kalman filter (EKF) is based on the local linearisation of the equations (2.65) and (2.66) using a first order Taylor series expansion. The posterior PDF is assumed to be Gaussian $p\left(\mathbf{x}^k \mid \mathbf{z}^{1:k}\right) = \mathcal{N}\left(\mathbf{x}^k; \bar{\mathbf{x}}^k, \Sigma^k\right)$. Therefore, [13]

$$\bar{\mathbf{x}}^{k+1|k} = \mathbf{f}^{k+1}\left(\bar{\mathbf{x}}^k\right) \quad (2.84)$$

$$\Sigma^{k+1|k} = \mathbf{Q}^{k+1} + \hat{\mathbf{F}}^{k+1} \Sigma^k \left(\hat{\mathbf{F}}^{k+1}\right)^T \quad (2.85)$$

$$\bar{\mathbf{x}}_u^{k+1} = \bar{\mathbf{x}}^{k+1|k} + \Psi^{k+1} \left(\mathbf{S}^{k+1}\right)^{-1} \left(\mathbf{z}^{k+1} - \mathbf{h}^{k+1}\left(\bar{\mathbf{x}}^{k+1|k}\right)\right) \quad (2.86)$$

$$\Sigma_u^{k+1} = \Sigma^{k+1|k} - \Psi^{k+1} \left(\mathbf{S}^{k+1}\right)^{-1} \left(\Psi^{k+1}\right)^T \quad (2.87)$$

where

$$\Psi^{k+1} = \Sigma^{k+1|k} \left(\hat{\mathbf{H}}^{k+1}\right)^T \quad (2.88)$$

$$\mathbf{S}^{k+1} = \hat{\mathbf{H}}^{k+1} \Sigma^{k+1|k} \left(\hat{\mathbf{H}}^{k+1}\right)^T + \mathbf{R}^{k+1} \quad (2.89)$$

and $\hat{\mathbf{F}}^{k+1}$ and $\hat{\mathbf{H}}^{k+1}$ are the Jacobians of $\mathbf{f}^{k+1}(\cdot)$ and $\mathbf{h}^{k+1}(\cdot)$ evaluated at $\hat{\mathbf{x}}^k$ and $\hat{\mathbf{x}}^{k+1|k}$, respectively:

$$\hat{\mathbf{F}}^{k+1} = \left[\nabla_{\mathbf{x}^k} \left(\mathbf{f}^{k+1} \left(\mathbf{x}^k \right) \right)^T \right]^T \Big|_{\mathbf{x}^k = \bar{\mathbf{x}}^k} \quad (2.90)$$

$$\hat{\mathbf{H}}^{k+1} = \left[\nabla_{\mathbf{x}^{k+1}} \left(\mathbf{h}^{k+1} \left(\mathbf{x}^{k+1} \right) \right)^T \right]^T \Big|_{\mathbf{x}^{k+1} = \bar{\mathbf{x}}^{k+1|k}} \quad (2.91)$$

When the nonlinearity is smooth, EKF can perform very well. However, if nonlinearities are more acute, its performance is severely degraded as it has a tendency to underestimate the mean square error matrix [100].

2.6.2 Approximation 2: unscented Kalman filter

The unscented Kalman filter (UKF) is based on the unscented transformation (UT) to approximate the moments given by (2.67), (2.68), (2.73), (2.77) and (2.81). The UT is a method for approximating, in a computationally efficient manner, the moments of a random variable which has undergone a nonlinear transformation [99,100]. This is done by selecting a number of sigma points with certain weights such that they match the first two moments of the random variable. The points go through the transformation and the sample statistics of the transformed sigma points are collected. In filtering problems, the UKF usually estimates the mean square error matrix more accurately than the EKF [100], it is easier to apply as it does not require the calculation of the Jacobians of the nonlinear functions and it can be applied to non-differentiable functions and even to discontinuous functions.

In the case of nonlinear filtering that is being analysed, i.e., additive noise, it should be noted that the UKF can be applied by using sigma points for the state or for the augmented state formed by the state itself plus the noise [206]. Using the sigma points only for the state reduces the computational complexity of the UKF and, thus, it is the one used in this thesis [203].

The posterior PDF is assumed to be Gaussian $p(\mathbf{x}^k | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^k; \bar{\mathbf{x}}^k, \Sigma^k)$. There are multiple possible choices of sigma points but a common one is to use $N_s = 2n_x + 1$ sigma points $(\mathcal{X}_k^0, \dots, \mathcal{X}_k^{2n_x})$ with weights $(w^0, \dots, w^{2n_x})$ such that [100]

$$\mathcal{X}_k^0 = \bar{\mathbf{x}}^k \quad (2.92)$$

$$\mathcal{X}_k^i = \bar{\mathbf{x}}^k + \left(\sqrt{\frac{n_x}{1 - w^0} \Sigma^k} \right)_i \quad (2.93)$$

$$w^i = \frac{1 - w^0}{2n_x} \quad (2.94)$$

$$\mathcal{X}_k^{i+n_x} = \bar{\mathbf{x}}^k - \left(\sqrt{\frac{n_x}{1 - w^0} \Sigma^k} \right)_i \quad (2.95)$$

$$w^{i+n_x} = \frac{1 - w^0}{2n_x} \quad (2.96)$$

where $i = 1, \dots, n_x$, w^0 is a parameter of the UKF and corresponds to the weight

of the sigma point located on the mean of the prior, n_x is the dimension of the state and $(\sqrt{\mathbf{P}})_i$ is the i th row or column of the matrix square root of $\mathbf{P}$. If the matrix square root $\mathbf{A}$ of $\mathbf{P}$ is of the form $\mathbf{P} = \mathbf{A}^T \mathbf{A}$, the sigma points are formed from the rows of $\mathbf{A}$. Contrarily, if the matrix square root $\mathbf{A}$ of $\mathbf{P}$ is of the form $\mathbf{P} = \mathbf{A} \mathbf{A}^T$, the sigma points are formed from the columns of $\mathbf{A}$ [100]. The matrix square root can be calculated using numerically efficient and stable methods such as the Cholesky decomposition [77]. It should be noted that if $w^0 < 0$ the covariance matrix of the transformation can become negative definite [191] so it is preferable to choose $w^0 \geq 0$.

The first two moments, $\bar{\mathbf{x}}^{k+1|k}$ and $\Sigma^{k+1|k}$, of the predicted density are calculated as

$$\mathcal{X}_{k+1|k}^{\prime i} = \mathbf{f}^{k+1}(\mathcal{X}_k^i), \quad i = 0, \dots, N_s - 1 \quad (2.97)$$

$$\bar{\mathbf{x}}^{k+1|k} = \sum_{i=0}^{N_s-1} w^i \mathcal{X}_{k+1|k}^{\prime i} \quad (2.98)$$

$$\Sigma^{k+1|k} = \sum_{i=0}^{N_s-1} w^i \left(\mathcal{X}_{k+1|k}^{\prime i} - \bar{\mathbf{x}}^{k+1|k} \right) \left(\mathcal{X}_{k+1|k}^{\prime i} - \bar{\mathbf{x}}^{k+1|k} \right)^T + \mathbf{Q}^{k+1} \quad (2.99)$$

For the update phase, the sigma points need to be recalculated using the procedure of equations (2.92)-(2.96) for $\bar{\mathbf{x}}^{k+1|k}$ and $\Sigma^{k+1|k}$ because the process noise has not been included in the calculation of the sigma points [203, 206] yielding $\mathcal{X}_{k+1|k}^i$ $i = 0, \dots, N_s - 1$. The transformed sigma points are

$$\mathcal{Z}_{k+1}^i = \mathbf{h}^{k+1}(\mathcal{X}_{k+1|k}^i), \quad i = 0, \dots, N_s - 1 \quad (2.100)$$

Finally, the moments required for the Kalman filter update are approximated as

$$\bar{\mathbf{z}}^{k+1|k} = \sum_{i=0}^{N_s-1} w^i \mathcal{Z}_{k+1}^i \quad (2.101)$$

$$\mathbf{S}^{k+1} = \sum_{i=0}^{N_s-1} w^i \left(\mathcal{Z}_{k+1}^i - \bar{\mathbf{z}}^{k+1|k} \right) \left(\mathcal{Z}_{k+1}^i - \bar{\mathbf{z}}^{k+1|k} \right)^T + \mathbf{R}^{k+1} \quad (2.102)$$

$$\Psi^{k+1} = \sum_{i=0}^{N_s-1} w^i \left(\mathcal{X}_{k+1|k}^i - \bar{\mathbf{x}}^{k+1|k} \right) \left(\mathcal{Z}_{k+1}^i - \bar{\mathbf{z}}^{k+1|k} \right)^T \quad (2.103)$$

2.6.3 Approximation 3: cubature Kalman filter

The cubature Kalman filter (CKF) applies a spherical-radial cubature rule to compute the necessary integrals to apply a Kalman filter [7, 8]. Nevertheless, the algo-

rithm boils down to the application of an unscented transformation in which $w^0 = 0$, as was previously presented in Section III.A in [100], to calculate the required moments.

The CKF uses $N_s = 2n_x$ cubature points ξ_i with equal weights

$$\xi_i = \sqrt{n_x} (\mathbf{I}_{n_x})_i \quad i = 0, \dots, n_x - 1 \quad (2.104)$$

$$\xi_i = -\sqrt{n_x} (\mathbf{I}_{n_x})_i \quad i = n_x, \dots, 2n_x - 1 \quad (2.105)$$

$$w^i = \frac{1}{2n_x} \quad i = 0, \dots, 2n_x - 1 \quad (2.106)$$

where $\mathbf{I}_{n_x}$ is the identity matrix of dimension n_x and $(\mathbf{I}_{n_x})_i$ denotes the i th column of the matrix $\mathbf{I}_{n_x}$. Assume the posterior at time k is $p(\mathbf{x}^k | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^k; \bar{\mathbf{x}}^k, \Sigma^k)$. Firstly, we calculate $\mathbf{P}$ such as $\Sigma^k = \mathbf{P}\mathbf{P}^T$ using the Cholesky decomposition [77]. Then, the sigma points are calculated as

$$\mathcal{X}_k^i = \mathbf{P}\xi_i + \bar{\mathbf{x}}^k \quad i = 0, \dots, 2n_x - 1 \quad (2.107)$$

The approximation of the first two moments of the predicted density $\bar{\mathbf{x}}^{k+1|k}$ and $\Sigma^{k+1|k}$ are calculated with the same equations as in the UKF.

Assuming the prior is $p(\mathbf{x}^{k+1} | \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{x}^{k+1}; \bar{\mathbf{x}}^{k+1|k}, \Sigma^{k+1|k})$ the sigma points for the update phase are

$$\mathcal{X}_{k+1|k}^i = \mathbf{P}'\xi_i + \bar{\mathbf{x}}^{k+1|k} \quad i = 0, \dots, 2n_x - 1 \quad (2.108)$$

where $\mathbf{P}'$ is such that $\Sigma^{k+1|k} = \mathbf{P}'(\mathbf{P}')^T$. The approximation of the first two moments of the posterior is done as in the UKF.

Chapter 3

Truncated unscented Kalman filtering

3.1 Introduction

As indicated in Chapter 2, the posterior PDF is often approximated in a recursion which involves two steps: prediction and update [9]. In the prediction step, the PDF of the state at the current time given all previous measurements is approximated. This PDF is referred to as the prior PDF. The probabilistic relationship between the measurement and the state is encapsulated by the likelihood. The update step involves using the likelihood along with Bayes' rule and the prior PDF to approximate the posterior PDF.

This chapter focuses on the update step because this is where the main difficulties lie in nonlinear filtering approximations. It has been shown in [32, 91] that the prediction of an approximation to the posterior results in a prior approximation with a smaller error. No such guarantee exists for the update step. In fact, in many cases the update step can potentially result in a significantly higher error. An important example in which the update can induce large errors is when the measurements are precise compared to the prior [151]. These observations suggest that approximation errors are primarily accumulated in successive update steps, hence the focus of this chapter on accurate approximation of the update step.

As outlined in Chapter 2, a large variety of nonlinear filtering approximations have been proposed. Particle filters obtain a weighted sample from the posterior by drawing from an importance density. An important virtue of PFs is that they provide an asymptotically exact approximation of the posterior PDF as the sample size tends to infinity [38]. However, the performance of a PF for a finite sample size

may be poor. Similar comments apply to point-mass filters based on deterministic grid approximations [28, 202]. For this reason there is still considerable interest in computationally efficient Gaussian approximations.

Once a Gaussian approximation to a non-Gaussian posterior PDF is adopted, the possibility of optimal inference disappears. Even so, there are many problems for which a Gaussian approximation can be of sufficient fidelity to provide estimates of similar accuracy to PFs at a fraction of the computational expense. The mean and covariance matrix of the Gaussian approximation are often obtained by approximating the Kalman filter (KF) recursion, although other methods have been suggested [44, 113]. The most-well known of the KF approximations were described in Section 2.6 including the extended KF (EKF), the unscented KF (UKF) and the cubature KF (CKF).

The aforementioned methods suppose that the KF provides a good approximation to the posterior and it is therefore desirable to approximate the KF as closely as possible. However, in certain conditions, e.g., if the measurements are sufficiently precise, the KF approximates the posterior poorly [140]. When this happens the performance of a KF approximation cannot be greatly improved by simply using a more accurate approximation method, as in the UKF compared to the EKF. Instead, the approach used to find an accurate but computationally efficient Gaussian approximation to the posterior must be reconsidered.

The key contributions of this chapter are as follows. First the notion that the performance of the KF can be improved by applying the KF update to a modified prior is proposed. This is formalised in the truncated Kalman filter (TKF) which applies to likelihoods with a bounded, connected support. The TKF involves applying the usual KF update to a prior truncated by the likelihood. A theoretical analysis for the case of a cubic measurement is presented to demonstrate the potential of this idea. There are two barriers to an exact implementation of the TKF. First, although it can be argued that all real likelihoods are of bounded support since no real sensor can provide an infinitely large measurement, most models assume a likelihood with unbounded support. Second, the moments required in the TKF recursion are generally intractable. The second contribution of the chapter is the development of a practical algorithm based on the TKF which addresses these issues. It uses a similar approach to the UKF and is therefore referred to as the truncated UKF (TUKF). The TUKF is of similar computational expense to other Gaussian approximations, such as the EKF and UKF, but is demonstrated to have much better performance in a number of examples.

The rest of the chapter is organised as follows. In Section 3.2, the update stage of the KF is reviewed for nonlinear measurement models. In Section 3.3, the TKF

is introduced when the measurement noise is bounded. In Section 3.4, the TKF approximation for unbounded noise support and the TUKF are addressed. Numerical examples examining the filter performance are provided in Section 3.5. Finally, conclusions are drawn in Section 3.6.

3.2 Kalman filter for nonlinear measurement models

In this section, the update phase of a Kalman filter for nonlinear measurement models is revisited. The time index is dropped for the clarity of presentation. The following measurement equation is assumed:

$$\mathbf{z} = \mathbf{h}(\mathbf{x}) + \boldsymbol{\eta} \quad (3.1)$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the state, $\mathbf{z} \in \mathbb{R}^{n_z}$ is the measurement, $\mathbf{h}(\cdot)$ is nonlinear function and $\boldsymbol{\eta}$ is a zero mean measurement noise with any PDF. The prior PDF of $\mathbf{x}$ is denoted by $p_0(\cdot)$.

From the Bayesian point of view, the MMSE estimator $\hat{\mathbf{x}}_{pos}$ of $\mathbf{x}$ given $\mathbf{z}$ is [13]:

$$\hat{\mathbf{x}}_{pos} = \mathbb{E}[\mathbf{x} | \mathbf{z}] \quad (3.2)$$

whose mean square error matrix is

$$\mathbf{P}_{pos} = \mathbb{E}[(\mathbf{x} - \hat{\mathbf{x}}_{pos})(\mathbf{x} - \hat{\mathbf{x}}_{pos})^T | \mathbf{z}] \quad (3.3)$$

which depends on the measurement and its trace is the MMSE. Calculating (3.2) and (3.3) is the ultimate objective as they are the first two moments of the posterior. Yet, they are very difficult to calculate as the posterior needs to be known.

Conversely, as indicated in Section 2.6, the updated mean of the KF provides the linear MMSE estimator of $\mathbf{x}$ given $\mathbf{z}$, $\hat{\mathbf{x}}_{u,0}$ [105]:

$$\hat{\mathbf{x}}_{u,0} = \hat{\mathbf{x}}_{p,0} + \boldsymbol{\Psi}_0 \mathbf{S}_0^{-1} (\mathbf{z} - \hat{\mathbf{z}}_0) \quad (3.4)$$

where $\hat{\mathbf{x}}_{p,0}$ is the mean of the prior distribution of $\mathbf{x}$ and

$$\hat{\mathbf{z}}_0 = \mathbb{E}[\mathbf{z}] = \int \mathbb{E}[\mathbf{z} | \mathbf{x}] p_0(\mathbf{x}) d\mathbf{x} \quad (3.5)$$

$$\mathbf{S}_0 = \text{cov}[\mathbf{z}] = \int \mathbb{E}[(\mathbf{z} - \hat{\mathbf{z}}_0)(\mathbf{z} - \hat{\mathbf{z}}_0)^T | \mathbf{x}] p_0(\mathbf{x}) d\mathbf{x} \quad (3.6)$$

$$\boldsymbol{\Psi}_0 = \text{cov}[\mathbf{x}, \mathbf{z}] = \int \mathbb{E}[(\mathbf{x} - \hat{\mathbf{x}}_{p,0})(\mathbf{z} - \hat{\mathbf{z}}_0)^T | \mathbf{x}] p_0(\mathbf{x}) d\mathbf{x} \quad (3.7)$$

The updated covariance matrix given by the KF, $\mathbf{P}_{u,0}$, is

$$\mathbf{P}_{u,0} = \mathbf{P}_{p,0} - \boldsymbol{\Psi}_0 \mathbf{S}_0^{-1} \boldsymbol{\Psi}_0^T \quad (3.8)$$

where $\mathbf{P}_{p,0}$ is the covariance of the prior distribution of $\mathbf{x}$. As shown in [105], $\mathbf{P}_{u,0}$ is the mean square error matrix averaged over all possible measurements even in a nonlinear set-up:

$$\mathbf{P}_{u,0} = \mathbb{E} \left[(\mathbf{x} - \hat{\mathbf{x}}_{u,0}) (\mathbf{x} - \hat{\mathbf{x}}_{u,0})^T \right] \quad (3.9)$$

In filtering problems, $\hat{\mathbf{x}}_{u,0}$ and $\mathbf{P}_{u,0}$ are usually regarded as approximations of the first two moments of the posterior, $\hat{\mathbf{x}}_{pos}$ and $\mathbf{P}_{pos}$. The reason for this is that, when the prior $p_0(\cdot)$ and the measurement noise are Gaussian, and $\mathbf{h}(\cdot)$ is a linear function, $\hat{\mathbf{x}}_{pos} = \hat{\mathbf{x}}_{u,0}$ and $\mathbf{P}_{pos} = \mathbf{P}_{u,0}$ having a closed form solution [105]. Also, if the measurement function is nonlinear but one assumes that the variable $(\mathbf{x}, \mathbf{z})$ is jointly Gaussian, KF update equations are equivalent to Bayes' rule [95]. However, when the measurement function is nonlinear, there is no guarantee that $\hat{\mathbf{x}}_{u,0}$ and $\mathbf{P}_{u,0}$ are close to $\hat{\mathbf{x}}_{pos}$ and $\mathbf{P}_{pos}$. In fact, using a second order Taylor approximation, it can be shown that measurement nonlinearities become more significant for smaller measurement noise variance and larger prior variance. This applies not only to the EKF [3, 97] but also more generally to the KF [140]. Then, in such conditions, all the algorithms that are based on Kalman filtering, such as the UKF, CKF, EKF or Monte Carlo KF (MCKF) are expected to approximate the posterior poorly. The MCKF is a KF in which the integrals (3.5), (3.6) and (3.7) are calculated via Monte Carlo simulation using samples from the prior.

3.2.1 Illustration of the KF's deficiencies

This section serves as an illustration to show the problems of Kalman filtering with nonlinear measurements models. Then, consider the following range-bearing measurement model [169]:

$$z_r = \sqrt{x^2 + y^2} + \eta_r \quad (3.10)$$

$$z_\theta = \arctan\left(\frac{y}{x}\right) + \eta_\theta \quad (3.11)$$

where $\mathbf{z} = [z_r, z_\theta]^T$, $\mathbf{x} = [x, y]^T$ represents the position of a target, η_r is the measurement noise for the range with variance σ_r^2 , η_θ is the measurement noise for the bearing with variance σ_θ^2 and these noises are zero-mean Gaussian distributed and independent. In this case, circular sums and subtractions are used when performing these operations on the angle measurement, z_θ [153]. The objective is to approximate the first two moments of the posterior as they correspond to the MMSE estimator

Figure 3.1: Samples from the posterior PDF and 3σ -ellipses for the true posterior and conventional KF-type algorithms. All conventional KF-type approximations are rather inaccurate.

and the covariance matrix of the estimation error, equations (3.2) and (3.3). The case when $p_0(\cdot)$ is Gaussian with mean $\hat{\mathbf{x}}_{p,0} = [5, 5]^T$ and covariance matrix $\mathbf{P}_{p,0} = 36\mathbf{I}_2$ is analysed. Besides, the noise parameters are $\sigma_r = 1$ m and $\sigma_\theta = 2\pi/180$ rad. For the UKF, $N_s = 2n_x + 1$ sigma points are used and the weight of the sigma point located on the mean of $p_0(\cdot)$ is $1/3$. For the MCKF, $2 \cdot 10^4$ samples are drawn from the prior to approximate (3.5), (3.6) and (3.7).

In Figure 3.1, $2 \cdot 10^4$ samples of the posterior PDF, which are obtained using Bayes' rule, are plotted. This is the PDF to approximate after receiving a certain measurement. The 3σ -ellipse of the posterior is shown in red colour. This ellipse is plotted as it represents the first two moments of the posterior [13]. For linear/Gaussian models, the KF's ellipse coincides with the ellipse of the posterior. However, in this nonlinear example, the ellipses given by the EKF, UKF, CKF or MCKF are rather inaccurate compared to the ellipse of the posterior. The results illustrate an important concept: improved approximation of the KF does not necessarily lead to improved approximation of the posterior. The MCKF with 20000 samples is by far the most accurate approximation of the KF. However it is arguably the worst approximation to the posterior. This suggests that any efforts to improve the accuracy of the KF moment approximations, as in the UKF and CKF, will not necessarily improve overall performance especially when nonlinearities are significant.

In the rest of the chapter, an algorithm that improves the approximation of the first two moments of the posterior will be developed based on approximating the measurement noise PDF as one with bounded support.

3.3 Truncated Kalman Filter with bounded noise support

In this section, the ideas behind the truncated Kalman filter (TKF) are presented. In Section 3.3.1, it is justified why applying the KF to a modified prior PDF instead of the real prior PDF can improve estimation performance when some conditions are met. In Section 3.3.2, an example in which the advantages of using the TKF over the KF are demonstrated is provided.

3.3.1 Theory

The state vector is written as $\mathbf{x} = [\mathbf{a}^T, \mathbf{b}^T]^T$, where $\mathbf{a} \in \mathbb{R}^{n_a}$, $\mathbf{b} \in \mathbb{R}^{n_b}$ and $n_x = n_a + n_b$, such that the measurement equation is

$$\mathbf{z} = \mathbf{h}(\mathbf{a}) + \boldsymbol{\eta} \quad (3.12)$$

where $\mathbf{h}(\cdot)$ is a function of some elements of the state. The TKF is derived under the assumptions:

A1 The measurement function $\mathbf{h}(\cdot)$ is a continuous, bijective function.

A2 The PDF of the additive noise has a bounded, connected support:

$$p_{\boldsymbol{\eta}}(\boldsymbol{\eta}) = 0 \quad \text{if} \quad \boldsymbol{\eta} \notin I_{\boldsymbol{\eta}} \subset \mathbb{R}^{n_z} \quad (3.13)$$

where $I_{\boldsymbol{\eta}}$ is an n_z -dimensional connected region.

Due to Assumption A2, the PDF of the measurement conditioned on the state can be written as

$$p(\mathbf{z}|\mathbf{x}) = p(\mathbf{z}|\mathbf{a}) = p_{\boldsymbol{\eta}}(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\boldsymbol{\eta}}}(\mathbf{z} - \mathbf{h}(\mathbf{a})) \quad (3.14)$$

where $\chi_{I_{\boldsymbol{\eta}}}(\cdot)$ is the indicator function on the subset $I_{\boldsymbol{\eta}}$.

Due to Assumption A1, (3.14) can be written as:

$$p(\mathbf{z}|\mathbf{x}) = p_{\boldsymbol{\eta}}(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) \quad (3.15)$$

where

$$I_{\mathbf{x}}(\mathbf{z}) = \left\{ \mathbf{x} \mid \mathbf{x} = \left[\left(\mathbf{h}^{-1}(\mathbf{z} - \boldsymbol{\eta}) \right)^T, \mathbf{b}^T \right]^T, \boldsymbol{\eta} \in I_{\boldsymbol{\eta}}, \mathbf{b} \in \mathbb{R}^{n_b} \right\} \quad (3.16)$$

$$= I_{\mathbf{a}}(\mathbf{z}) \times \mathbb{R}^{n_b} \quad (3.17)$$

The set $I_{\mathbf{x}}(\mathbf{z})$ depends on the current measurement $\mathbf{z}$, the support of the measurement noise $I_{\boldsymbol{\eta}}$ and the inverse function of $\mathbf{h}(\cdot)$. The posterior PDF of $\mathbf{x}$ applying Bayes' rule and (3.15) is:

$$\begin{aligned} p(\mathbf{x}|\mathbf{z}) &\propto p(\mathbf{z}|\mathbf{x}) \cdot p_0(\mathbf{x}) \\ &= p_{\boldsymbol{\eta}}(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) \end{aligned} \quad (3.18)$$

A usual approach to approximate the first two moments of the posterior is to apply a Kalman filter using $p_0(\cdot)$ as the prior (KF). However, (3.18) can be rewritten as

$$p(\mathbf{x}|\mathbf{z}) \propto p(\mathbf{z}|\mathbf{x}) p_1(\mathbf{x}; \mathbf{z}) \quad (3.19)$$

where

$$p_1(\mathbf{x}; \mathbf{z}) = \frac{1}{\varepsilon_1} p_0(\mathbf{x}) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) \quad (3.20)$$

and ε_1 is a normalising constant and $p_1(\cdot)$ is a modified “prior” PDF, which is parameterised by $\mathbf{z}$. It should be noted that $p_1(\cdot)$ is a truncated version of $p_0(\cdot)$ and the result of applying Bayes' rule, equations (3.18) and (3.19), is the same if $p_0(\cdot)$ or $p_1(\cdot)$ is used as the prior. This is illustrated in Figure 3.2.

Equation (3.19) suggests an alternative approach to KF using $p_1(\cdot)$ as the prior, what is referred to as the truncated KF (TKF). It should be noted that this approach does not calculate the conventional KF moments, i.e., (3.5), (3.6) and (3.7) as these are calculated with respect to $p_0(\cdot)$. The motivation behind this approach is that measurement nonlinearities are less significant when the prior variance is low [3, 97, 140]. Then, the use of $p_1(\cdot)$ instead of $p_0(\cdot)$ aims to reduce the variance of the prior PDF [29] so that the effect of nonlinearities decreases. Therefore, the KF applied to $p_1(\cdot)$ is expected to have higher performance than the KF applied to $p_0(\cdot)$.

3.3.2 Example

In this example the benefits of using $p_1(\cdot)$ rather than $p_0(\cdot)$ as the prior are shown when the KF and TKF are tractable. That is, integrals (3.5), (3.6) and (3.7) can be calculated analytically for $p_0(\cdot)$ and $p_1(\cdot)$. Consider a random parameter x , whose prior is $p_0(x) = \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2)$, which generates the measurement

$$z = ax^3 + \eta \quad (3.21)$$

where $a \in \mathbb{R}$ is a known parameter and $\eta \sim U_{[-\Delta/2, \Delta/2]}$ (uniform PDF in the interval $[-\Delta/2, \Delta/2]$). It is desired to compare the RMS errors of estimating x for both methods noting that the information provided by the measurement about x

Figure 3.2: Graphical representation of $p_0(\mathbf{x})$, $p_1(\mathbf{x}; \mathbf{z})$ and $p(\mathbf{z} | \mathbf{x})$ in one dimension. When the likelihood has a bounded support, Bayes' rule will exactly be met by the “prior” PDFs: $p_0(\mathbf{x})$ and $p_1(\mathbf{x}; \mathbf{z})$.

is higher when z increases as the likelihood is narrower. Consequently, the KF is expected to provide high errors for high values of z .

In this case, $I_x(z) = \left[\sqrt[3]{\frac{z-\Delta/2}{a}}, \sqrt[3]{\frac{z+\Delta/2}{a}} \right]$ and thus

$$p_1(x; z) = \frac{1}{\varepsilon_1} \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) \chi_{I_x(z)}(x) \quad (3.22)$$

Integrals (3.5), (3.6) and (3.7) can be calculated analytically for $p_0(\cdot)$ and $p_1(\cdot)$. The RMS errors of the KF and TKF estimators averaged over the posterior PDF for a particular measurement also admit a closed form expression. The exact results of those integrals and errors are given in Appendix A.1.

The cases when $a = 0.1$, $\hat{x}_{p,0} = 3$, $\sigma_{p,0}^2 = 10$ and $\Delta \in \{10, 50, 100\}$ are analysed. The RMS errors of the KF and TKF against the measurement values that occur for $x \in [\hat{x}_{p,0} - 3\sigma_{p,0}, \hat{x}_{p,0} + 3\sigma_{p,0}]$ and the size of $I_x(z)$ are plotted in Figure 3.3. When the measurement is lower than $\Delta/2$, i.e., the size of $I_x(z)$ is large, the KF and TKF perform similarly. When the measurement is higher than $\Delta/2$, $I_x(z)$ gets smaller and the RMS error of the KF increases considerably. There is one value of the measurement for which the KF's RMS error drops even for very informative measurements. This is because the estimate given by the KF is a linear function of the measurement, see (3.4), and for very informative measurements, the posterior mean is close to $x = \sqrt[3]{z/a}$ as the likelihood is narrow and concentrated around this point. Then, in this case, for $z > 0$, there is one point in which the KF estimate is close to the posterior mean for informative measurements producing a low RMS error but it increases rapidly.

On the other hand, the TKF performs much better in general. In addition,

Figure 3.3: (a) RMS errors of the KF and TKF and (b) size of $I_x(z)$ plotted against the measurement for different values of Δ . The TKF outperforms the KF especially when the size of $I_x(z)$ is small.

the higher the measurement is or the lower Δ is, the smaller the size of $I_x(z)$ is, and the TKF improvement over the KF is more acute as the measurement is more informative. The reason for this is that the smaller the size of $I_x(z)$ is, the smaller the variance of $p_1(\cdot)$ becomes compared to the variance of $p_0(\cdot)$ [29] and, therefore, the KF applied to $p_1(\cdot)$ should work better.

3.4 Truncated Kalman Filter with unbounded noise support

In this section, the TKF ideas are applied when the measurement noise support is unbounded. However, it should be noted that in reality the measurement noise support is always bounded as no noise can provide an infinitely large value. It is only the models which assume unbounded measurement noise support for convenience. Firstly, the example of Section 3.3.2 is reconsidered to demonstrate that truncating the measurement noise PDF and applying the TKF can provide a significant performance improvement over the KF with nonlinear measurement models. Secondly, the effects of some approximations that are made for general measurement models are analysed. These effects lead to the introduction of a variant of the TKF. Thirdly, the truncated unscented Kalman filter (TUKF) is proposed as an approximation to the TKF for general problems.

3.4.1 Truncation of the measurement noise

One possible way of using the TKF's ideas when the measurement noise PDF support is unbounded is to truncate it and directly apply the method of Section 3.3. The truncated noise PDF $p_{\eta_T}(\cdot)$ is calculated neglecting the areas where the original noise PDF $p_\eta(\cdot)$ is low according to a threshold Γ_η

$$p_{\eta_T}(\eta) = \frac{p_\eta(\eta)}{\varepsilon_\eta} \chi_{p_\eta(\eta) > \Gamma_\eta}(\eta) \quad (3.23)$$

where ε_η is the normalising constant such that the integral of the PDF is one.

Assuming the noise support is bounded when it is not and using the TKF provides a high performance gain in comparison with the KF for informative measurements. The implication is that it is a better approximation to Bayes' rule to truncate the noise and apply the TKF than approximating Bayes' rule as a linear function of the measurement (what the KF does). To demonstrate this, the example of Section 3.3.2 for $\Delta = 100$ is reconsidered but using a zero-mean Gaussian noise with variance $\Delta^2/12$, i.e., a noise with the same mean and variance but with unbounded support. The RMS error is calculated via Monte Carlo simulation using 10000 samples from the prior and is shown in Figure 3.4(a) for different values of Γ_η . Note that the main drop in the RMS error occurs for the measurements that make $I_x(z)$ decrease considerably as happened in Figure 3.3. It is also important to notice that for low values of z (uninformative measurements), the KF and TKF have roughly the same performance. Therefore, Figure 3.4(a) shows that it can be beneficial to truncate the noise PDF and apply the TKF if the measurement function is nonlinear.

3.4.2 Practical approach: use of a mixture

In the example of Section 3.4.1, a truncation threshold Γ_η was set and the TKF steps could be applied: calculate $I_{\mathbf{x}}(\mathbf{z})$, the first two moments of $p_1(\cdot)$ and the KF moments for $p_1(\cdot)$. In a general scenario, neither of these steps is possible without several approximations. As explained in the next section, the approximations used to calculate $p_1(\cdot)$ are:

AP1 The measurement function $\mathbf{h}(\cdot)$ is locally linear.

AP2 The marginal prior of $\mathbf{a}$, $p_0(\mathbf{a})$, is constant over the region $I_{\mathbf{a}}(\mathbf{z})$.

AP3 The measurement noise satisfies $\boldsymbol{\eta} \sim U_{I_\eta}$ where I_η is such that the truncated noise has the same first two moments as the real noise $\mathbf{E}[\boldsymbol{\eta}] = \mathbf{0}$ and $\text{cov}[\boldsymbol{\eta}] = \mathbf{R}$.

The implications of Approximations AP1 and AP2 are the following: for informative measurements, AP1 and AP2 are reasonable as the likelihood is narrow

(a)

(b)

(c)

Figure 3.4: RMS error plotted against the measurement for the KF and the TKF for different values of Γ_η , which are normalised with respect to $p_\eta(0)$: (a) Without approximations (b) With Approximations AP1 and AP2 with different thresholds and with Approximations AP1-AP2-AP3 (c) Proposed practical algorithm: TUKF with $\alpha = 1$, with adaptive α and UKF. The TUKF with adaptive α generally outperforms the UKF.

and the approximated TKF should work better than the KF but, for uninformative measurements, AP1 and AP2 are inaccurate as the likelihood is broad and the approximated TKF should work worse than the KF. This can be checked in Figure 3.4(b) where the RMS error of the TKF estimator, implemented with the true likelihood and prior replaced by approximations satisfying AP1 and AP2, is shown¹. It can be seen that, as the measurement becomes more informative, i.e. as z increases, the RMSE for the TKF under AP1 and AP2 approaches the RMSE obtained by the TKF with truncated measurement noise. However, for less informative measurements, the performance of the TKF deteriorates significantly compared to both the TKF with truncated measurement noise and the KF. These observations support the notion that AP1 and AP2 are reasonable for informative measurements but should not be applied for uninformative ones.

Considering only AP1 and AP2, a truncation threshold still needs to be set but it becomes less important as the measurement is more informative. However, Approximation AP3 simplifies the algorithm as it approximates the measurement noise PDF by a uniform PDF and, therefore, the truncation threshold is no longer necessary. The exact form of I_η is not needed in the practical algorithm, see Section 3.4.3. The RMS error under AP1, AP2 and AP3 is also shown in Figure 3.4(b). It can be seen that AP3 does not have an important effect on performance.

It is clear from Figure 3.4(b) that once AP1 and AP2 are applied, the TKF should only be used selectively, specifically when the measurements are informative. The way proposed in this chapter to solve this problem is based on reconsidering Bayes' rule with bounded noise support, equation (3.18). It is shown in Appendix A.3 that the prior

$$p_2(\mathbf{x}; \mathbf{z}) = \alpha_0 p_1(\mathbf{x}; \mathbf{z}) + (1 - \alpha_0) p_0(\mathbf{x}) \quad (3.24)$$

where $\alpha_0 \in [0, 1]$ also meets Bayes' rule exactly for bounded noise support. The parameter α_0 introduces a degree of freedom that allows several techniques to approximate the first two moments of the posterior. The KF uses $\alpha_0 = 0$ and the TKF $\alpha_0 = 1$. However, (3.24) suggests another equally valid option: apply a KF update to $p_0(\cdot)$ (conventional KF), another KF update to $p_1(\cdot)$ (TKF) and merge both results to form the final approximation to the posterior. Then, denoting the mean and covariance matrix of $p_1(\cdot)$ as $\hat{\mathbf{x}}_{p,1}$ and $\mathbf{P}_{p,1}$, the updated mean and covariance matrix are [169]:

$$\hat{\mathbf{x}}_{u,2} = \alpha \hat{\mathbf{x}}_{u,1} + (1 - \alpha) \hat{\mathbf{x}}_{u,0} \quad (3.25)$$

¹The linearisation of AP1 is done around the value that maximises the likelihood as explained in Section 3.4.3.1. Once $p_1(\cdot)$ is approximated under AP1 and AP2, the TKF moments are calculated analytically. The details can be found in Appendix A.2.

$$\begin{aligned} \mathbf{P}_{u,2} = & \alpha \left[\mathbf{P}_{u,1} + (\hat{\mathbf{x}}_{u,1} - \hat{\mathbf{x}}_{u,2}) (\hat{\mathbf{x}}_{u,1} - \hat{\mathbf{x}}_{u,2})^T \right] \\ & + (1 - \alpha) \left[\mathbf{P}_{u,0} + (\hat{\mathbf{x}}_{u,0} - \hat{\mathbf{x}}_{u,2}) (\hat{\mathbf{x}}_{u,0} - \hat{\mathbf{x}}_{u,2})^T \right] \end{aligned} \quad (3.26)$$

where $\alpha \in [0, 1]$ maintains the degree of freedom of α_0 . It is shown in Appendix A.4 that $\alpha = \alpha_0 / [\alpha_0 + (1 - \alpha_0) \varepsilon_1]$ where ε_1 comes from (3.20). As $\alpha_0 \in [0, 1]$ and $\varepsilon_1 \in [0, 1]$, $\alpha \in [0, 1]$ is another parameter that can be chosen freely and, therefore, α_0 does not need to be considered in the algorithm. The results of Figure 3.4(b) suggest that the TKF should be favoured when the measurements are informative and the KF should be emphasised for uninformative measurements. This corresponds to choosing α close to one for informative measurements and nearly zero for uninformative measurements. In the following section an algorithm for approximating (3.25) and (3.26) and adaptively selecting α is proposed.

3.4.3 Truncated unscented Kalman filter

A closed form solution of the TKF is rarely tractable because of the calculation of the region $I_{\mathbf{x}}(\mathbf{z})$ and integrals (3.5)-(3.7) for $p_0(\cdot)$ and $p_1(\cdot)$. Then, some approximations must be used: the first one to approximate the first two moments of $p_1(\cdot)$ and the second one to approximate the moments (3.5), (3.6) and (3.7) with respect to $p_0(\cdot)$ and $p_1(\cdot)$. These issues and the selection of α are addressed in this section.

3.4.3.1 Approximation of the first two moments of $p_1(\cdot)$

The calculation of $p_1(\cdot)$, which requires the calculation of $I_{\mathbf{a}}(\mathbf{z})$, does not have an analytical expression in general. In this section, it is explained how to approximate the first two moments of $p_1(\cdot)$ as they are everything one needs to know about $p_1(\cdot)$ in the practical implementation of the filter, as will be explained in Section 3.4.3.2.

Firstly, the region $I_{\mathbf{a}}(\mathbf{z})$ needs to be approximated. To this end, Approximation AP1 is used such that the measurement function $\mathbf{h}(\cdot)$ is approximated using a first-order Taylor series around $\mathbf{a} = \tilde{\mathbf{a}}(\mathbf{z})$. The value that maximises the likelihood $\tilde{\mathbf{a}}(\mathbf{z}) = \arg \max_{\mathbf{a}} p(\mathbf{z}|\mathbf{a})$ is chosen because it is the most probable observable state according to the measurement and $p_1(\cdot)$ is located around it. In many cases of interest, such as when the measurement noise is zero-mean Gaussian distributed, the value of $\mathbf{a}$ that maximises the likelihood is $\tilde{\mathbf{a}}(\mathbf{z}) = \mathbf{h}^{-1}(\mathbf{z})$. Therefore:

$$\mathbf{h}(\mathbf{a}) \approx \mathbf{h}(\tilde{\mathbf{a}}(\mathbf{z})) + \tilde{\mathbf{H}}(\mathbf{a} - \tilde{\mathbf{a}}(\mathbf{z})) \quad (3.27)$$

where

$$\tilde{\mathbf{H}} = \left[\nabla_{\mathbf{a}} \mathbf{h}^T(\mathbf{a}) \right]^T \Big|_{\mathbf{a}=\tilde{\mathbf{a}}(\mathbf{z})} \quad (3.28)$$

is the Jacobian of $\mathbf{h}(\cdot)$ evaluated at $\tilde{\mathbf{a}}(\mathbf{z})$.

The inverse of the measurement function $\mathbf{h}^{-1}(\cdot)$ is approximated using (3.27):

$$\mathbf{h}^{-1}(\mathbf{y}) \approx \tilde{\mathbf{a}}(\mathbf{z}) - \tilde{\mathbf{H}}^{-1}[\mathbf{y} - \mathbf{h}(\tilde{\mathbf{a}}(\mathbf{z}))] \quad (3.29)$$

Substituting $\tilde{\mathbf{a}}(\mathbf{z}) = \mathbf{h}^{-1}(\mathbf{z})$ and (3.29) into (3.16):

$$I_{\mathbf{a}}(\mathbf{z}) = \left\{ \mathbf{a} \mid \mathbf{a} = \tilde{\mathbf{a}}(\mathbf{z}) - \tilde{\mathbf{H}}^{-1}\boldsymbol{\eta}, \boldsymbol{\eta} \in I_{\boldsymbol{\eta}} \right\} \quad (3.30)$$

Then, (3.20) becomes

$$p_1(\mathbf{x}; \mathbf{z}) = p_1(\mathbf{a}, \mathbf{b}; \mathbf{z}) = \frac{1}{\varepsilon_1} \chi_{I_{\mathbf{a}}(\mathbf{z})}(\mathbf{a}) p_0(\mathbf{b} | \mathbf{a}) p_0(\mathbf{a}) \quad (3.31)$$

Under Approximation AP2, (3.31) can be written as

$$p_1(\mathbf{a}, \mathbf{b}; \mathbf{z}) = \frac{1}{\varepsilon_2} \chi_{I_{\mathbf{a}}(\mathbf{z})}(\mathbf{a}) p_0(\mathbf{b} | \mathbf{a}) \quad (3.32)$$

where ε_2 is a normalising constant. Integrating out variable $\mathbf{b}$ in (3.32):

$$p_1(\mathbf{a}; \mathbf{z}) = \frac{1}{\varepsilon_2} \chi_{I_{\mathbf{a}}(\mathbf{z})}(\mathbf{a}) \quad (3.33)$$

Under Approximation AP3, it is shown in Appendix A.5 that the mean $\boldsymbol{\mu}_{a,1}$ and covariance matrix $\boldsymbol{\Sigma}_{a,1}$ of $p_1(\mathbf{a}; \mathbf{z})$ are

$$\boldsymbol{\mu}_{a,1} = \int \mathbf{a} p_1(\mathbf{a}; \mathbf{z}) d\mathbf{a} = \tilde{\mathbf{a}}(\mathbf{z}) \quad (3.34)$$

$$\boldsymbol{\Sigma}_{a,1} = \int (\mathbf{a} - \boldsymbol{\mu}_{a,1}) (\mathbf{a} - \boldsymbol{\mu}_{a,1})^T p_1(\mathbf{a}; \mathbf{z}) d\mathbf{a} = \tilde{\mathbf{H}}^{-1} \mathbf{R} (\tilde{\mathbf{H}}^{-1})^T \quad (3.35)$$

Noting that $p_0(\mathbf{x}) = p_0(\mathbf{a}, \mathbf{b})$, the mean $\hat{\mathbf{x}}_{p,0}$ and covariance matrix $\mathbf{P}_{p,0}$ of $p_0(\cdot)$ is partitioned as

$$\hat{\mathbf{x}}_{p,0} = \begin{bmatrix} \mathbf{E}[\mathbf{a}] \\ \mathbf{E}[\mathbf{b}] \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mu}_{a,0} \\ \boldsymbol{\mu}_{b,0} \end{bmatrix} \quad (3.36)$$

$$\mathbf{P}_{p,0} = \begin{bmatrix} \text{cov}[\mathbf{a}] & \text{cov}[\mathbf{a}, \mathbf{b}] \\ \text{cov}[\mathbf{b}, \mathbf{a}] & \text{cov}[\mathbf{b}] \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Sigma}_{a,0} & \boldsymbol{\Sigma}_{ab,0} \\ \boldsymbol{\Sigma}_{ab,0}^T & \boldsymbol{\Sigma}_{b,0} \end{bmatrix} \quad (3.37)$$

Lastly, as shown in Appendix A.6, the mean $\hat{\mathbf{x}}_{p,1}$ and covariance matrix $\mathbf{P}_{p,1}$ of

$p_1(\mathbf{a}, \mathbf{b}; \mathbf{z})$ are:

$$\hat{\mathbf{x}}_{p,1} = \begin{bmatrix} \boldsymbol{\mu}_{a,1} \\ \boldsymbol{\mu}_{b,1} \end{bmatrix} \quad (3.38)$$

$$\mathbf{P}_{p,1} = \begin{bmatrix} \boldsymbol{\Sigma}_{a,1} & \boldsymbol{\Sigma}_{ab,1} \\ (\boldsymbol{\Sigma}_{ab,1})^T & \boldsymbol{\Sigma}_{b,1} \end{bmatrix} \quad (3.39)$$

where

$$\boldsymbol{\mu}_{b,1} = \boldsymbol{\mu}_{b,0} + \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} (\boldsymbol{\mu}_{a,1} - \boldsymbol{\mu}_{a,0}) \quad (3.40)$$

$$\boldsymbol{\Sigma}_{ab,1} = \boldsymbol{\Sigma}_{a,1} (\boldsymbol{\Sigma}_{a,0}^{-1})^T \boldsymbol{\Sigma}_{ab,0} \quad (3.41)$$

$$\begin{aligned} \boldsymbol{\Sigma}_{b,1} = & \boldsymbol{\Gamma} - (\boldsymbol{\mu}_{b,1} - \boldsymbol{\mu}_{b,0}) (\boldsymbol{\mu}_{b,1} - \boldsymbol{\mu}_{b,0})^T \\ & + \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} \left[\boldsymbol{\Sigma}_{a,1} + (\boldsymbol{\mu}_{a,1} - \boldsymbol{\mu}_{a,0}) (\boldsymbol{\mu}_{a,1} - \boldsymbol{\mu}_{a,0})^T \right] (\boldsymbol{\Sigma}_{a,0}^{-1})^T \boldsymbol{\Sigma}_{ab,0} \end{aligned} \quad (3.42)$$

and

$$\boldsymbol{\Gamma} = \boldsymbol{\Sigma}_{b,0} - \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} \boldsymbol{\Sigma}_{ab,0} \quad (3.43)$$

3.4.3.2 Approximation of the prior moments

This section describes how to approximate the moments (3.5), (3.6) and (3.7) with respect to $p_0(\cdot)$ and $p_1(\cdot)$. These approximations are based on the unscented transformation (UT) [99], or more precisely, on the UT for conditionally linear models [143]. The PDFs $p_0(\cdot)$ and $p_1(\cdot)$ can be written as:

$$p_j(\mathbf{a}, \mathbf{b}) = p_0(\mathbf{b}|\mathbf{a}) p_j(\mathbf{a}) \quad (3.44)$$

where $j \in \{0, 1\}$ and it is highlighted that the conditional PDF of $\mathbf{b}$ given $\mathbf{a}$ is the same for both PDFs because the truncation is applied to $\mathbf{a}$.

The first integral to approximate is (3.5). Substituting (3.12) and (3.44) into (3.5), (3.5) can be written as:

$$\hat{\mathbf{z}}_j = \int \mathbf{h}(\mathbf{a}) p_0(\mathbf{b}|\mathbf{a}) p_j(\mathbf{a}) d\mathbf{a} d\mathbf{b} \quad (3.45)$$

Integrating out $\mathbf{b}$:

$$\hat{\mathbf{z}}_j = \int \mathbf{h}(\mathbf{a}) p_j(\mathbf{a}) d\mathbf{a} \quad (3.46)$$

The UT approximation to (3.46) proceeds by selecting N_s sigma points $\mathcal{A}_j^1, \dots, \mathcal{A}_j^{N_s}$ along with weights $w^1, \dots, w^{N_s}$. These sigma points and weights can be found using

any of the methods discussed in [100]. In all the implementations in this thesis, $N_s = 2n_a + 1$ sigma points are used and the weight of the sigma point located on the mean is $1/3$. The transformed sigma-points are calculated as

$$\mathcal{Z}_j^i = \mathbf{h} \left(\mathcal{A}_j^i \right), \quad i = 1, \dots, N_s \quad (3.47)$$

Then, the UT approximation to (3.46) is

$$\hat{\mathbf{z}}_j = \sum_{i=1}^{N_s} w^i \mathcal{Z}_j^i \quad (3.48)$$

The second integral to approximate is (3.6)

$$\mathbf{S}_j = \int \mathbb{E} \left[(\mathbf{z} - \hat{\mathbf{z}}_j) (\mathbf{z} - \hat{\mathbf{z}}_j)^T | \mathbf{x} \right] p_0(\mathbf{b} | \mathbf{a}) p_j(\mathbf{a}) d\mathbf{a} d\mathbf{b} \quad (3.49)$$

Substituting (3.12) into (3.49), one gets

$$\mathbf{S}_j = \mathbf{R} + \int (\mathbf{h}(\mathbf{a}) - \hat{\mathbf{z}}_j) (\mathbf{h}(\mathbf{a}) - \hat{\mathbf{z}}_j)^T p_j(\mathbf{a}) d\mathbf{a} \quad (3.50)$$

The UT approximation to (3.50) can be calculated using (3.47):

$$\mathbf{S}_j = \mathbf{R} + \sum_{i=1}^{N_s} w^i \left(\mathcal{Z}_j^i - \hat{\mathbf{z}}_j \right) \left(\mathcal{Z}_j^i - \hat{\mathbf{z}}_j \right)^T \quad (3.51)$$

The third integral to approximate is (3.7)

$$\begin{aligned} \Psi_j &= \int \mathbb{E} \left[(\mathbf{x} - \hat{\mathbf{x}}_{p,j}) (\mathbf{z} - \hat{\mathbf{z}}_j)^T | \mathbf{x} \right] p_0(\mathbf{b} | \mathbf{a}) p_j(\mathbf{a}) d\mathbf{a} d\mathbf{b} \\ &= \int \left(\begin{bmatrix} \mathbf{a} \\ \mathbf{b} \end{bmatrix} - \hat{\mathbf{x}}_{p,j} \right) (\mathbf{h}(\mathbf{a}) - \hat{\mathbf{z}}_j)^T p_0(\mathbf{b} | \mathbf{a}) p_j(\mathbf{a}) d\mathbf{a} d\mathbf{b} \end{aligned} \quad (3.52)$$

Integrating out $\mathbf{b}$:

$$\Psi_j = \int \left(\begin{bmatrix} \mathbf{a} \\ \mathbf{v}_b(\mathbf{a}) \end{bmatrix} - \hat{\mathbf{x}}_{p,j} \right) (\mathbf{h}(\mathbf{a}) - \hat{\mathbf{z}}_j)^T p_j(\mathbf{a}) d\mathbf{a} \quad (3.53)$$

where

$$\mathbf{v}_b(\mathbf{a}) \triangleq \mathbb{E}[\mathbf{b} | \mathbf{a}] = \boldsymbol{\mu}_{b,0} + \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} (\mathbf{a} - \boldsymbol{\mu}_{a,0}) \quad (3.54)$$

Let $\mathcal{X}_j^i = \left[\left(\mathcal{A}_j^i \right)^T, \left(\mathbf{v}_b \left(\mathcal{A}_j^i \right) \right)^T \right]^T$ for $i = 1, \dots, N_s$. Then, (3.53) can be approxi-

mated as [143]

$$\Psi_j = \sum_{i=1}^{N_s} w^i \left(\mathcal{X}_j^i - \hat{\mathbf{x}}_{p,j} \right) \left(\mathcal{Z}_j^i - \hat{\mathbf{z}}_j \right)^T \quad (3.55)$$

3.4.3.3 Selection of α

Approximations AP1 and AP2 are accurate for informative measurements but inaccurate for uninformative measurements. Therefore, the degree of freedom given by α , which was obtained from Bayes' rule as indicated in Section 3.4.2, should be chosen such that it favours $p_1(\cdot)$ when its variance is small (this indicates that the measurement is informative) compared to the variance of $p_0(\cdot)$. Then, the following rule that meets these requirements is proposed:

$$\alpha = \frac{\gamma \text{tr}(\Sigma_{a,0})}{\gamma \text{tr}(\Sigma_{a,0}) + (1 - \gamma) \text{tr}(\Sigma_{a,1})} \quad (3.56)$$

where $\gamma \in [0, 1]$ is a parameter that controls the weights of the traces of the covariance matrices to select α and $\text{tr}(\mathbf{A})$ denotes the trace of matrix $\mathbf{A}$. When the information about $\mathbf{a}$ measured from the data is high compared to the prior [190], which means that $\text{tr}(\Sigma_{a,1})$ is low compared to $\text{tr}(\Sigma_{a,0})$, $\alpha \rightarrow 1$. On the contrary, when the information about $\mathbf{a}$ measured from the data is low, which means that $\text{tr}(\Sigma_{a,1})$ is high compared to $\text{tr}(\Sigma_{a,0})$, $\alpha \rightarrow 0$, and the TUKF boils down to a UKF. The steps of the TUKF are shown in Table 3.1 and the RMS error for the example using $\gamma = 0.1$ is given in Figure 3.4(c). Comparing this figure with Figures 3.4(a) and (b), it is clear that the TUKF with $\alpha = 1$ approximates the TKF under AP1, AP2 and AP3, the UKF is approximating the KF and the TUKF with adaptive α is approximating the TKF. It can be seen that the extra degree of freedom provided by adaptive selection of α enables the TUKF to significantly outperform the UKF. The key to the close correspondence between the performances of the TUKF and the TKF is that the TUKF seeks to approximate the TKF only in conditions where the TKF performs significantly better than the KF, i.e., when the measurements are informative. When the measurements are uninformative, the TUKF seeks to approximate the KF which performs similarly to the TKF (without approximations).

3.4.3.4 Single point TUKF

In this section, a slight modification of the TUKF is presented. It should be noted that the TUKF really improves the estimate given by conventional KF-type algorithms when the measurement is very informative with respect to the prior. In this case, the region $I_{\mathbf{a}}(\mathbf{z})$ is very small. As the support of the truncated prior is

Table 3.1: Truncated unscented Kalman filter update phase steps

- Apply a UKF to $p_0(\cdot)$ (characterised by $\mathbf{x}_{p,0}$ and $\mathbf{P}_{p,0}$)
 - Select the sigma points $\mathcal{A}_0^1, \dots, \mathcal{A}_0^{N_s}$.
 - Compute the transformed sigma points $\mathcal{Z}_0^1, \dots, \mathcal{Z}_0^{N_s}$ using (3.47).
 - Approximate $\hat{\mathbf{z}}_0$, $\mathbf{S}_0$ and Ψ_0 using (3.48), (3.51) and (3.55).
 - Calculate $\hat{\mathbf{x}}_{u,0}$ and $\mathbf{P}_{u,0}$ using (3.4) and (3.8).
- Apply a UKF to $p_1(\cdot)$ (characterised by $\mathbf{x}_{p,1}$ and $\mathbf{P}_{p,1}$)
 - Calculate $\mathbf{x}_{p,1}$ and $\mathbf{P}_{p,1}$ using (3.38) and (3.39).
 - Select the sigma points $\mathcal{A}_1^1, \dots, \mathcal{A}_1^{N_s}$.
 - Compute the transformed sigma points $\mathcal{Z}_1^1, \dots, \mathcal{Z}_1^{N_s}$ using (3.47).
 - Approximate $\hat{\mathbf{z}}_1$, $\mathbf{S}_1$ and Ψ_1 using (3.48), (3.51) and (3.55).
 - Calculate $\hat{\mathbf{x}}_{u,1}$ and $\mathbf{P}_{u,1}$ using $\hat{\mathbf{z}}_1$, $\mathbf{S}_1$ and Ψ_1 in (3.4) and (3.8).
- Calculate α using (3.56).
- Approximate the first two moments of the posterior using (3.25) and (3.26).

very small, another version of the algorithm uses a single sigma-point to represent $p_1(\mathbf{a}; \mathbf{z})$. The single sigma-point is chosen such that it matches the mean of $p_1(\mathbf{a}; \mathbf{z})$, given by (3.34). As it adds only one extra sigma-point to the collection of sigma-points of the UKF, it is referred to as single point TUKF (SP-TUKF).

In this case, the first two moments of $p_2(\mathbf{a}; \mathbf{z}) = \alpha p_1(\mathbf{a}; \mathbf{z}) + (1 - \alpha) p_0(\mathbf{a})$ are approximated by the sigma-point representation, and only one KF update is applied directly to $p_2(\mathbf{a}; \mathbf{z})$. The problem with this scheme is that if $\alpha \rightarrow 1$, the filter becomes unstable as the covariance matrix of $p_2(\cdot)$ tends to be singular. The way to solve it is to use α_{sp} rather than α such that

$$\alpha_{sp} = \alpha_{max} \cdot \alpha \tag{3.57}$$

where $\alpha_{max} < 1$ is the maximum value α_{sp} can take and α is given by (3.56). The steps of the SP-TUKF are given in Table 3.2. The computational burden of this version of the algorithm is lower than the TUKF as only one KF update is performed. However, it requires an extra parameter α_{max} and using only one extra sigma-point implies that it is not expected to attain the performance of the TUKF.

Table 3.2: Single point truncated unscented Kalman filter update phase steps

- Select the sigma points $\mathcal{A}_0^1, \dots, \mathcal{A}_0^{N_s}$ (using $\mathbf{x}_{p,0}$ and $\mathbf{P}_{p,0}$).
- Calculate $\boldsymbol{\mu}_{a,1}$ and $\boldsymbol{\Sigma}_{a,1}$ using (3.34) and (3.35).
- Calculate α_{sp} using (3.57).
- Approximate $p_2(\mathbf{a}; \mathbf{z})$ by the sigma points $(\mathcal{A}_2^1, \dots, \mathcal{A}_2^{N_s+1}) = (\mathcal{A}_0^1, \dots, \mathcal{A}_0^{N_s}, \boldsymbol{\mu}_{a,1})$ and weights $(w_2^1, \dots, w_2^{N_s+1}) = ((1 - \alpha_{sp}) w^1, \dots, (1 - \alpha_{sp}) w^{N_s}, \alpha_{sp})$.
- Approximate $\boldsymbol{\mu}_{a,2}$ and $\boldsymbol{\Sigma}_{a,2}$ using $(\mathcal{A}_2^1, \dots, \mathcal{A}_2^{N_s+1})$ and $(w_2^1, \dots, w_2^{N_s+1})$ as indicated in Appendix A.7.
- Approximate $\mathbf{x}_{p,2}$ and $\mathbf{P}_{p,2}$ using $\boldsymbol{\mu}_{a,2}$ and $\boldsymbol{\Sigma}_{a,2}$ and (3.38) and (3.39) applied to $p_2(\cdot)$ instead of $p_1(\cdot)$.
- Compute the transformed sigma points $\mathcal{Z}_2^1, \dots, \mathcal{Z}_2^{N_s+1}$ using (3.47).
- Approximate $\hat{\mathbf{z}}_2$, $\mathbf{S}_2$ and $\boldsymbol{\Psi}_2$ using (3.48), (3.51) and (3.55).
- Calculate $\hat{\mathbf{x}}_{u,2}$ and $\mathbf{P}_{u,2}$ using $\hat{\mathbf{z}}_2$, $\mathbf{S}_2$ and $\boldsymbol{\Psi}_2$ in (3.4) and (3.8).

3.4.4 Discussion on the TUKF

It should be kept in mind that, despite the similarity in nomenclature, the proposed method differs in a fundamental way from other Gaussian approximations such as the EKF, UKF and CKF. These filters all seek to approximate the KF recursion by an accurate approximation of (3.5)-(3.7). As demonstrated in Section 3.2 this is not always a desirable goal. Instead, the proposed method is based on the approximation of the TKF, which applies the KF recursion to a truncated prior. The impossibility of applying the TKF in general leads to a practical algorithm for adaptively switching between TKF and KF approximations. The use of the TKF approximation when warranted, i.e., when measurements are sufficiently precise, results in considerable performance improvement compared to KF approximations, as will be seen in Section 3.5.

Its main limitation though is that is not such a general tool as conventional KF-type algorithms, which can always be used, as the current form of the algorithm requires $\mathbf{h}(\cdot)$ to be a bijective function of some elements of the state. Nevertheless, if $\mathbf{h}(\cdot)$ is not bijective the posterior could be multimodal and none of the conventional KF-type algorithms would provide an accurate representation of the posterior, as they represent a unimodal distribution.

How to generalise the algorithm when $\mathbf{h}(\cdot)$ is not bijective is well worth exploring and will be a topic for future research. For some types of non-bijectivity, there are several disconnected regions of the state space that are likely to produce a given measurement. Then, one would need to truncate $p_0(\cdot)$ in each one of these regions, apply a KF to each truncated distribution and represent the posterior as a Gaussian mixture. For other types of non-bijectivity, there may be one quite large connected region and another approach should be taken.

In this chapter, $p_0(\cdot)$ and $p_1(\cdot)$ are approximated using the sigma-points given by the UKF. However, if the CKF or other methods were used, the foundations of the algorithm would be the same and it would only imply minor changes. The UKF has been chosen on account of the fact that it is the method that is used more widely.

The TUKF performs UKF updates for the components $p_1(\cdot)$ and $p_0(\cdot)$ of the mixture $p_2(\cdot)$ given in (3.24) and then combines the results. An alternative is to perform a single UKF update using the combined sigma points for $p_0(\cdot)$ and $p_1(\cdot)$, as is done in the SP-TUKF when $p_1(\cdot)$ is represented by a single point. The problem of determining which of these approaches is preferable is a topic for future research.

It should also be mentioned that there are some works that deal with Kalman filtering when the prior is truncated or there are inequality constraints [58, 180]. However, their approach and motivation have nothing to do with the TKF. In [180], the posterior is truncated because of the inequality constraints of the model. They apply a conventional KF with a linear/Gaussian model and, then, they calculate the first two moments of the truncated distribution. In [180], they provide an example with nonlinear measurement function but they linearise the measurement function like an EKF. In [58], the prior is truncated but they use linear/Gaussian models. Therefore, these methods do not try to address the problems of KF with nonlinear measurement models, they apply a conventional KF when either the prior or the posterior is truncated.

3.5 Numerical examples

In this section, four numerical examples are analysed to show the benefits of using the TUKF. The first example is the range-bearing scenario introduced in Section 3.2. A detailed analysis on how the algorithms approximate the posterior in the update phase without accounting for filtering is provided. The second example shows an update phase in a one dimensional case so that the posterior mean and variance are easily plotted. The third example deals with a tracking scenario with the range-bearing measurement model and, lastly, a univariate nonstationary growth model is

Figure 3.5: The TUKF and the SP-TUKF approximate the first two moments of the posterior much better than conventional KF-type algorithms.

analysed. The last two examples are included to show the performance benefits of the TUKF in filtering scenarios. In all the simulations in this section, the TUKF with $\gamma = 0.1$ and the SP-TUKF with $\alpha_{max} = 0.8$ are used.

3.5.1 Range-bearing case

In this case, the range-bearing scenario introduced in Section 3.2 is revisited to highlight the shortcomings of Kalman filtering with nonlinear measurement models. For the sake of illustration, Figure 3.1 is redrawn in Figure 3.5 but adding the ellipses corresponding to the TUKF and the SP-TUKF. It is clear that the TUKF approximation of the first two moments of the posterior is much more accurate than the rest of the algorithms.

Now this scenario is analysed more deeply plotting the Kullback-Leibler (KL) divergence assuming the statistics are Gaussian (for more details go to Appendix A.9) between the posterior PDF and the Gaussian approximations obtained by the different algorithms. For the purposes of this comparison, an accurate approximation to the posterior is obtained by drawing $2 \cdot 10^4$ samples from the prior and weighting the samples according to the likelihood.

The KL divergence with respect to the posterior given by Bayes' rule is estimated via Monte Carlo simulation with $2 \cdot 10^4$ runs. The KL divergence is obtained for several positions of the prior mean, $\hat{\mathbf{x}}_{p,0}$, and several true locations of the target to analyse the influence of the prior mean and the real location of the target on perfor-

Figure 3.6: Labelling of the true positions of the target around the prior mean. In order to assess the algorithms' performances, nine points around the prior mean are defined to place the target. In the simulations, these nine points form a square grid whose side is 5 m.

mance. In order to plot the results, nine points forming a square grid whose side is 5 meters around the prior mean are labelled according to Figure 3.6. Three scenarios, each one with a different prior mean $\hat{\mathbf{x}}_{p,0}$, are analysed: $[10, 10]^T$, $[20, 20]^T$ and $[30, 30]^T$. These scenarios are chosen to assess the effect of measurement precision. The rest of the parameters are those indicated in Section 3.2.

The approximated first two moments of $p_1(\cdot)$ are given in Appendix A.8. The trace of the covariance matrix of $p_1(\cdot)$, which is needed to obtain α , can be calculated using (3.35):

$$\text{tr}(\Sigma_{a,1}) = \sigma_r^2 + \sigma_\theta^2 z_r^2 \quad (3.58)$$

Then, in this example, the amount of information the measurement carries depends only on z_r^2 . When z_r is low, i.e., the target is closer to the radar, conventional KF-type algorithms approximate the posterior worse than when the target is far afield.

The KL divergence plotted against the true target position, as labelled in Figure 3.6, are shown in Figures 3.7(a), (c) and (e) for scenarios 1, 2 and 3, respectively. Note that the scaling of the y -axis changes for each scenario. The TUKF and the SP-TUKF have better performance than the MCKF, the UKF and the CKF in all the scenarios. The EKF performance is quite variable, sometimes performs quite well while others its performance is the worst one. There is no doubt that this is not a desirable property of an algorithm. In the first scenario, the MCKF has worse performance than the UKF and the CKF on the whole. Besides, the performance of these three filters is far from the performance attained by the TUKF and the SP-TUKF as in this case the target is located close to the origin and the measurement carries a lot of information. In the second scenario, the TUKF works

better than the rest of the algorithms in spite of the fact that the measurement carries less information than in the first scenario. Finally, in the third scenario, the TUKF and the SP-TUKF outperform all the conventional KF-type algorithms but their differences have been reduced as the measurement does not carry so much information as before. In addition, the performance of the TUKF is usually higher than the performance of the SP-TUKF except for the third scenario.

It should also be pointed out that the MCKF is usually outperformed by the UKF and the CKF in these cases. When the number of samples of the MCKF tends to infinity, its updated mean and covariance matrix converge to the updated mean and covariance matrix of the KF (given by the integrals (3.5), (3.6) and (3.7)). However, in nonlinear systems a close approximation to the KF does not necessarily imply a close approximation to the posterior PDF. This is clearly evident in Figure 3.5.

The RMSE plotted against the true target position, as labelled in Figure 3.6, are shown in Figures 3.7(b), (d) and (f) for scenarios 1, 2 and 3, respectively. Note that the scaling of the y -axis changes for each scenario. The estimate to calculate the RMSE for Bayes' rule is obtained using the mean of the posterior PDF. As happened with the KL divergence, the RMSE of the EKF is quite variable. In the first scenario, Figure 3.7(b), the RMSE for the TUKF and the SP-TUKF are generally much lower than the RMSE of the rest of the filters as the information of the measurement is high and is quite close to the RMSE attained by Bayes' rule. In the second scenario, Figure 3.7(d), the RMSE for the TUKF and the SP-TUKF are lower than the RMSE of the rest of the filters but the differences are closer as the measurement is less informative. Lastly, in scenario 3, Figure 3.7(f), where the measurement is much less informative, all the filters have roughly the same RMSE although the TUKF and the SP-TUKF posteriors are closer to the real posterior as pointed out when analysing the KL divergence, Figure 3.7(e). It can be confusing that the posterior mean, which is given by Bayes' rule, is not always the best estimator for some values of the state. However, the posterior mean is the MMSE estimator conditioned on a value of the measurement not the true value of the state, see Section 3.2, so its performance does not have to be the best one for a given value of the state. In fact, a blind estimator that estimates always the same value regardless of the measurement will achieve zero error when the true value of the state coincides with this blind estimate although its performance is very poor in general. To clarify this, the averaged RMSE for all the algorithms is shown in Table 3.3. As expected, the lowest error is given by Bayes' rule followed by the TUKF and the SP-TUKF and, in general, the EKF error is the highest one.

Figure 3.7: Range-bearing update: (a) KL divergence and (b) RMSE for scenario 1. (c) KL divergence and (d) RMSE for scenario 2. (e) KL divergence and (f) RMSE for scenario 3. The TUKF and the SP-TUKF vastly outperform the conventional KF-type algorithms in scenario 1. In scenario 3, all the algorithms give roughly the same RMSE although the KL divergence is slightly lower for the TUKF and the SP-TUKF as the measurement has much less information than in scenarios 1 and 2.

Table 3.3: Averaged RMSE for range-bearing scenarios

	MCKF	UKF	CKF	EKF	SP-TUKF	TUKF	Bayes' rule
Scenario 1	3.85	3.89	3.91	4.46	1.48	1.25	1.14
Scenario 2	2.09	2.11	2.09	2.19	1.43	1.45	1.42
Scenario 3	2.01	2.02	2.01	2.05	1.78	1.79	1.77

3.5.2 Update analysis in one dimension

The model analysed here represents an amplitude measurement based on the position x of a target assuming the sensor is located at the origin of the coordinate system. The objective is to estimate this position based on the measurement and the prior distribution, which is Gaussian with mean $\hat{x}_{p,0}$ and variance $P_{p,0}$. The measurement noise is zero-mean Gaussian with variance R . The nonlinear function of the measurement model in (3.1) is

$$h(x) = \begin{cases} A_0 & \text{if } 0 < x \leq d_0 \\ A_0 d_0^n / x^n & \text{if } x > d_0 \\ 0 & \text{elsewhere} \end{cases} \quad (3.59)$$

where A_0 is the saturation amplitude, d_0 is the saturation distance and n is the signal decay exponent. Note that, according to (3.59), negative values of x produce a signal of zero amplitude. This would represent a sensor that is only capable of measuring the amplitude of a signal produced by a target coming from a certain direction. Note that the amplitude measurement is presented in one dimension as the measurement function $h(\cdot)$ needs to be bijective at least in a region of the state space, $x > d_0$. In this case, the state is a scalar so the state is not partitioned as in Sections 3.3 and 3.4 simplifying the calculations.

The approximated first two moments of $p_1(\cdot)$ for a measurement $z > 0$ are given in Appendix A.8. If $z \leq 0$ then $\alpha = 0$ and the UKF is applied. Otherwise, if $\tilde{x}(z) > d_0$ and using (3.56), the parameter α is²:

$$\alpha = \frac{\gamma A_0^2 d_0^{2n} n^2 P_{p,0}}{\gamma A_0^2 d_0^{2n} n^2 P_{p,0} + (1 - \gamma) R \tilde{x}(z)^{2n+2}} \quad (3.60)$$

For different values of the true value of x , the posterior mean and variance and the KL divergence with respect to Bayes' rule are approximated using the UKF, the CKF, the SP-TUKF and the Monte Carlo KF (MCKF) using Monte Carlo simulation with 500 runs. The parameters used in the simulation are those shown in Table 3.4.

²For simplicity, if $0 < \tilde{x}(z) < d_0$, the same rule is used.

Table 3.4: Parameters of the 1-D simulation

Parameter	Value
n	1
A_0	100
d_0	0.1
R	1
$\hat{x}_{p,0}$	0
$P_{p,0}$	36

Figure 3.8: Kullback-Leibler divergence plotted against the real value of x : The TUKF provides the closest approximation of the posterior given by Bayes' rule.

The KL divergence with respect to Bayes' rule for the algorithms is shown in Figure 3.8. The TUKF and SP-TUKF approximations are much closer to the real posterior PDF than the MCKF, the UKF and the CKF. Besides, the MCKF and UKF have approximately the same performance and the CKF does not perform properly in this scenario. In addition to the KL divergence, it is also useful to look at the posterior mean and variance, shown in Figures 3.9 and 3.10. When the value of x is small, the TUKF and SP-TUKF posterior means are much closer to the real posterior mean than the means given by the MCKF and the UKF. The expected mean of the CKF in this case is very far from the real one. Besides, for small values of x , the posterior variance given by the UKF and the MCKF are very high while the true posterior variance is actually very small. This implies that conventional Kalman-filter-type algorithms do not use the information contained in the measurement properly (when x is small) as the uncertainty of the true posterior is rather small but Kalman's posterior variance is high. When x is high, the algorithms have a roughly similar performance. This is due to the fact that the TUKF and the SP-TUKF tend to behave as the UKF because the measurement is not very informative and α is low. In general, the TUKF outperforms the SP-TUKF as expected.

Figure 3.9: Posterior mean plotted against the real value of x : The TUKF and SP-TUKF posterior means represent the real posterior mean more reasonably than conventional KF-type algorithms.

Figure 3.10: Posterior variance plotted against the real value of x : The TUKF and SP-TUKF posterior variances represent the real posterior variance more reasonably than conventional KF-type algorithms. Also, the posterior variance of all the conventional KF-type algorithms is independent of the real value of x . This is due to the fact that the approximated posterior variance of the KF is averaged over all the measurements as explained in Sections 2.6 and 3.2.

Figure 3.11: RMSE for all the algorithms plotted against the real value of x : The TUKF and the SP-TUKF provide a lower RMSE than conventional KF-type algorithms on the whole.

Table 3.5: RMS error averaged over both the parameter and the measurement

Algorithm	RMS error
SP-TUKF	5.67
TUKF	5.62
UKF	5.96
CKF	34.35
MCKF	5.94
Bayes' rule	5.22

Finally, the RMS error for all the algorithms plotted against the real value of x is shown in Figure 3.11 and the RMS error averaged over both the parameter and the measurement for the algorithms is shown in Table 3.5. As expected, Bayes' rule outperforms the rest of the algorithms followed by the TUKF and the SP-TUKF.

3.5.3 Target tracking with range-bearing measurements

The previous examples highlighted the improvement in performance of the TUKF and the SP-TUKF over conventional KF-type algorithms only in the update step. In this example, the performance benefits are shown in a dynamic system. This example analyses a target tracking scenario using range-bearing measurements from a radar located at the origin of the coordinate system. The state vector at time k is $\mathbf{x}^k = [\mathbf{a}^k, \mathbf{b}^k]^T$ where $\mathbf{a}^k$ is the position vector and $\mathbf{b}^k$ is the velocity vector. The measurement model is given by (3.10) and (3.11) but bearing in mind that the position vector in this example is $\mathbf{a}^k$.

Figure 3.12: Scenario for the range-bearing tracking: (a) Target trajectory: The initial target position is represented by a blue circle. The target position and direction of movement every 10 time steps are represented by arrows. The radar location is represented by a red cross. (b) Distance from the target to the radar at each time step.

The dynamic model of the target is the nearly-constant velocity model [13]:

$$p(\mathbf{x}^{k+1} | \mathbf{x}^k) = \mathcal{N}(\mathbf{x}^{k+1}; \mathbf{F} \cdot \mathbf{x}^k, \mathbf{Q}) \quad (3.61)$$

$$\mathbf{F} = \begin{pmatrix} 1 & \tau \\ 0 & 1 \end{pmatrix} \otimes \mathbf{I}_2 \quad (3.62)$$

$$\mathbf{Q} = \sigma_u^2 \begin{pmatrix} \tau^3/3 & \tau^2/2 \\ \tau^2/2 & \tau \end{pmatrix} \otimes \mathbf{I}_2 \quad (3.63)$$

where $\mathcal{N}(\mathbf{x}; \bar{\mathbf{x}}, \mathbf{Q})$ is the Gaussian PDF evaluated at $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\mathbf{Q}$, $\otimes$ is the Kronecker product, τ is the sampling period and σ_u^2 is the continuous-time process noise intensity [13].

The scenario used to evaluate the performance of the algorithms is represented in Figure 3.12(a). The distance from the radar to the target at each time step is shown in Figure 3.12(b). This is important because conventional KF-type algorithms are expected to perform worse than the TUKF when the target gets close to the radar as explained in Section 3.5.1. The sampling period of the trajectory is $\tau = 1$ s, $\sigma_u = 10 \text{ m/s}^{3/2}$ and there are $l = 140$ time steps in the simulation. This trajectory corresponds to one realisation of the dynamic system described by (3.61). The tracking performance is evaluated calculating the RMS of the position error by a Monte Carlo simulation with $m = 200$ realisations. To assess the performance of the algorithms, each Monte Carlo run has a different sequence of measurements but

the j th Monte Carlo run for all the algorithms uses the same sequence. This is a variance reduction technique known as common random numbers. The use of this technique is beneficial to compare the algorithms as a lower number of Monte Carlo runs is needed [13, 171]. This technique is used in all the chapters of this thesis where Monte Carlo simulation is used to compare different algorithms.

The prior PDF of the target's state is:

$$\mathbf{x}^0 \sim \mathcal{N}(\mathbf{x}^0; \bar{\mathbf{x}}^0, \mathbf{Q}) \quad (3.64)$$

where, in each Monte Carlo run, $\bar{\mathbf{x}}^0$ is drawn from a Gaussian distribution whose mean is the real state of the target at time 0 and whose covariance matrix is $\mathbf{Q}$.

The proposed algorithms, the TUKF and the SP-TUKF, are compared with six algorithms for tracking: UKF, CKF, EKF, MCKF ($2 \cdot 10^4$ samples), Rao-Blackwellised UKF (RB-UKF) [143] and a PF based on sampling importance resampling with 10000 particles [9], whose steps are given in Table 2.3 in Chapter 2. The parameters of the algorithms are the same as in the previous examples. The RMS position plotted against time is shown in Figure 3.13(a). As expected, EKF, UKF, CKF, MCKF and RB-UKF perform poorly when the target gets close to the radar at around time steps 40 and 80, see Figure 3.12(b). This is due to the fact that the likelihood function is very narrow compared to the prior and conventional KF-type algorithms do not perform properly. In a dynamic system, this can lead to filter divergence as happens at around time 80. In this case, conventional KF-type algorithms manage to recover the track around 10 time steps later. However, the TUKF and the SP-TUKF provide a much more accurate estimate of the trajectory as they can deal with situations when the information given by the measurement is very high. They do it by adaptively calculating the parameter α , which identifies when the measurement carries a lot of information and conventional KF-type algorithms fail. The value of α plotted against time is shown in Figure 3.13(b). When the distance from the radar to the target increases, α decreases, as the information provided by the measurement is lower and conventional KF-type algorithms perform well. As expected, the PF also attains a low error at all time steps but with a much higher computational expense.

The robustness of the algorithms with respect to γ is shown in Table 3.6. One should note that for the SP-TUKF, $\gamma = 1$ does not mean that the information of the prior is rejected, as with the TUKF, because α_{sp} cannot be one, see (3.57). Recall that $\gamma \in [0, 1]$ is used in the calculation of the weight α given to the truncated prior. A low value of γ tends to reduce the weight given to the truncated prior while a high value of γ tends to place more emphasis on the truncated prior. The results of

Figure 3.13: (a) RMS position error plotted against time. When the target is close to the radar (around time steps 40 and 80), the SP-TUKF and the TUKF vastly outperform conventional KF approximations. (b) Value of α plotted against time for the TUKF. When the target gets closer to the radar, α increases.

Table 3.6: Time-averaged RMS position error for different values of γ (range-bearing)

Algorithm	γ										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
SP-TUKF	8.48	5.85	5.93	5.98	6.01	6.03	6.05	6.07	6.09	6.10	6.12
TUKF	8.48	5.74	5.84	5.94	6.02	6.09	6.16	6.22	6.28	6.33	6.38
UKF						8.19					
CKF						8.02					
MCKF						8.04					
EKF						6.48					
RB-UKF						8.48					
PF						5.70					

Table 3.6 show that the performances of the SP-TUKF and the TUKF do not vary greatly for $\gamma > 0$ although slightly better performance is obtained for smaller values of γ . It should be noted that even with small values of γ the weight α given to the truncated prior can be large, provided the measurements are sufficiently precise. This can be seen in Figure 3.13(b) which shows that large values of α are selected even with $\gamma = 0.1$. Also, Table 3.6 shows that the performance of the TUKF with $\gamma = 0.1$ is very close to the PF performance.

The averaged RMS position error over time for different measurement noise parameters is included in Table 3.7 where the cases in which the error is much higher than for the PF have been underlined. Cases in which the measurement is very informative and others in which it is very uninformative are analysed to

show that our method is robust in all situations. Firstly, one should note that SP-TUKF and TUKF are very close to PF performance especially for informative measurements. They outperform the conventional KF-type algorithms in all cases, especially, when the measurement is informative. This demonstrates the power of both the SP-TUKF and the TUKF in nonlinear filtering problems. When the measurement is very uninformative, cases: 2, 5 and 8, all filters behave similarly except for the EKF that gives higher errors. Moreover, among the conventional KF-type algorithms MCKF has the most robust performance as it never diverges and EKF diverges more frequently than the rest.

Table 3.7: Time-averaged RMS position error for different measurement noise parameters

Case	σ_r (m)	σ_θ (rad)	SP-TUKF	TUKF	UKF	CKF	MCKF	EKF	RB-UKF	PF
1	1	$0.2\pi/180$	1.19	1.18	5.10	<u>136.57</u>	5.76	<u>192.12</u>	<u>128.08</u>	1.34
2	1	$20\pi/180$	38.29	37.62	38.33	<u>37.98</u>	38.28	<u>50.15</u>	<u>37.93</u>	33.21
3	$\sqrt{10}$	$0.2\pi/180$	3.19	3.18	<u>161.11</u>	<u>222.31</u>	7.35	<u>99.47</u>	<u>251.3</u>	3.26
4	$\sqrt{10}$	$2\pi/180$	6.55	6.45	<u>11.26</u>	<u>9.86</u>	9.36	<u>7.35</u>	<u>10.21</u>	6.34
5	$\sqrt{10}$	$20\pi/180$	38.58	37.82	38.71	38.24	38.51	44.51	38.35	33.31
6	$\sqrt{0.1}$	$0.2\pi/180$	0.71	0.70	4.70	3.19	5.27	<u>128.31</u>	2.78	0.96
7	$\sqrt{0.1}$	$2\pi/180$	5.77	5.66	7.27	7.93	8.24	<u>5.92</u>	6.54	5.86
8	$\sqrt{0.1}$	$20\pi/180$	38.25	37.61	38.31	37.86	38.24	51.32	37.91	34.22

3.5.4 Univariate nonstationary growth model (UNGM)

The univariate nonstationary growth model (UNGM) is characterised by:

$$p\left(x^k \mid x^{k-1}\right) = \mathcal{N}\left(x^k; f^k\left(x^{k-1}\right), Q\right) \quad (3.65)$$

$$f^k\left(x^{k-1}\right) = \frac{x^{k-1}}{2} + \frac{25x^{k-1}}{1 + (x^{k-1})^2} + 8 \cos(1.2k) \quad (3.66)$$

$$p\left(z^k \mid x^k\right) = \mathcal{N}\left(z^k; \frac{(x^k)^3}{20}, R\right) \quad (3.67)$$

This model is analysed in [9, 113] but in this case the measurement function depends on $(x^k)^3$ rather than $(x^k)^2$ so that it is bijective. The same parameters as in [113] are used. That is, $Q = 1$, $R = 1$, data was generated using $x^0 = 0.1$ and the prior PDF at time step 0 is $x^0 \sim \mathcal{N}(x^0; 0, 1)$.

The simulation has 100 time steps and the true state for these time steps is plotted in Figure 3.14. The performance of the algorithms is analysed using Monte Carlo simulation with 200 runs. It should be noticed that, in this case, the measurement function, given by (3.67), depends on the whole state so the RB-UKF cannot be applied here. The parameters of the algorithms are the same as in the previous examples. The approximated first two moments of $p_1(\cdot)$ are given in Appendix A.8.

The RMS error plotted in log scale against time for the algorithms is shown in Figure 3.15(a). Only the first twenty time steps have been represented for the clarity of representation. The TUKF and the SP-TUKF have a much higher performance than conventional KF-type algorithms and their performance is close to PF's. In this case, the error given by the EKF is the highest one. The value of α against time for the first twenty time steps for the TUKF is plotted in Figure 3.15(b). Comparing Figure 3.15(b) with Figure 3.14, it can be seen that α is larger when $|x^k|$ increases as the measurement is more informative.

The averaged RMS position error over time is shown in Table 3.8. Again, highest performance is provided by the PF closely followed by the TUKF with $\gamma = 0.1$. The EKF, MCKF, UKF and CKF are far from the performance of the SP-TUKF and the TUKF.

The averaged RMS position error over time for different measurement noise parameters is included in Table 3.9. As happened in the previous example, the TUKF and the SP-TUKF perform much better than the conventional KF-type algorithms.

3.5 NUMERICAL EXAMPLES

Figure 3.14: True state for the univariate nonstationary growth model

Figure 3.15: UNGM scenario for the first 20 time steps: (a) RMS error plotted in log scale against time: The TUKF and the SP-TUKF have a much higher performance compared to conventional KF-type algorithms. (b) Value of α for the TUKF: When $|x^k|$ increases (see Figure 3.14), α increases as the measurement is more informative.

Table 3.8: Time-averaged RMS position error for different values of γ (UNGm)

Algorithm	γ										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
SP-TUKF	3.86	0.81	0.86	0.91	0.95	0.99	1.03	1.06	1.10	1.15	1.22
TUKF	3.86	0.64	0.69	0.75	0.81	0.88	0.94	1.01	1.08	1.16	1.27
UKF						3.86					
CKF						7.46					
MCKF						3.84					
EKF						26.94					
PF	0.61										

Table 3.9: Time-averaged RMS position error for different measurement noise parameters

R	SP-TUKF	TUKF	UKF	CKF	MCKF	EKF	PF
0.01	0.42	0.34	11.58	28.79	3.62	369.48	0.32
0.1	0.60	0.48	5.54	10.95	4.67	261.77	0.45
10	1.13	0.96	4.00	6.76	3.94	18.84	0.9
100	1.58	1.51	4.15	4.53	4.11	14.85	1.28

3.6 Conclusions

A new approach has been developed to approximate the first two moments of the posterior in nonlinear systems with a bijective measurement function. It does not aim to reach the performance of particle filters but to improve the performance of conventional Kalman-filter-type algorithms requiring low computational burden. Its foundations are to approximate the PDF of the measurement noise by a truncated one and approximate Kalman filter equations applied to a mixture of the prior and a truncated version of the prior. On the whole, this approach should achieve better performance than any conventional Kalman-filter-type algorithms if the system sometimes provides very informative measurements compared to the prior. As demonstrated in this chapter, if the measurement is very informative, conventional Kalman-filter-type algorithms do not work properly.

The drawback of the method is that it is not such a general tool as conventional Kalman-filter-type algorithms as the current form of the algorithm requires the measurement function to be bijective. However, this approach could be generalised in principle when the measurement function is not bijective by truncating the noise and finding the regions of the state space in which the likelihood is non-zero.

It should be mentioned that the TUKF will be used as part of a novel algorithm for tracking multiple targets with unknown sensor positions to be described in Chapter 6.

Chapter 4

Detection and tracking of multiple targets

4.1 Introduction

This chapter develops Bayesian filtering algorithms to track a variable and unknown number of targets. In Chapter 2, several general Bayesian filtering algorithms were explained and, consequently, in principle they could be adapted to this problem once it is posed in a Bayesian framework. However, the specific characteristics of multiple target tracking problems imply that the previously mentioned algorithms do not deal with this situation properly. This introduction firstly comments on traditional techniques to perform multiple target tracking. Then, it is explained how Bayesian techniques have been used and what is proposed in this chapter to improve on previously developed methods.

Multiple target tracking has usually been carried out by means of the technique known as multiple hypothesis tracking (MHT) [15]. This traditional technique is based on filtering a set of detections or thresholded measurements typically using Kalman-filter-type algorithms. The detections are used to update existing tracks or create new tracks, or they can be deemed false alarms as well. In this case, data association techniques must be used to determine the set of measurements associated to each track. However, this technique has one main drawback: its tracking performance is poor when there are targets which produce a low signal-to-noise ratio because of the severe quantization of the measurements. Track-before-detect techniques use raw sensor data and significantly improve the performance in this kind of situation [21, 48]. Adopting a fully Bayesian perspective to multiple target tracking, raw data is easily made available to the tracker, thus, improving tracking performance, and providing a recursive solution to the

problem [116–118, 139, 141, 142, 161]. In addition, the Bayesian approach can also be used to deal with thresholded measurement models that require a data association step such as the algorithms given in [94, 158, 183, 195, 197, 198]. The problem addressed in this chapter consists of a sensor network that can report raw measurements. Since the algorithms of [94, 158, 183, 195, 197, 198] are predominantly concerned with issues which arise due to data association in thresholded measurement models, they cannot be directly applied to this problem.

As indicated in Chapter 2, in the Bayesian approach to dynamic state estimation, the aim is to construct the posterior probability density function (PDF) of the state based on all available information. This chapter is concerned with PF approximations of the posterior PDF for multitarget tracking. In this context, timely and reliable detection of targets and accurate target state estimation require an importance density that efficiently samples both target states and the number of targets. Although careful design of the importance density is necessary for any importance sampling approximation, the potentially high dimension of the multitarget state, which is composed of all the individual target states, make this especially crucial in multitarget tracking [45]. Another problem in tracking an unknown number of targets is to maintain correspondence of target states over time so that tracks can be built. The following paragraphs discuss existing work on these problems and summarise how the proposed methods address the limitations of these approaches.

An important property of the measurement model commonly used in multitarget tracking is the approximate factorisation of the likelihood. This factorisation depends on the measurement model but usually applies for widely separated targets. As a result of this property only the states of targets which form clusters based on proximity need to be sampled jointly. This effectively reduces the dimensionality of the sampling problem. This idea was first exploited for sequential importance sampling in [161] and also used in [34, 116–118, 121, 141]. Although distant targets can be ignored it is desirable that samples of targets within a cluster be sampled jointly. This was achieved in [141] by jointly sampling clusters of target states from the optimal importance density (OID) [55]. The OID is not universally applicable because it is generally intractable although approximations to the OID using a bank of extended or unscented Kalman filters have been proposed [55, 169, 197]. However, jointly sampling from the OID, or an approximation, can be computationally expensive, even for relatively small clusters of targets. The aim is therefore to construct an importance density which considers the presence of neighbouring targets in a computationally efficient manner. The proposed parallel partition method (PP) achieves this aim by a simple modification of the independent partition method (IP) of [116]. PP is intended to improve the efficiency of sampling of the target states,

especially when the number of particles is low and a fast algorithm is required. In addition, it is shown in this chapter that what the well-known IP¹ method really does is sampling an auxiliary vector, therefore, providing more insight into multiple target tracking using PFs.

Efficient sampling of the target number is also vital in multitarget tracking. The simplest possibility is to draw the target number from an assumed prior [23]. This is highly inefficient as it removes and add targets regardless of what the measurements suggest is appropriate. The notion of using a secondary approximation to the posterior to construct an importance density for sampling target number was used in [61, 141]. There, an existence grid was used to determine both regions likely to contain new targets and existing targets which are likely to be false. This approach has the advantage of using the available measurements rather than relying on a prior but does not incorporate target dynamics and is therefore inappropriate for even moderately rapid targets. In this chapter, a PF approximation, referred to as the pre-track filter, is used to construct an importance density for the addition of new tracks. Sampling of target number is performed via likelihood ratio tests (LRTs) on the pre-tracks and existing tracks. LRTs are commonly used for track scoring in the well known MHT [15]. The LRT for the pre-tracks governs the addition of new targets, effectively transforming a pre-track to an existing track if appropriate, while the LRT for the existing tracks determines target removal. Then, the PF developed in this chapter has a two-layer scheme in which the posterior density of one layer is used to specify the importance density of the other layer to propose new targets. A similar idea, although not in the context of importance sampling, is used in the integrated probabilistic data association filter (IPDAF) and its generalisations where a target existence probability is used to determine the track status [150]. Nevertheless, the idea of using an subsidiary PF to sample the new targets is a new idea developed in this chapter.

The goal of tracking is to build up temporal sequences of target state estimates which represent the motion of targets over time. If the measurements provide information about target identity the correspondence between target state estimates at different times can be established reasonably easily. However, even if the measurements do not provide direct information about target identity, the prior dynamics can aid in maintaining correspondences between individual target state estimates at different times thus allowing tracks to be constructed. In [116, 117, 141], the joint multitarget probability density (JMPD) is proposed. The JMPD technique simultaneously estimates both the target states and the number of targets based on the

¹In this thesis, the IP method always refers to the IP method in [116]. The IP method in [161] is slightly different.

measurements. Hence, the track of a target is not explicitly identified from its beginning until its end. A similar comment applies to the usual random finite set (RFS) formulation of multiple target tracking [132, 199, 201]. This is not problematic if one is only interested in estimating the number of targets with their corresponding states at each time step. The issue of track construction was addressed in [141] by adding discrete-valued labels in the PF implementation. Here a more principled approach is adopted so that the labels are attached to the target state in the original formulation of the problem. The quantity of interest is now the number of targets and the concatenated vector containing the individual target states and corresponding labels. The posterior distribution of this quantity is referred to as the joint multitrack probability density (JMKPD). This modelling approach fits into the framework of [146]. The RFS framework can also be extended to permit track construction through the addition of labels [131]. A detailed comparison between vector and set notations can be found in Appendix B.

The rest of the chapter is organised as follows. In Section 4.2, the JMKPD and the target and sensor models are described. In Section 4.3, the PP method for known number of targets is explained, while it is generalised for unknown number of targets in Section 4.4. The comparison of the PP method with other algorithms is included in Section 4.5.

4.2 Joint multitrack probability density

This section deals with the JMKPD, the target model and the sensor model. Basically, the JMKPD corresponds to the approach used in [146] but augmenting each target state with a unique label so that the tracks are perfectly identified along time. Therefore, its mathematical correctness is justified as it is based on the one in [146]. Nevertheless, adding the unique labels to each track enables us to use an equivalent representation so that the required PDFs are written in a more compact form. This representation is one of the novelties introduced in this chapter.

Let t^k denote the number of tracks at time k . The dynamic state of track j is described at time k by vector $\mathbf{x}_j^k = [x_j^k, \dot{x}_j^k, y_j^k, \dot{y}_j^k]^T \in \mathbb{R}^4$ where $[x_j^k, y_j^k]^T$ is the position vector, $[\dot{x}_j^k, \dot{y}_j^k]^T$ is the velocity vector and superscript T means transpose. The augmented state of track j is described at time k by vector $\mathbf{r}_j^k = \left[(\mathbf{x}_j^k)^T, i_j^k \right]^T \in \mathbb{R}^4 \times \mathbb{N}$ where i_j^k is the label of track j . It is imposed that $i_1^k < i_2^k < \dots < i_{t^k}^k$ to ensure unique labelling of the tracks. The augmented multitrack state vector is formed by stacking the individual augmented track state vectors into one vector

$$\mathbf{R}^k = \left[\left(\mathbf{r}_1^k \right)^T, \dots, \left(\mathbf{r}_{t^k}^k \right)^T \right]^T = \left[\left(\mathbf{x}_1^k \right)^T, i_1^k, \dots, \left(\mathbf{x}_{t^k}^k \right)^T, i_{t^k}^k \right]^T.$$

Since the number t^k of targets is unknown, the variable

$$\left(\mathbf{R}^k, t^k \right) \in \bigcup_{t=0}^{\infty} \left(\mathbb{R}^4 \times \mathbb{N} \right)^t \times \{t\} : i_1^k < i_2^k < \dots < i_t^k.$$

Following [145, 146], the set $\mathbb{R}^0 = \{\theta\}$ where θ is a symbol for no target. A similar convention is used for $\mathbb{N}^0$. This can be viewed as a hybrid stochastic system where the discrete random variable t^k determines the dimensionality of $\mathbf{R}^k$ [116]. A clarification can be found in Appendix A.10.

The state at time k is observed through the measurement vector $\mathbf{z}^k$. In a Bayesian framework, estimation of $\left(\mathbf{R}^k, t^k \right)$ based on the sequence of measurements $\mathbf{z}^{1:k} = (\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^k)$ up to time k is achieved by the recursive computation of the posterior PDF $p\left(\mathbf{R}^k, t^k \mid \mathbf{z}^{1:k}\right)$. This is done in two phases: prediction and update [9]. The prediction phase makes use of the Chapman-Kolmogorov equation:

$$\begin{aligned} p\left(\mathbf{R}^k, t^k \mid \mathbf{z}^{1:k-1}\right) &= \sum_{t^{k-1}=0}^{\infty} \sum_{i_1^{k-1}=1}^{\infty} \dots \sum_{\substack{i_{t^{k-1}}^{k-1}=i_{t^{k-1}-1}^{k-1}+1 \\ i_{t^{k-1}-1}^{k-1}=i_{t^{k-1}-2}^{k-1}+1 \\ \vdots \\ i_1^{k-1}=i_0^{k-1}+1}}^{\infty} \int d\mathbf{x}_1^{k-1} \dots d\mathbf{x}_{t^{k-1}}^{k-1} \\ &\quad p\left(\mathbf{R}^k, t^k \mid \mathbf{R}^{k-1}, t^{k-1}\right) p\left(\mathbf{R}^{k-1}, t^{k-1} \mid \mathbf{z}^{1:k-1}\right) \end{aligned} \quad (4.1)$$

Note that (4.1) has the same form as the Chapman-Kolmogorov equation in [116, 141] but summing over the labels as they have been included in the multitarget state.

The update equation is:

$$p\left(\mathbf{R}^k, t^k \mid \mathbf{z}^{1:k}\right) \propto p\left(\mathbf{z}^k \mid \mathbf{R}^k, t^k\right) p\left(\mathbf{R}^k, t^k \mid \mathbf{z}^{1:k-1}\right) \quad (4.2)$$

When $t^{k-1} = 0$, $\mathbf{R}^{k-1}$ can only take the value θ , thus, it is a discrete state so the integrals in (4.1) come down to a substitution of the probability mass of that state. One should note that reordering the components of the augmented multitarget vector, calculating $p\left(\mathbf{R}^k, t^k \mid \mathbf{z}^{1:k}\right)$ is analogue to calculating $p\left(\left[\left(\mathbf{X}^k\right)^T, \left(\mathbf{I}^k\right)^T\right]^T, t^k \mid \mathbf{z}^{1:k}\right)$ where $\mathbf{X}^k = \left[\left(\mathbf{x}_1^k\right)^T, \dots, \left(\mathbf{x}_{t^k}^k\right)^T\right]^T \in \mathbb{R}^{4t^k}$ is the multitarget vector and $\mathbf{I}^k = \left[i_1^k, i_2^k, \dots, i_{t^k}^k\right]^T \in \mathbb{B}\left(t^k\right)$ is the label vector where $\mathbb{B}(t) = \{\mathbf{v} = [v_1, \dots, v_t]^T \in \mathbb{N}^t : v_1 < \dots < v_t\}$.

There is a bijective function whose domain and image are countable $f : \mathbb{B}(t) \times$

$\{t\} \rightarrow \Upsilon^t(\mathbb{N})$, where $\Upsilon^t(\mathbb{N})$ is the set of subsets of $\mathbb{N}$ of size t , such that $f(\mathbf{I}^k, t^k) = I^k$, where $I^k = \{i_1^k, i_2^k, \dots, i_{t^k}^k\}$ is the set of labels. Because of this one-to-one correspondence in a countable domain, computing $p(\mathbf{X}^k, I^k | \mathbf{z}^{1:k})$ where $(\mathbf{X}^k, I^k) \in \bigcup_{t=0}^{\infty} \mathbb{R}^{4t} \times \Upsilon^t(\mathbb{N})$ is equivalent to calculating $p(\mathbf{R}^k, t^k | \mathbf{z}^{1:k})$. Set notation will be used to denote the labels for ease of exposition. In addition, variable t^k can be dropped in the calculation of the posterior as it is implicit in the size of I^k and now, it is the size of I^k that governs the dimensionality of $\mathbf{X}^k$.

Consequently, the aim is to compute $p(\mathbf{X}^k, I^k | \mathbf{z}^{1:k})$ recursively. To this end, it is necessary to calculate (4.1) and (4.2) using a target behaviour model $p(\mathbf{X}^k, I^k | \mathbf{X}^{k-1}, I^{k-1})$ and a sensor model $p(\mathbf{z}^k | \mathbf{X}^k, I^k)$, which are analogue to their representation using the variables $(\mathbf{R}^k, t^k)$, see Appendix A.11 for more details. The target dynamic and sensor models are explained in the following subsections.

4.2.1 Multitrack model

The model takes into account the births, deaths and motions of the targets. It is assumed that the a priori evolution of the number of targets in track is modelled by an $M/M/\infty$ birth-death process [111]. This model assumes that the number of arrivals at a certain interval of time follows a Poisson distribution, the life of each track follows an exponential distribution and that the system is able to track infinite targets, i.e., it does not miss a target because the system has run out of memory. The model has two parameters: the a priori arrival rate λ of targets and the average life $1/\mu$ of a target. Then, the average number of a priori targets in track is λ/μ [111].

The a priori arrival rate of targets λ is taken to be proportional to the area A_S of the surveillance region, $\lambda = \lambda_A A_S$, where λ_A would denote the prior arrival of targets per area unit. This fact will be useful in the calculation of the final weights of the particles as explained in Section 4.4.2.

According to this model, the probability that b^k new targets appear at time k is

$$p(b^k) = \frac{e^{-\lambda\tau} (\lambda\tau)^{b^k}}{b^k!} \quad (4.3)$$

where τ is the sampling interval [199].

Due to the memoryless property of an exponential random variable [164], the probability that a target dies in a sampling interval is independent of the time the target has lived so far. Therefore, the probability that a target dies in a sampling interval is:

$$p_d = \int_0^{\tau} \mu \cdot e^{-\mu \cdot t} dt = 1 - e^{-\mu \cdot \tau} \quad (4.4)$$

Once how the number of targets evolves with time has been described, the evolution of the multitrack state is addressed. The following sets need to be defined: $S^{k-1} = \{s_1^{k-1}, s_2^{k-1}, \dots, s_{t_{k-1}-d^{k-1}}^{k-1}\}$ is the set of labels of the targets that survive from time $k-1$ to k where d^{k-1} is the number of targets that die at time $k-1$ and $N^k = \{n_1^k, n_2^k, \dots, n_{b^k}^k\}$ is the set of labels of the targets that are born at time k so $I^k = S^{k-1} \cup N^k$. Hereafter, $\mathbf{X}^{k+1}(S^k)$ denotes the components of the vector $\mathbf{X}^{k+1}$ that correspond with the set of tracks in S^k . Therefore:

$$p(\mathbf{X}^{k+1}, I^{k+1} | \mathbf{X}^k, I^k) = p(\mathbf{X}^{k+1}(S^k), S^k, \mathbf{X}^{k+1}(N^{k+1}), N^{k+1} | \mathbf{X}^k, I^k) \quad (4.5)$$

As shown in Appendix A.12, (4.5) can be written as

$$p(\mathbf{X}^{k+1}, I^{k+1} | \mathbf{X}^k, I^k) = p(N^{k+1}) p(S^k | I^k) p(\mathbf{X}^{k+1}(N^{k+1}) | N^{k+1}) \cdot p(\mathbf{X}^{k+1}(S^k) | \mathbf{X}^k, S^k) \quad (4.6)$$

Now the expressions for all the probabilities in (4.6) are given and their meaning explained. The first probability indicates how the new labels are assigned. To this end, let us partition the surveillance area $V \subset \mathbb{R}^2$ into M cells of the same size $V_1, V_2, \dots, V_M$. Assume that at most one target can appear in a given cell at a given sample time and that there are not any cells in which it is more likely that a target appears². The target appearing in the j th cell at the k th time is given the label $j + (k-1)M$. Let $F^k = \{(k-1)M + 1, \dots, kM\}$ denote the set of all possible labels that can be assigned at time k . Then

$$p(N^{k+1}) \propto \begin{cases} p(b^{k+1}) & \text{if } N^{k+1} \subseteq F^{k+1} \\ 0 & \text{otherwise} \end{cases} \quad (4.7)$$

where it should be noted that³ $|N^k| = b^k$.

The second probability in (4.6) indicates the set of labels of the targets that survive time k :

$$p(S^k | I^k) = \begin{cases} \prod_{j=1}^{t^k} [\chi_{S^k}(i_j^k) (1 - p_d) + (1 - \chi_{S^k}(i_j^k)) p_d] & \text{if } S^k \subseteq I^k \\ 0 & \text{otherwise} \end{cases} \quad (4.8)$$

where $\chi_S(B) = 1$ if $B \subseteq S$ and zero otherwise and it has been assumed that the life of a target is independent of the rest of the targets.

²Note that the model could be generalised if there were cells that are more likely to contain new targets but it would be more cumbersome. The same applies to the probability of target death.

³ $|A|$ denotes the cardinality of set A . That is, the number of elements in A .

The third probability in (4.6) indicates the a priori state of the targets that are born at time k :

$$p(\mathbf{X}^k(N^k) | N^k) = \prod_{j=1}^{b^k} U_{V_{n_j^k}}(x^k(n_j^k), y^k(n_j^k)) \cdot p_{v_0}(\dot{x}^k(n_j^k), \dot{y}^k(n_j^k)) \quad (4.9)$$

where $U_{V_{n_j^k}}(x, y)$ is a uniform random variable in the cell $V_{n_j^k}$, whose area is $A_{n_j^k} = A_S/M$ with A_S being the area of the surveillance region, evaluated at (x, y) , $p_{v_0}(\cdot)$ is the PDF of the initial velocity and

$$\mathbf{x}^k(n_j^k) = [x^k(n_j^k), \dot{x}^k(n_j^k), y^k(n_j^k), \dot{y}^k(n_j^k)]^T$$

refers to the state of a target whose label is n_j^k at time k .

The fourth probability in (4.6) is the dynamic model of the targets. A nearly constant velocity model is considered [13]:

$$p(\mathbf{X}^{k+1}(S^k) | \mathbf{X}^k, S^k) = \prod_{j=1}^{t^k-d^k} N(\mathbf{x}^{k+1}(s_j^k); \mathbf{F}\mathbf{x}^k(s_j^k), \mathbf{Q}) \quad (4.10)$$

$$\mathbf{F} = \mathbf{I}_2 \otimes \begin{pmatrix} 1 & \tau \\ 0 & 1 \end{pmatrix} \quad (4.11)$$

$$\mathbf{Q} = \sigma_u^2 \mathbf{I}_2 \otimes \begin{pmatrix} \tau^3/3 & \tau^2/2 \\ \tau^2/2 & \tau \end{pmatrix} \quad (4.12)$$

where it is assumed that the targets move independently, $N(\mathbf{x}; \bar{\mathbf{x}}, \mathbf{Q})$ is the Gaussian PDF evaluated at $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\mathbf{Q}$, $\mathbf{I}_m$ is the $m \times m$ identity matrix, $\otimes$ is the Kronecker product and σ_u^2 is the continuous-time process noise intensity [13].

4.2.2 Sensor models

Two sensor models are considered. They differ in how the signal-to-noise ratio SNR at each sensor is calculated. The first model is the pixelised sensor proposed in [116, 141]. This model assumes that a target produces a given SNR in a sensor only if the target is located in the pixel corresponding to that sensor. All the pixels form a perfect grid that covers the surveillance area. This model is used to compare the algorithm developed in this chapter with the existence grid [61, 141] as to the creation and deletion of tracks. The signal-to noise ratio produced by a target with

coordinates $[x, y]^T$ is:

$$SNR(x, y, x_s, y_s) = \begin{cases} SNR & |x - x_s| \leq l_x/2 \text{ and } |y - y_s| \leq l_y/2 \\ 0 & \text{otherwise} \end{cases} \quad (4.13)$$

where $[x_s, y_s]^T$ are the coordinates of the sensor, $[l_x, l_y]^T$ is the size of the pixels and SNR is assumed to be known.

In the second model, the signal-to-noise ratio produced by a target depends on the distance from the target to the sensor, r , following a piecewise linear function. The function is defined with $N + 1$ intervals whose endpoints and values are $\{r_0 = 0, r_1, \dots, r_N, r_{N+1} = \infty\}$ and $SNR_{r,i}$ $i \in \{1, \dots, N + 1\}$, respectively. Therefore:

$$SNR(r) = \begin{cases} SNR_{r,1} & 0 \leq r < r_1 \\ \dots & \dots \\ SNR_{r,N} & r_{N-1} \leq r < r_N \\ 0 & r \geq r_N \end{cases} \quad (4.14)$$

where $SNR_{r,i}$ and r_i are assumed to be known.

For both models it is assumed that there are M sensors located in the surveillance area with the vector $\mathbf{z}^k = [z_1^k, z_2^k, \dots, z_M^k]^T$ containing the measurements from each sensor, which are Rayleigh-distributed and independent in each sensor. This distribution corresponds to envelope-detected signals under a Gaussian model. This model has been used in radar systems [15]. Then, the PDF of the measurement at sensor j conditioned on the current multitarget state depends on the total signal-to-noise ratio at sensor j , SNR_j , which is the sum of the signal-to-noise ratios produced by each target [16], calculated using (4.13) or (4.14) depending on the sensor model. Therefore:

$$p(\mathbf{z}^k | \mathbf{X}^k, I^k) = \prod_{j=1}^M p(z_j^k | SNR_j) \quad (4.15)$$

where the dependency of SNR_j on $\mathbf{X}^k$ is not given explicitly. The likelihood for sensor j for raw data is:

$$p(z_j^k | SNR_j) = \frac{z_j^k}{1 + SNR_j} \exp\left(-\frac{(z_j^k)^2}{2(1 + SNR_j)}\right) \quad (4.16)$$

Binary detections are produced by thresholding the raw data using the threshold Γ :

$$\Gamma = \sqrt{-2 \ln P_{fa}} \quad (4.17)$$

where P_{fa} is the false alarm probability per sensor. Then, the likelihood for sensor j for binary data is:

$$p(z_j^k | SNR_j) = P_{fa}^{1/(1+SNR_j)} \delta(z_j^k - 1) + (1 - P_{fa}^{1/(1+SNR_j)}) \delta(z_j^k) \quad (4.18)$$

where $\delta(\cdot)$ denotes the Dirac delta. It should be noted that the algorithm proposed in this chapter can be applied to any image-based measurement model. The specific models described here are only used to demonstrate the good performance of the proposed algorithm in the simulations of Section 4.5.

4.3 Particle filter implementation for a fixed and known number of tracks

This section presents a PF to approximately solve (4.1) and (4.2) when the number of tracks is fixed and known. The algorithm is generalised for a variable and unknown number of tracks in Section 4.4. In this case, the number of tracks over time is constant $t^k = t$, $I^k = I = \{1, \dots, t\}$ the set of labels of the surviving tracks coincide with the labels at the previous step $S^k = I$ and no tracks are born at any time $N^k = \emptyset$. Therefore, the JMKPD comes down to estimating $p(\mathbf{X}^{0:k} | \mathbf{z}^{1:k})$ as the labels are implicit in vector $\mathbf{X}^{0:k}$, which denotes the collection of multitarget states from time 0 to k . In addition, as nonlinearity in our tracking model enters in the relationship between the measurements and the positions of the tracks and the relationship between the velocities and the positions is linear and Gaussian, the posterior of the velocity elements can be computed exactly using a Kalman filter conditional on the trajectory of the targets. This technique, known as Rao-Blackwellisation, improves the filter performance for a given sample size [6, 141], see Section 2.5.4 for details. The way it is done here is shown in Appendix A.13. Then, the posterior PDF of the target trajectories is decomposed as:

$$p(\mathbf{X}^{0:k} | \mathbf{z}^{1:k}) = p(\mathbf{P}^{0:k} | \mathbf{z}^{1:k}) \prod_{j=1}^t p(\mathbf{v}_j^{0:k} | \mathbf{p}_j^{0:k}) \quad (4.19)$$

where $\mathbf{P}^{0:k}$ represents the track positions from time 0 to k , $\mathbf{p}_j^{0:k}$ represents the positions of track j from time 0 to k and $\mathbf{v}_j^{0:k}$ represents the velocities of track j from time 0 to k . Due to the nonlinearity between the measurements and the positions of the tracks, the distribution of $\mathbf{P}^{0:k}$ conditional on $\mathbf{z}^{1:k}$ cannot be found exactly and is approximated using a PF.

Therefore, the Rao-Blackwellised PF representation of the posterior PDF from

time 0 to k is:

$$p(\mathbf{X}^{0:k} | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{P}^{0:k} - \mathbf{P}_i^{0:k}) \prod_{j=1}^t p(\mathbf{v}_j^{0:k} | \mathbf{p}_{j,i}^{0:k}) \quad (4.20)$$

where N_{par} is the number of particles, w_i^k is the normalised weight of particle i at time k , $\mathbf{P}_i^{0:k}$ is the position vector of the particle i from time 0 to k and $\mathbf{p}_{j,i}^{0:k}$ is the position vector of track j and particle i .

As the problem of interest is to estimate the multitrack state at time k , one can integrate out the previous positions and velocities from time 0 to $k-1$ in (4.20) to yield:

$$p(\mathbf{X}^k | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{P}^k - \mathbf{P}_i^k) \prod_{j=1}^t p(\mathbf{v}_j^k | \mathbf{p}_{j,i}^{0:k}) \quad (4.21)$$

It should be noted that even though the velocity of a track depends on the whole track trajectory, the particles for the whole trajectory do not have to be stored as the velocity is obtained recursively via a Kalman filter. Thus, equation (4.21) can be written as:

$$p(\mathbf{X}^k | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{P}^k - \mathbf{P}_i^k) \prod_{j=1}^t N(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \boldsymbol{\Sigma}_{j,i}^k) \quad (4.22)$$

where $\bar{\mathbf{v}}_{j,i}^k$ and $\boldsymbol{\Sigma}_{j,i}^k$ are the posterior mean and covariance matrix of the velocity elements conditional on the trajectory of positions for the i th particle and j th track. Then, what needs to be stored at time k is the set of Rao-Blackwellised particles $\mathbf{X}_i^k = (\mathbf{P}_i^k, \bar{\mathbf{v}}_{1,i}^k, \boldsymbol{\Sigma}_{1,i}^k, \dots, \bar{\mathbf{v}}_{t,i}^k, \boldsymbol{\Sigma}_{t,i}^k)$ for $i = 1, \dots, N_{par}$ with their weights.

The PF approximation of the posterior density of the positions at time $k+1$ using (4.22), (4.10), the Chapman-Kolmogorov equation and Bayes' rule is:

$$p(\mathbf{P}^{k+1} | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{P}^{k+1}) \sum_{i=1}^{N_{par}} w_i^k \prod_{j=1}^t p(\mathbf{p}_j^{k+1} | \mathbf{X}_i^k) \quad (4.23)$$

where $p(\mathbf{p}_j^{k+1} | \mathbf{X}_i^k)$ is Gaussian with mean $\mathbf{p}_{j,i}^k + \tau \bar{\mathbf{v}}_{j,i}^k$ and covariance matrix $\tau^2 \boldsymbol{\Sigma}_{j,i}^k + \mathbf{Q}_p$ where $\mathbf{Q}_p$ is the process noise covariance matrix for the position elements according to (4.12). It should be noted that in (4.23) there are PDFs conditioned on Rao-Blackwellised particles. This notation was explained in Section 2.5.4.

Then, the PF approximation of the posterior needs to obtain samples that are distributed according to (4.23). There are a lot of ways to approximate (4.23) using importance sampling, each one defining a PF. As pointed out in [141], the optimal importance density (OID) proposes particles jointly for tracks that are in close

proximity making use of the previous multitrack state and the current measurement. Otherwise, there could be particles which are proposed in undesirable parts of the multitrack state space and the performance of the PF is severely degraded. However, sampling from the OID is usually intractable and can be expensive even when it is tractable. Therefore, a suboptimal importance density called parallel partition (PP) method is proposed. The importance density developed here is a convenient, fast way to propagate particles if calculating or approximating the OID is not feasible or very demanding in terms of computation.

4.3.1 Parallel partition method

Here, the PP method as a modification of the well-known IP method is proposed. A detailed mathematical explanation on how both methods work via auxiliary sampling is also provided. IP and PP methods make the following assumption:

Assumption 1: Tracks are a priori independent with evenly distributed weights, $w_i^k = 1/N_{par}$, so that

$$p(\mathbf{P}^{k+1} | \mathbf{z}^{1:k}) \propto \prod_{j=1}^t \sum_{i=1}^{N_{par}} p(\mathbf{p}_j^{k+1} | \mathbf{x}_{j,i}^k) \quad (4.24)$$

Under Assumption 1, the posterior PDF (4.23) becomes:

$$p(\mathbf{P}^{k+1} | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{P}^{k+1}) \prod_{j=1}^t \sum_{i=1}^{N_{par}} p(\mathbf{p}_j^{k+1} | \mathbf{x}_{j,i}^k) \quad (4.25)$$

where $\mathbf{x}_{j,i}^k$ is the Rao-Blackwellised i th particle for track j at time k . It should be noted that getting weights evenly distributed at time k can be done by performing a resampling step of the particles at time k [9].

Then, what is proposed is to sample an auxiliary variable for each track such that the posterior is:

$$p(\mathbf{P}^{k+1}, \mathbf{a} | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{P}^{k+1}) \prod_{j=1}^t p(\mathbf{p}_j^{k+1} | \mathbf{x}_{j,a_j}^k) \quad (4.26)$$

where $\mathbf{a} = [a_1, \dots, a_t]^T \in \{1, \dots, N_{par}\}^t$ and a_j indicates that the component a_j for track j is chosen on the mixture given by the posterior, equation (4.25). The idea of sampling in a higher dimension has traditionally been used, for instance, in the auxiliary PF [9, 56, 165, 169]. If the auxiliary variables in (4.26) are integrated out, the PDF given by (4.25) is obtained. Consequently, this definition of auxiliary

variables is sound. If one draws a sample from the joint density (4.26) and then discards the part of the sample that corresponds with $\mathbf{a}$, then a sample from (4.25) is being produced as required. Note that this bears some resemblance to the auxiliary particle filter explained in Section 2.5.3.

If $\mathbf{P}_i^{k+1}$, $\mathbf{a}_i = [a_1^i, \dots, a_t^i]^T$ are drawn from an importance density $q_1(\cdot)$ for $i = 1, \dots, N_{par}$, the updated weights are calculated as:

$$w_i^{k+1} \propto \frac{p(\mathbf{z}^{k+1} | \mathbf{P}_i^{k+1}) \prod_{j=1}^t p(\mathbf{p}_{j,i}^{k+1} | \mathbf{x}_{j,a_j^i}^k)}{q_1(\mathbf{P}_i^{k+1}, \mathbf{a}_i | \mathbf{z}^{1:k+1})} \quad (4.27)$$

The importance density in the PP filter is taken to be:

$$q_1(\mathbf{P}^{k+1}, \mathbf{a} | \mathbf{z}^{1:k+1}) = \prod_{j=1}^t b_j(\mathbf{p}_j^{k+1}) p(\mathbf{p}_j^{k+1} | \mathbf{x}_{j,a_j}^k) \quad (4.28)$$

where

$$b_j(\mathbf{p}_j^{k+1}) \propto p(\mathbf{z}^{k+1} | \hat{\mathbf{p}}_1^{k+1|k}, \dots, \hat{\mathbf{p}}_{j-1}^{k+1|k}, \mathbf{p}_j^{k+1}, \hat{\mathbf{p}}_{j+1}^{k+1|k}, \dots, \hat{\mathbf{p}}_t^{k+1|k}) \quad (4.29)$$

and $\hat{\mathbf{p}}_j^{k+1|k}$ is the predicted position of track j averaged over all the particles:

$$\hat{\mathbf{p}}_j^{k+1|k} = \frac{1}{N_{par}} \sum_{i=1}^{N_{par}} (\mathbf{p}_{j,i}^k + \tau \bar{\mathbf{v}}_{j,i}^k) \quad (4.30)$$

In contrast, it should be noted that the IP method uses $b_j(\mathbf{p}_j^{k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{p}_j)$ rather than (4.29) but the rest of the steps are similar. Substituting (4.28) and (4.29) into (4.27), the unnormalised particle weights, second-stage weights, are:

$$w_i^{k+1} \propto \frac{p(\mathbf{z}^{k+1} | \mathbf{P}_i^{k+1})}{\prod_{j=1}^t b_j(\mathbf{p}_{j,i}^{k+1})} \quad (4.31)$$

The computational expense of evaluating (4.29) and then (4.31), can be lowered if we separate the tracks into $s \leq t$ clusters $C_1^{k+1}, \dots, C_s^{k+1}$ such that $\bigcup_{l=1}^s C_l^{k+1} = I$ and $\forall l \in \{1, \dots, s\}, \forall i \in C_l^{k+1}$

$$\|\hat{\mathbf{p}}_i^{k+1|k} - \hat{\mathbf{p}}_j^{k+1|k}\| \leq \Gamma_c \rightarrow j \in C_l^{k+1} \quad (4.32)$$

where Γ_c is a threshold. Equation (4.32) is a mathematical way to indicate that if the distance between the predicted positions of two tracks is below Γ_c , then, these

tracks belong to the same cluster⁴. Using the same reasoning as in [141], for a sufficiently large value of Γ_c , (4.29) can be written as:

$$b_j(\mathbf{p}_j^{k+1}) \propto p(\mathbf{z}^{k+1} | \hat{\mathbf{C}}_{l-\{j\}}^{k+1|k}, \mathbf{p}_j^{k+1}) \quad (4.33)$$

$\hat{\mathbf{C}}_{l-\{j\}}^{k+1|k}$ are the predicted positions of the tracks that belong to cluster l , in which track j is also included, except the position of track j averaged over all the particles. That is, the simplified equation (4.33) only considers the tracks that are close to track j in the general equation (4.29). In the rest of the section, the procedure for drawing samples from (4.28) is explained.

4.3.1.1 Sampling from the importance density

Here, it is explained how IP and PP draw samples from (4.28) recalling that the only difference between both methods is the definition of $b_j(\mathbf{p}_j^{k+1})$. One can write

$$q_1(\mathbf{P}^{k+1}, \mathbf{a} | \mathbf{z}^{1:k+1}) = \prod_{j=1}^t q_{1,j}(\mathbf{p}_j^{k+1}, a_j | \mathbf{z}^{1:k+1}) = \prod_{j=1}^t b_j(\mathbf{p}_j^{k+1}) p(\mathbf{p}_j^{k+1} | \mathbf{x}_{j,a_j}^k) \quad (4.34)$$

where $q_{1,j}(\cdot)$ is the importance density to draw positions and indices for track j . It should be noted that it is generally not possible to sample directly from $q_{1,j}(\cdot)$. Instead, the approximate method sampling/importance resampling in [170] is used to get samples from $q_{1,j}(\cdot)$.

General algorithm According to [170], one can define a “first-pass approximation” of $q_{1,j}(\mathbf{p}_j^{k+1}, a_j | \mathbf{z}^{1:k+1})$ called $h(\mathbf{p}_j^{k+1}, a_j | \mathbf{z}^{1:k+1})$. Then, M_{par} samples are drawn from $h(\mathbf{p}_j^{k+1}, a_j | \mathbf{z}^{1:k+1})$, which are denoted by $\mathbf{p}_{j,p}^*, a_{j,p}^*$ for $p = 1, \dots, M_{par}$ and the following quotient is calculated

$$r(\mathbf{p}_{j,p}^*, a_{j,p}^*) \propto \frac{q_{1,j}(\mathbf{p}_{j,p}^*, a_{j,p}^* | \mathbf{z}^{1:k+1})}{h(\mathbf{p}_{j,p}^*, a_{j,p}^* | \mathbf{z}^{1:k+1})} \quad (4.35)$$

Finally, N_{par} values of $(\mathbf{p}_j^{k+1}, a_j)$ are drawn from the distribution defined by $(\mathbf{p}_{j,p}^*, a_{j,p}^*)$ for $p = 1, \dots, M_{par}$ with probability proportional to $r(\mathbf{p}_{j,p}^*, a_{j,p}^*)$. The resulting particles are samples from $q_{1,j}(\cdot)$ [170].

⁴A track can belong to only one cluster and the computational burden of clustering is almost negligible in the algorithm.

Application to IP and PP In both IP and PP methods, $h(\cdot)$ is

$$h(\mathbf{p}_j^{k+1}, a_j | \mathbf{z}^{1:k+1}) \propto p(\mathbf{p}_j^{k+1} | \mathbf{x}_{j,a_j}^k) \quad (4.36)$$

and IP and PP sample from $h(\cdot)$ via stratified sampling [171]. The sample space of $(\mathbf{p}_j^{k+1}, a_j)$ is $\Omega = \mathbb{R}^2 \times \{1, \dots, N_{par}\}$. Then, one can define N_{par} disjoint subregions $\{D_1, \dots, D_{N_{par}}\}$ such that $D_i = \mathbb{R}^2 \times \{i\}$ and $\Omega = \bigcup_{i=1}^{N_{par}} D_i$. From (4.36), the probability that a sample belongs to region D_i is $1/N_{par}$, i.e., the regions are equally probable. Then, assuming M_{par} samples from $h(\cdot)$ are required, there are M_i samples in region D_i such that $\sum_{i=1}^{N_{par}} M_i = M_{par}$. Since the variance in these regions is the same, the optimal way to assign M_i is $M_i = M_{par}/N_{par}$ [171]. In the IP and PP method $M_{par} = N_{par}$, so $M_i = N_{par}/N_{par} = 1$ although it can be easily generalised for $M_{par} > N_{par}$. This implies that sampling from $h(\mathbf{p}_j^{k+1}, a_j | \mathbf{z}^{1:k+1})$ can be done going through all the regions (i.e. indices of the particles), drawing one position for each region (particle), noting that in each region the index is deterministic.

Once M_{par} samples have been drawn from (4.36), N_{par} values of $(\mathbf{p}_j^{k+1}, a_j)$ are drawn from the distribution defined by $(\mathbf{p}_{j,p}^*, a_{j,p}^*)$ for $p = 1, \dots, N_{par}$ with probability proportional to $r(\mathbf{p}_{j,p}^*, a_{j,p}^*) = b_j(\mathbf{p}_{j,p}^*)$ getting samples from $q_{1,j}(\cdot)$. The values $b_j(\mathbf{p}_{j,p}^*)$ for $p = 1, \dots, N_{par}$ are called the first-stage weights of the j th track.

These are the steps shown in Table 4.1. Note that in [161] and [116] they also sample following the steps in Table 4.1 but they do not justify why.

4.3.1.2 Discussion about the PP method

The important features of the PP method are subparticle crossover and the way the first-stage weights are chosen. Subparticle refers to the part of a particle that represents a single track. Therefore, a particle is made up of the subparticles of all the tracks. Then, by sampling an auxiliary variable for each track, $\mathbf{a}$ in (4.28), a particle at time step k is allowed to be formed by the propagation of subparticles that belonged to different particles at time $k-1$, see Figure 4.1. For example, a certain $\mathbf{a}_i = [a_1^i, \dots, a_t^i]^T$ indicates that to form particle i one is using subparticle a_1^i for target one, a_2^i for target two and so on, therefore, mixing them and doing subparticle crossover.

Subparticle crossover is also included in the IP method [116–118]. In fact, as indicated before, the IP method is equivalent to the PP method if $b_j(\mathbf{p}_j) \propto p(\mathbf{z}^{k+1} | \mathbf{p}_j)$ rather than (4.33). However, IP method performance plummets when there are several targets in close proximity. This loss of performance is due to the way the

Table 4.1: Independent and parallel partition subroutines

IP and PP subroutines for track or pre-track j with Rao-Blackwellisation

- For each particle $i = 1, \dots, N_{par}$ (Region D_i):
 - Get a sample for the position $\mathbf{p}_{j,i}^* \sim p(\mathbf{p}_j^{k+1} | \mathbf{x}_{j,i}^k)$ using (4.10).
 - Calculate the mean speed and covariance matrix using a Kalman filter $\bar{\mathbf{v}}_{j,i}^* \propto \Sigma_{j,i}^*$.
 - IF the IP method is used
 - * Calculate $b_j(\mathbf{p}_{j,i}^*) = p(\mathbf{z}^{k+1} | \mathbf{p}_{j,i}^*)$ using (4.15).
 - ELSE (the PP method is used)
 - * Calculate $b_j(\mathbf{p}_{j,i}^*) = p(\mathbf{z}^{k+1} | \mathbf{p}_{j,i}^*, \hat{\mathbf{C}}_{l-\{j\}}^{k+1|k})$ using (4.15) where l is the cluster of track j .
- Normalise $b_j(\mathbf{p}_{j,i}^*)$ to sum to one over $i = 1, \dots, N_{par}$.
- Resampling step for each track. For each particle $i = 1, \dots, N_{par}$:
 - Sample an index p from the distribution defined by $b_j(\mathbf{p}_{j,l}^*)$ over $l = 1, \dots, N_{par}$.
 - Set $\mathbf{x}_{j,i}^{k+1} = (\mathbf{p}_{j,p}^*, \bar{\mathbf{v}}_{j,p}^*, \Sigma_{j,p}^*)$ and the parent's index $a_j^i = p$.

Figure 4.1: Subparticle crossover for three tracks: A particle at time k can be made up of the propagation of the tracks that belonged to different particles at time $k-1$. The circles inside each subparticle stand for the first-stage weight of the subparticle, see (4.33). At time k , it is desirable to form particles with subparticles that have a high first-stage weight. In this case, $\mathbf{a}_i = [i, i, j]^T$ and $\mathbf{a}_j = [i, j, j]^T$.

first-stage weights are chosen as only one target is considered. This leads to the selection of subparticles that are appropriate for representing one target because the likelihood for that subparticle considering one target is high. Nonetheless, this does not imply that a particle formed by stacking good subparticles for representing one target will have a high likelihood when considering all the targets together. On the contrary, the PP method, selects the first-stage weights considering all the targets in the surroundings, then, the resulting particle lies in an area of high likelihood when there are several targets in close proximity.

It should be noted that Assumption 1, which is required for subparticle crossover, does not necessarily hold for a group of closely spaced tracks as their positions conditioned on the measurements could be dependent. However, assumptions of this kind are quite common in multitarget tracking algorithms. For instance the well-known joint probabilistic data association filter (JPDAF) [63] assumes a priori independence of the target states as does the Monte Carlo approximation to the JPDAF proposed in [197]. These methods have been shown to work very well despite the occasional invalidity of this assumption. In the context of dynamic Bayesian networks, assuming that the state is partitioned into independent subspaces improves estimation performance when the state space is of high dimension [27, 152]. Using Assumption 1 when it is not satisfied can be interpreted as a way of trading-off bias and variance. In particular, it is argued that the bias incurred in assuming a priori independence when it does not actually exist is less than the variance which results by the joint sampling of several track states, particularly when tracks states are sampled from the prior and the number of particles is relatively small. This argument is supported by simulation results in Section 4.5.

4.4 Particle filter implementation for a variable and unknown number of tracks

In the previous section the PP method was presented as a way to propagate particles in a multitarget tracking set-up when the number of targets is fixed and known. In this section, a two-layer PF is proposed to deal with the problem of tracking a variable and unknown number of targets. One layer deals with the termination of the tracks and the tracking itself while the other deals with track initiation. Both layers are PFs on their own, therefore, they are called track PF and pre-track PF, respectively. This idea is illustrated in Figure 4.2.

Firstly, the importance density for the track PF is described. The importance

Figure 4.2: Two-layer particle filter diagram. The measurements update the two layers simultaneously. When the pre-tracks are considered tracks, they pass to the upper layer. In the area where there is a track, no pre-tracks are initiated.

density is chosen so that it factorises as:

$$q\left(\mathbf{P}^{k+1}, I^{k+1}, \mathbf{a}, \mathbf{c} \mid \mathbf{z}^{1:k+1}\right) = q_p\left(\mathbf{P}^{k+1}\left(I^k\right), S^k, \mathbf{a} \mid \mathbf{z}^{1:k+1}\right) \cdot q_3\left(\mathbf{P}^{k+1}\left(N^{k+1}\right), N^{k+1}, \mathbf{c} \mid \mathbf{z}^{1:k+1}\right) \quad (4.37)$$

where $\mathbf{a} = [a_1, \dots, a_{t^k}]^T$ is the auxiliary vector of the PP method and, $\mathbf{c} = [c_1, \dots, c_{b^{k+1}}]^T \in \{0, \dots, N_{par}\}^{b^{k+1}}$ is an auxiliary vector used to create new tracks. The use of the variable $\mathbf{c}$ is explained in Section 4.4.2. The importance density $q_p(\cdot)$ propagates the labels of the surviving targets and their states. The importance density $q_3(\cdot)$ proposes the positions of the tracks that are born at time $k+1$ and assigns their labels.

The importance density $q_p(\cdot)$ is taken to be:

$$q_p\left(\mathbf{P}^{k+1}\left(I^k\right), S^k, \mathbf{a} \mid \mathbf{z}^{1:k+1}\right) = q_1\left(\mathbf{P}^{k+1}\left(I^k\right), \mathbf{a} \mid \mathbf{z}^{1:k+1}\right) q_2\left(S^k \mid \mathbf{P}^{k+1}, \mathbf{a}, \mathbf{z}^{1:k+1}\right) \quad (4.38)$$

where $q_1(\cdot)$ is the importance density of the PP method for known number of targets, equation (4.28), that only depends on the Rao-Blackwellised particles at time k , and $q_2(\cdot)$ determines the survival targets at time k . Note that $q_p(\cdot)$ samples the states of all the targets that were alive at the previous time (whose labels are given by I^k) and based on this, it samples the existence of the targets. A thorough explanation of the importance density (4.38) is given in Appendix A.14.

The importance density $q_2(\cdot)$ is defined as:

$$q_2\left(S^k \mid \mathbf{P}^{k+1}, \mathbf{a}, \mathbf{z}^{1:k+1}\right) =$$

$$\prod_{j=1}^{t^k} \left[q_{2,j} \left(i_{j,a_j}^k \in S^k \mid \mathbf{p}_{1,a_1}^{k-N_d+2:k+1}, \dots, \mathbf{p}_{t^k,a_{t^k}}^{k-N_d+2:k+1}, \mathbf{z}^{k-N_d+2:k+1} \right) \chi_{S^k} \left(i_{j,a_j}^k \right) + \right. \\ \left. \left(1 - q_{2,j} \left(i_{j,a_j}^k \in S^k \mid \mathbf{p}_{1,a_1}^{k-N_d+2:k+1}, \dots, \mathbf{p}_{t^k,a_{t^k}}^{k-N_d+2:k+1}, \mathbf{z}^{k-N_d+2:k+1} \right) \right) \left(1 - \chi_{S^k} \left(i_{j,a_j}^k \right) \right) \right] \quad (4.39)$$

where $q_{2,j} \left(i_{j,a_j}^k \in S^k \right)$ represents the probability that track i_{j,a_j}^k survives, $\mathbf{p}_{j,a_j}^{k-N_d+2:k+1}$ represents the trajectory of particle a_j of target j from time $k - N_d + 2$ to $k + 1$. Thus, N_d is the number of past time steps that are used to decide if a track survives. The importance density $q_2(\cdot)$ depends on several time instants so that the target's dynamics are taken into account to remove targets. In Section 4.4.1, this importance density is explained thoroughly.

Finally, the importance density for proposing new tracks $q_3 \left(\mathbf{p}^{k+1} \left(N^{k+1} \right), N^{k+1}, \mathbf{c} \mid \mathbf{z}^{1:k+1} \right)$ does not depend on previous target positions and another PF is used to draw samples from it, i.e., it is the second layer of the overall PF. In Section 4.4.2, this importance density is addressed.

4.4.1 Importance density for removing tracks

The removal of a track in a particle is performed via a likelihood ratio test (LRT) [106,192] based on N_d time instants as indicated by (4.39). The decision to remove a track considers several time instants so that target dynamics are taken into account. The two hypotheses for the test are H_1 : the track i_j^k is alive and H_0 : the track i_j^k is dead. Due to the fact that the importance density has to be non-zero wherever the prior is, a probability of acceptance of the result of the test has to be used:

$$q_{2,j} \left(i_j^k \in S^k \mid \mathbf{p}^{k-N_d+2:k+1}, \mathbf{z}^{k-N_d+2:k+1} \right) = \begin{cases} P_a & \Lambda_d = H_1 \\ 1 - P_a & \Lambda_d = H_0 \end{cases} \quad (4.40)$$

where P_a is the probability of accepting the hypothesis given by the test⁵ and Λ_d is the LRT [106]:

$$\frac{\prod_{l=-1}^{N_d-2} p \left(\mathbf{z}^{k-l} \mid H_1, \mathbf{p}^{k-l} \right)}{\prod_{l=-1}^{N_d-2} p \left(\mathbf{z}^{k-l} \mid H_0, \mathbf{p}^{k-l} \right)} \underset{\Lambda_d = H_0}{\overset{\Lambda_d = H_1}{\geq}} \eta_d \quad (4.41)$$

⁵ P_a is mainly used for theoretical reasons but in the simulations the output of the LRT is actually accepted. This could be interpreted as $P_a \approx 1$.

where η_d is a threshold. If track i_j^k belongs to cluster l^k at time k . Equation (4.41) can be simplified as:

$$\frac{\prod_{l=-1}^{N_d-2} p(\mathbf{z}^{k-l} | \mathbf{C}_{l^{k-l}}^{k-l})}{\prod_{l=-1}^{N_d-2} p(\mathbf{z}^{k-l} | \mathbf{C}_{l^{k-l}-\{i_j^k\}}^{k-l})} \begin{matrix} \Lambda_d = H_1 \\ \geq \\ \Lambda_d = H_0 \end{matrix} \geq \eta_d \quad (4.42)$$

where $\mathbf{C}_{l^k}^k$ represents the positions of the tracks in cluster l^k at time k and $\mathbf{C}_{l^k-\{i_j^k\}}^k$ represents the positions of the tracks in cluster l^k at time k except the position of track i_j^k .

4.4.2 Importance density for proposing new tracks. Pre-track particle filter

As stated previously, the importance density for proposing new tracks $q_3(\mathbf{P}^{k+1}(N^{k+1}), N^{k+1}, \mathbf{c} | \mathbf{z}^{1:k+1})$ is calculated by means of a PF. This filter is referred to as pre-track PF because it deals with the initialisation of tracks. As it is another PF, one must specify the a priori model, the measurement model and the importance density. That is, the posterior density of this layer is used to determine the importance density of the main PF to propose new targets.

In the proposed algorithm, the initiation of a pre-track in a sensor must meet several conditions. Firstly, that sensor and its neighbours must not have tracks or pre-tracks in their area of influence (area in which a target produces an SNR above zero). Secondly, if binary data sensors are being used, there has to be a detection in that sensor or, if raw data sensors are being used, the measurement in that sensor has to be higher than the threshold in (4.17) given P_{fa} . In this section, these two events are named false alarms for the sake of clarity. Hence, for raw and binary data, P_{fa} is the probability of initiating a pre-track in a sensor which does not have any targets in its surroundings.

A pre-track can be either a real track or a false track so the expected number of targets in this filter is higher than in the track PF. Besides, in the proposed algorithm, the pre-tracks have a deterministic life equal to N_b instants of time. After this time, a pre-track is considered either a real track, and then its subparticles are transferred to the track PF, or a false track and its subparticles are discarded. Therefore, the measurement model and the state evolution model are the same as in Section 4.2 except for the births and deaths of pre-tracks, equations (4.3) and (4.8). Then, the aim of the pre-track PF is to estimate $p(\tilde{\mathbf{X}}^k, \tilde{I}^k | \mathbf{z}^{1:k})$, where $\tilde{\mathbf{X}}^k$ and $\tilde{I}^k$ stand for the states and labels of the pre-tracks, and decide which of these pre-tracks are real and, consequently, are moved to the track PF.

A pre-track can be initiated because of two independent events: the arrival of a real target, modelled by (4.3) and the occurrence of a false alarm. The probability mass function of the number of false alarms $n_{P_{fa}}$ in each sampling period is a binomial distribution:

$$p(n_{P_{fa}}) = \binom{M}{n_{P_{fa}}} (P_{fa})^{n_{P_{fa}}} \cdot (1 - P_{fa})^{M - n_{P_{fa}}} \quad (4.43)$$

Under the assumption that $P_{fa} \ll 1$ and $M \gg 1$, the probability mass function in (4.43) can be approximated by a Poisson random variable [164]:

$$p(n_{P_{fa}}) \approx \frac{e^{-M \cdot P_{fa}} (M \cdot P_{fa})^{n_{P_{fa}}}}{n_{P_{fa}}!} \quad (4.44)$$

Then, the number of arrivals of pre-tracks in a time interval τ is the sum of two independent Poisson random variables, given by (4.3) and (4.44). The result is another Poisson random variable whose parameter is the sum of the parameters [111]. Therefore, the probability that there are $\tilde{b}^k$ births of pre-tracks is

$$p(\tilde{b}^k) \approx \frac{e^{-\tilde{\lambda}} (\tilde{\lambda})^{\tilde{b}^k}}{\tilde{b}^k!} \quad (4.45)$$

where $\tilde{\lambda} = \lambda \cdot \tau + M \cdot P_{fa}$. Thus, it can be noted that assuming that a pre-track is not missed due to a lack of memory, the number of pre-tracks in this filter is a queueing system $M/D/\infty$ because the arrivals of the pre-tracks follows a Poisson process and the lives of the pre-tracks are deterministic [111].

The importance density of the pre-track filter is split into three independent parts: evolution of the model, births and deaths of pre-tracks. The importance density for propagating the targets states $\tilde{q}_1(\cdot)$ uses the IP method because new targets are expected to appear isolated. In view of the deterministic life of a pre-track, there is no need to define an importance density $\tilde{q}_2(\cdot)$ because the pre-tracks are simply removed from the pre-track filter when they have lived for N_b time instants. The importance density $\tilde{q}_3(\cdot)$ used for the births of new pre-tracks works as follows: every time there is a detection in a cell and this cell and its neighbouring cells are not occupied by a track or a pre-track, a new pre-track is initiated with a position drawn from a uniform PDF in the area of the cell and the target velocity is initiated in a similar way as in Section V.C of [141]: When a pre-track is added to a particle of the pre-track layer, two integers η_1, η_2 are drawn such that $P(\eta_i = j) = 1/2$ for $j \in \{-1, 1\}$ $i \in \{1, 2\}$ and then select the initial distribution of the velocity to be Gaussian with mean $[\eta_1, \eta_2]^T$ and covariance matrix $9\mathbf{I}_2$. It should be noted that

the initial pre-track velocities have been implemented in this way in the simulations of Section 4.5 but any other initialisation is also possible.

Finally, an LRT based on N_b time instants is carried out to decide whether a pre-track in each particle is a real track or not. The two hypotheses are H_1 : the pre-track is a real track and H_0 : the pre-track is a false track. As the targets are expected to appear isolated, this test only takes the pre-track under consideration into account and the test becomes:

$$\frac{\prod_{l=-1}^{N_b-2} p(\mathbf{z}^{k-l} | H_1, \tilde{\mathbf{x}}^{k-l}(\tilde{i}_j^k))}{\prod_{l=-1}^{N_b-2} p(\mathbf{z}^{k-l} | H_0)} \underset{\Lambda_b = H_0}{\overset{\Lambda_b = H_1}{\geq}} \eta_b \quad (4.46)$$

where η_b is the threshold and $\tilde{\mathbf{x}}^{k-l}(\tilde{i}_j^k)$ denotes the state of the pre-track of label $\tilde{i}_j^k$ at time $k-l$.

Recall from (4.37) that drawing particles for newborn tracks requires an auxiliary variable $\mathbf{c} = [c_1, \dots, c_{b^{k+1}}]^T \in \{0, \dots, N_{par}\}^{b^{k+1}}$ where b^{k+1} is the number of pre-tracks in which at least one particle has passed the test (4.46) at time $k+1$. The variable $c_j = 0$ if the j th pre-track is discarded and $c_j \neq 0$ indicates which particle from the pre-track PF is used to draw a new particle in the track PF.

Using the fact that the weights of the pre-track PF are evenly distributed, the importance density is taken to be:

$$q_3(\mathbf{P}^{k+1}(N^{k+1}), N^{k+1}, \mathbf{c} | \mathbf{z}^{1:k+1}) = \prod_{j=1}^{b^{k+1}} q_{3,1}(c_j | \mathbf{z}^{1:k+1}) q_{3,2}(\mathbf{p}_j^{k+1} | \mathbf{z}^{1:k+1}, c_j) \cdot q_{3,3}(n_j^{k+1} | \mathbf{p}_j^{k+1}) \quad (4.47)$$

where

$$q_{3,1}(c_j = i | \mathbf{z}^{1:k+1}) = \begin{cases} d_{j,i}/N_{par} & i \neq 0 \\ 1 - \sum_{i=1}^{N_{par}} d_{j,i}/N_{par} & i = 0 \end{cases} \quad (4.48)$$

$$q_{3,2}(\mathbf{p}_j^{k+1} | \mathbf{z}^{1:k+1}, c_j) \propto p(\mathbf{z}^{k+1} | \mathbf{p}_j^{k+1}) p(\mathbf{p}_j^{k+1} | \tilde{\mathbf{x}}_{j,c_j}^k) \quad c_j > 0 \quad (4.49)$$

where $\tilde{\mathbf{x}}_{j,i}^k$ is the Rao-Blackwellised particle i for pre-track j and $d_{j,i}$ is the output of LRT (4.46) for pre-track j and particle i , i.e., $d_{j,i} = 1$ if $\Lambda_b = H_1$ or $d_{j,i} = 0$ otherwise. When $c_j = 0$, it means that no target is born and, thus, $q_{3,2}(\cdot)$ assigns $\mathbf{p}_j^{k+1} = \theta$, where θ is the symbol for no target [146]. Once c_j has been sampled for all the particles, the position of the newborn target is sampled from $q_{3,2}(\cdot)$ using the approximation in [170], i.e., via the IP method in Table 4.1. In addition, $q_{3,3}(\cdot)$ assigns the labels of these newborn tracks. It should be noted that according to the prior, see (4.7), the label of a target is deterministic and depends on the position

of the newborn target and the time index. Therefore, $q_{3,3}(\cdot)$ corresponds with the prior assuming that the target positions in all the particles lie in the same cell so that the target has the same label in all the particles. In addition, the mean and covariance matrix of the initial velocity of a track are given by a Kalman filter that uses the past positions of the corresponding pre-track.

One should note that the probability of transferring pre-track j to a particle in the track PF is $q_{3,1}(c_j > 0 | \mathbf{z}^{1:k+1}) = \sum_{i=1}^{N_{par}} d_{j,i}/N_{par} = n_{pass}/N_{par}$ where n_{pass} is the number of particles that have passed the test (4.46). Therefore, the more particles there are that pass the test, the more likely it is to initiate a track.

Then, using the prior model of evolution of targets and the measurement model in Section 4.2 and employing (4.37), (4.38), (4.40) and (4.47), the updated particle weights of the track PF are:

$$w_i^{k+1} \propto \frac{p(\mathbf{z}^{k+1} | \mathbf{P}_i^{k+1}) p(S_i^k | I_i^k) p(N_i^{k+1}) p(\mathbf{P}_i^{k+1} | N_i^{k+1})}{q_2(S_i^k) q_3(\mathbf{P}_i^{k+1} | N_i^{k+1}, N_i^{k+1}, \mathbf{c}_i) \prod_{j=1}^{t_i^k} b_j(\mathbf{p}_{j,i}^{k+1})} \quad (4.50)$$

where the conditioning of the importance densities $q_2(\cdot)$ and $q_3(\cdot)$ has been removed to simplify the notation and S_i^k and N_i^{k+1} are the sets that contain the labels of the surviving targets and the newborn targets of particle i , respectively. One should note that the prior PDF of the initial target positions, given by (4.9), is inversely proportional to $A_S^{b^{k+1}}$. On the other hand, $p(N^{k+1}) \propto A_S^{b^{k+1}}$, recalling from Section 4.2 that $\lambda = \lambda_A A_S$. Then, both factors cancel each other out and the update of the weights is independent of the area of the surveillance region. Finally, a summary of the integrated detection and tracking algorithm is given in Table 4.2.

4.5 Simulation results

In this section, the proposed algorithm is compared with other methods available in the literature. Firstly, the PP method is compared to other importance densities in the tracking part of the filter. Secondly, the two-layer PF is compared to other methods as for the estimation of the number of targets.

4.5.1 Tracking performance

In this section, the PP method is compared with other methods available in the literature: the above-mentioned IP, the adaptive proposal method that uses IP when the targets are far from each other and the coupled partition method (CP)

Table 4.2: Proposed algorithm for detection and tracking of multiple targets

- Separate the tracks into clusters using (4.32).
- For each track, propose a new state using PP subroutine, see Table 4.1.
- Calculate the quotient of the LRT for the tracks using (4.42).
- For each pre-track, propose a new state using IP subroutine, see Table 4.1.
- Calculate the quotient of the LRT for the pre-tracks using (4.46).
- For all the pre-tracks that have lived N_b time instants, evaluate the test (4.46).
 - If there are pre-tracks that pass the test, sample the subparticles of the newborn target using (4.47).
- For each particle in the track PF:
 - Remove the tracks by evaluating the importance density (4.40).
 - Update the weights of the particles using (4.50).
- Add pre-tracks to the pre-track PF according to $\tilde{q}_3(\cdot)$, see Section 4.4.2.
- Update the weights of the particles in the pre-track PF, see Section 4.4.2.
- Perform resampling to obtain an evenly weighted particle set for both layers.

Table 4.3: Properties of the importance densities

	Subparticle crossover (far)	Subparticle crossover (near)	Considers nearby targets
IP	✓	✓	×
CP	✓	×	×
PP	✓	✓	✓
AA	✓	×	✓
JA	×	×	✓

when they are in close proximity⁶ [116, 117], a jointly auxiliary PF (JA) [139, 165], an adaptive auxiliary PF (AA) and the UKF [100, 139]. JA refers to the traditional auxiliary PF, i.e., at each time step the multitarget state at the following time step is predicted, the first-stage weights are computed, then, one draws samples from the auxiliary variable and propagates the particles, see [139, 165] for details. Therefore, this PF never performs subparticle crossover. AA consists of an auxiliary PF that works by clustering as in PP. Then, for each cluster particles are drawn using an auxiliary PF. Therefore, this PF performs subparticle crossover for targets that are far, i.e., they do not belong to same cluster, and does not for targets that are in the same cluster. The properties of the importance densities of these PFs can be found in Table 4.3.

It should be pointed out that the structure of the PF shown in Table 4.2 is used for all algorithms including the creation and deletion of tracks. The only part that changes is the sampling of the surviving target positions. The implementation of the UKF also makes use of the clusters. When the targets are isolated, their states and covariance matrices are updated independently and when they belong to the same cluster they are updated jointly. Thus, the number of sigma points that are needed to match the first two moments [100] is $8N_c + 1$ where N_c is the number of targets in the cluster⁷. The implementation of CP in the simulations uses $R = 10$ possible realisations of the future state as in [116]. In addition, the proposed method is compared with tracking using random finite sets (RFS). Particularly, the posterior is approximated using multi-Bernoulli random finite sets (MBRFS) for raw data as explained in [201]. The implementation of MBRFS in the simulations removes targets with existence probabilities below 0.001 and the initial existence probability for a target is 0.005.

The considered scenario used to evaluate the performance of the algorithms consists of four targets whose trajectories cross at the same time, see Figure 4.3.

⁶In this thesis, CP refers to the adaptive proposal that makes use of the clusters in [116].

⁷The dimension of the state of a target is 4. Consequently, the dimension of the state of a cluster is $4N_c$ and therefore the number of sigma points is $2 \cdot 4N_c + 1$, see Section 2.6.2 for details.

Figure 4.3: Simulation scenario: (a) target trajectories and (b) the largest cluster size for the true target positions plotted against time for a clustering threshold $\Gamma_c = 200$ m. In (a), each target is identified at the beginning of its trajectory by a number and there is an arrow to indicate the target position every ten time steps.

Table 4.4: Simulation parameters

A priori model		Filter (binary)		Filter (raw)	
λ_A	$1/(100 \cdot \tau \cdot A_S)$	N_d	4	N_d	4
$1/\mu$	$(133 \cdot \tau)$	η_d	$5 \cdot 10^{-3}$	η_d	e^{-10}
σ_u	$5 \text{ m/s}^{3/2}$	N_b	3	N_b	3
		η_b	10^7	η_b	e^7

The sampling period of the trajectories is $\tau = 0.5$ s and there are $l = 130$ time steps in the simulation. The targets appear at different times. The common parameters to all the PFs used in the simulations are shown in Table 4.4. The surveillance area is a rectangle whose dimensions are $5000 \text{ m} \times 2500 \text{ m}$. In this area, there is a grid of 50×25 sensors whose side is 100 m. Each cell of the grid is one of the regions $V_1, \dots, V_M$ defined in Section 4.2.1. The SNR of a sensor produced by one target is given by⁸:

$$SNR(r) = \begin{cases} 20 \text{ dB} & r < 60 \text{ m} \\ 10 \text{ dB} & 60 \text{ m} \leq r < 80 \text{ m} \\ 0 & r \geq 80 \text{ m} \end{cases} \quad (4.51)$$

where r is the distance from the target to the sensor and $P_{fa} = 2 \cdot 10^{-5}$.

The tracking performance is evaluated by calculating the root mean square (RMS) of the position error by a Monte Carlo simulation with $m = 50$ realisations. To assess the tracking performance, we make a *Track-to-truth Assignment Table* [15] in which each track is associated with one target or with none at each

⁸This models corresponds to (4.14).

time step depending on the distance from the track to the real position of the target and a threshold that is taken to be $\Lambda = 120$ m. If the distance from a track to a target exceeds that threshold, the track cannot be associated with that target. The threshold for forming the clusters is $\Gamma_c = 200$ m. In addition, a track is assumed to be active if the sum of the weights containing the target exceed 0.7.

Let $\hat{\mathbf{p}}_i^k(j)$, $i = 1, \dots, t^k$, $k = 1, \dots, l$, $j = 1, \dots, m$ denote the position estimate of the i th target on the j th Monte Carlo realisation at time k , $e_i^k(j) = \|\mathbf{p}_i^k - \hat{\mathbf{p}}_i^k(j)\|_2$ denote the position error of the i th target on the j th Monte Carlo realisation at time k and $\chi_{i,j,k} = 1$ if $e_i^k(j) < \Lambda$ and zero otherwise. The variable $\chi_{i,j,k}$ indicates if the i th target is in track at time k on the j th Monte Carlo realisation. The time-averaged RMS position error E is then calculated by

$$E = \sqrt{\frac{\sum_{j=1}^m \sum_{k=1}^l \sum_{i=1}^{t^k} (e_i^k(j))^2 \cdot \chi_{i,j,k}}{\sum_{j=1}^m \sum_{k=1}^l \sum_{i=1}^{t^k} \chi_{i,j,k}}} \quad (4.52)$$

The time-averaged RMS position error using raw data for all the algorithms is shown in Figure 4.4(a). JA is the algorithm that performs more poorly due to the fact that it does not perform subparticle crossover when the targets are far apart. Therefore, this PF acutely suffers from the curse of dimensionality that consists of a drop in performance when the dimension of the state increases [45]. The UKF does not perform properly either. It is better than all the PFs when there are only 100 particles but its performance is far from the performance of PP when the number of particles is high because the measurement model is highly non-linear. Nevertheless, as pointed out in [139], the UKF has much better performance than JA in a multiple target tracking scenario. CP performs better than IP as was expected [116] but much worse than PP. This is due to the fact that IP and CP draw subparticles of the targets without taking into account the other targets. This is of crucial importance when raw data is used and there is a crossing in the target trajectories. Moreover, as AA does not perform subparticle crossover when the targets are in the same cluster, its performance is not as good as PP's because it is difficult to find particles in which all subparticles are good for representing the multitarget state when the number of particles is low. Nevertheless, when the number of particles increases, the importance of subparticle crossover decreases as it is easier to have particles in which all subparticles are appropriate for the multitarget state and the difference between AA and PP becomes smaller. On the other hand, MBRFS performance is far from PP, AA and CP performances. This was expected as MBRFS using raw data was developed under the assumption that the targets are far from each other [201].

The time-averaged RMS position error using binary data for all the algorithms is

Figure 4.4: Time-averaged RMS position error using raw data (a) and binary data (b). PP method outperforms the rest of the PFs because it performs subparticle crossover and considers nearby targets.

shown in Figure 4.4(b). The use of binary data implies a higher error as there is less information available. The same conclusions about the algorithms as when raw data is used can be drawn. However, as the number of particles increases, the difference between AA and PP is lower than before. This suggests that subparticle crossover is not so important with binary sensors as it is easier to find particles that agree with the measurement relaxing the need to find good subparticles among different particles.

The RMS position error and the estimated number of targets at each time step for raw data using 500 particles is shown in Figure 4.5. The simulation is divided into two intervals to explain the figure:

- First interval: Before time step 85. The four targets are far from each other: IP, CP, PP have the same performance as they are indeed the same PF when the targets are far from each other. AA has a high performance as well because it also uses subparticle crossover when the targets are not in the same cluster. MBRFS also provides good estimates as the targets are not close to each other. On the contrary, JA does not work properly because it does not use subparticle crossover and, then, it always requires a high number of particles. The UKF does not work properly either because the measurement model is highly nonlinear.
- Second interval: From time step 85 to 130. There are targets in close proximity. PP is the algorithm with the highest performance in RMS error and estimating the number of targets because it takes into account nearby targets and also

performs subparticle crossover. AA and UKF are roughly the algorithms with the second highest performance as they also take into account nearby targets. Conversely, IP and CP performances slump because they do not take into account targets in close proximity to propagate the particles. MBRFS does not perform well either because the targets are close to each other.

It should be clarified that the K-means algorithm for partition sorting has not been used in the PFs as suggested in [116]. K-means is an enforced method to speed up the labelling of the targets after a target crossing [24]. Then, including it would depart from the Bayesian perspective followed in this thesis. Nevertheless, it has also been implemented and, for this scenario, its effects are negligible as shown in Figure 4.6. This is because targets cross with different speed vectors so there are very few partition swappings [116].

The RMS error and the number of targets of the algorithms have been analysed using different metrics to show how the algorithms perform for these two different aspects. The optimal subpattern assignment (OSPA) metric can assess both performances simultaneously [176]. Therefore, for the sake of completeness, the mean OSPA (MOSPA) position error per time step is shown in Figure 4.5(c). The MOSPA error has been calculated with parameters $c = 120$ m and $p = 2$ [176]. The same conclusions can be drawn as with the previous metrics although MOSPA error does not indicate what part of the error corresponds to target number and to position error. At the beginning of the simulation, all the algorithms have a MOSPA error of $c = 120$ m as no targets have been added yet.

The number of surviving subparticles per time step is shown step in Figure 4.7. It should be noted that the implemented PFs perform a resampling stage after each time step. Then, the subparticles at time k are based on only some subparticles at time $k - 1$ (the surviving subparticles) because the rest have not been selected because of their low weight. The higher the number of surviving particles, the more likely it is that there are particles that represent the posterior well. A survival rate of between 30–80% is generally seen as acceptable according to [161]. Before time step 85, IP, CP, PP and AA have the same number of surviving subparticles. However, when the targets get closer, PP has the highest number of surviving subparticles and is able to maintain an acceptable survival rate. Before time step 4, all the algorithms have 100% survival rate as no targets have been added yet.

The execution times of the algorithms implemented in Matlab with C-MEX subroutines on a Pentium IV with a 2.4-GHz processor are shown in Table 4.5. The UKF is the fastest algorithm as it only approximates the first two moments of the posterior. In addition, processing raw data takes more time than processing binary

(a)

(b)

(c)

Figure 4.5: RMS position error (a) and estimated number of targets (b) using raw data per time step with 500 particles (c) MOSPA position error. Before time step 85, IP, CP, PP, AA and MBRFS have roughly the same performance because the targets are far from each other. When the targets get closer, PP outperforms the rest.

Figure 4.6: RMS position error for raw data using K-means partition swapping in all algorithms. The effects of K-means on performance are negligible in this scenario.

Figure 4.7: Percentage of surviving subparticles using 500 particles. PP has the highest survival rate.

Table 4.5: Execution times in seconds for 500 particles

	IP	CP	PP	AA	JA	UKF	MBRFS
Raw	32.7	35	34.3	37.4	39.2	0.5	5.5
Binary	29.8	30.5	30.2	30.1	32.1	0.5	5.4

data as the likelihood is faster to evaluate with binary data, see (4.16) and (4.18).

Lastly, the number of particles that are needed to obtain a certain performance in a scenario with an increasing number of targets is shown in Table 4.6. In the considered scenario, the number of targets whose trajectories cross at the same time is half the total number of targets and they do it in groups of two targets as shown in Figure 4.8. For a given error, MBRFS is able to track 32 targets with the same, albeit large, number of particles as with 4 targets because of its nonoverlapping assumption. This suggests that MBRFS would be the most suitable algorithm for tracking large numbers of targets. This conclusion is true only if targets remain well-separated. If several targets move in close proximity then the performance of MBRFS deteriorates, as seen in Figures 4.4 and 4.5. On the other hand, the proposed method (PP) provides a favourable trade-off between performance and computational expense. It is not limited to the case of well-separated targets and the required sample size increases by only three times as the number of targets is increased from 4 to 32. The remaining PFs fare much worse.

(a)

(b)

(c)

(d)

Figure 4.8: Scenario with increasing number of targets: (a) 4 targets (b) 8 targets (c) 12 targets (d) 16 targets. For more targets, this sequence continues. Note that the scaling of the x -axis changes.

Table 4.6: Required number of particles to obtain an RMS position error lower than 25 m

Method	Number of targets							
	4	8	12	16	20	24	28	32
PP	300	300	300	500	500	800	800	900
MBRFS	8700	8700	8700	8700	8700	8700	8700	8700
IP	>10000	>10000	>10000	>10000	>10000	>10000	>10000	>10000
CP	600	8300	>10000	>10000	>10000	>10000	>10000	>10000
AA	400	1600	2800	6200	>10000	>10000	>10000	>10000

4.5.2 Estimation of the number of targets

In this section, the two-layer particle filter is compared to the existence grid and the MBRFS with regard to the creation and deletion of tracks [61, 141]. In the existence grid, the foregoing importance densities $q_2(\cdot)$ and $q_3(\cdot)$ are determined by the grid as it indicates the targets that die and the newborn targets. The pixelised sensor model is used because the existence grid was designed to deal with this sensor model [141].

The existence grid provides a measure of the chances that a target is located in a cell, denoted as g_j^k in cell j at time k [141]. This existence grid is updated sequentially according to the measurement, the prior probability of adding a target in a cell α and the prior probability of removing a target β . In the simulations, rather than using the equation of the importance density to remove targets using the existence grid, equation (53) in [141], which permits a target removal probability exceeding one⁹, the following probability of removing target i is used:

$$\tau_i^k = \frac{\beta \cdot \left(1 - g_{v_i^k}^k\right)}{\sum_{l=1}^{t^{k-1}} \left(1 - g_{v_l^k}^k\right)} \quad (4.53)$$

where v_i^k denotes the cell occupied by the i th target at time k . In (4.53), the division of the denominator by the number of targets, which appears in (53) of [141], has been removed. The resulting target removal probability is at most β . The existence grid does not take into account target dynamics because the equation that models them, equation (4.10), is not used. Besides, the existence grid does not account for the possibility that a target might move to another cell. Then, the performance of the existence grid is much lower than the performance of the two-layer PF, especially when the targets move fast. In this case, performance mainly refers to the ability of the filter to create and delete tracks.

The scenario is shown in Figure 4.9. The prior parameters are the same as in the previous example, see Table 4.4, but with $\sigma_u = 10 \text{ m/s}^{3/2}$. The parameters of the two-layer PF are $N_b = 3$, $\eta_b = e^7$, $N_d = 5$, $\eta_d = e^{-2}$. The parameters for the existence grid are $\alpha = 2 \cdot 10^{-15}$ and $\beta = 0.9$. The surveillance area is a square whose side is 5000 m long. This region is divided into 50×50 cells and each cell scans a $100 \text{ m} \times 100 \text{ m}$ area.

The estimated number of targets using raw data with 500 particles is shown in Figure 4.10. Firstly, it should be noted that the existence grid does not work for

⁹This happens for instance when there are two targets, $g_1^k = 0.99$, $g_2^k = 0.001$ and $\beta = 0.6$, then $\tau_2^k = 1.188$ which is not correct as it is a probability.

Figure 4.9: Trajectories of the targets used to analyse the estimation of the number of targets

targets that move fast. It properly creates tracks but fails to delete them. MBRFS has trouble detecting target 4 as it is in close proximity to target 5 at time 80 because it cannot deal with targets in close proximity. With low SNR , MBRFS does not detect target 4. MBRFS clearly overestimates the number of targets at around time 110 when targets 1 and 6 cross. The two-layer PF estimates the number of targets better than the other methods. Its main problem is that there is a slight delay in the creation and deletion of tracks but clearly controlled by N_b and N_d .

As the SNR increases, MBRFS and the two-layer PF are able to estimate the number of targets faster and more accurately. The main difference between the performances at 10 dB and 20 dB is the speed with which tracks are initiated and deleted. The delayed track initiation at low SNR can be attributed to two factors. First, the creation of pre-tracks is often delayed because the probability of receiving a sufficiently strong measurement is lower at low SNR . Second, fewer pre-tracks are transferred to the track PF because, at low SNR , tracking is more difficult and the likelihood ratio (4.46) tends to be lower. The difficulty of tracking and the decrease in the likelihood ratio also explain the slower track deletion seen at low SNR .

For completeness, the RMS position error of the algorithms is also shown in Figure 4.10. The existence grid performs much worse than the two-layer PF and the MBRFS. These two algorithms perform similarly as regards RMS position error in this scenario. This is due to the fact that not many targets are in close proximity at the same time as analysed in Section 4.5.1. However, as explained before, the MBRFS detects the targets much worse and the analysis of this capability of detecting an unknown and variable number of targets was the purpose of the simulation scenario in this section.

4.5 SIMULATION RESULTS

Figure 4.10: Estimated number of targets and RMS position error with time: (a)-(b) Two-layer PF. (c)-(d) MBRFS. (e)-(f) Existence grid. The two-layer PF estimates the number of targets better than the rest.

4.6 Conclusions

A new method for the detection and tracking of multiple targets using the JMKPD has been proposed. The JMKPD is based on the mathematical formulation in [146] adding unique labels to the target states. Therefore, it allows the construction of tracks based on target state estimates over time when the number of targets is unknown and changes with time. Importantly, the labelling of the tracks, which is necessary to maintain the correspondence of target states over time, is done within the equations of the filter without any ad hoc methods. The inclusion of the labels also implies that, in principle, one is able to determine the probability that a target state estimate corresponds to a given target. This is of special interest after several targets have moved in close proximity and then separated as there could be confusion as to which target state estimate corresponds to each target. In this thesis, this problem is referred to as extraction of target labelling information and is explained in the next chapter for two targets.

The PP method has been devised to sample the posterior PDF in an efficient way and has been shown to provide better performance than other methods proposed previously. The benefits of the PP method are more important when the number of targets in close proximity is high and the number of particles is low. As for sampling the number of targets, a two-layer PF has been proposed. This way the states of the newborn targets are proposed via another PF and their dynamics can be taken into account to initiate tracks. The current form of the algorithm creates and deletes tracks based on LRTs independently for each track. While this approach has a low computational burden, it could be useful to use multiple hypothesis tests on the whole multitarget state to decide which targets should be removed. This would need much more computational resources but it is expected to improve detection performance in even more challenging scenarios. Besides, the use of fixed time lags and thresholds to evaluate the LRTs for creating and deleting tracks could be improved by selecting variable time lags and thresholds using sequential detection. These issues are also addressed in the next chapter.

It should be mentioned that the PP method will be used as part of a novel algorithm for tracking multiple targets with unknown sensor positions to be described in Chapter 6.

Chapter 5

Improved detection and extraction of target labelling information

5.1 Introduction

This chapter addresses two different topics that build on the technique presented in the previous chapter. It should be recalled that the algorithm in the previous chapter assumes that targets appear in isolation to detect them. This chapter proposes an improved method for multiple target detection for a fixed number of targets when this assumption is removed. In addition, track labels were included but the previous chapter does not address how to extract this information using the posterior PDF. Extracting this information means that the filter is able to determine the probability that a target state estimate corresponds with a specific target.

This chapter is organised as follows. Section 5.2 tackles the first problem and Section 5.3 the second.

5.2 Improved multiple target detection algorithms via sequential hypothesis tests

In the previous chapter, the problem of detection and tracking a variable and unknown number of targets in a surveillance area was addressed. However, the proposed method of creating tracks, or equivalently detecting targets, has several drawbacks:

- It only initiates one pre-track in the area of influence of a sensor that has a powerful measurement but there could be more.
- The labelling procedure of the model only allows one track to be created in the influence area of the sensor, see (4.7). However, there could be more targets appearing in the influence area of this sensor. This is not a problem as the labelling procedure can directly be generalised so that a certain number of tracks can be initiated in each cell.
- It applies a Neyman-Pearson detector to each one of the subparticles in the pre-track PF to decide if the subparticles represent real targets as explained in Section 4.4.2. This is perfectly valid because the support of the resulting importance density contains the support of the prior. Therefore, the PF approximation converges to the true posterior for increasing number of particles, see Section 2.5. Nevertheless, one should note that the real confirmation of the tracks takes place once the weights of the particles in the track PF, which includes the subparticles of the newborn targets, have been calculated. The same applies to the removal of tracks. The problem with this method is that the pre-track PF does not provide the most likely hypothesis (the pre-track is a real track or the pre-track is a false track) from the Bayesian point of view. Therefore, the performance of the subparticle Neyman-Pearson detector is not optimal. This is clarified throughout this section and a practical example is given in Section 5.2.3.3.
- It uses a fixed time-lag to create tracks or discard pre-tracks. Depending on the sequence of measurements, one could know fairly accurately the number of targets in an area in fewer time steps. Therefore, this scheme results in a delay in the creation tracks. The same applies to the removal of tracks.

These drawbacks are unimportant if targets are expected to appear isolated and delay is not a problem. In addition, these simplifications enabled the development of a filter with low computational burden and simplified the description of the algorithm. In this section, the problem of determining the number of targets that are in the area of interest is addressed from the Bayesian point of view taking these considerations into account. Therefore, the aim is not to detect and track an unknown and variable number of targets as in the previous chapter but to determine the fixed number of targets that there are. The algorithm developed in this section can be used to improve the creation of new tracks proposed in the previous chapter as the above-mentioned drawbacks are overcome.

Determining the fixed number of moving targets in the area of interest can be seen as a model assessment problem in which the aim is to choose the model from

a set of models that best fits the data [51]. In the Bayesian framework, this kind of problem is typically solved by calculating the evidence, which is used to compare the performance of different models at describing the data [62]. However, in nonlinear/non-Gaussian systems, the evidence is generally impossible to calculate analytically and, then, it must be approximated. Particle filters are a successful technique for filtering in such conditions [169] and, therefore, algorithms to calculate the evidence have been proposed when the importance density is the prior [20,51] or has a general form [5]. The inconvenience of these methods is that they use a fixed number of particles per hypothesis. Then, the computational resources are not assigned properly because models with low posterior probability take up valuable computational power.

As explained in Section 4.2, in a multiple target set-up, the models differ in the dimension of the state vector, which is proportional to the number of targets. Then, there is another possible approach to determine the model that best fits the data using the joint multitarget probability density (JMPD) [116]. The JMPD simultaneously estimates the number of targets with their states. Then, the idea is to initially draw particles with different number of targets and, as a consequence of recursive Bayesian filtering, the posterior distribution will tend to the right number of targets¹. In this case, the models with highest posterior probabilities have a higher number of particles and, then, the computational resources per hypothesis are assigned according to their posterior probability increasing the efficiency of the method. However, as will be explained, the way this method calculates the evidence is heavily influenced by particle degeneracy.

In this section, two sequential tests based on the previous ideas are built, their weaknesses are underlined and a new test with higher performance is proposed. The new test manages the computational resources among hypotheses and calculates the evidence using a robust method against particle degeneracy. The use of a sequential test is motivated by the fact that the observations are processed sequentially and it is desired to make the decision as soon as the decision quality is good enough.

The rest of this section is organised as follows. The problem is posed in a Bayesian framework in Section 5.2.1. Section 5.2.2 describes three sequential tests using particle filters. Finally, a case study is given in Section 5.2.3.

¹Note that for a scenario with fixed but unknown number of targets, the JMPD is analogue to the JMKPD as no labels are required to create tracks. This is explained in Section 5.2.2.2.

5.2.1 Formulation of the problem

The objective is to determine the fixed number of moving targets that there are in a certain area of interest. Mathematically, it can be seen as a multiple hypothesis test where one has to decide which of the following hypotheses $H_0, H_1, \dots, H_{N_h}$ best fits the data where N_h is the maximum number of targets that there could be and the hypothesis H_t stands for “there are t targets in the area of interest”. As throughout this thesis, the state of target i at time k is given by the four-dimensional vector $\mathbf{x}_i^k = [x_i^k, \dot{x}_i^k, y_i^k, \dot{y}_i^k]^T$ where $\mathbf{p}_i^k = [x_i^k, y_i^k]^T$ is the position vector and $\mathbf{v}_i^k = [\dot{x}_i^k, \dot{y}_i^k]^T$ is the velocity vector. The multitarget state vector is formed by stacking the individual target state vectors into one vector $\mathbf{X}^k = \left[(\mathbf{x}_1^k)^T, (\mathbf{x}_2^k)^T, \dots, (\mathbf{x}_t^k)^T \right]^T$. The measurement vector at time k is $\mathbf{z}^k$ and $\mathbf{z}^{1:k} = (\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^k)$ denotes the measurements up to time k .

The dynamic model of the targets is given by:

$$\mathbf{X}^k = \mathbf{f}(\mathbf{X}^{k-1}, \mathbf{w}^k) \quad (5.1)$$

where $\mathbf{f}(\cdot, \cdot)$ is a possibly nonlinear function, $\mathbf{w}^k$ is the process noise at time k and the prior PDF of state at time 0 is known $p(\mathbf{X}^0)$.

The measurement model is:

$$\mathbf{z}^k = \mathbf{h}(\mathbf{X}^k, \boldsymbol{\eta}^k) \quad (5.2)$$

where $\mathbf{h}(\cdot, \cdot)$ is a possibly nonlinear function and $\boldsymbol{\eta}^k$ is the measurement noise at time k .

In order to determine which hypothesis is the best one according to the data, it is necessary to calculate:

$$p(H_t | \mathbf{z}^{1:k}) = \frac{p(\mathbf{z}^{1:k} | H_t) p(H_t)}{p(\mathbf{z}^{1:k})} \quad (5.3)$$

where $p(H_t)$ is the prior probability of hypothesis t ,

$$p(\mathbf{z}^{1:k}) = \sum_{t=0}^{N_h} p(\mathbf{z}^{1:k} | H_t) p(H_t) \quad (5.4)$$

and $p(\mathbf{z}^{1:k} | H_t)$ is the evidence for hypothesis t [62] that can be written as:

$$p(\mathbf{z}^{1:k} | H_t) = p(\mathbf{z}^1 | H_t) \prod_{l=2}^k p(\mathbf{z}^l | \mathbf{z}^{1:l-1}, H_t) \quad (5.5)$$

To determine the number of targets, a sequential multihypothesis test is used. It consists of a stopping rule and a final decision rule [14]:

$$S_k = \text{first } k \text{ such that } p(H_t | \mathbf{z}^{1:k}) > \Gamma_t \quad (5.6)$$

$$\Theta = H_t, \quad \text{where } t = \arg \max_i p(H_i | \mathbf{z}^{1:k}) \quad (5.7)$$

where S_k is the stopping time, Θ is the final decision, Γ_t is the threshold applied to the posterior probability of the hypothesis H_t . Thus, if a posterior probability of the hypothesis H_t is higher than the threshold Γ_t , the sequential test stops and the hypothesis with highest posterior probability is chosen². Then, in order to carry out the sequential test, one needs to evaluate (5.3) which can be computed calculating the evidence for all the hypotheses, equation (5.5), and the prior probabilities of the hypotheses that are assumed to be the same.

When the collection of measurements up to time k are independent, the way to select Γ_t for each hypothesis so that the test is optimal according to some costs is analysed in [14]. However, in the system described by (5.1) and (5.2), the measurements up to time k are independent only when conditioned on the collection of multitarget states up to time k (see Section 2.2 for more details) so the results in [14] do not apply here.

5.2.2 Sequential multiple hypothesis tests

In this section, two sequential tests based on results previously available in the literature are built. In addition, a new algorithm that combines their good properties is developed. All the tests can work with any importance density.

5.2.2.1 Fixed number of particles sequential test

Let us assume there is a PF approximation of the posterior PDF at time k :

$$p(\mathbf{X}^{0:k} | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{X}^{0:k} - \mathbf{X}_i^{0:k}) \quad (5.8)$$

²Note that if $\Gamma_t > 0.5 \forall t \in \{0, 1, \dots, N_h\}$ there is only one hypothesis such that $p(H_t | \mathbf{z}^{1:k}) > \Gamma_t$ as the posterior probabilities sum to one.

where N_{par} is the number of particles, $\delta(\cdot)$ is the Dirac delta, $\mathbf{X}_i^{0:k}$ is the state of particle i up to time k and w_i^k is the weight of particle i at time k . The weights are normalised such that they sum to one. The particles and weights are chosen using the principle of importance sampling. This implies that the particles are drawn from an importance density $q(\mathbf{X}^{0:k} | \mathbf{z}^{1:k})$ and their weights meet the following proportionality:

$$w_i^k \propto \frac{p(\mathbf{X}_i^{0:k} | \mathbf{z}^{1:k})}{q(\mathbf{X}_i^{0:k} | \mathbf{z}^{1:k})} \quad (5.9)$$

If the importance density is chosen to factorise such that

$$q(\mathbf{X}^{0:k} | \mathbf{z}^{1:k}) = q(\mathbf{X}^k | \mathbf{X}^{0:k-1}, \mathbf{z}^{1:k}) q(\mathbf{X}^{0:k-1} | \mathbf{z}^{1:k-1}) \quad (5.10)$$

then samples can be obtained recursively and the weight update equation can be shown to be [169]

$$w_i^k \propto w_i^{k-1} \frac{p(\mathbf{z}^k | \mathbf{X}_i^k) p(\mathbf{X}_i^k | \mathbf{X}_i^{k-1})}{q(\mathbf{X}_i^k | \mathbf{X}_i^{0:k-1}, \mathbf{z}^{1:k})} \quad (5.11)$$

The aim, which is to approximate the evidence, equation (5.5), is done recursively by [5]

$$p(\mathbf{z}^k | \mathbf{z}^{1:k-1}) \approx \sum_{i=1}^{N_{par}} \tilde{w}_i^k \quad (5.12)$$

where the unnormalised weights $\tilde{w}_i^k$ are defined as

$$\tilde{w}_i^k = \frac{p(\mathbf{z}^k | \mathbf{X}_i^k) p(\mathbf{X}_i^k | \mathbf{X}_i^{k-1})}{q(\mathbf{X}_i^k | \mathbf{X}_i^{0:k-1}, \mathbf{z}^{1:k})} w_i^{k-1} \quad (5.13)$$

This derivation can also be found in Appendix A.15. Then, it is important to notice that the PDFs in (5.11) need to be known up to a normalising constant to update the weights of the particles. However, they need to be evaluated exactly to calculate the evidence, equation (5.13), as will be indicated in Section 5.2.3.1 for the auxiliary particle filter.

The test is built in the following way. For each hypothesis, there are a fixed number of particles which are propagated. The evidence is calculated sequentially using (5.12). Once a hypothesis meets condition (5.6), the test stops. The main advantage of this method is that the weights of the particles at time k only update the factor $p(\mathbf{z}^k | \mathbf{z}^{1:k-1}, H_t)$ in (5.5) rather than the whole $p(\mathbf{z}^{1:k} | H_t)$. When there is a resampling stage and some particles are killed off, it only affects that factor instead of the complete evidence. Therefore, this method is regarded as robust to

resampling (in comparison to the method explained in the next section). However, the problem arises when the number of particles is finite and especially if the number of hypotheses is high, as the number of particles in each hypothesis does not change with time and, consequently, computer power is not used wisely. Consequently, this test is referred to as fixed number of particles sequential test (FNP-ST).

5.2.2.2 Sequential test based on the JMPD

The JMPD is defined as the PDF $p(\mathbf{X}^k, t | \mathbf{z}^{1:k})$ where t is the number of targets recalling that the number of targets is constant over time. Then, assuming there is a particle approximation to the JMPD, the probability that there are t targets is [116]:

$$p(t | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} I_i(t) w_i^k \quad (5.14)$$

where $I_i(t) = 1$ if there are t targets in particle i or zero otherwise. As the hypotheses are the number of targets in the region, one must note that $p(H_t | \mathbf{z}^{1:k}) = p(t | \mathbf{z}^{1:k})$. Then, as the sequential test is based on $p(H_t | \mathbf{z}^{1:k})$, equation (5.6), this algorithm can also be used to build the test, referred to as joint multitarget probability density sequential test (JMPD-ST).

Initially, particles are drawn with different numbers of targets according to the prior. Then, equation (5.14) is computed at each time step. When a hypothesis meets condition (5.6), the test stops. In this case, resampling allows particles to change from one hypothesis to another. This increases the efficiency of the method as hypotheses with low probability will have a low number of particles and will not consume computational power. However, it should be noted that the posterior probabilities of the hypotheses at time k , given by (5.14), depend only on the weights of the particles at time k . It should be recalled that in FNP-ST, the weights at time k are used to update the factor $p(\mathbf{z}^k | \mathbf{z}^{1:k-1}, H_t)$ in (5.5) rather than the whole evidence $p(\mathbf{z}^{1:k} | H_t)$. This way the effect of particle degeneracy is more acute in JMPD-ST than in FNP-ST as the whole trajectory of a particle is used to calculate $p(H_t | \mathbf{z}^{1:k})$. Then, this method is not considered to be robust as regards particle degeneracy.

There is an interesting point that is worth explaining. In Chapter 4, it was indicated that it was necessary to include labels in the state vector in order to detect and track an unknown and variable number of targets. Therefore, it led to the definition of the JMKPD. Nonetheless, in this section, the labels have not been included. The reason for this is that, here, it is assumed that the number of targets

is fixed even though it is not known how many of them there are. Conditioned on an hypothesis, correspondence between target state estimates along time can be established without the need of labels. Therefore, in this case, it is not necessary to include the labels as they are implicit in the ordering of the multitarget state vector, then, the JMKPD comes down to the JMPD.

5.2.2.3 Variable number of particles sequential test

A new test called variable number of particles sequential test (VNP-ST) is developed. This test overcomes the drawbacks of the preceding methods to calculate the evidence, i.e., the FNP-ST uses a fixed number of particles per hypothesis and JMPD-ST calculates the evidence in a way that is heavily affected by particle degeneracy.

VNP-ST calculates the evidence using the approach of FNP-ST but allowing particles to move from one hypothesis to another in the resampling phase according to their posterior probabilities as in JMPD-ST. The algorithm assumes that the total number of particles of all the hypotheses is fixed N_{par} but the number of particles of each hypothesis at each time can change. Therefore, $N_{par} = \sum_{t=0}^{N_h} N_{par}^{t,k}$ where $N_{par}^{t,k}$ is the number of particles of hypothesis t and time k . Then, the posterior PDF for the t th hypothesis is represented by $\left\{ \left(w_{i,t}^k, \mathbf{X}_{i,t}^k \right) \right\}_{i=1}^{N_{par}^{t,k}}$ where $\mathbf{X}_{i,t}^k$ is the i th particle of the t th hypothesis at time k and $w_{i,t}^k$ its corresponding weight. After the resampling phase, the t th hypothesis is represented by the evenly distributed samples $\left\{ \left(1/N_{par}^{t,k+1}, \mathbf{X}_{i,t}^{*k} \right) \right\}_{i=1}^{N_{par}^{t,k+1}}$. The resampling step is done using systematic resampling accounting for the difference in the particle number. Finally, the steps of the adaptation of the number of particles are shown in Table 5.1.

5.2.3 Case study

In this section, the expression of the unnormalised weights, given by (5.13), is derived for the auxiliary particle filter. Then, the performances of the three sequential tests are analysed. Lastly, a comparison with the subparticle Neyman-Pearson detector to create new tracks is also given.

5.2.3.1 Implementation for the auxiliary particle filter

It is shown how to calculate the unnormalised weights, used by FNP-ST and VNP-ST, for the auxiliary particle filter. As explained in Section 2.5.3, the auxiliary particle filter uses an importance density $q(\mathbf{X}^k, i | \mathbf{z}^{1:k})$ which samples the pair

Table 5.1: Adaptation of the number of particles per hypothesis in VNP-ST

- Approximate $p(\mathbf{X}^k | \mathbf{z}^{1:k}, H_t)$ using a PF with any importance density and $p(H_t | \mathbf{z}^{1:k})$, (5.12) and (5.13), for each hypothesis. $\left\{ \left(w_{i,t}^k, \mathbf{X}_{i,t}^k \right) \right\}_{i=1}^{N_{par}^k}$
- Calculate the number of particles for each hypothesis:
 - Hypothesis $t \neq 0$: $N_{par}^{t,k+1} = \left\lfloor p(H_t | \mathbf{z}^{1:k}) \cdot N_{par} \right\rfloor$
 - Hypothesis $t = 0$: $N_{par}^{0,k+1} = N_{par} - \sum_{i=1}^{N_h} N_{par}^{i,k+1}$
- Resampling for the hypotheses. For each hypothesis $t = 0, \dots, N_h$
 - Construct the cumulative density function (CDF) of $w_{i,t}^k$ over $i = 1, \dots, N_{par}^{t,k}$: $(c_1, \dots, c_{N_{par}^{t,k}})$
 - * Assign $c_1 = w_{1,t}^k$
 - * For each particle $i = 2, \dots, N_{par}^{t,k}$
 - $c_i = c_{i-1} + w_{i,t}^k$
 - Start at the bottom of the CDF: $i = 1$.
 - Draw a starting point: $u_1 \sim U[0, 1/N_{par}^{t,k+1}]$.
 - For each particle $i = 1, \dots, N_{par}^{t,k+1}$
 - * Move along the CDF: $u_i = u_1 + (i - 1) / N_{par}^{t,k+1}$
 - * While $u_i > c_j$
 - $j = j + 1$
 - * Set $\mathbf{X}_{i,t}^{*k} = \mathbf{X}_{j,t}^k$

$\left\{ \left(\mathbf{X}_j^k, i^j \right) \right\}_{j=1}^{N_{par}}$ where i^j refers to the index of the particle at time $k-1$:

$$q \left(\mathbf{X}^k, i \mid \mathbf{z}^{1:k} \right) = q \left(i \mid \mathbf{z}^{1:k} \right) q \left(\mathbf{X}^k \mid i, \mathbf{z}^{1:k} \right) \quad (5.15)$$

where

$$q \left(\mathbf{X}^k \mid i, \mathbf{z}^{1:k} \right) \triangleq p \left(\mathbf{X}^k \mid \mathbf{X}_i^{k-1} \right) \quad (5.16)$$

and

$$q \left(i \mid \mathbf{z}^{1:k} \right) = \frac{p \left(\mathbf{z}^k \mid \boldsymbol{\mu}_i^k \right) w_i^{k-1}}{\sum_{n=1}^{N_{par}} p \left(\mathbf{z}^k \mid \boldsymbol{\mu}_n^k \right) w_n^{k-1}} \quad (5.17)$$

where $\boldsymbol{\mu}_i^k$ is some characterisation of $\mathbf{X}^k$ given $\mathbf{X}_i^{k-1}$ such as the mean or a sample from the prior [169]. The importance density for the particle index, given by (5.17), must be normalised, such that the sum over all the indexes is one, to evaluate (5.13). Then, using (5.17) and the procedure in [169], the unnormalised weight for particle j is:

$$\tilde{w}_j^k = \frac{p \left(\mathbf{z}^k \mid \mathbf{X}_j^k \right)}{p \left(\mathbf{z}^k \mid \boldsymbol{\mu}_{ij}^k \right)} \sum_{n=1}^{N_{par}} p \left(\mathbf{z}^k \mid \boldsymbol{\mu}_n^k \right) w_n^{k-1} \quad (5.18)$$

5.2.3.2 Simulations

It is assumed that there are a maximum of N_h targets ($N_h + 1$ hypotheses) in the region of interest. Each target moves independently following a nearly constant velocity model as in Section 4.2 with parameter $\sigma_u = 5 \text{ m/s}^{3/2}$ and the sampling period is $\tau = 0.5 \text{ s}$. This defines the process equation, see (5.1).

The prior PDF of the state at time 0 is

$$p \left(\mathbf{X}^0 \right) = \prod_{j=1}^t \mathcal{N} \left(\mathbf{p}_j^0; \bar{\mathbf{p}}^0, \boldsymbol{\Sigma}_p^0 \right) \mathcal{N} \left(\mathbf{v}_j^0; \bar{\mathbf{v}}^0, \boldsymbol{\Sigma}_v^0 \right) \quad (5.19)$$

where $\bar{\mathbf{p}}^0 = [2950, 1220]^T$, $\boldsymbol{\Sigma}_p^0 = 400\mathbf{I}_2$, $\bar{\mathbf{v}}^0 = [0, 0]^T$, $\boldsymbol{\Sigma}_v^0 = 9\mathbf{I}_2$ and the units are given in the international system.

There is a grid of M sensors of side 100 m that forms a rectangle whose dimensions are 5000 m $\times$ 2800 m, therefore there are 50×28 sensors, $M = 1400$. The measurement at time k is $\mathbf{z}^k = [z_1^k, z_2^k, \dots, z_M^k]^T$ where z_r^k is the measurement taken by sensor r at time k . The measurement equation at sensor r is:

$$z_r^k = h_r \left(\mathbf{X}^k \right) + \eta \quad (5.20)$$

Figure 5.1: Trajectories of the targets. Each target is identified at the end of its trajectory by a number and there is an arrow to indicate the target position every ten time steps.

where η is a zero-mean Gaussian noise with unit variance,

$$h_r(\mathbf{X}^k) = \sqrt{\sum_{j=1}^t SNR(d_{j,r,k})} \quad (5.21)$$

$$SNR(d_{j,r,k}) = \begin{cases} SNR_0 & d_{j,r,k} \leq d_0 \\ \frac{SNR_0 d_0^2}{d_{j,r,k}^2} & d_{j,r,k} > d_0 \end{cases} \quad (5.22)$$

$$d_{j,r,k} = \sqrt{(x_j^k - m_{x,r})^2 + (y_j^k - m_{y,r})^2} \quad (5.23)$$

and $[m_{x,r}, m_{y,r}]^T$ are the coordinates of sensor r , SNR_0 is the maximum signal-to-noise ratio produced by a target and d_0 is the saturation distance. Equation (5.21) models the amplitude of the received acoustic signal at a sensor from incoherently emitting targets neglecting the propagation time [178]. As the nonlinear measurement model depends only on the target positions and the relationship of the position and velocity is linear and Gaussian, Rao-Blackwellisation is applied and, consequently, the velocities are removed from the particle filter as they can be computed exactly using a Kalman filter, see Section 2.5.4 for details.

The trajectories of the targets used in the simulations are those shown in Figure 5.1. The hypothesis H_1 is evaluated with target one, hypothesis H_2 with targets one and two, hypothesis H_3 with targets one, two and three and hypothesis H_4 with targets one, two, three and four.

In the simulations $\Gamma_t = \Gamma = 0.99$ is used, i.e., the thresholds for all the hypotheses are the same. This implies that the hypotheses are not prioritised as the hypothesis whose posterior probability exceeds Γ is chosen. How to calculate the thresholds in

a dynamic system in an optimal way is well worth studying. The saturation distance is set to $d_0 = 10$ m and the maximum number of targets is $N_h = 5$.

For each hypothesis, the posterior PDF is approximated by an auxiliary particle filter that uses the predicted mean with 500 particles at the beginning. The probabilities of classification for true hypotheses from H_0 to H_4 are calculated based on Monte Carlo simulation with 100 runs for $SNR_0 = 6$ dB. The results are shown in Figure 5.2. JMPD-ST and VNP-ST have approximately the same classification performance when there are zero, one or two targets but VNP-ST is clearly better with three or four targets. In general, FNP-ST has the lowest performance.

It can also be noticed that the classification gets worse as the number of targets increases. This is easily understood by having a look at the measurement equation for one sensor, equation (5.20). If it is assumed that there are t targets and all of them are located at a distance lower than d_0 , the received signal follows a distribution $\mathcal{N}(z_r^k; \sqrt{t \cdot SNR_0}, 1)$. Using the results in Appendix A.9, the Kullback-Leibler (KL) divergence between the measurement PDFs when there are t targets and $t+1$ targets, is:

$$D_{KL}(p_t || p_{t+1}) = \frac{1}{2} \left(c + \left(\sqrt{(t+1) SNR_0} - \sqrt{t \cdot SNR_0} \right)^2 \right) \quad (5.24)$$

where c is a constant that does not depend on t or SNR_0 . As shown in Appendix A.16, the KL divergence is a strictly decreasing function of t which indicates that the more targets there are, the closer the PDFs of the measurements conditioned on the state are, and, therefore, the more difficult it is to classify. This fact also affects the stopping time. As shown in Figure 5.3, the stopping time for H_4 is higher than for H_1 . In general, when there are more targets, the stopping time increases because the KL divergence between the PDFs of the probable hypotheses is smaller. In the same way, the KL divergence is a strictly increasing function of SNR_0 , as also shown in Appendix A.16. Then, when SNR_0 increases, the stopping time is shorter as it is easier to classify because the KL divergence between posterior PDFs increases. When there is one target, FNP-ST and VNP-ST have roughly the same stopping time but when there are four targets VNP-ST determines the number of targets more quickly.

To sum up, when there are up to two targets, the algorithms have approximately the same performance as the posterior PDFs conditioned on each hypothesis are very different. However, for three or four targets, the posterior PDFs are closer and VNP-ST improves classification accuracy and reduces the time to reach a decision.

5.2 IMPROVED MULTIPLE TARGET DETECTION ALGORITHMS VIA SEQUENTIAL HYPOTHESIS TESTS

Figure 5.2: Probability of classification for the algorithms: VNP-ST has the best performance in general, especially, when the number of targets is three or four.

Figure 5.3: Stopping time for one and four targets. VNP-ST has a shorter stopping time in general.

Figure 5.4: Receiver operating characteristic for the subparticle Neyman-Pearson detector and fixed number of particles test.

5.2.3.3 Comparison with the subparticle Neyman-Pearson detector

In the previous chapter, a track was initiated based on a Neyman-Pearson detector applied to each of the subparticles of the pre-track particle filter, see (4.46). In this section, a comparison between this subparticle Neyman-Pearson (SNP) detector and the FNP-ST is provided as to their one-target detection capabilities. It should be noticed that here only one target is considered as the SNP detector used in (4.46) allows only one target to be initiated. It is compared with the FNP-ST because both algorithms represent the target state with the same number of particles at all time steps. Also, a fixed time lag, denoted as N_{birth} , is used to evaluate both tests as the SNP detector was designed for a fixed time lag. Therefore, the FNP-ST calculates (5.3) using N_{birth} time steps and then chooses the most likely hypothesis.

All the parameters of the simulations are those in Section 5.2.3 including the use of the auxiliary particle filter. It is considered that the SNP detector detects a target if at least 70% of the particles pass the test (4.46) and $N_{birth} = 4$. The receiver operating characteristics (ROC), which are a plot of the probability of detection against the probability of false alarm [193], for $SNR_0 \in \{6\text{dB}, 9\text{dB}, 12\text{dB}\}$ are shown in Figure 5.4. The FNP test outperforms the SNP detector. This is because the posterior of the two hypotheses for the FNP test, given by (5.3), are approximated from a Bayesian point of view while the SNP detector is an ad hoc detector. When SNR_0 increases, the difference between the detectors performance is lowered.

5.3 Extraction of target labelling information

In this thesis, extraction of target labelling information refers to the ability of a filter to determine the probability that a target state estimate corresponds to a given target. This is of special interest after several targets have moved in close proximity and then separated as there could be confusion as to which target state estimate corresponds to each target. For example, assume there are two vehicles in a certain area. One is allied and the other is an enemy and they are being tracked by a sensor network equipped with seismic sensors. The vehicles have the same characteristics so the signal measured by the seismic sensors cannot tell them apart and their motions either. However, in the beginning, when the targets are far from each other, the association of target state estimates and labels is obvious, i.e., it is known which one is the friendly one and which one is the enemy. The problem arises when they get closer and then separate. In this case, it is not straightforward to indicate the probability that a target state estimate corresponds to a specific target and one must obtain this information from the posterior PDF. This scenario is shown in Figure 5.5.

Figure 5.5: Target labelling problem. Three questions are to be answered once the targets have moved in close proximity and separated: Where are the targets? Which label should be assigned to each target state estimate with the lowest probability of wrong labelling? What is the probability of wrong labelling of the multitarget state estimate?

A very important feature that must be highlighted from the beginning is that the possibility of extracting labelling information properly depends on two different aspects of the algorithm: the approximation of the posterior PDF and the estimator of the multitarget state applied to this approximation of the posterior PDF. Both of them have to be carefully chosen to be able to extract the labelling information

properly and estimate the target states. Conventional particle filters with finite number of particles fail to do the former and estimators based on the MMSE fail to do the latter. This section explains why this information cannot be properly extracted using MMSE estimators based on conventional particle filters and, then addresses how to extract target labelling information when there are a fixed and known number of targets. Therefore, the following discussion is based on this case.

There are two usual approaches to represent the multitarget state in multiple target tracking from the Bayesian perspective. One uses sets [131] and the other uses vectors as done in this thesis. Usual set notation implies that targets are not labelled and, thus, one is not interested in knowing which target state estimate corresponds to each target [185] unless explicit labels are included in the target states [129]. Adopting the usual (unlabelled) set notation implies that all that matters is to estimate the target states. Contrarily, in vector notation, the order implicit in the components of a vector implies that targets are labelled when the number of targets is fixed and known as pointed out in Section 4.3³. Therefore, in this case, if the posterior PDF of the multitarget state is known, the information regarding target labels is included in the posterior, i.e., the probability that a target state estimate corresponds to a given target. It should be noted that if labelled set notation is used, the procedure to extract target labelling information is exactly the same as the one indicated in this section, which uses vector notation, because of the bijection that exists between labelled sets and vectors as indicated in Appendix B.

In the previous chapter, a Bayesian procedure to detect and track a variable and unknown number of targets was presented. It should be recalled that tracking from a Bayesian point of view was achieved by labelling the targets. However, it was not addressed how to extract the information about target labelling. This is due to the fact that information about target labelling is difficult to extract. This section poses the problem in a general Bayesian framework such that the probability of successful labelling of an estimator can be calculated.

It should be mentioned that the notion that the probability of successful labelling is linked to an estimator (of the multitarget state) is a new contribution of this thesis. Sufficient conditions the posterior for two targets must meet so that there is not any certainty at all or there is total certainty about the target labels are stated in [18] (see Remarks 1 and 2 in [18]). However, [18] does not address the general case in which there is a probability of successful labelling of an estimator that is different from one or one half. In addition, as will be indicated later on, the condition for which there is total certainty about the target labels is not sufficient as

³It should be noted that if the number of targets is variable and unknown the labels must explicitly be included in the state as in Chapter 4.

stated in [18], it is necessary but not sufficient. Other approaches to extract target labelling information such as [17] were published at the same time of the results presented in this section. Yet, the technique proposed in [17] is not general in the sense that it does not apply to general measurement models.

Particle filters are a technique to approximate the posterior PDF when the system is nonlinear non-Gaussian [169]. However, the necessary resampling stage used to overcome particle degeneracy implies that they are unable to reliably retain multiple modes of the posterior for extended periods [22,24]. The problem is that, after a target crossing, the posterior PDF could be multimodal and it is in this multimodality where the information about target labelling lies. Therefore, a conventional PF is bound to lose this multimodality [22,24] and, consequently, unable to provide the probability that a target state estimate corresponds to a given target. In this chapter, a PF especially designed to provide information about target labelling is proposed as it is able to maintain the multimodality of the posterior after a crossing. Even though the two-target case is addressed, the methodology can be generalised for multiple targets in principle.

As mentioned before, information about target labelling is inherently linked to state estimation. When two targets have always been far from each other, the MMSE estimator works well in the sense that it can estimate the positions of the targets (and it gives estimates where the targets actually are) and one can extract target labelling information. However, when two targets are in close proximity for a long time and then separate, the MMSE estimator always lies in the middle of the true positions of the targets [22,24]. Therefore, the MMSE estimator is not useful. On the contrary, the minimum mean optimal subpattern assignment (MMOSPA) estimator is expected to estimate the states of the targets properly at the expense of not labelling the estimated target states [83]. However, this original formulation of the MMOSPA estimator cannot be directly applied to the labelling problem as the estimated targets are not labelled and, consequently, one cannot obtain the probability of successful labelling. As a result, in this section, a couple of methods are proposed to label the MMOSPA estimator. The main drawback of the MMOSPA estimator is that it is very difficult to calculate and, therefore, its implementation in a dynamic system has only been obtained under some approximations [39,40]. Nevertheless, the PF proposed in this chapter readily enables the use of an estimator that roughly approximates the MMOSPA estimator with optimal labelling under certain conditions, i.e., it maximises the probability of successful labelling once the targets have separated after being in close proximity. In general, this estimator gives rise to lower MOSPA errors than the MMSE and maximum a posteriori (MAP) estimators without additional computational burden.

Prior to delving into technical details, it should be mentioned that in order to exploit the information about target labelling the notation used to denote the PDFs in most parts of the thesis, in which the PDF is described by its argument, is no longer appropriate. Therefore, now PDFs are represented by different letters that have nothing to do with their arguments as pointed out in Section 2.3.

The remainder of the section is organised as follows. In Section 5.3.1, the problem is introduced in the Bayesian framework. In Section 5.3.2, the previous work on how target labelling information can be obtained from the posterior PDF is revisited. Section 5.3.3 defines the concept of probability of successful labelling. Section 5.3.4 analyses the probability of successful labelling when the targets have separated after being in close proximity. Section 5.3.5 addresses the PF used to extract the information about target labelling. Target state estimation is addressed in Section 5.3.6. Simulations are included in Section 5.3.7.

5.3.1 Problem statement

As mentioned before, a two-target scenario is considered. The multitarget state vector at time k is represented by $\mathbf{X}^k = \left[(\mathbf{x}_1^k)^T, (\mathbf{x}_2^k)^T \right]^T \in \mathbb{R}^8$ where $\mathbf{x}_j^k$ is the state vector for target $j \in \{1, 2\}$ and the superscript T denotes transpose. The state vector for target j is $\mathbf{x}_j^k = [x_j^k, \dot{x}_j^k, y_j^k, \dot{y}_j^k]^T$ where $\mathbf{p}_j^k = [x_j^k, y_j^k]^T$ is the position vector and $\mathbf{v}_j^k = [\dot{x}_j^k, \dot{y}_j^k]^T$ is the velocity vector.

The posterior PDF of the state given the sequence of measurements up to time k $\mathbf{z}^{1:k} = (\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^k)$ can be calculated in two stages: prediction and update [9]. Assume that the posterior PDF at time k is available $\pi^k(\mathbf{X}^k)$. The PDF of the predicted state, prior PDF at time $k+1$, is calculated using the Chapman–Kolmogorov equation:

$$\varpi^{k+1}(\mathbf{X}^{k+1}) = \int f(\mathbf{X}^{k+1} | \mathbf{X}^k) \pi^k(\mathbf{X}^k) d\mathbf{X}^k \quad (5.25)$$

where $f(\mathbf{X}^{k+1} | \mathbf{X}^k)$ is known and represents the dynamic equation for the targets. It is also assumed that the targets move independently with similar dynamic models:

$$f(\mathbf{X}^{k+1} | \mathbf{X}^k) = f_d(\mathbf{x}_1^{k+1} | \mathbf{x}_1^k) f_d(\mathbf{x}_2^{k+1} | \mathbf{x}_2^k) \quad (5.26)$$

where $f_d(\cdot)$ is the PDF that models the dynamics of one target and is the same for both targets.

The update equation is given by Bayes' rule:

$$\pi^{k+1}(\mathbf{X}^{k+1}) \propto \ell(\mathbf{z}^{k+1} | \mathbf{X}^{k+1}) \varpi^{k+1}(\mathbf{X}^{k+1}) \quad (5.27)$$

where $\propto$ denotes proportionality and $\ell(\mathbf{z}^{k+1} | \mathbf{X}^{k+1})$ is the likelihood.

It is assumed that measurements do not provide any information about target identities. This implies that the likelihood is permutation invariant:

$$\ell(\mathbf{z}^{k+1} | \mathbf{X}^{k+1}) = \ell(\mathbf{z}^{k+1} | \mathbf{\Pi X}^{k+1}) \quad (5.28)$$

where

$$\mathbf{\Pi} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \otimes \mathbf{I}_4 \quad (5.29)$$

where $\mathbf{I}_m$ is the $m \times m$ identity matrix, $\otimes$ is the Kronecker product and, consequently, $\mathbf{\Pi}$ is a matrix that permutes target 1 and 2 in the state vector:

$$\mathbf{\Pi X}^k = \left[(\mathbf{x}_2^k)^T, (\mathbf{x}_1^k)^T \right]^T \quad (5.30)$$

Then, in principle, solving (5.25) and (5.27) recursively, one knows everything about the targets as all the available information can be found in the posterior including that about target labelling.

5.3.2 Information about target labelling: Previous work

This section deals with the previous work about the extraction of target labelling information. This is basically the unique decomposition of a PDF explained in [18]. Sufficient conditions the posterior PDF must meet so that there is no certainty at all about target labelling or absolute certainty about target labelling are stated in [18]. However, one of these conditions is not sufficient but necessary to ensure absolute certainty about target labelling. This is because [18] does not consider the fact that the information about target labelling depends on the estimator and the metric as will be explained in Section 5.3.3.

In the following, a summary on some results found in [18] is provided in Sections 5.3.2.1, 5.3.2.2 and 5.3.2.3. Moreover, the regions that play an important role in target labelling are identified in Section 5.3.2.4.

5.3.2.1 Permutation invariant PDF

The posterior PDF is defined to be permutation invariant (with respect to $\mathbf{\Pi}$) if

$$\pi^k(\mathbf{X}^k) = \pi^k(\mathbf{\Pi X}^k) \quad \forall \mathbf{X}^k \in \mathbb{R}^8 \quad (5.31)$$

In other words, one can interchange the state of both targets in the posterior

and the posterior is not altered. Therefore, it may be known where the two targets are, but each of the two target labels are equally probable, i.e., the correspondence between estimated states and target labels is not known. Then, in this case, all the information regarding the labels of the targets has been lost and any estimator will have a probability of successful labelling equal to one half. It should be noted that this is a sufficient condition for having a probability of successful labelling equal to one half but not necessary, see Appendix A.17.

An interesting property that provides more insight into the problem can be noticed rewriting (5.31) as

$$\pi^k(\mathbf{X}^k) = \frac{1}{2} [\pi^k(\mathbf{X}^k) + \pi^k(\mathbf{\Pi X}^k)] \quad (5.32)$$

The corresponding random finite set (RFS) density of the ordered posterior density, given by (5.27), is [83, 131]:

$$\pi^k(\{\mathbf{x}_1^k, \mathbf{x}_2^k\}) = \pi^k(\mathbf{X}^k) + \pi^k(\mathbf{\Pi X}^k) \quad (5.33)$$

Therefore, comparing (5.32) and (5.33), it is clear that the RFS density and the ordered density are proportional when the ordered density is permutation invariant⁴. The reason for this is that the information of order (labelling) is lost if the posterior is permutation invariant. As a result, the posterior has exactly the same information as the posterior of a set.

5.3.2.2 Permutation strictly variant PDF

The posterior PDF is defined to be permutation strictly variant (with respect to $\mathbf{\Pi}$) if

$$\pi^k(\mathbf{X}^k) \cdot \pi^k(\mathbf{\Pi X}^k) = 0 \quad \forall \mathbf{X}^k \in \mathbb{R}^8 \quad (5.34)$$

This implies that for every possible state $\mathbf{X}^k$, such that $\pi^k(\mathbf{X}^k) > 0$, $\mathbf{\Pi X}^k$ is not possible. This is a necessary condition so that there is no uncertainty at all regarding target labelling, however, it is not sufficient, see Appendix A.17. Note that in [18], it was stated that this is a sufficient condition as [18] did not take into account that the probability of successful labelling actually depends on the estimator and a metric as explained in Section 5.3.3.

⁴There is a proportionality constant because set integrals [131] are defined differently than their vector counterparts.

5.3.2.3 Unique decomposition of a PDF

When the posterior PDF is not either invariant or strictly variant, it is shown in [18] that it admits a unique decomposition

$$\pi^k(\mathbf{X}^k) = \alpha^k \pi_I^k(\mathbf{X}^k) + (1 - \alpha^k) \pi_V^k(\mathbf{X}^k) \quad (5.35)$$

where $\pi_V^k(\mathbf{X}^k)$ is a permutation strictly variant PDF and $\pi_I^k(\mathbf{X}^k)$ is a permutation invariant PDF and

$$\alpha^k = \int \min [\pi^k(\mathbf{X}^k), \pi^k(\Pi \mathbf{X}^k)] d\mathbf{X}^k \quad (5.36)$$

$$\pi_I^k(\mathbf{X}^k) = \min [\pi^k(\mathbf{X}^k), \pi^k(\Pi \mathbf{X}^k)] / \alpha^k \quad (5.37)$$

$$\pi_V^k(\mathbf{X}^k) = [\pi^k(\mathbf{X}^k) - \alpha^k \pi_I^k(\mathbf{X}^k)] / (1 - \alpha^k) \quad (5.38)$$

Therefore, the parameter $\alpha^k \in [0, 1]$ indicates how invariant a posterior PDF is.

5.3.2.4 Regions of interest of the unique decomposition

Here, an equivalent way to calculate the variant and invariant part of a PDF, given by (5.37) and (5.38), is described. This new interpretation will allow us to build a PF that preserves multimodality with an easier notation in Section 5.3.5.

The region A^k is defined as

$$\begin{aligned} A^k &= \{\mathbf{X}^k \in \mathbb{R}^8 : \pi^k(\mathbf{X}^k) < \pi^k(\Pi \mathbf{X}^k)\} \\ &= \{\mathbf{X}^k \in \mathbb{R}^8 : \varpi^k(\mathbf{X}^k) < \varpi^k(\Pi \mathbf{X}^k)\} \end{aligned} \quad (5.39)$$

and its complement

$$\bar{A}^k = \{\mathbf{X}^k \in \mathbb{R}^8 : \mathbf{X}^k \notin A^k\} \quad (5.40)$$

where (5.27) and (5.28) have been used. There are two important aspects that should be highlighted. The first one is that if the posterior is permutation invariant or permutation strictly variant, $A^k = \emptyset$ and $\bar{A}^k = \mathbb{R}^8$. The second one is that if $\alpha^k \in (0, 1)$:

$$\mathbf{X}^k \in A^k \leftrightarrow \Pi \mathbf{X}^k \in \bar{A}^k \quad (5.41)$$

According to the definitions (5.39) and (5.40), (5.37) and (5.38) can be written as

$$\pi_I^k(\mathbf{X}^k) = [\pi^k(\mathbf{X}^k) \chi_{A^k}(\mathbf{X}^k) + \pi^k(\Pi \mathbf{X}^k) \chi_{\bar{A}^k}(\mathbf{X}^k)] / \alpha^k \quad (5.42)$$

$$\pi_V^k(\mathbf{X}^k) = \chi_{\bar{A}^k}(\mathbf{X}^k) \left[\pi^k(\mathbf{X}^k) - \pi^k(\Pi \mathbf{X}^k) \right] / (1 - \alpha^k) \quad (5.43)$$

and α^k is

$$\begin{aligned} \alpha^k &= \int \pi^k(\mathbf{X}^k) \chi_{A^k}(\mathbf{X}^k) d\mathbf{X}^k + \int \pi^k(\Pi \mathbf{X}^k) \chi_{\bar{A}^k}(\mathbf{X}^k) d\mathbf{X}^k \\ &= 2 \int \pi^k(\mathbf{X}^k) \chi_{A^k}(\mathbf{X}^k) d\mathbf{X}^k \end{aligned} \quad (5.44)$$

and $\chi_{A^k}(\cdot)$ is the indicator function on the region A^k and it has been assumed that $\alpha^k \in (0, 1)$. Then, if it is needed to estimate α^k accurately, the posterior PDF in regions A^k and $\bar{A}^k$ should be approximated appropriately. The reason is due to (5.44) in which it is shown that α^k can be calculated as two times the integral of the posterior in region A^k .

5.3.2.5 Example

In this section, an example of the decomposition of the posterior PDF presented before is given. In order to draw the contour plots of the posterior PDF, one-dimensional targets are considered. Therefore, the multitarget state vector has two dimensions. It is assumed that the posterior PDF at time k is

$$\pi^k(\mathbf{X}^k) = \mathcal{N}(\mathbf{X}^k; [10, 11]^T, \mathbf{I}_2) \quad (5.45)$$

This implies that the expected position of target one is 10 and the expected position of target two is 11. Since the posterior PDF is Gaussian, the regions A^k and $\bar{A}^k$ can be calculated analytically as can α^k . The contour plot of the posterior PDF and the regions A^k and $\bar{A}^k$ are shown in Figure 5.6. In this case, $\alpha^k = 0.48$ and the contour plots of the permutation strictly invariant PDF and variant PDF are shown in Figures 5.7 and 5.8. It should be highlighted that the invariant PDF has a symmetry with respect to the line $x_1^k = x_2^k$ because of the property of equation (5.31). On the other hand, the strictly variant PDF lies in the region $\bar{A}^k$ and is zero in region A^k .

5.3.3 On the probability of successful labelling

This section defines the concept of probability of successful labelling used in this thesis. This is presented for a two-target case but it is generalisable for more targets. It is assumed that the posterior PDF $\pi^k(\cdot)$ is known.

The probability of successful labelling of an estimate $\hat{\mathbf{X}}^k$ obviously depends on the definition of successful labelling. The estimate $\hat{\mathbf{X}}^k$ has the right labelling if

Figure 5.6: Variant-invariant decomposition example: Contour plot of the posterior PDF

Figure 5.7: Variant-invariant decomposition example: Contour plot of the permutation strictly invariant PDF

Figure 5.8: Variant-invariant decomposition example: Contour plot of the permutation strictly variant PDF

$$d(\hat{\mathbf{X}}^k, \mathbf{X}^k) < d(\hat{\mathbf{X}}^k, \Pi \mathbf{X}^k) \quad (5.46)$$

where $d(\cdot, \cdot)$ is a distance and $\mathbf{X}^k$ is the multitarget state. The distance $d(\cdot, \cdot)$ could be, for example, the Euclidean distance but this might not be the best one depending on the case. For example, if the target state includes positions and velocities, it might be better to use a Mahalanobis distance to weight differently each component as treating positions and velocities equivalently might not be the best approach. In any case, it should be noticed that the probability of successful labelling depends on a metric and the estimator.

If the region $\mathcal{S}^k(\hat{\mathbf{X}}^k)$ is defined as

$$\mathcal{S}^k(\hat{\mathbf{X}}^k) = \{\mathbf{X}^k : d(\hat{\mathbf{X}}^k, \mathbf{X}^k) < d(\hat{\mathbf{X}}^k, \Pi \mathbf{X}^k)\} \quad (5.47)$$

the probability of successful labelling $P_{sl}(\hat{\mathbf{X}}^k)$ of the estimator $\hat{\mathbf{X}}^k$ becomes

$$P_{sl}(\hat{\mathbf{X}}^k) = \int_{\mathcal{S}^k(\hat{\mathbf{X}}^k)} \pi^k(\mathbf{X}^k) d\mathbf{X}^k \quad (5.48)$$

The probability of wrong labelling $P_{wl}(\hat{\mathbf{X}}^k)$ is

$$P_{wl}(\hat{\mathbf{X}}^k) = 1 - P_{sl}(\hat{\mathbf{X}}^k) = \int_{\bar{\mathcal{S}}^k(\hat{\mathbf{X}}^k)} \pi^k(\mathbf{X}^k) d\mathbf{X}^k \quad (5.49)$$

where $\bar{\mathcal{S}}^k(\hat{\mathbf{X}}^k)$ is the complement of $\mathcal{S}^k(\hat{\mathbf{X}}^k)$.

5.3.3.1 Example

Let us reconsider the example in Section 5.3.2.5. If the Euclidean metric and the MMSE or MAP estimator are used, $\hat{\mathbf{X}}^k = [10, 11]^T$, the region $\mathcal{S}^k(\hat{\mathbf{X}}^k) = \bar{A}^k$ and, therefore:

$$P_{sl}(\hat{\mathbf{X}}^k) = 1 - \frac{\alpha^k}{2} = 0.76 \quad (5.50)$$

That is, if the real target state lies in $\bar{A}^k$, the MMSE and MAP estimators have the right labelling but if the real target state lies in A^k , the MMSE and MAP estimators have the wrong labelling.

In general, it should be noted that $\mathcal{S}^k(\hat{\mathbf{X}}^k) \neq \bar{A}^k$ so the probability of successful labelling is not related to α^k . However, there is one case of special interest in which the probability of successful labelling can be calculated based on the decomposition mentioned above and α^k . This happens when one is interested in knowing the

probability of successful labelling once the targets have separated after being in close proximity as shown in Figures 5.5 and 5.9. This is addressed in the next section.

5.3.4 Probability of successful labelling: Targets have separated after being in close proximity

The most common case in which one wants to know the probability of successful labelling of an estimator is when the targets have moved in close proximity and then separated. In this case, the posterior can be represented as

$$\pi^k(\mathbf{X}^k) \approx (1 - a_c) \pi_n^k(\mathbf{X}^k) + a_c \pi_{\Pi n}^k(\mathbf{X}^k) \quad (5.51)$$

where $\pi_n^k(\cdot)$ is centred with respect to a value $\tilde{\mathbf{X}}_n^k$, $a_c^k \in [0, 1/2]$ without loss of generality and $\pi_{\Pi n}^k(\cdot)$ is defined as

$$\pi_{\Pi n}^k(\mathbf{X}) = \pi_n^k(\Pi \mathbf{X}) \quad (5.52)$$

where it should be noted that $\pi_n^k(\cdot)$ and $\pi_{\Pi n}^k(\cdot)$ practically do not overlap when the targets are sufficiently far from each other. The justification for (5.51) is given in Appendix A.18. In the following, α^k and the probability of wrong labelling are calculated and their relationship is shown.

Calculation of α^k In this case, A^k corresponds with the region in which $\pi_{\Pi n}^k(\cdot)$ is high and $\pi_n^k(\mathbf{X}^k) \approx 0$ if $\mathbf{X}^k \in A^k$. Therefore:

$$\alpha^k = 2 \int \pi^k(\mathbf{X}^k) \chi_{A^k}(\mathbf{X}^k) d\mathbf{X}^k = 2a_c \quad (5.53)$$

Calculation of the probability of wrong labelling As $a_c \in [0, 1/2]$, it is assumed that any “good” estimate $\hat{\mathbf{X}}^k$ is closer to $\tilde{\mathbf{X}}_n^k$ than to $\Pi \tilde{\mathbf{X}}_n^k$. Therefore, when the targets are sufficiently far from each other it is met that

$$\pi_n^k(\mathbf{X}^k) \approx 0 \quad \text{if } \mathbf{X}^k \in \bar{\mathcal{S}}^k(\hat{\mathbf{X}}^k) \quad (5.54)$$

Therefore, using (5.49)

$$P_{wl} = a_c \quad (5.55)$$

where it has been highlighted that the probability of wrong labelling is independent of the estimator and the metric if the targets are sufficiently far from each other.

Relation between α^k and the probability of wrong labelling Using (5.53) and (5.55), the probability of wrong labelling is

$$P_{wl} = \alpha^k / 2 \quad (5.56)$$

Accordingly, the probability of successful labelling is

$$P_{sl} = 1 - \alpha^k / 2 \quad (5.57)$$

Therefore, if the posterior PDF at time k is known when the targets have separated after being in close proximity, the probability of wrong labelling is $\alpha^k / 2$. Importantly, this probability of wrong labelling is independent of the metric used to evaluate the wrong labelling and the actual estimator if the targets are sufficiently far from each other.

5.3.5 Particle filter implementation

The objective is to approximate the posterior PDF such that one is able to estimate α^k for systems that are not necessarily linear and Gaussian. In this case, PFs are a technique to approximate the posterior [169]. However, they have an important shortcoming: if the posterior is multimodal, they cannot keep multimodality for a long time due to the necessary resampling stage [196]. From its definition, $\pi_I^k(\cdot)$ is permutation invariant with respect to $\mathbf{\Pi}$. Then, if there is a mode centred at $\mathbf{X}^k$, there will be another centred at $\mathbf{\Pi X}^k$. Then, the permutation invariant part of the posterior PDF is multimodal and conventional PFs fail to estimate α^k properly as they can only approximate one mode for a long time [22, 24, 144].

Reliable estimation of α^k requires the posterior to be accurately approximated in the regions A^k and $\bar{A}^k$. With this in mind, (5.41) suggests a posterior approximation of the form:

$$\pi^k(\mathbf{X}^k) \approx \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{X}^k - \mathbf{X}_i^k) + u_i^k \delta(\mathbf{X}^k - \mathbf{\Pi X}_i^k) \quad (5.58)$$

where N_{par} is the number of particles in each region A^k and $\bar{A}^k$, $\delta(\cdot)$ is the Dirac delta, $\mathbf{X}_i^k$ is the multitarget state of particle i which belongs to $\bar{A}^k$ with weight w_i^k , $\mathbf{\Pi X}_i^k$ is the “mirror” particle of $\mathbf{X}_i^k$ in A^k with weight u_i^k . The estimated value of α^k , $\hat{\alpha}^k$, using (5.58) and (5.44) is:

$$\hat{\alpha}^k = 2 \sum_{i=1}^{N_{par}} u_i^k \quad (5.59)$$

Therefore, if a particle filter approximation to the posterior PDF is used and the number of particles tends to infinity, the posterior PDF at time k will be perfectly represented and the probability of wrong labelling can be estimated as $\hat{\alpha}^k/2$ when the targets are sufficiently far from each other, see Section 5.3.4. Importantly, this probability of wrong labelling is independent of the distance used to evaluate the wrong labelling if the targets are sufficiently far from each other. However, when the number of particles is finite and especially if it is low, no conclusions can be drawn.

The PF approximation of the posterior given by (5.58) can be seen as a PF that explores multimodality as it is known that multimodality can exist because of the property of the permutation invariant part of the PDF. If the posterior is unimodal, which can only happen if the posterior is strictly variant, $u_i^k = 0$ for $i = 1, \dots, N_{par}$.

In order to get a PF approximation of the posterior at time $k+1$ with the form (5.58) based on a PF approximation of the posterior at time k also with the form (5.58), there are two aspects that should be taken into account for any implementation regardless of the importance density. The first one is the approximation of A^{k+1} based on (5.58). The second one has to do with the fact that mirror samples $\mathbf{X}_i^k$ and $\Pi\mathbf{X}_i^k$ always need to be drawn. These aspects as well as the proposed PF implementation are addressed in this section.

5.3.5.1 Approximation of the region A^{k+1}

Assuming the posterior at time k is represented by (5.58), using (5.25), the prior at time $k+1$ is

$$\varpi^{k+1}(\mathbf{X}^{k+1}) = \sum_{i=1}^{N_{par}} w_i^k f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) + u_i^k f(\mathbf{X}^{k+1} | \Pi\mathbf{X}_i^k) \quad (5.60)$$

Substituting (5.60) into (5.39), the region A^{k+1} becomes:

$$\begin{aligned} A^{k+1} = & \left\{ \mathbf{X}^{k+1} \in \mathbb{R}^8 : \sum_{i=1}^{N_{par}} w_i^k f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) + u_i^k f(\mathbf{X}^{k+1} | \Pi\mathbf{X}_i^k) \right. \\ & \left. < \sum_{i=1}^{N_{par}} w_i^k f(\mathbf{X}^{k+1} | \Pi\mathbf{X}_i^k) + u_i^k f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) \right\} \end{aligned} \quad (5.61)$$

5.3.5.2 Exploration of the mirror mode

In the PF approximation of (5.58), the creation of the particles $\Pi\mathbf{X}_i^k$ $i = 1, \dots, N_{par}$ is forced because it is known that the posterior PDF in the area the symmetric particles lie could be high, especially after the targets have been close to each other,

and that part of the posterior carries important information about the target labels. This section addresses the importance density used to draw particles this way.

Let us assume an intermediate sample is drawn from an importance density $\tilde{\mathbf{X}} \sim v(\cdot)$ and define the conditional PDF:

$$v_c(\mathbf{X}^k | \tilde{\mathbf{X}}) = \frac{1}{2} \delta(\mathbf{X}^k - \tilde{\mathbf{X}}) + \frac{1}{2} \delta(\mathbf{X}^k - \Pi \tilde{\mathbf{X}}) \quad (5.62)$$

The PDF $v_c(\cdot | \tilde{\mathbf{X}})$ is perfectly represented by the particles $\mathbf{X}_i^k = \tilde{\mathbf{X}}$ and $\mathbf{X}_{i+N_{par}}^k = \Pi \tilde{\mathbf{X}}$ with even weights. Therefore, the particles used to represent the posterior can be thought to be drawn in the following way.

Firstly, one draws $\tilde{\mathbf{X}} \sim v(\cdot)$ and then assigns the two samples that represent (5.62), $\mathbf{X}_i^k = \tilde{\mathbf{X}}$ and $\mathbf{X}_{i+N_{par}}^k = \Pi \tilde{\mathbf{X}}$. The importance density $q(\cdot)$ that is being used to obtain $\mathbf{X}_i^k$ and $\mathbf{X}_{i+N_{par}}^k$ can be calculated by integrating out $\tilde{\mathbf{X}}$ in the joint PDF of $\mathbf{X}^k$ and $\tilde{\mathbf{X}}$:

$$\begin{aligned} q(\mathbf{X}^k) &= \int v_c(\mathbf{X}^k, \tilde{\mathbf{X}}) d\tilde{\mathbf{X}} = \int v_c(\mathbf{X}^k | \tilde{\mathbf{X}}) v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}} \\ &= \int \frac{1}{2} [\delta(\mathbf{X}^k - \tilde{\mathbf{X}}) + \delta(\mathbf{X}^k - \Pi \tilde{\mathbf{X}})] v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}} \end{aligned} \quad (5.63)$$

As shown in Appendix A.20, (5.63) can be written as

$$q(\mathbf{X}^k) = \frac{1}{2} (v(\mathbf{X}^k) + v(\Pi \mathbf{X}^k)) \quad (5.64)$$

5.3.5.3 Problems with conventional particle filters

According to the definition of α^{k+1} , $\alpha^{k+1} \in [0, 1]$ [18]. However, the conventional PFs based on sequential importance sampling or auxiliary sampling [9] can provide values of $\hat{\alpha}^{k+1} > 1$ as explained in the following.

Let us write the posterior, given by (5.27) and (5.60), as

$$\pi^{k+1}(\mathbf{X}^{k+1}, i) \propto \ell(\mathbf{z}^{k+1} | \mathbf{X}^{k+1}) \cdot \varpi^{k+1}(\mathbf{X}^{k+1}, i) \quad (5.65)$$

where

$$\varpi^{k+1}(\mathbf{X}^{k+1}, i) = w_i^k f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) + u_i^k f(\mathbf{X}^{k+1} | \Pi \mathbf{X}_i^k) \quad (5.66)$$

and the auxiliary variable i denotes an index in the mixture given by the prior, see (5.60), as in [165].

Let $(\tilde{\mathbf{X}}_i^{k+1}, j^i) \sim v(\cdot)$ and $\tilde{\mathbf{X}}_{i+N_{par}}^{k+1} = \Pi \tilde{\mathbf{X}}_i^{k+1}$ for $i = 1, \dots, N_{par}$ accounting for the fact that, according to (5.64), $q(\cdot)$ satisfies $q(\tilde{\mathbf{X}}_i^{k+1}, j^i) = q(\Pi \tilde{\mathbf{X}}_i^{k+1}, j^i)$.

Firstly, it should be noted that sampling and evaluating the weights for sequential importance sampling would be equivalent to this approach but using $j^i = i$. The weights of the particles are then

$$\tilde{w}_i^{k+1} \propto \frac{\ell(\mathbf{z}^{k+1} | \tilde{\mathbf{X}}_i^{k+1}) \varpi^{k+1}(\tilde{\mathbf{X}}_i^{k+1}, j^i)}{q(\tilde{\mathbf{X}}_i^{k+1}, j^i)} \quad (5.67)$$

$$\tilde{u}_i^{k+1} \propto \frac{\ell(\mathbf{z}^{k+1} | \tilde{\mathbf{X}}_i^{k+1}) \varpi^{k+1}(\Pi \tilde{\mathbf{X}}_i^{k+1}, j^i)}{q(\tilde{\mathbf{X}}_i^{k+1}, j^i)} \quad (5.68)$$

To estimate α^{k+1} , see (5.44), it is necessary to determine which particles belong to A^{k+1} and which ones to $\bar{A}^{k+1}$ using (5.61). Therefore, for $i = 1, \dots, N_{par}$, let

$$\left(\mathbf{X}_i^{k+1}, \mathbf{X}_{i+N_{par}}^{k+1}, w_i^{k+1}, u_i^{k+1} \right) = \begin{cases} \left(\tilde{\mathbf{X}}_i^{k+1}, \tilde{\mathbf{X}}_{i+N_{par}}^{k+1}, \tilde{w}_i^{k+1}, \tilde{u}_i^{k+1} \right) & \tilde{\mathbf{X}}_i^{k+1} \in \bar{A}^{k+1} \\ \left(\tilde{\mathbf{X}}_{i+N_{par}}^{k+1}, \tilde{\mathbf{X}}_i^{k+1}, \tilde{u}_i^{k+1}, \tilde{w}_i^{k+1} \right) & \text{otherwise} \end{cases} \quad (5.69)$$

After this re-arrangement, it is known that $\varpi^{k+1}(\mathbf{X}_i^{k+1}) > \varpi^{k+1}(\mathbf{X}_{i+N_{par}}^{k+1})$. However, this does not imply an inequality on the weights w_i^{k+1}, u_i^{k+1} , as they are based on $\varpi^{k+1}(\mathbf{X}_i^{k+1}, i)$ rather than $\varpi^{k+1}(\mathbf{X}^{k+1})$. Therefore, there is no guarantee that $\hat{\alpha}^{k+1} = 2 \sum_{i=1}^{N_{par}} u_i^{k+1} < 1$, in fact, because of the normalisation

$$\sum_{i=1}^{N_{par}} (w_i^{k+1} + u_i^{k+1}) = 1 \quad (5.70)$$

$\hat{\alpha}^{k+1}$ could reach the value 2 and this does not make sense.

It should be noted that conventional PFs have $O(N_{par})$ computational expense. However, to estimate α^{k+1} , it has to be calculated whether a sample belongs to A^{k+1} or $\bar{A}^{k+1}$. This process has a $O(N_{par}^2)$ computational complexity because of the evaluation of (5.61) and therefore, the overall computational expense is $O(N_{par}^2)$.

5.3.5.4 Marginal particle filter

To overcome the problem of the estimation of α^{k+1} , a marginal PF is employed [110]. A marginal PF, as explained in Section 2.5.2, operates directly on the posterior PDF at the current time. In the marginal PF, samples are drawn in the same way as in conventional particle filtering but the weights are computed as

$$\tilde{w}_i^{k+1} \propto \frac{\ell(\mathbf{z}^{k+1} | \tilde{\mathbf{X}}_i^{k+1}) \varpi^{k+1}(\tilde{\mathbf{X}}_i^{k+1})}{q(\tilde{\mathbf{X}}_i^{k+1})} \quad (5.71)$$

$$\tilde{u}_i^{k+1} \propto \frac{\ell(\mathbf{z}^{k+1} | \tilde{\mathbf{X}}_i^{k+1}) \varpi^{k+1}(\Pi \tilde{\mathbf{X}}_i^{k+1})}{q(\tilde{\mathbf{X}}_i^{k+1})} \quad (5.72)$$

where

$$q(\tilde{\mathbf{X}}_i^{k+1}) = \sum_{j=1}^{N_{par}} q(\tilde{\mathbf{X}}_i^{k+1}, j) \quad (5.73)$$

$\varpi^{k+1}(\cdot)$ is given by (5.60) and (5.64) has been used. After the re-arrangement indicated in the Section 5.3.5.3, it is met that $w_i^{k+1} \geq u_i^{k+1}$. As shown in Appendix A.21, this implies that $\hat{\alpha}^{k+1} \leq 1$ as required.

The exact implementation used in the simulations in Section 5.3.7 is the auxiliary marginal PF [110] in which the importance density for drawing normal particles is:

$$v(\mathbf{X}^{k+1}) \propto \sum_{i=1}^{N_{par}} \mu_i^{k+1} \frac{[w_i^k f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) + u_i^k f(\mathbf{X}^{k+1} | \Pi \mathbf{X}_i^k)]}{w_i^k + u_i^k} \quad (5.74)$$

where the so-called first-stage weights are

$$\mu_i^{k+1} \propto \ell(\mathbf{z}^{k+1} | \hat{\mathbf{X}}_i^{k+1}) (w_i^k + u_i^k) \quad (5.75)$$

and $\hat{\mathbf{X}}_i^{k+1}$ is the predicted state of particle i . The steps of the algorithm are shown in Table 5.2. The drawback of the marginal PF in usual problems is that it has a $O(N_{par}^2)$ computational complexity. However, conventional PFs also have $O(N_{par}^2)$ computational complexity to estimate α^{k+1} . Therefore, marginal PFs do not add computational complexity to this problem. An alternative importance density is proposed in Appendix A.22.

5.3.6 Estimation and labelling

The procedure proposed in this section to estimate and label target states is to estimate the target states without considering target labels and then assign the labels to the unlabelled estimate. This is because if labels are included directly in the estimator, the common estimators are the MMSE and the MAP and they present some drawbacks. If the targets have moved in close proximity for a long time, the MMSE estimator is not useful as it always provides estimates in the middle of the positions of the targets [18]. In this case, it is more advisable to use the MAP estimator [57] although the tracking output may switch in an uncontrolled way between the two target states [19]. Nonetheless, a suitable measure for analysing the performance of estimators of unlabelled targets is the OSPA metric [176]. Accordingly, the optimal estimator is the MMOSPA estimator [83] rather than the MAP or MMSE estima-

Table 5.2: Auxiliary marginal PF with mirror particles

- For $i = 1 : N_{par}$
 - Calculate the first-stage weights μ_i^{k+1} using (5.75).
- Normalise μ_i^{k+1} to sum to one over $i = 1 : N_{par}$.
- For $i = 1 : N_{par}$
 - Draw a sample from $v(\cdot)$, given by (5.74):
 - * Sample an index j from the distribution defined by $(\mu_1^{k+1}, \dots, \mu_{N_{par}}^{k+1})$.
 - * Draw $\tilde{\mathbf{X}}_i^{k+1}$ from $f(\cdot | \mathbf{X}_j^{k+1})$ with probability $w_j^k / (w_j^k + u_j^k)$ or from $f(\cdot | \Pi \mathbf{X}_j^{k+1})$ with probability $u_j^k / (w_j^k + u_j^k)$.
 - Evaluate the weights of $\tilde{\mathbf{X}}_i^{k+1}$ and the mirror $\Pi \tilde{\mathbf{X}}_i^{k+1}$, $\tilde{w}_i^{k+1}$ and $\tilde{u}_i^{k+1}$ using (5.71) and (5.72).
 - Rearrange the particles and weights to form $\mathbf{X}_i^{k+1}$, $\Pi \mathbf{X}_i^{k+1}$, w_i^{k+1} and u_i^{k+1} using (5.61).

tors. However, its computational expense is high as it requires an iterative algorithm and its original formulation does not provide any target labelling information.

In this section, firstly, a procedure for labelling the MMOSPA estimator is proposed. Then it is explained how to approximate it using a particle filter of the form given by (5.58).

5.3.6.1 Labelling of the MMOSPA estimator

The square OSPA distance between the sets X and $\hat{X}$ using the Euclidean metric without cut-off distance for two targets is [176]:

$$d_o^2(X, \hat{X}) = \frac{1}{2} \min \left[d^2(\mathbf{X}, \hat{\mathbf{X}}), d^2(\Pi \mathbf{X}, \hat{\mathbf{X}}) \right] \quad (5.76)$$

where $\mathbf{X}$ and $\hat{\mathbf{X}}$ are any ordered version of the sets X and $\hat{X}$ and, $d(\cdot, \cdot)$ is the Euclidean distance.

The MMOSPA estimator is:

$$\hat{\mathbf{X}} = \int_{\mathcal{A}(\hat{\mathbf{X}})} [\mathbf{x}_1, \mathbf{x}_2]^T \pi^k(\{\mathbf{x}_1^k, \mathbf{x}_2^k\}) d\mathbf{x}_1 d\mathbf{x}_2 \quad (5.77)$$

where $\pi^k(\{\mathbf{x}_1^k, \mathbf{x}_2^k\})$ is given by (5.33) and

$$\mathcal{A}(\hat{\mathbf{X}}) = \left\{ \mathbf{X} : d^2(\mathbf{X}, \hat{\mathbf{X}}) = \min \left[d^2(\mathbf{X}, \hat{\mathbf{X}}), d^2(\Pi\mathbf{X}, \hat{\mathbf{X}}) \right] \right\} \quad (5.78)$$

It should be noted that one problem to calculate (5.77) is that the region of integration $\mathcal{A}(\hat{\mathbf{X}})$ depends on the estimate $\hat{\mathbf{X}}$ and an iterative algorithm has to be used [83].

Once $\hat{\mathbf{X}}$ has been calculated, it does not contain any information about target labelling. In fact, $\hat{\mathbf{X}}$ and $\Pi\hat{\mathbf{X}}$ are both MMOSPA estimators. However, one can choose any of these MMOSPA estimators according to different criteria to define a labelled MMOSPA estimator.

Once such criterion is to choose the labelled MMOSPA estimator $\hat{\mathbf{X}}_o$ with the most likely ordering as

$$\hat{\mathbf{X}}_{o1} = \begin{cases} \hat{\mathbf{X}} & \text{if } \pi^k(\hat{\mathbf{X}}) > \pi^k(\Pi\hat{\mathbf{X}}) \\ \Pi\hat{\mathbf{X}} & \text{otherwise} \end{cases} \quad (5.79)$$

That is, the most likely ordering is the one with the highest value in the posterior PDF. Therefore, the labelled MMOSPA estimator with the most likely ordering $\hat{\mathbf{X}}_{o1} \in \bar{\mathcal{A}}^k$ and, therefore, the probability of successful labelling when the targets are well-separated after having moved in close proximity is $1 - \alpha^k/2$, see Section 5.3.3.

Another criterion is to choose the labelled MMOSPA estimator $\hat{\mathbf{X}}_{o2}$ that maximises the probability of successful labelling

$$\hat{\mathbf{X}}_{o2} = \begin{cases} \hat{\mathbf{X}} & \text{if } P_{sl}(\hat{\mathbf{X}}^k) > P_{sl}(\Pi\hat{\mathbf{X}}^k) \\ \Pi\hat{\mathbf{X}} & \text{otherwise} \end{cases} \quad (5.80)$$

where $P_{sl}(\hat{\mathbf{X}}^k)$ is given by (5.48). Once the targets are well-separated after having moved in close proximity, $\hat{\mathbf{X}}_{o2} \in \bar{\mathcal{A}}^k$ and the probability of successful labelling becomes $1 - \alpha^k/2$, see Section 5.3.3.

Note that when the targets are well-separated after having moved in close proximity, $\hat{\mathbf{X}}_{o1}$ and $\hat{\mathbf{X}}_{o2}$ become the same estimator.

5.3.6.2 Approximation of the labelled MMOSPA estimator using the particle filter with mirror particles

The PF representation of the form given by (5.58) directly allows the use of an estimator that produces lower MOSPA errors than the MAP or MMSE estimators

without increasing the computational burden as demonstrated with simulations in Section 5.3.7. The estimator exploits the knowledge of the regions A^k and $\bar{A}^k$ and its motivation is based on the Mahalanobis metric and a Gaussian approximation of the prior as explained in the following.

Let us use the Mahalanobis metric instead of the Euclidean metric in (5.76):

$$d_{om}^2(X, \hat{X}) = \frac{1}{2} \min \left[d_{m,S}^2(\mathbf{X}, \hat{\mathbf{X}}), d_{m,S}^2(\Pi\mathbf{X}, \hat{\mathbf{X}}) \right] \quad (5.81)$$

where

$$d_{m,S}(\mathbf{X}, \hat{\mathbf{X}}) = \sqrt{(\mathbf{X} - \hat{\mathbf{X}})^T \mathbf{S}^{-1} (\mathbf{X} - \hat{\mathbf{X}})} \quad (5.82)$$

is the Mahalanobis distance between $\mathbf{X}$ and $\hat{\mathbf{X}}$ with covariance matrix $\mathbf{S}$. Then, the Mahalanobis MMOSPA (MMMOSPA) estimator can be defined as

$$\hat{\mathbf{X}} = \int_{\mathcal{M}_S(\hat{\mathbf{X}})} [\mathbf{x}_1, \mathbf{x}_2]^T \pi^k(\{\mathbf{x}_1^k, \mathbf{x}_2^k\}) d\mathbf{x}_1 d\mathbf{x}_2 \quad (5.83)$$

where

$$\mathcal{M}_S(\hat{\mathbf{X}}) = \left\{ \mathbf{X} : d_{m,S}^2(\mathbf{X}, \hat{\mathbf{X}}) = \min \left[d_{m,S}^2(\mathbf{X}, \hat{\mathbf{X}}), d_{m,S}^2(\Pi\mathbf{X}, \hat{\mathbf{X}}) \right] \right\} \quad (5.84)$$

If it is assumed that the prior at time k , $\varpi^k(\cdot)$, is Gaussian with mean $\bar{\mathbf{X}}^k$ and covariance matrix Σ^k , using (5.39) and (5.40), then

$$\bar{A}^k = \mathcal{M}_{\Sigma^k}(\bar{\mathbf{X}}^k) \quad (5.85)$$

Therefore, the “first-step” Gaussian MMMOSPA estimator $\hat{\mathbf{X}}_{om}$ can be defined as

$$\hat{\mathbf{X}}_{om} = \int_{\bar{A}^k} [\mathbf{x}_1, \mathbf{x}_2]^T \pi^k(\{\mathbf{x}_1^k, \mathbf{x}_2^k\}) d\mathbf{x}_1 d\mathbf{x}_2 \quad (5.86)$$

where “first-step” refers to the fact that it would be the first step of the recursion to calculate (5.83) starting at the predicted mean with covariance matrix Σ^k . Using (5.33) and (5.58), the PF approximation of the first-step Gaussian MMMOSPA estimator is

$$\hat{\mathbf{X}}_{om} = \sum_{i=1}^{N_{par}} (w_i^k + u_i^k) \mathbf{X}_i^k \quad (5.87)$$

which is the posterior mean in $\bar{A}^k$ of the corresponding set PDF. The estimator (5.87) will be compared with MAP and MMSE estimators as regards MOSPA state error in the next section.

An important characteristic of the estimator given by (5.86) is that it always

provides estimates lying in the area $\bar{A}^k$. Then, after the targets have moved in close proximity and separated, $\hat{\mathbf{X}}_{om}$ is approximating the labelled MMOSPA estimators $\hat{\mathbf{X}}_{o1}$ and $\hat{\mathbf{X}}_{o2}$ and it has the right labelling with probability $1 - \hat{\alpha}^k/2$ (if the number of particles tends to infinity) as indicated in Section 5.3.3.

5.3.7 Simulations

This section analyses the labelling information and estimation performance of the proposed algorithm in a scenario in which two targets are being tracked by a sensor network. The targets follow a nearly constant velocity model [13]:

$$p_d(\mathbf{x}_j^{k+1} | \mathbf{x}_j^k) = \mathcal{N}(\mathbf{x}_j^{k+1}; \mathbf{F}\mathbf{x}_j^k, \mathbf{Q}) \quad (5.88)$$

$$\mathbf{F} = \mathbf{I}_2 \otimes \begin{pmatrix} 1 & \tau \\ 0 & 1 \end{pmatrix} \quad (5.89)$$

$$\mathbf{Q} = \sigma_u^2 \mathbf{I}_2 \otimes \begin{pmatrix} \tau^3/3 & \tau^2/2 \\ \tau^2/2 & \tau \end{pmatrix} \quad (5.90)$$

where $j \in \{1, 2\}$, $\mathcal{N}(\mathbf{x}; \bar{\mathbf{x}}, \mathbf{Q})$ is the Gaussian PDF evaluated at $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\mathbf{Q}$, τ is the sampling period, and σ_u^2 is the continuous-time process noise intensity.

There is a grid of M sensors of side 10 m that forms a rectangle whose dimensions are 500 m $\times$ 700 m, i.e., there are 50×70 sensors and $M = 3500$. The measurement at time k is $\mathbf{z}^k = [z_1^k, z_2^k, \dots, z_M^k]^T$ where z_r^k is the measurement taken by sensor r at time k and its equation for sensor r is:

$$z_r^k = h_r(\mathbf{X}^k) + \eta \quad (5.91)$$

where η is a zero-mean Gaussian noise with unit variance,

$$h_r(\mathbf{X}^k) = \sqrt{\sum_{j=1}^2 SNR(d_{j,r,k})} \quad (5.92)$$

$$SNR(d_{j,r,k}) = \begin{cases} SNR_0 & d_{j,r,k} \leq d_0 \\ \frac{SNR_0 d_0^2}{d_{j,r,k}^2} & d_{j,r,k} > d_0 \end{cases} \quad (5.93)$$

$$d_{j,r,k} = \sqrt{(x_j^k - m_{x,r})^2 + (y_j^k - m_{y,r})^2} \quad (5.94)$$

and $[m_{x,r}, m_{y,r}]^T$ are the coordinates of sensor r , SNR_0 is the maximum signal-to-noise ratio produced by a target and d_0 is the saturation distance. Equation (5.92)

Table 5.3: Parameters of the simulation

Parameter	Value
SNR_0	30 dB
d_0	1 m
σ_u	1.8 m/s ^{3/2}
τ	0.5 s
N_{par}	10000

characterises the likelihood, see (5.28), and models the amplitude of the received acoustic signal at a sensor from incoherently emitting targets.

The simulations use the parameters and the scenario, with 79 time steps, shown in Table 5.3 and Figure 5.9, respectively. The trajectory of target one is fixed but the trajectory of target two is moved depending on a parameter called d_x . When $d_x = 0$ m, both target states from time step 30 to 50 are exactly the same but they differ before time step 30 and after time step 50. Then, the separation of the trajectories in the x -axis is d_x . Both target trajectories have been generated using the dynamic equation (5.88) fixing one part of the trajectory. That is, first, the trajectory of target 1 was obtained. Then, trajectory of target 2 from time step 30 to 50 was copied from target 1. As of time step 50, trajectory of target 2 was obtained by another realisation from (5.88). From time step 30 to 1, the trajectory of target 2 was generated backwards inverting the dynamic equation.

The prior PDF of the j th target at time step 0 is

$$\mathbf{x}_j^0 \sim \mathcal{N}(\mathbf{p}_j^0; \bar{\mathbf{p}}_j^0, \sigma_{p0}^2 \mathbf{I}_2) \mathcal{N}(\mathbf{v}_j^0; \bar{\mathbf{v}}_j^0, \sigma_{v0}^2 \mathbf{I}_2) \quad (5.95)$$

where $\mathbf{p}_j^0$ and $\mathbf{v}_j^0$ are the position and velocity vectors of target j at time 0, $\bar{\mathbf{p}}_j^0$ and $\bar{\mathbf{v}}_j^0$ are the expected position and velocity of target j at time 0 and, $\sigma_{p0}^2 \mathbf{I}_2$ and $\sigma_{v0}^2 \mathbf{I}_2$ are the covariance matrices of the position and velocity at time 0. In each Monte Carlo run, $\bar{\mathbf{p}}_j^0$ is drawn from a Gaussian distribution whose mean is the real position of target j at time 0 and whose covariance matrix is $\sigma_{p0}^2 \mathbf{I}_2$ and $\bar{\mathbf{v}}_j^0$ is drawn from a Gaussian distribution whose mean is the real velocity of target j at time 0 and whose covariance matrix is $\sigma_{v0}^2 \mathbf{I}_2$. In the simulations, $\sigma_{0p} = 0.1$ m and $\sigma_{v0} = 1$ m/s.

The constellations of particles at different time steps for an exemplar run with $d_x = 0$ m are shown in Figure 5.10. Before the targets get close, the particles are labelled $w_i^k > u_i^k$. However, as they get in close proximity $\hat{\alpha}^k$ increases and u_i^k increases. When $w_i^k \approx u_i^k$, the labels of the i th particle are lost as the particle $\mathbf{X}_i^k$ and $\Pi \mathbf{X}_i^k$ are equally probable. After they separate, the estimated $\hat{\alpha}^k$ reaches a steady value maintaining multimodality because the proposed PF is designed to keep the information about target labelling after a crossing.

 Figure 5.9: Scenario of the simulations for $d_x = 8$ m.

 (a) $k = 26$ ($\hat{\alpha}^{26} = 0$)

 (b) $k = 27$ ($\hat{\alpha}^{27} = 3.6 \cdot 10^{-129}$)

 (c) $k = 41$ ($\hat{\alpha}^{41} = 0.67$)

 (d) $k = 57$ ($\hat{\alpha}^{57} = 0.996$)

Figure 5.10: Constellations of particles at different time steps for $d_x = 0$ m and 5000 particles. For representation purposes, a particle is regarded as labelled if $w_i^k > 3u_i^k$. Before the target crossing, particles are labelled. Red colour indicates the position of target 1 and green target 2. Finally, particles become blue that indicates that they have almost lost labelling information.

Table 5.4: Probability of successful labelling P_{sl} and $\hat{\alpha}^{79}$ against the distance between targets

$d_x(\text{m})$	Averaged $\hat{\alpha}^{79}$	P_{sl}
0	0.94	0.68
2	0.94	0.6
4	0.57	0.5
6	0.01	0.75
8	0.003	0.98
10	~ 0	1
12	0	1
14	0	1

For different values of d_x , a Monte Carlo simulation with 100 realisations is carried out to obtain the averaged $\hat{\alpha}^{79}$ (value of $\hat{\alpha}^k$ at $k = 79$, the last time step of the simulation) and the probability of successful labelling P_{sl} . As the value of α^k depends on the measurements, the same sequence of measurements are used for each d_x . The target state estimates are successfully labelled if they are closer to the target states with the right labelling at the final step of the simulation. However, it should be noted that in this scenario the probability of successful labelling is independent of the estimator and the distance used to define successful labelling after the targets have separated as pointed out in Section 5.3.3.

The results are shown in Table 5.4. When the number of particles tends to infinity, $P_{sl} = 1 - \hat{\alpha}^{79}/2$ but this is not met for a limited number of particles. For $d_x \geq 8$ m, $\hat{\alpha}^{79}$ is very low and P_{sl} is very high. This indicates that for low values of $\hat{\alpha}^{79}$, one should be confident about the target labelling. On the contrary, for $d_x \leq 4$ m, $\hat{\alpha}^{79}$ is over 0.5 and the probability of track swap is close to 0.5. Therefore, for these values of $\hat{\alpha}^{79}$, one cannot trust the labelling. In general, it can be seen that as $\hat{\alpha}^{79}$ gets lower, the probability of successful labelling roughly increases as expected.

The value of first, second and third quartiles for $\hat{\alpha}^k$ against time for $d_x = 0$ m and $d_x = 4$ m are plotted in Figure 5.11. For $d_x = 0$ m, $\hat{\alpha}^k$ is nearly one at the end of the simulation. This means that the targets are not distinguishable as they have been a long time in close proximity. For $d_x = 4$ m, $\hat{\alpha}^k$ reaches an intermediate value that indicates that there could be confusion in the target labels. In addition, the variance of the estimator of $\hat{\alpha}^k$ is high as indicated by the quartiles.

Figure 5.12 shows the MOSPA state error for a conventional auxiliary PF and the PF with mirror particles using the MAP [57], the posterior mean (PM), the estimator that chooses the particle with maximum weight (MW) [30] and the PM in $\bar{A}^k$, see (5.87). It should be noted that using the MAP as proposed in [57] for a

Figure 5.11: First, second and third quartiles of $\hat{a}^k$ plotted against time for $N_{par} = 10000$: In red colour: $d_x = 0$ m, in black colour: $d_x = 4$ m. For $d_x = 0$ m, $\hat{a}^k$ is close to one at the end as expected.

Figure 5.12: MOSPA state error for a conventional auxiliary PF and the PF with mirror particles using MAP, the posterior mean (PM), the particle with maximum weight (MW) and the PM in $\bar{A}^k$: a) $d_x = 0$ b) $d_x = 4$. In the figure, Abar indicates $\bar{A}^k$. Averaged over all time steps, the posterior mean in $\bar{A}^k$ produces lower MOSPA errors.

Table 5.5: Time-averaged MOSPA state error

$d_x(\text{m})$	0	2	4	6	8	10	12	14
PM (conv PF)	4.27	4.96	4.04	2.68	2.71	2.58	2.55	2.44
MW (conv PF)	4.12	3.94	3.91	3.83	3.81	3.76	3.72	3.62
MAP (conv PF)	4.13	3.97	3.92	3.82	3.80	3.75	3.74	3.62
PM (mirror PF)	9.94	10.22	7.72	2.74	2.60	2.56	2.53	2.43
MW (mirror PF)	4.14	3.94	3.91	3.86	3.81	3.77	3.73	3.63
MAP (mirror PF)	4.10	3.97	3.90	3.85	3.81	3.76	3.73	3.60
PM in $\bar{A}^k$ (mirror PF)	2.99	2.81	2.78	2.66	2.58	2.56	2.53	2.43

conventional PF increases its computational complexity to $O(N_{par}^2)$, the same as a marginal PF. For $d_x = 2\text{ m}$, one of the 100 realisations diverged for both PFs and the results shown are averaged over the convergent realisations. When the targets separate, the PM should lie somewhere in the middle of the real target states due to the multimodality of the posterior. This behaviour can be seen in the Figure 5.12 as of time step 50 when the MOSPA error of the PM for both filters increases considerably. The MAP and MW estimators do not have this property and are able to maintain lower errors after the targets separate. However, before the targets separate, their MOSPA state errors are higher than the error of the PM and the PM in $\bar{A}^k$. By contrast, the PM in $\bar{A}^k$ achieves low MOSPA state errors during the whole simulation. Additionally, it should be noted that before the targets get together, i.e., before time 30, the PM and the PM in $\bar{A}^k$ for both the conventional and mirror PFs produce the same error.

The MOSPA state errors averaged over all time steps for the algorithms are shown in Table 5.5. In general, averaged over all time steps, the PM in $\bar{A}^k$ achieves lower MOSPA errors than the other estimators. In addition, its performance over the rest of the estimators gets better as d_x decreases.

5.4 Conclusions

This chapter addresses the detection of a fixed number of targets and the extraction of target labelling information from a Bayesian point of view. In the first part of the chapter, two results available in the literature have been applied to the problem of sequential multiple target detection using PFs. When they are applied directly, the problems which arise have been pointed out. Then, a new algorithm (VNP-ST) has been proposed that overcomes their deficiencies, i.e., it uses an adaptive number of particles for each hypothesis according to their posterior probabilities and the evidence approximation is done in a robust way against particle degeneracy. VNP-ST outperforms other tests in accuracy and time lag with an improvement that gets

higher with more challenging scenarios. In addition, it has been shown that for sensor networks which measure acoustic signals emitted incoherently from targets, it gets more and more difficult to determine the number of targets in the scenario as the real number of target increases. A future improvement of the algorithm is to set the thresholds in an optimal manner.

The second part of the chapter demonstrates that the information about target labels contained in the posterior can be used even if the measurements do not provide any information about target identities. The computational expense of the algorithm is $O(N_{par}^2)$, although, in principle, it can be sped up using the improved Gaussian transform or k -D trees [110, 136, 207]. Also, an estimator that lowers the MOSPA error compared to other widely used estimators without increasing the computational burden of the algorithm has been proposed.

A topic for future research is the design of a particle filter that allows the estimation of the probability of successful labelling of multitarget state estimates for more than two targets. This would be done by a mirror particle filter as the one developed in this chapter with mirror particles that would include all the possible permutations of the multitarget state. Once the posterior PDF is approximated by the mirror particle filter, the particles can be re-arranged using the unique decomposition of a PDF into variant and invariant parts for multiple targets [41] as done in this chapter although this is not necessary in practice. Also, it should be noted that, in the generalisation for variable and unknown number of targets, the labels must explicitly be included in the state vector as done in Chapter 4.

Chapter 6

Multitarget simultaneous localisation and mapping of a sensor network

6.1 Introduction

As stated in the introduction of this thesis, the positions of the sensors might not be accurately known when the network is deployed in the surveillance area. For instance, in battlefield surveillance, the sensor network might have to be deployed quickly and, therefore, there could be large uncertainty in the positions of the sensors unless they have a built-in GPS. In most cases, this option is not feasible because of its power consumption and the increase in the price of the sensors.

The main objective of this chapter is to develop a centralised algorithm for tracking multiple targets in such a scenario. However, this aim can be achieved more accurately by simultaneously estimating the sensors' positions. Thus, the problem of simultaneously estimating the positions of multiple targets and localising the sensors based on the sensors' measurements is addressed from a Bayesian point of view. This problem has previously been addressed for a single target in [103, 187] and is also related to the simultaneous localisation and mapping (SLAM) problem in robotics in which the robot estimates a map of the environment, i.e., the positions of the so-called landmarks, and its position based on its own measurements [10, 50, 59]. In the following discussion, the acronym R-SLAM is used for SLAM in robotics and SN-SLAM for SLAM in sensor networks, while in the rest of the chapter, SLAM will be used instead of SN-SLAM as context permits.

There have been two main approaches to tackle the R-SLAM problem: Kalman-

filter-type algorithms (KSLAM) [59, 108] and particle-filter-type algorithms (FastSLAM or FSLAM) [137, 138]. KSLAM estimates the state of the robot at the current time and the positions of the landmarks based on previous measurements using a Kalman filter (KF). However, this algorithm has two main problems. The first one is that it is based on Kalman filtering so its result is optimal for linear/Gaussian models but good performance and convergence are not guaranteed for nonlinear/non-Gaussian models, see Chapter 3. Moreover, the posterior correlation among landmarks' positions should be taken into account to build a more accurate map [33] as the current robot state and landmarks' positions conditioned on the past measurements are correlated. The computational expense of doing this is quadratic in the number of landmarks. On the contrary, FSLAM aims to estimate the whole robot trajectory, rather than the current robot state, using a particle filter [169]. It can be shown that, conditioned on the whole robot trajectory, the landmarks' positions are independent. Then, FSLAM uses approximate Rao-Blackwellisation [6] so that the posterior PDF of the map conditioned on a given particle is approximated by a set of independent Gaussians. This results in linear complexity in the number of landmarks, rather than the quadratic complexity that would be necessary if the landmarks' positions were correlated [59].

Information filtering [10] and smoothing [49] have also been applied to R-SLAM. These methods allow sparsification of the information matrix to be exploited in reducing computation. Information filtering estimates the state of the robot at the current time, as does KSLAM, while smoothing estimates the whole robot trajectory to exploit factorisation as in FSLAM with the difference that [49] uses batch processing. However, these methods are based on the linearisation of the process and measurement equations and, then, their performance is expected to be poor in highly nonlinear scenarios. Another approach to dealing with the R-SLAM problem is via finite set statistics in which the map is represented by a random set [147–149]. Within this framework, an approximation to the posterior PDF of the map conditioned on the robot trajectory based on the Gaussian mixture probability hypothesis density (PHD) filter [200] outperforms FSLAM using real data in environments with high clutter [147–149]. The use of this formulation to represent the map is also adopted in [128] although assuming the robot pose is known.

SN-SLAM is equivalent to R-SLAM considering that now the entities that sense (sensors/robots) are static and the ones which are measured (targets/landmarks) are dynamic. If this problem were solved via KSLAM, the KF recursive equations would be similar but using the new dynamic models. Conversely, the direct application of FSLAM to SN-SLAM, i.e., approximating the sensors' positions by particles and the targets' states by Gaussians, is not appropriate as particle filters are not easily

applied to the estimation of static parameters [56]. In addition, the number of sensors in a sensor network is usually high. Then, the positions of the sensors is a vector of large dimension and the particle filter will vastly suffer from the curse of dimensionality [45]. Therefore, it is better to approximate the whole trajectories of the targets by particles, as they are the dynamic part of the model, and conditioned on them, the sensors' positions are approximated by a set of independent Gaussians. Thus, in principle, the algorithms for R-SLAM can be easily adapted to SN-SLAM. It should be noted that the formulations that use information smoothing [49] or finite sets statistics [147–149] can also deal with moving landmarks although their generalisations have not been done yet.

There is an important difference between both problems using particle-filter-type algorithms when there are multiple robots/targets. In both SN-SLAM and R-SLAM, it is assumed that the sensing entities measure the sensed entities independently of other sensing entities. Then, in R-SLAM, a robot's measurement only depends on its (dynamic) state and the landmarks, where the posterior of the state is represented by single-entity particles. Then, a particle filter in R-SLAM could approximate the dynamic states using single-entity particles even in a multiple robot set-up. This is because a robot can build a map without considering other robots although merging individual maps would improve map estimation [64]. In SN-SLAM, a sensor's measurement depends only on its position and the states of all the targets. As a result, targets cannot be considered separately and the posterior of the (dynamic) state must be represented by multiple-entity particles. To summarise, the primary difference between R-SLAM and SN-SLAM is that single-entity particles cannot be used in SN-SLAM as all the targets have to be accounted for to build the map.

The FSLAM method has two shortcomings: one applies to both R-SLAM and SN-SLAM while the other applies only to its generalisation to multitarget SN-SLAM. The first problem is that accurately estimating the whole robot/target trajectory using a particle filter is impractical as one is trying to estimate a quantity of increasing dimension. Therefore, it does not produce consistent estimates in the long-term because of the degeneracy of the particles [11]. Nevertheless, FSLAM has been shown to work well in most real and simulated scenarios without having to propagate a covariance matrix of large dimensions [138]. The second problem arises because FSLAM was originally designed to deal with single-entity particles. As such, its application to multitarget SN-SLAM does not readily permit the use of techniques which have been found to be essential for computationally efficient approximation of the posterior in multitarget situations. One such technique is subparticle crossover [116, 141, 161] as explained in Chapter 4. It takes advantage of the structure of multiple-entity particles but it is not compatible with FSLAM as explained in Sec-

tion 6.3. Subparticle crossover allows the propagation of the multitarget particles, which are made of subparticles, each one representing a target state, by mixing subparticles of different particles.

In this chapter, a new SLAM algorithm for sensor networks is developed based on particle filtering when there are multiple targets. As mentioned in the previous paragraph, a well-known important element of efficient multitarget particle filtering is subparticle crossover [116, 141, 161]. Therefore, an algorithm referred to as multitarget SLAM (MSLAM), which permits its application, is developed. The key feature of this algorithm is that the distribution of the sensor positions is computed conditional on only the most recent multitarget state rather than the trajectories of multitarget states, as in FSLAM. The result of this is a map mixing step in which the maps from the particles are combined. Computationally efficient implementation of this idea requires certain approximations, to be described in Section 6.3.

A further contribution of this chapter arises in the approximation of the map posterior. Traditionally, KF approximations such as the extended Kalman filter (EKF) [59, 137, 138] or the unscented Kalman filter (UKF) [108] have been used. As pointed out in Chapter 3, the KF does not provide a reliable, accurate approximation of the posterior for informative measurements. In this case, the truncated unscented Kalman filter (TUKF) or its single point version (SP-TUKF), which are also described in Chapter 3, can significantly improve on these techniques. It will be shown that MSLAM combined with TUKF or SP-TUKF provides a vast improvement in performance compared to conventional techniques in multitarget situations.

This chapter is organised as follows. The target dynamic and measurement model are explained in Section 6.2. In Section 6.3, the differences between FSLAM and MSLAM are outlined. How to sample the positions of the targets, update the targets' velocities and calculate the particles' weights is addressed in Section 6.4. The update of the sensors' positions based on the TUKF is explained in Section 6.5. Simulation results are provided in Section 6.6. Finally, conclusions are drawn in Section 6.6.

6.2 Problem statement

It is assumed that there are a fixed and known number of targets, t , in the surveillance area. The state of target i is described at time k by the four-dimensional vector $\mathbf{x}_i^k = [x_i^k, \dot{x}_i^k, y_i^k, \dot{y}_i^k]^T$ where $[x_i^k, y_i^k]^T$ is the position vector and $[\dot{x}_i^k, \dot{y}_i^k]^T$ is the velocity vector and the superscript T means transpose. The multitarget state vector is formed by stacking the individual target state vectors into one vector

$\mathbf{X}^k = \left[\left(\mathbf{x}_1^k \right)^T, \left(\mathbf{x}_2^k \right)^T, \dots, \left(\mathbf{x}_t^k \right)^T \right]^T$. It is also assumed that there are M stationary sensors. The position of sensor i is described by the two-dimensional vector $\mathbf{m}_i = [m_{x,i}, m_{y,i}]^T$. The multisensor state vector (also called map) is formed by stacking the individual sensor positions into one vector $\mathbf{m} = [\mathbf{m}_1^T, \mathbf{m}_2^T, \dots, \mathbf{m}_M^T]^T$. Therefore, the aim is the recursive approximation of $p(\mathbf{X}^k, \mathbf{m} | \mathbf{z}^{1:k})$ where $\mathbf{z}^{1:k} = (\mathbf{z}^1, \mathbf{z}^2, \dots, \mathbf{z}^k)$ refers to the collection of measurements up to time k , and $\mathbf{z}^i$ is the set of measurements at time i . To do so, firstly the dynamic model of targets and the measurement model of the sensors are described.

The dynamic model of the targets is the nearly-constant velocity model [13]:

$$p(\mathbf{X}^{k+1} | \mathbf{X}^k) = \prod_{j=1}^t \mathcal{N}(\mathbf{x}_j^{k+1}; \mathbf{F}\mathbf{x}_j^k, \mathbf{Q}) \quad (6.1)$$

$$\mathbf{F} = \mathbf{I}_2 \otimes \begin{pmatrix} 1 & \tau \\ 0 & 1 \end{pmatrix} \quad (6.2)$$

$$\mathbf{Q} = \sigma_u^2 \mathbf{I}_2 \otimes \begin{pmatrix} \tau^3/3 & \tau^2/2 \\ \tau^2/2 & \tau \end{pmatrix} \quad (6.3)$$

where it is assumed that the movements of the targets are independent, $\mathcal{N}(\mathbf{x}; \bar{\mathbf{x}}, \mathbf{Q})$ is the Gaussian PDF evaluated at $\mathbf{x}$ with mean $\bar{\mathbf{x}}$ and covariance matrix $\mathbf{Q}$, $\mathbf{I}_m$ is the $m \times m$ identity matrix, $\otimes$ is the Kronecker product and σ_u^2 is the continuous-time process noise intensity [13].

The vector $\mathbf{z}^k = \left[\left(\mathbf{z}_1^k \right)^T, \left(\mathbf{z}_2^k \right)^T, \dots, \left(\mathbf{z}_M^k \right)^T \right]^T$ contains the measurements from the sensors at time k and $\mathbf{z}_j^k$ is the measurement from sensor j at time k . Measurements are independent in each sensor and do not depend on targets' velocities:

$$p(\mathbf{z}^k | \mathbf{X}^k, \mathbf{m}) = \prod_{r=1}^M p(\mathbf{z}_r^k | \mathbf{P}^k, \mathbf{m}_r) \quad (6.4)$$

where $\mathbf{P}^k$ is the vector with all the targets' positions.

It is also assumed that the sensors are equipped with N_a antennas aiming at different known directions $[\theta_1, \theta_2, \dots, \theta_{N_a}]^T$ which measure the amplitude of the signal coming from those directions independently [208]. Therefore, the measurement from sensor r at time k is represented by a vector $\mathbf{z}_r^k = [z_{r,1}^k, z_{r,2}^k, \dots, z_{r,N_a}^k]^T$ where each component represents the measurement in an antenna. The noise is

assumed to be Gaussian distributed so

$$p(\mathbf{z}_r^k | \mathbf{P}^k, \mathbf{m}_r) = \prod_{l=1}^{N_a} \mathcal{N}(z_{r,l}^k; \bar{z}_{r,l}^k, \sigma_z^2) \quad (6.5)$$

where $\sigma_z^2 = 1$ without loss of generality and the expected value of the amplitude is:

$$\bar{z}_{r,l}^k = \sqrt{\sum_{j=1}^t SNR(d_{j,r,k}) \cdot \exp\left(-\beta (g(\vartheta_{j,r,k}, \theta_l))^2\right)} \quad (6.6)$$

where

$$g(\vartheta, \theta) = \text{mod}(\vartheta - \theta + \pi, 2\pi) - \pi \quad (6.7)$$

$$SNR(d) = \begin{cases} SNR_0 & d \leq d_0 \\ \frac{SNR_0 d_0^2}{d^2} & d > d_0 \end{cases} \quad (6.8)$$

$$d_{j,r,k} = \sqrt{(x_j^k - m_{x,r})^2 + (y_j^k - m_{y,r})^2} \quad (6.9)$$

$$\vartheta_{j,r,k} = \arctan \frac{y_j^k - m_{y,r}}{x_j^k - m_{x,r}} + \begin{cases} 0 & x_j^k - m_{x,r} \geq 0 \\ \pi & y_j^k - m_{y,r} \geq 0, x_j^k - m_{x,r} < 0 \\ -\pi & y_j^k - m_{y,r} < 0, x_j^k - m_{x,r} < 0 \end{cases} \quad (6.10)$$

and $g(\vartheta, \theta)$ is a modular subtraction of angles ϑ and θ that accounts for angle wrapping [153], β is a parameter that controls the radiation pattern of the antenna, SNR_0 is the maximum signal-to-noise ratio produced by a single target in a sensor, and d_0 is the saturation distance. It should be noted that $d_{j,r,k}$ is the distance from target j to sensor r at time k , $\vartheta_{j,r,k}$ is the angle formed by the target j in relation to sensor r and the positive side of the x -axis at time k , $SNR(d_{j,r,k})$ is the maximum signal-to-noise ratio produced by target j in sensor r at time k and the exponential term in (6.6) corresponds to the radiation pattern of the antenna.

It is also assumed that there is some prior knowledge about the targets and sensors. That is, the prior at time 0 for target j is $p(\mathbf{x}_j^0)$ and for sensor r is:

$$p(\mathbf{m}_r) = \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_r^0, \mathbf{\Theta}_r^0) \quad (6.11)$$

where $\bar{\mathbf{m}}_r^0$ and $\mathbf{\Theta}_r^0$ are the prior mean and covariance matrix of sensor r .

6.3 FSLAM versus Multitarget SLAM

This section explains the characteristics of FSLAM, the reasons why it is not appropriate to deal with a multitarget scenario and MSLAM is introduced to address

its deficiencies.

6.3.1 FSLAM

FSLAM is based on approximating the PDF

$$p(\mathbf{X}^{0:k}, \mathbf{m} | \mathbf{z}^{1:k}) = p(\mathbf{P}^{0:k} | \mathbf{z}^{1:k}) p(\mathbf{V}^{0:k}, \mathbf{m} | \mathbf{P}^{0:k}, \mathbf{z}^{1:k}) \quad (6.12)$$

where $\mathbf{X}^{0:k}$ refers to the collection of the multitarget states from time 0 to k , $\mathbf{V}^{0:k}$ represents the targets' velocities from time 0 to k and $\mathbf{P}^{0:k}$ represents the targets' positions from time 0 to k . The posterior PDF of the targets' positions from time 0 to k is approximated using a particle filter [9]. It should be noted that, according to the model, the relationship between the targets' velocities and positions is linear and Gaussian and the measurements do not depend on the velocity. Then, Rao-Blackwellisation can be applied, see Section 2.5.4. Therefore, the velocities are removed from the particle filter and their PDF conditioned on the positions is calculated by a KF [6, 141]. In addition, FSLAM approximates the PDF of the positions of the sensors conditioned on the complete trajectories of the targets and the measurements by set of independent Gaussians, with linear complexity [137, 138]. Thus, the PDF of the multitarget positions from time 0 to k is represented by particles and the conditional PDFs of the positions of the sensors and the velocities are represented using Rao-Blackwellisation via KFs applied to each particle. However, sampling the complete multitarget trajectory in a proper way is impractical because of its increasing dimension and sampling degeneracy. This problem is more pronounced in a multitarget set-up, as it is even more difficult to sample the complete trajectory because the dimension of the state vector is larger and, consequently, particle degeneracy is more acute.

A representation of the propagation of the particles using FSLAM for multiple targets is shown in Figure 6.1 where the last three time instants have been plotted for two particles. As the complete trajectory is being sampled, to form particle i at a time all the targets of the same particle must be propagated simultaneously. This implies that one cannot mix different subparticles (part of the particle that represent one target) to form the new particles, as will be done in the following section. This hinders the process of sampling efficiently because the more targets there are, the more difficult it is to sample adequately as the dimension of the state increases [45]. In general, any sampling procedure that propagates all the targets within a particle at once will be compatible with FSLAM and (6.12) can be approximated. For example, particles are propagated using the prior in [137], while in [138], it is done using an EKF approximation to the optimal importance density. In Figure 6.1, it

Figure 6.1: FSLAM adapted to multiple targets. Each particle is represented by three boxes. Each box represents the part of the particle (subparticle) corresponding to a target. The whole multitarget particle has to be propagated at once as indicated by the arrows. Also, each particle has its own map independent of the maps of the rest of the particles.

is also illustrated that each particle has its own map conditioned on the whole past trajectory of that particle that is independent of the maps of the rest of the particles.

Another problem with FSLAM is that it suffers from map degeneracy. This arises because the map posterior is computed conditional on the trajectory of the target states. Resampling ensures that all particles at the current time have a common ancestor at some previous time. Sensors that have not been approached by any target since this time will have the same posterior means and covariance matrices for all particles, see Figure 6.2. Some techniques proposed to alleviate this problem deal with the resampling step [81, 119] or use parameter estimation rather than Rao-Blackwellisation for the map [133]. Instead, in this chapter, the algorithm MSLAM, which is explained in the next section, is proposed to lower the effect of these two shortcomings.

6.3.2 Multitarget SLAM

The algorithm developed here, MSLAM, is designed to propagate the particles of the targets more efficiently than FSLAM and to mitigate the map degeneracy problem. To this end, the parallel partition method (PP), see Section 4.3.1, is used to propagate the particles. This method was designed to properly deal with multitarget tracking with low computational burden. The use of this method implies that one needs to resort to a map mixing step that reduces the effect of map degeneracy as will be explained hereafter.

Figure 6.2: Map degeneracy in FSLAM: The red line indicates the target trajectory. The target state is represented by three particles at each time step. The arrows indicate who is the parent of a particle at the previous time step. The star denotes a real sensor position and the triangles denote its mean position for a particle. The dotted line indicates the coverage of the sensor. Therefore, the sensor position is only updated at time $k - 3$ remaining unaltered thereafter. At time $k - 3$, each particle has a different mean position for the sensor. However, at time k all the particles have the same mean position for the sensor that is one of the initial means as the three particles have a common ancestor and, thus, map degeneracy appears.

The PP method uses subparticle crossover. That is, it mixes parts of different particles to form new ones in the propagation step. This way the effect of particle degeneracy is lowered and the filter can work with fewer particles in a multitarget tracking set-up. However, carrying out subparticle crossover implies that the complete multitarget trajectories are not being sampled as the best subparticles of each target are taken to represent the new state. This implies that FSLAM, approximating (6.12), cannot be used as the complete multitarget trajectory is not being sampled. The way the PP method propagates the particles was shown in Figure 4.1 in Chapter 4. As different parts of the particles at time $k - 1$ are mixed to form the new particles at time k , it is obvious that the maps of the particles at time $k - 1$ also need to be mixed when the particles at time k are obtained. This map mixing step is one of the novelties of the proposed algorithm compared to FSLAM.

As opposed to FSLAM, MSLAM aims to compute

$$p(\mathbf{X}^k, \mathbf{m} | \mathbf{z}^{1:k}) = p(\mathbf{P}^k | \mathbf{z}^{1:k}) p(\mathbf{V}^k, \mathbf{m} | \mathbf{P}^k, \mathbf{z}^{1:k}) \quad (6.13)$$

It can be seen that the main difference between MSLAM and FSLAM is that in (6.13) one approximates the PDF of the current multitarget state, while in (6.12), the PDF of the complete multitarget trajectories is approximated. As in FSLAM, a particle filter approximation for the target positions is used such that (6.13) becomes:

$$p(\mathbf{X}^k, \mathbf{m} | \mathbf{z}^{1:k}) \approx \sum_{i=1}^{N_{par}} w_i^k \delta(\mathbf{P}^k - \mathbf{P}_i^k) p(\mathbf{V}^k, \mathbf{m} | \mathbf{P}_i^k, \mathbf{z}^{1:k}) \quad (6.14)$$

where N_{par} is the number of particles, $\delta(\cdot)$ is the Dirac delta, $\mathbf{P}_i^k$ is particle i that represents the positions of the targets at time k and w_i^k is its weight. It is assumed that the PDF $p(\mathbf{V}^k, \mathbf{m} | \mathbf{P}_i^k, \mathbf{z}^{1:k})$ is:

$$p(\mathbf{V}^k, \mathbf{m} | \mathbf{P}_i^k, \mathbf{z}^{1:k}) = \prod_{j=1}^t \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \Sigma_{j,i}^k) \cdot \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \Theta_{r,i}^k) \quad (6.15)$$

where $\mathbf{v}_j^k$ is the velocity of target j at time k , $\bar{\mathbf{v}}_{j,i}^k$ and $\Sigma_{j,i}^k$ are the mean and covariance matrix approximation of the velocity of target j for particle i and, $\bar{\mathbf{m}}_{r,i}^k$ and $\Theta_{r,i}^k$ are the mean and covariance matrix approximation of sensor r position for particle i . Note that from (6.15), one can write

$$p(\mathbf{V}^k | \mathbf{P}_i^k, \mathbf{z}^{1:k}) = \prod_{j=1}^t \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \Sigma_{j,i}^k) \quad (6.16)$$

$$p(\mathbf{m} | \mathbf{P}_i^k, \mathbf{z}^{1:k}) = \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \Theta_{r,i}^k) \quad (6.17)$$

The imposition of posterior independence and Gaussianity on the velocities and sensors' positions in (6.15) involves some approximations which will be described in the following sections. Nevertheless, it is going to be demonstrated in the simulations of Section 6.6 that for a multitarget scenario making these approximations has much better results than trying to estimate the complete trajectories of the targets with FSLAM.

There are two main reasons for this. The first one is that MSLAM can be applied with the PP method, which was specifically designed to reduce the number of particles in multiple target tracking. Recall that PP method cannot be applied with FSLAM because FSLAM requires sampling of the complete trajectories of all targets. Then, the approximation of $p(\mathbf{P}^k | \mathbf{z}^{1:k})$ using MSLAM is expected to be better than the approximation of $p(\mathbf{P}^k | \mathbf{z}^{1:k})$ using FSLAM when there are multiple targets. The second one is, as the dimension of $\mathbf{P}^k$ is much lower than the dimension of $\mathbf{P}^{0:k}$, which is a vector of increasing dimension, the particle filter approximation of the PDF $p(\mathbf{P}^k | \mathbf{z}^{1:k})$ will be considerably better than the approximation of $p(\mathbf{P}^{0:k} | \mathbf{z}^{1:k})$ [45]. The difficulty of approximating $p(\mathbf{P}^{0:k} | \mathbf{z}^{1:k})$ gives rise to the map degeneracy problem which was introduced in Section 6.3.1 and illustrated in Figure 6.2. Then, as the posterior PDF of the map in MSLAM depends on the particle approximation of $p(\mathbf{P}^k | \mathbf{z}^{1:k})$ rather than $p(\mathbf{P}^{0:k} | \mathbf{z}^{1:k})$, sample degeneracy is much less likely to adversely affect estimation of the map than in FSLAM. A better approximation of the posterior PDFs of the current position of the targets and the

Figure 6.3: Diagram of an MSLAM update step. Based on the approximation to the posterior PDF at time k and the sensors' measurements at time $k + 1$, an approximation to the posterior PDF at time $k + 1$ is obtained. The arrows show direct relationship between the parts of the algorithm that connect.

map implies that MSLAM should perform better than FSLAM in a multiple target set-up.

Two issues need to be addressed to completely define MSLAM. The first one is how to propagate the particles that represent the positions, which is done using the PP method, and the second is how to update the conditional PDFs of the targets' velocities and sensors' positions. In other words, an approximation of the posterior PDF at time $k + 1$ in a similar form of (6.14) and (6.15) is needed based on the posterior PDF at time k , given by (6.14) and (6.15), and the system described in Section 6.2. These subjects are addressed in the next sections and an illustrative diagram is shown in Figure 6.3.

6.4 Approximation of the target state posterior

In this section, it is explained how to sample the targets' positions, how to update the targets' velocities and how to calculate the weights of the particles.

6.4.1 Sampling the targets' positions

The PP method is used to sample the targets' positions. Recalling from Section 4.3.1, the PP method works with a posterior PDF for the multitarget state that can be factorised as:

$$p(\mathbf{X}^k | \mathbf{z}^{1:k}) \propto \prod_{j=1}^t \sum_{i=1}^{N_{par}} \delta(\mathbf{p}_j^k - \mathbf{p}_{j,i}^k) p(\mathbf{v}_j^k | \mathbf{p}_{j,i}^k, \mathbf{z}^{1:k}) \quad (6.18)$$

where, according to (6.15),

$$p(\mathbf{v}_j^k | \mathbf{p}_{j,i}^k, \mathbf{z}^{1:k}) = \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \boldsymbol{\Sigma}_{j,i}^k) \quad (6.19)$$

and $\mathbf{p}_j^k$ is the position of target j at time k , and $\mathbf{p}_{j,i}^k$ is the position of target j in particle i .

Using (6.1), integrating out the states of the targets at time k and applying Bayes' rule in (6.18), the posterior PDF for the positions at time $k+1$ (the samples need to be drawn from this PDF) is

$$p(\mathbf{P}^{k+1} | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) \prod_{j=1}^t \sum_{i=1}^{N_{par}} \mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,i}^{k+1|k}, \boldsymbol{\Xi}_{11,j,i}^{k+1}) \quad (6.20)$$

where

$$\mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,i}^{k+1|k}, \boldsymbol{\Xi}_{11,j,i}^{k+1}) = \int p(\mathbf{p}_j^{k+1} | \mathbf{p}_{j,i}^k, \mathbf{v}_j^k) \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \boldsymbol{\Sigma}_{j,i}^k) d\mathbf{v}_j^k \quad (6.21)$$

and $\bar{\mathbf{p}}_{j,i}^{k+1|k} = \mathbf{p}_{j,i}^k + \tau \bar{\mathbf{v}}_{j,i}^k$ and $\boldsymbol{\Xi}_{11,j,i}^{k+1} = \tau^2 \boldsymbol{\Sigma}_{j,i}^k + \mathbf{Q}_p$ are the predicted mean and covariance matrix, and $\mathbf{Q}_p$ is the process noise covariance matrix for the position elements according to (6.3), see Appendix A.23 for more details.

The PP method samples an auxiliary variable for each target such that

$$p(\mathbf{P}^{k+1}, \mathbf{a} | \mathbf{z}^{1:k+1}) \propto p(\mathbf{z}^{k+1} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) \prod_{j=1}^t \mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,a_j}^{k+1|k}, \boldsymbol{\Xi}_{11,j,a_j}^{k+1}) \quad (6.22)$$

where $\mathbf{a} = [a_1, \dots, a_t]^T$, with $a_j \in \{1, \dots, N_{par}\}$, indicates that one is choosing the component a_j on the mixture given by the posterior, equation (6.20), for target j .

If $\mathbf{P}_i^{k+1}$, $\mathbf{a}_i = [a_1^i, \dots, a_t^i]^T$ is drawn from an importance density $q(\cdot)$ for $i = 1, \dots, n$, the updated weights are calculated as:

$$w_i^{k+1} \propto \frac{p(\mathbf{z}^{k+1} | \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k}) \prod_{j=1}^t \mathcal{N}(\mathbf{p}_{j,i}^{k+1}; \bar{\mathbf{p}}_{j,a_j}^{k+1|k}, \boldsymbol{\Xi}_{11,j,a_j}^{k+1})}{q(\mathbf{P}_i^{k+1}, \mathbf{a}_i | \mathbf{z}^{1:k+1})} \quad (6.23)$$

The importance density of the PP method is, see (4.28):

$$q(\mathbf{P}^{k+1}, \mathbf{a} | \mathbf{z}^{1:k+1}) = \prod_{j=1}^t b_j(\mathbf{p}_j^{k+1}) \mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,a_j}^{k+1|k}, \boldsymbol{\Xi}_{11,j,a_j}^{k+1}) \quad (6.24)$$

where $b_j(\mathbf{p}_j^{k+1})$ should be selected proportional to:

$$p(\mathbf{z}^{k+1} | \mathbf{p}_j^{k+1}, \hat{\mathbf{P}}_{-\{j\}}^{k+1|k}, \mathbf{z}^{1:k}) = \int p(\mathbf{z}^{k+1} | \mathbf{p}_j^{k+1}, \hat{\mathbf{P}}_{-\{j\}}^{k+1|k}, \mathbf{m}) \cdot p(\mathbf{m} | \mathbf{p}_j^{k+1}, \hat{\mathbf{P}}_{-\{j\}}^{k+1|k}, \mathbf{z}^{1:k}) d\mathbf{m} \quad (6.25)$$

where $\hat{\mathbf{P}}_{-\{j\}}^{k+1|k}$ represents the predicted positions of the targets averaged over all the particles except for target j .

The first problem with (6.25) is that it cannot be calculated exactly. The second problem is that it is computationally demanding to calculate. For every $\mathbf{p}_j^{k+1}$, the approximation to the posterior PDF of the map $p(\mathbf{m} | \mathbf{p}_j^{k+1}, \hat{\mathbf{P}}_{-\{j\}}^{k+1|k}, \mathbf{z}^{1:k})$ changes and an integral based on it needs to be approximated. Then, to lower the computational burden, the following alternative is proposed:

$$b_j(\mathbf{p}_j^{k+1}) \propto \prod_{r=1}^M \int p(\mathbf{z}_r^{k+1} | \mathbf{p}_j^{k+1}, \hat{\mathbf{P}}_{-\{j\}}^{k+1|k}, \mathbf{m}_r) \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_r^k, \boldsymbol{\Theta}_r^k) d\mathbf{m}_r \quad (6.26)$$

where $\{\bar{\mathbf{m}}_1^k, \boldsymbol{\Theta}_1^k, \dots, \bar{\mathbf{m}}_M^k, \boldsymbol{\Theta}_M^k\}$ refers to the set of means and covariance matrices that represent an independent Gaussian approximation to the marginal PDF of the map at time k . This is obtained by integrating out the positions and velocities in (6.14):

$$\bar{\mathbf{m}}_r^k = \frac{1}{N_{par}} \sum_{i=1}^{N_{par}} \bar{\mathbf{m}}_{r,i}^k \quad (6.27)$$

$$\boldsymbol{\Theta}_r^k = \frac{1}{N_{par}} \sum_{i=1}^{N_{par}} \boldsymbol{\Theta}_{r,i}^k + (\bar{\mathbf{m}}_{r,i}^k - \bar{\mathbf{m}}_r^k) (\bar{\mathbf{m}}_{r,i}^k - \bar{\mathbf{m}}_r^k)^T \quad (6.28)$$

where it has been used that $w_i^k = 1/N_{par}$ as happens after a resampling stage.

Finally, the integral in (6.26) is approximated using the unscented transformation [100] as shown in Appendix A.24. The steps of the approximate sampling procedure of the PP method for MSLAM are shown in Table 6.1 and a thorough explanation

Table 6.1: Parallel partition subroutine for multitarget SLAM

PP Subroutine for target j

- For each particle $i = 1, \dots, N_{par}$:
 - Get a sample for the position $\mathbf{p}_{j,i}^* \sim N(\mathbf{p}_j^{k+1}, \bar{\mathbf{p}}_{j,i}^{k+1|k}, \Xi_{11,j,i}^{k+1})$.
 - Calculate $b_j(\mathbf{p}_{j,i}^*)$ given by (6.26) according to Appendix A.24.
- Normalise $b_j(\mathbf{p}_{j,i}^*)$ to sum to one over $i = 1, \dots, N_{par}$.
- Resampling step for each track. For each particle $i = 1, \dots, N_{par}$:
 - Sample an index p from the distribution defined by $b_j(\mathbf{p}_{j,l}^*)$ over $l = 1, \dots, N_{par}$.
 - Set $\mathbf{p}_{j,i}^{k+1} = \mathbf{p}_{j,p}^*$ and the parent's index $a_j^i = p$.

of this procedure is given in Section 4.3.1.

6.4.2 Update of the targets' velocities

Once the particles of the positions at time $k + 1$ have been drawn, this section addresses the update of the velocities of the targets, i.e, how to calculate (6.16). It is shown in Appendix A.25 that:

$$p(\mathbf{V}^{k+1} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) \propto \prod_{j=1}^t \sum_{i=1}^{N_{par}} \mathcal{N}(\mathbf{u}_j^{k+1}; \bar{\mathbf{u}}_{j,i}^{k+1}, \Xi_{j,i}^{k+1}) \quad (6.29)$$

where $\mathbf{u}_j^{k+1} = [\mathbf{p}_j^{k+1}, \mathbf{v}_j^{k+1}]^T$ and

$$\bar{\mathbf{u}}_{j,i}^{k+1} = [\mathbf{p}_{j,i}^k + \tau \bar{\mathbf{v}}_{j,i}^k, \bar{\mathbf{v}}_{j,i}^k]^T \quad (6.30)$$

$$\Xi_{j,i}^{k+1} = \begin{pmatrix} \Xi_{11,j,i}^{k+1} & \Xi_{12,j,i}^{k+1} \\ \Xi_{21,j,i}^{k+1} & \Xi_{22,j,i}^{k+1} \end{pmatrix} = \begin{pmatrix} \tau^2 \Sigma_{j,i}^k + \mathbf{Q}_p & \tau \Sigma_{j,i}^k + \mathbf{Q}_{pv} \\ \tau \Sigma_{j,i}^k + \mathbf{Q}_{vp} & \Sigma_{j,i}^k + \mathbf{Q}_v \end{pmatrix} \quad (6.31)$$

where $\mathbf{Q}_p$, $\mathbf{Q}_v$ are the covariance matrices of the positions and velocities, and $\mathbf{Q}_{pv}$ is the covariance matrix between the positions and velocities according to (6.3).

Equation (6.29) can be written as

$$p(\mathbf{V}^{k+1} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) = \prod_{j=1}^t \sum_{i=1}^{N_{par}} \rho_{1,i}(\mathbf{p}_j^{k+1}) \mathcal{N}(\mathbf{v}_j^{k+1}; \bar{\mathbf{v}}_{j,i}^{k+1|k}(\mathbf{p}_j^{k+1}), \Sigma_{j,i}^{k+1|k}) \quad (6.32)$$

where

$$\tilde{\rho}_{1,i}(\mathbf{p}_j^{k+1}) = \mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,i}^{k+1|k}, \Xi_{11,j,i}^{k+1}) \quad (6.33)$$

and $\rho_{1,i}(\mathbf{p}_j^{k+1})$ are the normalised $\tilde{\rho}_{1,i}(\mathbf{p}_j^{k+1})$ so that they sum to one in the variable i and, $\bar{\mathbf{v}}_{j,i}^{k+1|k}(\mathbf{p}_j^{k+1})$ and $\Sigma_{j,i}^{k+1|k}$ are calculated using the fundamental equations of linear estimation [13]:

$$\bar{\mathbf{v}}_{j,i}^{k+1|k}(\mathbf{p}_j^{k+1}) = \bar{\mathbf{v}}_{j,i}^k + \Xi_{12,j,i}^{k+1} (\Xi_{11,j,i}^{k+1})^{-1} (\mathbf{p}_j^{k+1} - \bar{\mathbf{p}}_{j,i}^{k+1|k}) \quad (6.34)$$

$$\Sigma_{j,i}^{k+1|k} = \Xi_{22,j,i}^{k+1} - \Xi_{12,j,i}^{k+1} (\Xi_{11,j,i}^{k+1})^{-1} \Xi_{21,j,i}^{k+1} \quad (6.35)$$

Then, according to (6.32), the posterior of the targets' velocities is a product of Gaussian mixtures. As for the next step of the recursion, see (6.18) and (6.19), it is assumed that the velocity of a target is approximated by a Gaussian PDF, each Gaussian mixture in (6.32) is approximated by a Gaussian PDF, such that (6.16) becomes [169]:

$$p(\mathbf{V}^{k+1} | \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k}) \approx \prod_{j=1}^t \mathcal{N}(\mathbf{v}_j^{k+1}; \bar{\mathbf{v}}_{j,i}^{k+1}, \Sigma_{j,i}^{k+1}) \quad (6.36)$$

where

$$\bar{\mathbf{v}}_{j,i}^{k+1} = \sum_{p=1}^{N_{par}} \rho_{1,p}(\mathbf{p}_{j,i}^{k+1}) \bar{\mathbf{v}}_{j,p}^{k+1|k}(\mathbf{p}_{j,i}^{k+1}) \quad (6.37)$$

$$\Sigma_{j,i}^{k+1} = \sum_{p=1}^{N_{par}} \rho_{1,p}(\mathbf{p}_{j,i}^{k+1}) \cdot \left[\Sigma_{j,p}^{k+1|k} + (\bar{\mathbf{v}}_{j,p}^{k+1|k}(\mathbf{p}_{j,i}^{k+1}) - \bar{\mathbf{v}}_{j,i}^{k+1}) (\bar{\mathbf{v}}_{j,p}^{k+1|k}(\mathbf{p}_{j,i}^{k+1}) - \bar{\mathbf{v}}_{j,i}^{k+1})^T \right] \quad (6.38)$$

It should be mentioned that approximating a Gaussian mixture by independent Gaussian PDFs has traditionally been done in the tracking community, for instance, in the well-known JPDAF [63].

6.4.3 Calculation of the particles' weights

In Section 6.4.1, it was explained how to propagate the particles using the PP method. Here, it is shown how to calculate their weights. Substituting (6.24) into (6.23), the particle weights are:

$$w_i^{k+1} \propto \frac{p(\mathbf{z}^{k+1} | \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k})}{\prod_{j=1}^t b_j(\mathbf{p}_{j,i}^{k+1})} \quad (6.39)$$

In order to calculate (6.39), one needs to obtain

$$p(\mathbf{z}^{k+1} | \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k}) = \int p(\mathbf{z}^{k+1} | \mathbf{P}_i^{k+1}, \mathbf{m}) \cdot p(\mathbf{m} | \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k}) d\mathbf{m} \quad (6.40)$$

The PDF $p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k})$ is calculated in Appendix A.26:

$$p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) = \sum_{i=1}^{N_{par}} \rho_{2,i}(\mathbf{P}^{k+1}) \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \boldsymbol{\Theta}_{r,i}^k) \quad (6.41)$$

where

$$\tilde{\rho}_{2,i}(\mathbf{P}^{k+1}) = \prod_{j=1}^t \mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,i}^{k+1|k}, \boldsymbol{\Xi}_{11,j,i}^{k+1}) \quad (6.42)$$

and $\rho_{2,i}(\mathbf{P}^{k+1})$ are the normalised $\tilde{\rho}_{2,i}(\mathbf{P}^{k+1})$ so that they sum to one in the variable i . Then, (6.41) indicates that the distribution of the map given the new positions of the targets at time $k+1$ and without taking into account the measurements at time $k+1$ is a mixture of independent Gaussian distributions that represent the maps of the particles at time k . This mixture of maps has one positive effect and one drawback. The positive effect is that it tends to reduce map degeneracy because the maps are shared among the particles in the following way. The map for a certain position at time $k+1$ is a mixture of Gaussians determined by the values of $\rho_{2,i}(\mathbf{P}^{k+1})$. The value $\rho_{2,i}(\mathbf{P}^{k+1})$ indicates the probability that the position $\mathbf{P}^{k+1}$ at time $k+1$ has been drawn from particle i at time k . This way if a particle at time $k+1$ matches only one particle at time k , it will inherit its map. On the contrary, if there are several particles at time k that match the new particle, the new particle will inherit a map that is a combination of those previous maps. The drawback is that the new map is not Gaussian distributed any more and that the sensors' positions become correlated. Therefore, in order to be able to estimate recursively the positions of the sensors using (6.15), the PDF in (6.41) is approximated by a product of independent Gaussian PDFs whose mean and covariance matrix are equal to the Gaussian mixture in (6.41). Using the mean and covariance matrix of a mixture of Gaussians [169]:

$$p(\mathbf{m} | \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k}) \approx \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^{k+1|k}, \boldsymbol{\Theta}_{r,i}^{k+1|k}) \quad (6.43)$$

where

$$\bar{\mathbf{m}}_{r,i}^{k+1|k} = \sum_{p=1}^{N_{par}} \rho_{2,p}(\mathbf{P}_i^{k+1}) \bar{\mathbf{m}}_{r,p}^k \quad (6.44)$$

$$\boldsymbol{\Theta}_{r,i}^{k+1|k} = \sum_{p=1}^{N_{par}} \rho_{2,p} \left(\mathbf{P}_i^{k+1} \right) \cdot \left[\boldsymbol{\Theta}_{r,p}^k + \left(\bar{\mathbf{m}}_{r,p}^k - \bar{\mathbf{m}}_{r,i}^{k+1|k} \right) \left(\bar{\mathbf{m}}_{r,p}^k - \bar{\mathbf{m}}_{r,i}^{k+1|k} \right)^T \right] \quad (6.45)$$

Finally, (6.43) is substituted into (6.40) and the integral in (6.40) is approximated by applying the unscented transformation to each sensor position [100] to obtain an approximation to (6.39).

6.5 Sensors' positions update

So far it has been explained how to sample the targets' positions, update the velocities and calculate the weights of the particles. This section completes the MSLAM recursion by updating the sensors' positions using the new measurements. This is done calculating (6.17) for the current time step. Applying Bayes' rule for the map of particle i to (6.43) and (6.4):

$$p \left(\mathbf{m} \mid \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k+1} \right) \propto \prod_{r=1}^M p \left(\mathbf{z}_r^{k+1} \mid \mathbf{P}_i^{k+1}, \mathbf{m}_r \right) p \left(\mathbf{m}_r \mid \mathbf{P}_i^{k+1}, \mathbf{z}^{1:k} \right) \quad (6.46)$$

In this section, the indexes that indicate the particle and time are dropped for the sake of clarity but noting that Bayes' rule for the sensors' positions needs to be performed for each particle and sensor at each time. Hence, the algorithm described in this section has to be carried out particle by particle and sensor by sensor. Usually the Bayesian recursion for the map is approximated using the EKF or UKF [59, 108]. In this section, the SP-TUKF and TUKF proposed in Chapter 3 are applied to this problem. To this end, it should be recalled that one needs to approximate the first two moments of the truncated prior $p_1(\cdot)$ and determine the selection of α . These issues are addressed in the following.

6.5.1 Approximation of the first two moments of the truncated prior

As shown in Appendix A.28, the first two moments of the truncated prior can be approximated as:

$$\hat{\mathbf{m}}_{p,1} = \tilde{\mathbf{m}}(\mathbf{z}) \quad (6.47)$$

$$\mathbf{P}_{p,1} = \left(\tilde{\mathbf{H}}^T \tilde{\mathbf{H}} \right)^{-1} \quad (6.48)$$

where $\tilde{\mathbf{m}}(\mathbf{z})$ is the value that maximises the likelihood and

$$\tilde{\mathbf{H}} = \left[\nabla_{\mathbf{m}_r} \mathbf{h}^T(\mathbf{m}_r) \right]^T \Big|_{\mathbf{m}_r = \hat{\mathbf{m}}_{p,1}} \quad (6.49)$$

is the Jacobian of $\mathbf{h}(\cdot)$ evaluated at $\hat{\mathbf{m}}_{p,1}$. The function $\mathbf{h}(\cdot)$ is given by (6.6) particularised for the current multitarget state and sensor. Note that the notation of Chapter 3 has been used to represent these moments and the sensor index r has been dropped. The exact form of $\tilde{\mathbf{H}}$ is given in Appendix A.27.

Therefore, the main difficulty to find the first two moments of the truncated prior is to calculate its mean as the covariance matrix can be easily obtained once the mean is available using (6.49). The mean of the truncated prior should minimise $(\mathbf{z}_r - \mathbf{h}(\mathbf{m}_r))^T \mathbf{R}^{-1} (\mathbf{z}_r - \mathbf{h}(\mathbf{m}_r))$ so that the likelihood attains its maximum where $\mathbf{R} = \mathbf{I}_{N_a}$ is the measurement noise covariance matrix for sensor r and $\mathbf{z}_r$ is the measurement of sensor r . Then, nonlinear least squares can be used to find it as in this case the measurement function is not bijective. Note that if it were, one could use the inverse of the measurement function as in Chapter 3.

However, as there is also some prior knowledge about the location of the sensor, equation (6.11), a penalty function is used to increase the chance that this mean is in a region where the prior is non-negligible and improve the convergence of the algorithm. Let us define:

$$\mathbf{L}(\mathbf{m}_r) = \left[(\mathbf{z}_r - \mathbf{h}(\mathbf{m}_r))^T, g(\mathbf{m}_r) \right]^T \quad (6.50)$$

where

$$g(\mathbf{m}_r) = \begin{cases} 0 & a(\mathbf{m}_r) < \Gamma_p \\ c_p(a(\mathbf{m}_r) - \Gamma_p) & \text{otherwise} \end{cases} \quad (6.51)$$

$$a(\mathbf{m}_r) = (\mathbf{m}_r - \bar{\mathbf{m}}_r^0)^T (\boldsymbol{\Theta}_r^0)^{-1} (\mathbf{m}_r - \bar{\mathbf{m}}_r^0) \quad (6.52)$$

and $\bar{\mathbf{m}}_r^0$ and $\boldsymbol{\Theta}_r^0$ are the initial mean and covariance matrix of the position of sensor r , $c_p > 0$ is a parameter of the penalty function and Γ_p sets the region where the penalty function applies. Then, the aim is to minimise the objective function:

$$\begin{aligned} J(\mathbf{m}_r) &= \mathbf{L}^T(\mathbf{m}_r) \mathbf{L}(\mathbf{m}_r) \\ &= (\mathbf{z}_r - \mathbf{h}(\mathbf{m}_r))^T (\mathbf{z}_r - \mathbf{h}(\mathbf{m}_r)) + g^2(\mathbf{m}_r) \end{aligned} \quad (6.53)$$

It is also assumed that $N_a > 2$ (there are more than two antennas) so that the problem is overdetermined (the sensor's position has dimension 2) and the Gauss-Newton method is used to solve it [107]. It should be noted that minimising (6.53) for every particle is time-consuming. Then, a two-phase procedure is proposed so that (6.53) is minimised globally, i.e., for the mean positions of the targets and then,

this global minimum is refined for each particle. The motivation is that once (6.53) has been minimised globally, the refinement needed for each one of the particles will take less steps to converge and, therefore, it is expected to lower the computational burden.

6.5.1.1 Global minimisation

In this step of the algorithm, the objective function, equation (6.53), is minimised for the mean positions of the targets (considering all the particles), i.e., these positions are used to calculate function $\mathbf{h}(\cdot)$. Then, the Gauss-Newton method uses the following iteration to obtain the sensor position $\mathbf{m}_r$ that minimises the objective function, equation (6.53) [107]:

$$\tilde{\mathbf{m}}_{r,k+1} = \tilde{\mathbf{m}}_{r,k} - \gamma_g \left(\mathbf{L}'_k^T \mathbf{L}'_k \right)^{-1} \mathbf{L}'_k^T \mathbf{L}(\tilde{\mathbf{m}}_{r,k}) \quad (6.54)$$

where $\tilde{\mathbf{m}}_{r,k}$ is the position of sensor r at iteration k , $\mathbf{L}'_k$ is the Jacobian of $\mathbf{L}(\mathbf{m}_r)$ evaluated at $\tilde{\mathbf{m}}_{r,k}$ and γ_g is a parameter that controls the convergence rate. The starting point of the iteration is the mean of the sensor at the current time step, given by (6.44).

Finally, the convergence criterion of the recursion (6.54) is

$$\left| \frac{J(\tilde{\mathbf{m}}_{r,k+1}) - J(\tilde{\mathbf{m}}_{r,k})}{J(\tilde{\mathbf{m}}_{r,k})} \right| < 10^{-5} \quad (6.55)$$

or if the number of steps of the recursion reaches N_{rg} , which is another parameter of the algorithm.

6.5.1.2 Local minimisation

The output of the global minimisation is used as the input of the local minimisation. In this step, a refinement of the position of the sensor for each particle is calculated. Thus, for each particle iteration (6.54) is carried out again bearing in mind that function $\mathbf{h}(\cdot)$ is different for each particle as it depends on the targets' positions, which are different for each particle. The convergence criterion of the recursion is again (6.55) or if the number of steps is equal to N_{rl} , which is another parameter of the algorithm.

The output of this algorithm is the approximation to the mean of the truncated prior $\hat{\mathbf{m}}_{p,1}$. The covariance matrix $\mathbf{P}_{p,1}$ is obtained using (6.48).

Table 6.2: Subroutine for updating a sensor position

- Sum the squared signal for all the antennas in that sensor $E_s = \sum_{i=1}^{N_a} z_{r,i}^2$ where $z_{r,i}$ is the i th component of $\mathbf{z}_r$.
 - IF $E_s > \Gamma_E$
 - * Calculate the global approximation to the mean of the truncated prior $\hat{\mathbf{m}}_{p,1}$ according to the procedure in Section 6.5.1.1.
 - * For each particle
 - Calculate the local approximation to the mean of the truncated prior $\hat{\mathbf{m}}_{p,1}$ according to the procedure in Section 6.5.1.2.
 - Calculate the covariance matrix of the truncated prior $\mathbf{P}_{p,1}$ using (6.48).
 - Calculate α using (6.56).
 - Apply the TUKF or SP-TUKF to update this sensor position.
 - ELSE
 - * For each particle
 - Set $\alpha = 0$, i.e., apply the UKF to update this sensor position.
 - END

6.5.2 Selection of α

The selection of α and α_{sp} is done in the same way as in Section 3.4.3.3:

$$\alpha = \frac{\gamma \text{tr}(\mathbf{P}_{p,0})}{\gamma \text{tr}(\mathbf{P}_{p,0}) + (1 - \gamma) \text{tr}(\mathbf{P}_{p,1})} \quad (6.56)$$

$$\alpha_{sp} = \alpha_{max} \alpha \quad (6.57)$$

where $\mathbf{P}_{p,0}$ is the covariance matrix of the prior for the particle under consideration, given by (6.45), $\mathbf{P}_{p,1}$ is the covariance matrix of the approximated truncated prior, given by (6.48), $\gamma \in [0, 1]$ is a parameter that controls the weights of the traces of the covariance matrices to select α_{sp} , $\alpha_{max} < 1$ is the maximum value α_{sp} can take, and $\text{tr}(\mathbf{A})$ denotes the trace of matrix $\mathbf{A}$.

Additionally, a threshold Γ_E is used. If the sum of the squared measurements in the antennas is under this threshold, $\alpha = 0$. The reason for using this threshold, is that if the energy of the received signal is not high enough, the value of α will be low. Then, the TUKF and SP-TUKF tend to be the UKF and one can avoid calculating the moments of the truncated prior and α and, therefore, reduce the computational burden of the algorithm. The steps of the subroutine for updating the sensors' position using the TUKF or SP-TUKF are shown in Table 6.2.

Table 6.3: Proposed algorithm for multitarget SLAM

- Calculate the sensors to update, using Γ_{sens} .
- For each target, sample its position using PP subroutine, see Table 6.1.
- Map and velocity mixing step: mix the maps of the sensors via (6.43) and the velocities via (6.36).
- Update the weights of each particle using (6.39).
- Update the positions of the sensors following the steps given in Table 6.2.
- Perform resampling to obtain an evenly weighted particle set.

6.5.3 Implementation of the algorithm

In the implementation of the algorithm, all the sensors' positions are not always updated and all the sensors' measurements are not used to propagate the particles at every time step because it is very time-consuming. The set of sensors that is used to do all the calculations is calculated based on a threshold Γ_{sens} . If the distance from any target estimate to a sensor is less than Γ_{sens} then the sensor is considered in the current step of the algorithm otherwise, it is not taken into account. To sum up, the steps of the complete algorithm for multitarget SLAM are shown in Table 6.3.

In addition, if the surveillance area were large enough one could consider applying target clustering as in Section 4.3.1. It has not been introduced here for the clarity of presentation and because in the scenario analysed in the following section, the targets are close to each other and the likelihood does not factorise over the clusters.

6.6 Simulation results

In this section, MSLAM is compared with FSLAM and with a filter that uses the PP method for tracking but without updating the sensors' positions. The last method is a suboptimal approach that takes into account the initial uncertainty in the positions of the sensors but does not update them. This method is analysed to show the improvement that can be made by estimating simultaneously the positions of the targets and the sensors. Bayes' rule for updating the conditional posterior PDF of the sensors' positions for both SLAM methods is approximated using the UKF, the SP-TUKF and the TUKF.

The FSLAM algorithm for multitarget SLAM cannot perform subparticle crosso-

Table 6.4: Model parameters

Parameter	σ_u	SNR_0	d_0	β	N_a	σ_0
Value	1.8 m/s ^{3/2}	32 dB	1 m	1	4	0.1 m

Table 6.5: Algorithm parameters

Parameter	Γ_{sens}	Γ_E	Γ_p	c_p	N_{rg}	N_{rl}	γ_g	α_{max}	γ	N_{par}
Value	25 m	40	9	100	100	100	0.1	0.8	0.1	300

ver as stated previously, so an auxiliary particle filter is used [165] to propagate the particles jointly for all the targets. In the implementation of the auxiliary particle filter, the likelihood of the predicted mean of the multitarget state is used as the first-stage weights.

Three targets are used to evaluate the performances of the algorithms with the trajectories shown in Figure 6.4. The sampling period of the trajectories is $\tau = 0.5$ s and there are $l = 99$ time steps in the simulation. The surveillance area is a square whose side is 150 m. The algorithms will be tested by varying the initial uncertainty of the sensors' positions for a multiple target and a single target set-up. The initial mean positions of the sensors form a grid whose side is 10 m. The initial covariance matrix of each sensor position is

$$\Theta_j^0 = \sigma_s^2 \mathbf{I}_2 \quad (6.58)$$

The prior PDF for the target's position is:

$$\mathbf{p}_i^0 \sim \mathcal{N}(\mathbf{p}_i^0; \bar{\mathbf{p}}_i^0, \sigma_0^2 \mathbf{I}_2) \quad (6.59)$$

for $i \in \{1, \dots, t\}$ where $\bar{\mathbf{p}}_i^0$ is the expected position of target i at time 0 and $\sigma_0^2 \mathbf{I}_2$ is the covariance matrix of the initial position at time 0 which is assumed to be the same for all the targets for simplicity. In each Monte Carlo run, $\bar{\mathbf{p}}_i^0$ is drawn from a Gaussian distribution whose mean is the real position of target i at time 0 and whose covariance matrix is $\sigma_0^2 \mathbf{I}_2$. In addition, the initial target velocity is taken as in Chapter 4. For each target in each particle, two integers η_1, η_2 are drawn such that $P(\eta_i = j) = 1/2$ for $j \in \{-1, 1\}$ $i \in \{1, 2\}$ and then the initial distribution of the target velocity is Gaussian with mean $[\eta_1, \eta_2]^T$ and covariance matrix $\mathbf{I}_2$. The parameters of the model used in the simulations are those shown in Table 6.4. The sensor's antennas are aiming at directions $(0^\circ, 90^\circ, 180^\circ, 270^\circ)$. The parameters of the algorithms used in the simulations are those shown in Table 6.5.

The tracking performance is evaluated by calculating the root mean square

Figure 6.4: Scenario of the simulations: Each target is identified at the beginning of its trajectory by a number. The trajectories have an asterisk to indicate the target position every ten time steps.

(RMS) of the position error for the targets and the sensors by a Monte Carlo simulation with $m = 50$ realisations. To assess the tracking performance, a *Track-to-truth Assignment Table* [15] is made so that each track is associated with one target or with none at each time step depending on the distance from the track to the real position of the target and a threshold that is taken to be $\Lambda = 10$ m. If the distance from a track to a target exceeds that threshold, the track cannot be associated with that target.

Let $\hat{\mathbf{p}}_i^k(j)$, $i = 1, \dots, t$, $k = 1, \dots, l$, $j = 1, \dots, m$ denote the position estimate of the i th target on the j th Monte Carlo realisation at time k , $e_i^k(j) = \|\mathbf{p}_i^k - \hat{\mathbf{p}}_i^k(j)\|_2$ denote the position error for the i th target on the j th Monte Carlo realisation at time k and $\chi_{i,j,k} = 1$ if $e_i^k(j) < \Lambda$ and zero otherwise. The variable $\chi_{i,j,k}$ indicates if the i th target is in track at time k on the j th Monte Carlo realisation. The RMS position error E for the targets is then calculated by

$$E = \sqrt{\frac{\sum_{j=1}^m \sum_{k=1}^l \sum_{i=1}^t (e_i^k(j))^2 \cdot \chi_{i,j,k}}{\sum_{j=1}^m \sum_{k=1}^l \sum_{i=1}^t \chi_{i,j,k}}} \quad (6.60)$$

The RMS position error for the targets and sensors are shown in Figure 6.5. To be fair with the comparisons, the results for FSLAM with 600 particles are also included as it has roughly the same computational burden in our implementation as MSLAM with 300 particles. The RMS position error for the sensors for the PP method corresponds to the average initial error as the sensors' positions are not updated, see Figure 6.5(b). It has been obtained by simulation although its theoretical value is $\sqrt{2}\sigma_s$. Therefore, the distance of this line to the line of another algorithm indicates

Figure 6.5: Performance of the algorithms for three targets: (a) RMS position error for the targets, (b) RMS position error for the sensors. In both cases, MSLAM-TUKF outperforms the rest of the algorithms.

the improvement or deterioration of the knowledge of the sensors' positions given by the algorithm. One should note that it may seem that the localisation of the sensors for these algorithms is poor by looking at Figure 6.5(b) but this error is averaged over all the sensors and not all the sensors are updated because the targets do not approach them.

There are three characteristics of the results in Figure 6.5 that are of interest: MSLAM versus FSLAM, the UKF versus the SP-TUKF and TUKF and, SLAM versus PP method with only tracking. Firstly, it can be seen that FSLAM performance is rather poor compared to MSLAM. This is due to the advantage of using subparticle crossover and a lower map degeneracy of MSLAM. Even if the number of particles of FSLAM is doubled, it is still far from the performance of MSLAM.

Secondly, the improvement of the estimate given by the TUKF and SP-TUKF over the UKF is more significant as σ_s increases. This happens because, as σ_s increases, the prior knowledge about the sensors' positions is increasingly vague and, then, the measurement carries more information compared to the prior. In this case, the TUKF and SP-TUKF clearly outperform the UKF. When the prior knowledge is very accurate compared with the information of the measurement, α_{sp} tends to zero, equation (6.57), and the TUKF and the SP-TUKF come down to the UKF. Therefore, for low σ_s , both methods have the same performance. Also, the TUKF performance is slightly better than the SP-TUKF.

Thirdly, if SLAM is not carried out properly, i.e., FSLAM is used in a multiple target set-up, it is better to use the PP method without updating the sensors' positions for $\sigma_s \leq 2$ m. In this case for FSLAM, simultaneously estimating the

map and the targets' positions is worse than just doing tracking. For MSLAM, simultaneously estimating the map and the targets' positions is never worse than just doing tracking as expected. As σ_s grows, both SLAM methods provide an advantage over only tracking. However, when the uncertainty in the sensors' positions is very low, $\sigma_s = 0.1$ m, MSLAM and PP performances are approximately the same. In that case, it might not be worth updating the sensors' positions as the improvement is negligible.

In the following, the estimate of the map for $\sigma_s = 4$ m is analysed more deeply. Figure 6.6 shows the final RMS position error map for the sensors for the algorithms. Every pixel of the figure represents the RMS for the sensor whose prior expected position at time 0 is located in that part of the surveillance area. The axes are labelled according to the initial mean positions of the sensors. It is suggested having a look at the trajectories of the targets in Figure 6.4 to understand Figure 6.6 better. It is clear that the estimates of the positions of the sensors that are located far from the trajectories of the targets are not improved as they acquire no useful information regarding their positions. The main difference between using the UKF and the TUKF (or the SP-TUKF) lies in the estimate of the sensors' positions near the initial positions of the targets. The UKF fails to provide a good estimate of the positions of such sensors. This affects the tracking and localisation at future times. This occurs because, initially, the information given by the measurements could be much higher than the information of the prior and, therefore, the UKF does not work properly. The main difference between FSLAM-TUKF and MSLAM-TUKF are the estimates of the sensors' positions that are updated after a certain time has passed. This is because FSLAM does not work well in a multiple target set-up and cannot track the targets properly. As tracking performance is poor, map estimation in FSLAM-TUKF is poor too. Then, it is clear the algorithm that was previously available in the literature, FSLAM-UKF, is vastly outperformed by the algorithm proposed here, MSLAM-TUKF because of two reasons: the use of an efficient particle filter for multitarget tracking and a more accurate approximation of the posterior of the map given by the TUKF.

The RMS position error of the targets per time step for $\sigma_s = 4$ m is shown in Figure 6.7. MSLAM-TUKF provides the lowest errors at practically all time steps. Moreover, the RMS position errors are distributed roughly evenly along all time steps, i.e., the error does not increase unboundedly with time.

The effect of varying the number of particles for MSLAM-TUKF and MSLAM-SP-TUKF on the RMS position error is shown in Figure 6.8. When the number of particles increases, the error decreases but just slightly. This indicates that 300 particles is a good choice as doubling the number of particles has almost no effect

Figure 6.6: Final RMS position error map for the sensors for $\sigma_s = 4$ m. MSLAM combined with the TUKF provides the best estimate of the map. Each pixel represents the RMS position error for a sensor.

Figure 6.7: RMS position error of the targets per time step for $\sigma_s = 4$ m. MSLAM combined with the TUKF provides the lowest errors at practically all time steps.

on estimation performance.

The RMS position error for the targets when only target number two is present in the scenario of Figure 6.4 is shown in Figure 6.9. In this case, the best algorithm is FSLAM-TUKF. MSLAM, which was designed to deal with a multitarget set-up, is worse than FSLAM for one target. The reason for this is that one cannot take advantage of subparticle crossover as there is only one target and, therefore, it cannot be performed. It should be recalled that the good properties of our method that are subparticle crossover and a map mixing step occur at the expense of approximating Gaussian mixtures by independent Gaussian, equation (6.43). However, for a one-target scenario it is better to estimate the whole target trajectory and potentially incur map degeneracy than to estimate the current state of the target without map degeneracy and make the approximations in Section 6.3.

It is interesting to compare the results of Figure 6.9 for one target with the results of Figure 6.5 for three targets. This provides an insight into how the PP method and MSLAM deal with the problem of dimensionality. While FSLAM severely suffers from the curse of dimensionality, when there are three targets the performance slumps, PP and MSLAM have roughly the same performance regardless of the number of targets because of subparticle crossover. Moreover, when $\sigma_s \geq 4$, the performance of MSLAM for three targets is better than for one target. This is because MSLAM is able to localise more sensors improving tracking performance. Then, this suggests that MSLAM-TUKF is an appropriate way to address SLAM in a scenario with multiple targets.

Finally, the execution times for the algorithms implemented in Matlab with C-

Figure 6.8: RMS position error of the targets for MSLAM with different number of particles. There is a slight improvement when the number of particles increases which is more important for high σ_s .

Figure 6.9: RMS position error for the target when only one target is present (target two in Figure 6.4). FSLAM-TUKF provides a slightly better estimate than MSLAM-TUKF.

Table 6.6: Execution times in seconds for $\sigma_s = 4$ m.

Algorithm (N_{par})	One target	Three targets
MSLAM-UKF (300)	38	153
MSLAM-TUKF (300)	40	189
MSLAM-SP-TUKF (300)	39	184
FSLAM-UKF (600)	43	160
FSLAM-TUKF (600)	44	240
FSLAM-SP-TUKF (600)	44	213
FSLAM-UKF (300)	21	79
FSLAM-TUKF (300)	22	106
FSLAM-SP-TUKF (300)	22	106
PP (300)	14	99

MEX subroutines on a Pentium IV with a 2.4-GHz processor for this scenario using 300 particles are shown in Table 6.6.

6.7 Conclusions

A new algorithm for multitarget simultaneous localisation and mapping of a sensor network has been developed. It vastly outperforms the well-known FSLAM in multitarget scenarios. There are two reasons for this improvement. Firstly, an efficient particle filter for multiple target tracking that allows subparticle crossover is used. Secondly, the TUKF provides a more accurate approximation of the posterior of the map than the UKF.

It is shown that a big improvement in the tracking of multiple targets can be achieved by simultaneously estimating the multitarget state and the sensors' positions compared to only estimating the multitarget state. This was expected as the measurements are used to update the PDF of all the random variables of the model rather than just the multitarget state.

The lines of future research are manifold. Firstly, calculating the posterior Cramér-Rao lower bound would be interesting to see how far this algorithm is from this bound. Secondly, although superior results were obtained for a multitarget situation, the proposed algorithm was slightly outperformed by FSLAM in a single target scenario. Presumably this is because posterior correlations in the sensors' positions are removed to save computational expense. This suggests that accounting for sensors' positions correlations will create a better estimate of the map. However, how to do it in a computationally efficient manner for MSLAM-TUKF is well-worth exploring. It has been assumed that the aiming directions of the antennas included

in the sensors are known. This implies that the orientations of the sensors are known. In a more realistic scenario, these orientations would be unknown so a generalisation of the algorithm for this case is also a topic of interest. In addition, it has also been assumed that the number of targets is known. Provided the measurements permit identifiability of all the unknown parameters, including the number of targets, a modification of the algorithm in Chapter 4 could be applied.

Chapter 7

Tracking of multiple targets using non-synchronised sensor networks

7.1 Introduction

This chapter deals with multiple target tracking using centralised sensor networks when the sensors are not synchronised. As will be explained, because of the lack of synchronisation of the sensors, traditional particle filters have a very high computational burden which could imply that they cannot be used for real-time applications. Consequently, the aim of this chapter is to develop a particle filter to perform multiple target tracking with a relatively low computational burden compared to conventional particle filters.

Every sensor measures and sends information according to an internal clock and, thus, it has its own sampling frequency and initial timing offset. It is usually assumed that these internal clocks are synchronised. However, estimating and compensating the timing offsets at the sensors using time synchronisation protocols imply a significant increase in the communication overhead and, consequently, an undesired reduction in the life of the battery-supported nodes [109, 194]. Then, non-synchronised sensor networks are preferable so as to optimise the life of the nodes if performance is not severely degraded.

Firstly, some techniques that have been proposed in the literature to perform tracking with non-synchronised sensors are discussed. In [194], the authors propose an algorithm that jointly estimates the targets positions and the timing offset of each sensor although the sampling frequency is assumed to be the same for all sensors.

In [124, 167], a tracking algorithm for non-synchronised, biased sensors is proposed with known sampling instants using a Gaussian approximation. An algorithm that considers sensors whose sampling instants are not completely known is given in [101] but it is constrained to linear/Gaussian models. Reference [168] analyses several algorithms for tracking using asynchronous sensors assuming the sampling instants are known. However, none of these methods aims to reduce the high computational burden that arises as a consequence of processing measurements taken at different time instants using sensor networks with a large number of sensors.

In this chapter, an algorithm based on particle filters, the asynchronous PF, is proposed to carry out multiple target tracking using non-synchronous sensors with low computational burden compared to the burden of conventional PFs. It is assumed that the dynamic model of the targets is linear and Gaussian but the measurement model is nonlinear/non-Gaussian. Moreover, the sampling instants of the sensors are known as in [124, 168].

The problem consists of estimating the states of the targets synchronously even though the sensors report measurements non-synchronously. This is due to the fact that it is assumed that the operator in charge of the network only needs an update of the surveillance system periodically according to an update period. The algorithm is developed for fixed and known number of targets for ease of exposition. However, it could easily be generalised for variable and unknown number of targets using the ideas in Chapter 4. It should also be mentioned that the asynchronous PF is applied to specific dynamic and measurement models as well as a specific importance density. Nevertheless, as presented in Section 7.3, it is a general algorithm which could be applied to other models and importance densities.

Other PFs especially designed for lowering the computational burden, to the point of being suitable for real-time applications, are analysed in [120]. An important feature of these algorithms is that their computational burden can be controlled adaptively. These algorithms are based on reducing the number of particles in each update step when the measurements arrive too quickly, discarding measurements or aggregating them (what will be called synchronous PF in this chapter) to reduce the computational burden. On the contrary, the asynchronous PF developed in this chapter propagates the particles several steps ahead but taking into account the time instant at which each measurement was taken. This way the particles are propagated fewer times speeding up the algorithm without the need of reducing the number of particles, which can make the filter diverge [56, 120]. Moreover, as the time instant each measurement was taken is accounted for, the performance of the asynchronous PF is very close to the traditional PF and vastly outperforms the synchronous PF. This is demonstrated by numerical simulations.

It is also worth remarking that the problem of tracking using an asynchronous PF is related to the problem of tracking with out-of-sequence measurements. In this problem, the measurements do not arrive in order and, then, the posterior PDF of the target state is updated with a measurement that was taken in the past. However, the motivation and solution of this problem is different to the problem analysed in this chapter as the state is estimated every time a measurement arrives and there is no reduction in computational burden when using out-of-sequence measurements. This problem for linear/Gaussian models is addressed in [12] and for nonlinear/non-Gaussian models in [159, 162].

This chapter is organised as follows. In Section 7.2, the kinematic model of the targets and the sensor model are addressed. In Section 7.3, the differences between the asynchronous and the traditional PF are pointed out and the equations for updating the weights of the particles in the asynchronous case are developed. In Section 7.4, the result of the previous section is applied to the multiple target case. The performance of several algorithms is compared in Section 7.5. Finally, conclusions are drawn in Section 7.6.

7.2 Problem statement

It is assumed there are a fixed and known number of targets, t , in the surveillance area. As stated in the introduction, the objective is to estimate the multitarget state periodically according to an update period T_u . Then, the time instant when the k th multitarget state is to be estimated takes place at time kT_u . The k th update interval is defined as the set of time instants from $(k-1)T_u$ to kT_u .

The state of target i is described at time kT_u by the four-dimensional vector $\mathbf{x}_i^{kT_u} = [x_i^{kT_u}, \dot{x}_i^{kT_u}, y_i^{kT_u}, \dot{y}_i^{kT_u}]^T$ where $[x_i^{kT_u}, y_i^{kT_u}]^T$ is the position vector and $[\dot{x}_i^{kT_u}, \dot{y}_i^{kT_u}]^T$ is the velocity vector. The multitarget state vector is formed by stacking the individual target state vectors into one vector $\mathbf{X}^{kT_u} = [(\mathbf{x}_1^{kT_u})^T, (\mathbf{x}_2^{kT_u})^T, \dots, (\mathbf{x}_t^{kT_u})^T]^T$. The multitarget states up to time kT_u have been observed by the collection of measurements $\mathbf{z}^{1:kT_u} = (\mathbf{z}^{T_u}, \mathbf{z}^{2T_u}, \dots, \mathbf{z}^{kT_u})$ up to time kT_u where $\mathbf{z}^{kT_u} = [z_1^{\tau_1^k}, z_2^{\tau_2^k}, \dots, z_{M^k}^{\tau_{M^k}^k}]^T$ represents the sensor measurements in the k th update interval, $z_j^{\tau_j^k}$ is the j th measurement in the k th update interval, which was taken at time¹ $\tau_j^k \in ((k-1)T_u, kT_u]$, and, M^k is the total number of measurements in the k th update interval. It is assumed that the com-

¹The set $(a, b]$ stands for the interval of numbers between a and b such that the endpoint a is not included in the set but b is.

munication delay is negligible so that the measurement time is the time of arrival at the fusion centre. To simplify the equations later on, vector $\mathbf{z}^{kT_u}$ is ordered such that $\tau_j^k \geq \tau_{j-1}^k$. The sensor that takes the j th measurement $z_j^{\tau_j^k}$ has the label $\iota_j^{\tau_j^k} \in \{1, \dots, M\}$ where M is the total number of sensors in the network.

From the Bayesian standpoint, the aim is to approximate the posterior PDF $p(\mathbf{X}^{kT_u} | \mathbf{z}^{1:kT_u})$ assuming $p(\mathbf{X}^{(k-1)T_u} | \mathbf{z}^{1:(k-1)T_u})$ is available [9]. The exact solution to the problem implies applying the prediction and update equations every time a measurement from a sensor is taken in the k th update interval. These time instants are represented by $\{\tau_j^k : j \in \{1, \dots, M^k\}\}$. The prediction equation from time τ_{j-1}^k to τ_j^k is given by the Chapman-Kolmogorov equation:

$$\begin{aligned} p(\mathbf{X}^{\tau_j^k} | \mathbf{z}^{1:(k-1)T_u}, z_1^{\tau_1^k}, \dots, z_{j-1}^{\tau_{j-1}^k}) \\ = \int d\mathbf{X}^{\tau_{j-1}^k} p(\mathbf{X}^{\tau_j^k} | \mathbf{X}^{\tau_{j-1}^k}) p(\mathbf{X}^{\tau_{j-1}^k} | \mathbf{z}^{1:(k-1)T_u}, z_1^{\tau_1^k}, \dots, z_{j-1}^{\tau_{j-1}^k}) \end{aligned} \quad (7.1)$$

The update equation at time τ_j^k is given by Bayes' rule:

$$\begin{aligned} p(\mathbf{X}^{\tau_j^k} | \mathbf{z}^{1:(k-1)T_u}, z_1^{\tau_1^k}, \dots, z_{j-1}^{\tau_{j-1}^k}, z_j^{\tau_j^k}) \\ = \frac{p(z_j^{\tau_j^k} | \mathbf{X}^{\tau_j^k}) p(\mathbf{X}^{\tau_j^k} | \mathbf{z}^{1:(k-1)T_u}, z_1^{\tau_1^k}, \dots, z_{j-1}^{\tau_{j-1}^k})}{p(z_j^{\tau_j^k} | \mathbf{z}^{1:(k-1)T_u}, z_1^{\tau_1^k}, \dots, z_{j-1}^{\tau_{j-1}^k})} \end{aligned} \quad (7.2)$$

where it has been used that the measurement of a sensor at time τ_j^k only depends on the state of the targets at time τ_j^k . A diagram of states of the problem is shown in Figure 7.1.

To completely define (7.1) and (7.2), the dynamic model of targets and the sensor model need to be specified. This is done in the next subsections.

7.2.1 Dynamic model of targets

In this section, the dynamic model of targets is specified. A continuous-time nearly-constant-velocity model, which is a continuous-time Markov process that characterises the movements of the targets, is used. For each target [12, 13]:

$$\mathbf{x}^{\tau_2} = \mathbf{F}(\tau_2, \tau_1) \mathbf{x}^{\tau_1} + \mathbf{w}(\tau_2, \tau_1) \quad (7.3)$$

where $\mathbf{x}^{\tau_2}$ is the state vector of the target at time τ_2 , $\mathbf{F}(\tau_2, \tau_1)$ is the transition matrix from time τ_1 to τ_2 , $\mathbf{w}(\tau_2, \tau_1)$ is the cumulative effect of the process noise

Figure 7.1: Diagram of states of the problem. The multitarget states that one is interested in estimating are represented by big circles. However, some of the collected measurements are originated from intermediate states, represented by small circles. The traditional PF needs to approximate the posterior PDF of all the states including the intermediate ones. Conversely, the asynchronous PF is able to approximate the posterior PDF of the required states without the posterior PDF of the intermediate ones.

from time τ_1 to τ_2 , which is assumed to be a zero-mean Gaussian noise, and $\tau_2 > \tau_1$. The transition matrix and the covariance matrix of the cumulative effect of the process noise are [13]:

$$\mathbf{F}(\tau_2, \tau_1) = \mathbf{I}_2 \otimes \begin{pmatrix} 1 & \Delta\tau \\ 0 & 1 \end{pmatrix} \quad (7.4)$$

$$\mathbf{Q}(\tau_2, \tau_1) = \sigma_u^2 \mathbf{I}_2 \otimes \begin{pmatrix} \frac{(\Delta\tau)^3}{3} & \frac{(\Delta\tau)^2}{2} \\ \frac{(\Delta\tau)^2}{2} & \Delta\tau \end{pmatrix} \quad (7.5)$$

where $\Delta\tau = \tau_2 - \tau_1$, $\mathbf{Q}(\tau_2, \tau_1)$ is the covariance matrix of the accumulated noise from time τ_1 to τ_2 , and σ_u^2 is the continuous-time process noise intensity [13]. Assuming that the movement of each target is independent of the rest, $p(\mathbf{X}^{\tau_2} | \mathbf{X}^{\tau_1})$ is characterised by (7.3).

7.2.2 Sensor model

The sensor model of Section 4.2.2 is used. That is, the measurements are independent from sensor to sensor (conditioned on the multitarget state) with a distribution which depends on the total signal-to-noise ratio. The signal-to-noise ratio produced by a single target depends on the distance from the target to the sensor, r , following a piecewise linear function. The function is defined with $N + 1$ intervals whose endpoints and values are $\{r_0 = 0, r_1, \dots, r_N, r_{N+1} = \infty\}$ and $SNR_{r,i}$

$i \in \{1, \dots, N + 1\}$, respectively. Therefore:

$$SNR(r) = \begin{cases} SNR_{r,1} & 0 \leq r < r_1 \\ \dots & \dots \\ SNR_{r,N} & r_{N-1} \leq r < r_N \\ 0 & r \geq r_N \end{cases} \quad (7.6)$$

The reason for using this model in this chapter is because of the simplicity of the equations that follow hereafter and also because any measurement model can be approximated using a piecewise linear function.

Assuming that the sensors report binary detections for the sake of simplicity and recalling that $z_j^{\tau_j^k}$ is the measurement taken by sensor $\iota_j^{\tau_j^k}$ at time τ_j^k . The likelihood for each sensor is [116, 141]:

$$\begin{aligned} p\left(z_j^{\tau_j^k} \mid \mathbf{X}^{\tau_j^k}\right) &= P_{fa}^{1/\left(1+SNR\left(\mathbf{X}^{\tau_j^k}, \iota_j^{\tau_j^k}\right)\right)} \delta\left(z_j^{\tau_j^k} - 1\right) \\ &+ \left(1 - P_{fa}^{1/\left(1+SNR\left(\mathbf{X}^{\tau_j^k}, \iota_j^{\tau_j^k}\right)\right)}\right) \delta\left(z_j^{\tau_j^k}\right) \end{aligned} \quad (7.7)$$

where $SNR\left(\mathbf{X}^{\tau_j^k}, \iota_j^{\tau_j^k}\right)$ is the signal-to-noise ratio at sensor $\iota_j^{\tau_j^k}$ produced by the multitarget state $\mathbf{X}^{\tau_j^k}$, P_{fa} is the probability of false alarm of the sensors and detections are assumed to be taken from a Rayleigh distribution. This distribution corresponds to envelope-detected signals under a Gaussian model. According to this model, the signal-to-noise ratio $SNR\left(\mathbf{X}^{\tau_j^k}, \iota_j^{\tau_j^k}\right)$ at sensor $\iota_j^{\tau_j^k}$ is the sum of the signal-to-noise ratios produced by each target [16].

7.3 Asynchronous particle filter

Firstly, the differences between a traditional and an asynchronous PF are pointed out recalling that the problem of interest is to estimate the multitarget state at the time steps $\{kT_u : k \in \mathbb{N}\}$. In a traditional PF, particles must be propagated and updated with measurements that are taken at the same time [9]. This is not a drawback in synchronised WSNs as all the measurements from the sensors are obtained at time steps $\{kT_u : k \in \mathbb{N}\}$. Then, one only needs to propagate and update the particles once to approximate the posterior PDF of the state at time kT_u based on $(k - 1)T_u$. However, in non-synchronised WSNs, the approximation of the posterior

Figure 7.2: Particle propagation for the asynchronous, traditional and synchronous particle filters in an update interval with two measurements. The arrows indicate how the particles are propagated. The stars indicate when the filter assumes the two measurements take place.

PDF at time kT_u needs to be done by propagating and updating the particles every time a measurement is received noting that all of them might have been taken at different time instants. The problem is that this algorithm is computationally very demanding for large sensor networks as many prediction and update steps are performed. On the other hand, the asynchronous PF propagates the particles only once and updates their weights based on all the past measurements from time $(k-1)T_u$ to kT_u accounting for their time instants. This solution is a much faster way to approximate the posterior than the traditional PF with just a slight decrease in performance as demonstrated by simulations in Section 7.5. Another fast method is to use a synchronous PF which is a suboptimal approach that does not take into account when the measurement was exactly taken [120]. With all the binary measurements received from a sensor in an update interval, the synchronous PF applies an OR function and updates the PF as if all the measurements had been taken at time steps $\{kT_u : k \in \mathbb{N}\}$. A diagram of the propagation of the particles for these filters is shown in Figure 7.2.

Once the differences between a traditional and an asynchronous PFs have been outlined, the equation of the propagation of the weights of the particles for the asynchronous case is calculated. According to (43) in [9], the weight of particle i at update instant number k , w_i^k , is:

$$w_i^k \propto \frac{p(\mathbf{X}_i^{0:kT_u} | \mathbf{z}^{1:kT_u})}{q(\mathbf{X}_i^{0:kT_u} | \mathbf{z}^{1:kT_u})} \quad (7.8)$$

where $\mathbf{X}_i^{0:kT_u}$ represents all the states of the targets of particle i in each update ins-

tant until time kT_u , $\mathbf{z}^{1:kT_u}$ is the vector that contains all the sensors' measurements until time kT_u and $q(\cdot)$ is the importance density for propagating particles. In the same way as in (44) in [9], the importance density is chosen to factorise such as:

$$q\left(\mathbf{X}^{0:kT_u} \mid \mathbf{z}^{1:kT_u}\right) = q\left(\mathbf{X}^{kT_u} \mid \mathbf{X}^{0:(k-1)T_u}, \mathbf{z}^{1:kT_u}\right) q\left(\mathbf{X}^{0:(k-1)T_u} \mid \mathbf{z}^{1:(k-1)T_u}\right) \quad (7.9)$$

Besides, using (45) in [9],

$$\begin{aligned} p\left(\mathbf{X}^{0:kT_u} \mid \mathbf{z}^{1:kT_u}\right) &= \frac{p\left(\mathbf{z}^{kT_u} \mid \mathbf{X}^{0:kT_u}, \mathbf{z}^{1:(k-1)T_u}\right) p\left(\mathbf{X}^{kT_u} \mid \mathbf{X}^{0:(k-1)T_u}, \mathbf{z}^{1:(k-1)T_u}\right)}{p\left(\mathbf{z}^{kT_u} \mid \mathbf{z}^{1:(k-1)T_u}\right)} \\ &\quad \cdot p\left(\mathbf{X}^{0:(k-1)T_u} \mid \mathbf{z}^{1:(k-1)T_u}\right) \end{aligned} \quad (7.10)$$

where $\mathbf{z}^{kT_u}$ are the measurements sent by the sensors from time $(k-1)T_u$ to kT_u as defined in Section 7.2. Then, $\mathbf{z}^{kT_u}$ does not depend only on the state at time kT_u but on a number of intermediate states from time $(k-1)T_u$ to kT_u . As given the state at time $(k-1)T_u$, these intermediate states do not depend on states or measurements before that time because of the Markov property of (7.3), one can write:

$$p\left(\mathbf{z}^{kT_u} \mid \mathbf{X}^{0:kT_u}, \mathbf{z}^{1:(k-1)T_u}\right) = p\left(\mathbf{z}^{kT_u} \mid \mathbf{X}^{kT_u}, \mathbf{X}^{(k-1)T_u}\right) \quad (7.11)$$

It should be mentioned that in a synchronous sensor network set-up with a traditional PF, $\mathbf{z}^{kT_u}$ includes the measurements and all of them are taken at time kT_u . Then, $\mathbf{z}^{kT_u}$ would depend only on $\mathbf{X}^{kT_u}$. This is clearly not the case in our set-up according to the definition of $\mathbf{z}^{kT_u}$ in Section 7.2.

Then, substituting (7.11) in (7.10):

$$\begin{aligned} p\left(\mathbf{X}^{0:kT_u} \mid \mathbf{z}^{1:kT_u}\right) &= \frac{p\left(\mathbf{z}^{kT_u} \mid \mathbf{X}^{(k-1)T_u}, \mathbf{X}^{kT_u}\right) p\left(\mathbf{X}^{kT_u} \mid \mathbf{X}^{(k-1)T_u}\right)}{p\left(\mathbf{z}^{kT_u} \mid \mathbf{z}^{1:(k-1)T_u}\right)} \\ &\quad \cdot p\left(\mathbf{X}^{0:(k-1)T_u} \mid \mathbf{z}^{1:(k-1)T_u}\right) \end{aligned} \quad (7.12)$$

Substituting (7.9) and (7.12) in (7.8), the equation of the weights' propagation for the asynchronous PF is:

$$w_i^k = w_i^{k-1} \frac{p\left(\mathbf{z}_i^{kT_u} \mid \mathbf{X}_i^{kT_u}, \mathbf{X}_i^{(k-1)T_u}\right) p\left(\mathbf{X}_i^{kT_u} \mid \mathbf{X}_i^{(k-1)T_u}\right)}{q\left(\mathbf{X}_i^{kT_u} \mid \mathbf{X}_i^{0:(k-1)T_u}, \mathbf{z}_i^{1:kT_u}\right)} \quad (7.13)$$

The result given by (7.13) is a general result that provides the equation of the propagation of the weights in an asynchronous PF for a general importance density. It should be mentioned that the difference between the equation of the propagation

of the weights in the traditional PF and the asynchronous PF is that the former uses the likelihood while the latter uses $p(\mathbf{z}^{kT_u} | \mathbf{X}_i^{kT_u}, \mathbf{X}_i^{(k-1)T_u})$, which is referred to as asynchronous likelihood so-called because of its resemblance to the likelihood in a traditional PF.

7.4 Application to multiple target tracking

In this section, the asynchronous PF is applied to multiple target tracking using a sensor network. The transition density $p(\mathbf{X}_i^{kT_u} | \mathbf{X}_i^{(k-1)T_u})$ is characterised by (7.3) assuming that the movements of all the targets are independent as in Chapter 4. In the subsequent, the asynchronous likelihood is calculated for the problem statement detailed in Section 7.2. In addition, an importance density $q(\cdot)$ based on the independent partition (IP) method is proposed [116, 161].

As in the previous chapters, in order to approximate the posterior PDF of the multiple target state, Rao-Blackwellisation is used because the relationship between the position and the velocity is linear and Gaussian. Thus, the velocity elements can be computed using a Kalman filter, see Section 2.5.4, and, consequently, the Rao-Blackwellised particle $\mathbf{X}_i^{kT_u} = (\mathbf{P}_i^{kT_u}, \bar{\mathbf{V}}_i^{kT_u}, \boldsymbol{\Sigma}_i^{kT_u})$ has three elements: the positions of the targets $\mathbf{P}_i^{kT_u}$, the expected velocities of the targets $\bar{\mathbf{V}}_i^{kT_u}$, and the covariance matrix of the velocity elements $\boldsymbol{\Sigma}_i^{kT_u}$. The importance density to propagate the particles in this multiple target tracking problem corresponds to the independent partition (IP) method [116] with Rao-Blackwellisation where the likelihood is substituted by the asynchronous likelihood as explained in the following.

7.4.1 Asynchronous likelihood

In this section, the asynchronous likelihood $p(\mathbf{z}^{kT_u} | \mathbf{X}_i^{kT_u}, \mathbf{X}_i^{(k-1)T_u})$ is calculated. Decomposing the Rao-Blackwellised particle into its components:

$$\begin{aligned} p(\mathbf{z}^{kT_u} | \mathbf{X}_i^{kT_u}, \mathbf{X}_i^{(k-1)T_u}) \\ = p(\mathbf{z}^{kT_u} | \mathbf{P}_i^{kT_u}, \bar{\mathbf{V}}_i^{kT_u}, \boldsymbol{\Sigma}_i^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}) \end{aligned} \quad (7.14)$$

Due to the fact that given $\mathbf{P}_i^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}$, applying a Kalman filter, one can obtain $\bar{\mathbf{V}}_i^{kT_u}, \boldsymbol{\Sigma}_i^{kT_u}$, the conditioning on these parameters can be dropped in (7.14). The particle index in the notation is also dropped for the sake of clarity and, as explained in Appendix A.29, (7.14) is approximated as:

$$\begin{aligned}
 p\left(\mathbf{z}^{kT_u} \mid \mathbf{P}^{kT_u}, \mathbf{P}^{(k-1)T_u}, \bar{\mathbf{V}}^{(k-1)T_u}, \boldsymbol{\Sigma}^{(k-1)T_u}\right) \\
 \simeq \prod_{j=1}^{M^k} p\left(z_j^{\tau_j^k} \mid \mathbf{P}^{kT_u}, \mathbf{P}^{(k-1)T_u}, \bar{\mathbf{V}}^{(k-1)T_u}, \boldsymbol{\Sigma}^{(k-1)T_u}\right) \quad (7.15)
 \end{aligned}$$

As the measurement at time τ_j^k , $z_j^{\tau_j^k}$, depends only on the positions of the targets at time τ_j^k , $\mathbf{P}^{\tau_j^k}$, each factor in (7.15) is integrated over $\mathbf{P}^{\tau_j^k}$ so that the conditioning of $z_j^{\tau_j^k}$ depends only on one parameter:

$$\begin{aligned}
 p\left(z_j^{\tau_j^k} \mid \mathbf{P}^{kT_u}, \mathbf{P}^{(k-1)T_u}, \bar{\mathbf{V}}^{(k-1)T_u}, \boldsymbol{\Sigma}^{(k-1)T_u}\right) \\
 = \int p\left(z_j^{\tau_j^k} \mid \mathbf{P}^{\tau_j^k}\right) \cdot p\left(\mathbf{P}^{\tau_j^k} \mid \mathbf{P}^{kT_u}, \mathbf{P}^{(k-1)T_u}, \bar{\mathbf{V}}^{(k-1)T_u}, \boldsymbol{\Sigma}^{(k-1)T_u}\right) d\mathbf{P}^{\tau_j^k} \quad (7.16)
 \end{aligned}$$

where $p\left(z_j^{\tau_j^k} \mid \mathbf{P}^{\tau_j^k}\right) = p\left(z_j^{\tau_j^k} \mid \mathbf{X}^{\tau_j^k}\right)$, as the measurement does not depend on the velocities, was given in Section 7.2.2.

Now, it is explained how to calculate the other PDF in (7.16). As it has been assumed that the targets move independently:

$$\begin{aligned}
 p\left(\mathbf{P}^{\tau_j^k} \mid \mathbf{P}^{kT_u}, \mathbf{P}^{(k-1)T_u}, \bar{\mathbf{V}}^{(k-1)T_u}, \boldsymbol{\Sigma}^{(k-1)T_u}\right) \\
 = \prod_{i=1}^t p\left(\mathbf{p}_i^{\tau_j^k} \mid \mathbf{p}_i^{kT_u}, \mathbf{p}_i^{(k-1)T_u}, \bar{\mathbf{v}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}\right) \quad (7.17)
 \end{aligned}$$

where $\mathbf{p}_i^{kT_u}$ is the position of target i at time kT_u , $\bar{\mathbf{v}}_i^{(k-1)T_u}$ is the expected velocity of target i at time $(k-1)T_u$ and $\boldsymbol{\Sigma}_i^{(k-1)T_u}$ is the covariance matrix of the velocity of target i at time $(k-1)T_u$. In Appendix A.30, it is shown that the distribution of $\mathbf{p}_i^{\tau_j^k}$ given $\mathbf{p}_i^{kT_u}$, $\mathbf{p}_i^{(k-1)T_u}$, $\bar{\mathbf{v}}_i^{(k-1)T_u}$ and $\boldsymbol{\Sigma}_i^{(k-1)T_u}$ is Gaussian and its mean and covariance matrix are given.

First, equation (7.16) is calculated for just one target and later on it will be generalised for multiple targets. Using (7.6) in (7.16):

$$\begin{aligned}
 p\left(z_j^{\tau_j^k} \mid \mathbf{p}_i^{kT_u}, \mathbf{p}_i^{(k-1)T_u}, \bar{\mathbf{v}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}\right) \\
 = \sum_{n=1}^{N+1} p\left(z_j^{\tau_j^k} \mid SNR_{r,n}\right) \\
 \cdot p\left(r_{n-1}^2 \leq r^2 \left(\mathbf{p}_i^{\tau_j^k}, \ell_j^{\tau_j^k}\right) < r_n^2 \mid \mathbf{p}_i^{kT_u}, \mathbf{p}_i^{(k-1)T_u}, \bar{\mathbf{v}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}\right) \quad (7.18)
 \end{aligned}$$

where

$$r^2 \left(\mathbf{p}_i^{\tau_j^k}, \ell_j^{\tau_j^k} \right) = \left(x_i^{\tau_j^k} - x_{s, \ell_j^{\tau_j^k}} \right)^2 + \left(y_i^{\tau_j^k} - y_{s, \ell_j^{\tau_j^k}} \right)^2 \quad (7.19)$$

is the square distance from target i to the sensor $\ell_j^{\tau_j^k}$ that takes the j measurement at time τ_j^k , whose position is known $\begin{bmatrix} x_{s, \ell_j^{\tau_j^k}} & y_{s, \ell_j^{\tau_j^k}} \end{bmatrix}^T$, and $\mathbf{p}_i^{\tau_j^k} = \begin{bmatrix} x_i^{\tau_j^k} & y_i^{\tau_j^k} \end{bmatrix}^T$.

Then, in order to calculate (7.18), it is needed to evaluate two probabilities: the first one is given by the measurement model, equation (7.7), and the second is calculated now. To do so, first, one should notice that the distribution of $\mathbf{p}_i^{\tau_j^k}$ given $\mathbf{p}_i^{kT_u}$, $\mathbf{p}_i^{(k-1)T_u}$, $\bar{\mathbf{v}}_i^{(k-1)T_u}$ and $\Sigma_i^{(k-1)T_u}$ is Gaussian with mean and covariance matrix given by (A.236) and (A.237) in Appendix A.30. Besides, as the process model is the same in the x and y components, the covariance matrix of the position of the intermediate state is always diagonal and their elements $\sigma_{i, \tau_j^k}^2$ are the same, provided that the prior covariance matrix of the position is also diagonal with equal elements. The value of $\sigma_{i, \tau_j^k}^2$ depends on the number of time instants that the target has been tracked and the a priori covariance matrix of the position of the targets, see (A.237) in Appendix A.30.

Therefore, according to (7.19), the square distance between the target i and sensor $\ell_j^{\tau_j^k}$ at time instant τ_j^k , $r^2 \left(\mathbf{p}_i^{\tau_j^k}, \ell_j^{\tau_j^k} \right)$, is the square sum of two Gaussian random variables with the same variance $\sigma_{i, \tau_j^k}^2$ and non-zero mean. Then, the variable $r^2 \left(\mathbf{p}_i^{\tau_j^k}, \ell_j^{\tau_j^k} \right) / \sigma_{i, \tau_j^k}^2$ follows a non-central chi-square distribution [82] with $m = 2$ degrees of freedom and parameter

$$\lambda = \lambda \left(\mathbf{p}_i^{\tau_j^k}, \ell_j^{\tau_j^k} \right) = \frac{\left(\bar{x}_i^{\tau_j^k} - x_{s, \ell_j^{\tau_j^k}} \right)^2 + \left(\bar{y}_i^{\tau_j^k} - y_{s, \ell_j^{\tau_j^k}} \right)^2}{\sigma_{i, \tau_j^k}^2} \quad (7.20)$$

where $\bar{x}_i^{\tau_j^k}$ and $\bar{y}_i^{\tau_j^k}$ are the components of the vector given by (A.236). Therefore,

$$\begin{aligned} & p \left(z_j^{\tau_j^k} \mid \mathbf{p}_i^{kT_u}, \mathbf{p}_i^{(k-1)T_u}, \bar{\mathbf{v}}_i^{(k-1)T_u}, \Sigma_i^{(k-1)T_u} \right) \\ &= \sum_{n=1}^{N+1} p \left(z_j^{\tau_j^k} \mid SNR_{r,n} \right) \left[F_{r^2 \left(\mathbf{p}_i^{\tau_j^k}, \ell_j^{\tau_j^k} \right) \mid \mathbf{p}_i^{0:kT_u}} \left(r_n^2 \right) - F_{r^2 \left(\mathbf{p}_i^{\tau_j^k}, \ell_j^{\tau_j^k} \right) \mid \mathbf{p}_i^{0:kT_u}} \left(r_{n-1}^2 \right) \right] \end{aligned} \quad (7.21)$$

where the distribution function of the square distance is [82]

$$F_{r^2 \left(\begin{smallmatrix} \tau_j^k \\ \mathbf{p}_i^{\tau_j^k}, \tau_j^k \end{smallmatrix} \right) | \mathbf{p}_i^{0:kT_u}} (r^2) = \frac{1}{2 \cdot \sigma_{i, \tau_j^k}^2} \int_0^{r^2} e^{-\left(\frac{x}{\sigma_{i, \tau_j^k}^2} + \lambda \right) / 2} \cdot I_0 \left(\sqrt{\lambda \frac{x}{\sigma_{i, \tau_j^k}^2}} \right) dx \quad (7.22)$$

and I_0 is the modified Bessel function of the first kind of order 0.

Generalising (7.21) for multiple targets and substituting the result into (7.15), the approximation of the asynchronous likelihood is:

$$\begin{aligned} p \left(\mathbf{z}^{kT} \mid \mathbf{P}^{kT_u}, \mathbf{P}^{(k-1)T_u}, \bar{\mathbf{V}}^{(k-1)T_u}, \boldsymbol{\Sigma}^{(k-1)T_u} \right) \\ \simeq \prod_{j=1}^{M^k} \sum_{i_1=1}^{N+1} \sum_{i_2=1}^{N+1} \dots \sum_{i_t=1}^{N+1} p \left(z_j^k \mid SNR_j = \sum_{n=1}^t SNR_{r, i_n} \right) \\ \cdot \prod_{l=1}^t \left[F_{r^2 \left(\begin{smallmatrix} \tau_j^k \\ \mathbf{p}_l^{\tau_j^k}, \tau_j^k \end{smallmatrix} \right) | \mathbf{p}_l^{0:kT_u}} (r_{i_l}^2) - F_{r^2 \left(\begin{smallmatrix} \tau_j^k \\ \mathbf{p}_l^{\tau_j^k}, \tau_j^k \end{smallmatrix} \right) | \mathbf{p}_l^{0:kT_u}} (r_{i_l-1}^2) \right] \end{aligned} \quad (7.23)$$

7.4.2 Importance density

The importance density that has been used to deal with this problem is the IP method for the sake of clarity. However, the PP method can be perfectly applied as the PP and IP methods are closely related as explained in Section 4.3.1. The IP method proposes the new state of a target independently of the rest of the targets. The only difference is that in this case, the first-stage weights are calculated using the asynchronous likelihood rather than the likelihood but the steps are similar to the ones in Section 4.3.1 and are reproduced in Table 7.1. Finally, a summary of the complete algorithm for multiple target tracking using an asynchronous PF is given in Table 7.2.

7.5 Simulation results

In this section, three different filters are compared: an asynchronous PF, a traditional PF and a synchronous PF. The RMS position error of the targets for the three filters is estimated in different sensor sampling set-ups.

Although it has been assumed that the sensors sample a continuous trajectory at any time in the theory, a sampled trajectory has been used for simulation purposes. This way, the trajectory is sampled with a sampling period τ . It is assumed that the update period T_u is multiple of τ , i.e., $T_u = N_{slot} \cdot \tau$ where N_{slot} is the number of

Table 7.1: Independent partition subroutine for the asynchronous particle filter
 IP Subroutine for target j with Rao-Blackwellisation

- For each particle $i = 1, \dots, N_{par}$:
 - Get a sample for the position at time kT_u $\mathbf{p}_{j,i}^* \sim p\left(\mathbf{p}_j^{kT_u} \mid \mathbf{x}_{j,i}^{(k-1)T_u}\right)$ using (7.3).
 - Calculate the mean speed and covariance matrix using a Kalman filter $\bar{\mathbf{v}}_{j,i}^*$ y $\Sigma_{j,i}^*$.
 - Calculate $b_j\left(\mathbf{p}_{j,i}^*\right) = p\left(\mathbf{z}^{k+1} \mid \mathbf{p}_{j,i}^*, \mathbf{x}_{j,i}^{(k-1)T_u}\right)$ using (7.15).
- Normalise $b_j\left(\mathbf{p}_{j,i}^*\right)$ to sum to one over $i = 1, \dots, N_{par}$.
- Resampling step for each track. For each particle $i = 1, \dots, N_{par}$:
 - Sample an index p from the distribution defined by $b_j\left(\mathbf{p}_{j,l}^*\right)$ over $l = 1, \dots, N_{par}$.
 - Set $\mathbf{x}_{j,i}^{kT_u} = \left\{\mathbf{p}_{j,p}^*, \bar{\mathbf{v}}_{j,p}^*, \Sigma_{j,p}^*\right\}$ and the parent's index $a_j^i = p$.

Table 7.2: Proposed algorithm for tracking multiple targets using non-synchronised sensors

- For each update interval:
 - For each target, propose a new state using IP subroutine, see Table 7.1.
 - Update the weights of the particles using (7.13).
 - Normalise the weights of the particles to sum to one.
 - Perform resampling to obtain an evenly weighted particle set.

possible instants (number of slots) in which a sensor can measure in an update interval. The sampling instants of the sensors, which are known, are obtained following this procedure:

- The initial sampling instant for sensor j , $T_{ini,j}$, is a uniform continuous random variable in the interval $[0, \tau \cdot N_{slot}]$.
- The sampling period for sensor j , $T_{s,j}$, is drawn from a Gaussian distribution with mean T_s and variance σ_T^2 at the beginning of the simulation. This variance indicates that the internal clocks of the sensors have a slightly different sampling frequency due to the fabrication process.
- Sensor j samples at the time instants² $T_j(n) = \left\lceil \frac{T_{ini,j} + n \cdot T_{s,j}}{\tau} \right\rceil \cdot \tau$ where $n \in \{0, 1, 2, 3, \dots, \infty\}$.

In the simulations, the surveillance area is a rectangle of dimensions $5000 \text{ m} \times 2500 \text{ m}$. The sensors are deployed in a square grid of dimensions 50×25 whose side is 100 m and their P_{fa} is set to $2 \cdot 10^{-5}$. The SNR depends on the distance from the target to the sensor in the following way:

$$SNR(r) = \begin{cases} 20 \text{ dB} & r < 60 \text{ m} \\ 10 \text{ dB} & 60 \text{ m} \leq r < 80 \text{ m} \\ 0 & r \geq 80 \text{ m} \end{cases} \quad (7.24)$$

There are two targets and their trajectories, which are plotted in Figure 7.3, have 2000 samples with sampling period $\tau = 0.05 \text{ s}$. They were generated using (7.3) with a process noise in which $\sigma_u = 5 \text{ m/s}^{3/2}$. There is one crossing in the trajectories. Nonetheless, the targets are not located at the same position at the same time so the IP method works properly as the targets are far enough from each other at all times [116]. The prior PDF of the initial positions of the targets is:

$$\mathbf{p}_i^0 \sim \mathcal{N}(\mathbf{p}_i^0; \bar{\mathbf{p}}_i^0, \sigma_0^2 \mathbf{I}_2) \quad (7.25)$$

for $i \in \{1, \dots, t\}$ with $\mathbf{I}_m$ denoting the $m \times m$ identity matrix, $\bar{\mathbf{p}}_i^0$ is the expected position of target i at time 0, $\sigma_0^2 \mathbf{I}_2$ is the covariance matrix of the initial position at time 0 which is assumed to be the same for all the targets for simplicity and $\sigma_0 = 10 \text{ m}$ in the simulations.

The asynchronous PF is compared with two algorithms: a traditional and a synchronous PF. The traditional PF entails propagating the particles using IP at every sampling time, i.e., every τ using the measurements of the sensors that are

²The notation $\lceil a \rceil$ stands for the minimum integer higher or equal to a .

Figure 7.3: Trajectories of the targets used to assess the asynchronous PF performance. Each target is identified at the beginning of its trajectory by a number and there is an arrow to indicate the target position every one hundred time steps. Note that the targets do not cross at the same time.

taken at that time. Therefore, when the parameter N_{slot} is high, i.e., the sensors are very asynchronous, this algorithm has a high computational burden (compared to the asynchronous particle filter) because it propagates and updates the particles too many times. The synchronous PF is a suboptimal approach that does not take into account when the measurement was exactly taken. With all the binary measurements received from a sensor in an update interval, the synchronous PF applies an OR function and updates the PF as if all the measurements had been taken in the last slot, N_{slot} , also using IP. This kind of filter is regarded in [120] as a real-time particle filter because it lowers the computational burden.

The algorithms were implemented using 800 particles in Matlab using C MEX-files on a Pentium IV with a 2.4-GHz processor. In each Monte Carlo run, $\bar{\mathbf{p}}_i^0$ is drawn from a Gaussian distribution whose mean is the real position of target i at time 0 and whose covariance matrix is $\Sigma^0 = \sigma_0^2 \mathbf{I}_2$.

Three scenarios are considered, each one with different sensor sampling period parameters as shown in Table 7.3. Scenarios one and two have different sampling periods and scenario three includes a non-zero variance in the sampling period to model the slight differences in the internal clocks.

The RMS position error and the execution time with respect to N_{slot} using a Monte Carlo simulation with 50 runs are shown in Figures 7.4, 7.5 and 7.6. It can be seen that the RMS position error increases with N_{slot} . The higher N_{slot} is, the higher the sampling period is (see Table 7.3), the fewer number of measurements there are and, therefore, the error increases. As there are fewer measurements to process, the execution time decreases. When $N_{slot} = 2$, the RMS position error of the

Table 7.3: Sampling period parameters of the sensors in the scenarios

Scenario	Mean T_s	Variance σ_T^2
1	$N_{slot} \cdot \tau$	0
2	$0.6 \cdot N_{slot} \cdot \tau$	0
3	$0.6 \cdot N_{slot} \cdot \tau$	$(0.01 \cdot N_{slot} \cdot \tau)^2$

three algorithms tends to be the same because the targets hardly move in this short update interval and, then, it is not very important if a sensor samples in the first or in the second slot of the update interval. On the contrary, as N_{slot} increases, the synchronous PF has a much worse performance than the other two filters as it does not consider the exact time instant when each measurement was taken. Besides, the performance of the asynchronous PF is very close to the traditional PF even though it is not an optimal approach (as it does not take account of measurement correlations as pointed out in Appendix A.29).

Figures 7.4, 7.5 and 7.6 also indicate that the shorter the sampling period is, the higher the performance is, i.e., the RMS position errors in scenarios two and three are lower than in scenario one. This is because there are more measurements and thus more information to reduce the errors. In scenarios two and three, the performance is approximately the same as the variance of the internal clocks of the sensors is low and the average number of measurements in the simulation is the same. It is also important to note that when the sensors are very asynchronous, N_{slot} is high, the execution time for the asynchronous PF is much lower than the traditional PF with almost similar performance. For instance, in scenario one the asynchronous PF takes approximately 35% of the execution time of the traditional PF for $N_{slot} = 40$, see Figure 7.7. This is why this filter could be considered a real-time PF as it reduces considerably the computational burden while keeping high accuracy in the estimations.

The RMS position error and execution time for scenario one and $N_{slot} = 40$ for different number of particles are shown in Figure 7.8. The asynchronous PF provides slightly higher RMS position errors than the traditional PF regardless of the number of particles. Moreover, the use of the asynchronous PF implies a higher decrease in execution time with respect to the traditional PF for a higher number of particles.

7.6 Conclusions

A new algorithm has been developed for tracking with low computational burden which keeps low RMS position errors when the sensors are not synchronised. Even though the algorithm has been presented for a particular model and importance

Figure 7.4: Asynchronous PF. Performance in scenario 1: (a) RMS position error and (b) execution time. The performance of the asynchronous particle filter is very close to the traditional particle filter with much lower computational burden when N_{slot} is high enough.

Figure 7.5: Asynchronous PF. Performance in scenario 2: (a) RMS position error and (b) execution time. The performance of the asynchronous particle filter is very close to the traditional particle filter with much lower computational burden when N_{slot} is high enough.

Figure 7.6: Asynchronous PF. Performance in scenario 3: (a) RMS position error and (b) execution time. The performance of the asynchronous particle filter is very close to the traditional particle filter with much lower computational burden when N_{slot} is high enough.

Figure 7.7: Normalised execution time with respect to the execution time of the traditional particle filter for scenario one. When N_{slot} is high enough, the asynchronous filter has much lower computational burden than the traditional particle filter.

Figure 7.8: Asynchronous PF. Performance in scenario 1 with $N_{slot} = 40$ against the number of particles: (a) RMS position error and (b) execution time.

density, the results in Section 7.3 can be applied to any model and importance density. If the integrals that appear in the calculation of the asynchronous likelihood do not admit a closed-form solution, an approximation based on the unscented transformation can be used as was done in Section 6.4 to calculate the weights of the PP method in SLAM.

The asynchronous PF has been compared with two different solutions to the problem. The proposed algorithm has much better performance than a synchronous PF that applies an OR function to all the measurements received by every sensor in an update interval. Moreover, the RMS position error of the new filter is just slightly worse than the one of the traditional PF but reducing considerably the computational expense. These two characteristics make the proposed algorithm a good alternative for real time systems which need better performance than the synchronous PF.

It should be mentioned that it has been assumed that the number of targets is fixed and known. However, the asynchronous PF can be easily generalised for variable and unknown number of targets using the approach indicated in Chapter 4.

Chapter 8

Conclusions and future work

This chapter presents the conclusions and important directions of future work of this thesis.

8.1 Conclusions

The problem considered in this thesis is multiple target tracking using a wireless sensor network. Chapter 2 introduces the mathematical framework, Chapters 3-5 contain the development of general tools for nonlinear filtering and multiple target tracking. The general concepts and algorithms are employed for specific problems in wireless sensor networks in Chapters 6-7. These issues are discussed thoroughly in the following.

The first contribution of this thesis is given in Chapter 3. It presents the truncated Kalman filter which is a general algorithm (in the sense that it does not solve a particular problem of multiple target tracking or sensor networks) to approximate the first two moments of the posterior PDF for non-linear/non-Gaussian filtering problems in a computationally efficient manner. The truncated Kalman filter re-thinks the update phase of Kalman filtering for non-linear measurement functions and constitutes a brand new algorithm based on approximating the measurement noise support as a bounded one. All the examples given along this thesis show that this new way of approximating the posterior PDF is much more accurate than conventional Kalman-filter-type algorithms in a variety of situations.

The main contributions of Chapter 4 are a two-layer particle filter and the parallel partition method for tracking a variable and unknown number of targets. The two-layer particle filter uses a subsidiary particle filter to build the importance density for sampling newborn targets. The rationale behind this idea is to take the target

dynamics into account to create new tracks from a Bayesian point of view. The parallel partition method has been shown to outperform a lot of previously developed tracking algorithms when targets move in close proximity. The resulting algorithm, which combines the two-layer particle filter and the parallel partition method, is applied to an example of tracking using sensor networks. However, it is a general algorithm and its ideas can easily be applied to tracking using other systems such as radars.

Chapter 5 deals with two subjects related to Chapter 4. While Chapter 4 is concerned with the detection and tracking of a variable and unknown number of targets, Chapter 5 deals with the detection of a fixed number of targets in an area of interest. As it is assumed that the number of targets is fixed, a more restrictive algorithm than in Chapter 4 can be used such that detection performance is improved. This algorithm could be used to improve track initiation in Chapter 4. Moreover, in Chapter 4, it was explained the necessity of including track labels in the multitarget state to perform Bayesian tracking. However, it is in Chapter 5 where the extraction of target labelling information is addressed. Due to the difficulty of the problem, only a two-target case is presented although the methodology can be generalised for multiple targets in principle.

Chapter 6 presents an algorithm that simultaneously localises the sensors and estimates the target states. Therefore, it is the first algorithm of this thesis that only applies to multiple target tracking using sensor networks. It is based on the algorithms widely known in robotics as simultaneous localisation and mapping. In addition, the application of the truncated Kalman filter to this problem is key to attaining low estimation errors.

Chapter 7 deals with multiple target tracking using sensor networks in which the sensors are not synchronised. As the sensors do not measure at the same time, a particle filter based on approximating the optimal Bayesian filtering equations has a high computational burden. To lower the computational burden, the asynchronous particle filter is proposed. In the analysed scenario, its performance is almost equivalent to a conventional particle filter's performance with an important decrease in computational burden.

8.2 Future work

The directions of future work of this thesis are manifold. They are divided into three main groups: truncated Kalman filtering, detection and tracking of multiple targets in general and detection and tracking of multiple targets using wireless sensor networks.

8.2.1 Truncated Kalman filtering

The truncated Kalman filter opens up new vistas of research. There are many aspects of the algorithm that could be further investigated. For example, the current selection of α works very well but it does not meet any optimisation rule. A detailed study on how to select α could improve performance. Also, there are some approximations that could be changed. For example, instead of using a first-order Taylor series of the measurement function to approximate the first two moments of the truncated prior, one could linearise the measurement function by statistical linear regression. This could also improve performance.

In addition, the current form of the algorithm requires the measurement function to be bijective and the measurement noise to be additive. However, if these conditions were not met, the main idea of the truncated Kalman filter could still be used in principle by finding the regions of high likelihood. It should be noted that if there are several disconnected regions of high likelihood, the posterior PDF could be better approximated as a Gaussian mixture rather than a single Gaussian. How to do this generalisation in a computationally efficient way is worth studying.

8.2.2 Detection and tracking of multiple targets in general

The final aim is to develop an efficient algorithm that detects and tracks a variable and unknown number of targets making use of labelling information in an efficient manner. Firstly though, the following steps should be taken.

The algorithm to detect and track multiple targets of Chapter 4 creates and deletes tracks based on likelihood ratio tests independently for each track. While this approach has a low computational burden, it would be necessary to take all the neighbouring targets into account in scenarios where targets are expected to appear and disappear in clusters. Chapter 5 gives some indications about how to do it but its ideas have not been implemented in a multiple target detection and tracking problem with variable and unknown number of targets.

The exploitation of target labelling information addressed in Chapter 5 provides many interesting lines of future research. The proposed algorithm is based on a two-target case because of the difficulty of extracting labelling information from the posterior PDF. The main limitation to generalise this algorithm for multiple targets is the computational complexity. The high computational complexity in the implementation is mainly due to three reasons. First and foremost, a marginal particle filter is used. This considerably increases the computational load but it seems to be an essential part of the algorithm. Second, the current form of the algorithm

does not apply Rao-Blackwellisation to the target velocities. If Rao-Blackwellisation were applied, the number of particles and the computational burden would decrease. However, it is an open problem to determine if Rao-Blackwellisation can be applied to a marginal particle filter that samples particles and mirror particles. Third, the proposed algorithm does not use subparticle crossover so its generalisation to multiple targets will greatly suffer from the curse of dimensionality. As happens with Rao-Blackwellisation, this is an open problem as it is not clear that subparticle crossover can be applied. It would also be very interesting to check if when the number of particles is high enough, the probability of successful labelling calculated by the filter is close to the estimated probability of successful labelling obtained by Monte Carlo simulation. However, the measurement model in Chapter 5 is highly nonlinear and a very high number of particles would be needed to estimate the probability of successful labelling accurately. Consequently, it could be a good alternative to use measurement model with milder nonlinearities so that the required number of particles is lower to analyse this thoroughly.

Another line of research is to develop multiple target tracking algorithms with more realistic dynamic and measurement models. For example, measurement models could also take into account the shape of the target as in [70, 75]. These algorithms would share the important properties highlighted in Chapter 4, e.g., subparticle crossover.

8.2.3 Detection and tracking of multiple targets using wireless sensor networks

There are a lot of interesting directions of future work in this respect considering the introduction of this thesis. There, some of the most important characteristics of wireless sensor networks were highlighted. This thesis has focused on centralised algorithms. Therefore, a distributed implementation of the algorithms is of interest so that they can be applied in networks of this type. Moreover, more sophisticated sensors that allow the use of sensor management techniques have not been considered either. Sensor management algorithms could employ the tracking information provided by the proposed algorithms to make decisions as to what sensing mode should be used, where the sensor should move, what sensors should be activated or what power level or waveforms should be used.

As regards the centralised simultaneous localisation and mapping algorithm developed in Chapter 6, there are also several lines of improvement. The algorithm assumes that the aiming directions of the sensor antennas are known. This implies that the orientations of the sensors are known. It is worth studying if the problem

can be solved without this assumption but this might not be possible. A prior step could be to assume that the sensor positions are known but their orientations are not. This problem can be solved and can shed light on how to solve the more general problem. Moreover, the approximation to the posterior PDF assumed that the sensor positions were independent to save computation burden. Accounting for their possible correlations will create a better estimate of the map and improve tracking as well. Therefore, how to do it in a computationally efficient manner is well-worth exploring.

It would be interesting to generalise the asynchronous particle filter explained in Chapter 7 such that the sampling instants of the sensors are not perfectly known but modelled as random variables. This would add realism to the simulations.

Appendix A

Additional mathematical derivations

A.1 TKF analytical example calculations

In this appendix, the moments for the Kalman filter and truncated Kalman filter of the analytical example in Section 3.3.2 are given. Finally, the mean square error of the estimators are also given.

A.1.1 Kalman filter moments

The Kalman filter moments are

$$\begin{aligned}\hat{z}_0 = \mathbb{E}[z] &= \int ax^3 \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= a(\hat{x}_{p,0}^3 + 3\hat{x}_{p,0}\sigma_{p,0}^2)\end{aligned}\tag{A.1}$$

$$\begin{aligned}S_0 = \text{var}[z] &= \int \mathbb{E}[(z - \hat{z}_0)^2 | x] \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= \int \mathbb{E}\left[(ax^3 - \hat{z}_0)^2 + \eta^2\right] \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= \frac{\Delta^2}{12} + \int (a^2x^6 + \hat{z}_0^2 - 2ax^3\hat{z}_0) \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= \frac{\Delta^2}{12} + a^2(9\hat{x}_{p,0}^4\sigma_{p,0}^2 + 36\hat{x}_{p,0}^2\sigma_{p,0}^4 + 15\sigma_{p,0}^6)\end{aligned}\tag{A.2}$$

$$\begin{aligned}
 \Psi_0 = \text{cov}[x, z] &= \int \mathbb{E} \left[(x - \hat{x}_{p,0}) (ax^3 + \eta - \hat{z}_0) | x \right] \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\
 &= \int (x - \hat{x}_{p,0}) (ax^3 - \hat{z}_0) \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\
 &= 3a\sigma_{p,0}^2 (\hat{x}_{p,0}^2 + \sigma_{p,0}^2)
 \end{aligned} \tag{A.3}$$

A.1.2 Useful relationships

Before addressing the truncated Kalman filter moments, some prior calculations need to be done. First, $\lambda_i(a, b)$ is defined as

$$\lambda_i(a, b) = \frac{a^i \phi(a) - b^i \phi(b)}{\Phi(b) - \Phi(a)} \tag{A.4}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ are the PDF and cumulative distribution function, respectively, of a standard normal random variable.

Using integration by parts, it can be shown that

$$\frac{1}{C(c_1, c_2)} \int_{c_1}^{c_2} x \mathcal{N}(x; 0, 1) dx = \lambda_0(c_1, c_2) \tag{A.5}$$

$$\frac{1}{C(c_1, c_2)} \int_{c_1}^{c_2} x^2 \mathcal{N}(x; 0, 1) dx = \lambda_1(c_1, c_2) + 1 \tag{A.6}$$

$$\frac{1}{C(c_1, c_2)} \int_{c_1}^{c_2} x^3 \mathcal{N}(x; 0, 1) dx = \lambda_2(c_1, c_2) + 2\lambda_0(c_1, c_2) \tag{A.7}$$

$$\frac{1}{C(c_1, c_2)} \int_{c_1}^{c_2} x^4 \mathcal{N}(x; 0, 1) dx = \lambda_3(c_1, c_2) + 3(\lambda_1(c_1, c_2) + 1) \tag{A.8}$$

$$\frac{1}{C(c_1, c_2)} \int_{c_1}^{c_2} x^5 \mathcal{N}(x; 0, 1) dx = \lambda_4(c_1, c_2) + 4(\lambda_2(c_1, c_2) + 2\lambda_0(c_1, c_2)) \tag{A.9}$$

$$\frac{1}{C(c_1, c_2)} \int_{c_1}^{c_2} x^6 \mathcal{N}(x; 0, 1) dx = \lambda_5(c_1, c_2) + 5(\lambda_3(c_1, c_2) + 3(\lambda_1(c_1, c_2) + 1)) \tag{A.10}$$

where $C(c_1, c_2) = \int_{c_1}^{c_2} \mathcal{N}(x; 0, 1) dx$. These integrals are found in the derivation of the truncated Kalman filter moments.

A.1.3 Truncated Kalman filter moments

Let us denote $b_1 = \sqrt[3]{\frac{z-\Delta/2}{a}}$ and $b_2 = \sqrt[3]{\frac{z+\Delta/2}{a}}$ so that $I_x(z) = [b_1, b_2]$ and $c_i = (b_i - \hat{x}_{p,0})/\sigma_{p,0}$ for $i = 1, 2$. The mean and variance of the truncated prior are

$$\hat{x}_{p,1} = \frac{1}{\varepsilon_1} \int_{b_1}^{b_2} x \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx$$

$$=\hat{x}_{p,0} + \sigma_{p,0}\lambda_0(c_1, c_2) \quad (\text{A.11})$$

$$\begin{aligned} \sigma_{p,1}^2 &= \frac{1}{\varepsilon_1} \int_{b_1}^{b_2} (x - \hat{x}_{p,1})^2 \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= \sigma_{p,0}^2 \left(1 - \lambda_0^2(c_1, c_2) + \lambda_1(c_1, c_2) \right) \end{aligned}$$

In the following, the moments of the Kalman filter applied to $p_1(\cdot)$ are calculated.

$$\begin{aligned} \hat{z}_1 &= \frac{1}{\varepsilon_1} \int_{b_1}^{b_2} ax^3 \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= \frac{1}{\varepsilon_1} \int_{c_1}^{c_2} a(\hat{x}_{p,0} + \sigma_{p,0}y)^3 \mathcal{N}(y; 0, 1) dy \\ &= a \left[\hat{x}_{p,0}^3 + 3\hat{x}_{p,0}^2\sigma_{p,0}\lambda_0(c_1, c_2) + 3\hat{x}_{p,0}\sigma_{p,0}^2(\lambda_1(c_1, c_2) + 1) \right. \\ &\quad \left. + \sigma_{p,0}^3(\lambda_2(c_1, c_2) + 2\lambda_0(c_1, c_2)) \right] \end{aligned} \quad (\text{A.12})$$

where it should be noted that $\varepsilon_1 = C(c_1, c_2)$.

$$\begin{aligned} S_1 &= \frac{\Delta^2}{12} + \frac{1}{\varepsilon_1} \int_{b_1}^{b_2} (ax^3 - \hat{z}_1)^2 \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= \frac{\Delta^2}{12} - \hat{z}_1^2 + a^2 \left[\hat{x}_{p,0}^6 + 6\hat{x}_{p,0}^5\sigma_{p,0}\lambda_0(c_1, c_2) + 15\hat{x}_{p,0}^4\sigma_{p,0}^2(\lambda_1(c_1, c_2) + 1) \right. \\ &\quad + 20\hat{x}_{p,0}^3\sigma_{p,0}^3(\lambda_2(c_1, c_2) + 2\lambda_0(c_1, c_2)) \\ &\quad + 15\hat{x}_{p,0}^2\sigma_{p,0}^4(\lambda_3(c_1, c_2) + 3(\lambda_1(c_1, c_2) + 1)) \\ &\quad + 6\hat{x}_{p,0}\sigma_{p,0}^5(\lambda_4(c_1, c_2) + 4(\lambda_2(c_1, c_2) + 2\lambda_0(c_1, c_2))) \\ &\quad \left. + \sigma_{p,0}^6(\lambda_5(c_1, c_2) + 5(\lambda_3(c_1, c_2) + 3(\lambda_1(c_1, c_2) + 1))) \right] \end{aligned} \quad (\text{A.13})$$

$$\begin{aligned} \Psi_1 &= \frac{1}{\varepsilon_1} \int_{b_1}^{b_2} (x - \hat{x}_{p,1}) (ax^3 - \hat{z}_1) \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= -\hat{x}_{p,1}\hat{z}_1 + a \left[\hat{x}_{p,0}^4 + 4\hat{x}_{p,0}^3\sigma_{p,0}\lambda_0(c_1, c_2) \right. \\ &\quad + 6\hat{x}_{p,0}^2\sigma_{p,0}^2(\lambda_1(c_1, c_2) + 1) + 4\hat{x}_{p,0}\sigma_{p,0}^3(\lambda_2(c_1, c_2) + 2\lambda_0(c_1, c_2)) \\ &\quad \left. + \sigma_{p,0}^4(\lambda_3(c_1, c_2) + 3(\lambda_1(c_1, c_2) + 1)) \right] \end{aligned} \quad (\text{A.14})$$

A.1.4 Mean square errors of the estimators

The updated mean given by the Kalman filter applied to $p_0(\cdot)$ is:

$$\begin{aligned}\hat{x}_{u,0} &= \hat{x}_{p,0} + \Psi_0/S_0(z - \hat{z}_0) \\ &= \hat{x}_{p,0} + \xi_0\end{aligned}\tag{A.15}$$

where $\xi_0 = \Psi_0/S_0(z - \hat{z}_0)$.

The updated mean given by the Kalman filter applied to $p_1(\cdot)$ is

$$\begin{aligned}\hat{x}_{u,1} &= \hat{x}_{p,1} + \Psi_1/S_1(z - \hat{z}_1) \\ &= \hat{x}_{p,0} + \sigma_{p,0}\lambda_0(c_1, c_2) + \Psi_1/S_1(z - \hat{z}_1) \\ &= \hat{x}_{p,0} + \xi_1\end{aligned}\tag{A.16}$$

where $\xi_1 = \sigma_{p,0}\lambda_0(c_1, c_2) + \Psi_1/S_1(z - \hat{z}_1)$.

The updated mean of the truncated Kalman filter is

$$\begin{aligned}\hat{x}_{u,2} &= \alpha\hat{x}_{u,1} + (1 - \alpha)\hat{x}_{u,0} \\ &= \hat{x}_{p,0} + \alpha\xi_1 + (1 - \alpha)\xi_0 \\ &= \hat{x}_{p,0} + \xi_2\end{aligned}\tag{A.17}$$

where $\xi_2 = \alpha\xi_1 + (1 - \alpha)\xi_0$.

The mean square error of the estimator $\hat{x}_{u,i}$ $i \in \{1, 2, 3\}$ conditional on a particular measurement is

$$\begin{aligned}\text{mse}(\hat{x}_{u,i}|z) &= \frac{1}{\varepsilon_1} \int_{b_1}^{b_2} (\hat{x}_{u,i} - x)^2 \mathcal{N}(x; \hat{x}_{p,0}, \sigma_{p,0}^2) dx \\ &= \frac{1}{\varepsilon_1} \int_{c_1}^{c_2} (\xi_i - \sigma_{p,0}y)^2 \mathcal{N}(y; 0, 1) dy \\ &= \xi_i^2 + \sigma_{p,0}^2(\lambda_1(c_1, c_2) + 1) - 2\xi_i\sigma_{p,0}\lambda_0(c_1, c_2) \\ &= \sigma_{p,0}^2(\lambda_1(c_1, c_2) + 1 - \lambda_0^2(c_1, c_2)) + (\xi_i - \sigma_{p,0}\lambda_0(c_1, c_2))^2 \\ &= \sigma_{p,1}^2 + (\xi_i - \sigma_{p,0}\lambda_0(c_1, c_2))^2\end{aligned}\tag{A.18}$$

A.2 TKF with unbounded noise support: Calculations using AP1 and AP2

In this appendix, the TKF moments when AP1 and AP2 are used to approximate the truncated prior $p_1(\cdot)$ in the example of Section 3.4.2 are given. Let us assume a measurement z has been received. Then, the value that maximises the likelihood is

$$\tilde{x}(z) = \sqrt[3]{z/a} \quad (\text{A.19})$$

Using AP1, the measurement equation $h(x)$ becomes

$$h(x) = z + 3a(\tilde{x}(z))^2(x - \tilde{x}(z)) \quad (\text{A.20})$$

The support of the truncated Gaussian measurement noise PDF is $[-\Delta/2, \Delta/2]$ where $\Delta = 2\sqrt{-2R \ln \Gamma_\eta}$ and Γ_η comes from¹ (3.23). The mean of the truncated noise is zero and its variance is

$$R_T = R \left[1 - \frac{\frac{\Delta}{\sqrt{R}} \phi\left(\frac{\Delta}{2\sqrt{R}}\right)}{2\Phi\left(\frac{\Delta}{2\sqrt{R}}\right) - 1} \right] \quad (\text{A.21})$$

Using (A.20), the support $[b_1, b_2]$ of the truncated prior $p_1(\cdot)$ is

$$b_1 = \sqrt[3]{z/a} - \frac{1}{3a(\tilde{x}(z))^2} \frac{\Delta}{2} \quad (\text{A.22})$$

$$b_2 = \sqrt[3]{z/a} + \frac{1}{3a(\tilde{x}(z))^2} \frac{\Delta}{2} \quad (\text{A.23})$$

Using AP2, $p_1(\cdot)$ is a uniform PDF in the interval $[b_1, b_2]$, whose first two moments are

$$\hat{x}_{p,1} = \frac{b_1 + b_2}{2} \quad (\text{A.24})$$

$$\sigma_{p,1}^2 = \frac{(b_2 - b_1)^2}{12} \quad (\text{A.25})$$

The TKF moments are

$$\begin{aligned} \hat{z}_1 &= \frac{1}{b_2 - b_1} \int_{b_1}^{b_2} ax^3 dx \\ &= \frac{a(b_2^4 - b_1^4)}{4(b_2 - b_1)} \end{aligned} \quad (\text{A.26})$$

¹The threshold Γ_η has been normalised with the respect to the maximum of the PDF of the original measurement noise (without truncation).

$$\begin{aligned}
 S_1 &= R_T + \frac{1}{b_2 - b_1} \int_{b_1}^{b_2} (ax^3 - \hat{z}_1)^2 dx \\
 &= R_T + \frac{1}{b_2 - b_1} \left[a^2 \frac{b_2^7 - b_1^7}{7} + \hat{z}_1^2 (b_2 - b_1) - 2a\hat{z}_1 \frac{b_2^4 - b_1^4}{4} \right] \quad (\text{A.27})
 \end{aligned}$$

$$\begin{aligned}
 \Psi_1 &= \frac{1}{b_2 - b_1} \int_{b_1}^{b_2} (x - \hat{x}_{p,1}) (ax^3 - \hat{z}_1) dx \\
 &= \frac{1}{b_2 - b_1} \left[a \frac{b_2^5 - b_1^5}{5} - \hat{z}_1 \frac{b_2^2 - b_1^2}{2} - a\hat{x}_{p,1} \frac{b_2^4 - b_1^4}{4} + \hat{x}_{p,1} \hat{z}_1 (b_2 - b_1) \right] \quad (\text{A.28})
 \end{aligned}$$

A.3 Truncated Kalman filter proof

In this appendix, it is shown that Bayes' rule applied to $p_2(\mathbf{x}; \mathbf{z})$ is the same as applied to $p_0(\mathbf{x})$ in the conditions indicated in Section 3.3. Bayes' rule applied to $p_2(\mathbf{x}; \mathbf{z})$, denoted as $\pi(\mathbf{x} | \mathbf{z})$, is

$$\pi(\mathbf{x} | \mathbf{z}) \propto p(\mathbf{z} | \mathbf{x}) p_2(\mathbf{x}; \mathbf{z}) \quad (\text{A.29})$$

Substituting (3.14), (3.24) and (3.20) into (A.29):

$$\begin{aligned}
 \pi(\mathbf{x} | \mathbf{z}) &\propto p_{\boldsymbol{\eta}}(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}(\mathbf{z})}}(\mathbf{x}) \cdot \\
 &\quad \left[\alpha_0 \frac{1}{\varepsilon_1} p_0(\mathbf{x}) \chi_{I_{\mathbf{x}(\mathbf{z})}}(\mathbf{x}) + (1 - \alpha_0) p_0(\mathbf{x}) \right] \quad (\text{A.30})
 \end{aligned}$$

$$= p_{\boldsymbol{\eta}}(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}(\mathbf{z})}}(\mathbf{x}) p_0(\mathbf{x}) \left(\alpha_0 \frac{1}{\varepsilon_1} + 1 - \alpha_0 \right) \quad (\text{A.31})$$

Using (3.14) in (A.31)

$$\begin{aligned}
 \pi(\mathbf{x} | \mathbf{z}) &\propto p(\mathbf{z} | \mathbf{x}) p_0(\mathbf{x}) \left(\alpha_0 \frac{1}{\varepsilon_1} + 1 - \alpha_0 \right) \\
 &\propto p(\mathbf{z} | \mathbf{x}) p_0(\mathbf{x}) \quad (\text{A.32})
 \end{aligned}$$

Therefore, (A.32) proves that Bayes' rule remains unaltered if $p_2(\mathbf{x}; \mathbf{z})$ is the prior instead of $p_0(\mathbf{x})$.

A.4 TKF: Application of two Kalman filters to the prior mixture

Using Bayes' rule and the mixture as the prior, see (3.24), the posterior is

$$p(\mathbf{x}|\mathbf{z}) = \frac{p(\mathbf{z}|\mathbf{x})(\alpha_0 p_1(\mathbf{x};\mathbf{z}) + (1 - \alpha_0) p_0(\mathbf{x}))}{\int p(\mathbf{z}|\mathbf{x})(\alpha_0 p_1(\mathbf{x};\mathbf{z}) + (1 - \alpha_0) p_0(\mathbf{x})) d\mathbf{x}} \quad (\text{A.33})$$

Using equations (3.20) and (3.15), Bayes' rule becomes

$$\begin{aligned} p(\mathbf{x}|\mathbf{z}) &= \frac{p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) \left(\alpha_0 \frac{1}{\varepsilon_1} p_0(\mathbf{x}) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) + (1 - \alpha_0) p_0(\mathbf{x}) \right)}{\int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) \left(\alpha_0 \frac{1}{\varepsilon_1} p_0(\mathbf{x}) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) + (1 - \alpha_0) p_0(\mathbf{x}) \right) d\mathbf{x}} \quad (\text{A.34}) \\ p(\mathbf{x}|\mathbf{z}) &= \frac{\frac{\alpha_0}{\varepsilon_1} p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) + (1 - \alpha_0) p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x})}{\left(\frac{\alpha_0}{\varepsilon_1} + (1 - \alpha_0) \right) \int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) p_0(\mathbf{x}) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) d\mathbf{x}} \quad (\text{A.35}) \end{aligned}$$

Normalising each term in the numerator

$$\begin{aligned} p(\mathbf{x}|\mathbf{z}) &= \frac{\frac{\alpha_0}{\varepsilon_1} \frac{p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x})}{\int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) d\mathbf{x}} \int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) d\mathbf{x}}{\left(\frac{\alpha_0}{\varepsilon_1} + (1 - \alpha_0) \right) \int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) p_0(\mathbf{x}) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) d\mathbf{x}} \\ &+ \frac{(1 - \alpha_0) \frac{p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x})}{\int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) d\mathbf{x}} \int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) d\mathbf{x}}{\left(\frac{\alpha_0}{\varepsilon_1} + (1 - \alpha_0) \right) \int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) p_0(\mathbf{x}) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) d\mathbf{x}} \quad (\text{A.36}) \end{aligned}$$

Simplifying with the denominator

$$p(\mathbf{x}|\mathbf{z}) = \frac{\frac{\alpha_0}{\varepsilon_1} \frac{p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) / \varepsilon_1}{\int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) d\mathbf{x} / \varepsilon_1} + (1 - \alpha_0) \frac{p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x})}{\int p_\eta(\mathbf{z} - \mathbf{h}(\mathbf{a})) \chi_{I_{\mathbf{x}}(\mathbf{z})}(\mathbf{x}) p_0(\mathbf{x}) d\mathbf{x}}}{\left(\frac{\alpha_0}{\varepsilon_1} + (1 - \alpha_0) \right)} \quad (\text{A.37})$$

Using again equations (3.20) and (3.15):

$$p(\mathbf{x}|\mathbf{z}) = \frac{\frac{\alpha_0}{\varepsilon_1} \frac{p(\mathbf{z}|\mathbf{x}) p_1(\mathbf{x};\mathbf{z})}{\int p(\mathbf{z}|\mathbf{x}) p_1(\mathbf{x};\mathbf{z}) d\mathbf{x}} + (1 - \alpha_0) \frac{p(\mathbf{z}|\mathbf{x}) p_0(\mathbf{x})}{\int p(\mathbf{z}|\mathbf{x}) p_0(\mathbf{x}) d\mathbf{x}}}{\left(\frac{\alpha_0}{\varepsilon_1} + (1 - \alpha_0) \right)} \quad (\text{A.38})$$

The first term in the numerator without $\frac{\alpha_0}{\varepsilon_1}$ is Bayes' rule applied to $p_1(\mathbf{x};\mathbf{z})$ and the second term without $(1 - \alpha_0)$ is Bayes' rule applied to $p_0(\mathbf{x};\mathbf{z})$. These terms

can be approximated using Kalman filters. Therefore, the output of the Kalman filter applied to $p_1(\mathbf{x}; \mathbf{z})$ has a weight in the final mixture given by

$$\alpha = \frac{\frac{\alpha_0}{\varepsilon_1}}{\frac{\alpha_0}{\varepsilon_1} + (1 - \alpha_0)} = \frac{\alpha_0}{\alpha_0 + (1 - \alpha_0) \varepsilon_1} \quad (\text{A.39})$$

and the equations for approximating the first two moments of the updated mixture are

$$\hat{\mathbf{x}}_{u,2} = \alpha \hat{\mathbf{x}}_{u,1} + (1 - \alpha) \hat{\mathbf{x}}_{u,0} \quad (\text{A.40})$$

$$\begin{aligned} \mathbf{P}_{u,2} = & \alpha \left[\mathbf{P}_{u,1} + (\hat{\mathbf{x}}_{u,1} - \hat{\mathbf{x}}_{u,2}) (\hat{\mathbf{x}}_{u,1} - \hat{\mathbf{x}}_{u,2})^T \right] \\ & + (1 - \alpha) \left[\mathbf{P}_{u,0} + (\hat{\mathbf{x}}_{u,0} - \hat{\mathbf{x}}_{u,2}) (\hat{\mathbf{x}}_{u,0} - \hat{\mathbf{x}}_{u,2})^T \right] \end{aligned} \quad (\text{A.41})$$

It should be noted that

$$\varepsilon_1 = \int p_0(\mathbf{x}) \chi_{I_{\mathbf{z}}}(\mathbf{x}) d\mathbf{x} \quad (\text{A.42})$$

This implies that, for a given measurement, ε_1 is a fixed parameter that belongs to $[0, 1]$. The parameter α_0 can be chosen freely as long as $\alpha_0 \in [0, 1]$. Therefore, using (A.39), it can be shown that α can also be chosen freely as long as $\alpha \in [0, 1]$.

A.5 TUKF: First two moments of the truncated prior (I)

In this appendix, the mean $\boldsymbol{\mu}_{a,1}$ and covariance matrix $\boldsymbol{\Sigma}_{a,1}$ of $p_1(\mathbf{a}; \mathbf{z})$, which is given by (3.33), are calculated. Firstly, ε_2 is calculated:

$$\varepsilon_2 = \int \chi_{I_{\mathbf{a}}(\mathbf{z})}(\mathbf{a}) d\mathbf{a} \quad (\text{A.43})$$

Changing the variable inside the integral using

$$\mathbf{a} = \tilde{\mathbf{a}}(\mathbf{z}) - \tilde{\mathbf{H}}^{-1} \mathbf{u} \quad (\text{A.44})$$

one gets

$$\varepsilon_2 = \int \chi_{I_{\mathbf{a}}(\mathbf{z})}(\tilde{\mathbf{a}}(\mathbf{z}) - \tilde{\mathbf{H}}^{-1} \mathbf{u}) \left| \det(\tilde{\mathbf{H}}^{-1}) \right| d\mathbf{u} \quad (\text{A.45})$$

where $\left| \det(\tilde{\mathbf{H}}^{-1}) \right|$ is the absolute value of the determinant of $\tilde{\mathbf{H}}^{-1}$.

Using (3.30), (A.45) becomes

$$\varepsilon_2 = \left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| \int \chi_{I_\eta}(\mathbf{u}) \, d\mathbf{u} = \left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| |I_\eta| \quad (\text{A.46})$$

where $|I_\eta|$ indicates the area of the support of the measurement noise.

The mean of $p_1(\mathbf{a}; \mathbf{z})$ is

$$\boldsymbol{\mu}_{a,1} = \frac{1}{\left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| |I_\eta|} \int \mathbf{a} \chi_{I_{\mathbf{a}}(\mathbf{z})}(\mathbf{a}) \, d\mathbf{a} \quad (\text{A.47})$$

Using the change of variables of (A.44):

$$\boldsymbol{\mu}_{a,1} = \frac{1}{\left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| |I_\eta|} \int \left(\tilde{\mathbf{a}}(\mathbf{z}) - \tilde{\mathbf{H}}^{-1} \mathbf{u} \right) \chi_{I_\eta}(\mathbf{u}) \left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| d\mathbf{u} \quad (\text{A.48})$$

Using the fact that the mean of the measurement noise is zero and using Approximation AP1 (the measurement noise PDF is uniform in region I_η):

$$\frac{1}{|I_\eta|} \int \mathbf{u} \chi_{I_\eta}(\mathbf{u}) \, d\mathbf{u} = 0 \quad (\text{A.49})$$

equation (A.48) becomes

$$\boldsymbol{\mu}_{a,1} = \tilde{\mathbf{a}}(\mathbf{z}) \quad (\text{A.50})$$

Using the same procedure, the covariance matrix of $p_1(\mathbf{a}; \mathbf{z})$ is

$$\begin{aligned} \boldsymbol{\Sigma}_{a,1} &= \frac{1}{\left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| |I_\eta|} \int (\mathbf{a} - \tilde{\mathbf{a}}(\mathbf{z})) (\mathbf{a} - \tilde{\mathbf{a}}(\mathbf{z}))^T \chi_{I_{\mathbf{a}}(\mathbf{z})}(\mathbf{a}) \, d\mathbf{a} \\ &= \frac{1}{\left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| |I_\eta|} \int \left(\tilde{\mathbf{H}}^{-1} \mathbf{u} \right) \left(\tilde{\mathbf{H}}^{-1} \mathbf{u} \right)^T \chi_{I_\eta}(\mathbf{u}) \left| \det \left(\tilde{\mathbf{H}}^{-1} \right) \right| d\mathbf{u} \end{aligned} \quad (\text{A.51})$$

In Section 3.4.3, the variance of the measurement noise is represented by $\mathbf{R}$. Then, (A.51) can be written as

$$\boldsymbol{\Sigma}_{a,1} = \tilde{\mathbf{H}}^{-1} \mathbf{R} \left(\tilde{\mathbf{H}}^{-1} \right)^T \quad (\text{A.52})$$

A.6 TUKF: First two moments of the truncated prior (II)

In this appendix, the first two moments of $p_1(\mathbf{a}, \mathbf{b}; \mathbf{z})$ are calculated considering the first two moments of $p_0(\mathbf{a}, \mathbf{b})$, given by (3.36) and (3.37), and the first two moments of $p_1(\mathbf{a}; \mathbf{z})$, given by (3.34) and (3.35), are known. Firstly, it should be noted that the truncation only affects the part of the PDF that corresponds with $\mathbf{a}$, see (3.30). Then, this implies that the conditional PDF of $\mathbf{b}$ given $\mathbf{a}$ is not affected by the truncation. To emphasise this, one can write:

$$p_1(\mathbf{a}, \mathbf{b}; \mathbf{z}) = p_1(\mathbf{a}; \mathbf{z}) p_0(\mathbf{b} | \mathbf{a}) \quad (\text{A.53})$$

Assuming that $p_0(\mathbf{a}, \mathbf{b})$ is Gaussian [13]

$$\mathbf{v}_b(\mathbf{a}) \triangleq \mathbb{E}[\mathbf{b} | \mathbf{a}] = \boldsymbol{\mu}_{b,0} + \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} (\mathbf{a} - \boldsymbol{\mu}_{a,0}) \quad (\text{A.54})$$

$$\boldsymbol{\Gamma} \triangleq \text{cov}[\mathbf{b} | \mathbf{a}] = \boldsymbol{\Sigma}_{b,0} - \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} \boldsymbol{\Sigma}_{ab,0} \quad (\text{A.55})$$

Now, the first two moments of $p_1(\mathbf{a}, \mathbf{b}; \mathbf{z})$ given by the partition of (3.38) and (3.39) are calculated. The mean $\boldsymbol{\mu}_{b,1}$:

$$\begin{aligned} \boldsymbol{\mu}_{b,1} &= \int \mathbf{b} p_1(\mathbf{a}, \mathbf{b}; \mathbf{z}) \, d\mathbf{a} d\mathbf{b} = \int \mathbf{b} p_0(\mathbf{b} | \mathbf{a}) p_1(\mathbf{a}; \mathbf{z}) \, d\mathbf{a} d\mathbf{b} \\ &= \int \mathbb{E}[\mathbf{b} | \mathbf{a}] p_1(\mathbf{a}; \mathbf{z}) \, d\mathbf{a} \end{aligned} \quad (\text{A.56})$$

Using (A.54) and (3.34) in (A.56):

$$\boldsymbol{\mu}_{b,1} = \boldsymbol{\mu}_{b,0} + \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} (\boldsymbol{\mu}_{a,1} - \boldsymbol{\mu}_{a,0}) \quad (\text{A.57})$$

The cross-covariance $\boldsymbol{\Sigma}_{ab,1}$:

$$\boldsymbol{\Sigma}_{ab,1} = \int (\mathbf{a} - \boldsymbol{\mu}_{a,1}) (\mathbf{b} - \boldsymbol{\mu}_{b,1})^T p_0(\mathbf{b} | \mathbf{a}) p_1(\mathbf{a}; \mathbf{z}) \, d\mathbf{a} d\mathbf{b} \quad (\text{A.58})$$

Using (A.54) in (A.58):

$$\boldsymbol{\Sigma}_{ab,1} = \int (\mathbf{a} - \boldsymbol{\mu}_{a,1}) \left(\boldsymbol{\mu}_{b,0} + \boldsymbol{\Sigma}_{ab,0}^T \boldsymbol{\Sigma}_{a,0}^{-1} (\mathbf{a} - \boldsymbol{\mu}_{a,0}) - \boldsymbol{\mu}_{b,1} \right)^T p_1(\mathbf{a}; \mathbf{z}) \, d\mathbf{a} \quad (\text{A.59})$$

Expanding the terms in (A.58) and using (3.35)

$$\Sigma_{ab,1} = \Sigma_{a,1} \left(\Sigma_{a,0}^{-1} \right)^T \Sigma_{ab,0} \quad (\text{A.60})$$

Lastly, using a similar procedure, the covariance $\Sigma_{b,1}$ can be calculated and is given by (3.42).

A.7 SP-TUKF: Approximation of the first two moments of $p_2(\cdot)$

This appendix clarifies how the first two moments of $p_2(\cdot)$ are approximated for the SP-TUKF, which is explained in Section 3.4.3.4. Assuming $(\mathcal{A}_2^1, \dots, \mathcal{A}_2^{N_s+1})$ and $(w_2^1, \dots, w_2^{N_s+1})$ are known, one gets

$$\mu_{a,2} = \sum_{i=1}^{N_s+1} w_2^i \mathcal{A}_2^i \quad (\text{A.61})$$

$$\Sigma_{a,2} = \sum_{i=1}^{N_s+1} w_2^i \left(\mathcal{A}_2^i - \mu_{a,2} \right)^T \left(\mathcal{A}_2^i - \mu_{a,2} \right) \quad (\text{A.62})$$

A.8 TUKF: First two moments of $p_1(\cdot)$ in the numerical examples

In this appendix, the approximated first two moments of $p_1(\cdot)$ are given explicitly for the examples in Section 3.5.

A.8.1 Range-bearing case

The approximated first two moments of $p_1(\cdot)$ are

$$\begin{aligned} \mu_{a,1} &= \tilde{\mathbf{a}}(\mathbf{z}) \\ &= [x_a, y_a]^T \\ &= [z_r \cos z_\theta, z_r \sin z_\theta]^T \\ \Sigma_{a,1} &= \begin{pmatrix} \frac{x_a^2 \sigma_r^2}{x_a^2 + y_a^2} + \sigma_\theta^2 y_a^2 & \frac{x_a y_a \sigma_r^2}{x_a^2 + y_a^2} - \sigma_\theta^2 x_a y_a \\ \frac{x_a y_a \sigma_r^2}{x_a^2 + y_a^2} - \sigma_\theta^2 x_a y_a & \frac{y_a^2 \sigma_r^2}{x_a^2 + y_a^2} + \sigma_\theta^2 x_a^2 \end{pmatrix} \end{aligned} \quad (\text{A.63})$$

A.8.2 One-dimensional update case

The approximated first two moments of $p_1(\cdot)$ are

$$\mu_{x,1} = \tilde{x}(z) = \sqrt[n]{\frac{A_0}{z}} d_0 \quad (\text{A.64})$$

$$\Sigma_{x,1} = \frac{\tilde{x}(z)^{2n-2} R}{A_0^2 d_0^{2n} n^2} \quad (\text{A.65})$$

A.8.3 Univariate nonstationary growth model

The approximated first two moments of $p_1(\cdot)$, assuming the measurement is $z = ax^3$, are

$$\mu_{x,1} = \tilde{x}(z) = \sqrt[3]{z/a} \quad (\text{A.66})$$

$$\Sigma_{x,1} = \frac{R}{9a^2 \tilde{x}(z)^4} \quad (\text{A.67})$$

A.9 Kullback-Leibler divergence

The Kullback Leibler divergence for the PDFs $p(\cdot)$ and $q(\cdot)$ is defined as

$$D_{KL}(p\|q) = \int_{-\infty}^{\infty} p(\mathbf{x}) \ln \frac{p(\mathbf{x})}{q(\mathbf{x})} d\mathbf{x} \quad (\text{A.68})$$

If the PDFs are Gaussian

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{x}_p, \Sigma_p) \quad (\text{A.69})$$

$$q(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \mathbf{x}_q, \Sigma_q) \quad (\text{A.70})$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ then [90]

$$D_{KL}(p\|q) = \frac{1}{2} \left[\text{tr} \left(\Sigma_q^{-1} \Sigma_p \right) + (\mathbf{x}_q - \mathbf{x}_p)^T \Sigma_q^{-1} (\mathbf{x}_q - \mathbf{x}_p) - \ln \left(\frac{|\Sigma_p|}{|\Sigma_q|} \right) - n_x \right] \quad (\text{A.71})$$

It should be mentioned that in Section 3.5, when the Kullback-Leibler divergence between the KF approximations and Bayes' rule is approximated, $p(\cdot)$ is approximated by Bayes' rule and $q(\cdot)$ is the KF approximation output.

A.10 Multitarget state explanation

According to Chapter 4, the variable that represents the multitarget state

$$(\mathbf{R}^k, t^k) \in \bigcup_{t=0}^{\infty} (\mathbb{R}^4 \times \mathbb{N})^t \times \{t\} : i_1^k < i_2^k < \dots < i_t^k \quad (\text{A.72})$$

In this appendix, this is clarified with an example in which the maximum number of targets is 2. Then, there can be zero, one or two targets and the dimensionality of $\mathbf{R}^k$ is controlled by the value of t^k . Therefore:

$$\begin{aligned} (\mathbf{R}^k, t^k) &\in (\theta \times \theta \times \{0\}) \\ &\cup (\mathbb{R}^4 \times \mathbb{N} \times \{1\}) \\ &\cup (\mathbb{R}^4 \times \mathbb{N} \times \mathbb{R}^4 \times \mathbb{N} \times \{2\}) \end{aligned} \quad (\text{A.73})$$

Generalising (A.73) for infinite possible targets results in (A.72).

A.11 Set and vector notation equivalence for the labels

In Section 4.2, it is indicated that the JMKPD representation of the form $p(\mathbf{X}^k, I^k | \mathbf{z}^{1:k})$ is analogue to the complete vector representation $p\left(\left[\left(\mathbf{X}^k\right)^T, \left(\mathbf{I}^k\right)^T\right]^T, t^k | \mathbf{z}^{1:k}\right)$ and that $p(\mathbf{X}^k, I^k | \mathbf{X}^{k-1}, I^{k-1})$ and $p(\mathbf{z}^k | \mathbf{X}^k, I^k)$ admit a vector representation. The only reason to represent the PDFs with the variables $(\mathbf{X}^k, I^k)$ instead of $\left(\left[\left(\mathbf{X}^k\right)^T, \left(\mathbf{I}^k\right)^T\right]^T, t^k\right)$ is for simplicity. The likelihood has the same form with both representations. In this appendix, the equivalent to $p(\mathbf{X}^k, I^k | \mathbf{X}^{k-1}, I^{k-1})$ in vector representation is given. Note that the same could be done for a random set representation because of the one-to-one correspondence between the elements.

The transitional prior is given by (4.6)

$$\begin{aligned} p(\mathbf{X}^{k+1}, I^{k+1} | \mathbf{X}^k, I^k) &= p(N^{k+1}) p(S^k | I^k) p(\mathbf{X}^{k+1} (N^{k+1}) | N^{k+1}) \\ &\quad \cdot p(\mathbf{X}^{k+1} (S^k) | \mathbf{X}^k, S^k) \end{aligned} \quad (\text{A.74})$$

Therefore, each one of the terms in (A.74) should be given in vector notation. Recall that

$$N^{k+1} = \{n_1^{k+1}, n_2^{k+1}, \dots, n_{b^{k+1}}^{k+1}\} \quad (\text{A.75})$$

$$S^k = \{s_1^k, s_2^k, \dots, s_{t^k-d^k}^k\} \quad (\text{A.76})$$

The following vectors are defined

$$\mathbf{N}^{k+1} = [n_1^{k+1}, n_2^{k+1}, \dots, n_{b^{k+1}}^{k+1}]^T \quad (\text{A.77})$$

$$\mathbf{S}^k = [s_1^k, s_2^k, \dots, s_{t^k-d^k}^k]^T \quad (\text{A.78})$$

where it is assumed that $n_1^{k+1} < n_2^{k+1} < \dots < n_{b^{k+1}}^{k+1}$ and $s_1^k < s_2^k < \dots < s_{t^k-d^k}^k$ without loss of generality as this ordering is always possible because the labels are unique, see Section 4.2. Note that there are bijections between N^{k+1} and $\mathbf{N}^{k+1}$ and, S^k and $\mathbf{S}^k$, and I^{k+1} and $\mathbf{I}^{k+1} = \left[(\mathbf{S}^k)^T, (\mathbf{N}^{k+1})^T \right]^T$.

The transitional prior in vector notation can be written as

$$\begin{aligned} p(\mathbf{X}^{k+1}, \mathbf{I}^{k+1}, t^{k+1} | \mathbf{X}^k, \mathbf{I}^k, t^k) = & p(\mathbf{N}^{k+1}, b^{k+1}) p(\mathbf{S}^k, t^k - d^k | \mathbf{I}^k, t^k) \\ & \cdot p(\mathbf{X}^{k+1} | \mathbf{N}^{k+1}) p(\mathbf{N}^{k+1}, b^{k+1}) \\ & \cdot p(\mathbf{X}^{k+1} | \mathbf{S}^k) p(\mathbf{X}^k, t^k, \mathbf{S}^k, t^k - d^k) \end{aligned} \quad (\text{A.79})$$

Now, all the terms in (A.79) are given. The first one is

$$p(\mathbf{N}^{k+1}, b^{k+1}) = p(N^{k+1}) \quad (\text{A.80})$$

where $p(N^{k+1})$ is given by (4.7). Therefore, because of the one-to-one correspondence in a countable domain, the vector is transformed into the set and the equivalence is obtained. The same happens for $p(\mathbf{S}^k, t^k - d^k | \mathbf{I}^k, t^k)$:

$$p(\mathbf{S}^k, t^k - d^k | \mathbf{I}^k, t^k) = p(S^k | I^k) \quad (\text{A.81})$$

where $p(S^k | I^k)$ is given by (4.8). The same applies to $p(\mathbf{X}^{k+1} | \mathbf{N}^{k+1}) p(\mathbf{N}^{k+1}, b^{k+1})$ and $p(\mathbf{X}^{k+1} | \mathbf{S}^k) p(\mathbf{X}^k, t^k, \mathbf{S}^k, t^k - d^k)$.

A.12 Simplification of the transition density of the JMKPD

The transition density of the JMKPD can be expanded as:

$$\begin{aligned} p(\mathbf{X}^{k+1}, I^{k+1} | \mathbf{X}^k, I^k) = & p(\mathbf{X}^{k+1} | S^k, \mathbf{X}^{k+1} | N^{k+1}, N^{k+1} | \mathbf{X}^k, I^k) \\ = & p(\mathbf{X}^{k+1} | S^k, \mathbf{X}^k, I^k) \end{aligned}$$

$$\cdot p\left(\mathbf{X}^{k+1}\left(N^{k+1}\right), N^{k+1} \mid \mathbf{X}^k, I^k, \mathbf{X}^{k+1}\left(S^k\right), S^k\right) \quad (\text{A.82})$$

$$\begin{aligned} &= p\left(\mathbf{X}^{k+1}\left(S^k\right), S^k \mid \mathbf{X}^k, I^k\right) \\ &\quad \cdot p\left(N^{k+1} \mid \mathbf{X}^k, I^k, \mathbf{X}^{k+1}\left(S^k\right), S^k\right) \\ &\quad \cdot p\left(\mathbf{X}^{k+1}\left(N^{k+1}\right) \mid \mathbf{X}^k, I^k, \mathbf{X}^{k+1}\left(S^k\right), S^k, N^{k+1}\right) \end{aligned} \quad (\text{A.83})$$

The set of the labels of the newborn targets at time $k+1$, N^{k+1} , does not depend on $\mathbf{X}^k, I^k, \mathbf{X}^{k+1}\left(S^k\right), S^k$. Besides, the states of the born targets are independent of the states of the targets that are alive and their labels. Therefore, (A.83) can be written as:

$$\begin{aligned} p\left(\mathbf{X}^{k+1}, I^{k+1} \mid \mathbf{X}^k, I^k\right) &= p\left(\mathbf{X}^{k+1}\left(S^k\right), S^k \mid \mathbf{X}^k, I^k\right) \\ &\quad \cdot p\left(N^{k+1}\right) p\left(\mathbf{X}^{k+1}\left(N^{k+1}\right) \mid N^{k+1}\right) \end{aligned} \quad (\text{A.84})$$

Equation (A.84) can be expanded as

$$\begin{aligned} p\left(\mathbf{X}^{k+1}, I^{k+1} \mid \mathbf{X}^k, I^k\right) &= p\left(S^k \mid \mathbf{X}^k, I^k\right) p\left(\mathbf{X}^{k+1}\left(S^k\right) \mid \mathbf{X}^k, I^k, S^k\right) \\ &\quad \cdot p\left(N^{k+1}\right) p\left(\mathbf{X}^{k+1}\left(N^{k+1}\right) \mid N^{k+1}\right) \end{aligned} \quad (\text{A.85})$$

The prior probability of death of a target does not depend on its state, as indicated by (4.4), therefore, neither does the set of labels of the surviving targets.

$$\begin{aligned} p\left(\mathbf{X}^{k+1}, I^{k+1} \mid \mathbf{X}^k, I^k\right) &= p\left(N^{k+1}\right) p\left(S^k \mid I^k\right) p\left(\mathbf{X}^{k+1}\left(N^{k+1}\right) \mid N^{k+1}\right) \\ &\quad \cdot p\left(\mathbf{X}^{k+1}\left(S^k\right) \mid \mathbf{X}^k, S^k\right) \end{aligned} \quad (\text{A.86})$$

A.13 Rao-Blackwellisation of target velocities

In this appendix it is explained how to obtain the conditional PDF of the velocity of a target given its current and past positions. This is needed to apply Rao-Blackwellisation and remove the velocity elements from the particle filter. Using equation (4.10) for one target

$$\mathbf{p}^k = \mathbf{p}^{k-1} + \tau \mathbf{v}^{k-1} + \mathbf{w}_p \quad (\text{A.87})$$

$$\mathbf{v}^k = \mathbf{v}^{k-1} + \mathbf{w}_v \quad (\text{A.88})$$

where $\mathbf{p}^k$ is the position of the target at time k , $\mathbf{v}^k$ is the velocity of the target at time k , $\mathbf{w}_p$ is the process noise related to the position and $\mathbf{w}_v$ is the process noise related to the velocity. According to the covariance matrix of equation (4.12), it should be noted that the noises $\mathbf{w}_p$ are $\mathbf{w}_v$ correlated.

The mean and the covariance matrix of the Gaussian random variable $\mathbf{v}^k$ given the positions up to time k , $\mathbf{p}^{0:k}$, are calculated using the fundamental equation of linear estimation [13]:

$$\mathbb{E}[\mathbf{v}^k | \mathbf{p}^{0:k}] = \mathbb{E}[\mathbf{v}^k | \mathbf{p}^{0:k-1}] + \Psi \mathbf{S}^{-1} (\mathbf{p}^k - \mathbb{E}[\mathbf{p}^k | \mathbf{p}^{0:k-1}]) \quad (\text{A.89})$$

$$\text{cov}[\mathbf{v}^k | \mathbf{p}^{0:k}] = \text{cov}[\mathbf{v}^k | \mathbf{p}^{0:k-1}] - \Psi \mathbf{S}^{-1} \Psi^T \quad (\text{A.90})$$

where

$$\Psi = \text{cov}[\mathbf{v}^k, \mathbf{p}^k | \mathbf{p}^{0:k-1}] \quad (\text{A.91})$$

$$\mathbf{S} = \text{cov}[\mathbf{p}^k | \mathbf{p}^{0:k-1}] \quad (\text{A.92})$$

All the terms of equations (A.89) and (A.90) are:

- Term 1:

$$\mathbb{E}[\mathbf{v}^k | \mathbf{p}^{0:k-1}] = \bar{\mathbf{v}}^{k-1} \quad (\text{A.93})$$

where $\bar{\mathbf{v}}^{k-1}$ is the mean velocity at time $k-1$ given the positions up to time $k-1$, $\mathbf{p}^{0:k-1}$.

- Term 2:

$$\Psi = \tau \Sigma^{k-1} + \mathbf{Q}_{vp} \quad (\text{A.94})$$

where $\mathbf{Q}_{vp}$ is the covariance matrix of the process noises $\mathbf{w}_v$ and $\mathbf{w}_p$, and Σ^{k-1} is the covariance matrix of the velocity at time $k-1$ given the positions up to time $k-1$, $\mathbf{p}^{0:k-1}$.

- Term 3:

$$\mathbf{S} = \tau^2 \Sigma^{k-1} + \mathbf{Q}_p \quad (\text{A.95})$$

where $\mathbf{Q}_p$ is the covariance matrix of $\mathbf{w}_p$.

- Term 4:

$$\mathbb{E}[\mathbf{p}^k | \mathbf{p}^{0:k-1}] = \mathbf{p}^{k-1} + \tau \bar{\mathbf{v}}^{k-1} \quad (\text{A.96})$$

- Term 5:

$$\text{cov}[\mathbf{v}^k | \mathbf{p}^{0:k}] = \Sigma^{k-1} + \mathbf{Q}_v \quad (\text{A.97})$$

A.14 Sampling states and labels of the surviving targets

This appendix clarifies how the two-layer particle filter developed in Section 4.4 samples the target states and labels of the surviving targets. The importance density is given by (4.38):

$$q_p \left(\mathbf{P}^{k+1} \left(I^k \right), S^k, \mathbf{a} \mid \mathbf{z}^{1:k+1} \right) = q_1 \left(\mathbf{P}^{k+1} \left(I^k \right), \mathbf{a} \mid \mathbf{z}^{1:k+1} \right) q_2 \left(S^k \mid \mathbf{P}^{k+1}, \mathbf{a}, \mathbf{z}^{1:k+1} \right) \quad (\text{A.98})$$

where $q_1(\cdot)$ is the importance density of the PP method and $q_2(\cdot)$ samples the labels of the surviving targets. The importance density of the PP method for unknown and variable number of targets is

$$q_1 \left(\mathbf{P}^{k+1} \left(I^k \right), \mathbf{a} \mid \mathbf{z}^{1:k+1} \right) = \prod_{j=1}^{t^k} b_j \left(\mathbf{p}^{k+1} \left(i_j^k \right) \right) p \left(\mathbf{p}^{k+1} \left(i_j^k \right) \mid \mathbf{x}_{j,a_j}^k \right) \quad (\text{A.99})$$

where $I^k = \{i_1^k, i_2^k, \dots, i_{t^k}^k\}$, $\mathbf{P}^k = \left[\left(\mathbf{p}_1^k \right)^T, \dots, \left(\mathbf{p}_{t^k}^k \right)^T \right]$ and it should be recalled that set I^k is needed to determine the dimensionality of vector $\mathbf{P}^k$. $\mathbf{P}^{k+1} \left(I^k \right) = \left[\left(\mathbf{p}^{k+1} \left(i_1^k \right) \right)^T, \dots, \left(\mathbf{p}^{k+1} \left(i_{t^k}^k \right) \right)^T \right]^T$ is a vector of dimension t^k that includes the positions of the targets that were alive at the previous time assuming they have not died at the current time.

Note that $q_1(\cdot)$ depends on I^k , $\mathbf{p}^{0:k} \left(i_1^k \right), \dots, \mathbf{p}^{0:k} \left(i_{t^k}^k \right)$ and $\mathbf{z}^{1:k+1}$. I^k is needed to know the labels and the number of targets. The whole trajectories of each one of the targets and the measurements are needed to apply Rao-Blackwellisation. It should be mentioned that the trajectories of the targets can be obtained as their labels are known. All the information about the target trajectories and past measurements is summarised in the Rao-Blackwellised subparticles $\mathbf{x}_{j,1}^k, \dots, \mathbf{x}_{j,t^k}^k$ noting that these Rao-Blackwellised subparticles do not need to include the labels as the labels are associated with their order as happens with vector $\mathbf{P}^k$.

The important issue that is worth highlighting about (4.38) is that, firstly, the positions of all the targets that were alive at the previous time are sampled and, then, conditioned on them, the labels of these targets are sampled. The aim of this appendix is to explain why this is possible and why it can be beneficial compared to sampling first the labels and then the target positions. This appendix does not deal with the PP method and, therefore, does not give the reasons why sampling the auxiliary variable $\mathbf{a}$ can improve performance in multitarget tracking scenarios as this issue is properly addressed in Section 4.3.1. Therefore, this auxiliary variable will be dropped for the sake of clarity in the rest of this appendix.

In the following, several ways of sampling states and labels are analysed to explain how to draw samples and update the weights using (4.38). For the ease of explanation, in Section A.14.1, the sampling procedures are explained using target existences (see Appendix B.2 for more details) rather than labels while in Section A.14.2 labels are used directly in the explanation.

A.14.1 Explanation using target existences

A.14.1.1 Posterior of the target states and labels

Note that S^k is a subset of I^k and N^k is a subset in which the maximum number of elements is the number of cells. Therefore, both sets have a maximum number of elements². This fact is important in the following.

In general, using vector notation, the variable $([x_1, \dots, x_t], [i_1, \dots, i_t], t)$ is used to represent the posterior PDF as the number of targets is unknown and variable. However, to represent the posterior PDF of the surviving targets at a time conditioned on the state at the previous time, it is easier to represent it in the variable $([x'_1, \dots, x'_N], [e_1, \dots, e_N])$ where N is the number of targets that were alive at the previous time, $e_i = 1$ indicates that the i th target exists at the current time (the i th target is still alive), $e_i = 0$ indicates that the i th target does not exist at the current time (the i th target has died) and x'_i is the state of the i th target assuming it exists at the following time. In fact, according to the prior, the posterior $p_a(\cdot)$ of the surviving targets is of the form

$$p_a([x'_1, \dots, x'_N], [e_1, \dots, e_N]) = \prod_{i=1}^N [p_{d,i} \delta(e_i - 0) + (1 - p_{d,i}) \delta(e_i - 1) \pi_i(x'_i)] \quad (\text{A.100})$$

where $p_{d,i}$ is the probability of death of target i , $\pi_i(\cdot)$ is the posterior of the i th target and it has been assumed that the target existences and positions are independent.

Note that the posterior PDF $p_b(\cdot)$ in the variable $([x_1, \dots, x_t], [i_1, \dots, i_t], t)$ where $t \in \{0, \dots, N\}$ could be represented as

$$p_b([x_1, \dots, x_t], [i_1, \dots, i_t], t) = p(t) p([i_1, \dots, i_t] | t) p([x_1, \dots, x_t] | [i_1, \dots, i_t], t)$$

but it has a more cumbersome expression than the posterior with target existences. However, as indicated in Appendix B, the representations are equivalent, and the-

²Even though the development described in this appendix is particularised for the survival of the targets, it should be noted that it can also be applied to the newborn targets.

refore

$$p_a([x'_1, \dots, x'_N], [e_1, \dots, e_N]) = p_b\left([x'_{i_1}, \dots, x'_{i_t}], [i_1, \dots, i_t], \sum_{i=1}^N e_i\right) \quad (\text{A.101})$$

where $[i_1, \dots, i_t]$ is the ordered vector that contains the positions of $[e_1, \dots, e_N]$ that are equal to one.

A.14.1.2 Types of importance densities

As it has been assumed that the posterior is of the form given by (A.100), sampling the posterior PDF of the target positions and existences can be done independently for each target. Hence, in the following, a single target case is addressed by analysing three ways of sampling although what is written in this section can be generalised when there are multiple targets and their positions and existences are not independent. In the first method, sampling of the existence is done before sampling the state while in the others, sampling of the state is done before sampling the existence.

The posterior for one target is

$$p(x, e) = p_d \delta(e - 0) + (1 - p_d) \delta(e - 1) \pi(x) \quad (\text{A.102})$$

where the index that indicates the specific target and the tildes of x' have been dropped and $x \in \mathbb{R}$ for simplicity. It is important to note that for $e = 0$ the state x does not exist.

Importance density of type 1: The importance density is

$$q(x, e) = q_1 \delta(e - 0) + (1 - q_1) \delta(e - 1) \pi_q(x) \quad (\text{A.103})$$

where $q_1 \in [0, 1]$ and $\pi_q(\cdot)$ is the importance density of the target state.

Sampling from this importance density is done in the following way.

- Draw a sample e_i where $e_i = 1$ with probability $(1 - q_1)$ and $e_i = 0$ with probability q_1
- If $e_i = 1$
 - Draw a sample $x_i \sim \pi_q(\cdot)$

Therefore, there are samples of the form $\{e_i = 0\}_{i=1}^{N_0}$ and $\{e_i = 1, x_i\}_{i=1}^{N_1}$.

If a sample has $e_i = 0$, its importance weight is

$$w_i \propto p_d / q_1 \quad (\text{A.104})$$

If a sample has $e_i = 1$, its importance weight is

$$w_i \propto \frac{(1 - p_d) \pi(x_i)}{(1 - q_1) \pi_q(x_i)} \quad (\text{A.105})$$

Importance density of type 2: There might be situations in which drawing from an importance density of type 1 is not the best option as will be demonstrated in the next section.

Let us consider the importance density

$$q'(x, e) = \pi_q(x) [q_2(x) \delta(e - 0) + (1 - q_2(x)) \delta(e - 1)] \quad (\text{A.106})$$

where $q_2(x) \in [0, 1] \forall x \in \mathbb{R}$ and is a function of the value of x that is drawn³.

Sampling from this importance density is done in the following way:

- Draw a sample $x_i \sim \pi_q(\cdot)$
- Draw a sample e_i where $e_i = 1$ with probability $(1 - q_2(x_i))$ and $e_i = 0$ with probability $q_2(x_i)$

The problem with this scheme is that even for $e_i = 0$ a value x_i has been drawn but it should not be considered when updating the weights of the particles such that $e_i = 0$ as it does not exist. Therefore, for $e = 0$ the value of x must be integrated out and the resulting importance density is

$$q(x, e) = \int q_2(\xi) \pi_q(\xi) d\xi \delta(e - 0) + (1 - q_2(x)) \pi_q(x) \delta(e - 1) \quad (\text{A.107})$$

Sampling from this importance density can be done in the following way:

- Draw a sample $x_i \sim \pi_q(\cdot)$
- Draw a sample e_i where $e_i = 1$ with probability $(1 - q_2(x_i))$ and $e_i = 0$ with probability $q_2(x_i)$
- If $e_i = 0$ discard x_i

³Note that $q_2(x) \in [0, 1] \forall x \in \mathbb{R}$ ensures that $\int \pi_q(x) q_2(x) dx \in [0, 1]$.

Therefore, if a sample has $e_i = 0$, its importance weight is

$$w_i \propto \frac{p_d}{q(e_i)} = \frac{p_d}{\int \pi_q(x) q_2(x) dx} \quad (\text{A.108})$$

where

$$q(e) = \int q(x, e) dx \quad (\text{A.109})$$

If a sample has $e_i = 1$, its importance weight is

$$w_i \propto \frac{(1 - p_d) \pi(x_i)}{(1 - q_2(x_i)) \pi_q(x_i)} \quad (\text{A.110})$$

It should be noted that the difficulty of this method is to calculate (A.109) but it is possible in some cases as in the example of the following section and can provide performance benefits.

Importance density of type 3: The importance density of type 2 has the problem that calculating (A.109) might not be possible. However, another approach is also possible by considering the posterior PDF:

$$p'(x, e) = \pi(x) [p_d \delta(e - 0) + (1 - p_d) \delta(e - 1)] \quad (\text{A.111})$$

Note that this PDF is equivalent to the real posterior, which is given by (A.102), but assuming the target state also exists even if the target has disappeared. This does not seem to make sense in the real world but this PDF can be used to draw samples from the true posterior (A.102). Note that this bears some resemblance to auxiliary particle filtering for $e = 0$ as an auxiliary variable x is being used.

The importance density given by (A.106) is used. Therefore, if a sample has $e_i = 0$, its importance weight is

$$w_i \propto \frac{p_d \pi(x_i)}{q_2(x_i) \pi_q(x_i)} \quad (\text{A.112})$$

If a sample has $e_i = 1$, its importance weight is

$$w_i \propto \frac{(1 - p_d) \pi(x_i)}{(1 - q_2(x_i)) \pi_q(x_i)} \quad (\text{A.113})$$

Therefore,

$$p'(x, e) \approx \sum_{i=1}^{N_{par}} w_i \delta(x - x_i) \delta(e - e_i) \quad (\text{A.114})$$

It should be noted that the following relationship can be established:

$$p(x, e) = \begin{cases} \int p'(x, e) dx & e = 0 \\ p'(x, e) & e = 1 \end{cases} \quad (\text{A.115})$$

Therefore, once the particle filter approximation to $p'(\cdot)$ is obtained, one can integrate out the target state for $e = 0$ and obtain the particle filter approximation to $p(x, e)$:

$$p(x, e) \approx \delta(e - 0) \sum_{i=1}^{N_{par}} w_i \chi_{\{0\}}(e_i) + \delta(e - 1) \sum_{i=1}^{N_{par}} w_i \chi_{\{1\}}(e_i) \delta(x - x_i) \quad (\text{A.116})$$

where $\chi_{\{a\}}(e_i) = 1$ if $e_i = a$ and zero otherwise.

There is one important fact that should be highlighted. Note that the importance density of type 2 uses the marginal PDF to evaluate the weights for $e = 0$ while the importance density of type 3 makes use of the auxiliary variable to evaluate the weights and then integrates out this auxiliary variable for $e = 0$. This bears some resemblance to what the marginal particle filter (see Section 2.5.2) and auxiliary particle filter (see Section 2.5.3) do. The variance of the weights of the marginal particle filter is lower than the variance of the weights of the auxiliary particle filter [110]. Therefore, the importance density of type 2 is expected to have a higher performance than the importance density of type 3 but recalling that calculating (A.109) might not be possible.

A.14.1.3 Numerical example

This numerical example aims to demonstrate how the sampling procedures described in Section A.14.1.2 can be applied.

The posterior PDF is assumed to be

$$p(x, e) = \frac{a\delta(e - 0) + b\delta(e - 1)\mathcal{N}(x; 0, 1)u(x) + c\delta(e - 1)\mathcal{N}(x; 0, 1)u(-x)}{a + b/2 + c/2} \quad (\text{A.117})$$

where $a \geq 0, b \geq 0, c \geq 0$ and $u(x) = 1$ for $x > 0$ and zero otherwise. The corresponding $p'(\cdot)$ is

$$p'(x, e) = \frac{a\delta(e - 0) + b\delta(e - 1)u(x) + c\delta(e - 1)u(-x)}{a + b/2 + c/2} \mathcal{N}(x; 0, 1) \quad (\text{A.118})$$

Note that this posterior PDF would represent the following scenario. The dynamic model of the target is given by the PDF $\mathcal{N}(\cdot; 0, 1)$. The prior probability of

Table A.1: Likelihood for the numerical example

State	Likelihood
$e = 0$	$(1 - P_{fa}) P_{fa}$
$e = 1, x > 0$	$(1 - P_d) P_{fa}$
$e = 1, x < 0$	$P_d (1 - P_{fa})$

target death is p_d . There are two binary sensors. One measures in the positions $x > 0$ and the other in $x < 0$ with a probability of detection P_d and a probability of false alarm P_{fa} . It is assumed that the sensor in $x < 0$ measures 1 (detection) and the sensor in $x > 0$ measures 0 (no detection). The likelihood for this specific measurement is given in Table A.1.

Therefore, equation (A.117) models this scenario if

$$\begin{aligned} a &= (1 - P_{fa}) P_{fa} p_d \\ b &= (1 - P_d) P_{fa} (1 - p_d) \\ c &= P_d (1 - P_{fa}) (1 - p_d) \end{aligned}$$

Importance density of type 1: The value q_1 is chosen as $a / (a + b/2 + c/2)$ and $\pi_q(x) = \mathcal{N}(x; 0, 1)$. Note that in this example, it would be possible to sample directly from $p(x | e = 1)$. However, this is not possible in general and, in this section, it is desired to analyse different methods when one can only sample from the prior PDF not directly from the posterior.

For a particle such that $e_i = 0$, its weight is

$$w_i \propto \frac{a / (a + b/2 + c/2)}{q_1} = 1 \quad (\text{A.119})$$

For a particle such that $e_i = 1$, its weight is

$$w_i \propto \frac{(bu(x_i) + cu(-x_i)) / (a + b/2 + c/2)}{(1 - q_1)} \quad (\text{A.120})$$

Importance density of type 2: As before $\pi_q(x) = \mathcal{N}(x; 0, 1)$ and

$$q_2(x) = \begin{cases} q_{21} & x < 0 \\ q_{22} & x > 0 \end{cases} \quad (\text{A.121})$$

where $q_{21} = a / (a + b)$ and $q_{22} = a / (a + c)$.

For a particle such that $e_i = 0$, its weight is

$$w_i \propto \frac{a / (a + b/2 + c/2)}{\frac{1}{2} (q_{21} + q_{22})} \quad (\text{A.122})$$

For a particle such that $e_i = 1$, its weight is

$$w_i \propto \frac{(bu(x_i) + cu(-x_i)) / (a + b/2 + c/2)}{(1 - q_{21}) u(-x_i) + (1 - q_{22}) u(x_i)} \quad (\text{A.123})$$

Importance density of type 3: The importance density is the same as in method two. However, the way the weights are computed is different. For a particle such that $e_i = 0$, its weight is

$$w_i \propto \frac{a / (a + b/2 + c/2)}{q_{21} u(-x_i) + q_{22} u(x_i)} \quad (\text{A.124})$$

For a particle such that $e_i = 1$, its weight is

$$w_i \propto \frac{(bu(x_i) + cu(-x_i)) / (a + b/2 + c/2)}{(1 - q_{21}) u(-x_i) + (1 - q_{22}) u(x_i)} \quad (\text{A.125})$$

Performance: These methods are analysed plotting the effective sampling size (see Section 2.5.1.1 for more details) and the estimated values of

$$a_n = \frac{a}{a + b/2 + c/2} \quad (\text{A.126})$$

$$b_n = \frac{b/2}{a + b/2 + c/2} \quad (\text{A.127})$$

$$c_n = \frac{c/2}{a + b/2 + c/2} \quad (\text{A.128})$$

given by

$$\hat{a}_n = \sum_i w_i \chi_{\{0\}}(e_i) \quad (\text{A.129})$$

$$\hat{b}_n = \sum_{i=1}^{N_{par}} w_i \chi_{A^+}([e_i, x_i]) \quad (\text{A.130})$$

$$\hat{c}_n = \sum_{i=1}^{N_{par}} w_i \chi_{A^-}([e_i, x_i]) \quad (\text{A.131})$$

where

$$A^+ = \{[e, x] : e = 1 \wedge x > 0\} \quad (\text{A.132})$$

Figure A.1: Effective sampling size for three sampling methods.

Figure A.2: RMS error of the estimated coefficients for three sampling methods.

$$A^- = \{[e, x] : e = 1 \wedge x < 0\} \quad (\text{A.133})$$

The effect of c on the effective sampling size and RMS error is analysed. The parameters of the simulation are $a = 0.1$, $b = 0.75$, the number of particles is 10000 and the number of Monte Carlo runs 1000. The effective sampling size is shown in Figure A.1. The second way of sampling always provides the highest sampling size, especially, for low c . The effective sampling size of the third way of sampling is higher than the first way of sampling but lower than the second. The result is that the second way of sampling provides lower errors to estimate b and c , especially for low c , as shown in Figure A.2. The errors of the first and third methods of sampling are roughly the same in this set-up.

A.14.2 Explanation using sets for the labels

In this section, the explanation in Section A.14.1 is given when the labels are represented by sets without introducing target existence notation. This is an alternative explanation to that given in Section A.14.1.

The posterior PDF of the surviving targets can be written as

$$p(\mathbf{P}^{k+1}(S^k), S^k)$$

Note that the aim is to obtain weighted samples from this density. For each $S^k \subseteq I^k$, define a density $p'(\cdot)$ such that

$$p(\mathbf{P}^{k+1}(S^k), S^k) = \int p'(\mathbf{P}^{k+1}(I^k), S^k) d\mathbf{P}^{k+1}(I^k \setminus S^k) \quad (\text{A.134})$$

where the set $A \setminus B = \{x \in A : x \notin B\}$.

The importance density $q'(\cdot)$ is used to draw samples from $p'(\cdot)$. The samples from $q'(\cdot)$ can be weighted in two ways. In the type 2 importance density, the weights are

$$w_i \propto \frac{p(\mathbf{P}_i^{k+1}(S_i^k), S_i^k)}{q(\mathbf{P}_i^{k+1}(S_i^k), S_i^k)} \quad (\text{A.135})$$

where

$$q(\mathbf{P}^{k+1}(S^k), S^k) = \int q'(\mathbf{P}^{k+1}(I^k), S^k) d\mathbf{P}^{k+1}(I^k \setminus S^k) \quad (\text{A.136})$$

In the type 3 importance density, the weights are

$$w_i \propto \frac{p'(\mathbf{P}_i^{k+1}(I^k), S_i^k)}{q'(\mathbf{P}_i^{k+1}(I^k), S_i^k)} \quad (\text{A.137})$$

In this case the subparticles $\mathbf{P}_i^{k+1}(I^k \setminus S_i^k)$ are discarded to obtain the desired Monte Carlo approximation of $p(\cdot)$.

The specific implementation is as follows. The density $p'(\cdot)$ is

$$\begin{aligned} p'(\mathbf{P}^{k+1}(I^k), S^k) &\propto p(\mathbf{z}^k | \mathbf{P}^{k+1}(I^k), S^k) \\ &\prod_{i \in I^k} p(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) [p_d(1 - \chi_{S^k}(i)) + (1 - p_d)\chi_{S^k}(i)] \\ &= p(\mathbf{z}^k | \mathbf{P}^{k+1}(S^k), S^k) \end{aligned}$$

$$\prod_{i \in S^k} p(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) (1 - p_d) \prod_{i \in I^k \setminus S^k} p(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) p_d \quad (\text{A.138})$$

where $p(\mathbf{z}^k | \mathbf{P}^{k+1}(I^k), S^k) = p(\mathbf{z}^k | \mathbf{P}^{k+1}(S^k), S^k)$ as the “positions” of the targets that have died do not affect the sensor measurements. Using (A.134), the density $p(\cdot)$ is

$$p(\mathbf{P}^{k+1}(S^k), S^k) \propto p(\mathbf{z}^k | \mathbf{P}^{k+1}(S^k), S^k) \prod_{i \in S^k} p(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) (1 - p_d) \prod_{i \in I^k \setminus S^k} p_d \quad (\text{A.139})$$

The importance density $q'(\cdot)$ is

$$\begin{aligned} q'(\mathbf{P}^{k+1}(I^k), S^k) &= \prod_{i \in I^k} q_{pos}(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) \\ &\quad \left[(1 - q_{ex}(\mathbf{p}^{k+1}(i))) (1 - \chi_{S^k}(i)) + q_{ex}(\mathbf{p}^{k+1}(i)) \chi_{S^k}(i) \right] \\ &= \prod_{i \in S^k} q_{pos}(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) q_{ex}(\mathbf{p}^{k+1}(i)) \\ &\quad \prod_{i \in I^k \setminus S^k} q_{pos}(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) (1 - q_{ex}(\mathbf{p}^{k+1}(i))) \end{aligned} \quad (\text{A.140})$$

where $q_{pos}(\cdot)$ is the importance density that samples the target positions and $q_{ex}(\cdot)$ indicates the probability that a target survives in the overall importance density.

For the type 3 importance density the weights are easily found using (A.137). For the type 2 importance density the weights require $q(\cdot)$ which can be written as

$$\begin{aligned} q(\mathbf{P}^{k+1}(I^k), S^k) &= \prod_{i \in S^k} q_{pos}(\mathbf{p}^{k+1}(i) | \mathbf{p}^k(i)) q_{ex}(\mathbf{p}^{k+1}(i)) \\ &\quad \prod_{i \in I^k \setminus S^k} \int q_{pos}(\mathbf{p} | \mathbf{p}^k(i)) (1 - q_{ex}(\mathbf{p})) d\mathbf{p} \end{aligned} \quad (\text{A.141})$$

It should be noted that the numerical example in Section A.14.1.3 could have been written analogously using the nomenclature of this section.

A.14.3 Discussion on the application to the two-layer PF

The algorithm to propagate the target states and remove tracks of the two-layer PF uses an importance density of type 3. It samples the target states of all the targets that were alive at the previous time, which are indicated by $\mathbf{P}^{k+1}(I^k)$ in (4.38), and then it samples the labels of these targets, which are indicated by set S^k .

However, as the PP method is used in the two-layer PF, it should be noted that an auxiliary variable was introduced. This auxiliary variable has not been considered in this appendix for the clarity of explanation.

A.15 Approximating the evidence using particle filters

This derivation appears in Section 5.2.2. The objective is to calculate the evidence recursively using

$$p(\mathbf{z}^k | \mathbf{z}^{1:k-1}) = \int p(\mathbf{z}^k | \mathbf{X}^k) p(\mathbf{X}^{0:k} | \mathbf{z}^{1:k-1}) d\mathbf{X}^{0:k} \quad (\text{A.142})$$

where the conditioning on H_t has been dropped for the sake of clarity. This integral can be approximated by Monte Carlo integration drawing samples from the importance density $q(\mathbf{X}^{0:k} | \mathbf{z}^{1:k})$:

$$p(\mathbf{z}^k | \mathbf{z}^{1:k-1}) = \int \frac{p(\mathbf{z}^k | \mathbf{X}^k) p(\mathbf{X}^{0:k} | \mathbf{z}^{1:k-1})}{q(\mathbf{X}^{0:k} | \mathbf{z}^{1:k})} q(\mathbf{X}^{0:k} | \mathbf{z}^{1:k}) d\mathbf{X}^{0:k} \quad (\text{A.143})$$

Substituting (5.10) into (A.143) and developing some terms:

$$p(\mathbf{z}^k | \mathbf{z}^{1:k-1}) = \int \frac{p(\mathbf{z}^k | \mathbf{X}^k) p(\mathbf{X}^k | \mathbf{X}^{k-1}) p(\mathbf{X}^{0:k-1} | \mathbf{z}^{1:k-1})}{q(\mathbf{X}^k | \mathbf{X}^{0:k-1}, \mathbf{z}^{1:k})} \frac{p(\mathbf{X}^{0:k-1} | \mathbf{z}^{1:k-1})}{q(\mathbf{X}^{0:k-1} | \mathbf{z}^{1:k-1})} q(\mathbf{X}^{0:k} | \mathbf{z}^{1:k}) d\mathbf{X}^{0:k} \quad (\text{A.144})$$

Drawing N_{par} samples from $q(\mathbf{X}^{0:k} | \mathbf{z}^{1:k})$, $\{\mathbf{X}_i^{0:k}\}_{i=1}^{N_{par}}$, (A.144) is approximated as

$$p(\mathbf{z}^k | \mathbf{z}^{1:k-1}) \approx \sum_{i=1}^{N_{par}} \frac{p(\mathbf{z}^k | \mathbf{X}_i^k) p(\mathbf{X}_i^k | \mathbf{X}_i^{k-1}) p(\mathbf{X}_i^{0:k-1} | \mathbf{z}^{1:k-1})}{q(\mathbf{X}_i^k | \mathbf{X}_i^{0:k-1}, \mathbf{z}^{1:k})} \frac{p(\mathbf{X}_i^{0:k-1} | \mathbf{z}^{1:k-1})}{q(\mathbf{X}_i^{0:k-1} | \mathbf{z}^{1:k-1})} \quad (\text{A.145})$$

Using $k-1$ instead of k in (5.9), (A.145) becomes

$$p(\mathbf{z}^k | \mathbf{z}^{1:k-1}) \approx \sum_{i=1}^{N_{par}} \frac{p(\mathbf{z}^k | \mathbf{X}_i^k) p(\mathbf{X}_i^k | \mathbf{X}_i^{k-1})}{q(\mathbf{X}_i^k | \mathbf{X}_i^{0:k-1}, \mathbf{z}^{1:k})} w_i^{k-1} \quad (\text{A.146})$$

Defining the unnormalised weights as

$$\tilde{w}_i^k = \frac{p(\mathbf{z}^k | \mathbf{X}_i^k) p(\mathbf{X}_i^k | \mathbf{X}_i^{k-1})}{q(\mathbf{X}_i^k | \mathbf{X}_i^{0:k-1}, \mathbf{z}^{1:k})} w_i^{k-1} \quad (\text{A.147})$$

equation (A.146) can be written as:

$$p(\mathbf{z}^k | \mathbf{z}^{1:k-1}) \approx \sum_{i=1}^{N_{par}} \tilde{w}_i^k \quad (\text{A.148})$$

A.16 Kullback-Leibler divergence: Proof

In this section, it is shown why the KL divergence in the example of Section 5.2.3.2 is a decreasing function of the number of targets and an increasing function of SNR_0 .

Let us write (5.24) as

$$\begin{aligned} D_{KL} &= \frac{1}{2} \left(c + \left(\sqrt{(t+1)SNR_0} - \sqrt{t \cdot SNR_0} \right)^2 \right) \\ &= \frac{1}{2} \left(c + SNR_0 (2t + 1 - 2\sqrt{t^2 + t}) \right) \end{aligned} \quad (\text{A.149})$$

A.16.1 Behaviour with the number of targets

Here, it is shown that (A.149) is a decreasing function of the number of targets $t \geq 0$ taking into account that $SNR_0 > 0$. Therefore, what needs to be shown is that

$$\frac{\partial D_{KL}}{\partial t} < 0, \quad \forall t \geq 0, \forall SNR_0 > 0 \quad (\text{A.150})$$

Calculating the derivative with respect to t

$$\frac{\partial D_{KL}}{\partial t} = \frac{SNR_0}{2} \left(2 - \frac{2t+1}{\sqrt{t^2+t}} \right) \quad (\text{A.151})$$

In order to show that $\frac{\partial D_{KL}}{\partial t} < 0$ one must obtain its maximum and it must be lower than 0 noting that $SNR_0 > 0$ does not affect the result. To this end, one calculates the second derivative and equals it to zero

$$\frac{\partial^2 D_{KL}}{\partial t^2} = \frac{SNR_0}{4(t^2+t)^{3/2}} \quad (\text{A.152})$$

For $SNR_0 > 0$ and $t > 0$, $\frac{\partial D_{KL}}{\partial t}$ has no maximums nor minimums and it is always an increasing function as (A.152) is always higher than 0. Therefore, one must study the limits

$$\lim_{t \rightarrow 0} \frac{\partial D_{KL}}{\partial t} = -\infty \quad (\text{A.153})$$

$$\lim_{t \rightarrow \infty} \frac{\partial D_{KL}}{\partial t} = 0 \quad (\text{A.154})$$

This shows (A.150).

A.16.2 Behaviour with SNR_0

Here, it is shown that (A.149) is an increasing function of SNR_0 . Therefore, what needs to be shown is that

$$\frac{\partial D_{KL}}{\partial SNR_0} > 0, \quad \forall t \geq 0, \forall SNR_0 > 0 \quad (\text{A.155})$$

Calculating the derivative with respect to SNR_0

$$\frac{\partial D_{KL}}{\partial SNR_0} = \frac{1}{2} \left(2t + 1 - 2\sqrt{t^2 + t} \right) \quad (\text{A.156})$$

In order to show that $\frac{\partial D_{KL}}{\partial SNR_0} > 0$ one must obtain its minimum and it must be higher than 0.

$$\frac{\partial^2 D_{KL}}{\partial SNR_0 \partial t} = \frac{1}{2} \left(2 - \frac{2t + 1}{\sqrt{t^2 + t}} \right) \quad (\text{A.157})$$

Using (A.151) and (A.150), one gets

$$\frac{\partial^2 D_{KL}}{\partial SNR_0 \partial t} < 0, \quad \forall t \geq 0, \forall SNR_0 > 0 \quad (\text{A.158})$$

Therefore, for $SNR_0 > 0$ and $t > 0$, $\frac{\partial D_{KL}}{\partial SNR_0}$ has no maximums nor minimums and it is always a decreasing function because of (A.158). Therefore, one must study the limits

$$\lim_{t \rightarrow 0} \frac{\partial D_{KL}}{\partial SNR_0} = \frac{1}{2} \quad (\text{A.159})$$

$$\lim_{t \rightarrow \infty} \frac{\partial D_{KL}}{\partial SNR_0} = 0 \quad (\text{A.160})$$

This shows (A.155).

A.17 Example: Unique decomposition of a PDF

In this appendix, an example is given to illustrate the idea that when the information about target labelling is lost at a certain time (the probability of successful labelling of any estimate is 1/2), it does not necessarily mean that the PDF is permutation invariant. Also, it clarifies the fact that a permutation strictly variant posterior PDF does not imply that the target labels are perfectly known.

Consider the following posterior PDF

$$\pi^k(\mathbf{X}^k) = \frac{1}{2}\mathcal{N}(\mathbf{X}^k; [10, 1000]^T, \mathbf{I}_2) + \frac{1}{2}\mathcal{N}(\mathbf{X}^k; [2000, 80]^T, \mathbf{I}_2) \quad (\text{A.161})$$

To be exact with the reasoning, let us assume that the Gaussian PDFs represented in (A.161) have a bounded support around their mean. Therefore, this posterior PDF is permutation strictly variant. However, for any estimate $\hat{\mathbf{X}}^k \in \mathbb{R}^2$, the probability of successful labelling (for example, using the Euclidean metric) is 1/2, see Section 5.3.3.

There are two important conclusions. The first one is that if a PDF is permutation strictly variant, it does not imply that the target labels are perfectly known. Therefore, the fact that a PDF is permutation strictly variant is a necessary condition so that there is absolute certainty about target labels. The second one is that, as the probability of successful labelling is 1/2 regardless of the estimate, there is a PDF in which the information about target labelling is lost but it is not permutation invariant. Therefore, a permutation invariant PDF is a sufficient condition so that the labelling information is lost but it is not necessary.

A.18 Form of the posterior after a target crossing (I)

It is known that the posterior PDF is multimodal after the targets have moved in close proximity [24, 144]. Therefore, it can be represented as

$$\pi^k(\mathbf{X}^k) = (1 - a_c^k) \pi_n^k(\mathbf{X}^k) + a_c^k \pi_c^k(\mathbf{X}^k) \quad (\text{A.162})$$

where $\pi_n^k(\cdot)$ is centred with respect to a value $\tilde{\mathbf{X}}_n^k$ and $\pi_c^k(\cdot)$ is centred with respect to its mirror $\Pi\tilde{\mathbf{X}}_n^k$ and $a_c^k \in [0, 1/2]$ without loss of generality as $\pi_n^k(\cdot)$ denotes the component of the mixture with highest weight. The motivation behind the existence of this representation is that the likelihood remains unchanged for the values $\mathbf{X}^k$ and $\Pi\mathbf{X}^k$ and, therefore, two modes tend to appear in mirror positions. Note that this can only happen if the prior is simultaneously non-negligible in points of the form $\mathbf{X}^k$ or $\Pi\mathbf{X}^k$ which implies that the states of target one and two can be interchanged and confusion in the labelling of the estimates appears. Also note that if the mirror mode does not appear is equivalent to saying that $a_c^k = 0$ so (A.162) is a general equation.

Let us denote the approximate supports of $\pi_n^k(\cdot)$ and $\pi_c^k(\cdot)$ (regions where they

are not negligible) as D_n^k and D_c^k . The PDF $\pi_{\Pi c}^k(\cdot)$ is defined as

$$\pi_{\Pi c}^k(\mathbf{X}) = \pi_c^k(\Pi\mathbf{X}) \quad (\text{A.163})$$

Assume that the targets have moved in close proximity, they are not going to be in close proximity again and k^0 is the time step when they separate. This implies that their supports do not overlap, that is:

$$D_n^k \cap D_c^k = \emptyset \quad \text{if } k > k^0 \quad (\text{A.164})$$

Note that the transitional prior and the likelihood are exactly the same for $\pi_n^k(\cdot)$ and $\pi_{\Pi c}^k(\cdot)$. Therefore, one way to look into this problem is to analyse what happens when two different initial priors (in this case initial time step refers to k^0) are used but the transitional prior and the likelihood are the same. In short, one system is characterised by the PDFs $\pi_n^{k^0}(\cdot)$, $f(\cdot|\cdot)$ and $\ell(\cdot|\cdot)$ while the other by $\pi_{\Pi c}^{k^0}(\cdot)$, $f(\cdot|\cdot)$ and $\ell(\cdot|\cdot)$ noting that the non-negligible values of the PDFs $\pi_n^{k^0}(\cdot)$ and $\pi_{\Pi c}^{k^0}(\cdot)$ roughly lie in the same area.

Under some conditions, the following property is met [32, 53, 163]

$$\lim_{k \rightarrow \infty} \left\| \pi_n^k(\cdot) - \pi_{\Pi c}^k(\cdot) \right\|_{TV} \rightarrow 0 \quad (\text{A.165})$$

where $\|\cdot\|_{TV}$ represents the total variation norm. For example, if the measurement function meets some conditions and the measurements are sufficiently informative, (A.165) is met regardless of the initial PDF and the transitional prior [163]. In other words, if (A.165) is met, then the filter is able to forget its initial PDF. The ability of a filter to forget its initial condition is an extensive field of research and new results are expected to appear in the near future relaxing the conditions under which the filter forgets its initial condition [54].

In the following, it is assumed that (A.165) is met even though it should be taken into account that each measurement model should be carefully analysed and that new results on the forgetfulness of the filter might appear in the near future. Under this assumption and considering (A.165), one can write

$$\pi_n^k(\mathbf{X}) \approx \pi_{\Pi c}^k(\mathbf{X}) \quad \text{if } k > k^e \quad (\text{A.166})$$

where k^e is a certain time step. Equation (A.166) can be written as

$$\pi_n^k(\Pi\mathbf{X}) \approx \pi_{\Pi c}^k(\Pi\mathbf{X}) \quad \text{if } k > k^e \quad (\text{A.167})$$

$$\pi_{\Pi n}^k(\mathbf{X}) \approx \pi_c^k(\mathbf{X}) \quad \text{if } k > k^e \quad (\text{A.168})$$

It is shown in Appendix A.19 that (A.168) implies that, as of a certain time step, the weights of the mixture in (A.162) do not change with time and one can write

$$\pi^k(\mathbf{X}^k) \approx (1 - a_c) \pi_n^k(\mathbf{X}^k) + a_c \pi_{\Pi n}^k(\mathbf{X}^k) \quad (\text{A.169})$$

where $a_c = a_c^{k^e}$. As α^k is related to a_c , when (A.169) happens, α^k has reached a steady value as well.

It should be noted that if (A.165) is not met, i.e., no steady state is achieved, something counterintuitive happens. Let us imagine that two targets are far from each other, then get in close proximity and then separate. They have been separated for a long time and now they are very far from each other. If (A.165) is not met, it implies that with each new measurement, α^k and the probability of successful labelling change accordingly and, they will never achieve a steady value. This does not seem not make sense at least in usual tracking problems.

A.18.1 Subsequent encounters

Now, let us assume the targets move again in close proximity and then separate. Equation (A.169) still holds, the only difference is that (A.164) does not hold because $\pi_n^k(\cdot)$ is multimodal in values of the form $\tilde{\mathbf{X}}_n^k$ and $\Pi \tilde{\mathbf{X}}_n^k$. Why? Let us concentrate on $\pi_n^k(\cdot)$, which is unimodal with respect to the mirror before the target crossings. Because of the same reasons mentioned above (in that case, referring to the posterior PDF), the density $\pi_n^k(\cdot)$ after a target crossing is multimodal in values $\tilde{\mathbf{X}}_n^k$ and $\Pi \tilde{\mathbf{X}}_n^k$ reaching a steady state of the form

$$\pi_n^k(\mathbf{X}^k) \approx (1 - b_c) \pi_m^k(\mathbf{X}^k) + b_c \pi_{\Pi m}^k(\mathbf{X}^k) \quad (\text{A.170})$$

Therefore, if (A.170) is substituted into (A.169), the posterior PDF becomes

$$\pi^k(\mathbf{X}^k) \approx (1 - c_c) \pi_m^k(\mathbf{X}^k) + c_c \pi_{\Pi m}^k(\mathbf{X}^k) \quad (\text{A.171})$$

where

$$c_c = b_c + a_c - 2a_c b_c \quad (\text{A.172})$$

It should be noted that $a_c \in [0, 0.5]$ and $b_c \in [0, 0.5]$. Therefore, it can be shown that $c_c \in [0, 0.5]$ and $c_c \geq a_c$ and $c_c \geq b_c$. As the probability of wrong labelling is c_c (when the targets are far from each other), it implies that after the second encounter, the probability of wrong labelling cannot decrease.

A.18.2 Conclusions

In principle, α^k could increase or decrease when targets are moving in close proximity. However, the steady state of α^k after the targets have separated is always higher than the steady state of α^k before the targets are in close proximity. This admits a very easy explanation.

Let us assume the targets are far from each other and they get in close proximity. When the targets are in close proximity, the filter might still have some memory about the labelling. Let us assume there is some confusion about the target labels ($\alpha^k > 0$) at a certain time. Nevertheless, in the next step, if a very accurate measurement is obtained, this accurate measurement might only be possible if target 1 is exclusively in one area and target 2 is exclusively in another area. That is, there is no way target 1 and target 2 are in the same area in spite of the fact that their states could have been confused in the previous time step. This is because of the memory of the filter.

When the targets separate for a long time, the filter tends to lose its memory and, therefore, a new measurement cannot provide useful information about target labelling. This happens when the posterior is of the form (A.169). However, when they get in close proximity again, the value of α^k cannot be lower than the previous steady state and the probability of wrong labelling cannot decrease as expected.

A.19 Form of the posterior after a target crossing (II)

Using (A.168) into (A.162) and particularising for time step k^e , one gets

$$\pi^{k^e}(\mathbf{X}^{k^e}) \approx (1 - a_c^{k^e}) \pi_n^{k^e}(\mathbf{X}^{k^e}) + a_c^{k^e} \pi_{\Pi n}^{k^e}(\mathbf{X}^{k^e}) \quad (\text{A.173})$$

The prior at time $k^e + 1$ is

$$\begin{aligned} \varpi^{k^e+1}(\mathbf{X}^{k^e+1}) &\approx (1 - a_c^{k^e}) \int f(\mathbf{X}^{k^e+1} | \mathbf{X}^{k^e}) \pi_n^{k^e}(\mathbf{X}^{k^e}) d\mathbf{X}^{k^e} \\ &\quad + a_c^{k^e} \int f(\mathbf{X}^{k^e+1} | \mathbf{X}^{k^e}) \pi_{\Pi n}^{k^e}(\mathbf{X}^{k^e}) d\mathbf{X}^{k^e} \\ &= (1 - a_c^{k^e}) \int f(\mathbf{X}^{k^e+1} | \mathbf{X}^{k^e}) \pi_n^{k^e}(\mathbf{X}^{k^e}) d\mathbf{X}^{k^e} \\ &\quad + a_c^{k^e} \int f(\mathbf{X}^{k^e+1} | \Pi \mathbf{X}^{k^e}) \pi_{\Pi n}^{k^e}(\Pi \mathbf{X}^{k^e}) d\mathbf{X}^{k^e} \\ &= (1 - a_c^{k^e}) \int f(\mathbf{X}^{k^e+1} | \mathbf{X}^{k^e}) \pi_n^{k^e}(\mathbf{X}^{k^e}) d\mathbf{X}^{k^e} \end{aligned}$$

$$\begin{aligned}
 & + a_c^{k^e} \int f(\Pi \mathbf{X}^{k^e+1} | \mathbf{X}^{k^e}) \pi_n^{k^e}(\mathbf{X}^{k^e}) d\mathbf{X}^{k^e} \\
 & = (1 - a_c^{k^e}) \varpi_n^{k^e+1}(\mathbf{X}^{k^e+1}) + a_c^{k^e} \varpi_{\Pi n}^{k^e+1}(\mathbf{X}^{k^e+1})
 \end{aligned} \tag{A.174}$$

where

$$\varpi_n^{k^e+1}(\mathbf{X}^{k^e+1}) = \int f(\mathbf{X}^{k^e+1} | \mathbf{X}^{k^e}) \pi_n^{k^e}(\mathbf{X}^{k^e}) d\mathbf{X}^{k^e} \tag{A.175}$$

The posterior at time $k^e + 1$ is

$$\pi^{k^e+1}(\mathbf{X}^{k^e+1}) \approx \frac{\ell(\mathbf{z}^{k^e+1} | \mathbf{X}^{k^e+1}) \varpi^{k^e+1}(\mathbf{X}^{k^e+1})}{\int \ell(\mathbf{z}^{k^e+1} | \mathbf{X}^{k^e+1}) \varpi^{k^e+1}(\mathbf{X}^{k^e+1}) d\mathbf{X}^{k^e+1}} \tag{A.176}$$

As it is met that

$$\int \ell(\mathbf{z}^{k^e+1} | \mathbf{X}^{k^e+1}) \varpi_n^{k^e+1}(\mathbf{X}^{k^e+1}) d\mathbf{X}^{k^e+1} = \int \ell(\mathbf{z}^{k^e+1} | \mathbf{X}^{k^e+1}) \varpi_{\Pi n}^{k^e+1}(\mathbf{X}^{k^e+1}) d\mathbf{X}^{k^e+1} \tag{A.177}$$

equation (A.176) becomes

$$\pi^{k^e+1}(\mathbf{X}^{k^e+1}) \approx (1 - a_c^{k^e}) \pi_n^{k^e+1}(\mathbf{X}^{k^e+1}) + a_c^{k^e} \pi_{\Pi n}^{k^e+1}(\mathbf{X}^{k^e+1}) \tag{A.178}$$

where

$$\pi_n^{k^e+1}(\mathbf{X}^{k^e+1}) = \frac{\ell(\mathbf{z}^{k^e+1} | \mathbf{X}^{k^e+1}) \varpi_n^{k^e+1}(\mathbf{X}^{k^e+1})}{\int \ell(\mathbf{z}^{k^e+1} | \mathbf{X}^{k^e+1}) \varpi_n^{k^e+1}(\mathbf{X}^{k^e+1}) d\mathbf{X}^{k^e+1}} \tag{A.179}$$

This implies that for $k > k^e$, the weights of the mixture in (A.178) do not change with time and one can write

$$\pi^k(\mathbf{X}^k) \approx (1 - a_c) \pi_n^k(\mathbf{X}^k) + a_c \pi_{\Pi n}^k(\mathbf{X}^k) \tag{A.180}$$

where $a_c = a_c^{k^e}$.

A.20 Importance density for mirror particles

In this appendix, integral (5.63) is calculated:

$$\begin{aligned}
 q(\mathbf{X}^k) & = \int \frac{1}{2} [\delta(\mathbf{X}^k - \tilde{\mathbf{X}}) + \delta(\mathbf{X}^k - \Pi \tilde{\mathbf{X}})] v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}} \\
 & = \frac{1}{2} \int \delta(\mathbf{X}^k - \tilde{\mathbf{X}}) v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}} + \frac{1}{2} \int \delta(\mathbf{X}^k - \Pi \tilde{\mathbf{X}}) v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}}
 \end{aligned}$$

$$= \frac{1}{2}v(\mathbf{X}^k) + \frac{1}{2} \int \delta(\mathbf{X}^k - \mathbf{\Pi}\tilde{\mathbf{X}}) v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}} \quad (\text{A.181})$$

It is known that [35]

$$\delta(\mathbf{X}^k - \mathbf{\Pi}\tilde{\mathbf{X}}) = \frac{\delta(\tilde{\mathbf{X}} - \mathbf{\Pi}^{-1}\mathbf{X}^k)}{|\det \mathbf{\Pi}|} \quad (\text{A.182})$$

In addition, for any permutation matrix $\mathbf{\Pi}$, it holds that $\mathbf{\Pi}^{-1} = \mathbf{\Pi}^T$ and $|\det(\mathbf{\Pi})| = 1$ where $\det(\mathbf{\Pi})$ is the determinant of $\mathbf{\Pi}$ [184] and when there are only two targets $\mathbf{\Pi}^{-1} = \mathbf{\Pi}$. Therefore,

$$\begin{aligned} q(\mathbf{X}^k) &= \frac{1}{2}v(\mathbf{X}^k) + \frac{1}{2} \int \delta(\mathbf{X}^k - \mathbf{\Pi}\tilde{\mathbf{X}}) v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}} \\ &= \frac{1}{2}v(\mathbf{X}^k) + \frac{1}{2} \int \frac{\delta(\tilde{\mathbf{X}} - \mathbf{\Pi}^{-1}\mathbf{X}^k)}{|\det \mathbf{\Pi}|} v(\tilde{\mathbf{X}}) d\tilde{\mathbf{X}} \\ &= \frac{1}{2}v(\mathbf{X}^k) + \frac{1}{2}v(\mathbf{\Pi}^{-1}\mathbf{X}^k) \\ &= \frac{1}{2}(v(\mathbf{X}^k) + v(\mathbf{\Pi}\mathbf{X}^k)) \end{aligned} \quad (\text{A.183})$$

A.21 Proof that $\hat{\alpha}^k \leq 1$ using the marginal PF

In this appendix, it is shown that if $w_i^k \geq u_i^k$ for $i = 1, \dots, N_{par}$ then $\hat{\alpha}^k \leq 1$ when the marginal PF is used. Summing over all the particles in $w_i^k \geq u_i^k$:

$$\sum_{i=1}^{N_{par}} w_i^k \geq \sum_{i=1}^{N_{par}} u_i^k \quad (\text{A.184})$$

Then,

$$\sum_{i=1}^{N_{par}} w_i^k + \sum_{i=1}^{N_{par}} u_i^k \geq \sum_{i=1}^{N_{par}} u_i^k + \sum_{i=1}^{N_{par}} u_i^k \quad (\text{A.185})$$

Using (5.59) and (5.70) in (A.185)

$$\hat{\alpha}^k \leq 1 \quad (\text{A.186})$$

A.22 Alternative importance density for the mirror PF

In Section 5.3.5.4, an importance density is proposed for the mirror PF. That importance density is the one used in the simulations of Section 5.3.7 so it appears in

the main body of this thesis. However, a simpler approach is possible.

In Section 5.3.5.4, the importance density for drawing normal particles is:

$$v(\mathbf{X}^{k+1}) \propto \sum_{i=1}^{N_{par}} \mu_i^{k+1} \frac{[w_i^k f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) + u_i^k f(\mathbf{X}^{k+1} | \Pi \mathbf{X}_i^k)]}{w_i^k + u_i^k} \quad (\text{A.187})$$

and sampling from this importance density is done following these steps:

- Sample an index j from the distribution defined by $(\mu_1^{k+1}, \dots, \mu_{N_{par}}^{k+1})$.
- Draw $\tilde{\mathbf{X}}_i^{k+1}$ from $f(\cdot | \mathbf{X}_j^{k+1})$ with probability $w_j^k / (w_j^k + u_j^k)$ or from $f(\cdot | \Pi \mathbf{X}_j^{k+1})$ with probability $u_j^k / (w_j^k + u_j^k)$.

However, as the final samples are $\tilde{\mathbf{X}}_i^{k+1}$ and the mirror $\Pi \tilde{\mathbf{X}}_i^{k+1}$, it does not matter if $\tilde{\mathbf{X}}_i^{k+1}$ is drawn from $f(\cdot | \mathbf{X}_j^{k+1})$ or $f(\cdot | \Pi \mathbf{X}_j^{k+1})$ so this way of sampling can be simplified.

Another way to explain this is to notice that the final importance density is, see (5.64)

$$\begin{aligned} q(\mathbf{X}^k) &= \frac{1}{2} (v(\mathbf{X}^k) + v(\Pi \mathbf{X}^k)) \\ &\propto \sum_{i=1}^{N_{par}} \mu_i^{k+1} (f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) + f(\mathbf{X}^{k+1} | \Pi \mathbf{X}_i^k)) \end{aligned} \quad (\text{A.188})$$

Therefore, if one uses the importance density

$$v_2(\mathbf{X}^{k+1}) \propto \sum_{i=1}^{N_{par}} \mu_i^{k+1} f(\mathbf{X}^{k+1} | \mathbf{X}_i^k) \quad (\text{A.189})$$

instead of $v(\cdot)$, the final importance density $q(\cdot)$ is the same. Sampling from $v_2(\cdot)$ is done following these steps:

- Sample an index j from the distribution defined by $(\mu_1^{k+1}, \dots, \mu_{N_{par}}^{k+1})$.
- Draw $\tilde{\mathbf{X}}_i^{k+1}$ from $f(\cdot | \mathbf{X}_j^{k+1})$.

A.23 MSLAM prediction step for the targets' positions

It was stated in Section 6.4.1 that

$$\mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,i}^{k+1|k}, \Xi_{11,j,i}^{k+1}) = \int p(\mathbf{p}_j^{k+1} | \mathbf{p}_{j,i}^k, \mathbf{v}_j^k) \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \Sigma_{j,i}^k) d\mathbf{v}_j^k \quad (\text{A.190})$$

where $\bar{\mathbf{p}}_{j,i}^{k+1|k} = \mathbf{p}_{j,i}^k + \tau \bar{\mathbf{v}}_{j,i}^k$ and $\Xi_{11,j,i}^{k+1} = \tau^2 \Sigma_{j,i}^k + \mathbf{Q}_p$. Even though this derivation is direct as it corresponds to the prediction step of a Kalman filter, it is clarified in this appendix. Using the prediction equation given by (6.1), one can write

$$\mathbf{p}_j^{k+1} = \mathbf{p}_{j,i}^k + \tau \mathbf{v}_j^k + \boldsymbol{\eta}_p \quad (\text{A.191})$$

where $\boldsymbol{\eta}_p$ is the process noise for the position elements, $\mathbf{v}_j^k$ is distributed according to $\mathcal{N}(\mathbf{v}_j^k, \bar{\mathbf{v}}_{j,i}^k, \Sigma_{j,i}^k)$ and, $\mathbf{p}_{j,i}^k$ and τ are known. According to (6.1) and (6.3), $\boldsymbol{\eta}_p$ is Gaussian distributed with zero mean and covariance matrix

$$\mathbf{Q}_p = \sigma_u^2 \frac{\tau^3}{3} \mathbf{I}_2 \quad (\text{A.192})$$

As all the relationships in (A.191) are linear and Gaussian, $\mathbf{p}_j^{k+1}$ is Gaussian with mean and covariance matrix given by

$$\begin{aligned} \mathbb{E}[\mathbf{p}_j^{k+1}] &= \mathbb{E}[\mathbf{p}_{j,i}^k + \tau \mathbf{v}_j^k + \boldsymbol{\eta}_p] \\ &= \mathbf{p}_{j,i}^k + \tau \bar{\mathbf{v}}_{j,i}^k \end{aligned} \quad (\text{A.193})$$

$$\text{cov}[\mathbf{p}_j^{k+1}] = \tau^2 \Sigma_{j,i}^k + \mathbf{Q}_p \quad (\text{A.194})$$

A.24 MSLAM: Approximation of the first-stage weights using the unscented transformation

This appendix shows how to approximate the integral that appears in the calculation of the first-stage weights, given by (6.26), using the unscented transformation [100]. That is, the aim is to approximate the integral:

$$\int p(\mathbf{z}_r^{k+1} | \mathbf{p}_j^{k+1}, \hat{\mathbf{P}}_{-\{j\}}^{k+1|k}, \mathbf{m}_r) \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_r^k, \boldsymbol{\Theta}_r^k) d\mathbf{m}_r \quad (\text{A.195})$$

This integral can be written equivalently as

$$f(\mathbf{z}_r^{k+1}; j) = \int p(\mathbf{z}_r^{k+1} | \mathbf{m}_r; j) p(\mathbf{m}_r) d\mathbf{m}_r \quad (\text{A.196})$$

where $p(\mathbf{m}_r) = \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_r^k, \boldsymbol{\Theta}_r^k)$ and, with a slight abuse of notation, $p(\mathbf{z}_r^{k+1} | \mathbf{m}_r; j)$ denotes the PDF of the measurement parameterised by the j th particle. Note that

$p(\mathbf{z}_r^{k+1} | \mathbf{m}_r; j) = p(\mathbf{z}_r^{k+1} | \mathbf{p}_j^{k+1}, \hat{\mathbf{P}}_{-\{j\}}^{k+1|k}, \mathbf{m}_r)$ and is characterised by

$$\mathbf{z}_r^{k+1} = \mathbf{h}_j(\mathbf{m}_r) + \boldsymbol{\eta} \quad (\text{A.197})$$

where $\mathbf{h}_j(\cdot)$ is the measurement function of sensor r as a function of the sensor position particularised for $\mathbf{p}_j^{k+1}$ and $\hat{\mathbf{P}}_{-\{j\}}^{k+1|k}$ and $\boldsymbol{\eta}$ is the measurement noise.

It should be noted that $f(\mathbf{z}_r^{k+1}; j)$ is a PDF whose first two moments are given by

$$\mathbb{E}[\mathbf{z}_r^{k+1}; j] = \int \mathbf{z}_r^{k+1} f(\mathbf{z}_r^{k+1}; j) d\mathbf{z}_r^{k+1} \quad (\text{A.198})$$

$$= \int \mathbb{E}[\mathbf{z}_r^{k+1} | \mathbf{m}_r; j] p(\mathbf{m}_r) d\mathbf{m}_r \quad (\text{A.199})$$

$$= \int \mathbf{h}_j(\mathbf{m}_r) p(\mathbf{m}_r) d\mathbf{m}_r \quad (\text{A.200})$$

$$\text{cov}[\mathbf{z}_r^{k+1}; j] = \int (\mathbf{h}_j(\mathbf{m}_r) - \mathbb{E}[\mathbf{z}_r^{k+1}; j]) (\mathbf{h}_j(\mathbf{m}_r) - \mathbb{E}[\mathbf{z}_r^{k+1}; j])^T p(\mathbf{m}_r) d\mathbf{m}_r + \mathbf{R} \quad (\text{A.201})$$

where $\mathbf{R}$ is the covariance matrix of the measurement noise.

The expectations (A.200) and (A.201) can be approximated using the unscented transformation selecting $N_s = 2 \cdot 2 + 1$ sigma points $(\mathcal{M}_r^0, \dots, \mathcal{M}_r^4)$ with weights $(w^0, \dots, w^4)$, where it has been used that the position of the sensor has dimension 2, as indicated in [100] or Section 2.6:

$$\mathbb{E}[\mathbf{z}_r^{k+1}; j] \approx \sum_{i=0}^4 w^i \mathbf{h}_j(\mathcal{M}_r^i) \quad (\text{A.202})$$

$$\text{cov}[\mathbf{z}_r^{k+1}; j] \approx \sum_{i=0}^4 w^i (\mathbf{h}_j(\mathcal{M}_r^i) - \mathbb{E}[\mathbf{z}_r^{k+1}; j]) (\mathbf{h}_j(\mathcal{M}_r^i) - \mathbb{E}[\mathbf{z}_r^{k+1}; j])^T + \mathbf{R} \quad (\text{A.203})$$

Lastly, the value of the integral is approximated by

$$f(\mathbf{z}_r^{k+1}; j) \approx \mathcal{N}(\mathbf{z}_r^{k+1}; \mathbb{E}[\mathbf{z}_r^{k+1}; j], \text{cov}[\mathbf{z}_r^{k+1}; j]) \quad (\text{A.204})$$

A.25 MSLAM: Update of the targets' velocities

In this appendix, an important result used to calculate $p(\mathbf{V}^{k+1} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k})$ based on $p(\mathbf{V}^k | \mathbf{P}^k, \mathbf{z}^{1:k})$ is provided:

$$\begin{aligned}
 p(\mathbf{V}^{k+1} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) &= \int p(\mathbf{V}^{k+1}, \mathbf{X}^k | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) d\mathbf{X}^k \\
 &= \int p(\mathbf{V}^{k+1} | \mathbf{P}^{k+1}, \mathbf{X}^k) p(\mathbf{X}^k | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) d\mathbf{X}^k \\
 &= \int \prod_{j=1}^t p(\mathbf{v}_j^{k+1} | \mathbf{p}_j^{k+1}, \mathbf{x}_j^k) \frac{p(\mathbf{P}^{k+1} | \mathbf{X}^k)}{p(\mathbf{P}^{k+1} | \mathbf{z}^{1:k})} p(\mathbf{X}^k | \mathbf{z}^{1:k}) d\mathbf{X}^k \quad (\text{A.205})
 \end{aligned}$$

Substituting (6.18) into (A.205):

$$\begin{aligned}
 p(\mathbf{V}^{k+1} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) &\propto \\
 \int \prod_{j=1}^t p(\mathbf{v}_j^{k+1} | \mathbf{p}_j^{k+1}, \mathbf{x}_j^k) \prod_{j=1}^t p(\mathbf{p}_j^{k+1} | \mathbf{x}_j^k) \cdot \prod_{j=1}^t \sum_{i=1}^{N_{par}} \delta(\mathbf{p}_j^k - \mathbf{p}_{j,i}^k) p(\mathbf{v}_j^k | \mathbf{p}_{j,i}^k, \mathbf{z}^{1:k}) d\mathbf{X}^k \quad (\text{A.206})
 \end{aligned}$$

$$\begin{aligned}
 &= \prod_{j=1}^t \sum_{i=1}^{N_{par}} \int p(\mathbf{v}_j^{k+1} | \mathbf{p}_j^{k+1}, \mathbf{p}_{j,i}^k, \mathbf{v}_j^k) \cdot p(\mathbf{p}_j^{k+1} | \mathbf{p}_{j,i}^k, \mathbf{v}_j^k) p(\mathbf{v}_j^k | \mathbf{p}_{j,i}^k, \mathbf{z}^{1:k}) d\mathbf{v}_j^k \quad (\text{A.207})
 \end{aligned}$$

$$\begin{aligned}
 &= \prod_{j=1}^t \sum_{i=1}^{N_{par}} \int p(\mathbf{p}_j^{k+1}, \mathbf{v}_j^{k+1} | \mathbf{p}_{j,i}^k, \mathbf{v}_j^k) \cdot p(\mathbf{v}_j^k | \mathbf{p}_{j,i}^k, \mathbf{z}^{1:k}) d\mathbf{v}_j^k \quad (\text{A.208})
 \end{aligned}$$

$$\begin{aligned}
 &= \prod_{j=1}^t \sum_{i=1}^{N_{par}} \mathcal{N}(\mathbf{u}_j^{k+1}; \bar{\mathbf{u}}_{j,i}^{k+1}, \Xi_{j,i}^{k+1}) \quad (\text{A.209})
 \end{aligned}$$

where $\mathbf{u}_j^{k+1} = [\mathbf{p}_j^{k+1}, \mathbf{v}_j^{k+1}]^T$ and $\bar{\mathbf{u}}_{j,i}^{k+1}$ and $\Xi_{j,i}^{k+1}$ are given by (6.30) and (6.31), respectively, and we have used (6.1) and (6.19). It should be noted that $\mathbf{u}_j^{k+1}$ is the vector $\mathbf{x}_j^{k+1}$ but swapping its components so that it is easier to express the result of integral (A.208), which corresponds with a prediction step.

A.26 MSLAM: Map mixing step

In this appendix, we provide a recursive way to calculate $p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k})$ based on $p(\mathbf{m} | \mathbf{P}^k, \mathbf{z}^{1:k})$:

$$\begin{aligned}
p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) &= \int p(\mathbf{m}, \mathbf{X}^k | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) d\mathbf{X}^k \\
&= \int \frac{p(\mathbf{P}^{k+1} | \mathbf{m}, \mathbf{X}^k, \mathbf{z}^{1:k})}{p(\mathbf{P}^{k+1} | \mathbf{z}^{1:k})} p(\mathbf{m}, \mathbf{X}^k | \mathbf{z}^{1:k}) d\mathbf{X}^k \\
&\propto \int p(\mathbf{P}^{k+1} | \mathbf{X}^k) p(\mathbf{m}, \mathbf{X}^k | \mathbf{z}^{1:k}) d\mathbf{X}^k
\end{aligned} \tag{A.210}$$

Substituting (6.14) and (6.15) into (A.210):

$$\begin{aligned}
p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) &\propto \int p(\mathbf{P}^{k+1} | \mathbf{X}^k) \sum_{i=1}^{N_{par}} \delta(\mathbf{P}^k - \mathbf{P}_i^k) \cdot \\
&\quad \cdot \prod_{j=1}^t \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \Sigma_{j,i}^k) \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \Theta_{r,i}^k) d\mathbf{P}^k d\mathbf{V}^k \\
&= \int \sum_{i=1}^{N_{par}} p(\mathbf{P}^{k+1} | \mathbf{P}_i^k, \mathbf{V}^k) \prod_{j=1}^t \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \Sigma_{j,i}^k) \cdot \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \Theta_{r,i}^k) d\mathbf{V}^k
\end{aligned} \tag{A.211}$$

Using the fact that the dynamics of the targets are independent, equation (6.1):

$$\begin{aligned}
p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) &\propto \sum_{i=1}^{N_{par}} \prod_{j=1}^t \int p(\mathbf{P}_j^{k+1} | \mathbf{p}_{j,i}^k, \mathbf{v}_j^k) \mathcal{N}(\mathbf{v}_j^k; \bar{\mathbf{v}}_{j,i}^k, \Sigma_{j,i}^k) d\mathbf{v}_j^k \\
&\quad \cdot \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \Theta_{r,i}^k)
\end{aligned} \tag{A.212}$$

Substituting (6.21) into (A.212) gives

$$\begin{aligned}
p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) &\propto \sum_{i=1}^{N_{par}} \prod_{j=1}^t \mathcal{N}(\mathbf{p}_j^{k+1}; \bar{\mathbf{p}}_{j,i}^{k+1|k}, \Xi_{11,j,i}^{k+1}) \cdot \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \Theta_{r,i}^k) \\
&= \sum_{i=1}^{N_{par}} \tilde{\rho}_{2,i}(\mathbf{P}^{k+1}) \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \Theta_{r,i}^k)
\end{aligned} \tag{A.213}$$

where $\tilde{\rho}_{2,i}(\mathbf{P}^{k+1})$ is given by (6.42).

Then, (A.213) can be written as:

$$p(\mathbf{m} | \mathbf{P}^{k+1}, \mathbf{z}^{1:k}) = \sum_{i=1}^{N_{par}} \rho_{2,i}(\mathbf{P}^{k+1}) \prod_{r=1}^M \mathcal{N}(\mathbf{m}_r; \bar{\mathbf{m}}_{r,i}^k, \boldsymbol{\Theta}_{r,i}^k) \quad (\text{A.214})$$

where $\rho_{2,i}(\mathbf{P}^{k+1})$ are the normalised $\tilde{\rho}_{2,i}(\mathbf{P}^{k+1})$ so that they sum to one.

A.27 MSLAM: Jacobian of the measurement function

The Jacobian of the measurement function is needed to apply the TUKF and SP-TUKF to update the positions of the sensors. The matrix $\tilde{\mathbf{H}}$ has the form:

$$\tilde{\mathbf{H}} = \begin{pmatrix} \frac{\partial z_1}{\partial m_x} & \frac{\partial z_2}{\partial m_x} & \cdots & \frac{\partial z_{N_a}}{\partial m_x} \\ \frac{\partial z_1}{\partial m_y} & \frac{\partial z_2}{\partial m_y} & \cdots & \frac{\partial z_{N_a}}{\partial m_y} \end{pmatrix}^T \quad (\text{A.215})$$

where $\mathbf{z} = [z_1, z_2, \dots, z_{N_a}]^T$ and $\mathbf{m} = [m_x, m_y]^T$ denote the measurement and position of the sensor that is being updated, respectively. The partial derivative of the measurement in the j th antenna with respect to m_x is:

$$\frac{\partial z_j}{\partial m_x} = \frac{1}{\bar{z}_j} \sum_{i=1}^t \frac{SNR_0 d_0^2 \exp(-\beta (g(\vartheta_i, \theta_j))^2)}{d_i^4} (x_i - m_x - \beta g(\vartheta_i, \theta_j) (y_i - m_y)) \quad (\text{A.216})$$

where $[x_i, y_i]^T$ are the coordinates of target i , ϑ_i is the angle formed with the target in relation to sensor j , d_i is the distance from the target i to the sensor, $\bar{z}_j$ is the expected value of the measurement in the j th antenna for the current targets and $g(\cdot, \cdot)$ is given by (6.7). The partial derivative of the measurement in the j th antenna with respect to m_y is:

$$\frac{\partial z_j}{\partial m_y} = \frac{1}{\bar{z}_j} \sum_{i=1}^t \frac{SNR_0 d_0^2 \exp(-\beta (g(\vartheta_i, \theta_j))^2)}{d_i^4} (y_i - m_y + \beta g(\vartheta_i, \theta_j) (x_i - m_x)) \quad (\text{A.217})$$

A.28 MSLAM-TUKF: moments of the truncated prior for overdetermined measurement equations

In this appendix, an expression for the mean and covariance matrix of the truncated prior is given when the measurement equation is overdetermined. This problem arises in Chapter 6 in the MSLAM problem.

The measurement equation can be written in a simplified way as

$$\mathbf{z} = \mathbf{h}(\mathbf{m}) + \boldsymbol{\eta} \quad (\text{A.218})$$

where $\mathbf{z} = [z_1, z_2, \dots, z_{N_a}]^T$ and $\mathbf{m} = [m_x, m_y]^T$ denote the measurement and position of the sensor that is being updated, respectively, $\mathbf{h}(\cdot)$ is the measurement function that includes the multitarget state and $\boldsymbol{\eta}$ is the measurement noise such that $\mathbb{E}[\boldsymbol{\eta}] = 0$ and $\text{cov}[\boldsymbol{\eta}] = \mathbf{R}$ which is approximated by a truncated one with support I_η .

Using a first order Taylor series expansion around $\tilde{\mathbf{m}}(\mathbf{z})$, (A.218) can be written as

$$\mathbf{z} \approx \mathbf{h}(\tilde{\mathbf{m}}(\mathbf{z})) + \tilde{\mathbf{H}}(\mathbf{m} - \tilde{\mathbf{m}}(\mathbf{z})) + \boldsymbol{\eta} \quad (\text{A.219})$$

where $\tilde{\mathbf{H}}$ is given in (A.27) and $\tilde{\mathbf{m}}(\mathbf{z})$ is the value that maximises the likelihood, see Section 3.4.3.1. It should be recalled that $\mathbf{z}$ and $\tilde{\mathbf{m}}(\mathbf{z})$ are known vectors and the aim is to calculate the first two moments of $\mathbf{m}$. For Gaussian noise, $\tilde{\mathbf{m}}(\mathbf{z})$ minimises $(\mathbf{z} - \mathbf{h}(\mathbf{m}))^T \mathbf{R}^{-1} (\mathbf{z} - \mathbf{h}(\mathbf{m}))$. Therefore, the following approximation is used

$$\mathbf{z} \approx \mathbf{h}(\tilde{\mathbf{m}}(\mathbf{z})) \quad (\text{A.220})$$

Substituting (A.220) into (A.219)

$$-\boldsymbol{\eta} \approx \tilde{\mathbf{H}}(\mathbf{m} - \tilde{\mathbf{m}}(\mathbf{z})) \quad (\text{A.221})$$

which is an overdetermined equation and the least squares solution can be used [174] to provide

$$\mathbf{m} \approx \tilde{\mathbf{m}}(\mathbf{z}) - \tilde{\mathbf{H}}^+ \boldsymbol{\eta} \quad (\text{A.222})$$

where $\tilde{\mathbf{H}}^+ = (\tilde{\mathbf{H}}^T \tilde{\mathbf{H}})^{-1} \tilde{\mathbf{H}}^T$ is the Moore-Penrose pseudoinverse. Therefore, the support of the truncated prior is approximated as

$$I_{\mathbf{m}}(\mathbf{z}) \approx \left\{ \mathbf{m} \mid \mathbf{m} = \tilde{\mathbf{m}}(\mathbf{z}) - \tilde{\mathbf{H}}^+ \boldsymbol{\eta}, \boldsymbol{\eta} \in I_\eta \right\} \quad (\text{A.223})$$

The only difference between the support of the truncated prior for overdetermined measurement functions and bijective ones is the use of $\tilde{\mathbf{H}}^+$ instead of $\tilde{\mathbf{H}}^{-1}$, see (3.30). Therefore, using the same calculations as in Section 3.4.3.1, the approximation to the mean $\hat{\mathbf{m}}_{p,1}$ and covariance matrix $\mathbf{P}_{p,1}$ of the truncated prior can be obtained substituting $\tilde{\mathbf{H}}^+$ for $\tilde{\mathbf{H}}^{-1}$ in (3.34) and (3.35):

$$\hat{\mathbf{m}}_{p,1} = \tilde{\mathbf{m}}(\mathbf{z}) \quad (\text{A.224})$$

$$\mathbf{P}_{p,1} = \tilde{\mathbf{H}}^+ \mathbf{R} \left(\tilde{\mathbf{H}}^+ \right)^T \quad (\text{A.225})$$

Considering the fact that $\mathbf{R} = \mathbf{I}$ in the MSLAM problem, (A.225) becomes

$$\mathbf{P}_{p,1} = \left(\tilde{\mathbf{H}}^T \tilde{\mathbf{H}} \right)^{-1} \quad (\text{A.226})$$

A.29 Asynchronous PF: Approximation of the asynchronous likelihood

This appendix gives an approximation of the PDF

$$p \left(\mathbf{z}^{kT_u} \mid \mathbf{P}^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u} \right) \quad (\text{A.227})$$

Integrating over all possible positions of the targets at each sampling time, (A.227) can be expressed as:

$$\begin{aligned} & p \left(\mathbf{z}^{kT_u} \mid \mathbf{P}^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u} \right) \\ &= \int \dots \int p \left(\mathbf{z}^{kT_u} \mid \mathbf{P}^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}, \mathbf{P}^{\tau_1^k}, \mathbf{P}^{\tau_2^k} \dots \mathbf{P}^{\tau_{M^k}^k} \right) \\ & \quad \cdot p \left(\mathbf{P}^{\tau_1^k}, \mathbf{P}^{\tau_2^k} \dots \mathbf{P}^{\tau_{M^k}^k} \mid \mathbf{P}^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u} \right) d\mathbf{P}^{\tau_1^k} d\mathbf{P}^{\tau_2^k} \dots d\mathbf{P}^{\tau_{M^k}^k} \end{aligned} \quad (\text{A.228})$$

Since the process is Markovian and the measurements at each time are independent conditioned on the positions of the targets at that time, equation (A.228) can be written as:

$$\begin{aligned} & p \left(\mathbf{z}^{kT_u} \mid \mathbf{P}^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u} \right) \\ &= \int \dots \int \prod_{j=1}^{M^k} p \left(z_j^{\tau_j^k} \mid \mathbf{P}^{\tau_j^k} \right) \\ & \quad \cdot p \left(\mathbf{P}^{\tau_j^k} \mid \mathbf{P}^{kT_u}, \mathbf{P}^{\tau_{j-1}^k}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u} \right) d\mathbf{P}^{\tau_1^k} d\mathbf{P}^{\tau_2^k} \dots d\mathbf{P}^{\tau_{M^k}^k} \end{aligned} \quad (\text{A.229})$$

where it has been considered that $\tau_0^k = (k-1)T_u$. Equation (A.229) points out that theoretically the PDF given by (A.227) could not be written as the product of the PDFs of the components of $\mathbf{z}^{kT_u}$, i.e., the measurements are correlated when conditioned on the parameters given in (A.227). However, evaluating the integral in (A.229) is time-consuming as it is a multivariate integral of high dimension.

Therefore, (A.227) is approximated as if the measurements were independent:

$$p\left(\mathbf{z}^{kT_u} \mid \mathbf{P}^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}\right) \approx \prod_{j=1}^{M^k} p\left(z_j^k \mid \mathbf{P}^{kT_u}, \mathbf{P}_i^{(k-1)T_u}, \bar{\mathbf{V}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}\right) \quad (\text{A.230})$$

Even though (A.230) is an approximation as the correlations among the measurements are neglected, it is demonstrated with numerical simulations that the performance of the asynchronous filter is close to the traditional particle filter but with a considerably reduced computational burden.

A.30 Asynchronous PF: Calculation of the PDF of a target position at a sampling time instant

This appendix gives the expression of the PDF

$$p\left(\mathbf{p}_i^{\tau_j^k} \mid \mathbf{p}_i^{kT_u}, \mathbf{p}_i^{(k-1)T_u}, \bar{\mathbf{v}}_i^{(k-1)T_u}, \boldsymbol{\Sigma}_i^{(k-1)T_u}\right)$$

that appears in (7.17). Using (7.3), the variable $\mathbf{p}_i^{\tau_j^k}$ is related to the known variables by the equations:

$$\mathbf{p}_i^{kT_u} = \mathbf{p}_i^{\tau_j^k} + T_2 \mathbf{v}_i^{\tau_j^k} + \mathbf{w}_{p,2} \quad (\text{A.231})$$

$$\mathbf{p}_i^{\tau_j^k} = \mathbf{p}_i^{(k-1)T_u} + T_1 \mathbf{v}_i^{(k-1)T_u} + \mathbf{w}_{p,1} \quad (\text{A.232})$$

$$\mathbf{v}_i^{\tau_j^k} = \mathbf{v}_i^{(k-1)T_u} + \mathbf{w}_{v,1} \quad (\text{A.233})$$

where $T_1 = \tau_j^k - (k-1)T_u$, $T_2 = kT_u - \tau_j^k$, $\mathbf{w}_{p,1}$ and $\mathbf{w}_{v,1}$ are the accumulated process noises of the position and the velocity in the interval T_1 , and $\mathbf{w}_{p,2}$ is the accumulated process noise of the position in the interval T_2 . The process noises of the position and velocity that correspond to the same interval are correlated according to (7.5). Substituting (A.233) in (A.231):

$$\mathbf{p}_i^{kT_u} = \mathbf{p}_i^{\tau_j^k} + T_2 \mathbf{v}_i^{(k-1)T_u} + T_2 \mathbf{w}_{v,1} + \mathbf{w}_{p,2} \quad (\text{A.234})$$

Substituting (A.232) in (A.234):

$$\mathbf{p}_i^{kT_u} = \mathbf{p}_i^{(k-1)T_u} + (T_1 + T_2) \mathbf{v}_i^{(k-1)T_u} + \mathbf{w}_{p,1} + T_2 \mathbf{w}_{v,1} + \mathbf{w}_{p,2} \quad (\text{A.235})$$

The variable $\mathbf{p}_i^{\tau_j^k}$, given $\mathbf{p}_i^{kT_u}$, $\mathbf{p}_i^{(k-1)T_u}$, $\bar{\mathbf{v}}_i^{(k-1)T_u}$ and $\boldsymbol{\Sigma}_i^{(k-1)T_u}$, is Gaussian be-

cause all the terms involved are Gaussian and the equations are linear. Its mean and covariance matrix are given by the fundamental equation of linear estimation [13]:

$$\mathbb{E} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:kT_u} \right] = \mathbb{E} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right] + \mathbf{\Psi} \mathbf{S}^{-1} \left(\mathbf{p}_i^{kT_u} - \mathbb{E} \left[\mathbf{p}_i^{kT_u} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right] \right) \quad (\text{A.236})$$

$$\text{cov} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:kT_u} \right] = \text{cov} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right] - \mathbf{\Psi} \mathbf{S}^{-1} \mathbf{\Psi}^t \quad (\text{A.237})$$

where

$$\mathbf{S} = \text{cov} \left[\mathbf{p}_i^{kT_u} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right]$$

$$\mathbf{\Psi} = \text{cov} \left[\mathbf{p}_i^{\tau_j^k}, \mathbf{p}_i^{kT_u} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right]$$

Now, all the terms in (A.236) and (A.237) are calculated.

- Term 1: Using (A.232):

$$\mathbb{E} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right] = \mathbf{p}_i^{(k-1)T_u} + T_1 \bar{\mathbf{v}}_i^{(k-1)T_u} \quad (\text{A.238})$$

- Term 2: Using (A.232), (A.238) and (A.235)

$$\mathbf{\Psi} = T_1 (T_1 + T_2) \mathbf{\Sigma}_i^{(k-1)T_u} + \mathbf{Q}_{p,1} + T_2 \mathbf{Q}_{pv,1} \quad (\text{A.239})$$

where $\mathbf{Q}_{p,1}$ and $\mathbf{Q}_{pv,1}$ are the covariances matrices of the process noise of the position and the velocity in the interval T_1 , respectively.

- Term 3: Using (A.235):

$$\mathbf{S} = (T_1 + T_2)^2 \mathbf{\Sigma}_i^{(k-1)T_u} + \mathbf{Q}_{p,1} + T_2 \mathbf{Q}_{pv,1} + T_2 \mathbf{Q}_{vp,1} + (T_2)^2 \mathbf{Q}_{v,1} + \mathbf{Q}_{p,2} \quad (\text{A.240})$$

- Term 4: Using (A.235):

$$\mathbb{E} \left[\mathbf{p}_i^{kT_u} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right] = \mathbf{p}_i^{(k-1)T_u} + (T_1 + T_2) \bar{\mathbf{v}}_i^{(k-1)T_u} \quad (\text{A.241})$$

- Term 5: Using (A.232):

$$\text{cov} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:(k-1)T_u} \right] = (T_1)^2 \mathbf{\Sigma}_i^{(k-1)T_u} + \mathbf{Q}_{p,1} \quad (\text{A.242})$$

Therefore,

$$p \left(\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{kT_u}, \mathbf{p}_i^{(k-1)T_u}, \bar{\mathbf{v}}_i^{(k-1)T_u}, \mathbf{\Sigma}_i^{(k-1)T_u} \right) =$$

$$\mathcal{N} \left(\mathbf{p}_i^{\tau_j^k}; \mathbb{E} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:kT_u} \right], \text{cov} \left[\mathbf{p}_i^{\tau_j^k} \middle| \mathbf{p}_i^{0:kT_u} \right] \right) \quad (\text{A.243})$$

Appendix B

Comparison between set and vector notations for tracking

In this appendix, a comparison between the common notations of sets and vectors for representing the multitarget state for tracking is given. The main conclusion is that, as there is a bijection between labelled sets and labelled vectors, both notations lead to the same results. Firstly, a situation with fixed and known number of targets will be presented. Secondly, the results will be generalised for variable and unknown number of targets.

B.1 Fixed and known number of targets

When the multitarget state is represented by a vector, the order inherent in vector notation implies that targets are labelled and, therefore, one can construct tracks from a purely Bayesian point of view, i.e., one can infer each target state along time. Set notation (using only the kinematic state in the elements of the set as is usually the case) implies that the targets are not labelled and, therefore, the only information of interest is to know where the targets are at each time without accounting for their labels. In fact, tracks cannot be built from a purely Bayesian point of view although ad hoc approaches can be used. Set notation can handle the problem of labelling if labels are explicitly included in each target state.

There are several issues that are worth highlighting about labelling:

1. Labels are not parameters in the usual sense. For fixed and known number of targets, they are assigned deterministically at the beginning of the simulation. For unknown and variable number of targets, they are assigned deterministically when a target is born. In addition, labels do not change their value along

time and they are unique in the sense that there cannot be two targets with the same label. Therefore, they do not behave as normal parameters that have to be estimated. Their function is to allow the creation of tracks for unknown and variable number of targets and also for fixed and known number of targets using set notation.

2. There is a bijection between the multitarget state vector and the multitarget set with labels. That is, once the labels have been included in the target states, they always retain the same value along time and, therefore, this implies that there is an ordering in the set that can be perfectly represented by a vector. For example, if one target is labelled with label “1” and another with label “2”, one can sort this set into a vector in which the state of the target with label “1” is represented by the first components of the multitarget vector and the state of target “2” by the last components. It should be noted that this ordering is always possible as the labels are unique.
3. As a consequence of 2., it is not surprising that the posterior PDF that represents the multitarget state with a set with labels or a vector are equivalent in the sense that they contain the same information. However, the posterior PDF using set notation is represented with a more cumbersome notation as it includes Dirac delta functions representing the labels plus another terms that permute targets “1” and “2” in the posterior PDF. This will be clarified in the next sections.
4. The posterior PDF is calculated recursively using the prediction and update steps. The update step consists of applying the product of the likelihood and the prior plus a normalisation. This step is basically the same for set and vector notation as the information obtained from the measurement is accounted for by a product, which is the same operation for both representations. However, the prediction step is different for both notations. Set notation uses set integrals while vector notation uses vector integrals. Set integrals are much more complex than vector integrals as they involve the calculation of vector integrals for all possible orderings of the set. However, as a consequence of 2., set integrals applied to labelled sets provide the same results as vector integrals applied to vectors as will be indicated in the next sections.

Therefore, vector notation seems to have an advantage over (labelled) set notation for fixed and known number of targets as one achieves the same results without using complex set integrals. This is explained in the following sections that analyse the prediction step for different notations. The results presented here can be generalised

for tracking a fixed and known number of targets but, for simplicity, it is assumed that there are only two targets.

B.1.1 Posterior PDF representation

Let us assume target “1” state is $\mathbf{x}^k$ and target “2” state is $\mathbf{y}^k$. Then, the multitarget state vector is $[\mathbf{x}^k, \mathbf{y}^k]$ and its corresponding set $\{\mathbf{x}^k, \mathbf{y}^k\}$. Note that the multitarget state vector should be written as $\left[(\mathbf{x}^k)^T, (\mathbf{y}^k)^T\right]^T$ instead of $[\mathbf{x}^k, \mathbf{y}^k]$. However, to simplify notation, $[\mathbf{x}^k, \mathbf{y}^k]$ will be used in this section.

The vector posterior PDF is

$$\pi^k([\mathbf{x}^k, \mathbf{y}^k]) \quad (\text{B.1})$$

and the corresponding set posterior PDF is [83]

$$\pi^k(\{\mathbf{x}^k, \mathbf{y}^k\}) = \pi^k([\mathbf{x}^k, \mathbf{y}^k]) + \pi^k([\mathbf{y}^k, \mathbf{x}^k]) \quad (\text{B.2})$$

Now the state is augmented with the labels. Therefore, the vector becomes $[\mathbf{x}^k, i_x^k, \mathbf{y}^k, i_y^k] = [\mathbf{x}^k, 1, \mathbf{y}^k, 2]$ and the set $\{[\mathbf{x}^k, i_x^k], [\mathbf{y}^k, i_y^k]\} = \{[\mathbf{x}^k, 1], [\mathbf{y}^k, 2]\}$ where it has been assumed without loss of generality that the labels are “1” and “2” as the specific values of the labels are unimportant. The vector density is

$$\pi^k([\mathbf{x}^k, i_x^k, \mathbf{y}^k, i_y^k]) = \pi^k([\mathbf{x}^k, \mathbf{y}^k]) \delta(i_x^k - 1) \delta(i_y^k - 2) \quad (\text{B.3})$$

which is equivalent to (B.1) but adding Dirac delta functions for the labels. It should be noted that the labels do not add any extra information. The set density with the labels becomes [83]

$$\begin{aligned} \pi^k(\{[\mathbf{x}^k, i_x^k], [\mathbf{y}^k, i_y^k]\}) &= \pi^k([\mathbf{x}^k, i_x^k, \mathbf{y}^k, i_y^k]) + \pi^k([\mathbf{y}^k, i_y^k, \mathbf{x}^k, i_x^k]) \\ &= \pi^k([\mathbf{x}^k, \mathbf{y}^k]) \delta(i_x^k - 1) \delta(i_y^k - 2) \\ &\quad + \pi^k([\mathbf{y}^k, \mathbf{x}^k]) \delta(i_y^k - 1) \delta(i_x^k - 2) \end{aligned} \quad (\text{B.4})$$

Note that the information using set notation with labels is the same as using vector notation. If labels are used, the posterior using labelled set notation will always be of the form given by (B.4) in which the labels are represented by Dirac delta functions. Therefore, if one wanted to extract target labelling information from the posterior PDF using set notation, it should be done decomposing one of the vector densities that appear in (B.4) using the same procedure as in Chapter 5

for vector notation.

B.1.2 Prediction phase: vector notation without labels

In vector notation, the prior at time $k + 1$ is

$$\varpi^{k+1}([\mathbf{x}^{k+1}, \mathbf{y}^{k+1}]) = \int f([\mathbf{x}^{k+1}, \mathbf{y}^{k+1}] | [\mathbf{x}^k, \mathbf{y}^k]) \pi^k([\mathbf{x}^k, \mathbf{y}^k]) d\mathbf{x}^k d\mathbf{y}^k \quad (\text{B.5})$$

B.1.3 Prediction phase: vector notation with labels

The transition density with labels is

$$\begin{aligned} & f([\mathbf{x}^{k+1}, i_x^{k+1}, \mathbf{y}^{k+1}, i_y^{k+1}] | [\mathbf{x}^k, i_x^k, \mathbf{y}^k, i_y^k]) \\ &= f([\mathbf{x}^{k+1}, \mathbf{y}^{k+1}] | [\mathbf{x}^k, \mathbf{y}^k]) \delta(i_x^{k+1} - i_x^k) \delta(i_y^{k+1} - i_y^k) \end{aligned} \quad (\text{B.6})$$

where it has been used that the labels do not change over time. Therefore, the prior at time $k + 1$ with labels using vector notation is

$$\begin{aligned} \varpi^{k+1}([\mathbf{x}^{k+1}, i_x^{k+1}, \mathbf{y}^{k+1}, i_y^{k+1}]) &= \int f([\mathbf{x}^{k+1}, \mathbf{y}^{k+1}] | [\mathbf{x}^k, \mathbf{y}^k]) \delta(i_x^{k+1} - i_x^k) \delta(i_y^{k+1} - i_y^k) \\ &\quad \cdot \pi^k([\mathbf{x}^k, \mathbf{y}^k]) \delta(i_x^k - 1) \delta(i_y^k - 2) d\mathbf{x}^k d\mathbf{y}^k di_x^k di_y^k \end{aligned} \quad (\text{B.7})$$

$$\begin{aligned} &= \int f([\mathbf{x}^{k+1}, \mathbf{y}^{k+1}] | [\mathbf{x}^k, \mathbf{y}^k]) \pi^k([\mathbf{x}^k, \mathbf{y}^k]) \delta(i_x^{k+1} - 1) \delta(i_y^{k+1} - 2) d\mathbf{x}^k d\mathbf{y}^k \\ &\quad (\text{B.8}) \end{aligned}$$

$$= \varpi^{k+1}([\mathbf{x}^{k+1}, \mathbf{y}^{k+1}]) \delta(i_x^{k+1} - 1) \delta(i_y^{k+1} - 2) \quad (\text{B.9})$$

B.1.4 Prediction phase: set notation without labels

In set notation, the prior at time $k + 1$ is

$$\varpi^{k+1}(\{\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\}) = \int f(\{\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\} | \{\mathbf{x}^k, \mathbf{y}^k\}) \pi^k(\{\mathbf{x}^k, \mathbf{y}^k\}) \partial \{\mathbf{x}^k, \mathbf{y}^k\} \quad (\text{B.10})$$

Using equation (14.14) in [131]

$$\varpi^{k+1}(\{\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\}) = \frac{1}{2} \int f(\{\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\} | \{\mathbf{x}^k, \mathbf{y}^k\}) \pi^k(\{\mathbf{x}^k, \mathbf{y}^k\}) d\mathbf{x}^k d\mathbf{y}^k \quad (\text{B.11})$$

As indicated in equation (13.35) in [131]:

$$f\left(\left\{\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right\} \mid \left\{\mathbf{x}^k, \mathbf{y}^k\right\}\right) = f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{x}^k, \mathbf{y}^k\right]\right) + f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{y}^k, \mathbf{x}^k\right]\right) \quad (\text{B.12})$$

Therefore

$$\begin{aligned} \varpi^{k+1}\left(\left\{\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right\}\right) &= \frac{1}{2} \int \left[f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{x}^k, \mathbf{y}^k\right]\right) + f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{y}^k, \mathbf{x}^k\right]\right) \right] \\ &\quad \left[\pi^k\left(\left[\mathbf{x}^k, \mathbf{y}^k\right]\right) + \pi^k\left(\left[\mathbf{y}^k, \mathbf{x}^k\right]\right) \right] d\mathbf{x}^k d\mathbf{y}^k \\ &= \frac{1}{2} \left[\int f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{x}^k, \mathbf{y}^k\right]\right) \pi^k\left(\left[\mathbf{x}^k, \mathbf{y}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \right. \\ &\quad + \int f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{x}^k, \mathbf{y}^k\right]\right) \pi^k\left(\left[\mathbf{y}^k, \mathbf{x}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \\ &\quad + \int f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{y}^k, \mathbf{x}^k\right]\right) \pi^k\left(\left[\mathbf{x}^k, \mathbf{y}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \\ &\quad \left. + \int f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{y}^k, \mathbf{x}^k\right]\right) \pi^k\left(\left[\mathbf{y}^k, \mathbf{x}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \right] \quad (\text{B.13}) \end{aligned}$$

Assuming that both targets move independently with the same dynamics

$$f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{x}^k, \mathbf{y}^k\right]\right) = f_d\left(\mathbf{x}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{y}^k\right) \quad (\text{B.14})$$

Substituting (B.14) into (B.13):

$$\begin{aligned} \varpi^{k+1}\left(\left\{\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right\}\right) &= \frac{1}{2} \left[\int f_d\left(\mathbf{x}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{y}^k\right) \pi^k\left(\left[\mathbf{x}^k, \mathbf{y}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \right. \\ &\quad + \int f_d\left(\mathbf{x}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{y}^k\right) \pi^k\left(\left[\mathbf{y}^k, \mathbf{x}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \\ &\quad + \int f_d\left(\mathbf{x}^{k+1} \mid \mathbf{y}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{x}^k\right) \pi^k\left(\left[\mathbf{x}^k, \mathbf{y}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \\ &\quad \left. + \int f_d\left(\mathbf{x}^{k+1} \mid \mathbf{y}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{x}^k\right) \pi^k\left(\left[\mathbf{y}^k, \mathbf{x}^k\right]\right) d\mathbf{x}^k d\mathbf{y}^k \right] \\ &= \frac{1}{2} \left[2\varpi^{k+1}\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right]\right) + 2\varpi^{k+1}\left(\left[\mathbf{y}^{k+1}, \mathbf{x}^{k+1}\right]\right) \right] \\ &= \varpi^{k+1}\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right]\right) + \varpi^{k+1}\left(\left[\mathbf{y}^{k+1}, \mathbf{x}^{k+1}\right]\right) \quad (\text{B.15}) \end{aligned}$$

Therefore, the prediction phase using sets is equivalent to the prediction phase using vectors and then obtaining its corresponding set prediction density.

B.1.5 Prediction phase: set notation with labels

The transition density in this case is

$$\begin{aligned}
& f\left(\left\{\left[\mathbf{x}^{k+1}, i_x^{k+1}\right], \left[\mathbf{y}^{k+1}, i_y^{k+1}\right]\right\} \mid \left\{\left[\mathbf{x}^k, i_x^k\right], \left[\mathbf{y}^k, i_y^k\right]\right\}\right) \\
&= f\left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1}\right] \mid \left[\mathbf{x}^k, \mathbf{y}^k\right]\right) \delta\left(i_x^{k+1}-i_x^k\right) \delta\left(i_y^{k+1}-i_y^k\right) \\
&\quad + f\left(\left[\mathbf{y}^{k+1}, \mathbf{x}^{k+1}\right] \mid \left[\mathbf{x}^k, \mathbf{y}^k\right]\right) \delta\left(i_y^{k+1}-i_x^k\right) \delta\left(i_x^{k+1}-i_y^k\right) \\
&= f_d\left(\mathbf{x}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{y}^k\right) \delta\left(i_x^{k+1}-i_x^k\right) \delta\left(i_y^{k+1}-i_y^k\right) \\
&\quad + f_d\left(\mathbf{y}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{x}^{k+1} \mid \mathbf{y}^k\right) \delta\left(i_y^{k+1}-i_x^k\right) \delta\left(i_x^{k+1}-i_y^k\right)
\end{aligned} \tag{B.16}$$

The prior density at time $k+1$ results from the set integral

$$\begin{aligned}
& \varpi^{k+1}\left(\left\{\left[\mathbf{x}^{k+1}, i_x^{k+1}\right], \left[\mathbf{y}^{k+1}, i_y^{k+1}\right]\right\}\right) \\
&= \int f\left(\left\{\left[\mathbf{x}^{k+1}, i_x^{k+1}\right], \left[\mathbf{y}^{k+1}, i_y^{k+1}\right]\right\} \mid \left\{\left[\mathbf{x}^k, i_x^k\right], \left[\mathbf{y}^k, i_y^k\right]\right\}\right) \\
&\quad \cdot \pi^k\left(\left\{\left[\mathbf{x}^k, i_x^k\right], \left[\mathbf{y}^k, i_y^k\right]\right\}\right) \partial\left\{\left[\mathbf{x}^k, i_x^k\right], \left[\mathbf{y}^k, i_y^k\right]\right\}
\end{aligned} \tag{B.17}$$

Substituting (B.4) and (B.16) into (B.17), the prior is

$$\begin{aligned}
& \varpi^{k+1}\left(\left\{\left[\mathbf{x}^{k+1}, i_x^{k+1}\right], \left[\mathbf{y}^{k+1}, i_y^{k+1}\right]\right\}\right) \\
&= \frac{1}{2} \cdot \left[\int f_d\left(\mathbf{x}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{y}^k\right) \delta\left(i_x^{k+1}-i_x^k\right) \delta\left(i_y^{k+1}-i_y^k\right) \right. \\
&\quad \cdot \pi^k\left(\left[\mathbf{x}^k, \mathbf{y}^k\right]\right) \delta\left(i_x^k-1\right) \delta\left(i_y^k-2\right) d\mathbf{x}^k d\mathbf{y}^k d i_x^k d i_y^k \\
&\quad + \int f_d\left(\mathbf{x}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{y}^{k+1} \mid \mathbf{y}^k\right) \delta\left(i_x^{k+1}-i_x^k\right) \delta\left(i_y^{k+1}-i_y^k\right) \\
&\quad \cdot \pi^k\left(\left[\mathbf{y}^k, \mathbf{x}^k\right]\right) \delta\left(i_y^k-1\right) \delta\left(i_x^k-2\right) d\mathbf{x}^k d\mathbf{y}^k d i_x^k d i_y^k \\
&\quad + \int f_d\left(\mathbf{y}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{x}^{k+1} \mid \mathbf{y}^k\right) \delta\left(i_y^{k+1}-i_x^k\right) \delta\left(i_x^{k+1}-i_y^k\right) \\
&\quad \cdot \pi^k\left(\left[\mathbf{x}^k, \mathbf{y}^k\right]\right) \delta\left(i_x^k-1\right) \delta\left(i_y^k-2\right) d\mathbf{x}^k d\mathbf{y}^k d i_x^k d i_y^k \\
&\quad \left. + \int f_d\left(\mathbf{y}^{k+1} \mid \mathbf{x}^k\right) f_d\left(\mathbf{x}^{k+1} \mid \mathbf{y}^k\right) \delta\left(i_y^{k+1}-i_x^k\right) \delta\left(i_x^{k+1}-i_y^k\right) \right. \\
&\quad \cdot \pi^k\left(\left[\mathbf{y}^k, \mathbf{x}^k\right]\right) \delta\left(i_y^k-1\right) \delta\left(i_x^k-2\right) d\mathbf{x}^k d\mathbf{y}^k d i_x^k d i_y^k \Big]
\end{aligned} \tag{B.18}$$

The results of these integrals are

$$\begin{aligned}
 & \varpi^{k+1} \left(\left\{ \left[\mathbf{x}^{k+1}, i_x^{k+1} \right], \left[\mathbf{y}^{k+1}, i_y^{k+1} \right] \right\} \right) = \\
 & \frac{1}{2} \cdot \left[\varpi^{k+1} \left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1} \right] \right) \delta \left(i_x^{k+1} - 1 \right) \delta \left(i_y^{k+1} - 2 \right) \right. \\
 & + \varpi^{k+1} \left(\left[\mathbf{y}^{k+1}, \mathbf{x}^{k+1} \right] \right) \delta \left(i_x^{k+1} - 2 \right) \delta \left(i_y^{k+1} - 1 \right) \\
 & + \varpi^{k+1} \left(\left[\mathbf{y}^{k+1}, \mathbf{x}^{k+1} \right] \right) \delta \left(i_x^{k+1} - 2 \right) \delta \left(i_y^{k+1} - 1 \right) \\
 & \left. + \varpi^{k+1} \left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1} \right] \right) \delta \left(i_x^{k+1} - 1 \right) \delta \left(i_y^{k+1} - 2 \right) \right] \quad (\text{B.19})
 \end{aligned}$$

The prior at time $k + 1$ using set notation with labels finally becomes

$$\begin{aligned}
 & \varpi^{k+1} \left(\left\{ \left[\mathbf{x}^{k+1}, i_x^{k+1} \right], \left[\mathbf{y}^{k+1}, i_y^{k+1} \right] \right\} \right) = \\
 & \delta \left(i_x^{k+1} - 1 \right) \delta \left(i_y^{k+1} - 2 \right) \varpi^{k+1} \left(\left[\mathbf{x}^{k+1}, \mathbf{y}^{k+1} \right] \right) \\
 & + \delta \left(i_x^{k+1} - 2 \right) \delta \left(i_y^{k+1} - 1 \right) \varpi^{k+1} \left(\left[\mathbf{y}^{k+1}, \mathbf{x}^{k+1} \right] \right) \quad (\text{B.20})
 \end{aligned}$$

The prior at time $k + 1$ using labelled sets, equation (B.20), contains the same information as the prior at time $k + 1$ using vectors, equation (B.5). That is, given the prior of the labelled set one can get the prior of the vector and the other way round. In fact, the way to compute (B.20) is calculating (B.5) adding the Dirac deltas of the labels and adding a term in which the target states are permuted. The only difference is that when set notation is used, complex set integrals must be used.

B.1.6 PHD filter

The probability hypothesis density (PHD) filter is a popular method to carry out multiple target tracking when the multitarget state is represented by a set. The integral of the PHD in any region of the state space is the expected number of targets in that region [60, 130]. In this section, the PHD is written explicitly for the unlabelled and labelled set cases.

B.1.6.1 PHD for sets without labels

The PHD of the posterior PDF for sets without labels is defined as, see (65) in [130]

$$D_s^k(\mathbf{x}) = \int \pi^k(\{\mathbf{x}, \mathbf{w}\}) d\mathbf{w}$$

$$\begin{aligned}
 &= \int \left[\pi^k([\mathbf{x}, \mathbf{w}]) + \pi^k([\mathbf{w}, \mathbf{x}]) \right] d\mathbf{w} \\
 &= \pi_1^k(\mathbf{x}) + \pi_2^k(\mathbf{x})
 \end{aligned} \tag{B.21}$$

where

$$\pi_1^k(\mathbf{x}) = \int \pi^k([\mathbf{x}, \mathbf{w}]) d\mathbf{w} \tag{B.22}$$

$$\pi_2^k(\mathbf{x}) = \int \pi^k([\mathbf{w}, \mathbf{x}]) d\mathbf{w} \tag{B.23}$$

That is, $\pi_1^k(\cdot)$ and $\pi_2^k(\cdot)$ are the marginal posterior PDFs for target “1” and “2” respectively.

B.1.6.2 PHD for labelled sets

The PHD of the posterior PDF for labelled sets is

$$\begin{aligned}
 D_{sl}^k([\mathbf{x}, i_x]) &= \int \left[\pi^k([\mathbf{x}, i_x, \mathbf{w}, i_w]) + \pi^k([\mathbf{w}, i_w, \mathbf{x}, i_x]) \right] d\mathbf{w} di_w \\
 &= \int \pi^k([\mathbf{x}, \mathbf{w}]) \delta(i_x - 1) \delta(i_w - 2) d\mathbf{w} di_w \\
 &\quad + \int \pi^k([\mathbf{w}, \mathbf{x}]) \delta(i_w - 1) \delta(i_x - 2) d\mathbf{w} di_w \\
 &= \pi_1^k(\mathbf{x}) \delta(i_x - 1) + \pi_2^k(\mathbf{x}) \delta(i_x - 2)
 \end{aligned} \tag{B.24}$$

Therefore, the PHD for labelled sets corresponds with the sum of the marginal posterior PDFs of the targets each one multiplied by a Dirac delta function with the labels. This information is also available when vectors are used and gives more insight into the issue of what the PHD for a labelled set is. The PHD has the same information as an approximation to the posterior using vector notation when one only approximates the marginal PDFs of the targets. That is, the possible dependence among target states is not taken into account.

B.2 Variable and unknown number of targets

In this appendix, several notations to represent the multitarget state are analysed: a labelled vector (JMKPD), an unlabelled vector (JMPD), a labelled set, an unlabelled set and a vector with existence variables. First, some comments on these notations are given. Second, a multitarget situation is represented for the different notations.

The JMKPD introduced in Chapter 4 represents the multitarget state as $\mathbf{R}^k =$

$$\left[\left(\mathbf{r}_1^k \right)^T, \dots, \left(\mathbf{r}_{t^k}^k \right)^T \right]^T = \left[\left(\mathbf{x}_1^k \right)^T, i_1^k, \dots, \left(\mathbf{x}_{t^k}^k \right)^T, i_{t^k}^k \right]^T \text{ where}$$

$$\left(\mathbf{R}^k, t^k \right) \in \bigcup_{t=0}^{\infty} \left(\mathbb{R}^4 \times \mathbb{N} \right)^t \times \{t\} : i_1^k < i_2^k < \dots < i_t^k$$

It should be noted that the corresponding labelled set multitarget state representation is $X^k = \left\{ \left[\left(\mathbf{x}_1^k \right)^T, i_1^k \right], \dots, \left[\left(\mathbf{x}_{t^k}^k \right)^T, i_{t^k}^k \right] \right\}$. It is clear that there is a bijection between $\mathbf{R}^k$ and X^k and therefore, both notations lead to the same results. Moreover, any prior model available with set notation with labels can be equivalently represented with vector notation and the other way round. The main difference from a practical point of view is that one model uses set integrals and the other vector integrals.

Another common representation for tracking a variable and unknown number of targets is using $\mathbf{X}^k = \left[\left(\mathbf{x}_1^k \right)^T, e_1^k, \dots, \left(\mathbf{x}_{t^{max}}^k \right)^T, e_{t^{max}}^k \right]^T$ where t^{max} is the maximum number of targets that can appear and, existence variables $e_i^k \in \{0, 1\}$ have been introduced to indicate if the target is death or alive [177, 198]. However, this notation introduces a philosophical problem. Let us assume we are guarding a surveillance area and targets start to appear until there are t^{max} . At each time, it is known where each particular target is at a time. However, let us assume all targets disappear and after a very long time a target shows up. Then, we would need to allocate one of the “subvectors” in vector $\mathbf{X}^k$ to represent the target. The problem is that would be indicating that exactly the same target that disappeared long time ago has shown up again. It might be possible that this representation could be extended for $t^{max} \rightarrow \infty$ and this problem would disappear but, to the author’s knowledge, this approach has never been justified.

In the following, a multitarget situation is represented using different notations for the multitarget state.

B.2.1 Situation at time k

Assume the situation at time k is the following:

- There are two possible targets: target with “label 1” exists with probability p_1 and target with “label 2” exists with probability p_2 . Their possible locations are far from each other such that their probabilities of existence are independent.
- Target with “label 1” is located around $\bar{\mathbf{x}}_1^k$ such that its marginal PDF of the state conditional on its existence is $\pi_1^k(\cdot)$ (which is concentrated around $\bar{\mathbf{x}}_1^k$).

- Target with “label 2” is located around $\bar{\mathbf{x}}_2^k$ such that its marginal PDF of the state conditional on its existence is $\pi_2^k(\cdot)$ (which is concentrated around $\bar{\mathbf{x}}_2^k$).
- If the targets exist, they have always been far from each other such that their states given the past measurements are independent.

This scenario is represented using the techniques mentioned before.

B.2.2 Posterior PDF at time k using a vector with existence variables

It is assumed that $t^{max} = 2$ for the sake of simplicity. Therefore, the posterior PDF at time k is

$$\pi^k \left(\left[\mathbf{x}_1^k, e_1^k, \mathbf{x}_2^k, e_2^k \right] \right) = \left[(1 - p_1) \delta(e_1^k - 0) + \pi_1^k(\mathbf{x}_1^k) p_1 \delta(e_1^k - 1) \right] \cdot \left[(1 - p_2) \delta(e_2^k - 0) + \pi_2^k(\mathbf{x}_2^k) p_2 \delta(e_2^k - 1) \right] \quad (\text{B.25})$$

$$\begin{aligned} &= (1 - p_1)(1 - p_2) \delta(e_1^k - 0) \delta(e_2^k - 0) \\ &\quad + (1 - p_1) p_2 \pi_2^k(\mathbf{x}_2^k) \delta(e_1^k - 0) \delta(e_2^k - 1) \\ &\quad + p_1 (1 - p_2) \pi_1^k(\mathbf{x}_1^k) \delta(e_1^k - 1) \delta(e_2^k - 0) \\ &\quad + p_1 p_2 \pi_1^k(\mathbf{x}_1^k) \pi_2^k(\mathbf{x}_2^k) \delta(e_1^k - 1) \delta(e_2^k - 1) \end{aligned} \quad (\text{B.26})$$

B.2.3 Posterior PDF at time k using a labelled vector

The state is of the form $(\mathbf{R}^k, t^k)$ where t^k governs the dimensionality of $\mathbf{R}^k$. Therefore, conditioned on $t^k > 0$, the state is of the form $t^k = 1 \rightarrow \mathbf{R}^k = \left[\left(\mathbf{x}_1^k \right)^T, i_1^k \right]^T$ and $t^k = 2 \rightarrow \mathbf{R}^k = \left[\left(\mathbf{x}_1^k \right)^T, i_1^k, \left(\mathbf{x}_2^k \right)^T, i_2^k \right]^T$. For $t^k = 0$, we use the symbol for no target. It should be noted that inference using this kind of state representation is usually known as Bayesian hierarchical model [4, 76, 79, 80, 96, 146]. The posterior PDF at time k is

$$\begin{aligned} \pi^k(\mathbf{R}^k, t^k) &= \pi^k(\mathbf{R}^k | t^k) \pi^k(t^k) \\ &= P_0 \delta(t^k - 0) \\ &\quad + P_1 \delta(t^k - 1) \left[\frac{p_1(1 - p_2)}{P_1} \delta(i_1^k - 1) \pi_1^k(\mathbf{x}_1^k) \right. \\ &\quad \left. + \frac{p_2(1 - p_1)}{P_1} \delta(i_1^k - 2) \pi_2^k(\mathbf{x}_1^k) \right] \\ &\quad + P_2 \delta(t^k - 2) \delta(i_1^k - 1) \delta(i_2^k - 2) \pi_1^k(\mathbf{x}_1^k) \pi_2^k(\mathbf{x}_2^k) \end{aligned} \quad (\text{B.27})$$

where $P_i \quad i \in \{0, 1, 2\}$ is the probability that i targets exist:

$$P_0 = (1 - p_1)(1 - p_2) \quad (\text{B.28})$$

$$P_1 = p_1(1 - p_2) + (1 - p_1)p_2 \quad (\text{B.29})$$

$$P_2 = p_1 p_2 \quad (\text{B.30})$$

Note that (B.26) and (B.27) are equivalent in the sense that convey the same information. This representation is more complicated than the previous one but has the advantage that it does not require the definition of a maximum number of targets.

B.2.4 Posterior PDF at time k using a vector without labels

The state is of the form $(\mathbf{X}^k, t^k)$ where t^k governs the dimensionality of $\mathbf{X}^k$. Therefore, conditioned on $t^k > 0$, the state is of the form $t^k = 1 \rightarrow \mathbf{X}^k = \mathbf{x}_1^k$ and $t^k = 2 \rightarrow \mathbf{X}^k = \left[(\mathbf{x}_1^k)^T, (\mathbf{x}_2^k)^T \right]^T$. Note that the state is equivalent to the one using vector with labels but removing the labels. Therefore, integrating out the labels in (B.27), one gets the posterior PDF for this case.

$$\begin{aligned} \pi^k(\mathbf{X}^k, t^k) &= \pi^k(\mathbf{X}^k | t^k) \pi^k(t^k) \\ &= P_0 \delta(t^k - 0) \\ &\quad + P_1 \delta(t^k - 1) \left[\frac{p_1(1 - p_2)}{P_1} \pi_1^k(\mathbf{x}_1^k) + \frac{p_2(1 - p_1)}{P_1} \pi_2^k(\mathbf{x}_1^k) \right] \\ &\quad + P_2 \delta(t^k - 2) \pi_1^k(\mathbf{x}_1^k) \pi_2^k(\mathbf{x}_2^k) \end{aligned} \quad (\text{B.31})$$

There are some aspects in (B.31) that are worth highlighting. When there are zero targets, the distribution makes sense. When there are two targets, it is known which target is which and the distribution makes sense¹. However, when there is only one target, it is not known whether it is target one or target two. Conditioned on two targets it is known which target is which but conditioned on one target, it is not known which target is alive. This information was available in the previous representations that could give an accurate picture of the scenario. However, this representation does not because it loses information about labelling and, importantly, it is lost only for one target. Therefore, using vectors without labels for variable and unknown number of targets is not sound.

A way to solve this problem is to say that ordering does not matter. That is,

¹Note that conditioned on two targets, one can infer where target “1” is or where target “2” is so the labelling information is available.

when there are two targets

$$\pi^k \left(\left[\left(\mathbf{x}_1^k \right)^T, \left(\mathbf{x}_2^k \right)^T \right]^T \middle| t^k = 2 \right) = \pi^k \left(\left[\left(\mathbf{x}_2^k \right)^T, \left(\mathbf{x}_1^k \right)^T \right]^T \middle| t^k = 2 \right).$$

This solution is usually known as the JMPD [116]. The resulting PDF $\pi_K^k(\cdot)$ is

$$\begin{aligned} \pi_K^k(\mathbf{X}^k, t^k) &= \pi_K^k(\mathbf{X}^k | t^k) \pi^k(t^k) \\ &= P_0 \delta(t^k - 0) \\ &\quad + P_1 \delta(t^k - 1) \left[\frac{p_1(1-p_2)}{P_1} \pi_1^k(\mathbf{x}_1^k) + \frac{p_2(1-p_1)}{P_1} \pi_2^k(\mathbf{x}_1^k) \right] \\ &\quad + \frac{P_2}{2} \delta(t^k - 2) \left[\pi_1^k(\mathbf{x}_1^k) \pi_2^k(\mathbf{x}_2^k) + \pi_1^k(\mathbf{x}_2^k) \pi_2^k(\mathbf{x}_1^k) \right] \end{aligned} \quad (\text{B.32})$$

This way, the labelling information is not available for one and two targets. However, this representation is equivalent to the random finite set formulation (without labels) at the expense of losing labelling information [130] as will be shown in Section B.2.6.

B.2.5 Posterior PDF at time k using a labelled set

The possible values of the set are $X^k = \{\emptyset\}$, $X^k = \left\{ \left[\left(\mathbf{x}_1^k \right)^T, i_1^k \right]^T \right\}$ and $X^k = \left\{ \left[\left(\mathbf{x}_1^k \right)^T, i_1^k \right]^T, \left[\left(\mathbf{x}_2^k \right)^T, i_2^k \right]^T \right\}$. The posterior PDF is

$$\begin{aligned} \pi^k(X^k) &= P_0 \delta(X^k - \emptyset) \\ &\quad + P_1 \delta \left(X^k - \left\{ \left[\left(\mathbf{x}_1^k \right)^T, 1 \right]^T \right\} \right) \frac{p_1(1-p_2)}{P_1} \pi_1^k(\mathbf{x}_1^k) \\ &\quad + P_1 \delta \left(X^k - \left\{ \left[\left(\mathbf{x}_1^k \right)^T, 2 \right]^T \right\} \right) \frac{p_2(1-p_1)}{P_1} \pi_2^k(\mathbf{x}_1^k) \\ &\quad + P_2 \delta \left(X^k - \left\{ \left[\left(\mathbf{x}_1^k \right)^T, 1 \right]^T, \left[\left(\mathbf{x}_2^k \right)^T, 2 \right]^T \right\} \right) \\ &\quad \cdot \left[\pi_1^k(\mathbf{x}_1^k) \pi_2^k(\mathbf{x}_2^k) \delta(i_1^k - 1) \delta(i_2^k - 2) \right. \\ &\quad \left. + \pi_1^k(\mathbf{x}_2^k) \pi_2^k(\mathbf{x}_1^k) \delta(i_1^k - 2) \delta(i_2^k - 1) \right] \end{aligned} \quad (\text{B.33})$$

Therefore, the information contained in the posterior PDF using the labelled vector approach, given by (B.27), and the labelled set approach, given by (B.33), are the same. The result is that complex set integrals are not necessary to calculate

the prediction equation as they are equivalent to vector ones if labels are used as explained for known and fixed number of targets in Section B.1.

B.2.6 Posterior PDF at time k using a set without labels

The possible values of the set are $X^k = \{\emptyset\}$, $X^k = \{\mathbf{x}_1^k\}$ and $X^k = \{\mathbf{x}_1^k, \mathbf{x}_2^k\}$. The posterior PDF is

$$\begin{aligned} \pi^k(X^k) = & P_0 \delta(X^k - \emptyset) \\ & + P_1 \delta(X^k - \{\mathbf{x}_1^k\}) \left[\frac{p_1(1-p_2)}{P_1} \pi_1^k(\mathbf{x}_1^k) + \frac{p_2(1-p_1)}{P_1} \pi_2^k(\mathbf{x}_1^k) \right] \\ & + P_2 \delta(X^k - \{\mathbf{x}_1^k, \mathbf{x}_2^k\}) \left[\pi_1^k(\mathbf{x}_1^k) \pi_2^k(\mathbf{x}_2^k) + \pi_1^k(\mathbf{x}_2^k) \pi_2^k(\mathbf{x}_1^k) \right] \end{aligned} \quad (\text{B.34})$$

B.2.7 Comments on the mathematical validity of a labelled vector

It is known that if a vector (without labels) is used to represent the state, there are mathematical inconsistencies because one cannot define a proper metric and therefore estimation error becomes a vague concept [201]. However, as there is a bijection between the labelled vector approach and the labelled set approach, that is, one can go from one representation to another and the other way round, a metric defined in the space of labelled sets also works if labelled vectors are used so in this sense they are equivalent. In fact, any PDF (such as the ones that model target birth and death) written in a labelled set framework can be written equivalently in a labelled vector framework because of this one-to-one correspondence.

The only differences between the two representations is the use of set integrals in one case or vector integrals in another and a more compact representation of the PDF in the vector case. As vector integrals are much easier to compute, it is concluded that this representation should be preferable to a labelled set representation in spite of the fact that they lead exactly to the same results.

B.3 Conclusions

When tracking a fixed and known number of targets, (unlabelled) set notation implies that it cannot be known which target is which along time, all that matters is the target states. In fact, target tracking from a purely Bayesian point of view cannot be performed using (unlabelled) set notation as the correspondence between target states along time cannot be established as they are not labelled. Nevertheless, if one is just interested in estimating target states without tracking, approximating

the corresponding set posterior PDF can be better than approximating the vector posterior PDF. This is evident in the set JPDAF compared to the JPDAF [185]. If labelled sets are used, the posterior PDF has the same information as a vector's one. However, labelled set notation uses a much more cumbersome representation of the posterior PDF and uses complex set integrals.

When tracking variable and unknown number of targets, labels must be included to perform tracking from a Bayesian point of view. As there is a bijection between labelled set notation and labelled vector notation, both methods are equivalent with the difference that random set notation implies the use of complex set integrals. However, any system model or filtering algorithm developed using labelled vector notation can be equivalently written using labelled set notation and the other way round. Consequently, the algorithms in Chapter 4 could also have been developed analogously using random set notation. In fact, even the MSLAM algorithm in Chapter 6 could have also be written using random set notation by labelling the sensors and the targets.

Bibliography

- [1] N. Ahmed, M. Rutten, T. Bessell, S. Kanhere, N. Gordon, and S. Jha, “Detection and tracking using particle-filter-based wireless sensor networks,” *IEEE Transactions on Mobile Computing*, vol. 9, no. 9, pp. 1332–1345, Sept. 2010.
- [2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks,” *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, Aug 2002.
- [3] S. S. Ali-Löytty, “Box Gaussian mixture filter,” *IEEE Transactions on Automatic Control*, vol. 55, no. 9, pp. 2165–2169, Sept. 2010.
- [4] C. Andrieu and A. Doucet, “Joint Bayesian model selection and estimation of noisy sinusoids via reversible jump MCMC,” *IEEE Transactions on Signal Processing*, vol. 47, no. 10, pp. 2667–2676, Oct. 1999.
- [5] C. Andrieu, A. Doucet, S. Singh, and V. Tadic, “Particle methods for change detection, system identification, and control,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 423–438, Mar. 2004.
- [6] C. Andrieu and A. Doucet, “Particle filtering for partially observed Gaussian state space models,” *Journal of the Royal Statistical Society: Series B*, vol. 64, pp. 827–836, 2002.
- [7] I. Arasaratnam and S. Haykin, “Cubature Kalman filters,” *IEEE Transactions on Automatic Control*, vol. 54, no. 6, pp. 1254–1269, June 2009.
- [8] I. Arasaratnam, S. Haykin, and T. Hurd, “Cubature Kalman filtering for continuous-discrete systems: Theory and simulations,” *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 4977–4993, Oct. 2010.
- [9] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002.

- [10] T. Bailey and H. Durrant-Whyte, “Simultaneous localization and mapping (SLAM): part II,” *IEEE Robotics Automation Magazine*, vol. 13, no. 3, pp. 108–117, Sept. 2006.
- [11] T. Bailey, J. Nieto, and E. Nebot, “Consistency of the FastSLAM algorithm,” in *Proceedings 2006 IEEE International Conference on Robotics and Automation, ICRA*, May 2006, pp. 424–429.
- [12] Y. Bar-Shalom, “Update with out-of-sequence measurements in tracking: exact solution,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 3, pp. 769–777, Jul. 2002.
- [13] Y. Bar-Shalom, T. Kirubarajan, and X. R. Li, *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., 2002.
- [14] C. Baum and V. Veeravalli, “A sequential procedure for multihypothesis testing,” *IEEE Transactions on Information Theory*, vol. 40, no. 6, pp. 1994–2007, Nov. 1994.
- [15] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Artech House, 1999.
- [16] W. D. Blair and M. Brandt-Pearce, “Unresolved Rayleigh target detection using monopulse measurements,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 2, pp. 543–552, Apr. 1998.
- [17] H. Blom and E. Bloem, “Decomposed particle filtering and track swap estimation in tracking two closely spaced targets,” in *14th International Conference on Information Fusion*, 2011, pp. 787–794.
- [18] —, “Permutation invariance in Bayesian estimation of two targets that maneuver in and out formation flight,” in *12th International Conference on Information Fusion*, July 2009, pp. 1296–1303.
- [19] H. Blom, E. Bloem, Y. Boers, and H. Driessen, “Tracking closely spaced targets: Bayes outperformed by an approximation?” pp. 1–8, July 2008.
- [20] Y. Boers and H. Driessen, “A particle-filter-based detection scheme,” *IEEE Signal Processing Letters*, vol. 10, no. 10, pp. 300–302, Oct. 2003.
- [21] —, “Particle filter track-before-detect application using inequality constraints,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1483–1489, Oct. 2005.

- [22] —, “The mixed labeling problem in multi target particle filtering,” in *10th International Conference on Information Fusion*, July 2007, pp. 1–7.
- [23] Y. Boers and J. N. Driessen, “Multitarget particle filter track before detect application,” *IEE Proceedings Radar, Sonar and Navigation*, vol. 151, no. 6, pp. 351–357, Dec. 2004.
- [24] Y. Boers, E. Sviestins, and H. Driessen, “Mixed labelling in multitarget particle filtering,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 2, pp. 792–802, April 2010.
- [25] M. Bolic, P. Djuric, and S. Hong, “Resampling algorithms and architectures for distributed particle filters,” *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2442–2450, July 2005.
- [26] E. Bølviken and G. Storvik, “Deterministic and stochastic particle filters in state-space models,” in *Sequential Monte Carlo Methods in Practice*. Springer, 2001.
- [27] X. Boyen and D. Koller, “Tractable inference for complex stochastic processes,” in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 33–42.
- [28] R. S. Bucy and K. D. Senne, “Digital synthesis of non-linear filters,” *Automatica*, vol. 7, no. 3, pp. 287–298, May 1971.
- [29] K. Burdett, “Truncated means and variances,” *Economics Letters*, vol. 52, pp. 263–267, 1996.
- [30] J. Candy, “Bootstrap particle filtering,” *IEEE Signal Processing Magazine*, vol. 24, no. 4, pp. 73–85, July 2007.
- [31] O. Cappé, S. Godsill, and E. Moulines, “An overview of existing methods and recent advances in sequential Monte Carlo,” *Proceedings of the IEEE*, vol. 95, no. 5, pp. 899–924, May 2007.
- [32] O. Cappé, E. Moulines, and T. Rydén, *Inference in Hidden Markov models*. Springer Science + Business Media, Inc., 2005.
- [33] J. A. Castellanos, J. D. Tardos, and G. Schmidt, “Building a global map of the environment of a mobile robot: the importance of correlations,” in *Proceedings IEEE International Conference on Robotics and Automation*, vol. 2, Apr. 1997, pp. 1053–1059.

- [34] V. Cevher and J. H. McClellan, “General direction-of-arrival tracking with acoustic nodes,” *IEEE Transactions on Signal Processing*, vol. 53, no. 1, pp. 1–12, Jan. 2005.
- [35] S. Chakraborty, “Some applications of Dirac’s delta function in statistics for more than one random variable,” *Applications and applied mathematics*, vol. 3, no. 1, pp. 42–54, June 2008.
- [36] J.-F. Chamberland and V. V. Veeravalli, “Decentralized detection in sensor networks,” *IEEE Transactions on Signal Processing*, vol. 51, no. 2, pp. 407–416, Feb. 2003.
- [37] T. Chen, T. Schön, H. Ohlsson, and L. Ljung, “Decentralized particle filter with arbitrary state decomposition,” *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 465–478, Feb. 2011.
- [38] D. Crisan and A. Doucet, “A survey of convergence results on particle filtering methods for practitioners,” *IEEE Transactions on Signal Processing*, vol. 50, no. 3, pp. 736–746, Mar. 2002.
- [39] D. F. Crouse, P. Willett, and Y. Bar-Shalom, “Developing a real-time track display that operators do not hate,” *IEEE Transactions on Signal Processing*, vol. 59, no. 7, pp. 3441–3447, July 2011.
- [40] D. F. Crouse, P. Willett, L. Svensson, D. Svensson, and M. Guerriero, “The set MHT,” in *14th International Conference on Information Fusion*, 2011, pp. 2134–2141.
- [41] D. F. Crouse, P. Willett, and Y. Bar-Shalom, “Generalizations of Blom and Bloem’s PDF decomposition for permutation-invariant estimation,” in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2011, pp. 3840–3843.
- [42] D. Culler, D. Estrin, and M. Srivastava, “Guest editors’ introduction: Overview of sensor networks,” *Computer*, vol. 37, no. 8, pp. 41–49, Aug. 2004.
- [43] W. Dargie and C. Poellabauer, *Fundamentals of Wireless Sensor Networks*. John Wiley & Sons, 2010.
- [44] F. Daum, “Nonlinear filters: beyond the Kalman filter,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 20, no. 8, pp. 57–69, Aug. 2005.
- [45] F. Daum and J. Huang, “Curse of dimensionality and particle filters,” in *Proceedings IEEE Aerospace Conference*, vol. 4, 8-15 2003, pp. 1979–1993.

- [46] —, “Particle flow for nonlinear filters with log-homotopy,” in *Proceedings of the SPIE*, vol. 6969, 2008.
- [47] —, “Particle flow for nonlinear filters,” in *Proceedings of the IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP)*, 2011, pp. 5920–5923.
- [48] S. J. Davey, M. G. Rutten, and B. Cheung, “A comparison of detection performance for several track-before-detect algorithms,” in *EURASIP Journal on Advances in Signal Processing*, vol. 2008, 2008, pp. 1–10.
- [49] F. Dellaert and M. Kaess, “Square root SAM: Simultaneous localization and mapping via square root information smoothing,” *The International Journal of Robotics Research*, vol. 25, pp. 1181–1203, December 2006.
- [50] M. W. M. G. Dissanayake, P. Newman, S. Clark, H. F. Durrant-Whyte, and M. Csorba, “A solution to the simultaneous localization and map building (SLAM) problem,” *IEEE Transactions on Robotics and Automation*, vol. 17, no. 3, pp. 229–241, Jun. 2001.
- [51] P. Djuric, “Monitoring and selection of dynamic models by Monte Carlo sampling,” in *Proceedings of the IEEE Signal Processing Workshop on Higher-Order Statistics*, 1999, pp. 191–194.
- [52] P. Djuric, J. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. Bugallo, and J. Míguez, “Particle filtering,” *IEEE Signal Processing Magazine*, vol. 20, no. 5, pp. 19–38, Sep 2003.
- [53] R. Douc, G. Fort, E. Moulines, and P. Priouret, “Forgetting the initial distribution for hidden Markov models,” *Stochastic Processes and their Applications*, vol. 119, no. 4, pp. 1235–1256, 2009.
- [54] R. Douc, E. Gassiat, B. Landelle, and E. Moulines, “Forgetting of the initial distribution for non-ergodic hidden Markov chains,” submitted to the *Annals of Applied Probability*.
- [55] 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.
- [56] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. Springer, 2001.

- [57] H. Driessen and Y. Boers, “MAP estimation in particle filter tracking,” in *IET Seminar on Target Tracking and Data Fusion: Algorithms and Applications*, April 2008, pp. 41–45.
- [58] M. Dueker, “Kalman filtering with truncated normal state variables for Bayesian estimation of macroeconomic models,” *Economics Letters*, vol. 93, no. 1, pp. 141–149, Oct. 2006.
- [59] H. Durrant-Whyte and T. Bailey, “Simultaneous localization and mapping: part I,” *IEEE Robotics Automation Magazine*, vol. 13, no. 2, pp. 99–110, June 2006.
- [60] O. Erdinc, P. Willett, and Y. Bar-Shalom, “The bin-occupancy filter and its connection to the PHD filters,” *IEEE Transactions on Signal Processing*, vol. 57, no. 11, pp. 4232–4246, Nov. 2009.
- [61] M. Fallon and S. Godsill, “Multi target acoustic source tracking with an unknown and time varying number of targets,” in *Hands-Free Speech Communication and Microphone Arrays, HSCMA 2008*, May 2008, pp. 77–80.
- [62] W. J. Fitzgerald, “Markov chain Monte Carlo methods with applications to signal processing,” *Signal Processing*, vol. 81, no. 1, pp. 3–18, 2001.
- [63] T. Fortmann, Y. Bar-Shalom, and M. Scheffe, “Sonar tracking of multiple targets using joint probabilistic data association,” *IEEE Journal of Oceanic Engineering*, vol. 8, no. 3, pp. 173–184, Jul. 1983.
- [64] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart, “Distributed multirobot exploration and mapping,” *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1325–1339, July 2006.
- [65] Q. Gao, K. Blow, and D. Holding, “Simple algorithm for improving time synchronisation in wireless sensor networks,” *Electronics Letters*, vol. 40, no. 14, pp. 889–891, July 2004.
- [66] A. F. García-Fernández and J. Grajal, “Multitarget tracking using the joint multitrack probability density,” in *12th International Conference on Information Fusion*, July 2009, pp. 595–602.
- [67] —, “Asynchronous particle filter for tracking using non-synchronous sensor networks,” *Signal Processing*, vol. 91, no. 10, pp. 2304–2313, Oct. 2011.
- [68] —, “Sequential multiple target detection using particle filters,” in *IEEE Workshop on Statistical Signal Processing*, 2011, pp. 749–752.

- [69] A. F. García-Fernández, J. Grajal, and M. R. Morelande, “Two-layer particle filter for multiple target detection and tracking,” *submitted to IEEE Transactions on Aerospace and Electronic Systems*, 2011.
- [70] A. F. García-Fernández, J. Grajal, and O. A. Yeste-Ojeda, “Analysis of ISAR images of a helicopter by a facet model,” in *International Conference on Radar*, Sept. 2008, pp. 32–37.
- [71] A. F. García-Fernández, M. R. Morelande, and J. Grajal, “Multitarget simultaneous localization and mapping of a sensor network,” *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4544–4558, Oct. 2011.
- [72] —, “Nonlinear filtering update phase via the single point truncated unscented Kalman filter,” in *14th International Conference on Information Fusion*, 2011, pp. 17–24.
- [73] —, “Particle filter for extracting target label information when targets move in close proximity,” in *14th International Conference on Information Fusion*, 2011, pp. 795–802.
- [74] —, “Truncated unscented Kalman filtering,” *submitted to IEEE Transactions on Signal Processing*, 2011.
- [75] A. F. García-Fernández, O. A. Yeste-Ojeda, and J. Grajal, “Facet model of moving targets for ISAR imaging and radar back-scattering simulation,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 46, no. 3, pp. 1455–1467, July 2010.
- [76] S. J. Godsill, “On the relationship between Markov chain Monte Carlo methods for model uncertainty,” *Journal of Computational and Graphical Statistics*, vol. 10, no. 2, pp. 1–19, 2001.
- [77] G. H. Golub and C. F. Van Loan, *Matrix computations*. The Johns Hopkins University Press, 1996.
- [78] N. Gordon, D. Salmond, and A. Smith, “Novel approach to nonlinear/non-Gaussian Bayesian state estimation,” *IEE Proceedings-F on Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, Apr. 1993.
- [79] P. J. Green, “Reversible jump Markov chain Monte Carlo computation and Bayesian model determination,” *Biometrika*, vol. 82, no. 4, pp. 711–732, Dec. 1995.

-
- [80] —, “Trans-dimensional Markov chain Monte Carlo,” in *Highly Structured Stochastic Systems*, P. J. Green, N. L. Hjort, and S. Richardson, Eds. Oxford University Press, 2003.
- [81] G. Grisetti, C. Stachniss, and W. Burgard, “Improved techniques for grid mapping with Rao-Blackwellized particle filters,” *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 34–46, Feb. 2007.
- [82] J. A. Gubner, *Probability and random processes for electrical and computer engineers*. University Press, 2006.
- [83] M. Guerriero, L. Svensson, D. Svensson, and P. Willett, “Shooting two birds with two bullets: How to find minimum mean OSPA estimates,” in *13th Conference on Information Fusion*, July 2010, pp. 1–8.
- [84] F. Gustafsson, *Adaptive Filtering and Change Detection*. John Wiley & Sons, 2000.
- [85] —, “Particle filter theory and practice with positioning applications,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, no. 7, pp. 53–82, July 2010.
- [86] F. Gustafsson, S. Saha, and U. Orguner, “The benefits of down-sampling in the particle filter,” in *14th International Conference on Information Fusion*, 2011, pp. 277–282.
- [87] R. Handcock, D. Swain, G. Bishop-Hurley, K. Patison, T. Wark, P. Valencia, P. Corke, and C. O’Neill, “Monitoring animal behaviour and environmental interactions using wireless sensor networks, GPS collars and satellite remote sensing,” *Sensors*, vol. 9, pp. 3586–3603, 2009.
- [88] S. Haykin, *Adaptive Filter Theory*. Prentice-Hall, 2002.
- [89] A. O. Hero III, D. A. Castañón, D. Cochran, and K. Kastella, *Foundations and Applications of Sensor Management*. Springer, 2008.
- [90] J. Hershey and P. Olsen, “Approximating the Kullback Leibler divergence between Gaussian mixture models,” in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 4, April 2007, pp. 317–320.
- [91] M. Hürzeler, “Statistical methods for general state-space models,” Ph.D. dissertation, Swiss Federal Institute of Technology Zurich, 1998.

- [92] X.-L. Hu, T. B. Schön, and L. Ljung, “A general convergence result for particle filtering,” *IEEE Transactions on Signal Processing*, vol. 59, no. 7, pp. 3424–3429, July 2011.
- [93] X.-L. Hu, T. Schön, and L. Ljung, “A basic convergence result for particle filtering,” *IEEE Transactions on Signal Processing*, vol. 56, no. 4, pp. 1337–1348, April 2008.
- [94] C. Hue, J. Le Cadre, and P. Perez, “Tracking multiple objects with particle filtering,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 3, pp. 791–812, Jul 2002.
- [95] K. Ito and K. Xiong, “Gaussian filters for nonlinear filtering problems,” *IEEE Transactions on Automatic Control*, vol. 45, no. 5, pp. 910–927, May 2000.
- [96] A. Jasra, A. Doucet, D. A. Stephens, and C. C. Holmes, “Interacting sequential Monte Carlo samplers for trans-dimensional simulation,” *Computational Statistics & Data Analysis*, vol. 52, no. 4, pp. 1765–1791, 2008.
- [97] A. H. Jazwinski, *Stochastic Processes and Filtering Theory*. Academic Press, 1970.
- [98] A. M. Johansen and A. Doucet, “A note on auxiliary particle filters,” *Statistics & Probability Letters*, vol. 78, no. 12, pp. 1498–1504, 2008.
- [99] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, “A new method for the nonlinear transformation of means and covariances in filters and estimators,” *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, Mar. 2000.
- [100] S. J. Julier and J. K. Uhlmann, “Unscented filtering and nonlinear estimation,” *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004.
- [101] —, “Fusion of time delayed measurements with uncertain time delays,” in *Proceedings of the 2005 American Control Conference*, vol. 6, June 2005, pp. 4028–4033.
- [102] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Transactions of the ASME-Journal of Basic Engineering*, vol. 82 (Series D), pp. 35–45, 1960.
- [103] N. Kantas, S. S. Singh, and A. Doucet, “Distributed online self-localization and tracking in sensor networks,” in *5th International Symposium on Image and Signal Processing and Analysis ISPA*, Sept. 2007, pp. 498–503.

-
- [104] S. Kar and J. M. F. Moura, “Gossip and distributed Kalman filtering: Weak consensus under weak detectability,” *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1766–1784, April 2011.
 - [105] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, 1993.
 - [106] —, *Fundamentals of Statistical Signal Processing: Detection Theory*. Prentice-Hall, 1998.
 - [107] R. Kelley, *Iterative Methods for Optimization*. SIAM, 1999.
 - [108] C. Kim, R. Sakthivel, and W. K. Chung, “Unscented FastSLAM: A robust and efficient solution to the SLAM problem,” *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 808–820, Aug. 2008.
 - [109] J. Kim, J. Lee, E. Serpedin, and K. Qaraqe, “A robust estimation scheme for clock phase offsets in wireless sensor networks in the presence of non-Gaussian random delays,” *Signal Processing*, vol. 89, no. 6, pp. 1155–1161, June 2009.
 - [110] M. Klaas, N. de Freitas, and A. Doucet, “Toward practical n^2 Monte Carlo: The marginal particle filter,” in *Uncertainty in Artificial Intelligence*, 2005.
 - [111] L. Kleinrock, *Queueing Systems*. John Wiley & Sons, 1976.
 - [112] A. Kong, J. S. Liu, and W. H. Wong, “Sequential imputations and Bayesian missing data problems,” *Journal of the American Statistical Association*, vol. 89, no. 425, pp. 278–288, 1994.
 - [113] J. H. Kotecha and P. M. Djuric, “Gaussian particle filtering,” *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2592–2601, Oct. 2003.
 - [114] —, “Gaussian sum particle filtering,” *IEEE Transactions on Signal Processing*, vol. 51, no. 10, pp. 2602–2612, Oct. 2003.
 - [115] C. Kreucher, “An information-based approach to sensor resource allocation,” Ph.D. dissertation, University of Michigan, 2005.
 - [116] C. Kreucher, K. Kastella, and A. O. Hero III, “Multitarget tracking using the joint multitarget probability density,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 4, pp. 1396–1414, Oct. 2005.
 - [117] C. M. Kreucher, A. O. Hero III, K. D. Kastella, and M. R. Morelande, “An information-based approach to sensor management in large dynamic networks,” *Proceedings of the IEEE*, vol. 95, no. 5, pp. 978–999, May 2007.

- [118] C. M. Kreucher, M. R. Morelande, K. Kastella, and A. O. Hero III, "Joint multi-target particle filtering," in *Foundations and Applications of Sensor Management*, A. O. Hero III, D. A. Castañon, D. Cochran, and K. Kastella, Eds. Springer Science + Business Media, 2008.
- [119] N. Kwak, G.-W. Kim, and B.-H. Lee, "A new compensation technique based on analysis of resampling process in FastSLAM," *Robotica*, vol. 26, pp. 205–217, March 2008.
- [120] C. Kwok, D. Fox, and M. Meila, "Real-time particle filters," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 469–484, Mar. 2004.
- [121] I. Kyriakides, D. Morrell, and A. Papandreou-Suppappola, "Sequential Monte Carlo methods for tracking multiple targets with deterministic and stochastic constraints," *IEEE Transactions on Signal Processing*, vol. 56, no. 3, pp. 937–948, March 2008.
- [122] T. Lefebvre, H. Bruyninckx, and J. De Schuller, "Comment on "a new method for the nonlinear transformation of means and covariances in filters and estimators" [and authors' reply]," *IEEE Transactions on Automatic Control*, vol. 47, no. 8, pp. 1406–1409, Aug. 2002.
- [123] T. Lefebvre, H. Bruyninckx, and J. D. Schutter, "Kalman filters for nonlinear systems: a comparison of performance," *International Journal of Control*, vol. 77, no. 7, pp. 639–653, May 2004.
- [124] X. Lin, Y. Bar-Shalom, and T. Kirubarajan, "Multisensor multitarget bias estimation for general asynchronous sensors," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 3, pp. 899–921, July 2005.
- [125] B. Liu, F. Ren, J. Shen, and H. Chen, "Advanced self-correcting time synchronization in wireless sensor networks," *IEEE Communications Letters*, vol. 14, no. 4, pp. 309–311, April 2010.
- [126] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*. Springer, 2001.
- [127] M. B. Luca, S. Azou, G. Burel, and A. Serbanescu, "On exact Kalman filtering of polynomial systems," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 53, no. 6, pp. 1329–1340, June 2006.
- [128] C. Lundquist, L. Hammarstrand, and F. Gustafsson, "Road intensity based mapping using radar measurements with a probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 59, no. 4, pp. 1397–1408, April 2011.

-
- [129] W.-K. Ma, B.-N. Vo, S. Singh, and A. Baddeley, "Tracking an unknown time-varying number of speakers using TDOA measurements: a random finite set approach," *IEEE Transactions on Signal Processing*, vol. 54, no. 9, pp. 3291–3304, Sept. 2006.
- [130] R. P. S. Mahler, "Multitarget Bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1152–1178, Oct. 2003.
- [131] —, *Statistical Multisource-Multitarget Information Fusion*. Artech House, 2007.
- [132] R. Mahler, "Statistics 101" for multisensor, multitarget data fusion," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, no. 1, pp. 53–64, Jan. 2004.
- [133] R. Martínez-Cantín, N. de Freitas, and J. A. Castellanos, "Analysis of particle methods for simultaneous robot localization and mapping and a new algorithm: Marginal-SLAM," in *IEEE International Conference on Robotics and Automation*, April 2007, pp. 2415–2420.
- [134] M. Maskery and V. Krishnamurthy, "Network-enabled missile deflection: Games and correlation equilibrium," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 3, pp. 843–863, July 2007.
- [135] J. Míguez, "Analysis of parallelizable resampling algorithms for particle filtering," *Signal Processing*, vol. 87, no. 12, pp. 3155–3174, 2007, Special Section: Information Processing and Data Management in Wireless Sensor Networks.
- [136] R. Mittelman and E. Miller, "Nonlinear filtering using a new proposal distribution and the improved fast Gauss transform with tighter performance bounds," *IEEE Transactions on Signal Processing*, vol. 56, no. 12, pp. 5746–5757, Dec. 2008.
- [137] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, "FastSLAM: A factored solution to the simultaneous localization and mapping problem," in *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2002, pp. 593–598.
- [138] —, "FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *Proceedings of the International Conference on Artificial Intelligence (IJCAI)*, 2003, pp. 1151–1156.

- [139] M. R. Morelande, "Tracking multiple targets with a sensor network," in *9th International Conference on Information Fusion*, July 2006, pp. 1–7.
- [140] M. R. Morelande and A. F. García-Fernández, "Analysis of the Kalman filter for nonlinear measurements," *under preparation*, 2011.
- [141] M. R. Morelande, C. M. Kreucher, and K. Kastella, "A Bayesian approach to multiple target detection and tracking," *IEEE Transactions on Signal Processing*, vol. 55, no. 5, pp. 1589–1604, May. 2007.
- [142] M. R. Morelande and B. Moran, "Multiple target detection and tracking with a sensor network," in *10th International Conference on Information Fusion*, July 2007, pp. 1–8.
- [143] —, "An unscented transformation for conditionally linear models," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 3, April 2007, pp. 1417–1420.
- [144] M. R. Morelande and A. Zhang, "Uniform sampling for multiple target tracking," in *14th International Conference on Information Fusion*, 2011, pp. 283–289.
- [145] S. Mori and C.-Y. Chong, "Point process formalism for multiple target tracking," in *Proceedings of the Fifth International Conference on Information Fusion*, vol. 1, 2002, pp. 10–17.
- [146] S. Mori, C.-Y. Chong, E. Tse, and R. Wishner, "Tracking and classifying multiple targets without a priori identification," *IEEE Transactions on Automatic Control*, vol. 31, no. 5, pp. 401–409, May 1986.
- [147] J. Mullane, B.-N. Vo, M. D. Adams, and B.-T. Vo, "A random-finite-set approach to Bayesian SLAM," *IEEE Transactions on Robotics*, vol. 27, no. 2, pp. 268–282, April 2011.
- [148] J. Mullane, B.-N. Vo, and M. Adams, "Rao-Blackwellised PHD SLAM," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2010, pp. 5410–5416.
- [149] J. Mullane, B.-N. Vo, M. Adams, and W. Wijesoma, "A random set formulation for Bayesian SLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept. 2008, pp. 1043–1049.
- [150] D. Musicki, R. Evans, and S. Stankovic, "Integrated probabilistic data association," *IEEE Transactions on Automatic Control*, vol. 39, no. 6, pp. 1237–1241, Jun. 1994.

- [151] C. Musso, N. Oudjane, and F. Le Gland, “Improving regularised particle filters,” in *Sequential Monte Carlo Methods in Practice*, A. Doucet, N. de Freitas, and N. Gordon, Eds. Springer-Verlag, 2001.
- [152] B. Ng, L. Peshkin, and A. Pfeffer, “Factored particles for scalable monitoring,” in *Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence*, 2002, pp. 370–377.
- [153] N. Nikolaidis and I. Pitas, “Nonlinear processing and analysis of angular signals,” *IEEE Transactions on Signal Processing*, vol. 46, no. 12, pp. 3181–3194, Dec. 1998.
- [154] R. Niu and P. K. Varshney, “Performance analysis of distributed detection in a random sensor field,” *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 339–349, Jan. 2008.
- [155] —, “Distributed detection and fusion in a large wireless sensor network of random size,” *EURASIP Journal on Wireless Communications and Networking*, vol. 4, pp. 462–472, 2005.
- [156] R. Niu, P. K. Varshney, and Q. Chen, “Distributed detection in a large wireless sensor network,” *Information Fusion*, vol. 7, pp. 380–394, 2006.
- [157] P.-J. Nordlund and F. Gustafsson, “Marginalized particle filter for accurate and reliable terrain-aided navigation,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 45, no. 4, pp. 1385–1399, Oct. 2009.
- [158] S. Oh, S. Russell, and S. Sastry, “Markov chain Monte Carlo data association for multi-target tracking,” *IEEE Transactions on Automatic Control*, vol. 54, no. 3, pp. 481–497, March 2009.
- [159] B. N. Oreshkin, X. Liu, and M. J. Coates, “Efficient delay-tolerant particle filtering,” *IEEE Transactions on Signal Processing*, vol. 59, no. 7, pp. 3369–3381, July 2011.
- [160] D. Orlando, F. Ehlers, and G. Ricci, “A maximum likelihood tracker for multistatic sonars,” in *13th International Conference on Information Fusion*, July 2010, pp. 1–6.
- [161] M. Orton and W. Fitzgerald, “A Bayesian approach to tracking multiple targets using sensor arrays and particle filters,” *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 216–223, Feb. 2002.

- [162] M. Orton and A. Marrs, "Particle filters for tracking with out-of-sequence measurements," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 2, pp. 693–702, April 2005.
- [163] N. Oudjane and S. Rubenthaler, "Stability and uniform particle approximation of nonlinear filters in case of non ergodic signals." *Stochastic Analysis and Applications*, vol. 23, no. 3, pp. 421–448, 2005.
- [164] A. Papoulis, *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 1991.
- [165] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, Jun. 1999.
- [166] J. Predt, S. Kulkarni, and H. Poor, "Distributed learning in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 23, no. 4, pp. 56–69, July 2006.
- [167] A. Rafati, B. Moshiri, and J. Rezaei, "A new algorithm for general asynchronous sensor bias estimation in multisensor-multitarget systems," in *10th International Conference on Information Fusion*, July 2007, pp. 1–8.
- [168] B. Ristic and M. S. Arulampalam, "Tracking a manoeuvring target using angle-only measurements: algorithms and performance," *Signal Processing*, vol. 83, no. 6, pp. 1223–1238, June 2003.
- [169] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, 2004.
- [170] D. B. Rubin, "A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The SIR algorithm," *Journal of the American Statistical Association*, vol. 82, no. 398, pp. 543–546, 1987.
- [171] R. Y. Rubinstein and D. P. Kroese, *Simulation and the Monte Carlo method*. John Wiley & Sons, Inc., 2008.
- [172] A. Satish and R. Kashyap, "Multiple target tracking using maximum likelihood principle," *IEEE Transactions on Signal Processing*, vol. 43, no. 7, pp. 1677–1695, Jul. 1995.
- [173] V. Savic, A. Población, S. Zazo, and M. García, "Indoor positioning using nonparametric belief propagation based on spanning trees," *EURASIP Journal on Wireless Communications and Networking*, vol. 2010, p. 12, 2010.

- [174] L. Scharf, *Statistical Signal Processing: Detection, Estimation and Time Series Analysis*. Addison Wesley, 1990.
- [175] T. Schön, F. Gustafsson, and P.-J. Nordlund, “Marginalized particle filters for mixed linear/nonlinear state-space models,” *IEEE Transactions on Signal Processing*, vol. 53, no. 7, pp. 2279–2289, July 2005.
- [176] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, “A consistent metric for performance evaluation of multi-object filters,” *IEEE Transactions on Signal Processing*, vol. 56, no. 8, pp. 3447–3457, Aug. 2008.
- [177] F. Septier, J. Cornebise, S. Godsill, and Y. Delignon, “A comparative study of Monte-Carlo methods for multitarget tracking,” in *IEEE Signal Processing Workshop*, 2011, pp. 205–208.
- [178] X. Sheng and Y.-H. Hu, “Maximum likelihood multiple-source localization using acoustic energy measurements with wireless sensor networks,” *IEEE Transactions on Signal Processing*, vol. 53, no. 1, pp. 44–53, Jan. 2005.
- [179] D. Simon, *Optimal State Estimation, Kalman, H_∞ , and Nonlinear Approaches*. John Wiley & Sons, 2006.
- [180] D. Simon and D. L. Simon, “Constrained Kalman filtering via density function truncation for turbofan engine health estimation,” *International Journal of Systems Science*, vol. 41, no. 2, pp. 159–171, Feb. 2010.
- [181] V. Smidl and A. Quinn, “Variational Bayesian filtering,” *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp. 5020–5030, Oct. 2008.
- [182] L. Song and D. Hatzinakos, “A cross-layer architecture of wireless sensor networks for target tracking,” *IEEE/ACM Transactions on Networking*, vol. 15, no. 1, pp. 145–158, Feb. 2007.
- [183] S. Särkkä, A. Vehtari, and J. Lampinen, “Rao-Blackwellized particle filter for multiple target tracking,” *Information Fusion*, vol. 8, no. 1, pp. 2–15, Jan. 2007.
- [184] G. Strang, *Introduction to linear algebra*. Wellesley - Cambridge Press, 2003.
- [185] L. Svensson, D. Svensson, M. Guerriero, and P. Willett, “Set JPDA filter for multitarget tracking,” *IEEE Transactions on Signal Processing*, vol. 59, no. 10, pp. 4677–4691, Oct. 2011.

- [186] A. Swami, Q. Zhao, Y.-W. Hong, and L. Tong, *Wireless Sensor Networks. Signal Processing and Communications Perspectives*. John Wiley & Sons, 2007.
- [187] C. Taylor, A. Rahimi, J. Bachrach, H. Shrobe, and A. Grue, "Simultaneous localization, calibration, and tracking in an ad hoc sensor network," in *The Fifth International Conference on Information Processing in Sensor Networks, IPSN.*, 2006, pp. 27–33.
- [188] R. Tharmarasa, T. Kirubarajan, M. L. Hernandez, and A. Sinha, "PCRLB-based multisensor array management for multitarget tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 2, pp. 539–555, April 2007.
- [189] R. Tharmarasa, T. Kirubarajan, J. Peng, and T. Lang, "Optimization-based dynamic sensor management for distributed multitarget tracking," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 39, no. 5, pp. 534–546, Sept. 2009.
- [190] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramer-Rao bounds for discrete-time nonlinear filtering," *IEEE Transactions on Signal Processing*, vol. 46, no. 5, pp. 1386–1396, May 1998.
- [191] R. van der Merwe, "Sigma-point Kalman filters for probabilistic inference in dynamic state-space models," Ph.D. dissertation, Oregon Health and Science University, 2004.
- [192] H. L. Van Trees, *Detection, Estimation and Modulation Theory. Part I. Detection, Estimation and Linear Modulation Theory*. John Wiley & Sons, Inc., 2001.
- [193] P. K. Varshney, *Distributed Detection and Data Fusion*. Springer, 1997.
- [194] M. Vemula, J. Míguez, and A. Artés-Rodríguez, "A sequential Monte Carlo method for target tracking in an asynchronous wireless sensor network," in *4th Workshop on Positioning, Navigation and Communication*, March 2007, pp. 49–54.
- [195] T. Vercauteren, D. Guo, and X. Wang, "Joint multiple target tracking and classification in collaborative sensor networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 714–723, April 2005.

- [196] J. Vermaak, A. Doucet, and P. Perez, “Maintaining multimodality through mixture tracking,” in *Proceedings of the Ninth IEEE International Conference on Computer Vision, 2003*, vol. 2, Oct. 2003, pp. 1110–1116.
- [197] J. Vermaak, S. J. Godsill, and P. Perez, “Monte Carlo filtering for multi target tracking and data association,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 1, pp. 309–332, Jan. 2005.
- [198] J. Vermaak, S. Maskell, and M. Briers, “A unifying framework for multi-target tracking and existence,” in *8th International Conference on Information Fusion*, July 2005, pp. 250–258.
- [199] M. Vihola, “Rao-Blackwellised particle filtering in random set multitarget tracking,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 2, pp. 689–705, April 2007.
- [200] B.-N. Vo and W.-K. Ma, “The Gaussian mixture probability hypothesis density filter,” *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, Nov. 2006.
- [201] B.-N. Vo, B.-T. Vo, N.-T. Pham, and D. Suter, “Joint detection and estimation of multiple objects from image observations,” *IEEE Transactions on Signal Processing*, vol. 58, no. 10, pp. 5129–5141, Oct. 2010.
- [202] M. Šimandl, J. Královec, and T. Söderström, “Advanced point-mass method for nonlinear state estimation,” *Automatica*, vol. 42, pp. 1133–1145, 2006.
- [203] E. A. Wan and R. van der Merwe, “The unscented Kalman filter,” in *Kalman Filtering and Neural Networks*, S. Haykin, Ed. John Wiley & Sons, Inc., 2001.
- [204] G. Werner-Allen, K. Lorincz, M. Ruiz, O. Marcillo, J. Johnson, J. Lees, and M. Welsh, “Deploying a wireless sensor network on an active volcano,” *IEEE Internet Computing*, vol. 10, no. 2, pp. 18–25, March-April 2006.
- [205] Y. Wu, D. Hu, M. Wu, and X. Hu, “A numerical-integration perspective on Gaussian filters,” *IEEE Transactions on Signal Processing*, vol. 54, no. 8, pp. 2910–2921, Aug. 2006.
- [206] —, “Unscented Kalman filtering for additive noise case: augmented versus nonaugmented,” *IEEE Signal Processing Letters*, vol. 12, no. 5, pp. 357–360, May 2005.

BIBLIOGRAPHY

- [207] C. Yang, R. Duraiswami, N. Gumerov, and L. Davis, “Improved fast Gauss transform and efficient kernel density estimation,” in *Proceedings of the Ninth IEEE International Conference on Computer Vision*, Oct. 2003, pp. 664–671.
- [208] C.-L. Yang, S. Bagchi, and W. Chappell, “Topology insensitive location determination using independent estimates through semi-directional antennas,” *IEEE Transactions on Antennas and Propagation*, vol. 54, no. 11, pp. 3458–3472, Nov. 2006.
- [209] J. Yick, B. Mukherjee, and D. Ghosal, “Wireless sensor network survey,” *Computer Networks*, vol. 52, pp. 2292–2330, Aug. 2008.
- [210] F. Zhao, J. Shin, and J. Reich, “Information-driven dynamic sensor collaboration,” *IEEE Signal Processing Magazine*, vol. 19, no. 2, pp. 61–72, Mar 2002.
- [211] T. Zhao and A. Nehorai, “Distributed sequential Bayesian estimation of a diffusive source in wireless sensor networks,” *IEEE Transactions on Signal Processing*, vol. 55, no. 4, pp. 1511–1524, April 2007.